UNIVERSITY OF CALIFORNIA
RIVERSIDE

Robust Online Learning Enabled by Information Theory

A Dissertation submitted in partial satisfaction
of the requirements for the degree of

Doctor of Philosophy

in

Electrical Engineering

by

Sajjad Bahrami

June 2021

Dissertation Committee:

Professor Ertem Tuncel, Chairperson
Professor Ilya Dumer
Professor Yingbo Hua

Copyright by
Sajjad Bahrami
2021

The Dissertation of Sajjad Bahrami is approved:

_____

_____

_____

Committee Chairperson

University of California, Riverside

Acknowledgments

I would like to express my sincere gratitude to everyone who made my Ph.D. graduation happen. First of all, I should say from the bottom of my heart that how much I am grateful for working under supervision of Dr. Ertem Tuncel. I will forever be thankful to him for the time he spent in helping me and for his great insight into research, immense knowledge, consistent support, invaluable guidance, encouragement, and especially his friendship and empathy during my Ph.D. As we all were working from home due to the COVID-19 pandemic, I should also extend my heartfelt thanks to Dr. Tuncel's family for their patience during our discussions on my research. It is a great privilege and honor to have known Dr. Tuncel and I am extremely grateful for what he has offered me.

I would also like to thank my dissertation committee members; professors Ilya Dumer and Yingbo Hua, and my oral qualifying examination committee members; professors Jay A. Farrell and Stefano Lonardi for their valuable feedback on my research.

My life at Riverside was a wonderful time where I had the chance to know and spend time with many talented and supportive friends. I would like to say thank to all of them, especially, Amin (Baniasadi), Mohammad, Amin (Safdari), Hadi, and Reza. My Special thanks goes to my lovely partner Hoda for her continuing support.

And finally, my deepest appreciation is dedicated to my beloved parents and brother for their endless love and kind prayers, and all sacrifices that they have made for me throughout my life. I think of you every single day and feel you right next to me regardless of how far we are and how long we were not able to visit each other. I love you so much and I am always blessed to have you in my life.

The content of this thesis is a reprint of the following publications:

1. Sajjad Bahrami and Ertem Tuncel. "A New Approach to Online Regression Based on

Maximum Correntropy Criterion." 2019 IEEE 29th International Workshop on Machine Learning for Signal Processing (MLSP). IEEE, 2019.

2. Sajjad Bahrami and Ertem Tuncel. "Mitigating Outlier Effect in Online Regression: An Efficient Usage of Error Correntropy Criterion." 2020 International Joint Conference on Neural Networks (IJCNN). IEEE, 2020.

3. Sajjad Bahrami and Ertem Tuncel. "An Efficient Running Quantile Estimation Technique alongside Correntropy for Outlier Rejection in Online Regression." 2020 IEEE International Symposium on Information Theory (ISIT). IEEE, 2020.

4. Sajjad Bahrami and Ertem Tuncel. "Challenging the Deployment of Fiducial Points in Minimum Error Entropy." submitted to 2021 IEEE Information Theory Workshop (ITW).

5. Sajjad Bahrami and Ertem Tuncel. "Trimmed Minimum Error Entropy for Robust Online Regression." submitted to IEEE Transactions on Signal Processing.

To my lovely parents and my brother Saeed.

ABSTRACT OF THE DISSERTATION

Robust Online Learning Enabled by Information Theory

by

Sajjad Bahrami

Doctor of Philosophy, Graduate Program in Electrical Engineering
University of California, Riverside, June 2021
Professor Ertem Tuncel, Chairperson

In this thesis, we incorporate information theory into statistical signal processing and machine learning in order to achieve robust learning in presence of outliers (or generally in environments with non-Gaussian structure). In other words, by incorporating information theory especially when structure of environment is non-Gaussian, the efficiency of information extraction from data and consequently precision of the learning are increased. Note that, although the well-known central limit theorem creates the expectations that we should see the Gaussian distribution everywhere in the real world, this is not necessarily true. Indeed central limit theorem is concluded under some assumptions that do not hold in many real scenarios. For instance, in many real scenarios heavy-tailed distributions arise which result in outliers. The problem of learning a system affected by these types of distributions is of great importance inasmuch as conventional learning approaches that are mostly based on Gaussian assumption are not effective anymore. In this thesis, we address this problem by means of information theory.

Our work finds broad applications from communication channel estimation, adaptive equalization, adaptive echo noise cancellation, system identification, underwater and satellite communications, blind source separation to physics, biology, computer science, the social sciences, and beyond where real-world environments may change in time and are conducive to

non-Gaussian probability density functions. In such environments higher order statistics of data is needed, therefore conventional methodologies are not efficient anymore.

Specifically, the problem of online linear regression (or interchangeably linear adaptive filtering) in environments corrupted by non-Gaussian noise is addressed in this thesis. In such environments, the error between system outputs and labels (or desired responses) does not follow a Gaussian distribution and there might exist abnormally large error samples (or outliers). The main challenge we face here is how to keep our supervised learning problem least affected by these unwanted and misleading outliers. Information theory helps us with correntropy and entropy as two robust information-theoretic criteria. Error correntropy is a similarity measure between system output and label and consequently should be maximized while error entropy denotes the uncertainty about a system and should be minimized. Although each of these two criteria has an advantage over the other one, both of them show superior performance compared to conventional mean square error in aforementioned non-Gaussian environments.

In this thesis, we discuss the shortcomings of these algorithms and compare them. Moreover, we improve the algorithms based on error correntropy and error entropy by excluding major outliers from the learning process after detecting them in each iteration. Note that major outliers may occur frequently due to the nature of non-Gaussian environments and the system parameters should not be updated based on them. We use a filter to detect major outliers whose width is updated in each learning iteration based on running quartiles of the error samples. More precisely, quartiles of a data set are robust quantities of data against outliers and we use them to define boundaries by which we determine if an error sample is an outlier or not. These quartiles are functions of time and should be updated once a new error sample is available. This means that we need to deal with a running quartile problem. We propose an efficient technique for running quantile estimation based on the non-uniform quantization of error samples. This method does not need to store and sort all error samples in each iteration. Using this method alongside

algorithms based on correntropy and entropy, we reject major outliers and achieve improved convergence speed and/or steady-state misalignment in the learning curves of both algorithms.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Nowadays, we have access to a large amount of data thanks to computers and the Web. The main challenge in processing of data is how to extract as much information as possible from it. In order to deal with this huge amount of data effectively, we need to continuously adapt the existing techniques of data processing to the real-world problems where non-Gaussian probability density functions (PDFs) are very likely to be observed [1].

Although introductory probability and statistics gives the impression that Gaussian distribution should be observed everywhere as a result of central limit theorem (CLT), observing non-Gaussian PDFs in real-world signal processing is not only unsurprising but expected. The reason for that is the existence of a generalized and more comprehensive version of CLT. Moreover, when random variables are combined in other ways beyond summation, non-Gaussian PDFs are more expected to emerge [2]. These PDFs might create major outliers and might even be time-variant [1]. As an example of existence of non-Gaussian PDFs in practice we can name Pareto distribution which is a heavy-tailed distribution and has been observed in many applications from computer science to physics, biology and beyond [2]. Another example is existence of non-Gaussian PDFs in mobile massive multiple input multiple output (MIMO)

visible light communication (VLC) [3]. VLC is a new low-cost tool for 5th generation and beyond communication systems in which throughput can be increased by using a huge amount of transmit light emitting diodes (LEDs) and photo-detectors [4]. Despite the potential of massive MIMO VLC for high speed communication, PDF of the additive interference is non-Gaussian for mobile massive MIMO VLC which means that we need more efficient methods than conventional minimum mean squared error (MMSE) for channel estimation. Underwater acoustic communication is another real scenario where a non-Gaussian noise called impulsive noise is present due to the human activities (such as oil and gas exploration, shipping, and sonar-related applications) and natural sources (such as water agitation, earthquakes, and bio-acoustic sounds) [5–9]. We can also point to presence of non-Gaussian noise in power line communications (PLC). PLC can be used in many smart grid applications because of the availability of the power line infrastructure which results in decrease in the costs. A very important challenge in PLC is how to overcome additive noise which is impulsive? This noise can be generated externally due to conduction and radiation and/or originates from electrical devices which are linked to the power lines [10, 11]. We can also name presence of the impulsive noise in image or audio signals [12] and many other real-world examples in which the noise is not Gaussian.

Above examples show how broadly non-Gaussian PDFs appear in various applications and arise the question ”can we use conventional learning cost functions for these scenarios?” In order to train a system in a supervised learning problem usually second-order moment of the error between labels and system outputs is utilized as the cost function [1]. However, its shortcoming is the fact that learning precision may degrade substantially if the environment is being corrupted by non-Gaussian noise like the above real-world examples [1, 13]. Although some other cost functions have been proposed to overcome this issue such as least mean fourth (LMF) [14] and sign algorithm [15] where the fourth-order moment of the error and absolute value of the error

[1] Note that label is sometimes called desired response and we use them interchangeably throughout this thesis.

are used, respectively, they have some weaknesses and do not capture the whole statistics of the error. Information-theoretic learning proposes entropy and correntropy as alternatives to conventional cost functions which are simple and also more comprehensive descriptors of the error [16, 17]. In fact, they contain higher-order statistics of the error.

Error correntropy is a similarity measure between labels and system outputs and therefore should be maximized during the learning process. This similarity measure is defined based on a kernel function (usually including but not limited to Gaussian kernel [18, 19]). Some applications of correntropy in image processing and wireless communication can be found in recent works such as [20], [21] and [22].

On the other hand, error entropy denotes the uncertainty about a system that should be minimized. In other words, ideally we want error PDF to be a delta function at $e = 0$ which is achieved by error entropy minimization. Renyi's entropy of order r is a family of entropies for which Shannon entropy is the limiting case of $r \to 1$ [23] and its quadratic version, i.e., $r = 2$ is used in the context of information-theoretic learning because of its estimation simplicity from error samples [1]. Some applications of error entropy minimization can be found in recent works such as [3, 24] and [25] in the context of wireless communication and power systems.

The overall performance of the algorithms based on maximum corrrentropy criterion (MCC) and minimum error entropy (MEE) can even be improved by doing some manipulations as shown in literature such as [26–29], and [30].

1.1 Contributions of the Dissertation

This dissertation addresses the supervised learning problem of online linear regression (or interchangeably linear adaptive filtering) in environments corrupted by non-Gaussian noise. This problem is illustrated in Figure 1.1. First, note that throughout the thesis random variables,

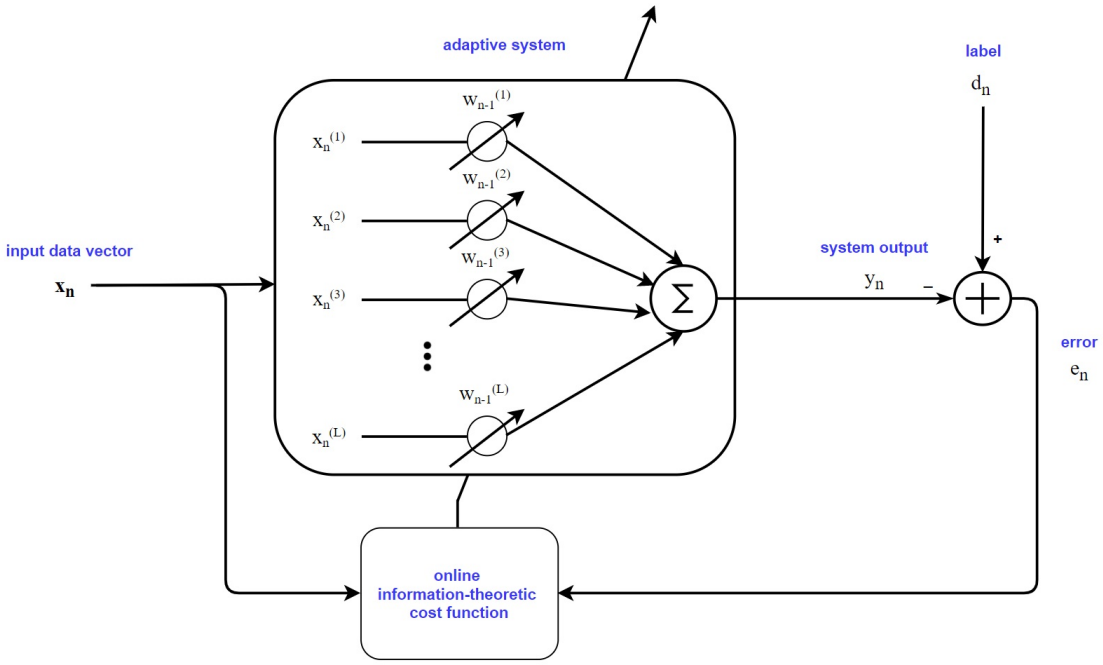


Figure 1.1: Online linear regression based on information-theoretic cost functions.

their realization and vectors are denoted by uppercase letters, lowercase letters and boldface letters, respectively. Moreover, $\| \, . \, \|$ denotes 2 norm and we use the terms adaptive filtering and online regression interchangeably. Now lets get back to Figure 1.1. The goal is to find the parameters of a linear system $y_n = \mathbf{x}_n^T \mathbf{w}_{n-1}$ in which $\mathbf{x}_n = \left[x_n^{(1)}, x_n^{(2)}, \cdots, x_n^{(L)}\right]^T$ is the input vector at time instant n, $\mathbf{w}_{n-1} = \left[w_{n-1}^{(1)}, w_{n-1}^{(2)}, \cdots, w_{n-1}^{(L)}\right]^T$ is a vector denoting system parameters estimated at time instant $n-1$, L indicates the length of the parameter vector and y_n denotes system output estimated at time instant n. Error samples at time n are obtained as $e_i = d_i - \mathbf{x}_i^T \mathbf{w}_{n-1}$ where d_i denotes label at time instant i. We want to concentrate error samples around $e = 0$ with time. More precisely, the error between system outputs and labels in such environments does not follow a Gaussian distribution and there might exist abnormally large error samples (or outliers). The main challenge we face here is how to keep this supervised learning problem least affected by these unwanted and misleading outliers. Information-theoretic learning helps us with correntropy and entropy as two robust criteria. In this thesis, we discuss the shortcomings of algorithms based on correntropy and entropy, and compare them. Moreover, we improve

these algorithms mainly by excluding major error samples (or outliers) from the learning process after detecting them in each iteration. Note that major outliers may occur frequently due to the nature of non-Gaussian environments and the system parameters should not be updated based on them. We use a filter to detect major outliers whose width is updated in each learning iteration based on running quartiles of the error samples. More precisely, quartiles of a data set are robust quantities of data against outliers and we use them to define boundaries by which we determine if an error sample is an outlier or not. These quartiles are functions of time n and should be updated once a new error sample is available. This means that we need to deal with a running quartile problem. We propose an efficient technique for running quartile estimation based on the non-uniform quantization of error samples. This method does not need to store and sort all error samples in each iteration. Using this method alongside algorithms based on correntropy and entropy, we reject major outliers and achieve improved convergence speed and/or steady-state misalignment in the learning curves of both algorithms. We state our contributions in more detail in the following.

1.1.1 A Hybrid Approach to Online Regression Based on Maximum Correntropy Criterion

The problem of online regression in the presence of non-Gaussian noise is addressed. A new hybrid algorithm based on recursive maximum correntropy criterion (RMCC) and gradient-based MCC is proposed, and demonstrated to outperform previous works in terms of both convergence speed and steady-state misalignment. At the same time, the proposed algorithm benefits from lower computational complexity compared to some of these algorithms.

1.1.2 Mitigating Outlier Effect in Online Regression: An Efficient Usage of Error Correntropy Criterion

A modified version of MCC is proposed with application in online linear regression (or linear adaptive filtering) again. Indeed, we consider the existing adaptive maximum correntropy criterion algorithm (known as AMCC) in which a free parameter called kernel bandwidth is also adapted as an extra step to learn a linear system more efficiently . We improve AMCC by simply eliminating major outliers during learning process. This elimination leads to better steady-state performance compared to previous algorithms.

1.1.3 An Efficient Running Quartile Estimation Technique alongside Correntropy for Outlier Rejection in Online Regression

We propose an efficient technique to detect and then exclude abnormally large error samples from the learning process in an online linear regression (or linear adaptive filtering) in the presence of non-Gaussian noise. This technique is indeed a running quartile estimation based on quantization of error samples alongside which correntropy leads to lower steady-state misalignment compared to previous algorithms.

1.1.4 Challenging the Deployment of Fiducial Points in Minimum Error Entropy

The well-known algorithm for robust adaptive learning or online linear regression called minimum error entropy with fiducial points (MEEF) is challenged. In MEE error entropy is minimized to extract as much information as possible about the data generating system. However, this minimum entropy can also occur for other error PDFs not located at the origin inasmuch as entropy is shift-invariant. In these cases, an undesired estimate of the system parameters is obtained. Therefore, an extra step needs to be taken to concentrate error samples around the

origin. The most celebrated approach towards that end is MEEF, in which some external and artificial zero error samples, called fiducial points (not generated by the underlying system), are added to the cost function as the reference points to force actual error samples to get concentrated around them. Using these fiducial points translates MEEF into a weighted combination of MEE and MCC. We show that incorporating these fiducial points into MEE can even degrade the steady-state misalignment and/or convergence speed.

1.1.5 Trimmed Minimum Error Entropy for Robust Online Regression

Again, online linear regression (or linear adaptive filtering) in environments corrupted by non-Gaussian noise (especially those with heavier tail than that of Gaussian) is addressed. As stated earlier, in such environments the error between the system output and the label also does not follow a Gaussian distribution and there might exist abnormally large error samples (or outliers) which mislead the learning process. The main challenge is how to keep the supervised learning problem least affected by these unwanted and misleading outliers. MEE as an information-theoretic algorithm based on Renyi's entropy has been employed to take on this issue. However, this minimization might not result in a desired estimator inasmuch as entropy is shift-invariant, i.e., by minimizing the error entropy, error samples may not be necessarily concentrated around zero. We use our proposed quantization technique by which not only aforementioned need of setting errors around the origin in MEE is addressed, but also major outliers are rejected from MEE-based learning and MEE performance is improved from convergence rate, steady-state misalignment, and testing error points of view.

1.2 Outline

We present some preliminaries about information-theoretic cost functions and some related works in Chapter 2. Chapter 3 is devoted to our hybrid algorithm based on MCC for online linear regression. Chapter 4 presents our modification for AMCC algorithm. In Chapter 5 we propose our running quartile estimation technique for outlier detection in an online linear regression. In Chapters 6 and 7 we challenge the deployments of fiducial points in MEE and propose an alternative to MEEF, respectively. Finally, in Chapter 8 we conclude this thesis and highlight the potential future work.

Chapter 2

Preliminaries

2.1 Introduction

Online regression (or interchangeably adaptive filtering) is a strong tool for many signal processing and machine learning applications. This is because of its satisfactory performance despite the possible change of statistics with time in an unknown environment. In an online regression (or adaptive filtering), we adjust some unknown parameters based on the estimation error between desired response and output of the adaptive system. According to the way that we obtain the desired response we can classify the applications of online regression and adaptive filtering into four categories, i.e., prediction, interference cancellation, inverse modeling, and identification [31]. We can name many different applications under these categories in many various fields such as seismology, financial engineering, communications, control, biomedical engineering, etc. For example, the spectrum analysis is an application under the prediction class in which the power spectrum of a signal is estimated. We can also name beamforming and noise cancellation as two applications under interference cancellation in which the goals are to mitigate the harmful effect of the interfering signals on the signal of interest, and to increase the signal to noise ratio (SNR) by eliminating the noise from a received signal, respectively. Regarding

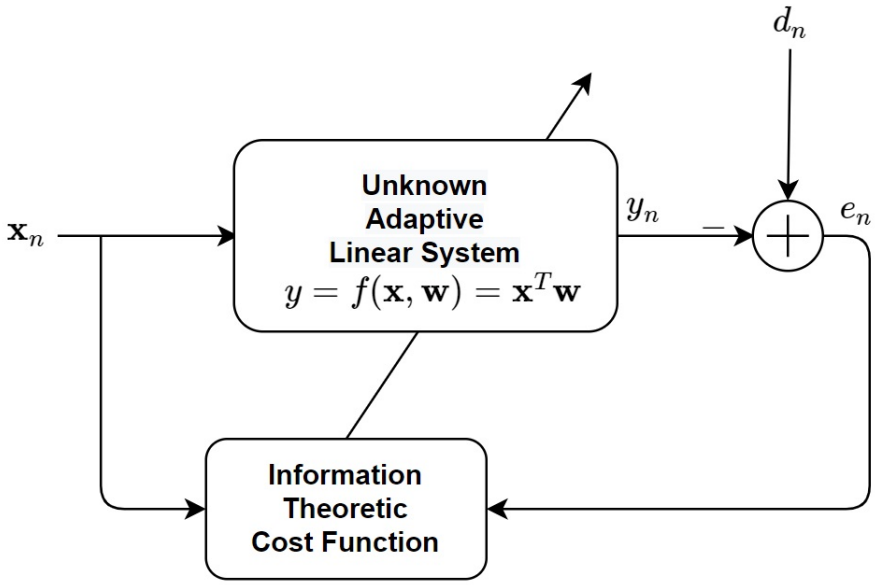


Figure 2.1: Online linear regression.

an example for application under inverse modeling we can mention channel equalization in which our goal is to manipulate the channel output such that the combination of the channel and equalizer becomes an approximately ideal communications link. And as for the identification we can classify system identification under this category of applications in which the goal is to approximate an unknown system with an adaptive system.

Throughout this thesis we consider an online *linear* regression or equivalently *linear* adaptive filtering illustrated in Figure 2.1. Linear adaptive filtering also finds many applications such as channel estimation, channel equalization, active noise cancellation and so on [1, 31, 32]. The goal of the linear adaptive filtering at each time instant n is to update its estimate of system parameter vector and obtain $\mathbf{w}_n$ by observing a new sample pair $(\mathbf{x}_n, d_n)$. $\mathbf{x}_n = \left[x_n, x_{n-1}, \cdots, x_{n-L+1}\right]^T$ and d_n are the input vector and label, respectively at time instant n, and L denotes the number of elements in system parameter vector $\mathbf{w}$. Moreover, at this time instant, output of the system and error sample are calculated as $y_n = \mathbf{x}_n^T\mathbf{w}_{n-1}$ and $e_n = d_n - y_n$, respectively. In other words, the adaptive learning aims to accumulate error samples around $e = 0$ with time. As seen in Figure 2.1 and discussed in Chapter 1 we optimize an information-theoretic cost

function not the conventional mean squared error (MSE). It is worth reminding that in many real-world signal processing and machine learning applications, especially nonlinear topologies, we encounter non-Gaussian probability density functions (PDFs), e.g., underwater communications in which the Gaussian assumption cannot be made anymore due to the existence of impulsive noise. In such realistic scenarios, the noise might have a heavy-tailed distribution, there might exist severe outlier noise, and the error PDF might even change in time and consequently we need higher order statistics of error not just its variance which is the case for the MSE cost function. Therefore, information-theoretic cost functions are used as powerful alternatives to the MSE.

In the following two important information-theoretic cost functions for online learning and adaptive filtering, i.e., correntropy and quadratic Renyi's entropy are described in more details.

2.2 Correntropy and Quadratic Renyi's Entropy

In any adaptive learning structure like Figure 2.1 we need to optimize a cost function. Although even up to now, the most commonly used cost functions are based on the moments of the data [33], e.g., variance, skewness, and kurtosis, which are the 2nd, 3rd, and 4th central moments, respectively, when the error is non-Gaussian, these cost functions are not reliable and we need to take into account higher order statistics of data as well [1]. For instance, a very well-known cost function is MSE which is defined as follows:

$$J_{MSE}(E) = \mathbf{E}\{E^2\},$$

and E denotes the error between the desired response and system output. MSE is minimized in order to learn an adaptive system. As seen, this cost function is nothing but variance of the error. Obviously, if error distribution follows a Gaussian behaviour then minimizing the

variance of the error in the vicinity of $e = 0$ with time will result in a satisfactory estimate, however error distribution in many real-world scenarios is not Gaussian. Indeed, variance is not a comprehensive descriptor of the error statistics in these cases and we need higher order moments of the error as well. Therefore, we need to look for other general and robust descriptors of the data statistics in order to improve the performance of the learning algorithm.

In recent years, two effective cost functions, namely correntropy and quadratic Renyi's entropy, have been employed by information-theoretic learning in non-Gaussian environments as superior alternatives to the famous and most commonly used cost function, i.e., mean square error (MSE) [16, 17]. The relation between algorithms based on correntropy and quadratic Renyi's entropy has been investigated in [34]. In the literature, entropy and correntropy are sometimes interpreted as counterparts of variance and correlation, respectively [1]. Both correntropy and quadratic Renyi's entropy involve higher-order data statistics and therefore they are expected to outperform MSE, which only contains second-order moment.

2.2.1 Correntropy

Error correntropy criterion (ECC) is a local similarity measure between any two random variables D and Y, or specifically in our case a similarity measure between system output and desired response that should be maximized. It is defined as the expectation of a kernel function of error over error distribution (or joint distribution of D and Y), i.e.,

$$v(D,Y) = E_{D,Y}\{G_\sigma(D-Y)\} = E_E\{G_\sigma(E)\} = v(E), \tag{2.1}$$

in which E denotes the error $D-Y$, and $G_\sigma(.)$ is a smooth kernel function with σ as its kernel bandwidth (which affects the shape of the kernel function). Although many kernel functions can be used [18, 19], Gaussian kernel is the most popular choice used because of its nice properties.

Gaussian kernel is defined as follows:

$$G_\sigma(D-Y) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp(-\frac{\| D-Y \|^2}{2\sigma^2}). \tag{2.2}$$

Note that ECC is called a local cost function inasmuch as it mostly focuses on the error observations that are within the kernel bandwidth σ. This can weaken the performance of ECC if most of the error observations are outside of this range because the learning process is not sensitive to them anymore. As we discussed earlier, although when environment is badly affected with Gaussian noise MSE gives the optimum solution [35] we need higher-order statistics in non-Gaussian environments [1] and this is the key superiority of correntropy over MSE. Indeed, if we calculate Taylor expansion of Gaussian kernel in (2.1) we observe that correntropy contains even-order moments of the error (not only second-order moment as in MSE [36]), so we can rewrite (2.1) as follows:

$$v(D,Y) = \frac{1}{\sqrt{2\pi}\sigma} \sum_{n=0}^{\infty} \frac{(-1)^n}{2^n \sigma^{2n} n!} E_{D,Y}\left\{(D-Y)^{2n}\right\}.$$

2.2.2 Quadratic Renyi's Entropy

We cannot simply provide a particular definition for information inasmuch as information is a deep concept. However, error entropy criterion (EEC) denotes the uncertainty about a system that can be minimized to extract as much information as possible about the data generating system. The logic behind entropy minimization is to bring error samples closer to each other in with time, more precisely, we strive to ideally set the distribution of the error as an impulse at the origin. In this thesis, we consider Alfred Renyi's definition of entropy which is a generalization of Shannon entropy [1, 23, 37]. Indeed Renyi's entropy of order r is a parametric family of entropies with parameter r where Shannon entropy is the limiting case of $r \to 1$. Renyi's entropy

of order r for a random variable X is defined as follows:

$$H_r(X) = \frac{1}{1-r} \log \int f^r(x) dx,$$

where $f(x)$ is the PDF of random variable X. Throughout the thesis we consider $r = 2$ which results in the following quadratic Renyi's entropy:

$$H_2(X) = -\log \int f^2(x) dx = -\log \mathbf{E}\{f(X)\}, \tag{2.3}$$

where $\mathbf{E}\{f(X)\} \triangleq I_2(X)$ is sometimes called information potential. It is worth noting that in the definition of information potential every sample from distribution $f(X)$ is given a weight by PDF itself. Moreover, estimator of the information potential is fairly straightforward [1]. These facts make the quadratic Renyi's entropy a very powerful cost function especially in presence of non-Gaussian noise. As for the higher-order statistics, EEC contains all higher-order moments of the PDF (if they exist) hence we can consider it as a global descriptor of the PDF. This is shown by using Taylor expansion of the PDF $f(x)$ as follows:

$$\begin{aligned} & f(x) = f(0) + f^{(1)}(0)x + \frac{f^{(2)}(0)}{2!}x^2 + \frac{f^{(3)}(0)}{3!}x^3 + \cdots \\ \Rightarrow \quad & I_2(X) = \mathbf{E}\{f(X)\} = f(0) + f^{(1)}(0)\mathbf{E}\{X\} + \frac{f^{(2)}(0)}{2!}\mathbf{E}\{X^2\} + \frac{f^{(3)}(0)}{3!}\mathbf{E}\{X^3\} + \cdots \end{aligned}$$

where $f^{(i)}(0)$ is the ith derivative of PDF at $x = 0$. Note that PDFs in practice are usually smooth and consequently they are continuously differentiable.

