SYNCHRONIZATION TECHNIQUES FOR COHERENT UNDERWATER ACOUSTIC RECEIVERS

by

Xiao Liu

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy

at

Dalhousie University
Halifax, Nova Scotia
February 2019

Table of Contents

List of Tables

List of Figures

Abstract

In wireless communication systems, synchronization is one of the most important issues. The requirement for synchronization is especially intensified when there is strong channel distortion. For instance, the Doppler effect can shift the carrier in the frequency domain and scale the signal in the time domain. Similarly, the multipath propagation channel poses problems to the conventional synchronization methods.

This dissertation studies the synchronization techniques, including the symbol timing and carrier frequency recovery for coherent wireless receivers. The application is focused on the underwater acoustic communications, where time-varying multipath fading dominates the channel characteristics. The conventional synchronization techniques are generally derived based on the maximum likelihood principle, such that the second order statistics of the received data are utilized. However, this may not be an optimum solution in fading channels.

In this work, a new entropy-based synchronization criterion is explored. Synchronization is achieved by minimizing the entropy estimated from the eye diagram and the constellation diagram. Key implementation details are addressed towards the realization of entropy based synchronization algorithms. In addition, the performance is evaluated in controlled conditions. It is shown that entropy minimization has great potential and offers certain advantages for synchronization in wireless communication, particularly for pulse shaping filters with small excess bandwidth, as well as in multipath fading channels.

Furthermore, the latest deep learning technique is applied to synchronize the baseband signal. A neural network based coherent receiver is designed. Unlike the conventional receiver which consists of a series of function blocks, the neural network based receiver does not explicitly implement any function blocks. Its function is trained from end to end to achieve over all optimization. As such, this new receiver structure has the potential to outperform the conventional receivers in nonlinear or nonparametric propagation channels.

Acknowledgements

First, I would like to thank my supervisor, Dr. Jean-François Bousquet, for giving me the opportunity to have a journey through my doctoral research program. His continuous help and support allow me to explore a challenging engineering problem with confidence and courage.

I would also thank other members of my supervisory committee, Dr. Jacek Ilow, Dr. David Barclay, for their helpful and constructive comments and contributions on my thesis work.

I am thankful to Hui Xiong and Alireza Karami who joined the doctoral program in the same time as me, and shared many insights with me. Thanks also go to other graduate students in UW-Stream lab and in our department. They are Graham McIntyre, Habib Mirhedayati Roudsari, Andrew Dobbin, Afolarin Egbewande, Hossein Ghannadrezaii, Ningcheng Gaoding, Sri Raghu Padyath, Danqing Yin, Wan Peng. This is not a complete list, but I thank you all for the help and friendship.

Finally, I would like to thank my family members for all the love and support they have provided. Special thanks go to my wife Qi Kong. Her constant supports are always my power to make progress.

Taking a PhD is no doubt a bitter journey, therefore, these help and support is essential to me. Taking a PhD also refined my knowledge and expanded my horizon. Every time I thought I could solve a problem, an unknown frontier emerges. Now it is time to make a conclusion, and this dissertation is a summary of my years of work. I wish it would be helpful to the advance of technology.

Chapter 1

Introduction

1.1 Background

This dissertation is motivated by the demand for high speed, reliable underwater acoustic communication. Using acoustic propagation, the communication quality is highly affected by the acoustic channel conditions. Specifically, the acoustic propagation channel in the ocean is dispersive in both time and frequency domain. This is due to the channel characteristics such as multipath propagation, time variability and Doppler effect. The symbol timing and carrier frequency recovery that are investigated in this dissertation are key techniques that can maintain the reliability of the underwater acoustic communication link by mitigating the channel dispersion.

These tasks are approached in this dissertation in two different ways. One relies on information theory, which has been studied by researchers for decades. The second method is based on the fast development of deep learning techniques, which draw huge attention in the last few years. In this section, the background of underwater acoustic communication is reviewed, and the basics of deep learning techniques are introduced.

1.1.1 Underwater Acoustic Communication

The Earth's oceans are often recognized to be the last frontier. The lack of air and high pressure makes it inhospitable for human beings; therefore, it remains extremely difficult for people to explore. However, there is an increasing level of interest to deploy instruments below the sea. For example, oceanographers and geologists wish to track the underwater ecosystem activities and the seismic activities. Also, the oil and gas industry is deploying colossal platforms to extract natural resources. Finally, there is also a need from the military to survey our littoral coast for suspicious activities.

Because of the vastness of the ocean, combined with the fact that there is no

infrastructure to transfer data, a key challenge in the exploration of the oceans is to establish wireless communication from or between the instruments deployed in the sea without the burden of heavy cables.

Commonly used electromagnetic (EM) transmission in free space is not suitable for long distance underwater communication, because its range is significantly restricted by the attenuation introduced by the high permittivity and high conductivity characteristics of seawater [37]. Radio communication, although it has renowned merits in the terrestrial wireless network field, has only be applied in short-range and high-bandwidth underwater communications systems. In 2006, an underwater microwave communication was demonstrated system with a transmission data rate of 500 kbps over 90 m [48]. This performance was improved to 10 Mbps over 100 m for underwater sensing networks in 2009 [55]. Also, a variant of EM communication, magnetic induction (MI) communication has also been applied under the water [18]. Nonetheless, acoustic transmission remains the preferable technique for long transmission range [10].

Optical wave or laser technology, which is known for its very high capacity, has also recently drawn increasing attention in the research community. The latest research developments can be found in [28, 59, 61], where the highest bit rate can reach up to 25 Gbps with 680-nm red-light vertical-cavity surface-emitting laser. However, the major drawbacks are the requirement of clear water, strict alignment of devices and very short reliable transmission distance (normally less than 10 meters). These requirements greatly limit its underwater application.

In comparison to the technologies described above, water is a relative ideal medium for acoustic waves to propagate for tens or even hundreds of kilometers [24], including in salty, fresh or even muddy water. For this reason, the acoustic wave or sound wave is the preferred technology for wireless underwater communication.

Similar to other telecommunication systems, the physical layer of the underwater acoustic communication system is responsible for the transmission and reception of the modulated acoustic wave between devices in the water medium, and its goal is to deliver the data through the channel with a high degree of reliability. A clear understanding of the underwater channel physics and dynamics is critical for the design of the transmitter and receiver. Therefore, the next section will provide a high

level review on the underwater acoustic channel characteristics.

1.1.2 Underwater Acoustic Propagation Channels

The fundamental feature that differentiates underwater acoustic communication with other wireless communication is the acoustic propagation channel. In other words, the acoustic propagation channel defines the design of underwater acoustic communication. Therefore, it is critical to understand the behaviour of the channel before building underwater acoustic communication systems. A comprehensive description of underwater acoustic channels can be found in [51]. In short, the dynamic nature makes the channel *doubly dispersive*, with multipath propagation effects causing time dispersion and Doppler shifts causing frequency dispersion. Below is a brief introduction of the four major channel characteristics.

Bandwidth

In an unbounded media, the acoustical signal propagates spherically, and the power decays with the square of the distance. The path loss in an acoustic channel also depends on the signal frequency. For frequencies above a few tens of kHz, absorption due to chemical relaxation reduces the energy of the signal as the frequency increases. As a result, the acoustic propagation is preferred at low frequencies, and the available bandwidth for communication is extremely limited in comparison to that of RF communication. To achieve reasonable throughput, acoustic transmission bandwidth is often more than 10% of the carrier frequency. Since the bandwidth is not negligible with respect to the centre frequency, the underwater acoustic channel is in fact wideband. This channel characteristic makes the communication system sensitive to the multipath channel condition.

Multipath

A notable property of underwater acoustic channels is that the sound can be reflected by the surface, bottom and other objects. For propagation in deep water conditions, the sound path bends due to the inhomogeneity of the water, which is known as the refraction phenomenon. Because of the above two effects, the signal at the receiver

is the sum of copies from all paths with independent delays and magnitudes. In addition, the sound speed in the water is very low, about 1500 m/s, therefore, the delays between each path arrival spread over tens or even hundreds of milliseconds, resulting in frequency-selective signal distortion. To model such a channel, a common practice is to use a FIR filter, so that the overall impulse response that is contributed by p path arrivals is given by

$$h(t) = \sum_{p} h_p(t - \tau_p), \tag{1.1}$$

where h_p is the signal amplitude from each path. The corresponding path delay τ_p is obtained by $\tau_p = l_p/c$, where l_p is the path length, and c is the sound speed in the water. The sound speed is usually variable as a function of depth, temperature and salinity, but in the shallow water, it can be considered as a constant value approximately equal to 1500 m/s for convenience of calculation.

Small Scale Time Variation

The channel is subject to large-scale and small-scale time variation [43]. However, only the small-scale or short-time variation is often considered within one transmission. It comes from two sources: the changes in the propagation medium and those that occur due to the relative motion between the transmitter and the receiver. The latter cause will be discussed next. Short-time variation is usually introduced by surface waves and the water current, which cause the displacement of the scatterers and reflectors. This results in the change of path length with respect to time. Unlike a RF channel, where several statistical models are available to characterize the channel probability distribution, there is no consensus on statistical characterization of acoustic communication channels [51].

Experimental results suggest that some channels exhibit Rician or Rayleigh fading [6]. Both models describe the channel impulse response as a Gaussian process irrespective of the distribution of the individual components. If there is no dominant path, the envelope of the channel response is Rayleigh distributed. In comparison, the Rician fading occurs when one of the paths, typically a line of sight signal or some strong reflection signals, is much stronger than the others.

Doppler Scaling

The relative motion between the transmitter and the receiver results in the received signal being subject to a shift in the frequency domain as well as a scaling in the time domain (wave length). This phenomenon is known as Doppler scaling, and it contributes to an additional change in the channel response with respect to time. The Doppler scaling caused by motion is illustrated in Fig. 1.1.

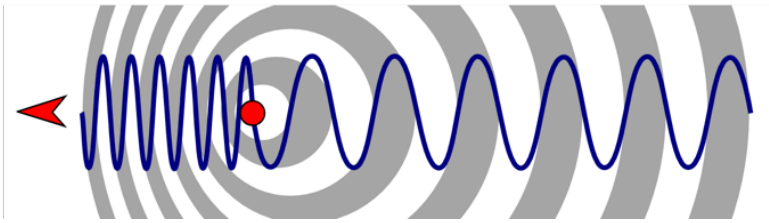


Figure 1.1: Doppler scaling caused by motion.

The scaling factor is found by

$$\alpha = \frac{v}{c}, \tag{1.2}$$

where v is relative velocity (positive if the receiver is moving towards the source), and c is the sound speed. Then, the observed frequency is given by

$$f' = (1 + \alpha)\, f, \tag{1.3}$$

where f is the source frequency. Because the speed of sound c is very low, α is usually not small enough to be negligible.

Especially for coherent communications, the symbol phase is extremely sensitive to the change of the carrier frequency. Therefore, even with small Doppler scaling, the reliability of the communication fails if there is no compensation. For an underwater acoustic communication application using fixed nodes, the Doppler effect can still exist because of drifting with waves or currents, as well as water surface motion. Fig. 1.2 is an example that shows how the impulse response of an underwater acoustic channel evolves with time due to the motion.

1.1.3 Deep Learning and Machine Learning

This dissertation is trying to solve the synchronization problem brought by the acoustic propagation channels through two approaches. One is based on the information

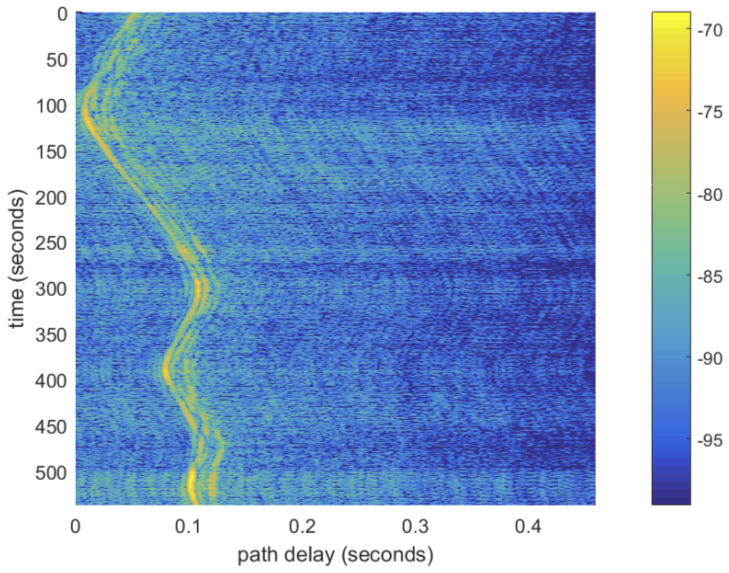


Figure 1.2: Channel impulse response evolving with time due to motion.

theory, which is well documented in many literature. The other method is to apply the latest deep learning techniques, which has become one of the most popular and fast developing technologies, and is applied in a ubiquitous fashion.

In 2018, self-driving vehicles have become available, and our daily routines are being organized by virtual assistants. As such, a society managed using artificial intelligence (AI) can be envisaged for the near future. This section will briefly introduce the basic concept of AI, machine learning and deep learning.

Artificial Intelligence and Machine Learning

The concept of AI was born in the 1950s, when Turing proposed to make computers "think" like a human being. AI is a general field that encompasses machine learning and deep learning, but that also includes many more approaches.