We need PDF of error between desired response and system output to calculate error correntropy and quadratic Renyi's error entropy as seen in (2.1) and (2.3). Nevertheless, statistics of the error is unknown in general. Moreover, PDF of error evolves with time during the online (or adaptive) learning process. In other words, we only have error samples (lets say a window of

N error samples) not error statistics, therefore we need to estimate the error PDF by using some non-parametric PDF estimation method.

2.2.3 Parzen Non-parametric PDF estimation

If we have a set of samples from an unknown PDF and aim to estimate this PDF, there are two different methods for this purpose, i.e., parametric and non-parametric PDF estimation. In parametric PDF estimation, we assume that the unknown PDF belongs to a known parametric family of distributions while in non-parametric method we make no assumption on the family of the PDF. In the online and adaptive learning problem that we focus on in this thesis, we need error PDF where this PDF in unknown and therefore, we need to estimate it from error samples based on a non-parametric method. Parzen PDF estimation, as a non-parametric method [38, 39], helps with quadratic Renyi's entropy estimation in MEE and also makes maximizing correntropy as a similarity measure meaningful in MCC. Indeed, having past N error samples, we use Parzen non-parametric PDF estimation throughout the thesis to estimate error PDF $p(e)$ at time instant n as follows:

$$p_E(e) \approx \frac{1}{N} \sum_{i=0}^{N-1} G_\sigma(e - e_{n-i}) = \hat{p}_E^{(n)}(e), \tag{2.4}$$

where $G_\sigma(.)$ is the Gaussian kernel with kernel bandwidth σ. As seen in above relation, Parzen non-parametric estimator uses a window to weight different available samples in order to determine $p_E(e)$ at a specific e. In other words, it provides a smooth estimate of error PDF based on Gaussian kernel which is indeed the sum of kernel functions centered at the error samples. Note that we use Gaussian kernel because the related PDF estimator is simple, continuous and differentiable, however, in general we could use other kernel functions that satisfy some specific conditions [40]. It is worth mentioning that kernel bandwidth σ affects the PDF estimator and if

we have a very large number of samples from the underlying PDF, then $\hat{p}_E^{(n)}(e)$ will converge to the unknown PDF $p_E(e)$ as σ slowly approaches 0 with n. However, in practice there is a compromise between bias and variance of this estimator [38]. Generally speaking, estimating the kernel bandwidth σ is not an easy task to do, and although many previous researches have striven to provide a kernel bandwidth with satisfactory performance, there is no single best kernel bandwidth selection technique yet. In fact, all of these methods for estimating σ can be considered optimal based on their related criteria and most of them are data-driven [39]. Some of these methods can be found in [39–54].

2.2.4 Relation Between Parzen PDF Estimation and ECC or EEC

As discussed earlier, we only have error samples but not the error distribution, hence we are not able to calculate error correntropy and quadratic Renyi's error entropy exactly. Nevertheless, we can estimate these cost functions at each time instant. Remember that our problem is an online learning problem in which we need to update our estimate of the unknown parameters with time, therefore we encounter online (or stochastic) cost functions. In other words, we need to estimate cost function at each time instant and then use it to update our estimate of the unknown parameters. In the following we see how estimate of the error correntropy is related to error PDF by (2.4) while quadratic Renyi's error entropy directly uses (2.4) for entropy estimation.

Consider again error correntropy definition (2.1). In this definition $G_\sigma(.)$ is the same Gaussian kernel used in (2.4) for error PDF estimation. We use sample mean approximation of correntropy (2.1) at time instant n from data samples $\{d_{n-i}, y_{n-i}\}$ (or equivalently $\{e_{n-i}\}$), $i =$

$0, \cdots, N-1$, as follows:

$$\hat{v}^{(n)}(D,Y) = \frac{1}{N}\sum_{i=0}^{N-1} G_\sigma(d_{n-i} - y_{n-i}) = \frac{1}{N}\sum_{i=0}^{N-1} G_\sigma(e_{n-i}), \tag{2.5}$$

in which D and Y are two random variables denoting desired response and system output, respectively, and random variable $E = D - Y$ denotes the error. The above relation is indeed the online cost function which is optimized in MCC using some technique, for instance stochastic gradient ascent, and calculating this cost function has a computational complexity of $O(N)$ in each iteration.

Obviously, if we use the kernel function in Parzen error PDF estimation the same as the one we use for correntropy estimation we see that correntropy estimation is nothing but estimation of error PDF at zero. In other words, $\hat{v}^{(n)}(D,Y) = \hat{p}_E^{(n)}(0)$ is concluded from (2.4) and (2.5) which means that if we maximize the estimate of correntropy we indeed maximize the estimate of error PDF at 0. This is the reason that maximum correntropy criterion is meaningful. The related algorithm is called maximum correntropy criterion. In addition, if we take a closer look at (2.5) we can see that correntropy is a similarity measure between two random variables D and Y. In fact, abnormally large error samples (or outliers) are given small weights and are filtered out by Gaussian kernel while ones with smaller values have larger contribution in the learning process inasmuch as they are assigned larger weights. Note that we could use other kernel functions for error correntropy [18, 19, 55] and incorporate more higher-order moments, however, we consider Gaussian kernel because of its nice properties, nevertheless, even for Gaussian kernel correntropy contains all even-order moments of the error PDF (if they exist) based on the Taylor expansion which as stated earlier shows superiority of ECC over MSE for learning in non-Gaussian environments.

As for calculating quadratic Renyi's error entropy we need to estimate error PDF first.

Remember that quadratic Renyi's error entropy is defined as follows:

$$H_2(E) = -\log I_2(E),$$

where $I_2(E) = \mathbf{E}\{p_E(E)\} = \int p_E^2(e)de$ is called information potential and $p_E(e)$ denotes the error PDF. Since $\log(.)$ is a monotonically increasing function, the following optimization problems are the same:

$$\min_{\mathbf{w}} H_2(E) = \max_{\mathbf{w}} I_2(E),$$

and hence it suffices to maximize information potential $I_2(E)$. This means that we only need to estimate the information potential at each time instant. However, similar to the case of MCC, we are dealing with an extremely large data-set where we are receiving continuously new data (error) samples with time, and so it is not efficient to use a batch estimator to incorporate all data samples for information potential estimation. Therefore, we utilize an online (or stochastic) approach and estimate information potential at time instant n from past N error samples as follows:

$$I_2(E) = \mathbf{E}\{p_E(E)\} \approx \frac{1}{N}\sum_{i=0}^{N-1} p_E(e_{n-i}) = \hat{I}_2^{(n)}(E). \tag{2.6}$$

Note that although online approaches cannot usually optimize our cost function precisely, they are able to quickly process an extremely large data-set and get close enough to the optimum solution [56]. As stated earlier, we do not know the error statistics, therefore we estimate $p_E(e_{n-i})$ in (2.6) from error samples by using the Parzen non-parametric PDF estimation (2.4) as follows:

$$p_E(e_{n-i}) \approx \frac{1}{N}\sum_{j=0}^{N-1} G_\sigma(e_{n-i} - e_{n-j}). \tag{2.7}$$

Substituting (2.7) into (2.6), we have the following estimate of information potential from past N error samples at time instant n:

$$\hat{I}_2^{(n)}(E) = \frac{1}{N^2} \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} G_\sigma(e_{n-i} - e_{n-j}). \tag{2.8}$$

Relation (2.8) is indeed the online cost function of MEE that we optimize using some technique such as stochastic gradient ascent. MEE has a computational complexity of $O(N^2)$ in each iteration. It is worth highlighting that, EEC contains all higher-order moments of the error PDF (if they exist) regardless of the type of the kernel function we use for Parzen non-parametric PDF estimation as opposed to ECC.

In the next chapters of this thesis we explain our contribution towards improving the performance of the algorithms based on ECC and EEC for online linear regression (or interchangeably linear adaptive filtering) in environments corrupted by non-Gaussian noise.

Chapter 3

A Hybrid Approach Based on Correntropy

3.1 Introduction

As we discussed in previous chapters, it is well-known that the learning algorithm based on MSE is not reliable in environments that are corrupted by non-Gaussian noise, or when there are outliers. Note that non-Gaussian noise may exist in the real world, for instance due to imperfect measurements or existence of limited number of noise sources. We said the reason that MSE is not reliable in non-Gaussian environments is the fact that MSE employs second order statistics of error process while we need a cost function that involves higher order statistics. Correntropy is an alternative choice that can provide us with higher order statistics of the underlying error PDF and the algorithm based on this cost function for online linear regression is robust against non-Gaussian noise. Remember that we consider linear adaptive filtering or online linear regression where the system parameters are learned continuously as new data samples are received. This problem can have a wide range of applications from channel

equalization to network traffic prediction and many others. Consider a linear filter for which the goal is to find the filter parameters so that the error between desired response and system output is minimized. We maximize error correntrpy which results in MCC algorithm. As seen in previous chapter we only have data samples (lets say a window of N data samples) not data statistics, therefore we estimate the error PDF by using Parzen non-parametric PDF estimation. Although both MEE and MCC are cost functions superior to MSE in the presence of non-Gaussian noise or outliers, there are two significant advantages to MCC over MEE. First, as opposed to MCC, we should regularize MEE after minimizing error entropy by putting the error at the origin while this is not always an easy task to do (we will discuss about this fact in next chapters) [57]. Second, computational complexity of MEE at each adaptation step is $O(N^2)$ which is higher than that of MCC which is $O(N)$.

Recursive least squares (RLS) algorithm is known to be desirably accurate and converges faster than the least mean square (LMS) algorithm. However, its shortcoming is its high computational complexity compared to LMS algorithms. Both RLS and LMS are based on second order statistics of the error, and consequently are mostly useful in linear systems or when the environment is under the effect of Gaussian noise. To tackle presence of non-Gaussian noise in the environment and also benefit from advantages of recursive algorithms, an RLS-type algorithm based on MCC (RMCC) was proposed in [58]. Later on, this algorithm was regularized in [59] by adding a general convex function to MCC in order to deal with the problem of sparse system identification (sparse systems have many near-zero coefficients). In addition, a kernel recursive adaptive filtering based on MCC was proposed in [60] for tackling both system non-linearity and presence of non-Gaussian noise in the environment.

Kernel bandwidth has a significant impact on robust learning based on MEE and MCC. In fact, value of kernel bandwidth imposes a compromise between fast convergence and low steady-state misalignment. Therefore, we can also optimize this free parameter as an extra step

in order to converge faster and moreover achieve a lower steady-state misalignment. It is worth noting that this free parameter is more flexibly adaptable in MCC compared to MEE because of the smooth dependency of correntropy to it [17]. Many authors have tried to adapt kernel bandwidth in information-theoretic criteria [61–67]. Most of kernel bandwidth adaptations have been proposed in gradient-based methods. Note that adapting kernel bandwidth in an RMCC algorithm is not as straightforward as gradient-based methods, for instance authors in [67] proposed a method on kernel bandwidth adaptation in RMCC based on minimizing Kullback-Leibler divergence between true and estimated version of error PDF, however computational complexity of this approach is high. On the other hand, although adapting kernel bandwidth in gradient-based methods is not computationally as expensive as that of RMCC, the convergence speed in these cases is even slower than RMCC without variable kernel bandwidth.

In this chapter, we consider linear adaptive filtering as an online linear regression problem and propose a joint recursive, gradient-based approach based on MCC. The proposed algorithm starts with RMCC to benefit from its fast convergence speed and then switches to a gradient-based algorithm in which we can adapt kernel bandwidth with a low computational complexity and achieve lower steady-state misalignment. Therefore, our approach results in both fast convergence speed and low steady-state misalignment. The content of this chapter has been published in [68].

In the following, we explain RMCC and LMS-type MCC in more details.

3.2 RLS-type and LMS-type MCC

As mentioned earlier, we consider a linear adaptive filtering (or equivalently an online linear regression). Here, d_n denotes label (or desired response) at time instant n which has been corrupted by measurement noise. Moreover, $\mathbf{x}_n = \left[x_n, x_{n-1}, \cdots, x_{n-L+1}\right]^T$ is the input vector at

time instant n with length L (L denotes size of the adaptive filter as well). The output of this linear filter is $y_n = \mathbf{x}_n^T \mathbf{w}_{n-1}$ where $\mathbf{w}_{n-1}$ denotes filter parameters vector estimated at previous time instant. Error sample at time n is obtained as $e_n = d_n - \mathbf{x}_n^T \mathbf{w}_{n-1}$.

Remember from previous chapter that correntropy is a similarity measure between two random variables D and Y and is defined as,

$$v(D,Y) = E_{D,Y}\left\{G_\sigma(D-Y)\right\} = E_E\left\{G_\sigma(E)\right\}, \tag{3.1}$$

in which E denotes error $D-Y$, $G_\sigma(.)$ is a kernel function and σ is kernel bandwidth. Expectation in correntropy is approximated by sample mean since we only have data samples $\{d_n, y_n\}$, $n = 1,\cdots,N$, (or equivalently $\{e_n\}, n = 1,\cdots,N$) and not the joint PDF of D and Y (or equivalently not PDF of error E) therefore, we have,

$$\hat{v}(D,Y) = \frac{1}{N}\sum_{n=1}^{N} G_\sigma(d_n - y_n) = \frac{1}{N}\sum_{n=1}^{N} G_\sigma(e_n). \tag{3.2}$$

Throughout this chapter we consider the following kernel,

$$G_\sigma(D-Y) = \exp\left(-\frac{\| D-Y \|^2}{2\sigma^2}\right).$$

It is worth reminding from previous chapter that the reason we can use correntropy as an objective function to minimize error between labels and filter output is the fact that approximation of correntropy in (3.2) is related to Parzen error PDF estimation. Actually, using Parzen windows for error PDF estimation from N samples, we have:

$$\hat{p}_E(e) = \frac{1}{N}\sum_{n=1}^{N} G_\sigma(e - e_n).$$

It is clear that $\hat{v}(D,Y) = \hat{p}_E(0)$, therefore, maximizing estimate of error PDF at 0 is equivalent to maximizing estimate of correntropy. As mentioned earlier, correntropy maximization algorithm is called MCC in literature.

Basically, there are two main types of algorithms based on MCC: RLS-type (or recursive) and LMS-type (or stochastic gradient-based) algorithms. We review these algorithms in the following subsections.

3.2.1 Recursive MCC

Consider the correntropy definition in (3.1). At each time instant n we can use all previous error samples (not only past N samples) to estimate correntropy and then learn the system. Therefore, we can write the following online cost function at time instant n for RMCC algorithm:

$$J_{RMCC}(\mathbf{w}_n) = \sum_{k=1}^{n} \lambda^{n-k} \exp\left(-\frac{\left(d_k - \mathbf{x}_k^T \mathbf{w}_n\right)^2}{2\sigma^2}\right), \tag{3.3}$$

where $0 << \lambda < 1$ is the forgetting factor. By using a forgetting factor, we are actually approximating (3.1) with weighted sample mean which is suitable for non-stationary environments. Note that (3.3) emphasizes on recent error samples in learning process. The $J_{RMCC}(\mathbf{w}_n)$ is maximized by taking the derivative with respect to $\mathbf{w}_n$ and setting it equal to zero. We obtain,

$$\begin{aligned} &\sum_{k=1}^{n} \lambda^{n-k} \exp\left(-\frac{\left(d_k - \mathbf{x}_k^T \mathbf{w}_n\right)^2}{2\sigma^2}\right) \left(d_k - \mathbf{x}_k^T \mathbf{w}_n\right) \mathbf{x}_k = 0, \\ &\Longrightarrow \sum_{k=1}^{n} \lambda^{n-k} g_k d_k \mathbf{x}_k = \sum_{k=1}^{n} \lambda^{n-k} g_k \mathbf{x}_k \mathbf{x}_k^T \mathbf{w}_n, \end{aligned} \tag{3.4}$$

in which $g_k = \exp\left(-\frac{\left(d_k - \mathbf{x}_k^T \mathbf{w}_n\right)^2}{2\sigma^2}\right)$. We can define matrix $R_{\mathbf{x}}(n)$ and vector $R_{d\mathbf{x}}(n)$ as follows:

$$R_{\mathbf{x}}(n) \triangleq \sum_{k=1}^{n} \lambda^{n-k} g_k \mathbf{x}_k \mathbf{x}_k^T,$$
$$R_{d\mathbf{x}}(n) \triangleq \sum_{k=1}^{n} \lambda^{n-k} g_k d_k \mathbf{x}_k.$$

Substituting above matrix and vector in (3.4) and solving it for $\mathbf{w}_n$, we obtain:

$$\mathbf{w}_n = R_{\mathbf{x}}(n)^{-1} R_{d\mathbf{x}}(n). \tag{3.5}$$

We write recursive versions of $R_{\mathbf{x}}(n)$ and $R_{d\mathbf{x}}(n)$ as follows to avoid increasing memory depth when calculating at each time instant,

$$R_{\mathbf{x}}(n) = \lambda R_{\mathbf{x}}(n-1) + g_n \mathbf{x}_n \mathbf{x}_n^T,$$
$$R_{d\mathbf{x}}(n) = \lambda R_{d\mathbf{x}}(n-1) + g_n d_n \mathbf{x}_n. \tag{3.6}$$

Moreover, to avoid computational load of computing inverse of $R_{\mathbf{x}}(n)$ we can use matrix inversion lemma as follows:

$$R_{\mathbf{x}}^{-1}(n) = \left[\lambda R_{\mathbf{x}}(n-1) + g_n \mathbf{x}_n \mathbf{x}_n^T\right]^{-1} = \frac{R_{\mathbf{x}}^{-1}(n-1)}{\lambda} - \frac{\left(\frac{R_{\mathbf{x}}^{-1}(n-1)\mathbf{x}_n}{\lambda}\right)\left(\frac{\mathbf{x}_n^T R_{\mathbf{x}}^{-1}(n-1)}{\lambda}\right)}{\left(g_n^{-1} + \frac{\mathbf{x}_n^T R_{\mathbf{x}}^{-1}(n-1)\mathbf{x}_n}{\lambda}\right)}. \tag{3.7}$$

Now, substituting (3.7) and (3.6) in (3.5) and doing some calculations, we can obtain following recursive solution for weight vector $\mathbf{w}_n$,

$$\mathbf{w}_n = \mathbf{w}_{n-1} + \left(\frac{R_{\mathbf{x}}^{-1}(n-1)\mathbf{x}_n}{\lambda g_n^{*-1} + \mathbf{x}_n^T R_{\mathbf{x}}^{-1}(n-1)\mathbf{x}_n}\right) e_n,$$

in which g_n is replaced with $g_n^* = \exp\left(-\frac{\left(d_n - \mathbf{x}_n^T \mathbf{w}_{n-1}\right)^2}{2\sigma^2}\right)$ since g_n is a function of $\mathbf{w}_n$ which is not known yet.

Remark 3.1: Advantages of RMCC can be sorted as follows:

- As opposed to gradient-based methods, there is no need to determine stepsize as an extra parameter.
- As seen in (3.3), since we are taking into account previous error samples, RMCC converges very fast to a desirable steady-state misalignment compared to gradient-based methods.

Remark 3.2: Disadvantages of RMCC can be sorted as follows:

- Computational complexity of RMCC is higher than gradient-based methods.
- In order to achieve a better convergence speed and lower steady-state misalignment we can adapt kernel bandwidth. However, adapting kernel bandwidth in RMCC is not as straightforward as it is in gradient-based methods.

3.2.2 LMS-type Method based on MCC

Consider (3.1) again. In LMS-type methods a gradient ascent search is used based on stochastic approximation in which expectation operator of (3.1) is removed and only current error sample is used. Therefore, we get the following cost function:

$$\begin{aligned} & J_{MCC}\left(\mathbf{w}_{n-1}\right) = \exp\left(-\frac{e_n^2}{2\sigma^2}\right) \\ \Longrightarrow \quad & \nabla J_{MCC}\left(\mathbf{w}_{n-1}\right) = \frac{1}{\sigma^2}\exp\left(-\frac{e_n^2}{2\sigma^2}\right) e_n \mathbf{x}_n \\ \Longrightarrow \quad & \mathbf{w}_n = \mathbf{w}_{n-1} + \frac{\mu}{\sigma^2}\exp\left(-\frac{e_n^2}{2\sigma^2}\right) e_n \mathbf{x}_n, \end{aligned} \tag{3.8}$$

where μ denotes the stepsize.

Remark 3.3: The advantages of LMS-type methods based on MCC are simplicity and also ease of elaborating kernel bandwidth in order to achieve a good performance in terms of convergence speed and steady-state misalignment.

In the following section, we propose our new approach based on MCC.

3.3 The Proposed Hybrid MCC

In this section, we propose a method which is a combination of RMCC and an LMS-type method to learn the underlying linear system. We start off with RMCC at initial iterations to benefit from its fast convergence speed, and then when we are close enough to steady-state misalignment of RMCC we switch to a new LMS-type method that not only is computationally less expensive but also can update kernel bandwidth and achieve a lower steady-state misalignment compared to RMCC. Therefore, the resultant overall algorithm is both fast and reliably accurate.

First, we present our new LMS-type method. As seen in (3.8), change in σ affects terms $\frac{\mu}{\sigma^2}$ and $\exp\left(-\frac{e_n^2}{2\sigma^2}\right)$ in opposite directions, therefore we choose an alternative for the cost function as follows:

$$\begin{aligned} & J_{NEW}\left(\mathbf{w}_{n-1}\right) = \sigma^2 f\left(\sigma^2\right) \exp\left(-\frac{e_n^2}{2\sigma^2}\right) \\ \Longrightarrow \quad & \mathbf{w}_n = \mathbf{w}_{n-1} + \mu f\left(\sigma^2\right) \exp\left(\frac{-e_n^2}{2\sigma^2}\right) e_n \mathbf{x}_n. \end{aligned} \tag{3.9}$$

Now, if σ is chosen to be variable, then $f(.)$ is also variable and consequently we can control the weight updating rule above. Assume that kernel bandwidth has a linear relation with instantaneous error, as considered in [65], i.e.,

$$\sigma_n = k_\sigma |e_n|, \tag{3.10}$$

where k_σ is a positive constant. This is a reasonable choice for σ_n, because it means that when e_n is small (large or equivalently an outlier) then σ_n is small (large) too and according to shape of Gaussian kernel a larger (smaller) weight is assigned to the error sample.

Next step after having a variable kernel bandwidth would be to specify a proper function $f(.)$ which changes with the variable σ in a way that makes (3.9) more efficient. Generally speaking, we expect the function $f(.)$ to behave such that when $|e_n|$ is not an outlier (i.e., it is reasonably large), filter parameters should be updated normally based on (3.9). On the other hand, if error sample $|e_n|$ is an outlier or abnormally large, updating rule (3.9) should not rely on this outlier error sample. Moreover, when $|e_n|$ is very small we expect a slow update of filter parameters. We consider following function $f(.)$ in (3.9):

$$f\left(\sigma^2\right) = \frac{\alpha\left(1-\exp\left(-\sigma^2\right)\right)}{\sqrt{1+\left(\frac{\sigma^2}{\mathscr{B}}\right)^{2m}}}. \tag{3.11}$$

This function has the above mentioned properties and is inspired by the Butterworth filter, where m and $\mathscr{B}$ are filter order and bandwidth, respectively. Suppose that parameters α (positive gain) and m are fixed (although they can be adapted). Bandwidth $\mathscr{B}$ is the other parameter which has a significant effect in (3.11) and we try to adapt it in each iteration. Indeed, it determines whether σ_n (or based on (3.10) error sample $|e_n|$) is an outlier or not. We select $\mathscr{B}$ at time instant n as average of all past error samples. This choice of $\mathscr{B}$ helps us diminish the effect of outlier error samples by making the value of $f\left(\sigma^2\right)$ very small. This means that weight vector is updated very slowly in (3.9) when an outlier occurs.

Learning curves of these two algorithms are shown in Figure 3.1. As seen in the figure,

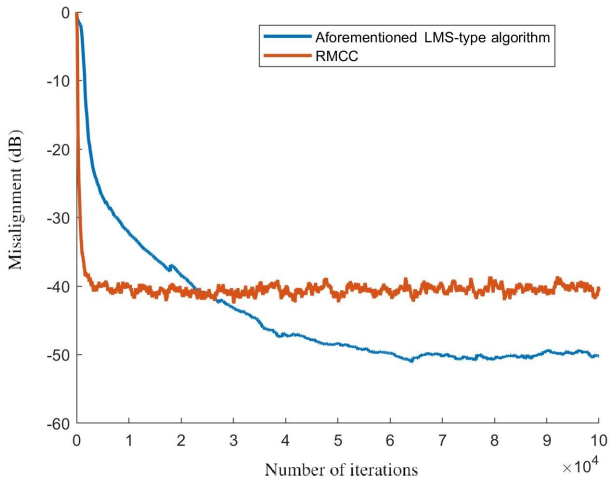
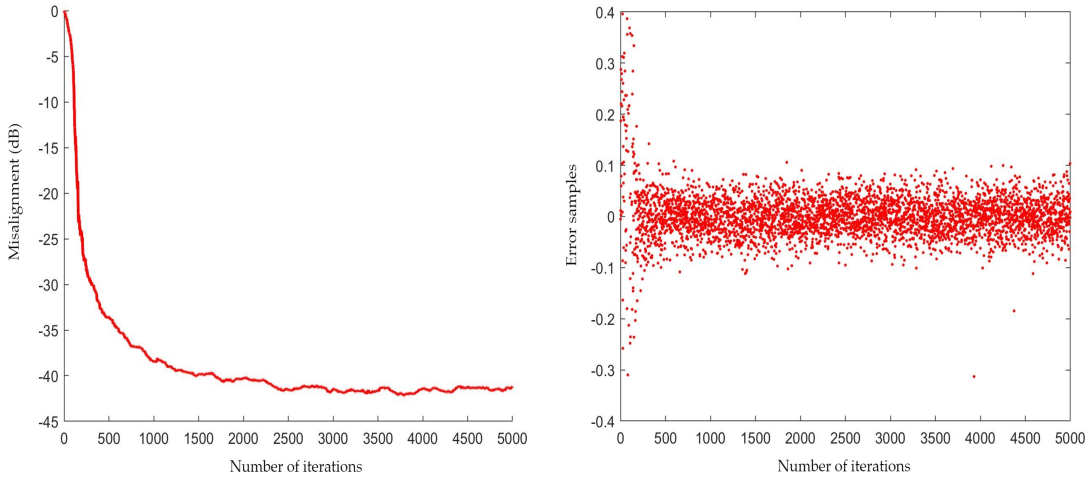


Figure 3.1: Learning curves of RMCC with $\sigma = 0.3$ and mentioned LMS-type algorithm.

RMCC has a fast convergence speed while the presented LMS-type algorithm benefits from lower steady-state misalignment. We next specify when we should switch from RMCC algorithm to aforementioned LMS-type algorithm. Based on Figure 3.1, it seems that an ideal time to switch would be when the RMCC algorithm gets close to its steady-state behavior. However, how could we recognize the switching point? Figure 3.2 illustrates how steady-state in the learning curve of RMCC is related to change in the error samples density. Indeed, when learning curve approaches its steady-state in Figure 3.2a, the percentage of error samples around zero in Figure 3.2b increases. We can see how this switching point starts being approached around the same iteration in both figures (around iteration 500). We therefore put a threshold for the percentage of error samples around zero and switch to the other algorithm after the threshold is exceeded. The algorithm of our proposed method is given in Algorithm 1.

Remark 3.4: Note that, after some iterations, the percentage of error samples around zero does not change significantly, therefore, in practice, a multiple of iteration number related to the mentioned threshold can be considered as the switching point to the other algorithm in order to be sure that we have reached the steady-state.



(a) (b)

Figure 3.2: Switching point (around iteration 500) in plots of (a): learning curve of RMCC and (b): error samples density around zero

In the next section, simulation results show efficiency of our method compared to other methods.