Machine learning is trying to make the computer learn on its own to solve certain problem without specific human orders. Therefore, the programmers don't need to write data processing rules by hand, but the computer can automatically learn the rules by examining the data. This is quite different from the conventional data processing programming, where humans input rules and data to be processed, and the computer gives answers. In comparison, using machine learning, humans input data

and the answers expected from the data, and the computer gives rules. Then, the rules can be applied to new data to generate answers.

Machine learning has a short history, but it has become the most popular and successful subset of AI thanks to the availability of large data set and fast computing platforms. A machine learning algorithm searches for useful representations of some input data, within a predefined space of possibilities, using guidance from a feedback signal.

During the short history of machine learning, a few algorithms have been introduced [17]. The logistic regression is one of the earliest forms of machine learning, and it is still widely used to this day. It may be the first algorithm a data scientist will try on a dataset to get a feel for the classification task at hand. Later, the early-stage neural networks, or artificial neural networks (ANN), were invented during mid-1980s to 1990s. Although it has been replaced by its modern variants, some of its core ideas are still kept today. The support vector machine (SVM) with kernel methods was considered the most powerful machine learning algorithm until around 2010, and then, the decision trees, such as the random forests, and gradient boosting machines took its place. Until today, the random forests algorithms are the first choice as a shallow learning technique for nonperceptual data processing.

The reason that these algorithms do not perform well for nonperceptual data processing is that the inputs of these algorithms listed above are usually the features extracted from a given dataset. The process of feature extraction from the dataset is a crucial step and it is known as the "feature engineering". The feature extraction is usually done manually, because classical shallow algorithms do not have hypothesis spaces rich enough to learn useful features by themselves. However, the feature engineering is rather difficult for perceptual data processing.

Deep Learning

Deep learning is a subset of machine learning, and its relationships between AI and machine learning are shown in Fig. 1.3. It has not been researched actively until early 2010s, then, it became a revolution. Today, it is a highly preferred algorithm in the field.

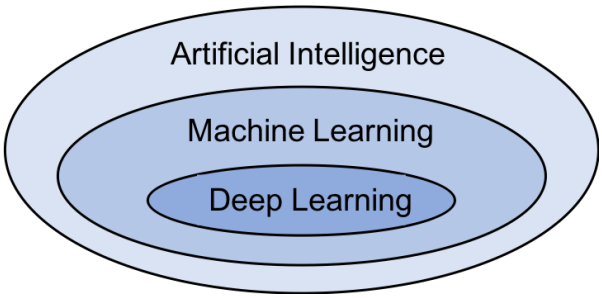


Figure 1.3: Artificial intelligence, machine learning and deep learning.

Deep learning features successive layers, and the number of layers, usually tens or hundreds, is called the depth of the model. In contrast, other machine approaches (shallow learning) normally have one or two layers. Almost all layers in deep learning consist of artificial neural layers, so the deep learning always refers to the deep neural networks. Fig. 1.4 shows the difference between a shallow neural network and a deep neural network.

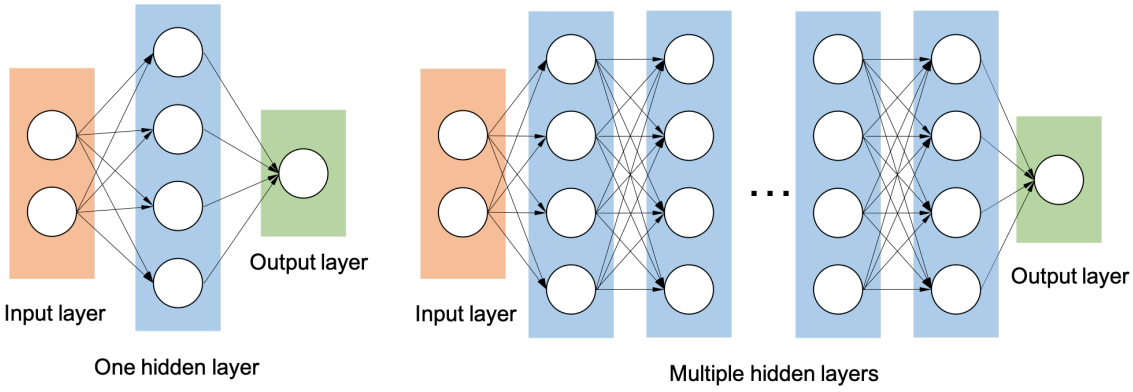


Figure 1.4: Example of a shallow neural network (left) and a deep neural network (right).

The increasing number of layers enables the models to have hypothesis spaces rich enough to learn useful features by themselves. As a result, the deep learning technique brought remarkable results on problems such as image recognition [53] and natural language processing [9] that are natural to humans but tough to machines.

The reasons for which the capability of deep learning techniques have grown fast only in recent years are: 1) the availability of graphical processing units (GPUs) that excel at massively parallel computing; 2) the availability of a large amount of labeled data that can be used for network training; and 3) more sophisticated algorithms are

developed to help the training of deep learning models.

1.2 Problem Statement

In a wireless communication system, the purpose of the physical layer is to deliver raw data through a dispersive channel as accurately as possible. This includes the study of the transmitter, the channel characteristics and the receiver. This dissertation mainly focuses on the signal processing at the receivers, and focuses especially on the synchronization problem. In this section, a standard receiver structure to mitigate distortion in a doubly dispersive channel is introduced in Section 1.2.1, and the need for symbol timing and carrier recovery are detailed in Section 1.2.2. Specifically, the need for synchronization is intensified when there is channel introduced distortion (such as multipath and Doppler).

1.2.1 A Standard Coherent Receiver Structure

In this dissertation, a single carrier, phase coherent communication system is utilized, because of its widespread use in underwater acoustic modems [50]. The transmit signal has two features: 1) the information is modulated to the carrier's phase symbol by symbol, and 2) the bandwidth is limited by a pulse shaping filter at the transmitter. A standard digital receiver (without the synchronizer) has a sequential structure as shown in Fig. 1.5.

Figure 1.5: A standard coherent receiver structure.

The roles of each these functional blocks to process the data in a digital modem are introduced briefly here.

1. **Analog-to-digital converter (ADC)**: The continuous-time signal is sampled and discretized at the receiver front-end by an ADC. The sampling is triggered by a local clock.

2. **Down converter**: The down converter converts the passband signal to complex-valued baseband by multiplying the signal with a pair of locally generated orthogonal carriers.

3. **Receiver filter**: Usually a root raised cosine (RRC) filter matched to the the transmitter filter is applied at the receiver for maximizing the signal-to-noise ratio (SNR).

4. **Equalizer**: An equalizer is required to mitigate the multipath introduced inter-symbol interference (ISI).

5. **Demodulator**: The demodulator maps the complex-valued baseband data samples into bit streams based on the modulation schemes.