3.4 Simulation Results

We model an impulsive noise environment as follows. Input samples x_i are distributed as $x_i \sim \mathcal{N}(0,1)$. The unit vector $\mathbf{w}_{opt} \in \mathscr{R}^L$ is the optimum weight vector of the unknown filter, generated randomly and $L = 100$ denotes the filter length. At each time instant n, the desired signal is modeled as,

$$d_n = \mathbf{x}_n^T \mathbf{w}_{opt} + \nu_n + \eta_n,$$

in which $\nu_n \sim \mathcal{N}\left(0, \sigma_{\nu,n}^2\right)$ and η_n are white Gaussian and impulsive measurement noises, respectively.

Algorithm 1 Online Linear Regression Based on the Proposed Hybrid MCC

Inputs: $\{\mathbf{x}_n, d_n\}$
Output: $\mathbf{w}_n$
Initialisation :

$$\mathbf{w}_0 = \mathbf{0},\ \sigma = \sigma_0,\ \mathscr{B}_0 = 0,$$

Fixed parameters :

$$\lambda = 0.999,\ R_{\mathbf{x}}^{-1}(0) = \gamma I,\ \varepsilon = \varepsilon_0,$$
$$k_\sigma = 20,\ \alpha = 5,\ m = 15$$

for each iteration n **do**
  $e_n = d_n - \mathbf{x}_n^T \mathbf{w}_{n-1}$

$$\mathscr{B}_n = \frac{((n-1)\,\mathscr{B}_{n-1} + e_n)}{n}$$

  if (percentage of error samples around zero $< \varepsilon_0$) **then**
    $g_n^* = \exp\left(-\frac{e_n^2}{2\sigma_2}\right)$

$$G_n \triangleq \frac{R_{\mathbf{x}}^{-1}(n-1)\mathbf{x}_n}{\lambda g_n^{*-1} + \mathbf{x}_n^T R_{\mathbf{x}}^{-1}(n-1)\mathbf{x}_n}$$

$$R_{\mathbf{x}}^{-1}(n) = \frac{R_{\mathbf{x}}^{-1}(n-1)}{\lambda} - G_n \frac{\mathbf{x}^T R_{\mathbf{x}}^{-1}(n-1)}{\lambda}$$

    $\mathbf{w}_n = \mathbf{w}_{n-1} + G_n e_n$
  else
    $\sigma_n = k_\sigma |e_n|$

$$\mathbf{w}_n = \mathbf{w}_{n-1} + \mu \left(\frac{\alpha\left(1 - \exp\left(-\sigma_n^2\right)\right)}{\sqrt{1 + \left(\frac{\sigma_n^2}{\mathscr{B}_{n-1}}\right)^{2m}}} \right) \exp\left(\frac{-e_n^2}{2\sigma_n^2}\right) e_n \mathbf{x}_n$$

  end if
end for
return $\mathbf{w}_n$ =0

Assume that SNR between signal and white Gaussian measurement noise is 30dB, i.e.,

$$\mathrm{SNR} = 10\log_{10}\left(\frac{E\left\{\left[\mathbf{x}_n^T \mathbf{w}_{opt}\right]^2\right\}}{\sigma_{v,n}^2}\right) = 30.$$

We model impulsive measurement noise as $\eta_n = \beta_n \omega_n$ where $\beta_n \sim Bernoulli(p)$ and p denotes

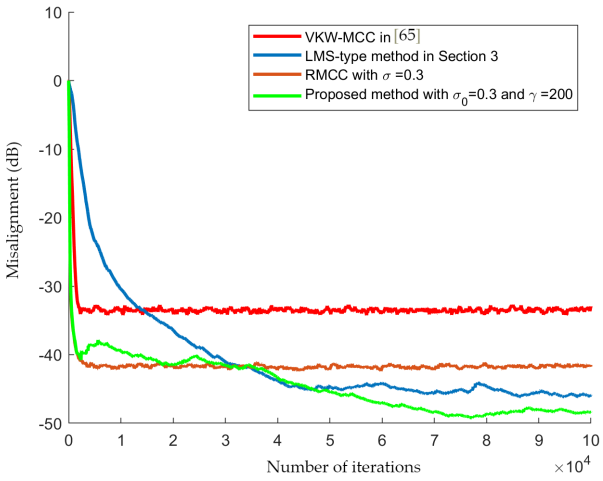


Figure 3.3: Learning curves: misalignment vs. iteration. Step size μ is set to 0.01 and probability of impulses in noise is $p = 0.2$.

the probability of existence of impulses in noise and,

$$\omega_n \sim \mathcal{N}\left(0, 1000E\left\{\left[\mathbf{x}_n^T \mathbf{w}_{opt}\right]^2\right\}\right).$$

We start with RMCC and continue this algorithm for 10 times of the iteration in which the percentage of error samples around zero is $\varepsilon_0 = \%80$. After that, we switch to the LMS-type method described before. Having weight vector $\mathbf{w}_n$ at each time instant n, we can calculate misalignment based on the normalized mean-square deviation (NMSD) as follows:

$$\text{misalignment}_n = 10\log_{10}\left(\| \frac{\mathbf{w}_n - \mathbf{w}_{opt}}{\mathbf{w}_{opt}} \|^2\right).$$

Fig. 3.3 depicts the learning curves of different algorithms for 100000 iterations obtained by averaging over 10 independent trials. As seen, our proposed method outperforms other algorithms either in terms of convergence rate, steady-state misalignment, or both. A significant advantage of our method compared to RMCC is that its convergence speed is almost as good as RMCC while it is computationally less expensive. In addition, it achieves the lowest steady-state misalignment

compared to all these algorithms.

3.5 Conclusion

A hybrid technique based on MCC for online linear regression (or linear adaptive filtering) is addressed in this chapter. We propose a new method based on correntropy to confront presence of outliers and impulsive noise in the environment. Our method either converges faster, achieves lower steady-state, or benefits from both compared to other algorithms. It is worth mentioning that in addition to these advantages it is also computationally less expensive compared to recursive MCC. However, there are some disadvantages to our proposed method. More precisely, finding switching point is challenging, extra hyper-parameters are need to be adjusted, and, most importantly, error sample mean is not a robust and reliable measure for outlier detection.

In the next chapter, we strive to address the above shortcomings of our proposed hybrid method based on MCC.

Chapter 4

Outlier-Rejected Adaptive MCC

4.1 Introduction

As discussed previously, in learning theory, it is widely established that MSE is not a reliable cost function when the error either is non-Gaussian or contains outliers. Although non-Gaussian error seems to be abnormal based on the central limit theorem, it exists in the real world due to for instance ambient component noises or imperfect measurements, and should not be ignored. We saw that a very good and efficient alternative to MSE criterion in aforementioned environments is ECC. In fact, correlation can be substituted with correntropy in information-theoretic learning context. Again, assume we deal with an linear adaptive filter (which can be considered as an online linear regression problem) in which we try to learn the parameters of a system continuously as we are receiving new data samples such that error between system output and labels (or desired responses) is minimized. Remember that ECC in this supervised learning problem is a local similarity measure between system output and label which is maximized because when error correntropy is defined based on the same kernel function used for Parzen error PDF estimation, its estimation from error samples reduces to estimation of error PDF at zero. It is worth reminding that ECC has lower computational complexity compared to the other

information-theoretic cost function, i.e., EEC. Moreover, it does not need any extra regularizations whereas EEC needs to put the error at the origin after minimizing the error entropy, e.g., by biasing the system output.

Both ECC and EEC use kernel functions in which kernel bandwidth can be viewed as a free parameter that can be also optimized to increase learning precision. Interestingly, this free parameter is much more flexible to be optimized in MCC compared to MEE. Although MEE and MCC have been compared to each other in some senses (for instance an interesting information-theoretic comparison of MEE and MCC criteria can be found in [34]), this superiority of MCC (i.e., more flexibly adaptable kernel bandwidth) has not been paid enough attention. This flexibility in kernel bandwidth adaptation for MCC emerges from smooth dependency of MCC to the value of the kernel bandwidth since error PDF estimation does not matter when we consider MCC while in MEE, the kernel bandwidth can not be selected arbitrarily in the sense that very large or very small values for kernel bandwidth are prohibited. The reason for this prohibition is that MEE uses directly entropy estimation, and consequently Parzen error PDF estimation is involved, therefore the value of kernel bandwidth should be selected in a way that it compromises between bias and variance of Parzen error PDF estimation.

Adaptation of kernel bandwidth as an extra step in learning process has been already considered in the literature [61–67]. For instance, in [61] and [67] Kullback-Leibler (KL) divergence between the true and estimated error distribution is minimized as a second cost function in the overall adaptation problem. However, this approach will not be very efficient (for instance when the initial weight vector is far from the optimal weight vector, it can not improve the convergence speed [69]). In another work [62], the author tried to adapt the kernel bandwidth for MCC based on the shape of error distribution which is measured by its kurtosis. However, a satisfying estimation of the shape of error distribution is not a straightforward task to do. Authors in another paper [63] tried to propose a simple algorithm for kernel bandwidth

adaptation that involves no extra free parameters. In this algorithm, the kernel bandwidth in each iteration is updated based on instantaneous error and changed to a predetermined kernel bandwidth in order to avoid divergence when the updated kernel bandwidth is smaller than this predetermined value. Although this approach converges faster than previous algorithms, it almost keeps their steady-state behavior. The same authors modified their method in [64] in which they used another kernel bandwidth update rule. Although it is still based on instantaneous error and predetermined kernel bandwidth, the new update rule helps the method not only to converge faster but also to achieve a slightly lower steady-state excess mean square error (EMSE) than that of MCC. Authors in [65] changed the Gaussian kernel used in the MCC definition and developed a new algorithm to update kernel bandwidth which converges faster than previous algorithms, and even can achieve a lower steady-state misalignment especially in the environment where impulsive noise is considerably likely. Although the structure of their algorithm is very similar to previous gradient based algorithms, it is computationally more expensive compared to them. We also proposed a hybrid method, described in Chapter 3, which achieves both fast convergence and low steady-state misalignment, however many parameters are left to be predetermined. In addition, this approach is very sensitive to step size variation.

In this chapter, we consider again linear adaptive filtering (or online linear regression) used in various signal processing and machine learning applications. We modify AMCC algorithm in [64] by stopping the learning process in each step whenever we face a major outlier. This method results in significant decrease in steady-state misalignment at the cost of some extra computations in order to check whether an error sample is a major outlier or not in each iteration. The content of this chapter has been published in [70].

In the following, we describe our outlier-rejected adaptive MCC algorithm in more details.

4.2 Our Proposed Outlier-Rejected AMCC

Remember from previous chapters the definition of error correntropy and its estimation from error samples (2.1 and 2.5). We consider following kernel function for MCC in this chapter:

$$G_\sigma(D-Y) = \exp(-\frac{\| D-Y \|^2}{2\sigma^2}).$$

Note that based on this kernel function, error samples are given a weight where error samples with smaller values have larger weights and consequently larger contribution in objective function utilized in adaptation and on the other hand, large error samples, or outliers, are given small weights and are filtered out by this kernel function. Again, in the online linear regression (or linear adaptive filtering) that we consider through this chapter, D is a random variable denoting desired response and Y denotes the output of learned linear filter.

In filtering out the outliers, kernel bandwidth σ has a significant role. In fact, kernel bandwidth determines the magnitude of weight that should be assigned to a specific error sample. Generally, it affects the convergence rate and steady-state misalignment, therefore a proper method for adaptation of σ will increase adaptive filtering efficiency.

Throughout the chapter we strive to learn the parameters of the aforementioned adaptive system. As mentioned earlier, d_n denotes label (or desired signal) at time instant n. Note that d_n is corrupted with measurement noise. Furthermore, $\mathbf{x}_n = \left[x_n, x_{n-1}, \cdots, x_{n-L+1}\right]^T$ is the input vector at time instant n with length L where L denotes the size of adaptive filter as well. We choose linear model for adaptive filter, therefore the output of this filter is $y_n = \mathbf{x}_n^T \mathbf{w}_{n-1}$ where $\mathbf{w}_{n-1}$ is filter parameter estimated at time instant $n-1$. Error sample at time n is obtained as $e_n = d_n - \mathbf{x}_n^T \mathbf{w}_{n-1}$.

Note that correntropy is a bounded function and we use gradient ascent technique

to maximize it. For simplicity, we consider stochastic gradient ascent in which we drop the expectation operator in (2.1) and only use the current error sample to approximate the correntropy. Therefore we have following online objective function:

$$J_{MCC}\left(\mathbf{w}_{n-1}\right) = G_{\sigma}\left(d_n - \mathbf{x}_n^T \mathbf{w}_{n-1}\right) = G_{\sigma}\left(e_n\right). \tag{4.1}$$

Then, gradient ascent algorithm is as follows:

$$\begin{aligned} \mathbf{w}_n &= \mathbf{w}_{n-1} + \mu \nabla J_{MCC}\left(\mathbf{w}_{n-1}\right) \\ &= \mathbf{w}_{n-1} + \frac{\mu}{\sigma^2} \exp\left(\frac{-e_n^2}{2\sigma^2}\right) e_n \mathbf{x}_n, \end{aligned} \tag{4.2}$$

where μ is step size and $\nabla J_{MCC}\left(\mathbf{w}_{n-1}\right)$ denotes the gradient of online objective function (5.2) with respect to $\mathbf{w}_{n-1}$.

As mentioned earlier, we modify AMCC algorithm to achieve lower steady-state misalignment. Therefore, we briefly review AMCC algorithm [64] first, then move forward to explaining the modification.

4.2.1 Review of AMCC Algorithm

Consider linear adaptive filtering in an environment with impulsive noise. Kernel bandwidth σ_n at each iteration is obtained such that (5.3) approaches to its optimum weight value faster. To this end, kernel bandwidth σ_n should maximize following term of (5.3) in each iteration:

$$h\left(\sigma^2\right) = \frac{1}{\sigma^2} \exp\left(-\frac{e_n^2}{2\sigma^2}\right). \tag{4.3}$$

Note that at each iteration n, error sample e_n has already been determined before updating kernel

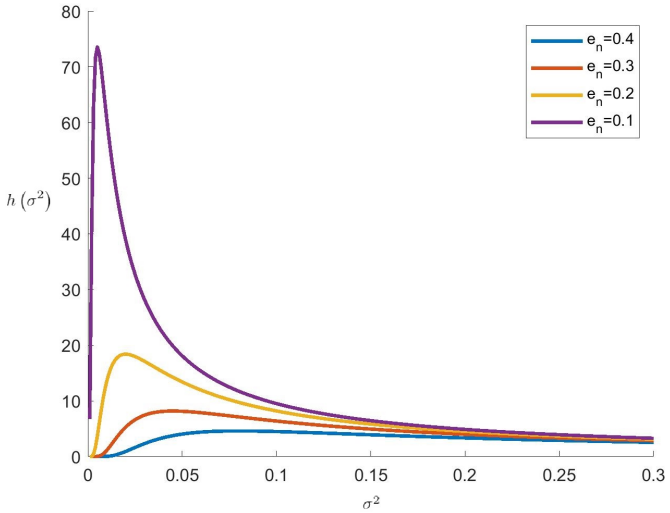


Figure 4.1: Plot of the function $h\left(\sigma^2\right)$ for different values of e_n.

bandwidth, therefore we only need to take derivative of (4.3) with respect to σ^2. We have:

$$\frac{\partial h\left(\sigma^2\right)}{\partial\left(\sigma^2\right)} = -\frac{1}{\sigma^4}\exp\left(-\frac{e_n^2}{2\sigma^2}\right) + \left(\frac{e_n^2}{2\left(\sigma^2\right)^3}\right)\exp\left(-\frac{e_n^2}{2\sigma^2}\right) = 0$$

$$\Longrightarrow \sigma_n^2 = \left\{\frac{{e_n}^2}{2}, \infty\right\}.$$

Clearly, we have:

$$h\left(\sigma^2\right)\Big|_{\sigma^2=\frac{e_n^2}{2}} = \frac{2}{e_n^2}\exp\left(-1\right) > \lim_{\sigma^2\to\infty} h\left(\sigma^2\right) = 0,$$

therefore, $\sigma_n^2 = \frac{e_n^2}{2}$ maximizes the function $h\left(\sigma^2\right)$ for a fixed e_n. Figure 4.1 shows how $h\left(\sigma^2\right)$ in (4.3) varies with σ^2 for different values of e_n. As seen in Figure 4.1, the maximum of function $h\left(\sigma^2\right)$ which obtained at $\sigma_n^2 = \frac{e_n^2}{2}$ tends to infinity as e_n tends to zero, therefore in order to avoid divergence of the algorithm (5.3) when e_n is getting smaller, adaptive kernel bandwidth is modified as follows:

$$\sigma_n^2 = \frac{e_n^2}{2} + \sigma_0^2, \tag{4.4}$$

in which, as explained in [64], kernel bandwidth at iteration n switches to a predetermined kernel bandwidth σ_0 when error e_n is small.

Thus far we only talked about an adaptive kernel bandwidth for MCC that speeds up the convergence rate. From now on we focus on decreasing steady-state misalignment.

4.2.2 Modification to AMCC Algorithm

In order to reach a lower steady-state misalignment we employ a filter with variable bandwidth in each iteration to reject major outliers. In fact, we modify the algorithm (5.3) as follows:

$$\mathbf{w}_n = \mathbf{w}_{n-1} + \frac{\mu}{\sigma_n^2} f(e_n) \exp\left(\frac{-e_n^2}{2\sigma_n^2}\right) e_n \mathbf{x}_n, \tag{4.5}$$

in which $f(e_n)$ is the filter and σ_n^2 is substituted from (4.4). However, what are the boundaries that specify major outliers? In other words, how we can determine the variable bandwidth of the filter $f(e_n)$? We use running quartiles of the error samples. Generally speaking, median (or generally any quantile) of a data set is a robust quantity of data against outliers, therefore we can use the concept of outer fences to determine filter boundaries for major outlier rejection [71, 72]. Figure 4.2 denotes these boundaries. In this figure, Q_1, Q_2 and Q_3 are lower quartile (or 25th percentile), median, and upper quartile (or 75th percentile), respectively. In addition, $IQR = Q_3 - Q1$ stands for inter-quartile range, and outer fences are shown by:

$$lower\ extreme = Q_1 - 3 \times IQR,$$

$$upper\ extreme = Q_3 + 3 \times IQR.$$

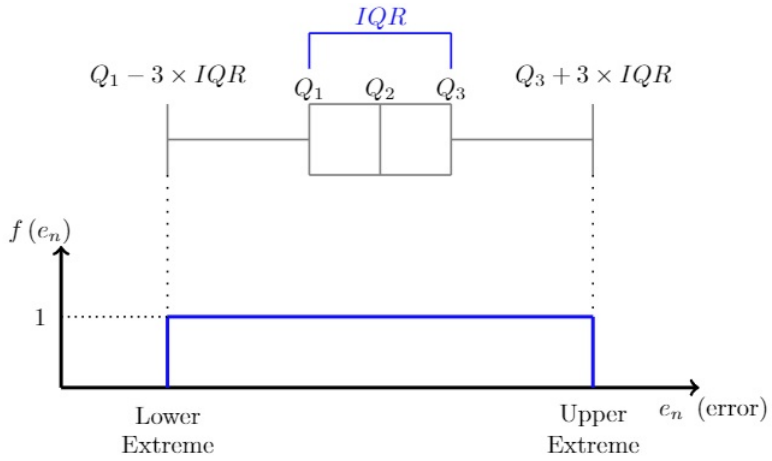


Figure 4.2: Plot of the function $f(e_n)$ at time instant n.

Algorithm 2 Outlier-rejected AMCC Algorithm for Online Linear Regression

Inputs: $\{\mathbf{x}_n, d_n\}$
Output: $\mathbf{w}_n$
Initialisation : $\mathbf{w}_0 = \mathbf{0}$ and σ_0

for each iteration n **do**
  $e_n = d_n - \mathbf{x}_n^T \mathbf{w}_{n-1}$
  $\sigma_n^2 = \frac{e_n^2}{2} + \sigma_0^2$
  Sort error samples and find Q_1 and Q_3
  $IQR = Q_3 - Q_1$
  Lower Extreme $= Q_1 - 3 \times IQR$
  Upper Extreme $= Q_3 + 3 \times IQR$
  if (Lower Extreme $\leq e_n \leq$ Upper Extreme) **then**
    $f(e_n) = 1$
  else
    $f(e_n) = 0$
  end if

$$\mathbf{w}_n = \mathbf{w}_{n-1} + \frac{\mu}{\sigma_n^2} f(e_n) \exp\left(\frac{-e_n^2}{2\sigma_n^2}\right) e_n \mathbf{x}_n$$

end for
return $\mathbf{w}_n$ =0

Note that quartiles are functions of time n which means these filter extremes vary once new error sample is available, therefore we have to deal with running quartiles. Running quartile estimation from data samples has been widely studied in literature [73–76]. In this chapter, we simply use order statistics in which we sort all observation samples at each time instant n. The complexity of this operation is $O(n)$ and we need to store all previous data samples, however we

could choose a proper algorithm from aforementioned algorithms in order to decrease memory or computational requirements. Our algorithm is shown in Algorithm 2.

4.3 Simulation Results

In this section, our simulation results show how our method outperforms other methods in term of steady-state misalignment. We use the model in [65, 77, 78] for impulsive noise environment. We assume that input samples x_i are drawn as $x_i \sim \mathcal{N}(0,1)$. Optimum weight vector of unknown filter is generated randomly and is a unit vector $\mathbf{w}_{opt} \in \mathscr{R}^L$ where $L = 5$ denotes the filter length. The desired signal at time instant n is modeled as,

$$d_n = \mathbf{x}_n^T \mathbf{w}_{opt} + \nu_n + \eta_n,$$

where $\nu_n \sim \mathcal{N}\left(0, \sigma_{v,n}^2\right)$ and η_n are white Gaussian and impulsive measurement noises, respectively. We assume that there is 30dB signal to white Gaussian measurement noise ratio where this signal to noise ratio is calculated as follows:

$$\text{SNR} = 10\log_{10}\left(\frac{E\left\{\left[\mathbf{x}_n^T \mathbf{w}_{opt}\right]^2\right\}}{\sigma_{v,n}^2}\right).$$

Impulsive measurement noise is created as $\eta_n = \beta_n \omega_n$ where $\beta_n \sim Bernoulli(p)$ in which p is probability of success (or equivalently the probability of existence of impulses in noise) and $\omega_n \sim \mathcal{N}\left(0, 1000E\left\{\left[\mathbf{x}_n^T \mathbf{w}_{opt}\right]^2\right\}\right)$. We assume $p = 0.2$ in our simulations. At each time instant n, we obtain $\mathbf{w}_n$, and accordingly we obtain misalignment based on the following NMSD:

$$\text{misalignment}_n = 10\log_{10}\left(\frac{\| \mathbf{w}_n - \mathbf{w}_{opt} \|^2}{\| \mathbf{w}_{opt} \|^2}\right).$$

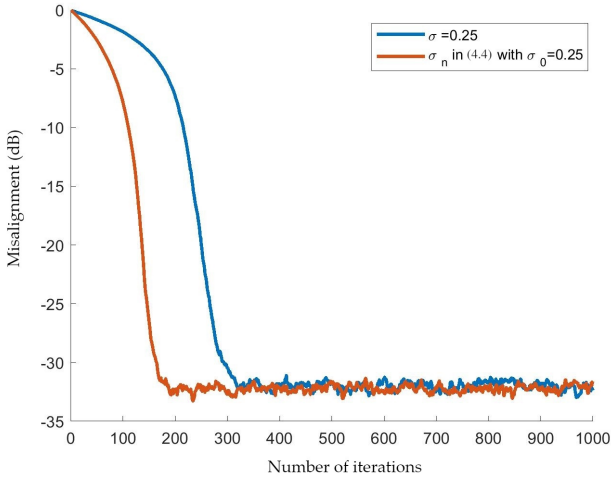


Figure 4.3: Learning curves of MCC (5.3) with fixed $\sigma = 0.25$ and with variable σ_n from (4.4) with $\sigma_0 = 0.25$ ($\mu = 0.01$).

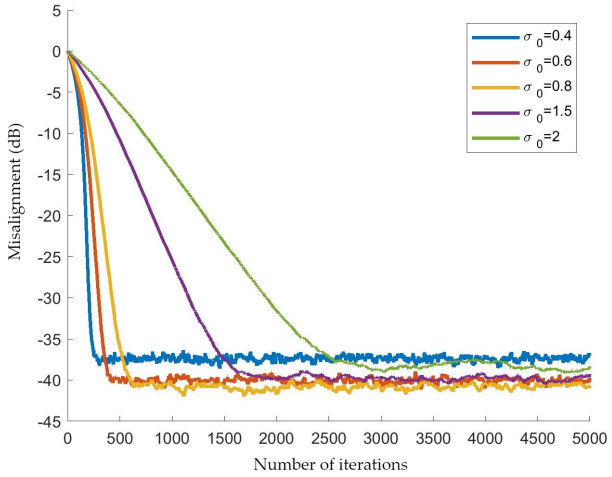


Figure 4.4: Learning curves of MCC (5.3) with variable σ_n from (4.4) with different values of σ_0 ($\mu = 0.01$).

First, consider (5.3) in which there is no filter $f(e_n)$, and $\sigma_n^2 = \frac{e_n^2}{2} + \sigma_0^2$. Figure 4.3 shows how using (4.4) for σ_n^2 can increase convergence rate. Figure 4.4 shows how learning curves of (5.3) with σ_n from (4.4) vary with different values of predetermined kernel bandwidth σ_0. As seen in this figure, by increasing predetermined kernel bandwidth σ_0 convergence rate always decreases while steady-state misalignment decreases first and then it increases.

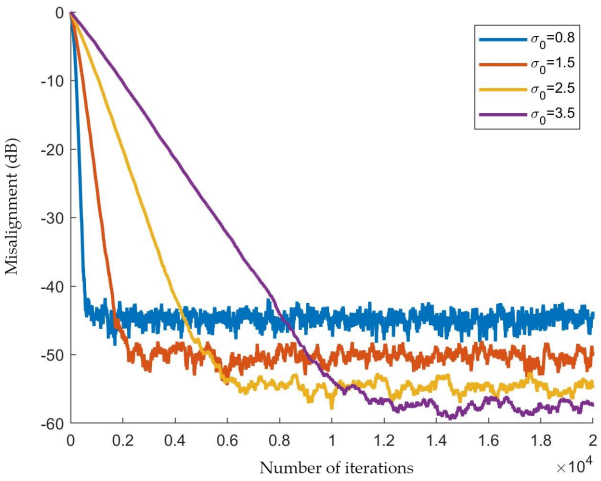


Figure 4.5: Learning curves of our algorithm (5.4) with different values of σ_0 ($\mu = 0.01$).

Now, what happens when we employ filter $f(e_n)$ in (5.3), i.e., when we use (5.4). As illustrated in Figure 4.5, when we employ the filter $f(e_n)$, learning curve always converges slower to a lower steady-state misalignment when we increase predetermined kernel bandwidth σ_0. Let us discuss the learning curve behaviour in Figures 4.4 and 4.5. As seen in these figures, by increasing predetermined kernel bandwidth σ_0 in (5.3) and (5.4) the gradient ascent whole step size $\frac{\mu}{\sigma_n^2}$ decreases and both algorithms converge slower. However, the steady-state misalignment behaviour of our algorithm is different with that of (5.3). The reason is that when there is no filter $f(e_n)$, although we are decreasing whole step size value $\frac{\mu}{\sigma_n^2}$ with increasing σ_0 and we expect to achieve lower steady-state misalignment with iterations, at the same time we are giving a big weight to outliers (i.e., large error samples) based on the correntropy definition which can result in higher steady-state misalignment. Therefore there is a trade-off between these two factors and once σ_0 is large enough the latter factor dominates the other one. We observe that this issue has been resolved in our algorithm as shown in Figure 4.5 in which learning curve always achieves lower steady-state misalignment with increase in σ_0.

Figure 4.6 illustrates how learning curve of our algorithm changes with step size μ. As expected, larger step size results in faster convergence to a higher steady-state misalignment.