1.2.2 Symbol Timing and Carrier Recovery

Ideally, the sampling clock of the ADC is synchronized to the signal symbols. To achieve maximum noise immunity, the samples upon which the receiver's decision is based should be taken in the middle of the shaping pulse, or in other words, at the instants of maximum eye opening. However, the ADC has a free-running sampling clock, and therefore, are not synchronized to the symbols. Consequently, the receiver must include a timing recovery block which makes an estimate of the symbol clock, compensates the error digitally and applies the estimate before making decisions.

To demodulate coherent signals, such as phase-shift keying (PSK) and quadrature amplitude modulation (QAM), the receiver must generate a local carrier that has the same frequency and initial phase as the received signal. If the initial phase is not accurate, the baseband signal has a constant phase offset, which is relatively easy to correct. However, if there is a frequency offset, the phase error of the baseband signal varies as a function of time, making it impossible to demodulate the correct symbol, especially when the frequency offset is time-varying.

The time-varying symbol delay and carrier frequency offset is particularly common in mobile channels, and is particularly apparent for the underwater acoustic channel. The relative motion between the transmitter and the receiver causes a scaling of symbol period in the time domain and a shift of the carrier frequency because of

the Doppler effect. Therefore, a continuous estimation and compensation of the synchronization error is essential to maintain the link reliability. Note that the term *synchronization* used in this dissertation only refer to the symbol timing and carrier recovery if without further explanation.

The received signal after the multipath propagation is a summation of arrivals from multiple paths with various delays and amplitudes. The synchronization under such extreme conditions are rarely studied. Note that the fractionally-spaced decision feedback equalizer (FS-DFE) can compensate for the symbol timing and carrier frequency offset in multipath conditions. However, there are some drawbacks that are difficult to overcome. These are:

1. The additional synchronization task increases the computational load of an equalizer and makes it difficult to converge and track the time-varying channel. Especially when there is strong Doppler effect, the equalizer window may become invalid after a long period of time [49].

2. The fractionally-spaced feedforward filter that intends to compensate for symbol timing offset significantly increases the order and the complexity of the equalizer.

3. The FS-DFE is not sensitive to the symbol timing phase, or a constant timing offset within a symbol period. However, it does not consider the time domain dilation of the signal. A leading symbol rate recovery is required to prevent failure of tracking symbols if operating in strong Doppler conditions.

Therefore, an FS-DFE cannot totally replace a synchronizer in a wireless receiver.

1.3 Contributions

There are two major contributions in this dissertation. They are the study of entropy minimization as a synchronization criterion [29,30], and the application of deep learning technique for wireless receivers. Other contributions and studies can be found in Chapter 2 and in [2, 31].

Synchronization based on entropy minimization

A unified synchronization criterion, relying on entropy minimization (EM) as an alternative to the maximum likelihood (ML) criterion is studied in this dissertation. Specifically, the entropy of the eye diagram is evaluated for symbol timing recovery, and the entropy of the constellation diagram is measured for carrier frequency recovery. For both applications, the synchronization parameter that leads to a minimum entropy value is considered to be optimum.

For demonstration purposes, a custom estimation algorithm is provided to evaluate the entropy of the signal eye diagram and constellation. Many practical issues such as the undesired local minima, insufficient oversampling rate and the vanishing of the gradient are addressed to improve the practicability of the algorithm.

The performance of the proposed symbol timing and carrier frequency recovery criterion is also evaluated in controlled conditions. The effects of various channel impairments, including noise and multipath, are analyzed on the system performance. Particularly, in multipath fading and small excess bandwidth conditions, the timing recovery using the EM algorithm can significantly improve the equalizer's convergence, and its symbol error rate outperforms that of the ML algorithm.

Deep learning for wireless receivers

In recent years, deep learning has received significant attention. However, it is rarely applied to solve problems in wireless communications. In this dissertation, the deep convolutional neural networks (CNN) is applied to solve the symbol timing and carrier frequency problems. In addition, a coherent receiver design is proposed, which is entirely based on CNN.

The neural networks for synchronization are trained with supervised learning. The training process does not presume any channel conditions or sampling rate but only learning from the training data set. The training set can be either synthesized from numerical simulation or measured from all kinds of realistic channel conditions. Therefore, such an algorithm is more versatile, providing significant advantage compared to the conventional algorithms, particularly for extremely varying conditions

The network for symbol timing offset estimation uses the magnitudes of the baseband sample sequences as the input. It is also insensitive to the carrier frequency offset. The network for carrier frequency recovery uses timing recovered samples as the input to estimate the frequency offset. Both estimation networks have similar structure and process data in a feedforward manner. To test the feasibility of such neural networks, the sea trial measurement data are used for verification. The results show that the proposed CNN based synchronization algorithms have comparable performance to the conventional algorithms.

Deep neural networks are considered as a type of complex nonlinear function. Similarly, a standard wireless receiver that is composed of several function blocks can also be treated as one nonlinear function. Its input is the passband signal and the output is the demodulator decisions. Based on this analogy, a wireless receiver that is entirely redesigned using CNN is proposed in this dissertation. Unlike the standard CNN structure for image processing, several modifications have been made to accommodate the nature of coherent communications. The symbol timing is achieve implicitly, similar to that in the fractionally-space equalizer. However, the real challenges come from the carrier frequency recovery and equalization, and they are addressed carefully in the dissertation.

1.4 Outline

There are two fundamental synchronization problems encountered in wireless coherent communication systems: symbol timing and carrier recovery. This dissertation addresses these two problems for coherent communication, where the modulation scheme embeds the information in the symbol phases.

Chapter 2 summarizes the current synchronization techniques. These algorithms are generally categorized by the need of reference data and their structures. This chapter also covers the topic of the cost function, the algorithm complexity, as well as the figure of merit for performance evaluation. In addition, an original feedforward timing recovery circuit structure is presented. Also, a classic receiver structure, fractionally-spaced equalizer with build-in phase lock loop (PLL), is implemented to recover the sea trial measurement data. This is an especially powerful receiver

structure for severe multipath and Doppler effect dominated channels.

Chapter 3 presents a new entropy minimization criterion for both symbol timing and carrier frequency recovery for wireless receivers [29]. Synchronization is achieved by minimizing the entropy estimated from the eye diagram and the constellation diagram. Key implementation details are addressed towards the realization of entropy based synchronization algorithms. In addition, the performance is evaluated in controlled conditions. It is shown that, as an alternative to the maximum likelihood criterion, entropy minimization has great potential and offers certain advantages for synchronization in wireless communication, particularly for pulse shaping filters with small excess bandwidth, as well as in multipath fading channels.

Chapter 4 takes one step forward and bring the latest deep learning techniques to wireless communications as an advanced signal processing method. An overview of the deep learning methodology is given in the beginning of the chapter. Then, the symbol timing and carrier frequency recovery problems are reengineered using the CNN models. Customized CNN models are designed to accommodate the characteristics of coherent communications. Next, a new coherent wireless receiver entirely based on CNN models is proposed. The major issues including carrier frequency compensation and model training are addressed, and the performance is tested with both synthesized and measurement data.