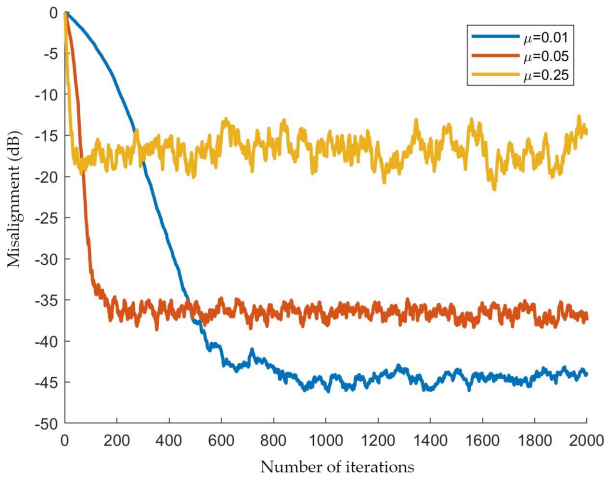


Figure 4.6: Learning curves of our algorithm (5.4) for $\sigma_0 = 0.8$ and different values of step size μ.

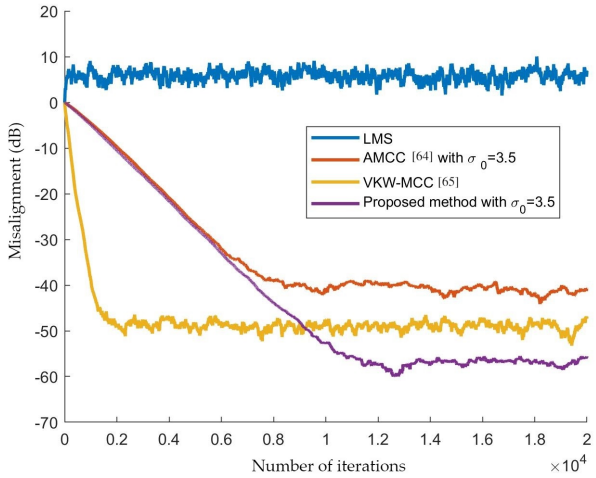


Figure 4.7: Learning curves of different algorithms ($\mu = 0.01$).

Finally, Figure 4.7 shows how our proposed method outperforms other algorithms from steady-state misalignment point of view. These learning curves for 20000 iterations are obtained by averaging over 10 independent trials. Step size μ is set to 0.01. As seen, the LMS algorithm diverges to a high steady-state misalignment when impulses occur in the noise (or equivalently when we have outlier error samples). AMCC algorithm in [64] outperform LMS when there is impulse in noise. VKW-MCC in [65] is both faster and achieves a lower misalignment compared to previous algorithms. Finally, our algorithm converges to the lowest steady-state misalignment

compared to other algorithms.

Note that we could combine our algorithm with a fast algorithm (e.g., recursive MCC) and propose a hybrid method like what we did in previous chapter in which the overall hybrid algorithm not only achieves a lower steady-state misalignment but also converges faster.

4.4 Conclusion

This chapter addresses again the problem of online linear regression (or linear adaptive filtering) which has many applications such as channel estimation. We consider the presence of outliers and impulsive noise in the environment. In this chapter, we use error samples running quartiles to find out whether a new error sample is a major outlier or not. If it is, we stop learning process (in which we use an existing algorithm called AMCC) based on that error sample and wait for next error sample to continue the learning process. Simulation results show that our algorithm achieves lower steady-state performance compared to previous results. Despite the aforementioned advantage of our proposed algorithm, a predetermined kernel bandwidth is needed which can be challenging and more importantly it is computationally expensive because of sorting in each iteration.

In the next chapter, we propose an efficient technique for major outlier detection and rejection.

Chapter 5

Efficient Outlier Detection in MCC

5.1 Introduction

Remember that in environments in which error is non-Gaussian or contains outliers, MSE is not a reliable cost function and as alternatives we could use ECC or EEC instead as robust cost functions. Interestingly, the relation between information-theoretic criteria EEC and ECC has been shown in [34] where authors found that difference between these cost functions is an Euclidean distance that under its variation their performances become more similar or more different. However, we use ECC in this chapter because of its simplicity. In this chapter, we consider again an online linear regression problem in which the goal is learning a system in a way that error between system output and desired responses is minimized. It is worth reminding that we could use the term linear adaptive filtering instead, however for simplicity we refer to our problem in this chapter only as online linear regression. Recall that the process is continuous, i.e., as new data is received the parameters of the system are updated. In previous chapter, we used qurtiles to detect and reject major outliers from learning process, however the method that we used for this purpose, i.e., sorting, is not efficient inasmuch as we needed to store all previous error samples and sort them to find quartiles while the complexity of this operation increases with

time. In this chapter, major outlier detection and rejection is done based on quartiles once again, however we use an efficient technique for estimating these quartiles. Our proposed technique in this chapter alongside MCC benefits from lower steady-state misalignment compared to previous algorithms.

Recall error correntropy definition (2.1) and its following estimate from data samples $\{d_n, y_n\}$, $n = 1 \cdots N$, (or equivalently $\{e_n\}, n = 1 \cdots N$), using sample mean:

$$\hat{v}(D,Y) = \frac{1}{N}\sum_{n=1}^{N} G_\sigma(d_n - y_n) = \frac{1}{N}\sum_{n=1}^{N} G_\sigma(e_n), \tag{5.1}$$

where random variable D denotes desired response and random variable Y denotes the output of learned linear system. We saw that maximizing the estimate of correntropy $\hat{v}(D,Y)$ is equivalent to maximizing the estimate of error PDF (obtained from Parzen method) at 0. Moreover, we saw the superiority of correntropy over MSE in presence of outliers (i.e., abnormally large error samples resulted from impulsive noise. In fact, we use $G_\sigma(D-Y) = \exp(-\frac{\|D-Y\|^2}{2\sigma^2})$ as the kernel function, hence from (5.1) we obviously see that this kernel function gives a weight to each error sample such that large error samples have small weights and are filtered out, while error samples with smaller values have larger weights and consequently have larger contribution in learning process.

Kernel bandwidth σ plays a significant role in outlier rejection. In fact, kernel bandwidth specifies how large the weight of a specific error sample should be. Although a good algorithm to update σ will increase learning efficiency [61] we skip σ adaptation in this chapter and only focus on elimination of major outliers (which have a destructive effect on steady-state misalignment).

Throughout this chapter, we try again to learn the parameters of a linear system. $\mathbf{x}_n = \left[x_n, x_{n-1}, \cdots, x_{n-L+1}\right]^T$ is the input vector at time instant n where L denotes the number

of system parameters. Output of this linear system is $y_n = \mathbf{x}_n^T \mathbf{w}_{n-1}$ where $\mathbf{w}_{n-1}$ is the system parameter estimated at time instant $n-1$. Error sample at time n is obtained as $e_n = d_n - \mathbf{x}_n^T \mathbf{w}_{n-1}$ where d_n denotes desired response at time instant n corrupted with measurement noise.

We use stochastic gradient ascent to maximize correntropy in which we drop the expectation operator in (2.1) and only use the current error sample. Thus online objective function is as follows:

$$J_{MCC}(\mathbf{w}_{n-1}) = G_\sigma\left(d_n - \mathbf{x}_n^T \mathbf{w}_{n-1}\right) = G_\sigma(e_n). \tag{5.2}$$

Then, we have the following stochastic gradient ascent algorithm:

$$\begin{aligned} \mathbf{w}_n &= \mathbf{w}_{n-1} + \mu \nabla J_{MCC}(\mathbf{w}_{n-1}) \\ &= \mathbf{w}_{n-1} + \frac{\mu}{\sigma^2} \exp\left(\frac{-e_n^2}{2\sigma^2}\right) e_n \mathbf{x}_n, \end{aligned} \tag{5.3}$$

in which μ and σ denote learning rate and kernel bandwidth, respectively and $\nabla J_{MCC}(\mathbf{w}_{n-1})$ is the gradient of online objective function (5.2) with respect to $\mathbf{w}_{n-1}$.

Major outliers (i.e., very large error samples) have significant destructive effect on steady-state misalignment. In the following, we describe our efficient running quartile estimation technique and combine it with MCC such that the learning process in each adaptation step is stopped by using our technique whenever we face a major outlier in error samples. This combination results in a lower steady-state misalignment compared to previous algorithms. The content of this chapter has been published in [79].

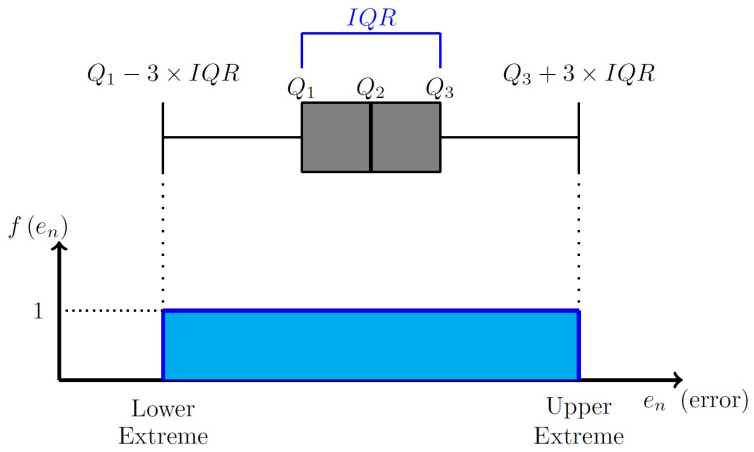


Figure 5.1: Function $f(e_n)$ at time instant n.

5.2 Our Running Quartile Estimation Technique

In this section, we propose our method for online regression in an environment with impulsive noise. Before we begin describing our method, note that when impulsive noise occurs and error sample $|e_n|$ is abnormally large, the system parameters should not be updated based on this major outlier error sample. Recall that, for this purpose, we employ a filter with variable bandwidth in (5.3) to detect and reject major outliers in each iteration. This results in decreasing steady-state misalignment. In fact we modify the algorithm (5.3) as follows:

$$\mathbf{w}_n = \mathbf{w}_{n-1} + \frac{\mu}{\sigma^2} f(e_n) \exp\left(\frac{-e_n^2}{2\sigma^2}\right) e_n \mathbf{x}_n, \tag{5.4}$$

where $f(e_n)$ is the filter. However, how can we determine the variable bandwidth of this filter that specifies the boundaries helping us find major outliers? We use running quartiles of the error samples to find these boundaries. Generally speaking, as any quantile of a data set is a robust quantity of data against outliers we use the concept of outer fences to determine filter boundaries for major outlier rejection [71, 72]. Figure 5.1 illustrates these boundaries in which Q_1, Q_2 and Q_3 are lower quartile (or 25th percentile), median, and upper quartile (or 75th percentile), respectively. Moreover, $IQR = Q_3 - Q_1$ stands for inter-quartile range, and outer

fences are defined as:

$$\text{upper extreme} = Q_3 + 3 \times IQR,$$

$$\text{lower extreme} = Q_1 - 3 \times IQR.$$

Obviously Q_1 and Q_3 are the only important quartiles to be obtained in each adaptation step. These quartiles are functions of time n and consequently filter extremes based on them should be updated once a new error sample is available. This means that we have to deal with a running quartile problem. we could simply use order statistics to obtain aforementioned quartiles in which we sort all observed samples at each time instant n (as we did in previous chapter). The shortcoming of this simple method is the fact that we have to store all previous data samples. The complexity of this operation is $O(n)$. Although there are many algorithms for running quartile estimation from data samples to decrease memory or computational requirements, we present our own algorithm that is as efficient as previous algorithms and further concentrates on estimation of only Q_1 and Q_3 and fits our problem.

We use non-uniform quantization [80] to quantize error samples, therefore, we deal with a small number of quantization levels (or bins) instead of all data samples in order to obtain quartiles. In our problem, we expect that most of error samples accumulate around $e = 0$ over time, so we use the following compressor function by which smaller error samples around $e = 0$ are quantized with more precision while error samples farther away from $e = 0$ are quantized less precisely:

$$C(e) = \begin{cases} \frac{1}{1+\exp(-\alpha_1 e)}, & e < 0 \\ \frac{1}{1+\exp(-\alpha_2 e)}, & 0 \leq e. \end{cases}$$

In the compressor function above, $0 < C(e) < 1$ and the function $\frac{1}{1+\exp(-\alpha_i e)}$ is called the logistic

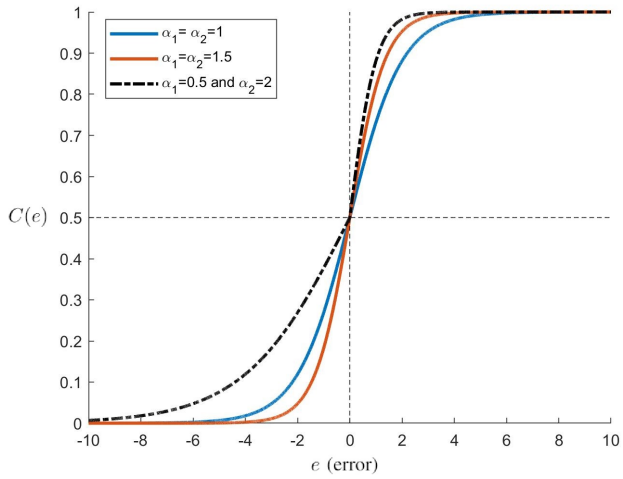


Figure 5.2: Compressor function used in our non-uniform quantizer.

function. The parameters α_1 and α_2 determine the precision of quantization for $e < 0$ and $0 \leq e$, respectively. Figure 5.2 demonstrates the compressor function above. Indeed, we assume the compressor function $C(e)$ as a cumulative distribution function (CDF) therefore 0.25 and 0.75 on the y-axis in Figure 5.2 are related to Q_1 and Q_3, respectively. It is important to mention that we are making following assumption: since we work with data stream and we expect error samples to get so close to $e = 0$ and oscillate around it over time (ideally achieve $e = 0$ over time), the median of error samples is assumed 0. The following describes how this non-uniform quantizer works. y-axis in Figure 5.2 is uniformly divided to quantization bins and each new error sample e_n is compressed as $C(e_n)$, and is put in one of these bins. A counter is assigned to each of the bins denoting the number of error samples contained in that bin and bins below. Once a new error sample is available, these counters are updated and new Q_1 and Q_3 are obtained. Consequently, instead of storing all error samples and sorting them in each adaptation step to find Q_1 and Q_3, we only need to use bin counters. In the following we summarize our running Q_1 and Q_3 estimation technique in more details:

- For the first M data samples (in our problem error samples), simply store and sort them to find Q_1 and Q_3.

- Afterwards, adjust α_1 and α_2 in aforementioned compressor function accordingly,

$$\begin{aligned} 0.25 &= \frac{1}{1+\exp\left(-\alpha_1 Q_1\right)}, \\ \Longrightarrow \alpha_1 &= \frac{-\ln(3)}{Q_1}, \end{aligned} \tag{5.5}$$

and similarly,

$$\begin{aligned} 0.75 &= \frac{1}{1+\exp\left(-\alpha_2 Q_3\right)}, \\ \Longrightarrow \alpha_2 &= \frac{\ln(3)}{Q_3}, \end{aligned} \tag{5.6}$$

where ln stands for natural logarithm.

- We know in our online regression problem that, first, error samples are expected to get closer and closer to $e = 0$ over time. This emphasizes that quantization precision is important around $e = 0$. Second, given the first M error samples we already have an understanding about the range of error samples. According to these two points we can set an ε that denotes the maximum acceptable quantization error around $e = 0$. Note that each error sample e is quantized as follows:

$$e_q = \Delta . \left\lfloor \frac{C(e)}{\Delta} \right\rfloor,$$

where $\lfloor x \rfloor$ and Δ denote the largest integer less than or equal to x and quantization step size, respectively. Now we obtain two quantization step sizes below based on the selected ε:

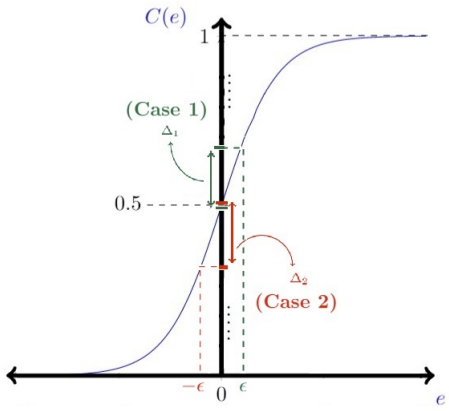


Figure 5.3: Δ_1 and Δ_2 obtained in case 1 ($C = 0.5$ is one of the quantization levels) and case 2 ($C = 0.5$ is not a quantization level), respectively, based on the maximum acceptable quantization error ε around $e = 0$.

1. If $C = 0.5$ is one of the quantization levels then,

$$\Delta_1 = C(\varepsilon) - 0.5$$
$$\Rightarrow \Delta_1 = \frac{1}{1+\exp(-\alpha_2\varepsilon)} - 0.5.$$

2. If $C = 0.5$ is not a quantization level then,

$$\Delta_2 = 0.5 - C(-\varepsilon)$$
$$= 0.5 - \frac{1}{1+\exp(-\alpha_1(-\varepsilon))}$$
$$\Rightarrow \Delta_2 = 0.5 - \frac{1}{1+\exp(\alpha_1\varepsilon)}.$$

An illustration of above cases 1 and 2 is shown in Figure 5.3. Finally, step size of the quantizer is selected as,

$$\Delta = \min\{\frac{1}{M}, \Delta_1, \Delta_2\},$$

and number of quantization levels (or bins) is equal to:

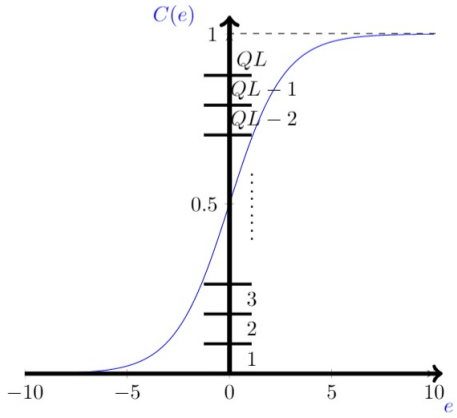


Figure 5.4: Counters of quantization bins are assigned a number from 1 to QL.

$$QL = \left\lceil \frac{1}{\Delta} \right\rceil,$$

where $\lceil x \rceil$ is the least integer greater than or equal to x.

- Counters of quantization bins are given a number from 1 to QL as depicted in Figure 5.4. In addition, number of error samples are set to QL.

- Once a new error sample e_{new} is available, it is quantized and the counters related to the bin number $\left\lfloor \frac{C(e_{new})}{\Delta} \right\rfloor + 1$ and all bins above are incremented. Now, for each bin we compute the percentage of error samples that are contained in this bin and bins below as:

$$\text{Bin}_i = \frac{\text{counter}_i}{\text{number of samples}},$$

where $i \in \{1, 2, \cdots, QL-1, QL\}$ is bin index. Having the number above for each bin, we can update Q_1 and Q_3 after receiving every new error sample. Note that as error samples are getting closer to $e = 0$ over time we expect that bins related to Q_1 and Q_3 also get closer to the bin related to $e = 0$, i.e., the bin that contains $C(0) = 0.5$. To obtain updated Q_1, we calculate $\left\lceil \frac{\text{Bin}_i}{0.25} \right\rceil$ and find the maximum index among all bins for which $\left\lceil \frac{\text{Bin}_i}{0.25} \right\rceil = 1$.

Let this index be found as I_1, then $Q_{1,new}$ is calculated as follows:

$$\Delta.(I_1-1) = C(Q_{1,new}) = \frac{1}{1+\exp(-\alpha_1.Q_{1,new})}$$
$$\Longrightarrow Q_{1,new} = -\frac{\ln\left(\frac{1}{\Delta.(I_1-1)}-1\right)}{\alpha_1}. \tag{5.7}$$

Similarly, we can obtain Q_3 as follows. Calculate $\left\lfloor \frac{\text{Bin}_i}{0.75} \right\rfloor$ and find the minimum index among all bins for which $\left\lfloor \frac{\text{Bin}_i}{0.75} \right\rfloor = 1$. Assume this index is I_3, then after similar calculations $Q_{3,new}$ is obtained as follows:

$$Q_{3,new} = -\frac{\ln\left(\frac{1}{\Delta.(I_3-1)}-1\right)}{\alpha_2}.$$

Remark 5.1: Note that error samples become smaller over time and are mostly around $e = 0$, therefore quantization precision becomes more important for error samples close to $e = 0$. Moreover, we know α_1 and α_2 play significant role in quantization precision. Consequently, we can improve our technique in a way that when error samples approach to $e = 0$ we update α_1 and α_2. To this end, whenever $C(Q_1)$ and $C(Q_3)$ are at a specific distance of $C(0) = 0.5$, parameters α_1 and α_2 are updated. Indeed, when bin index related to Q_1 is equal to,

$$I_{1,c} = \left\lfloor \frac{0.5}{\Delta} \right\rfloor - \lfloor \beta.QL \rfloor,$$

where we specifically consider $\beta \in (0, 0.2)$ to make sure that $I_{1,c}$ is a valid bin index, then we

should update α_1 as follows:

$$\begin{aligned}
\Delta.\,(I_{1,c}-1) &= C(Q_{1,new}) \\
\overset{(5.7)}{\Longrightarrow} Q_{1,new} &= -\frac{\ln\left(\frac{1}{\Delta.(I_{1,c}-1)}-1\right)}{\alpha_1}, \\
\overset{(5.5)}{\Longrightarrow} \alpha_{1,new} &= -\frac{\ln(3)}{Q_{1,new}}.
\end{aligned}$$

Similarly, when bin index related to Q_3 is equal to,

$$I_{3,c} = \left\lfloor \frac{0.5}{\Delta} \right\rfloor + \lfloor \beta.QL \rfloor,$$

new α_2 is obtained as follows:

$$\begin{aligned}
Q_{3,new} &= -\frac{\ln\left(\frac{1}{\Delta.(I_{3,c}-1)}-1\right)}{\alpha_2}, \\
\overset{(5.6)}{\Longrightarrow} \alpha_{2,new} &= \frac{\ln(3)}{Q_{3,new}}.
\end{aligned}$$

Note that if bin index related to Q_1 (Q_3) is equal to $I_{1,c}$ ($I_{3,c}$), we update both α_1 and α_2 and reset number of error samples to QL and assign the counters of quantization bins a number from 1 to QL as depicted in Figure 5.4.

The whole above process of finding first and third quartiles of error PDF can be viewed as estimating the cumulative distribution function (CDF) of the error (under some assumptions) first and then obtaining these quartiles from this CDF. Therefore, at each time instant we have the estimates of first and third quartiles of the error PDF. The algorithm of the proposed error samples running quartile estimation technique is shown in Algorithm 3. This algorithm is much more efficient than order statistics because there is no need to store all previous data samples, and also complexity of this operation is not $O(n)$.

Algorithm 3 Our Proposed Error Samples Running Quartile Estimation Technique

Input: $\{e_n\}$
Outputs: $Q_{1,n}$ and $Q_{3,n}$ (first and third quartiles at time instant n)
Initialisation : M and β
1: **if** ($n <$ M) **then**
2: Sort $\{e_1, e_2, \cdots, e_n\}$ and obtain $Q_{1,n}$ and $Q_{3,n}$
3: **else if** ($n =$ M) **then**
4: Sort $\{e_1, e_2, \cdots, e_M\}$ and obtain $Q_{1,M}$ and $Q_{3,M}$
5: Calculate α_1 and α_2:

$$\alpha_1 = \frac{-\ln(3)}{Q_{1,\mathrm{M}}} \quad \text{and} \quad \alpha_2 = \frac{\ln(3)}{Q_{3,\mathrm{M}}}$$

6: Select ε
7: Calculate Δ_1 and Δ_2:

$$\Delta_1 = \frac{1}{1+\exp(-\alpha_2 \varepsilon)} - 0.5 \quad \text{and} \quad \Delta_2 = 0.5 - \frac{1}{1+\exp(\alpha_1 \varepsilon)}$$

8: Select step size of the quantizer:

$$\Delta = \min\{\frac{1}{M}, \Delta_1, \Delta_2\}$$

9: Set NES (number of error samples) = QL (number of quntization levels or bins) where QL=$\lceil \frac{1}{\Delta} \rceil$
10: Create a vector $\mathrm{COUNTER}_{1\times QL}$ containing counters to be updated with initialisation $\mathrm{COUNTER}(i) = i,\ i = 1, 2, \cdots, QL$
11: Calculate $I_{1,c}$ and $I_{3,c}$:

$$I_{1,c} = \left\lfloor \frac{0.5}{\Delta} \right\rfloor - \lfloor \beta.QL \rfloor \quad \text{and} \quad I_{3,c} = \left\lfloor \frac{0.5}{\Delta} \right\rfloor + \lfloor \beta.QL \rfloor$$

12: **else**
13: $\mathrm{COUNTER}(j) = \mathrm{COUNTER}(j) + 1$ for $j \geq \left\lfloor \frac{C(e_{new})}{\Delta} \right\rfloor + 1$
14: Compute the percentage of error samples contained in each bin i and bins below it:

$$\mathrm{Bin}_i = \frac{\mathrm{COUNTER}(i)}{\mathrm{NES}}$$

15: $\mathrm{NES} = \mathrm{NES} + 1$
16: Find $I_1 = \underset{\mathrm{i}}{\mathrm{argmax}} \left(\left\lceil \frac{\mathrm{Bin}_i}{0.25} \right\rceil == 1 \right)$ and $I_3 = \underset{\mathrm{i}}{\mathrm{argmin}} \left(\left\lfloor \frac{\mathrm{Bin}_i}{0.75} \right\rfloor == 1 \right)$
17: Calculate $Q_{1,n}$ and $Q_{3,n}$:

$$Q_{1,n} = -\frac{\ln\left(\frac{1}{\Delta.(I_1-1)} - 1\right)}{\alpha_1} \quad \text{and} \quad Q_{3,n} = -\frac{\ln\left(\frac{1}{\Delta.(I_3-1)} - 1\right)}{\alpha_2}$$

18: **if** ($I_1 == I_{1,c}$ or $I_3 == I_{3,c}$) **then**
19: Go back to line 5 and update α_1 and α_2 (replace $Q_{1,M}$ and $Q_{3,M}$ with $Q_{1,n}$ and $Q_{3,n}$, respectively)
20: Go to lines 9 and 10
21: **else**
22: Continue
23: **end if**
24: **end if**
25: **return** $Q_{1,n}$ and $Q_{3,n}$ =0

Moreover, the combination of the MCC algorithm and the running quartile estimation technique described in Algorithm 3 for online linear regression is shown in Algorithm 4 in which our running quartile estimation technique is employed to detect and reject major outliers.

Algorithm 4 Combination of Our Proposed Running Quartile Estimation Technique and MCC for Online Linear Regression

Inputs: $\{\mathbf{x}_n, d_n\}$
Output: $\mathbf{w}_n$
Initialisation : μ, σ and $\mathbf{w}_0 = \mathbf{0}$
for each iteration n **do**
  $e_n = d_n - \mathbf{x}_n^T \mathbf{w}_{n-1}$
  Use our running Q_1 and Q_3 estimation technique
  $IQR = Q_3 - Q_1$
  Upper Extreme $= Q_3 + 3 \times IQR$
  Lower Extreme $= Q_1 - 3 \times IQR$
  if (Lower Extreme $\leq e_n \leq$ Upper Extreme) **then**
    $f(e_n) = 1$
  else
    $f(e_n) = 0$
  end if

$$\mathbf{w}_n = \mathbf{w}_{n-1} + \frac{\mu}{\sigma^2} f(e_n) \exp\left(\frac{-e_n^2}{2\sigma^2}\right) e_n \mathbf{x}_n$$

end for
return $\mathbf{w}_n = 0$

5.3 Simulation Results

In this section, we show how our method outperforms other methods in term of steady-state misalignment. Impulsive noise environment is modeled again as in [65, 77, 78]. Input samples x_i are drawn as $x_i \sim \mathcal{N}(0,1)$ and optimum weight vector of unknown system is generated randomly as a unit vector $\mathbf{w}_{opt} \in \mathscr{R}^L$ where $L = 5$. We consider $v_n \sim \mathcal{N}\left(0, \sigma_{v,n}^2\right)$ as white Gaussian noise and η_n as impulsive measurement noise, and model desired signal (or label) at time instant n as,

$$d_n = \mathbf{x}_n^T \mathbf{w}_{opt} + v_n + \eta_n.$$

Assume 30dB signal to white Gaussian measurement noise ratio calculated as follows:

$$\text{SNR} = 10\log_{10}\left(\frac{E\left\{\left[\mathbf{x}_n^T \mathbf{w}_{opt}\right]^2\right\}}{\sigma_{v,n}^2}\right).$$

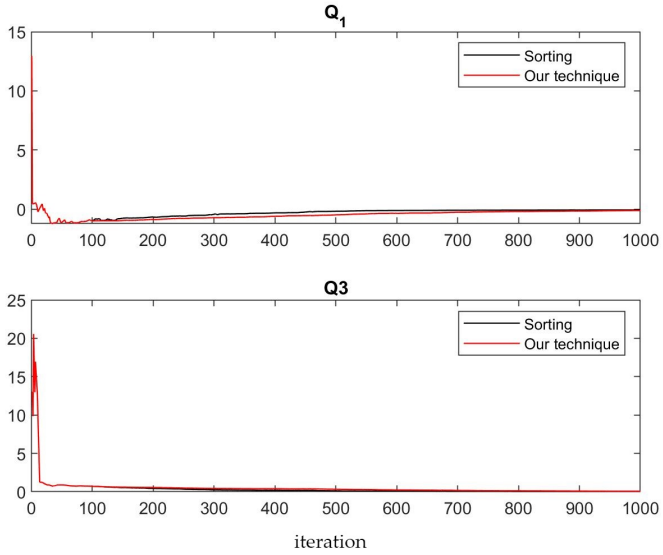


Figure 5.5: Running Q_1 and Q_3 estimation using sorting and our technique.

Impulsive measurement noise is modeled as $\eta_n = \beta_n \omega_n$ in which $\beta_n \sim Bernoulli(p)$ and p is probability of existence of impulses in noise (probability of success) and,

$$\omega_n \sim \mathcal{N}\left(0, 1000E\left\{\left[\mathbf{x}_n^T \mathbf{w}_{opt}\right]^2\right\}\right).$$

Assume $p = 0.2$ in our simulations. $\mathbf{w}_n$ is obtained at each time instant n, and misalignment is calculated based on the NMSD as follows:

$$\text{misalignment}_n = 10\log_{10}\left(\frac{\| \mathbf{w}_n - \mathbf{w}_{opt} \|^2}{\| \mathbf{w}_{opt} \|^2}\right).$$

First, skip filter $f(e_n)$ and consider (5.3) with $\mu = 0.01$ and $\sigma = 0.8$. We run (5.3) and obtain error samples for 1000 iterations. Figure 5.5 shows how sorting and our technique perform almost the same in obtaining Q_1 and Q_3 of error samples. We set $M = 100$ and $\varepsilon = 0.01$ in our technique.

In Figure 5.6 we show the performance of our algorithm (5.4) under different values of

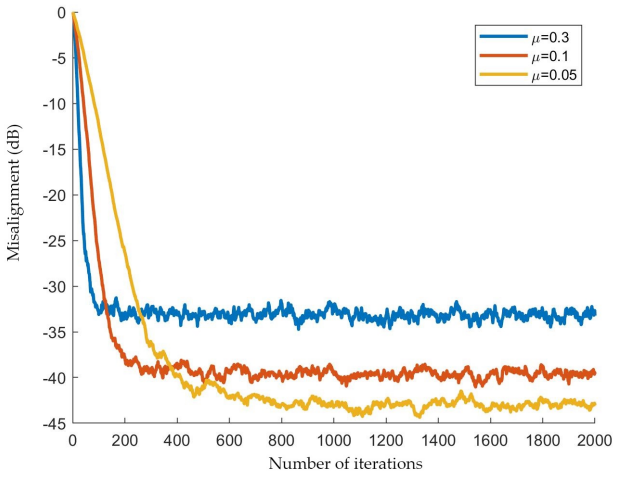


Figure 5.6: Learning curves of our algorithm (5.4) for different values of μ ($\sigma = 1.5$).

learning rate μ. As we expected larger learning rate results in a faster convergence to a higher steady-state misalignment. These learning curves are obtained by averaging over 50 independent trials.

Finally, Figure 5.7 shows that our algorithm outperforms (from misalignment point of view) previous algorithms, even VKW-MCC for which kernel bandwidth σ is also updated in each iteration. These learning curves are obtained by averaging over 10 independent trials. Kernel bandwidth and learning rate are set to $\sigma = 3$ and $\mu = 0.01$, respectively. Moreover, we set $M = 100$ and $\varepsilon = 0.01$ in our algorithm. Note that again although stopping learning process when major outlier occurs seems to cause slower convergence, we could use a hybrid method, like our what we did before to combine our algorithm with fast algorithms such as RMCC. Figure 5.7 also shows that LMS algorithm (which is based on second order statistics of error) diverges to a high steady-state misalignment when there are outlier error samples (or equivalently when impulses occur in the noise).

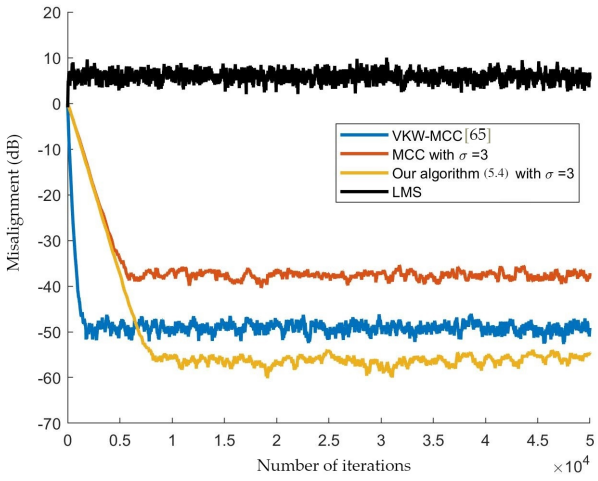


Figure 5.7: Learning curves of different algorithms with $\mu = 0.01$.

5.4 Conclusion

In this chapter we address the online linear regression problem in the presence of non-Gaussian noise. We propose an efficient method to obtain running quartiles of the error samples. This method does not need to store and sort all error samples in each iteration. Using this method alongside maximum correntropy criterion, we reject major outliers and achieve a lower steady-state misalignment compared to previous algorithms.

In the next chapter, we start talking about the other information-theoretic cost function, i.e., EEC in more details.

Chapter 6

Challenging MEEF

6.1 Introduction

Thus far, we discussed only error correntropy as an alternative to MSE in non-Gaussian environments where distributions may be for instance multi-modal, impulsive, or heavy-tailed. Outlier effect is one of the main challenges that we must deal with in such environments. From now on, we will focus on quadratic Renyi's error entropy as the other robust information-theoretic cost function. In general, since entropy measures uncertainty about a system, we minimize it to learn the underlying unknown system.

Although an outlier may have either a negative or positive role [72], outliers in our problem are not informative and arose from non-Gaussian measurement noise. Throughout the chapter, we mitigate this outlier effect in a non-Gaussian environment. We consider an online linear regression or linear adaptive filtering again in which we receive new data samples at each time instant and use it to update the parameters of the underlying system. Our goal is to minimize the error between linear system output and desired responses even when environment is severely affected by outliers. To this end, we employ entropy. As entropy denotes the average dispersion of data, we minimize it to concentrate the errors [23]. In other words, when we use error entropy

as cost function and minimize it we indeed attempt to ideally set the distribution of error as an impulse. However, as error entropy minimization is shift-invariant we must take some further steps to concentrate errors specifically around $e = 0$.

Therefore, locating error samples around the origin is a must in learning based on MEE inasmuch as entropy is shift-invariant and to this end, the first idea that comes into the mind is to add a constraint to error entropy minimization to set mean of the error to zero. This is usually done by adding the error sample mean to the output of the system, however estimate of error mean from error samples is not always straightforward for non-Gaussian distributions [1]. Indeed, if we have n independent and identically distributed (i.i.d.) random variables X_i with finite mean μ and finite variance σ_x^2, classical CLT implies that:

$$\frac{X_1 + X_2 + \cdots + X_n}{n} \stackrel{d}{=} \mu + \frac{Z}{\sqrt{n}} + \frac{o(\sqrt{n})}{n},$$

where $Z \sim \mathcal{N}(0, \sigma_x^2)$ and $Y_1 \stackrel{d}{=} Y_2$ means random variables Y_1 and Y_2 have the same distribution. This result gives a more precise expression for $\frac{X_1+X_2+\cdots+X_n}{n}$ compared to classical law of large numbers (LLN) and is based on the assumptions that random variables are i.i.d. and they have finite mean and variance. However, assume that mean and/or variance are not finite, then this result does not hold and $\frac{X_1+X_2+\cdots+X_n}{n}$ cannot converge to a Gaussian distribution inasmuch as Gaussian distribution is defined based on finite mean and finite variance. Therefore a generalized version of central limit theorem exists and states that there are an entire class of distributions as candidate limiting distributions where Gaussian is one of them [2], hence sample mean approximation of actual mean is not accurate for all distributions and because of this drawback, another approach called MEEF is employed in order to locate error samples around the origin in which some artificial zero error samples (or fiducial points) are added to the learning process [57]. The logic behind this approach is that since entropy minimization tries to bring error samples

closer to each other in each learning step, adding some artificial zero error samples will force all error samples to get closer to the origin.

Although this approach is the most celebrated one for locating error PDF in vicinity of the origin in MEE-based learning and has been used in recent works [24, 28], in this chapter we challenge MEEF and show in an example that it can even result in poorer performance compared to the approach of adding error sample mean to the system output.

First, lets review briefly MCC and MEE. Recall that error statistics is in general unknown. Correntropy is defined as follows:

$$v(E) = \mathbf{E}\{G_\sigma(E)\}, \quad \text{where} \quad E = D - Y$$

in which D and Y are random variables corresponding to desired response and system output, respectively. Correntropy is estimated online by sample mean at time instant n from N error samples as:

$$\hat{v}^{(n)}(E) = \frac{1}{N}\sum_{i=0}^{N-1} G_\sigma(e_{n-i}).$$

Recall that correntropy is a similarity measure, therefore above online cost function is maximized.

Quadratic Renyi's entropy is defined as:

$$H_2(E) = -\log \int p^2(e)de,$$

where $I_2(E) \triangleq \mathbf{E}\{p(E)\}$ is called information potential and since $\log(.)$ is a monotonically increasing function, information potential maximization is equivalent to entropy minimization,

i.e.,:

$$\max_{\mathbf{w}} I_2(E) = \min_{\mathbf{w}} H_2(E).$$

Using sample mean and Parzen PDF estimation, we obtain online estimate of information potential at time instant n from past N error samples as follows:

$$\begin{aligned}\hat{I}_2^{(n)}(E) &= \frac{1}{N}\sum_{i=0}^{N-1}\hat{p}^{(n)}(e_{n-i}) \\ &\overset{(2.4)}{=} \frac{1}{N^2}\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}G_\sigma(e_{n-i}-e_{n-j}).\end{aligned} \tag{6.1}$$

As seen in the (6.1), maximum of $\hat{I}_2^{(n)}(E)$ occurs when all error samples are equal which is consistent with the purpose of entropy minimization.

We use stochastic gradient ascent to maximize the cost function. In stochastic algorithms gradient of the online estimate of cost function J at time instant n and from past N error samples is used (here J is either information potential or correntropy). We indeed use following stochastic update rule:

$$\mathbf{w}_n = \mathbf{w}_{n-1} + \alpha\nabla\hat{J}^{(n)}(\mathbf{w}_{n-1}), \tag{6.2}$$

where $\mathbf{w}_n$ is unknown system parameter vector estimated at time instant n and α denotes learning rate. It is worth reminding that although stochastic algorithms cannot generally optimize a cost function precisely, they are very fast which is very important in the case of extremely large data-set and also they can get close enough to the optimum [56]. Recall that error correntropy maximization and error entropy minimization are called MCC and MEE, respectively.

In the following section, adding fiducial points to MEE in order to locating error

samples around the origin is challenged. As discussed earlier, although this approach is the most celebrated one for locating error PDF in vicinity of the origin in MEE-based learning, in the rest of this chapter we challenge MEEF and show in an example that it can even result in poorer performance compared to the approach of adding error sample mean to the system output.

6.2 What is wrong with MEEF?

As discussed above, relation (6.1) is maximized when all error samples are equal. In other words, the MEE-based learning strives to concentrate error samples in each time instant n using (6.2). It means adding some fiducial zero error samples will give more weight to the origin and consequently other error samples are absorbed toward $e = 0$ in each update. Therefore, the MEEF online cost function is as follows:

$$\begin{aligned}\hat{J}_F^{(n)}(E) =& \frac{1}{(N+M)^2}\sum_{i=0}^{N+M-1}\sum_{j=0}^{N+M-1} G_\sigma(e_{n-i}-e_{n-j}) \\ =& \frac{1}{(N+M)^2}\sum_{i=0}^{N-1}\sum_{j=0}^{N-1} G_\sigma(e_{n-i}-e_{n-j}) \\ &+ \frac{2M}{(N+M)^2}\sum_{i=0}^{N-1} G_\sigma(e_{n-i}) + \frac{M^2}{(N+M)^2} G_\sigma(0),\end{aligned}$$

where $e_{n-N}, e_{n-(N+1)}, \cdots, e_{n-(N+M-1)}$ denote M fiducial points and N is the number of actual error samples. Hence, the MEEF cost function is equivalent to a weighted combination of those of MCC and MEE as follows :

$$\hat{J}_F^{(n)}(E) \propto \frac{N^2}{(N+M)^2}\hat{I}_2^{(n)}(E) + \frac{2MN}{(N+M)^2}\hat{v}^{(n)}(E).$$

Maximizing the MEEF cost function has been interpreted as simultaneously considering entropy minimization (which forces error samples to concentrate) and correntropy maximization (which

forces this concentration to be around the origin) [28, 57].

Despite the claimed benefit of incorporating fiducial points into the MEE-based learning, it can be destructive if used in an environment with a noise for which MEE outperforms MCC significantly. Note that although to the best of our knowledge there is no analytical result that shows superiority of MEE over MCC, MEE is expected to outperform MCC in general in the sense that it gets closer to the desired solution as shown in many experiments like [34] and [81]. The reason for this superiority is the fact that entropy gives weights to different error samples based on their own error PDF (not based on a predetermined kernel as in correntropy definition) and it incorporates all higher order statistics of error regardless of the kernel type that is going to be used for Parzen PDF estimation. Nevertheless, noise statistics is unknown and we do not know whether learning performances of MCC and MEE under this noise are significantly different or not. As stated in [34] there is a distance between MCC and MEE cost functions obtained as follows by expanding Euclidean distance between error PDF and Gaussian kernel:

$$\begin{aligned} D_{ED}\left(p(e),G_\sigma(e)\right) &= \int \left[p(e)-G_\sigma(e)\right]^2 de \\ &= \overbrace{\int p^2(e)de}^{I_2(E)} + \overbrace{\int G_\sigma^2(e)de}^{\frac{1}{2\sigma\sqrt{\pi}}} - 2\overbrace{\int p(e)G_\sigma(e)de}^{v(E)} \\ &\Longrightarrow I_2(E) + \frac{1}{2\sigma\sqrt{\pi}} = 2v(E) + D_{ED}\left(p(e),G_\sigma(e)\right). \end{aligned} \tag{6.3}$$

Clearly for a fixed kernel bandwidth σ and also zero Euclidean distance between error PDF and Gaussian kernel we have $v(E) \propto I_2(E)$ which means MCC and MEE result in the same solution. However, in order to utilize information-theoretic cost functions more efficiently the kernel bandwidth σ is also usually updated in each time instant, for instance by using Kullback-Leibler divergence [61], which means that σ is also a function of the system parameter and is not fixed. Although in this chapter we assume a fixed σ for sake of simplicity, even in this case the

Euclidean distance is still another term in (7.1) which is a function of system parameter. The reason for that is evolution of error PDF $p(e)$ in each time instant which means that $p(e)$ is also a function of system parameter . This is shown in the following. First, note that in linear adaptive filtering, label at time instant n is shown as $d_n = \mathbf{x}_n^T \mathbf{w}_{opt} + \nu_n$ where ν_n denotes random noise at time instant n which is independent of input vector $\mathbf{x}_n$. Moreover, $\mathbf{w}_{opt}$ is parameter vector of the system which is unknown and is going to be learned. Error at time instant n is calculated as:

$$\begin{aligned} e_n = d_n - \mathbf{x}_n^T \mathbf{w}_{n-1} &= \mathbf{x}_n^T (\mathbf{w}_{opt} - \mathbf{w}_{n-1}) + \nu_n \\ &= \tilde{y}_n + \nu_n, \end{aligned} \tag{6.4}$$

where $\tilde{y}_n \triangleq \mathbf{x}_n^T (\mathbf{w}_{opt} - \mathbf{w}_{n-1})$. Let elements of input vector $\mathbf{x}_n$ be i.i.d. with PDF $p_X(x)$ and ν_n has PDF $p_N(\nu)$ where sample space of all random variables is the set of real numbers. PDF of $\tilde{y}_n$ at time instant n is obtained as follows:

$$\begin{aligned} p_{\tilde{Y}}^{(n)}(\tilde{y}) = &\left(\prod_{i=1}^{L} \left| \frac{1}{\mathbf{w}_{opt} - \mathbf{w}_{n-1}} \right| \right) p_X \left(\frac{\tilde{y}}{(\mathbf{w}_{opt} - \mathbf{w}_{n-1})_{(1)}} \right) \\ &* p_X \left(\frac{\tilde{y}}{(\mathbf{w}_{opt} - \mathbf{w}_{n-1})_{(2)}} \right) * \cdots * p_X \left(\frac{\tilde{y}}{(\mathbf{w}_{opt} - \mathbf{w}_{n-1})_{(L)}} \right), \end{aligned} \tag{6.5}$$

where $(\mathbf{w}_{opt} - \mathbf{w}_{n-1})_{(i)}$ is the i^{th} element of vector $\mathbf{w}_{opt} - \mathbf{w}_{n-1}$ and $*$ denotes convolution operation. This PDF is a function of time n because of its dependence to the estimated parameter vector $\mathbf{w}_{n-1}$. Since $\tilde{y}$ and ν are independent, error PDF $p(e)$ at time instant n can be obtained from (6.4) as follows:

$$p^{(n)}(e) = p_{\tilde{Y}}^{(n)}(e) * p_N(e), \tag{6.6}$$

and from (6.5) and (6.6) it is concluded that error PDF is also evolving with time and is a

function of the estimated parameter vector $\mathbf{w}_{n-1}$ (or equivalently a function of time n). Therefore, the Euclidean distance between error PDF and Gaussian kernel in (7.1) is a function of vector parameter as well. This emphasizes that even for a fixed kernel bandwidth σ, correntropy and entropy may not be equivalent cost functions (i.e., their estimates of parameter vector may be different). We can also say that if the Euclidean distance between **noise** PDF $p_N(v)$ and Gaussian kernel is not negligible, then we expect a considerable Euclidean distance between **error** PDF $p(e)$ and Gaussian kernel as well inasmuch as $p_N(v)$ is the optimum evolution of error PDF $p(e)$ as shown in the following. Assume the algorithm gets close enough to the desired solution, i.e., $|\mathbf{w}_{opt}-\mathbf{w}_{n-1}|\leq \varepsilon_n$ such that $\varepsilon_n \to 0$ for $n \geq \eta$. Then, we have the following based on Cauchy-Schwarz inequality:

$$
\begin{aligned}
&|\tilde{y}_n| = |\mathbf{x}_n^T(\mathbf{w}_{opt}-\mathbf{w}_{n-1})| \leq |\mathbf{x}_n||\mathbf{w}_{opt}-\mathbf{w}_{n-1}| \\
&\Rightarrow \tilde{y}_n \to 0 \quad \text{for} \quad n \geq \eta \\
&\Rightarrow p_{\tilde{Y}}^{(n)}(\tilde{y}) \to \delta(\tilde{y}) \quad \text{for} \quad n \geq \eta \\
&\overset{(6.6)}{\Rightarrow} p^{(n)}(e) \to p_N(e) \quad \text{for} \quad n \geq \eta.
\end{aligned}
$$

Note that a considerable Euclidean distance between error PDF and Gaussian kernel results in a considerable difference between MCC and MEE cost functions as seen in (7.1) and therefore it is more likely that MCC and MEE show different performances. Authors is [34] show in an illustrative example that when $D_{ED}\left(p_N(e), G_\sigma(e)\right)$ increases the difference between performances of MEE and MCC also increases.

Therefore learning a linear system based on MCC and MEE in any environment corrupted by a noise for which $D_{ED}\left(p_N(e), G_\sigma(e)\right)$ is not negligible may result in different estimates. However, which one is closer to the desired solution in such environments? Although many previous experimental results have shown the superiority of MEE performance, to the best

of our knowledge no analytical proof exists so far. Nevertheless, it is not surprising if one expects MEE to outperform MCC in such environments for the following reasons:

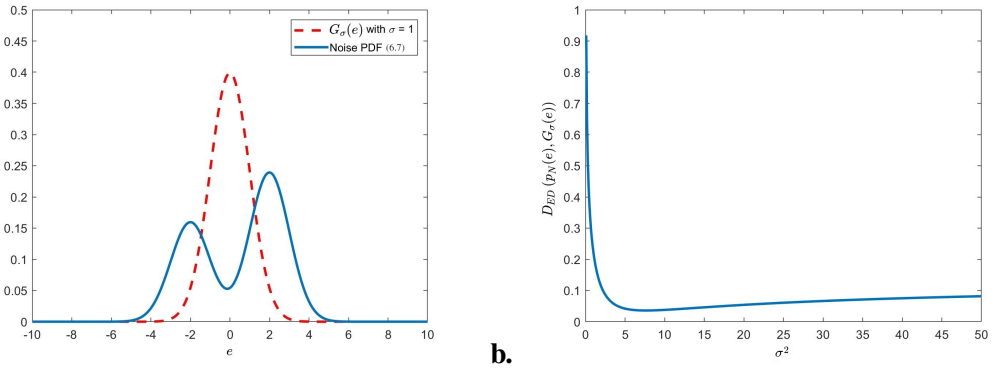
- First, information potential (i.e., $I_2(E) = \mathbf{E}\{p(E)\}$) as the cost function of MEE gives weight to error samples based on the PDF of error. It means that if an error sample is highly probable, even if it is large, it is not considered as an outlier which is right. On the other side, correntropy (i.e., $v(E) = \mathbf{E}\{G_\sigma(E)\}$) as the cost function of MCC gives weight to error samples based on a *predetermined* kernel function while noise statistics is unknown, therefore there is a possibility that some highly occurring error samples take small weights by correntropy and are NOT considered significant during the learning process.

- Second, information potential incorporates all higher order moments of the error [1] while for correntropy this highly depends on type of the kernel function. For instance if Gaussian kernel is selected (which is usually the case because of its properties), then by using Taylor expansion of exponential function we can see only even-order moments of error are incorporated in correntropy.

Based on the above facts, quadratic Renyi's error entropy seems to be a more comprehensive descriptor of the error compared to correntropy, and therefore we expect a better performance for MEE in estimating the system parameter when environments are corrupted by a noise under which the performances of MCC and MEE differ. As an example that strengthens our expectation of superiority of MEE over MCC in such environments, consider following mixture of two Gaussians as noise distribution:

$$p_N(v) = 0.4p_G(v;-2,1) + 0.6p_G(v;2,1), \tag{6.7}$$

where $p_G(v;\mu,\sigma^2)$ denotes a Gaussian PDF with mean μ and variance σ^2. The Euclidean



a. **b.**

Figure 6.1: Gaussian kernel with $\sigma = 1$ and mixture of two Gaussians (6.7) (figure a). Euclidean distance (6.8) between mixture of two Gaussians (6.7) and Gaussian kernel with kernel bandwidth σ as a function of σ^2 (figure b).

distance between this multi-modal noise (Figure 6.1-a) and Gaussian kernel with kernel bandwidth σ is non-zero for any choice of σ which implies that there is a difference between MCC and MEE cost functions as discussed before. Interestingly it is well-known that MCC performance in presence of a multi-modal noise like (6.7) is poor [1] while MEE shows satisfactory performance [82]. This confirms our claim that it is likely to observe different performances from MCC and MEE (where MEE is expected to be superior) when Euclidean distance between the noise and Gaussian kernel is considerable. This Euclidean distance is calculated as follows:

$$\begin{aligned} D_{ED}(p_N(e), G_\sigma(e)) &= \int [p_N(e) - G_\sigma(e)]^2 de \\ &\overset{(a)}{=} 0.1492 + \frac{1}{4\pi\sigma^2} - \frac{2}{\sqrt{2\pi(1+\sigma^2)}} \exp\left(-\frac{2}{1+\sigma^2}\right), \end{aligned} \tag{6.8}$$

where (a) is resulted from the fact that integral of product of two Gaussian PDFs with means μ_1 and μ_2 and variances σ_1^2 and σ_2^2 is equal to [83]:

$$\begin{aligned} &\int p_G(x;\mu_1,\sigma_1^2) p_G(x;\mu_2,\sigma_2^2) dx = \\ &\qquad \frac{1}{\sqrt{2\pi\left(\sigma_1^2+\sigma_2^2\right)}} \exp\left(-\frac{(\mu_1-\mu_2)^2}{2(\sigma_1^2+\sigma_2^2)}\right). \end{aligned} \tag{6.9}$$

Figure 6.1-b shows Euclidean distance (6.8) as a function of σ^2. Therefore, based on aforementioned explanations, expecting a more promising performance from MEE compared to MCC should not be surprising. Lets get back to our main goal in this chapter which is challenging MEEF. As discussed earlier, MEEF is a combination of MEE and MCC, therefore for environments contaminated by noises under which MEE outperforms MCC, combining MCC with MEE will result in weakening MEE performance. Next section shows this shortcoming of MEEF in an example.