Chapter 5 summarizes and concludes the whole dissertation.

Chapter 2

Symbol Timing and Carrier Recovery for Coherent Receivers

In this chapter, three aspects of conventional synchronization techniques are discussed. Section 2.1 reviews the fundamentals of the conventional symbol timing and carrier recovery method. It begins with the synchronizer structure, and then focuses on the estimation and error detection algorithms. For both synchronization tasks, the complexity and the performance are analyzed. Section 2.2 demonstrates a feedforward synchronization implementation. It is designed to compensate for the Doppler effect introduced time scaling and carrier frequency offset at the time. Section 2.3 presents a realistic underwater acoustic communication example. The channel introduced distortion is analyzed with the received signal. A classic coherent receiver that is specifically designed for such a scenario is implemented and tested. Note that all the synchronization algorithms and the corresponding receivers discussed are digitally implemented in discrete-time.

2.1 A Review of Standard Synchronization Techniques

2.1.1 Fundamentals of Synchronization

In this section, the system architecture will be presented first, and then, maximum likelihood estimation will be discussed. Finally, the figures of merit will be introduced.

System Architecture

Several issues must be considered for the implementation of the synchronizer at a coherent wireless receiver. If the expected symbol timing or frequency offsets are very large, a coarse recovery is required to reduce the error before fine tuning [47]. Also, the order of the two synchronization tasks, symbol timing and carrier recovery, is generally interchangeable. However, as a convention, the timing recovery is always assumed to be realized before the carrier frequency recovery in the following discussion, unless

explicitly mentioned. This is because many timing recovery schemes are insensitive to the carrier frequency offset and the carrier recovery schemes can be implemented at the symbol rate.

The most important issue is the choice of configuration and algorithms of the synchronizer. There are fundamentally two types of synchronizer configurations: feedforward and feedback. The feedforward structure directly extracts the synchronization parameter from the incoming signal x with a symbol timing or carrier frequency offset estimator, and applies the correction to the data samples, as shown in Fig. 2.1 The objective is to make the estimate γ as close to its true value as possible. This configuration offers a fast acquisition speed but may be slow for tracking in time varying environment, therefore, it is best suit for burst communications.

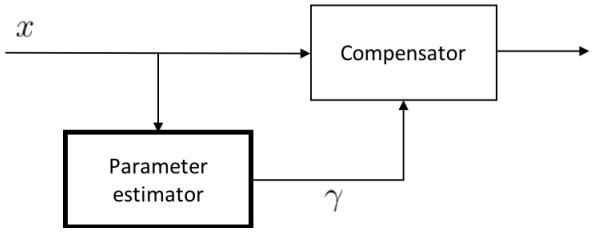


Figure 2.1: A general feedforward synchronizer configuration.

On the other hand, the feedback configuration is known for its good tracking performance, so it is usually applied in continuous communications. Instead of the direct estimation of the synchronization parameters, it feeds the error signal back to correct the received signal as shown in Fig. 2.2. The objective is to achieve synchronization by minimizing the error signal. The feedback configuration features an error detector, which produces an error signal e proportional to the difference between the local estimation and the actual synchronization parameter. The error signal is filtered by a loop filter to reduce the error variance, and the filter output γ is used to compensate for the received signal.

The estimator in the feedforward configuration and the error detector in the feedback configuration can be further classified depending on the availability of reference data: when reference data is available, data-aided (DA) estimation is preferred to optimize the accuracy of the estimator; whereas the non-data-aided (NDA) estimators are unaware of the information that was transmitted, and provide synchronization

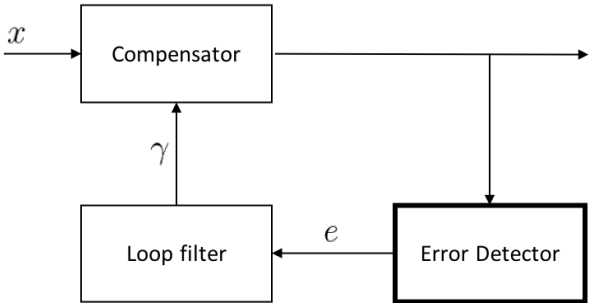


Figure 2.2: A general feedback synchronizer configuration.

using the spectral and temporal characteristics of the received signal. Decision-directed (DD) estimation is a variant of NDA method: it uses the decisions made at the receiver as the reference data. It has better performance than the ordinary NDA methods in benign conditions. However, it is highly subjective to the reliability of the decision device. As such, in practical scenarios, it can be unstable.

Maximum Likelihood Estimation

In [32], it is indicated that most of the existing synchronization algorithms can be derived from the maximum likelihood (ML) criterion or its approximation. A simple explanation of the ML based synchronization parameter estimation is given below. First, the received signal $r(t)$ is modelled as the sum of transmit signal and the additive noise, as

$$r(t) = s(t, \boldsymbol{\gamma}) + w(t), \tag{2.1}$$

where $s(t, \gamma)$ is the transmit signal as a function of time, and $\boldsymbol{\gamma}$ is a set of unknown synchronization parameters, including the carrier frequency offset f_Δ, carrier phase offset θ and the symbol timing offset τ. The noise is denoted as $w(t)$.

The synchronization parameter set $\boldsymbol{\gamma}$ is obtained with the observation $r(t)$. One popular and intuitive approach to this problem is based on the ML criterion. For a trial $\tilde{\boldsymbol{\gamma}}$, the estimated received signal is defined as $\tilde{s}(t) = s(t, \tilde{\boldsymbol{\gamma}})$, where the tilde indicates a trial value. So the estimated received signal with the synchronization parameters is modelled as

$$\tilde{s}(t) \triangleq e^{j(2\pi \tilde{f_\Delta} t + \tilde{\theta})} \sum_i c_i g(t - iT - \tilde{\tau}), \tag{2.2}$$

where c_i is the i-th modulated symbol, $g(t)$ is a pulse shaping filter that used in the transmitter and T represents the symbol period. The resemblance between $\tilde{s}(t)$ and $r(t)$ is expected to be maximized. The most likely value of $\boldsymbol{\gamma}$ is obtained when the probability $p(r|\tilde{\boldsymbol{\gamma}})$ achieves a maximum. Hence, the ML estimate of $\boldsymbol{\gamma}$ is expressed as

$$\hat{\boldsymbol{\gamma}}_{ML}(r) = \arg\left\{ \max_{\tilde{\boldsymbol{\gamma}}} \left\{ p(r|\tilde{\boldsymbol{\gamma}}) \right\} \right\}. \tag{2.3}$$

To be specific, if the noise is assumed to be Gaussian with a spectral density N_0, and the observation interval is $0 < t < T_0$, without detailed derivation, the *likelihood function* for carrier frequency recovery that is maximized is given by [33]

$$\Lambda(r|\tilde{f}_\Delta, \tilde{\theta}) = \exp\left\{ \frac{1}{N_0} \int_0^{T_0} \mathrm{Re}\left\{ r(t)\tilde{s}^*(t) \right\} dt - \frac{1}{2N_0} \int_0^{T_0} |\tilde{s}(t)|^2 dt \right\}, \tag{2.4}$$

where $\{\}^*$ is the complex conjugate operator. The first integral in (2.4) is the *inner product* of $r(t)$ and $\tilde{s}(t)$, and the inner product represents the resemblance of the two signals, while the second integral is simply the average transmit signal energy.