6.3 Simulation Results

We consider system model of Figure 2.1 for linear adaptive filtering. For sake of computational simplicity, we drop expectation operator from both $v(E) = \mathbf{E}\{G_\sigma(E)\}$ and $I_2(E) = \mathbf{E}\{p(E)\}$ and obtain following stochastic cost functions:

$$
\begin{aligned}
&v_s(E) = G_\sigma(E) \;\Rightarrow\; \hat{v}_s^{(n)}(E) = G_\sigma(e_n), \\
&I_{2,s}(E) = p(E) \;\Rightarrow\; \hat{I}_{2,s}^{(n)}(E) = p(e_n) \overset{(a)}{\approx} \frac{1}{N}\sum_{i=0}^{N-1} G_\sigma\left(e_n - e_{n-i}\right),
\end{aligned}
$$

where (a) is resulted from Parzen PDF estimation. Similarly, stochastic cost function of MEEF with M fiducial points will be as follows:

$$
\begin{aligned}
\hat{J}_{F,s}^{(n)}(E) &= \frac{1}{N+M}\sum_{i=0}^{N+M-1} G_\sigma\left(e_n - e_{n-i}\right) \\
&= \frac{M}{N+M}\hat{v}_s^{(n)}(E) + \frac{N}{N+M}\hat{I}_{2,s}^{(n)}(E).
\end{aligned}
$$

Let $\hat{J}^{(n)}(E)$ be stochastic cost function of either MCC, MEE or MEEF. Stochastic update rule of (6.2) is used in order to learn an adaptive linear system over time where kernel bandwidth σ and learning rate α are considered fixed for each simulation. Recall that in MEE, error sample

mean is added to system output in each iteration. It is worth restating system model in Figure 2.1. The system output at time instant n is $y_n = \mathbf{x}_n^T \mathbf{w}_{n-1}$, $\mathbf{x}_n = \left[x_n, x_{n-1}, \cdots, x_{n-L+1}\right]^T$ is the input vector to the system at time instant n where $x_i \sim \mathcal{N}(0,1)$, and $\mathbf{w}_{n-1} = \left[w_{n-1}^{(1)}, w_{n-1}^{(2)}, \cdots, w_{n-1}^{(L)}\right]^T$ denotes the system parameters estimated at time instant $n-1$. In addition, label at time instant n is denoted by $d_n = \mathbf{x}_n^T \mathbf{w}_{opt} + v_n$ where $\mathbf{w}_{opt}$ is the unknown system parameter vector to be learned ($\mathbf{w}_{opt} \in \mathscr{R}^L$ is generated randomly in our simulation as a unit vector where $L = 5$) and v_n is the noise. Note that misalignment at time instant n is measured as follows:

$$\text{misalignment}_n = 20\log_{10}\left(\| \mathbf{w}_n - \mathbf{w}_{opt} \|\right).$$

In our simulations we consider exponential noise, i.e., $p_{N,\exp}(v) = \lambda \exp(-\lambda v)$ for $v \geq 0$ and 0 otherwise, for two reasons: first, it is a light-tailed distribution and consequently classical CLT holds for that and sample mean approximation of its mean will not be problematic [2] (i.e., no need to use fiducial point in order to locate error samples around the origin). Second, the Euclidean distance between this PDF and Gaussian kernel is non-zero for any kernel bandwidth σ and large enough λ which indicates the possibility of different learning performances for MCC and MEE as discussed before (indeed it has been shown in [34] that when λ increases the difference between performances of MEE and MCC also increases). To show the latter we obtain the following expression for Euclidean distance between exponential PDF and Gaussian kernel:

$$\begin{aligned} &D_{ED}\left(p_{N,\exp}(e), G_\sigma(e)\right) \overset{(a)}{=} \\ &\frac{\lambda}{2} + \frac{1}{\sqrt{4\pi\sigma^2}} - 2\lambda \exp\left(\frac{\lambda^2\sigma^2}{2}\right) Q(\lambda\sigma), \end{aligned} \tag{6.10}$$

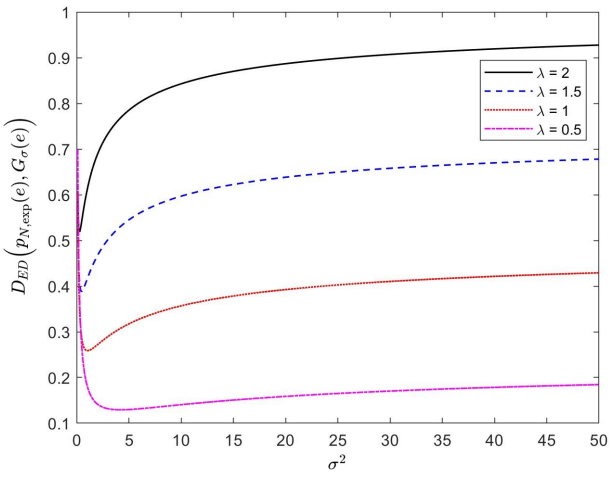


Figure 6.2: Euclidean distance between exponential PDF with parameter λ and Gaussian kernel with kernel bandwidth σ as a function of σ^2.

where $Q(\beta) = \frac{1}{\sqrt{2\pi}} \int_\beta^\infty \exp\left(-\frac{x^2}{2}\right) dx$ is called Q-function and (a) is resulted from (6.9) and the following identity:

$$\int_0^\infty \exp\left(-\frac{1}{2}ax^2 + bx\right) dx = \sqrt{\frac{2\pi}{a}} \exp\left(\frac{b^2}{2a}\right) Q\left(-\frac{b}{\sqrt{a}}\right).$$

Proof of the above identity is straightforward and is left to the reader. Figure 6.2 illustrates Euclidean distance (6.10) as a function of σ^2 for different values of λ. As seen in this Figure, this Euclidean distance is always non-zero for different values of σ and large enough λ which shows a difference between MCC and MEE cost functions under this noise. Interestingly, this difference in cost functions is translated into different performances of MCC and MEE where MEE is superior [34]. Therefore adding fiducial points to MEE will degrade the performance. Following simulation result matches our expectation. In this simulation we consider convergence speed and steady-state misalignment as two key factors to evaluate the performances of MCC, MEE and MEEF under exponential noise. Given a specific learning rate-kernel bandwidth pair, i.e., (α, σ), each algorithm is run and its convergence speed is obtained by i_{conv} which is defined

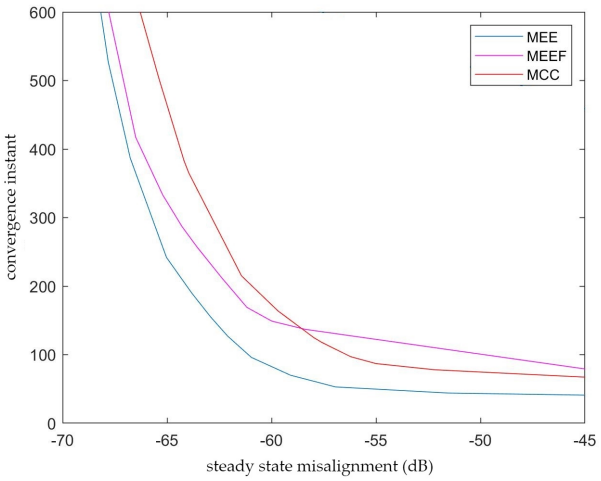


Figure 6.3: Convergence instant vs. steady-state misalignment for MCC, MEE, and MEEF with 1 fiducial point obtained from different pairs of learning rate α and kernel bandwidth σ such that $\alpha \in [0.05 : 0.005 : 0.1]$ and $\sigma \in [0.2 : 0.1 : 1.4]$ (in presence of exponential noise and 50dB SNR).

as the first time instant for which we have the following:

$$\text{misalignment}_{i_{conv}} \leq \text{ steady} - \text{state misalignment } + \ 2,$$

where steady-state misalignment is obtained as the sample mean of the misalignments within the last 200 time instants. We assume 50dB SNR defined as follows:

$$\text{SNR} = 10\log_{10}\left(\frac{E\left\{\left[\mathbf{x}_n^T\mathbf{w}_{opt}\right]^2\right\}}{E\{v_n^2\}}\right),$$

and plot convergence instant (i_{conv}) versus steady-state misalignment achievable bounds for MCC, MEE and MEEF with 1 fiducial point in Figure 6.3. These bounds are obtained from convex hull of all points resulted from pairs (α,σ) such that $\alpha \in [0.05 : 0.005 : 0.1]$ and $\sigma \in [0.2 : 0.1 : 1.4]$. It is worth mentioning that convergence speed and steady-state misalignment for each pair (α,σ) are obtained by averaging learning curves over 50 Monte Carlo simulations. As seen in this figure, MEE outperforms both MCC and MEEF and therefore we conclude that combination of

MEE with MCC as MEEF is even destructive under aforementioned exponential noise which matches our expectation.

Remark 6.1: The main contribution of this chapter is to show that employing fiducial points is not always a viable alternative to adding error sample mean to the system output when learning a system based on MEE. In other words, MEEF can be just as inefficient in locating error PDF at the origin as MEE with error sample mean added to the system output.

6.4 Conclusion

In this chapter we challenge the well-known MEEF algorithm for robust linear adaptive filtering under non-Gaussian noise. This algorithm is indeed a combination of MCC and MEE. We discuss the possibility of observing superiority of MEE over MCC when Euclidean distance between error PDF and Gaussian kernel is not negligible. Therefore combining MEE with MCC as MEEF will degrade MEE performance especially under noise statistics that MCC performs significantly poorer than MEE. As an example to show this drawback of MEEF we illustrate its weak performance in presence of exponential noise which is a noise under which performances of MCC and MEE are different.

As we saw in this chapter MEEF can be just as inefficient in locating error PDF at the origin as MEE with error sample mean added to the system output. It is worth mentioning that we did not propose our alternative to these methods in this chapter, however, we do propose a new algorithm called "*Trimmed MEE*" in the next chapter which shows superior performance compared to both aforementioned methods.

Chapter 7

Trimmed MEE

7.1 Introduction

Recall that both MEE and MCC are deployed in information-theoretic learning context as robust criteria to deal with non-Gaussianity of the environment which becomes problematic if conventional MSE is used. In other words, superiority of information-theoretic cost functions becomes more clear when the error distribution is non-Gaussian and this happens if the filter topology is nonlinear (which was not of interest in this thesis) or the label (or noise in the label) is non-Gaussian (which we do address in this thesis). In this chapter, online linear regression in environments corrupted by non-Gaussian noise is addressed again. We discussed in previous chapters that in such environments there might exist abnormally large error samples (or outliers) which mislead the learning process. The main challenge is how to keep the supervised learning problem least affected by these unwanted and misleading outliers. We also saw in Chapter 6 that an information-theoretic algorithm based on quadratic Renyi's entropy, called minimum error entropy abbreviated to MEE, has been employed in recent years to take on this issue. However, this minimization might not result in a desired estimator inasmuch as entropy is shift-invariant, i.e., by minimizing the error entropy, error samples may not be necessarily concentrated around

zero. In this chapter, we strive to address this problem.

As discussed, entropy as a robust information-theoretic cost function contains all higher-order moments and although the algorithm based on error entropy is computationally more expensive than error correntropy, entropy is a more general descriptor of the underlying error statistics [30]. Indeed, MEE is expected to have a superior performance compared to MCC in general at the cost of higher computational complexity as shown in many experimental results [34,81,84]. The problem of MCC arises from being a local criterion that takes into account mostly the errors within the Gaussian kernel bandwidth, while error modes might in fact be far from the origin. On the other hand, MEE's superior performance emerges from self-adjusting the weights of different error samples based on the error distribution itself. Therefore, MCC may not perform as efficiently as MEE in non-Gaussian noises with a light-tail or multi-modal distribution [1]. Moreover, previous work have tackled computational bottleneck of entropy approximation in large-scale data sets, e.g., [85] and [30], where fast Gauss transform and quantization were employed, respectively, to reduce computational complexity. Some results regarding consistency, robustness, uniqueness of the solution, sufficient and necessary conditions for MEE algorithm can be found in [13] and [86]. Moreover, authors in [82] show that even when large outliers exist in both input and output variables, MEE can result in a very close solution to the optimum value. Some applications of entropy minimization in adaptive system training, neural networks, blind deconvolution, parameter estimation, blind source separation, digital communication channel equalization, and channel estimation for massive multiple input multiple output (MIMO) communication can be found in [3, 16, 87–92]. Bayesian estimation based on MEE is also addressed in [81] and [93].

As we discussed in previous chapter, outlier effect is one of the main challenges that we must deal with when we work with non-Gaussian (multi-modal, impulsive, heavy-tailed, etc.) environments. Although an outlier may have either a negative or positive role, outliers in our

problem are not informative and arose from non-Gaussian measurement noise. Throughout the chapter, we mitigate this outlier effect in a non-Gaussian environment. We consider an online linear regression in which we receive new data samples at each time instant and use it to update the parameters of the underlying system. Our goal is to minimize the error between linear system output and desired responses even when environment is severely affected by outliers. To this end, we minimize entropy as the average dispersion of data to concentrate error samples in the vicinity of origin. Recall that it means we attempt to ideally set the distribution of error as an impulse, however, since error entropy minimization is shift-invariant we must take some further steps to concentrate error samples.

In this chapter, first, we review briefly existing solutions for concentrating error samples around $e = 0$ in MEE. Then, we propose our alternative to these existing solutions and call it *Trimmed MEE* in which we combine our error sample running quartile estimation technique, proposed in Chapter 5 (Algorithm 3), with MEE and use this combination in online linear regression. This helps us to detect major outliers in error samples and mitigate their destructive effect in the learning by not considering them in both processes of MEE-based learning and concentrating error samples around $e = 0$. The whole process improves MEE performance from convergence rate and steady-state misalignment points of view especially when the environment is affected by a non-Gaussian noise with heavier tail than that of Gaussian.

7.2 Combining Outlier Detection with MEE

Before diving into more details about how to combine outlier detection with MEE, recall from previous chapter that the difference between cost functions of MCC and MEE can be obtained by using Euclidean distance between error PDF $p_E(e)$ and Gaussian kernel as

follows [34]:

$$
\begin{aligned}
&D_{ED}\left(p_E(e), G_\sigma(e)\right) = \int \left(p_E(e) - G_\sigma(e)\right)^2 de \\
&= \overbrace{\int \left(p_E(e)\right)^2 de}^{I_2(E)} + \overbrace{\int \left(G_\sigma(e)\right)^2 de}^{\frac{1}{2\sigma\sqrt{\pi}}} - 2\overbrace{\int p_E(e) G_\sigma(e) de}^{v(E)} \\
&\Longrightarrow I_2(E) + \frac{1}{2\sigma\sqrt{\pi}} = 2v(E) + D_{ED}\left(p_E(e), G_\sigma(e)\right). \qquad (7.1)
\end{aligned}
$$

Obviously, there is a difference between MCC and MEE cost functions based on (7.1), and consequently their optimum solution may be different as well. If Gaussian kernel bandwidth σ is fixed (not adaptive) and also Euclidean distance between error PDF and Gaussian kernel is zero, then MCC and MEE are equivalent and result in the same solution. Recall that Gaussian kernel bandwidth σ is a free parameter in both MCC and MEE cost functions that can be optimized during the learning process to increase algorithm efficiency. It indeed determines the magnitude of the weights assigned to each error sample and it is a function of error. Optimizing this bandwidth has been widely addressed in literature, for instance by minimizing Kullback–Leibler divergence between the true and estimated error distribution, using shape of error distribution measured by its kurtosis, using instantaneous error in each iteration, and so forth. In this chapter, we assume a fixed kernel bandwidth for the sake of simplicity.

Now, lets focus on concentrating error samples around $e = 0$ in MEE. We have discussed our expectation of MEE superiority over MCC. However, alongside the higher computational complexity of MEE compared to MCC that has been addressed in previous work as stated earlier, another difficulty associated with MEE is that the error PDF needs to be moved to the origin, as entropy is shift-invariant. Towards that end, the following approaches have been proposed in the literature:

1. Adding sample mean of the labels (sample mean up to time instant n) to the output of

the linear system as a bias term [1]: Although this approach is very simple and works for labels with symmetric PDFs, environments in many real world scenarios are corrupted by asymmetric noises whose tails are heavier than that of the Gaussian distribution. These distributions contain many large outliers and consequently sample mean may be very misleading. In other words, sample mean may fail to converge in probability to the expected value, and law of large numbers does not hold in such environments [94].

2. Minimization of Error Entropy with Fiducial points or MEEF [57]: This approach suggests to consider a fiducial zero vector of arbitrary length M whose elements are indeed points of reference and help to fix the peak of the error PDF at the origin. Consequently, error entropy minimization forces the PDF to approach an impulse around $e = 0$. Now, information potential at each time instant n using past N error samples plus M fiducial points is denoted by $\hat{I}^{(n)}_{2,F}(E)$ and is estimated as follows:

$$\hat{I}^{(n)}_{2,F}(E) = \frac{1}{(N+M)^2} \sum_{i=0}^{N+M-1} \sum_{j=0}^{N+M-1} G_\sigma(e_{n-i} - e_{n-j}),$$

where $[e_{n-N}, e_{n-(N+1)}, \cdots, e_{n-(N+M-1)}] = \underline{\mathbf{0}}_{1\times M}$ denotes fiducial zero vector. Then, above relation can be rewritten as follows:

$$\begin{aligned}\hat{I}^{(n)}_{2,F}(E) = & \frac{1}{(N+M)^2} \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} G_\sigma(e_{n-i} - e_{n-j}) \\ & + \frac{2M}{(N+M)^2} \sum_{i=0}^{N-1} G_\sigma(e_{n-i}) + \frac{M^2}{(N+M)^2} G_\sigma(0). \end{aligned} \tag{7.2}$$

Relation (7.2) can be interpreted as a weighted combination of the error entropy criterion (first term on the right hand side) and the error correntropy criterion (second term on the right hand side). The first term strives to make the error PDF as close as possible to an impulse, while the second term pushes the peak of the error PDF towards $e = 0$.

This approach can outperform the previous one [28, 57]. However, there is an obvious trade-off: As we increase the number of fiducial points, i.e., M, the cost function (7.2) gets closer to that of MCC. Although this will make the role of the correntropy-related term more emphasized, thereby moving the peak of the error PDF towards the origin more aggressively, the accuracy of the entropy estimation would suffer. In other words, as seen in (7.1), this can be problematic and deteriorate the performance of MEE when the difference between MCC and MEE cost functions is not negligible.

Due to the drawbacks of above methods for concentrating error samples around $e = 0$ in MEE, we propose a new approach called *Trimmed MEE*. The key idea behind this proposed method is to stop incorporating abnormally large errors (or major outliers) into the learning process. Recall that major outliers in error samples can be very misleading as they differ significantly from other observations, therefore we strive to exclude them from other error samples. We use our technique in Algorithm 3 in which we employ running quartiles of the error samples in order to detect major outliers.

A major outlier in error samples denotes that either an abnormally large label (significantly different from most of the labels) or an abnormally large noise in label has occurred. In either way, it seems that the benefit of just ignoring this major outlier error sample and not incorporating it in the learning process would be more than using it. This is obvious inasmuch as:

- If the label itself is significantly different from rest of the labels, learning based on that will update the parameters in favor of this major outlier not most of the data, therefore the resultant parameters will not be promising.

- If the label has been corrupted by an abnormally large noise, updating parameters based on this heavily corrupted data sample will be very misleading.

Throughout this chapter, we stop incorporating major outliers into online linear regression

by detecting and eliminating them from MEE using our proposed running quartile estimation technique and finding lower and upper extremes as depicted in Figure 5.1 This can be interpreted as ignoring heavy part of the tail in error PDF and only use lighter part for learning.

Recall that as discussed earlier, simply adding sample mean of labels as a bias to the learned system output or using fiducial points in order to locate error PDF at the origin may be problematic. In the following, we deploy our running quartile estimation technique to address this problem by modifying first method in which we add a *proper* bias to the learned system output. First, we discuss the possible problem of adding a bias in more details here. Recall that outliers can be very misleading in sample mean approximation inasmuch as very large values can significantly shift sample mean approximation away from actual mean in each time instant. Outliers in error samples usually arise from heavy part of the tail of either label noise distribution or label distribution itself. Note that sampling from a distribution with a rather heavy tail results in mostly "normal" values with a few "abnormal" values (outliers). As examples, we can name LogNormal, α-stable, and Weibull which belong to a very well-known class of non-Gaussian distributions with severe heavy tails. Lets investigate this class of non-Gaussian distributions, called heavy-tailed, as an example to show that why sample mean is not always a straightforward method for approximation of actual mean. A heavy-tailed random variable is defined as follows:

Definition 7.1 [2]: Random variable E is said to be heavy-tailed if its CDF $F(e) = P(E \leq e)$ has the following property for all $\mu > 0$:

$$\lim_{e \to \infty} sup \frac{1 - F(e)}{\exp(-\mu e)} = \infty.$$

This means that a distribution is heavy-tailed if its tail is heavier than that of any exponential distribution. For these kind of distributions, law of large numbers (LLN) and central limit theorem (CLT) do not hold and sample mean approximation may not converge to the actual mean [2].

Some moments of heavy-tailed distribution may not exist (even mean and variance), for instance, Cauchy distribution is a heavy-tailed distribution for which neither mean nor variance exist and obviously sample mean approximation does not make sense. Interestingly, even if all moments exist for a heavy-tailed distribution and sample mean converges to actual mean, the convergence will be very slower than that of a light-tailed distribution. This can be easily shown, however first we need the following Lemma to state and prove Corollary 2.

Lemma 1 *[2]: Following two statements are equivalent for random variable E:*

- *E is heavy-tailed*
- $M_E(s) \triangleq E\{e^{sE}\} = \infty$ *for all* $s > 0$

where $M_E(s)$ *denotes moment generating function of E.*

Proof. Provided in [2]. ■

Next, we use above lemma to prove following corollary.

Corollary 2 *Consider n independent heavy-tailed random variables* $E_1, E_2, \cdots, E_n$*, then sum random variable* $S_n = E_1 + E_2 + \cdots + E_n$ *is also a heavy-tailed random variable.*

Proof. Writing the moment generating function of the sum random variable, we have:

$$M_{S_n}(s) \overset{(a)}{=} M_{E_1} \times M_{E_2} \times \cdots \times M_{E_n} \overset{(b)}{=} \infty \quad \text{for all s} > 0,$$

where (a) is due to the independence of random variables $E_1, E_2, \cdots, E_n$ from each other and (b) is because of the fact that these random variables are heavy-tailed and consequently based on Lemma 1 their moment generating function is ∞. Therefore, sum random variable S_n is also heavy-tailed based on Lemma 1. ■

Now, we show even if all moments exist for a heavy-tailed distribution and sample mean

converges to actual mean, the convergence will be very slow. Consider n independent and identically distributed ($i.i.d.$) heavy-tailed random variables $E_i,\ i = 1,2,\cdots,n$, with mean m and variance σ^2. Similar to the proof of the corollary 2 we can show that $Z_n = \frac{S_n - nm}{\sqrt{n}}$ also has a heavy-tailed distribution with mean 0 and variance σ^2. Therefore, any realization z_n of random variable Z_n at each time instant n can be written as follows:

$$z_n = \frac{s_n - nm}{\sqrt{n}} \quad \Longrightarrow \quad \frac{s_n}{n} = m + \frac{z_n}{\sqrt{n}}, \tag{7.3}$$

where s_n is a realization of heavy-tailed distribution S_n with mean nm and variance $n\sigma^2$. $\frac{s_n}{n}$ is sample mean approximation that can be heavily distorted around the true mean m inasmuch as samples from a heavy-tailed distribution Z_n contain very large values. Compare this with what we have for a light-tailed distribution. Consider n $i.i.d.$ light-tailed random variables $E_i,\ i = 1,2,\cdots,n$, with mean m and variance σ^2. Based on CLT for large n we have,

$$\frac{s_n}{n} \approx m + \frac{z}{\sqrt{n}}, \tag{7.4}$$

where s_n and z are realizations of sum random variable S_n with mean nm and variance $n\sigma^2$ and a Gaussian distribution with mean 0 and variance σ^2, respectively. Comparing (7.3) and (7.4) we can see although sample mean $\frac{s_n}{n}$ converges to actual mean m as n goes to ∞ for both of these heavy-tailed and light-tailed scenarios, convergence in light-tailed scenario is much faster. This is because of the fact that samples of a light-tailed distribution never differ significantly from the mean while samples from a heavy-tailed distribution contain very large samples or outliers, therefore $\frac{z}{\sqrt{n}}$ approaches much faster to zero than $\frac{z_n}{\sqrt{n}}$ as n increases.

In above example we saw that for non-Gaussian distributions, approximation of actual mean using sample mean may be problematic. Now, lets get back to our main problem in this

section. It is clear now why adding sample mean of labels to system output as a bias term in order to locate error PDF at the origin may be problematic: this sample mean approximation may not be accurate if label or noise in label has a distribution with heavier tail than that of Gaussian. As discussed above, an important characterization of light-tailed distributions is that the samples are concentrated around the mean of the distribution not far away from that and are mostly of a similar size. In contrast, sampling from a distribution with heavier tail yields a few very large samples in addition to many small ones that can tremendously dominate the sum (or equivalently sample mean approximation). Inasmuch as we clean MEE and do not incorporate major outliers (or very large error samples) in learning process, we can assume them eliminated from error PDF which means we lighten the possible error PDF with a heavier tail than that of Gaussian and consequently sample mean approximation is not problematic anymore. Recall that we use again our proposed running quartile estimation technique to detect and eliminate major outliers from sample mean approximation.

Eventually, to sum up this section, our proposed algorithm for online linear regression, called *Trimmed* MEE, is shown in Algorithm 5. At each time instant n the output of the system is learned as follows:

$$y_n = \mathbf{x}_n^T \mathbf{w}_{n-1} + \text{BIAS}_{n-1},$$

where BIAS_{n-1} is the sample mean calculated at time instant $n-1$ from samples of the tail-lightened but non-centered error PDF. The tail-lightened and centered error is denoted by (note that major outliers are not considered in following calculations):

$$e_n = d_n - y_n = d_n - (\mathbf{x}_n^T \mathbf{w}_{n-1} + \text{BIAS}_{n-1}). \tag{7.5}$$

Algorithm 5 Trimmed MEE

Inputs: $\{\mathbf{x}_n, d_n\}$

Output: $\mathbf{w}_n$ and BIAS_n

Initialisation : M, β, μ, σ, N, $\text{counter}_{NO} = 0$ (number of non-outliers), $\bar{d} = 0$, $\bar{\mathbf{x}} = \mathbf{0}$, $\text{BIAS}_0 = 0$ and $\mathbf{w}_0 = \mathbf{0}$

for each iteration n **do**
  $e_n = d_n - (\mathbf{x}_n^T \mathbf{w}_{n-1} + \text{BIAS}_{n-1})$
  Use Algorithm 3 to obtain $Q_{1,n}$ and $Q_{3,n}$
  $IQR_n = Q_{3,n} - Q_{1,n}$
  UE_n =$Q_{3,n} + 3 \times IQR_n$ (upper Extreme at time instant n)
  LE_n =$Q_{1,n} - 3 \times IQR_n$ (lower Extreme at time instant n)
  if $(LE_n \leq e_n \leq UE_n)$ **then**
    Calculate,

$$\bar{d} = \frac{d_n + (\text{counter}_{NO})\bar{d}}{\text{counter}_{NO} + 1}, \quad \bar{\mathbf{x}} = \frac{\mathbf{x}_n + (\text{counter}_{NO})\bar{\mathbf{x}}}{\text{counter}_{NO} + 1},$$

$$\text{counter}_{NO} = \text{counter}_{NO} + 1,$$

$$\nabla_2^{(n)}(E) = \frac{1}{N^2\sigma^2} \sum_{\substack{i=0 \\ LE_n \leq \text{for } e_{n-i} \text{ and } e_{n-j} \leq UE_n}}^{N-1} \sum_{j=0}^{N-1} \left[G_\sigma(e_{n-i} - e_{n-j})(e_{n-i} - e_{n-j})(\mathbf{x}_{n-i} - \mathbf{x}_{n-j}) \right],$$

    $\mathbf{w}_n = \mathbf{w}_{n-1} + \mu \nabla \hat{I}_2^{(n)}(E)$, $\text{BIAS}_n = \bar{d} - \bar{\mathbf{x}}^T \mathbf{w}_n$
  else
    $\mathbf{w}_n = \mathbf{w}_{n-1}$, $\text{BIAS}_n = \text{BIAS}_{n-1}$
  end if
end for
return $\mathbf{w}_n$ and BIAS_n =0

In the next section, simulation results are shown and discussed.