For symbol timing recovery, the likelihood as a function of the trial timing offset $\tilde{\tau}$ is given by

$$\Lambda(r|\tilde{\tau}) = \exp\left\{ \frac{1}{N_0} \int_0^{T_0} r(t)\tilde{s}(t) dt - \frac{1}{2N_0} \int_0^{T_0} \tilde{s}^2(t) dt \right\}. \tag{2.5}$$

It can be found that (2.5) has a very similar format to (2.4).

ML based synchronization estimators are derived from the likelihood functions in various approaches. For example, the second integral part might be dropped in the carrier frequency recovery since it is independent of the synchronization parameters. Also, when deriving many types of NDA symbol timing error detectors [15], the second integral part is dropped for simplification purpose, but it will bring high level self noise in return [33]. Moreover, when searching for the maximum point of a likelihood function, one common approach is to take the derivative of likelihood function. The maximum occurs with the vanishing of the derivative with respect to the synchronization parameters. This approach is the theoretical origin for many error detectors, and particularly used in feedback configurations.

It should be stressed that the ML estimation assumes that the additive Gaussian noise is the only channel distortion, in other words, the channel is modeled using

second order statistics. However, in doubly dispersive channels, the second order statistics based ML estimation may not lead to the optimal solution. This issue will be future discussed in the next chapter.

Performance and Limitation

A figure of merit must be defined to evaluate the accuracy that can be achieved by a synchronizer. The accuracy limit shall be established, such that it can provide benchmarks for the synchronizer implementations. From the parameter estimation theory, this limit is known as the Cramer-Rao bound (CRB) [23]. However, the true CRB is difficult to obtain when applied to synchronization problems. Therefore, the modified Cramer-Rao bound (MCRB), a variant to the CRB is adopted [34]. Similar to the previous discussion on the ML estimation criterion, only a brief overview is given here, and a more rigorous demonstration can be also found in [33].

Consider an estimation procedure for a synchronization parameter $\tilde{\gamma}$, and the corresponding estimate $\hat{\gamma}(r)$, where r is the observation. The estimate $\hat{\gamma}(r)$ is a random variable depending on the different observations r, and its expectation coincides with the true value of τ if it is an unbiased estimation. Being unbiased means that on average, the estimator yields the true value of the unknown τ.

As a first order statistic, the unbiased expectation is not a concern for most estimators, whereas the second order statistic of the estimation error is often used as the figure of merit to evaluate a estimator's performance. The errors $\hat{\gamma}(r) - \gamma$ usually scatter around zero, and its dispersion is quantified by the *variance*. The minimum error dispersion that can be achieved is given by the Cramer-Rao bound, which is a lower limit to the variance of any unbiased estimator. This bound is given by

$$\mathrm{Var}\left\{\hat{\gamma}(r) - \gamma\right\} \geq CRB(\gamma), \tag{2.6}$$

However, this limit is not quite feasible, and as indicated in [8], this drawback is resolved by the modified Cramer-Rao (MCRB) bound.

The MCRB for the carrier frequency estimation is given by

$$MCRB(f_\Delta) = \frac{3}{2\pi^2 L_0^3 T^2 E_s/N_0}, \tag{2.7}$$

where L_0 symbols with period T are used for estimation, and the energy per symbol to noise power spectral density is denoted as E_s/N_0. On the other hand, the MCRB for the symbol timing offset estimation is given by

$$MCRB(\tau) = \frac{T^2}{8\pi^2 L_0 \xi E_s/N_0}, \tag{2.8}$$

where ξ is a function of the rolloff factor α and is defined as

$$\xi = \frac{1}{12} + \alpha^2 \left(\frac{1}{4} - \frac{2}{\pi^2} \right). \tag{2.9}$$

One direct conclusion that can be drawn from (2.7) and (2.8) is that the MCRB is inversely proportional to the signal-to-noise ratio E_s/N_0, as well as the the observation length L_0. Moreover, for symbol timing recovery, the MCRB is also inversely proportional to the signal bandwidth.

2.1.2 Symbol Timing Recovery

The purpose of symbol timing recovery is to recover a clock at the symbol rate or a multiple of the symbol rate from the received modulated waveform. The symbol clock is essential for demodulation but rarely transmitted with the signal. Therefore, it is more common to implement the additional circuit which can derive the clock from the received signal itself.

As discussed in Section 2.1.1, there are feedback and feedforward configurations for symbol timing recovery. There are four popular timing error detectors (TED) for the feedback configuration: Zero-Crossing, Mueller-Muller, Gardner, and Early-Late. The most widely used feedforward configuration is the O&M estimator. Note that all these algorithms assume that there is single significant path in the channel with additive Gaussian noise.

In the following discussion, only the input's in-phase component $x(kT + \hat{\tau}_k)$ is considered. For complex valued input, the expression can be easily expanded by adding a quadrature component.

Zero-Crossing Detector

The Zero-Crossing detector [15] is a DD technique that requires two samples per symbol at the synchronizer's input. The expression is given by

$$e(k) = (\hat{c}_{k-1} - \hat{c}_k)x(kT - T/2 + \hat{\tau}), \tag{2.10}$$

where $\hat{c}_k$ is the correctly estimated k-th decision symbol. It performs well in low SNR conditions for all values of excess bandwidth, and in moderate SNR conditions for large rolloff factors. But its timing variance increases significantly with a reduced rolloff factor.