7.3 Simulation Results

Throughout this section we consider the linear adaptive filtering problem illustrated in Figure 7.1 which can be viewed as an online linear regression (for MEE and Trimmed MEE the BIAS block is also considered). In this Figure we have $\tilde{d}_n = \mathbf{x}_n^T \mathbf{w}_{opt}$ and $\tilde{y}_n = \mathbf{x}_n^T \mathbf{w}_{n-1}$ where $\mathbf{x}_n = \left[x_n, x_{n-1}, \cdots, x_{n-L+1}\right]^T$ and $\mathbf{w}_{n-1} = \left[w_{n-1}^{(1)}, w_{n-1}^{(2)}, \cdots, w_{n-1}^{(L)}\right]^T$ denote the input vector to the system at time instant n and system parameters estimated at time instant $n-1$, respectively. We

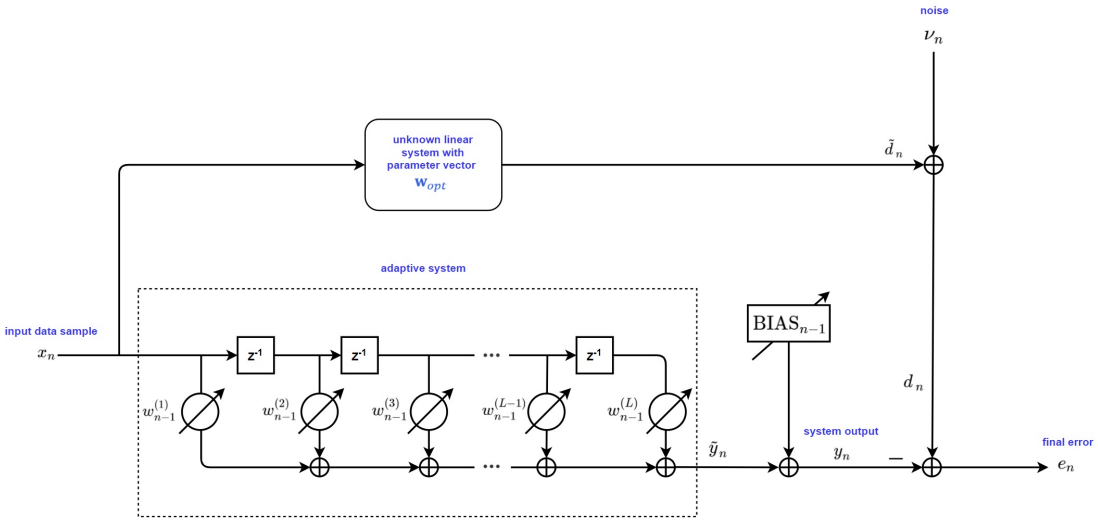


Figure 7.1: Linear adaptive filtering.

employ Algorithm 5 to adapt this linear system and obtain its parameter vector $\mathbf{w}$ and BIAS in each time instant. Moreover, for the sake of computational simplicity, we assume a fixed kernel bandwidth σ and also drop the expectation operator from both (2.1) and (2.6) to use the following online cost functions for stochastic MCC and stochastic MEE, respectively:

$$
\begin{aligned}
v_s(E) = G_\sigma(E) \quad &\Longrightarrow \quad \hat{v}_s^{(n)}(E) = G_\sigma(e_n), \\
I_{2,s}(E) = p(E) \quad &\Longrightarrow \quad \hat{I}_{2,s}^{(n)}(E) = p(e_n) \overset{\text{Parzen}}{=} \frac{1}{N}\sum_{i=0}^{N-1} G_\sigma\left(e_n - e_{n-i}\right),
\end{aligned}
$$

and consequently we have the following online cost function for stochastic MEEF with M fiducial point:

$$
\begin{aligned}
\hat{I}_{2,F,s}^{(n)}(E) &= \frac{1}{N+M}\sum_{i=0}^{N+M-1} G_\sigma\left(e_n - e_{n-i}\right) \\
&= \frac{N}{N+M}\hat{I}_{2,s}^{(n)}(E) + \frac{M}{N+M}\hat{v}_s^{(n)}(E).
\end{aligned}
$$

In addition, in our simulations throughout this section we draw input data samples x_i in Figure 7.1 from a standard normal distribution, i.e., $x_i \sim \mathcal{N}(0,1)$. The unknown system parameter vector

(or optimum weight vector to be learned) is generated randomly as a unit vector $\mathbf{w}_{opt} \in \mathscr{R}^L$ where $\| \mathbf{w}_{opt} \| = 1$ and $L = 5$. We also measure misalignment at time instant n for an algorithm based on NMSD as follows:

$$\begin{aligned} \text{misalignment}_n &= 20\log_{10}\left(\frac{\| \mathbf{w}_n - \mathbf{w}_{opt} \|}{\| \mathbf{w}_{opt} \|}\right) \\ &= 20\log_{10}\left(\| \mathbf{w}_n - \mathbf{w}_{opt} \|\right). \end{aligned}$$

7.3.1 Comparison of MCC and MEE

Recall that there is a difference between MCC and MEE cost functions as derived in (7.1) which means more difference between these cost functions will cause more difference in the results obtained based on them. We also expect MEE, as discussed earlier, to be a more comprehensive cost function than MCC and have a superior performance compared with that, therefore using fiducial points in MEEF, which alter MEE cost function to a weighted combination of MEE and MCC cost functions, will badly affect MEE performance whenever the difference between MEE and MCC is substantial. For instance, as shown in [34], for exponential noise the difference between two cost functions can become considerable. We can conclude then combination of MEE with MCC as MEEF is expected not to be helpful. This fact is shown in Figure 7.2 for different values of learning rate μ and kernel bandwidth σ. More precisely, note that two important factors to evaluate a learning algorithm are its convergence rate and steady-state misalignment. Given a specific pair (μ, σ) we run each algorithm and obtain steady-state misalignment for that as the sample mean of the misalignments within the last 200 time instants. Moreover, given a specific pair (μ, σ) we denote convergence rate for each algorithm by $i_{conv.}$

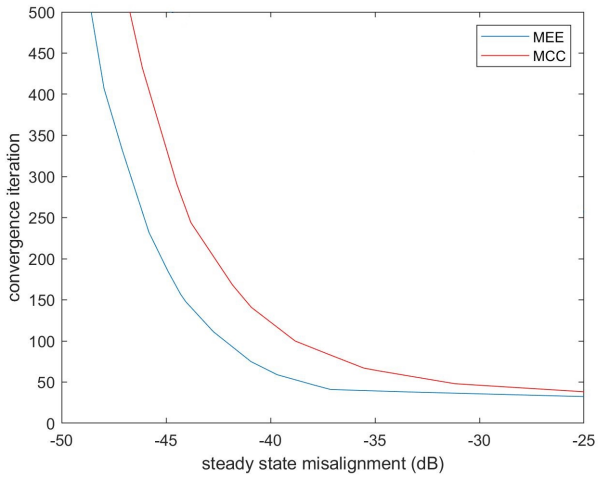


Figure 7.2: Convergence iteration vs. steady-state misalignment for MEE and MCC algorithms obtained from different pairs (μ, σ) such that $\mu \in [0.05 : 0.005 : 0.1]$ and $\sigma \in [0.2 : 0.1 : 1.4]$ (in presence of exponential noise and 30dB SNR).

and define it as the first time instant (or iteration) for which we have the following:

$$\text{misalignment}_{i_{conv.}} \leq \text{ steady} - \text{state misalignment } + \text{ 2.}$$

Now, recall that label (or desired signal) at time instant n is modeled as $d_n = \mathbf{x}_n^T \mathbf{w}_{opt} + v_n$. We assume an exponential noise (which is light-tailed), i.e., $v \sim Exp(\lambda)$, with following PDF:

$$f(v) = \lambda \exp(-\lambda v), \; v \geq 0 \;\text{ and }\; f(v) = 0, \text{ otherwise}$$

and 30dB signal to noise ratio (SNR) calculated as follows:

$$\text{SNR} = 10\log_{10}\left(\frac{E\left\{\left[\mathbf{x}_n^T \mathbf{w}_{opt}\right]^2\right\}}{E\{v_n^2\}}\right),$$

which results in $\lambda = \sqrt{2000}$. Achievable bounds in Figure 7.2 are obtained from convex hull of all points resulted from pairs (μ, σ) such that $\mu \in [0.05 : 0.005 : 0.1]$ and $\sigma \in [0.2 : 0.1 : 1.4]$ where for each pair (μ, σ) the convergence rate and steady-state misalignment results are obtained

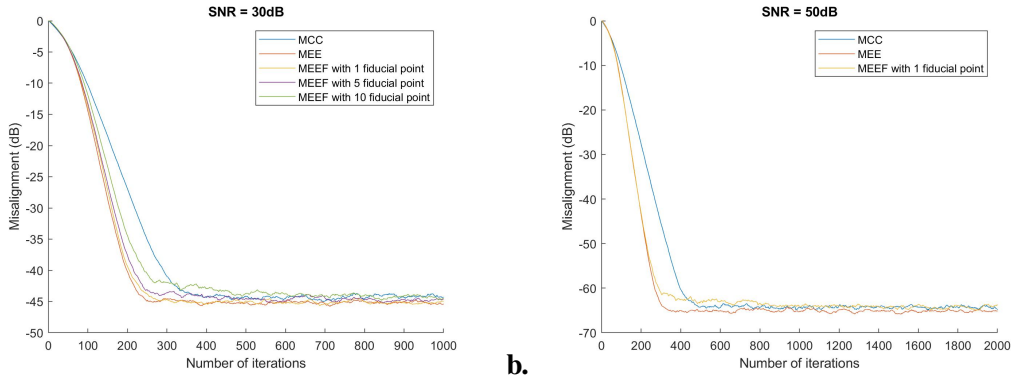


Figure 7.3: Learning curves of different algorithms under exponential noise averaged over 200 Monte Carlo simulations with $\mu = 0.05$ and $\sigma = 1$ (a. SNR=30dB and b. SNR=50dB).

by averaging learning curves over 50 Monte Carlo simulations. As seen in Figure 7.2, MEE outperforms MCC significantly for exponential noise. This can be translated to the fact that using MEEF for an environment corrupted by this exponential noise is not helpful inasmuch as it degrades MEE performance by incorporating MCC into it (as seen in previous chapter as well). This is shown in Figure 7.3-a in which learning curves (averaged over 200 Monte Carlo simulations) of MEE, MCC and MEEF are compared. As seen in this Figure, as we increase number of fiducial points, the performance of the MEE algorithm deteriorates.

It is worth mentioning that, as discussed in [34], when the parameter of the exponential noise λ is increased the difference between MCC and MEE cost functions becomes larger as well and therefore we expect more destructive effect of incorporating MCC into MEE as MEEF. This is shown in Figure 7.3-b where we assume 50dB SNR which results in a larger λ ($\lambda = \sqrt{2 \times 10^5}$) and as depicted in this Figure even one fiducial point corrupts MEE performance significantly. Recal from previous chapter that convergence iteration versus steady-state misalignment achievable bounds for MEE, MEEF with 1 fiducial point and MCC were illustrated in Figure 6.3 for 50dB SNR. As seen in Figure 6.3, MEE outperforms both MCC and MEEF.

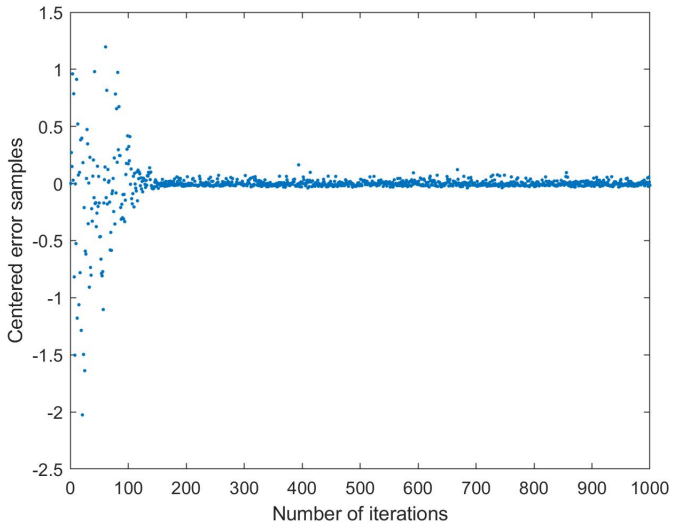
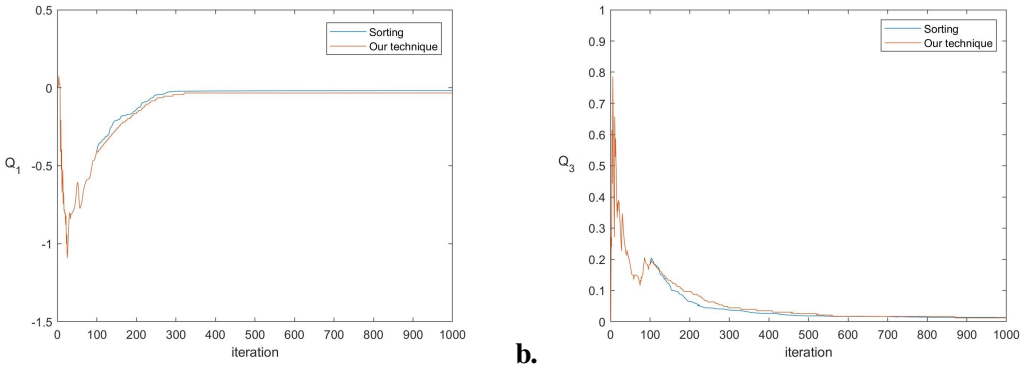


Figure 7.4: Scatter plot of final error samples of MEE algorithm of Figure 7.3-a.

7.3.2 Performance Analysis of Algorithm 3 for Error Samples Generated from MEE Algorithm

As discussed in Chapter 5, ground truth for evaluation of our proposed error samples running quartile estimation technique are the results obtained for $Q_{1,n}$ and $Q_{3,n}$ at time instant n from order statistics by simply sorting final error samples (centered error samples) in each time instant. We consider final error samples (7.5) resulted from running MEE algorithm in Figure 7.3-a. The scatter plot of these error samples are shown in Figure 7.4. As seen in this Figure, centered error samples are accumulated around the origin with time (and consequently Q_1 and Q_3 also get closer to the origin with time) which verifies our assumption in Algorithm 1 that median of error samples is assumed zero. It is worth mentioning that as expected we do not see major outliers in centered error samples in Figure 7.4 inasmuch as exponential noise is light-tailed. Now we plot $Q_{1,n}$ and $Q_{3,n}$ of these centered error samples in Figure 7.5-a and Figure 7.5-b, respectively obtained from both order statistics and our proposed technique with $M = 100$ and $\varepsilon = 0.01$ (Algorithm 3). As seen in this Figure, the difference between $Q_{1,n}$ ($Q_{3,n}$) obtained from sorting and that obtained from our technique is negligible.



a. **b.**

Figure 7.5: Lower quartile Q_1 and upper quartile Q_3 of error samples of Figure 7.3-a estimated at each time instant based on sorting and Algorithm 1 (our technique with $M = 100$ and $\varepsilon = 0.01$).

In the following, we show how our technique in Algorithm 3 will lighten an impulsive noise which has a heavier tail than that of Gaussian and is modeled as a mixture of two Gaussian distributions (with means equal to 0 and standard deviations equal to 10^{-4} and 10) as follows:

$$v \sim 0.9\mathcal{N}(0, 10^{-8}) + 0.1\mathcal{N}(0, 100). \tag{7.6}$$

The second term of above impulsive noise generates abnormally large noise samples or outliers. Scatter plot and histogram (with 100 bins with equal size) of 10000 samples drawn from this distribution are shown in Figure 7.6-a and Figure 7.6-b, respectively. Next, we deploy our running quartile technique in Algorithm 3 to obtain $Q_{1,n}$ and $Q_{3,n}$ and then calculate upper and lower extremes based on them. Now, we use these extremes to detect and exclude these major outliers from noise samples which results in scatter plot of non-outlier noise samples in Figure 7.6-c (974 noise samples out of total 10000 noise samples have been detected and excluded as major outliers). As seen in this Figure, non-outlier noise samples are concentrated around $v = 0$. Finally, Figure 7.6-d shows histogram of these non-outlier noise samples that can be interpreted as histogram of samples drawn from lightened version of impulsive noise (7.6) (again number of equal-size bins is 100).

We can readily calculate the mean of random variable v distributed according to

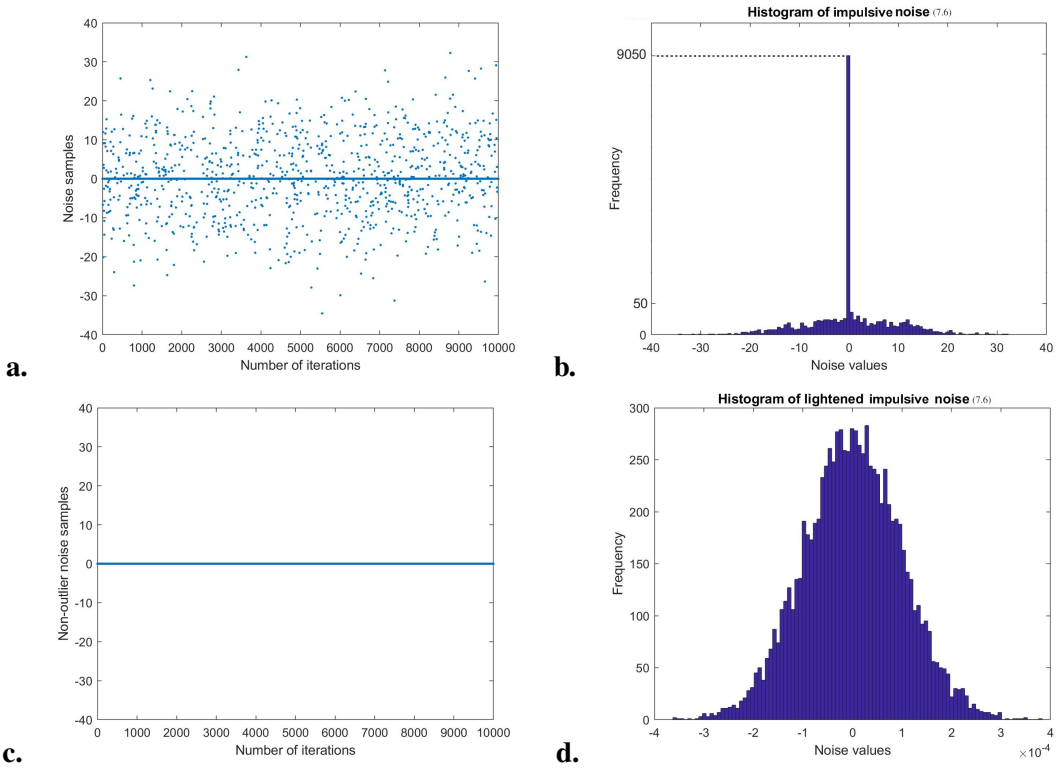


Figure 7.6: Scatter plot and histogram of noise samples drawn from impulsive noise (7.6) (Figures a. and b.) and those of noise samples drawn from lightened version of impulsive noise (7.6) obtained by using Algorithm 3 (Figures c. and d.).

impulsive distribution (7.6) which is 0. Figure 7.7 depicts how sample mean approximation will converge to actual mean, i.e., 0. As illustrated in this Figure, if we use all noise samples including major outliers convergence occurs with many fluctuations around the actual mean while when we exclude these major outliers, convergence to the mean of the lightened noise distribution happens very fast and more consistently and smoothly.

7.3.3 Trimmed MEE

In this subsection, MEE and MEEF are compared with our proposed Trimmed MEE in Algorithm 5 for online linear regression in which we deploy our proposed error samples running quartile estimation technique (Algorithm 3) in order to detect and exclude major outliers (or abnormally large error samples) from learning process. Recall that for MEE and Trimmed MEE

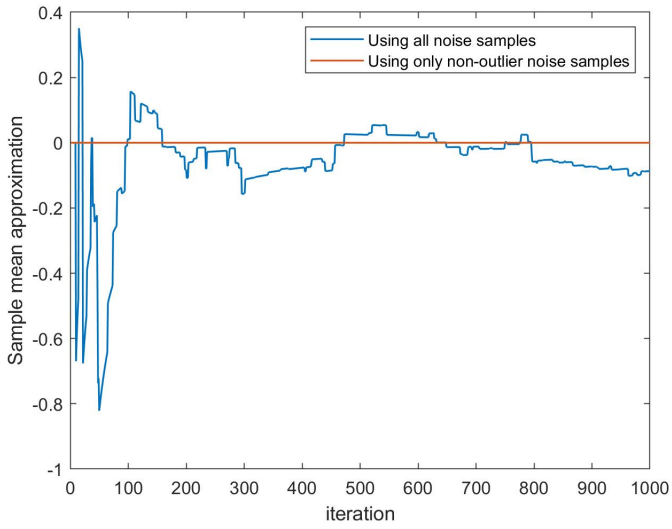


Figure 7.7: Sample mean approximation using all noise (7.6) samples (including major outliers) versus using only non-outlier noise samples (detected based on Algorithm 3 and concept of the upper and lower extremes).

algorithms we add a bias (error sample mean) obtained from all error samples and non-outlier error samples, respectively to the output of the system in order to locate the final error samples around the origin. First, we consider exponential noises $v_1 \sim Exp(\sqrt{2000})$ and $v_2 \sim Exp(\sqrt{2 \times 10^5})$ then Gaussian noise $v_3 \sim \mathcal{N}(0, 10^{-3})$ which are light-tailed, hence we do not need to be concerned about destructive effect of major outliers. In other words we expect to obtain similar results regardless of using MEE or Trimmed MEE for environments affected by these noises as shown in Figures 7.8-a, 7.8-b and 7.8-c, where for exponential noises Trimmed MEE even shows a slightly better steady-state misalignment performance compared to MEE which is not surprising. Learning curve of MEEF also shows similar behaviour to MEE and Trimmed MEE, however for exponential noise MEEF performance deteriorates as SNR is increased inasmuch as increase in SNR means increase in λ which results in larger gap between MEE and MCC cost functions [34]. In order to show superiority of Trimmed MEE over MEE and MEEF in presence of impulsive noises, we consider following symmetric and asymmetric mixture of Gaussians

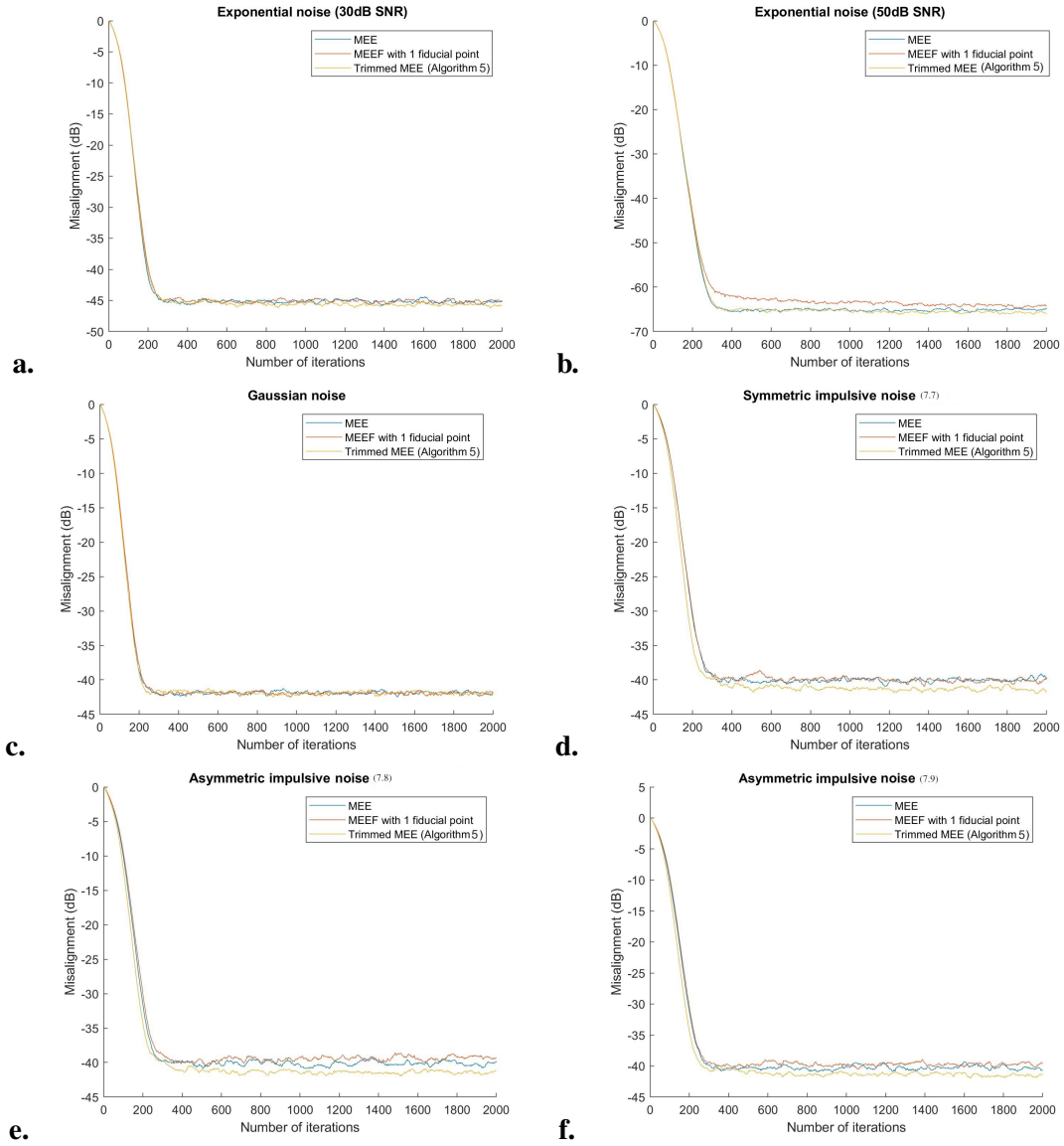


Figure 7.8: Learning curves of MEE, MEEF and Trimmed MEE under different noises averaged over 200 Monte Carlo simulations with $\mu = 0.05$ and $\sigma = 1$.

noises:

$$v_1 \sim 0.9\mathcal{N}(0,10^{-3}) + 0.1\mathcal{N}(0,1000), \tag{7.7}$$

$$v_2 \sim 0.9\mathcal{N}(0,10^{-3}) + 0.1\mathcal{N}(10,1000). \tag{7.8}$$

$$v_3 \sim 0.9\mathcal{N}(-5,10^{-3}) + 0.1\mathcal{N}(10,1000). \tag{7.9}$$

As seen in Figures 7.8-d, 7.8-e and 7.8-f, for all symmetric and asymmetric impulsive noises (7.7), (7.8) and (7.9) Trimmed MEE outperforms MEE and MEEF from both convergence rate and steady-state misalignment point of views. Learning curves of MEE and MEEF are similar, however for asymmetric noises (7.8) and (7.9) MEE shows a slightly better performance. It is worth mentioning that all learning curves of Figure 7.8 are obtained by averaging over 200 independent Monte Carlo simulations. The claimed strength of MEEF in literature is its ability to locate the majority of errors between system outputs and labels around the origin which becomes more clear when we do error analysis. However, this claim is valid as long as the gap between MEE and MCC cost functions is not large. This fact is shown in Table 7.1 where testing error analysis of these three algorithms (MEE, MEEF with 1 fiducial point and Trimmed MEE) is done for noises discussed in Figure 7.8. The error analysis is done based on the following metric called mean absolute error (MAE):

$$\text{MAE} = \frac{1}{T}\sum_{t=1}^{T}|d_t - y_t| = \frac{1}{T}\sum_{t=1}^{T}|e_t|,$$

where T is the number of testing samples. Again, we ran 200 independent Monte Carlo simulations, each with 2000 iterations ($\mu = 0.05,\ \sigma = 1,\ M = 100$ and $\varepsilon = 0.01$), learned the system parameters in each simulation and then obtained testing errors based on 2000 testing samples for each simulation, therefore each MAE shown in Table 7.1 is obtained by averaging over 200

Table 7.1: Testing Mean Absolute Errors of Different Algorithms under Different Noises

	MEE	MEEF	Trimmed MEE
Exponential noise with 30dB SNR	0.0171±0.0012	0.0232±0.0016	**0.0169±0.0010**
Exponential noise with 50dB SNR	**0.0017±0.0001**	0.0023±0.0002	**0.0017±0.0001**
Gaussian noise with 30dB SNR	0.0262±0.0016	0.0262±0.0016	0.0262±0.0016
Mixture of Gaussians noise (7.7)	2.6669±0.4625	2.5824±0.4206	**2.5485±0.4043**
Mixture of Gaussians noise (7.8)	3.4912±0.5754	2.6762±0.4812	**2.6469±0.4272**
Mixture of Gaussians noise (7.9)	4.0958±0.6527	7.1222±0.4281	**2.8526±0.5376**

MAEs. The smallest MAE amongst all three algorithms for each noise is highlighted in bold. As seen in this Table, for all noises Trimmed MEE gives the best result. More precisely, for exponential noise MEE and Trimmed MEE outperforms MEEF, as expected because for this noise gap between MEE and MCC cost functions is large. For Gaussian noise the performance of all algorithms is the same. For mixture of two Gaussians noises, which have heavier tails than that of Gaussian, as long as the gap between MEE and MCC is not large MEEF outperforms MEE, although Trimmed MEE still shows the best performance. However, once this noise makes a large gap between MEE and MCC cost functions, as noise (7.9) does, MEEF shows a weaker performance even than MEE while Trimmed MEE shows the best performance again.

Figure 7.9 shows the testing error histograms (obtained based on 2000 testing samples) for these three algorithms under noises (7.8) and (7.9). As shown in this Figure, we can see again superiority of our algorithm where mass of the testing errors (approximately 90% of them) is closer to the origin in Trimmed MEE compared to MEE and MEEF. Note that for noise (7.9), which makes large gap between MEE and MCC cost functions, MEEF even performs worse than MEE.