Mueller-Muller Detector

The Mueller-Muller is a DD timing error that requires one sample per symbol at the synchronizer's input. When the input signal has Nyquist pulses, this method has no self noise. However, it is sensitive to the noise, especially when there is a large excess bandwidth. The expression is given by

$$e(k) = \hat{c}_{k-1}x(kT + \hat{\tau}) - \hat{c}_k x\big((k-1)T + \hat{\tau}\big). \tag{2.11}$$

Gardner Detector

The Gardner detector [14] is an NDA feedback method that is independent of carrier phase recovery. It requires two samples per symbol at the synchronizer's input. In the presence of noise, the performance of this timing recovery method improves as the excess bandwidth increases. Its expression is given by

$$e(k) = \big[x(kT - T + \hat{\tau}) - x(kT + \hat{\tau})\big]x(kT - T/2 + \hat{\tau}) \tag{2.12}$$

Early-Late Detector

The Early-Late detector is an NDA feedback method that requires two samples per symbol at the synchronizer's input. In the presence of noise, the performance of this timing recovery method improves as the pulse's excess bandwidth increases. Compared to the Gardner method, the Early-Late method has higher self noise and thus

does not perform as well as the Gardner method in systems with high SNR values. The expression is given by

$$e(k) = x(kT + \hat{\tau})\big[x(kT + T/2 + \hat{\tau}) - x(kT - T/2 + \hat{\tau})\big] \tag{2.13}$$

The performance of these four TEDs in terms of the timing error variance as a function of E_b/N_0 has been presented in the literature. One feature that should be stressed is that the timing error variance which is also known as jitter variance, has an irreducible error floor as E_b/N_0 increases. The explanation is that timing errors generate ISI which, in turn, produces decision errors even in the absence of noise.

O&M Timing Estimator

The O&M algorithm is a simple implementation of feedforward symbol timing estimation algorithm. It uses as low as 4 samples per symbol, and the timing offset is directly given by

$$\hat{\tau} = \frac{T}{2\pi} \arg\left\{ \sum_{i=1}^{N} x_i^2 e^{-j2\pi(i-1)/N_{sps}} \right\}, \tag{2.14}$$

where N samples with the oversampling rate $N_{sps} \geq 4$ are used. The operation $\arg\{\cdot\}$ returns the phase angles in radians. It has a similar performance as the Gardner detector with large rolloff factor, and superior performance with small rolloff factor. The O&M algorithm is generally the best choice when applying the feedforward symbol timing recovery.

2.1.3 Carrier Frequency Recovery

Due to the clock mismatch and the Doppler scaling, the local generated carrier is not equal to that of the received signal. As the frequency difference accumulates, it leads to an increasing phase error in the down-converted baseband signal. This issue is especially critical for coherent communications where the information is transmitted by the carrier's phase.

A well known feedback frequency tracking method is to implement a phase lock loop (PLL) with a phase error detector (PED), such that the frequency is recovered by phase compensation. However, a traditional PED can only find a phase error from a single tone without coherent modulation. Therefore, the core idea of designing

a PED, or a frequency estimator for feedforward carrier recovery is to remove the modulation.

Data-Aided Phase Error Detector

If the transmitted symbol is known, the modulation of the input sample can be removed by multiplying it with the complex conjugate of the data symbol. The DA PED is obtained in a straightforward fashion by

$$e_k = \arg\left\{x_k c_k^*\right\}. \tag{2.15}$$

The DA method is the first choice when the data is known, since it always outperforms the NDA or DD methods.

Power-of-N Phase Error Detector

Power-of-N carrier recovery is an NDA feedback method. An integer N is defined such that $c_k^N = 1$ for all c_k. For example, if the modulation scheme is M-PSK , then $N = M$. So the error signal using power-of-N method is given by

$$e_k = \arg\left\{\left(x_k x_{k-1}^*\right)^N\right\}, \tag{2.16}$$

where the purpose of $x_k x_{k-1}^*$ is to eliminate a constant symbol phase offset, and the power-of-N can remove the modulation. This PED can be also converted to a carrier frequency estimator for a feedforward configuration. The estimator's expression is given by

$$f_\Delta = \frac{1}{2N\pi T} \arg\left\{\sum_{k=1}^{L_0-1}\left(x_k x_{k-1}^*\right)^N\right\} \tag{2.17}$$

Note that the power-of-N operation also amplifies the noise, making it only suitable for high SNR conditions.

Decision-Directed Phase Error Detector

At high SNR, the DD carrier recovery usually provides better performance than those NDA methods, therefore, it is a standard PED in feedback carrier recovery.

The expression of a DD PED depends on the modulation scheme. For BPSK or PAM, the error signal is given by

$$e_k = \text{sgn}\left\{ \text{Re}(x_k)\right\} \text{Im}(x_k), \tag{2.18}$$

where sgn$\{\cdot\}$is the sign operator, which gives a decision from x_k. As for QAM or QPSK, the phase error is given by

$$e_k = \text{sgn}\left\{ \text{Re}(x_k)\right\} \text{Im}(x_k) - \text{sgn}\left\{ \text{Im}(x_k)\right\} \text{Re}(x_k). \tag{2.19}$$

Data-Aided Frequency Offset Estimator

Unlike the PED, the carrier frequency offset estimator is utilized in the feedforward configuration. As discussed before, the modulation can be removed by multiplying with the complex conjugate of the data symbol. Define $z_k \triangleq x_k c_k^*$; then, the autocorrelations of $z_k$is obtained by

$$R(m) \triangleq \frac{1}{L_0 - m} \sum_{k=m}^{L_0-1} z_k z_{k-m}^*, \tag{2.20}$$

where $1 \leq m \leq L_0 - 1$. So the Fitz's estimator is given by

$$f_\Delta = \frac{1}{\pi N(N+1)T} \sum_{m=1}^{N} \arg\left\{R(m)\right\}. \tag{2.21}$$

While the L&R's estimator is expressed as

$$f_\Delta = \frac{1}{\pi(N+1)T} \arg\left\{ \sum_{m=1}^{N} R(m) \right\}. \tag{2.22}$$

The detailed derivation and their performance analysis are omitted, since they can be found in [33], [35] and their references.

2.2 A Feedforward Synchronization Implementation

The previous section summarizes the standard synchronization techniques. In this section, a Doppler compensation implementation is presented to maintain a reliable communication link at the receiver in highly mobile condition. In Section 2.2.1, the effects of Doppler scaling on the synchronization are detailed. The standard feedback

timing recovery loop has been modified into a feedforward structure to achieve fast timing convergence and power efficiency in Section 2.2.2. The timing error is tracked by a Gardner detector through all samples before down-sampling. A control unit uses the timing error information to dynamically adjust the sampling time as well as the carrier frequency offset such that the Doppler shift is fully compensated. Finally in Section 2.2.3, the synchronizer's behaviour is tested in controlled conditions.

2.2.1 Doppler and its Effects on Synchronization

The Doppler effect has a significant impact on coherent underwater communication systems, particularly in mobile conditions. When sound wave propagates through water, the Doppler effect manifests itself in both frequency shift and time scaling. The Doppler scaling factor α is defined as the ratio of the speed of motion with respect to the velocity of sound. The expression can be found in (1.2), and the shifted frequency of received signal is given by (1.3). In passband communication, modulation with a carrier is used to transmit high speed data in a limited frequency band. At the receiver, an undesirable carrier frequency offset will be present at the output of the down-converter. Consequently, the constellation of the received data will rotate.