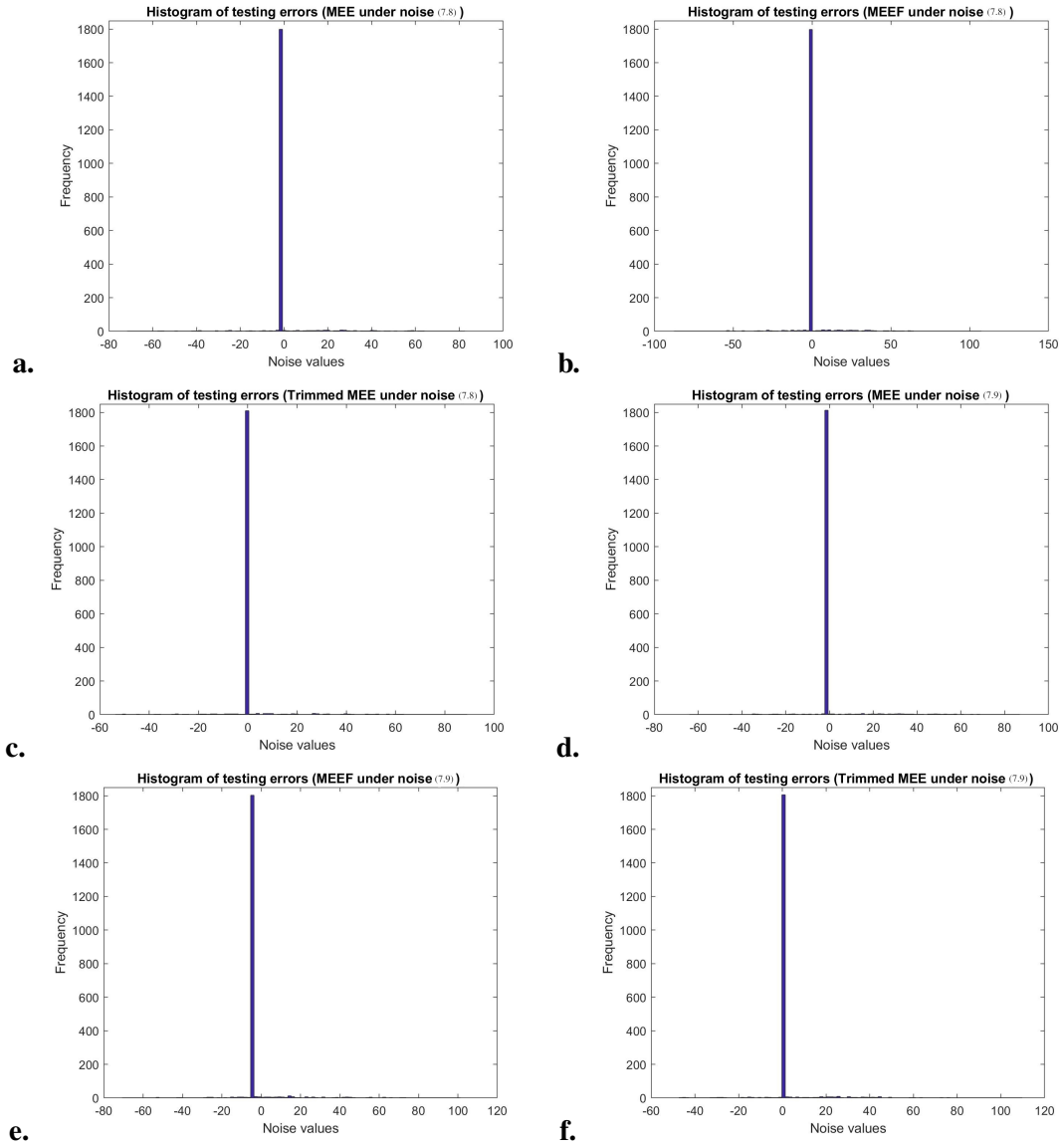


Figure 7.9: Testing error histograms of MEE, MEEF and Trimmed MEE obtained based on 2000 testing samples under Mixture of two Gaussians noises (7.8) and (7.9).

7.4 Conclusion

In this chapter, we address again robust online linear regression in the presence of non-Gaussian noises, especially those with heavier tails than that of Gaussian. Error entropy as a robust cost function has been utilized for robust learning under non-Gaussianity, however since entropy is shift-invariant we need to take some extra steps to locate the error PDF around the origin. To this end, two methods has been proposed nevertheless we show the shortcomings

of these methods in this chapter and propose our alternative. Indeed, we employ our online algorithm proposed in Chapter 5 in order to find the running quartiles of the error samples. Then, we use them to detect and eliminate major outliers from learning procedure based on MEE. We call our proposed method *Trimmed MEE*. Simulation results show the robustness of our algorithm to non-Gaussian noises and its superiority over known methods in locating error PDF around the origin. In more details, proposed algorithm results in a learning curve with faster convergence to lower steady-state misalignment and also achieves lower testing error compared to other algorithms.

In the next chapter, we conclude this thesis and make some recommendations for future research directions.

Chapter 8

Conclusion and Future Work

In this thesis, we discussed the fact that although the well-known classical central limit theorem in introductory statistics creates the expectations that we must see the Gaussian distribution everywhere in the real world, this is not necessarily true and the emergence of non-Gaussian distributions in the environments such as computer systems and networks, stocks and many more should be expected not surprising. Indeed, classical central limit theorem is concluded under some assumptions that do not hold in many real scenarios. In real world, many non-Gaussian distributions may arise that result in outliers. The problem of learning a system affected by these distributions is of great importance inasmuch as conventional learning approaches that are mostly based on Gaussian assumption are not effective anymore. In this thesis, we addressed this problem by means of information theory as a more comprehensive learning paradigm that encompasses effectively data distributions other than Gaussian as well. More precisely, we improved the performances of correntropy and quadratic Renyi's entropy as two robust information-theoretic cost functions in the online linear regression (or linear adaptive filtering).

In Chapter 3, we proposed a new hybrid algorithm based on recursive MCC and

gradient-based MCC. This algorithm was demonstrated to outperform previous work in terms of both convergence speed and steady-state misalignment. At the same time, the proposed algorithm benefited from lower computational complexity compared to many other algorithms.

In Chapter 4, a modified version of MCC was proposed where an existing MCC algorithm with adaptive kernel bandwidth was improved to learn the underlying linear system more efficiently . In this improved version of MCC, abnormally large error samples or major outliers were detected by using error quartiles and concepts of the inner and outer fences. Quartiles were obtained by simply sorting error samples in each iteration. This elimination of major outliers from learning process leads to better steady-state performance compared to previous algorithms.

Chapter 5 was devoted to proposing an efficient technique to detect, and then exclude abnormally large error samples from MCC. This technique is indeed an efficient running quartile estimation based on quantization of error samples which is used instead of sorting. Using these efficiently obtained quartiles and concepts of the inner and outer fences, we were able to reject major outliers. This technique alongside correntropy led to lower steady-state misalignment compared to previous algorithms.

In Chapter 6, we challenged the deployment of fiducial points in MEE. We saw that error entropy is minimized to extract as much information as possible about the data generating system. However, this minimum entropy can also occur for other error PDFs not located at the origin inasmuch as entropy is shift-invariant. In these cases, an undesired estimate of the system parameters is obtained. Therefore, an extra step needs to be taken to concentrate error samples around the origin. The most celebrated approach towards that end is MEEF, in which some external and artificial zero error samples, called fiducial points (not generated by the underlying system), are added to the cost function as the reference points to force actual error samples to get concentrated around them. Using these fiducial points translates MEEF into a weighted

combination of MEE and MCC. We showed that incorporating these fiducial points into MEE can even degrade the steady-state misalignment and/or convergence speed.

Eventually, we proposed an alternative to MEEF in Chapter 7 to concentrate error samples around zero. We used our proposed quantization technique by which not only aforementioned need of setting errors around the origin in MEE was addressed, but also major outliers were rejected from MEE-based learning and MEE performance was improved from convergence rate and steady-state misalignment points of view.

Despite various researches on aforementioned information-theoretic cost functions and many recent improvements, still many aspects of them have remained untouched. For example, we have seen so far that real world data sets are conducive to non-Gaussian PDFs that can even change in time, therefore data distribution cannot be found in terms of finite set of parameters, consequently we need to use a non-parametric PDF estimation method such as Parzan to estimate information-theoretic cost functions from unknown statistics. Better non-parametric PDF estimation will yield more accurate results and need to be further investigated. As another research direction, we can point to adaptation of kernel bandwidth. To the best of our knowledge, there is no universal updating technique available for kernel bandwidth adaptation yet, while proper adaptation of this free parameter can improve the learning performance significantly. In addition to above research directions, note that we only discussed the deployment of information-theoretic cost functions in online linear regression, while it seems that the appealing part of incorporating information theory into learning is the fact that we can simply modify many other conventional supervised and unsupervised learning methodologies [1]. As an example, we can launch more investigations on incorporating information theory into deep learning. This is well-known that deep learning outperforms traditional machine learning as the scale of data increases, therefore, due to abundance of data these days and also having access to powerful graphics processing units (GPUs) it is extensively being used as a strong tool, especially for feature

engineering on its own [95–98]. Despite vast studies on deep architectures (such as [99–101]), comprehensive deep learning topology for both Gaussian and non-Gaussian environments needs further investigation [102]. As another direction for future research, note that the benefits of incorporating information theory into learning are achieved at the cost of higher computational complexity that may pose computational bottlenecks for large-scale data sets, therefore we can search for more efficient approximates of information-theoretic cost functions to mitigate this issue.

To sum up, we saw that the efficient extraction of information from data is the fundamental goal of data processing, however, if the environment does not follow Gaussian distribution, employing conventional learning methods may fail to achieve satisfactory results. Unfortunately, in many real world applications data statistics are unknown, in other words, we do not know whether data follows a Gaussian behaviour or not, therefore, there is no guarantee that we can efficiently extract information. This means, any future research on robust learning needs to address efficient extraction of information in data processing and propose a comprehensive learning paradigm with no need to have knowledge about statistics of the underlying environment. The outcomes of these researches may find many new applications in computer science, biology, physics, the social sciences, engineering, and everywhere that non-Gaussian distributions are extensively observed.

Bibliography

[1] J. C. Principe, *Information theoretic learning: Renyi's entropy and kernel perspectives.* Springer Science & Business Media, 2010.

[2] A. W. J. Nair and B. Zwart, *The Fundamentals of Heavy Tails: Properties, Emergence, and Estimation.* Preprint, California Institute of Technology, 2020.

[3] R. Mitra and V. Bhatia, "Minimum error entropy criterion based channel estimation for massive-mimo in vlc.," *in IEEE Transactions on Vehicular Technology*, vol. 68, no. 1, pp. 1014–1018, 2019.

[4] H. Y. K. Xu and Y.-J. Zhu, "Channel-adapted spatial modulation for massive mimo visible light communications," *IEEE Photonics Technology Letters*, vol. 28, no. 23, pp. 2693–2696, 2016.

[5] S. Z. X. Kuai, H. Sun and E. Cheng, "Impulsive noise mitigation in underwater acoustic ofdm systems," *IEEE Transactions on Vehicular Technology*, vol. 65, no. 10, pp. 8190–8202, 2016.

[6] S. N. Z. H. P. Chen, Y. Rong and A. J. Duncan, "Joint channel estimation and impulsive noise mitigation in underwater acoustic ofdm communication systems," *IEEE Transactions on Wireless Communications*, vol. 16, no. 9, pp. 6165–6178, 2017.

[7] M. Stojanovic and J. Preisig, "Underwater acoustic communication channels: Propagation models and statistical characterization," *IEEE communications magazine*, vol. 47, no. 1, pp. 84–89, 2009.

[8] S. Banerjee and M. Agrawal, "On the performance of underwater communication system in noise with gaussian mixture statistics," in *2014 Twentieth National Conference on Communications (NCC)*, pp. 1–6, IEEE, 2014.

[9] S. N. P. Chen, Y. Rong and Z. He, "Joint channel and impulsive noise estimation in underwater acoustic ofdm systems," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 11, pp. 10567–10571, 2017.

[10] M. N. J. Lin and B. L. Evans, "Impulsive noise mitigation in powerline communications using sparse bayesian learning," *IEEE Journal on Selected Areas in Communications*, vol. 31, no. 7, pp. 1172–1183, 2013.

[11] A. S. S. Galli and Z. Wang, "For the grid and through the grid: The role of power line communications in the smart grid," *Proceedings of the IEEE*, vol. 99, no. 6, pp. 998–1027, 2011.

[12] E. A. H. A. S. Sadrizadeh, N. Zarmehi and F. Marvasti, "A fast iterative method for removing impulsive noise from sparse signals," *IEEE Transactions on Circuits and Systems for Video Technology*, 2020.

[13] Q. W. D. X. Z. T. Hu, J. Fan, "Learning theory approach to minimum error entropy criterion.," *Journal of Machine Learning Research*, vol. 14, no. 2, 2013.

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

[15] V. Mathews and S. Cho, "Improved convergence analysis of stochastic gradient adaptive filters using the sign algorithm," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 35, no. 4, pp. 450–454, 1987.

[16] D. Erdogmus and J. C. Principe, "An error-entropy minimization algorithm for supervised training of nonlinear adaptive systems," *IEEE Transactions on Signal Processing*, vol. 50, no. 7, pp. 1780–1786, 2002.

[17] P. P. W. Liu and J. C. Principe, "Correntropy: Properties and applications in non-gaussian signal processing," *IEEE Transactions on signal processing*, vol. 55, no. 11, pp. 5286–5298, 2007.

[18] H. Z. N. Z. J. C. P. B. Chen, L. Xing, "Generalized correntropy for robust adaptive filtering," *IEEE Transactions on Signal Processing*, vol. 64, no. 13, pp. 3376–3387, 2016.

[19] X. Z. Y. Zhu, H. Zhao and B. Chen, "Robust generalized maximum correntropy criterion algorithms for active noise control," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 28, pp. 1282–1292, 2020.

[20] C. S. M. Zhang, Y. Gao and M. Blumenstein, "Robust tensor decomposition for image representation based on generalized correntropy," *IEEE Transactions on Image Processing*, vol. 30, pp. 150–162, 2020.

[21] Y. D. T. J. J. L. N. Zhou, B. Chen and Y. Xu, "Maximum correntropy criterion-based robust semisupervised concept factorization for image representation," *IEEE transactions on neural networks and learning systems*, vol. 31, no. 10, pp. 3877–3891, 2019.

[22] G. K. E. Sfeir, R. Mitra and V. Bhatia, "Performance analysis of maximum-correntropy based detection for scma," *IEEE Communications Letters*, 2020.

[23] T. M. Cover and J. A. Thomas, *Elements of information theory*. John Wiley & Sons, 2006.

[24] G. K. G. D. R. Mitra, Rangeet and G. Poitau, "Error analysis of localization based on minimum-error entropy with fiducial points," *IEEE Communications Letters*, 2020.

[25] S. W. W. M. L. Dang, B. Chen and P. Ren, "Robust power system state estimation with minimum error entropy unscented kalman filter," *IEEE Transactions on Instrumentation and Measurement*, vol. 69, no. 11, pp. 8797–8808, 2020.

[26] Y. L. J. C. P. B. Chen, X. Wang, "Maximum correntropy criterion with variable center," *IEEE Signal Processing Letters*, vol. 26, no. 8, pp. 1212–1216, 2019.

[27] H. Z. N. Z. B. Chen, J. Wang and J. C. Principe, "Convergence of a fixed-point algorithm under maximum correntropy criterion," *IEEE Signal Processing Letters*, vol. 22, no. 10, pp. 1723–1727, 2015.

[28] Y. G. J. C. Y. Xie, Y. Li and B. Chen, "Fixed-point minimum error entropy with fiducial points," *IEEE Transactions on Signal Processing*, vol. 68, pp. 3824–3833, 2020.

[29] S. Y. S. D. B. Chen, R. Ma and J. Qin, "Granger causality analysis based on quantized minimum error entropy criterion," *IEEE Signal Processing Letters*, vol. 26, no. 2, pp. 347–351, 2019.

[30] N. Z. B. Chen, L. Xing and J. C. Principe, "Quantized minimum error entropy criterion," *IEEE transactions on neural networks and learning systems*, vol. 30, no. 5, pp. 1370–1380, 2018.

[31] S. S. Haykin, *Adaptive filter theory*. Pearson Education India, 2014.

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

[33] A. Papoulis and P. S. U., *Probability, random variables, and stochastic processes*. McGraw-Hill, 2002.

[34] A. R. Heravi and G. A. Hodtani, "A new information theoretic relation between minimum error entropy and maximum correntropy," *IEEE Signal Processing Letters*, vol. 25, no. 7, pp. 921–925, 2018.

[35] S. M. Kay, *Fundamentals of statistical signal processing*. Prentice Hall PTR, 1993.

[36] S. Zhao, B. Chen, and J. C. Principe, "Kernel adaptive filtering with maximum correntropy criterion," in *The 2011 International Joint Conference on Neural Networks*, pp. 2012–2017, IEEE, 2011.

[37] A. Rényi *et al.*, "On measures of entropy and information," in *Proceedings of the Fourth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Contributions to the Theory of Statistics*, The Regents of the University of California, 1961.

[38] E. Parzen, "On estimation of a probability density function and mode," *The annals of mathematical statistics*, vol. 33, no. 3, pp. 1065–1076, 1962.

[39] A. Gramacki, *Nonparametric kernel density estimation and its computational aspects*. Springer, 2018.

[40] B. W. Silverman, *Density estimation for statistics and data analysis*. Routledge, 2018.

[41] A. W. Bowman, "An alternative method of cross-validation for the smoothing of density estimates," *Biometrika*, vol. 71, no. 2, pp. 353–360, 1984.

[42] A. W. Bowman and A. Azzalini, *Applied smoothing techniques for data analysis: the kernel approach with S-Plus illustrations*, vol. 18. OUP Oxford, 1997.

[43] R. Cao, A. Cuevas, and W. G. Manteiga, "A comparative study of several smoothing methods in density estimation," *Computational Statistics & Data Analysis*, vol. 17, no. 2, pp. 153–176, 1994.

[44] W. K. Härdle, M. Müller, S. Sperlich, and A. Werwatz, *Nonparametric and semiparametric models*. Springer Science & Business Media, 2004.

[45] N.-B. Heidenreich, A. Schindler, and S. Sperlich, "Bandwidth selection for kernel density estimation: a review of fully automatic selectors," *AStA Advances in Statistical Analysis*, vol. 97, no. 4, pp. 403–433, 2013.

[46] M. C. Jones, J. S. Marron, and S. J. Sheather, "A brief survey of bandwidth selection for density estimation," *Journal of the American statistical association*, vol. 91, no. 433, pp. 401–407, 1996.

[47] M. C. Jones, J. S. Marron, and S. J. Sheather, "Progress in data-based bandwidth selection for kernel density estimation," tech. rep., North Carolina State University. Dept. of Statistics, 1992.

[48] B. U. Park and J. S. Marron, "Comparison of data-driven bandwidth selectors," *Journal of the American Statistical Association*, vol. 85, no. 409, pp. 66–72, 1990.

[49] M. Rudemo, "Empirical choice of histograms and kernel density estimators," *Scandinavian Journal of Statistics*, pp. 65–78, 1982.

[50] D. W. Scott, *Multivariate density estimation: theory, practice, and visualization*. John Wiley & Sons, 2015.

[51] D. W. Scott and G. R. Terrell, "Biased and unbiased cross-validation in density estimation," *Journal of the american Statistical association*, vol. 82, no. 400, pp. 1131–1146, 1987.

[52] S. J. Sheather and M. C. Jones, "A reliable data-based bandwidth selection method for kernel density estimation," *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 53, no. 3, pp. 683–690, 1991.

[53] B. A. Turlach, "Bandwidth selection in kernel density estimation: A review," in *CORE and Institut de Statistique*, Citeseer, 1993.

[54] M. P. Wand and M. C. Jones, *Kernel smoothing*. CRC press, 1994.

[55] B. Chen, X. Wang, N. Lu, S. Wang, J. Cao, and J. Qin, "Mixture correntropy for robust learning," *Pattern Recognition*, vol. 79, pp. 318–327, 2018.

[56] L. Bottou and Y. LeCun, "Large scale online learning," *Advances in neural information processing systems*, vol. 16, pp. 217–224, 2004.

[57] W. Liu, P. P. Pokharel, and J. C. Principe, "Error entropy, correntropy and m-estimation," in *16th IEEE Signal Processing Society Workshop on Machine Learning for Signal Processing*, pp. 179–184, Arlington, VA 2006.

[58] W. Ma, H. Qu, and J. Zhao, "Estimator with forgetting factor of correntropy and recursive algorithm for traffic network prediction," in *25th Chinese Control and Decision Conference (CCDC)*, pp. 490–494, Guiyang, China, May 2013.

[59] X. Zhang, K. Li, Z. Wu, Y. Fu, H. Zhao, and B. Chen, "Convex regularized recursive maximum correntropy algorithm," *Signal Processing*, vol. 129, pp. 12–16, Dec. 2016.

[60] Z. Wu, J. Shi, X. Zhang, W. Ma, and B. Chen, "Kernel recursive maximum correntropy," *Signal Processing*, vol. 117, pp. 11–16, Dec. 2015.

[61] A. Singh and J. C. Principe, "Kernel width adaptation in information theoretic cost functions," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 2062–2065, Dallas, TX, Mar. 2010.

[62] S. Zhao, B. Chen, and J. C. Principe, "An adaptive kernel width update for correntropy," in *International Joint Conference on Neural Networks (IJCNN)*, pp. 1–5, Australia, June 2012.

[63] W. Wang, J. Zhao, H. Qu, B. Chen, and J. C. Principe, "A switch kernel width method of correntropy for channel estimation," in *International Joint Conference on Neural Networks (IJCNN)*, pp. 1–7, Ireland, July 2015.

[64] W. Wang, J. Zhao, H. Qu, B. Chen, and J. C. Principe, "An adaptive kernel width update method of correntropy for channel estimation," in *IEEE International Conference on Digital Signal Processing (DSP)*, pp. 916–920, Singapore, July 2015.

[65] F. Huang, J. Zhang, and S. Zhang, "Adaptive filtering under a variable kernel width maximum correntropy criterion," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 64, pp. 1247–1251, Oct. 2017.

[66] L. Shi, H. Zhao, and Y. Zakharov, "An improved variable kernel width for maximum correntropy criterion algorithm," *IEEE Transactions on Circuits and Systems II: Express Briefs*, Nov. 2018.

[67] H. Radmanesh and M. Hajiabadi, "Recursive maximum correntropy learning algorithm with adaptive kernel size," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 65, pp. 958–962, July 2018.

[68] S. Bahrami and E. Tuncel, "A new approach to online regression based on maximum correntropy criterion," in *2019 IEEE 29th International Workshop on Machine Learning for Signal Processing (MLSP)*, pp. 1–6, IEEE, 2019.

[69] W. Wang, J. Zhao, H. Qu, B. Chen, and J. C. Principe, "Convergence performance analysis of an adaptive kernel width mcc algorithm," *AEU-International Journal of Electronics and Communications*, vol. 76, pp. 71–76, 2017.

[70] S. Bahrami and E. Tuncel, "Mitigating outlier effect in online regression: An efficient usage of error correntropy criterion," in *2020 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–6, IEEE, 2020.

[71] J. W. Tukey *et al.*, *Exploratory data analysis*, vol. 2. Reading, Mass., 1977.

[72] N. M. R. SURI, M. N. Murty, and G. Athithan, *Outlier detection: techniques and applications*. Springer, 2019.

[73] R. Jain and I. Chlamtac, "The p2 algorithm for dynamic calculation of quantiles and histograms without storing observations," *Communications of the ACM*, vol. 28, no. 10, pp. 1076–1085, 1985.

[74] H. L. Hammer and A. Yazidi, "Smooth estimates of multiple quantiles in dynamically varying data streams," *Pattern Analysis and Applications*, pp. 1–12, 2019.

[75] N. Tiwari and P. C. Pandey, "A technique with low memory and computational requirements for dynamic tracking of quantiles," *Journal of Signal Processing Systems*, vol. 91, no. 5, pp. 411–422, 2019.

[76] O. Arandjelović, "Targeted adaptable sample for accurate and efficient quantile estimation in non-stationary data streams," *Machine Learning and Knowledge Extraction*, vol. 1, no. 3, pp. 848–870, 2019.

[77] I. Song, P. Park, and R. W. Newcomb, "A normalized least mean squares algorithm with a step-size scaler against impulsive measurement noise," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 60, no. 7, pp. 442–445, 2013.

[78] F. Huang, J. Zhang, and S. Zhang, "Combined-step-size affine projection sign algorithm for robust adaptive filtering in impulsive interference environments," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 63, no. 5, pp. 493–497, 2015.

[79] S. Bahrami and E. Tuncel, "An efficient running quantile estimation technique alongside correntropy for outlier rejection in online regression," in *2020 IEEE International Symposium on Information Theory (ISIT)*, pp. 2813–2818, IEEE, 2020.

[80] K. Sayood, *Introduction to data compression*. Morgan Kaufmann, 2017.

[81] B. Chen, L. Dang, Y. Gu, N. Zheng, and J. C. Príncipe, "Minimum error entropy kalman filter," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 2019.

[82] B. Chen, L. Xing, B. Xu, H. Zhao, and J. C. Principe, "Insights into the robustness of minimum error entropy estimation," *IEEE transactions on neural networks and learning systems*, vol. 29, no. 3, pp. 731–737, 2016.

[83] P. Bromiley, "Products and convolutions of gaussian probability density functions," *Tina-Vision Memo*, vol. 3, no. 4, p. 1, 2003.

[84] S. Peng, W. Ser, B. Chen, L. Sun, and Z. Lin, "Robust constrained adaptive filtering under minimum error entropy criterion," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 65, no. 8, pp. 1119–1123, 2018.

[85] S. Han, S. Rao, and J. Principe, "Estimating the information potential with the fast gauss transform," in *International Conference on Independent Component Analysis and Signal Separation*, pp. 82–89, Springer, 2006.

[86] B. Chen, Y. Zhu, J. Hu, and M. Zhang, "On optimal estimations with minimum error entropy criterion," *Journal of the Franklin Institute*, vol. 347, no. 2, pp. 545–558, 2010.

[87] B. Chen, J. Hu, L. Pu, and Z. Sun, "Stochastic gradient algorithm under (h, φ)-entropy criterion," *Circuits, Systems & Signal Processing*, vol. 26, no. 6, pp. 941–960, 2007.

[88] M. Deb and T. Ogunfunmi, "Using information theoretic learning techniques to train neural networks," in *2017 51st Asilomar Conference on Signals, Systems, and Computers*, pp. 351–355, IEEE, 2017.

[89] D. Erdogmus, K. E. Hild, J. C. Principe, M. Lazaro, and I. Santamaria, "Adaptive blind deconvolution of linear channels using renyi's entropy with parzen window estimation," *IEEE Transactions on Signal Processing*, vol. 52, no. 6, pp. 1489–1498, 2004.

[90] E. Wolsztynski, E. Thierry, and L. Pronzato, "Minimum-entropy estimation in semi-parametric models," *Signal Processing*, vol. 85, no. 5, pp. 937–949, 2005.

[91] D.-T. Pham and F. Vrins, "Local minima of information-theoretic criteria in blind source separation," *IEEE Signal Processing Letters*, vol. 12, no. 11, pp. 788–791, 2005.

[92] I. Santamaría, D. Erdogmus, and J. C. Principe, "Entropy minimization for supervised digital communications channel equalization," *IEEE Transactions on Signal Processing*, vol. 50, no. 5, pp. 1184–1192, 2002.

[93] B. Chen and J. C. Principe, "Some further results on the minimum error entropy estimation," *Entropy*, vol. 14, no. 5, pp. 966–977, 2012.

[94] F. M. Dekking, C. Kraaikamp, H. P. Lopuhaä, and L. E. Meester, *A Modern Introduction to Probability and Statistics: Understanding why and how*. Springer Science & Business Media, 2005.

[95] R. Sen, H.-F. Yu, and I. Dhillon, "Think globally, act locally: A deep neural network approach to high-dimensional time series forecasting," *arXiv preprint arXiv:1905.03806*, 2019.

[96] Q. Lou and L. Jiang, "She: A fast and accurate deep neural network for encrypted data," *arXiv preprint arXiv:1906.00148*, 2019.

[97] A. Siddique, S. Oymak, and V. Hristidis, "Unsupervised paraphrasing via deep reinforcement learning," in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 1800–1809, 2020.

[98] P. Dütting, Z. Feng, H. Narasimhan, D. Parkes, and S. S. Ravindranath, "Optimal auctions through deep learning," in *International Conference on Machine Learning*, pp. 1706–1715, PMLR, 2019.

[99] D. Zou and Q. Gu, "An improved analysis of training over-parameterized deep neural networks," *arXiv preprint arXiv:1906.04688*, 2019.

[100] Y. Xu, P. Cao, Y. Kong, and Y. Wang, "L_dmi: A novel information-theoretic loss function for training deep nets robust to label noise.," in *NeurIPS*, pp. 6222–6233, 2019.

[101] S. Oymak and M. Soltanolkotabi, "End-to-end learning of a convolutional neural network via deep tensor decomposition," *arXiv preprint arXiv:1805.06523*, 2018.

[102] L. Chen, H. Qu, and J. Zhao, "Generalized correntropy based deep learning in presence of non-gaussian noises," *Neurocomputing*, vol. 278, pp. 41–50, 2018.