Excessive Doppler also results in a scaling of the waveform in the time domain. If the symbol period of the transmitted signal is T_0, the period of received signal T' is given by

$$T' = (1 - \alpha)T_0 \tag{2.23}$$

In other words, after $1/\alpha$ symbol periods, the down sampler will have one symbol offset, and it increases with time. To track the symbol timing at the receiver, after conversion to baseband, the received signal can be decimated or interpolated at the symbol rate to align at the center of the symbol period. However, in a time varying environment in which the Doppler scaling factor α is variable, the sampling will be shifting slowly off the ideal sampling instant due to time-domain compression or dilation. In highly mobile underwater acoustic applications, only multiplying a time dependent frequency shift $e^{j2\pi\alpha t}$ to the signal can not represent the effect of Doppler accurately, and the time scaling is a more accurate model. Note that time scaling is

a nonlinear process, that can excessively increase simulation time, so is often avoided in practice. In these conditions, to compensate the Doppler shift, symbol timing recovery must also be realized along with carrier frequency recovery [47].

In conventional DA Doppler estimation techniques, a known waveform is transmitted in the frame as a preamble. At the receiver, a bank of discrete correlators with different Doppler shifted replicas of the transmitted waveform are implemented. The branch yielding the largest correlation peak is then selected to determine the Doppler estimate [22]. It is a fast parallel algorithm but typically requires significant hardware resources. Another technique is to measure the duration of a received data packet by interleaving known waveforms before and after a data packet [47]. These DA technique are spectrally inefficient, particularly the latter, because the known waveform is repeated twice. More importantly, these algorithms do not allow tracking of the timing within a fraction of the symbol period during the frame acquisition.

2.2.2 A Feedforward Doppler Compensation Structure

The proposed Doppler compensator is shown in Fig. 2.3. In the upper branch, the signal is decimated to the symbol rate before carrier frequency recovery. A high oversampling rate is assumed, such that the symbol clock recovery does not need interpolation between samples. The lower branch in Fig. 2.3 includes the Gardner TED and a control unit. It shall be stressed that the Gardner TED is applied in a feedforward configuration. The Gardner TED feeds the error signal forward to the control unit. The control unit is programmed to adjust a counter which provides the down-sampler with sampling instances (strobes) at the symbol rate, such that it will sample at maximum eye opening and minimize the ISI. Moreover, the control unit also interprets the timing information to generate an estimate of the carrier frequency offset. The output from the down-sampler is weighed by the carrier frequency offset estimation to compensate it.

The features of the proposed Doppler compensation circuitry are: 1) a feedforward configuration, 2) carrier frequency offset estimation with symbol timing, and 3) a control unit for both symbol timing and carrier recovery. Note that in this design, a symbol error detector is used in a feedforward manner such that it can achieve a

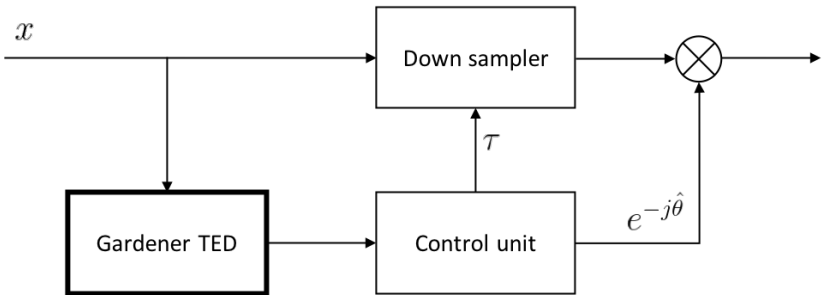


Figure 2.3: A feedforward Doppler compensation structure using Gardner detector.

faster convergence rate and keep a good tracking ability at the same time.

The S-curve provides the output of TED $S(\tau)$ at different sampling phase τ. For the Gardner TED, the S-curve is given by [33]

$$S(\tau) = K \sin\left(\frac{2\pi\tau}{T}\right) \tag{2.24}$$

where the ideal sampling phase is $\tau = 0$. The factor K is a function of symbol period T and rolloff factor. It can be treated as a constant for a certain signal setting. In this work, the errors fed by the Gardner TED to the control unit can be analyzed with the S-curve. A decision can be made when $S(\tau) = 0$. It indicates that $\tau = 0$ is reached at the current sample. However, the Gardner TED also obtains zero values at $\tau = \pm T/2$, in which case a false decision is made. Instead of looking for zeros, considering the sinusoidal curve in (2.24), the decisions should be made at zero-crossing points on the curve where the slope is positive. Another feature of the Gardner TED is that its accuracy relays on the symbol transition. If the symbols in the window of interest have the same polarity, the detector's output is not reliable.

In this work, to avoid false detection due to no symbol transition, the slope of the error is also evaluated, and compared to a threshold. The clock is re-aligned only when the TED error sequence (averaging may applied at low SNR) is crossing zero and that the slope is greater than the threshold.

There is a configurable counter inside the control unit that triggers the down-sampler. The counter is reset to zero once it reaches the oversampling rate, and at this instant, the down-sampler selects the current sample. The counter is useful when consecutive symbols have the same polarity, and there is no output from the Gardner TED.

Without Doppler scaling, the Gardner TED's decisions are always made when the counter is back to zero. In contrast, with Doppler scaling, the counter leaps forward or backward by 1 step (to avoid instability) to re-align its zero with the decision instant when they are not aligned. Through this process, although the scaled symbol rate may be a fraction multiple of the sampling rate, the average sampling frequency of down-sampler is adjusted to $1/T'$.

After symbol timing recovery, the next task is to compensate for the carrier frequency offset. There are several DA methods [12] [34] which require ideal symbol timing. Alternatively, in this work, the carrier frequency offset is estimated from the symbol period scaling.

From (2.23), α can be estimated using

$$\alpha = \frac{T_0 - T'}{T_0} \tag{2.25}$$

The Doppler scaling factor α can either be smoothed through several symbols or dynamically adjusted symbol by symbol. The output of the down-sampler is then multiplied with $e^{-j\alpha w_0 t}$ to compensate for the Doppler introduced carrier frequency offset.

2.2.3 Simulation Results

To verify the behaviour of the timing recovery circuit, the performance of the proposed implementation in an AWGN channel is modeled. To demonstrate Doppler compensation for a constant envelope modulation, QPSK is chosen here. An RRC pulse $g(t)$ with a rolloff factor 0.5 is employed. The Doppler scaling factor α is 1%, and the SNR in the channel is 10 dB. At the receiver, after coarse down-conversion and the matched filter, the baseband signal is processed by the Gardner TED.

Fig. 2.4 shows a 10-symbol window of the received baseband signal in the in-phase branch, as well as the Gardner TED's output and the "Set Counter" events that corrects the sampling decision instant. The the "Set Counter" events are located in the middle of symbols, and they only appear when the error sequence is crossing zero with a large positive slope. Between Symbols 6 and 7, there is no transition, leading the Gardner TED incapable of making a decision. That is where the free running counter inside the control unit becomes useful.

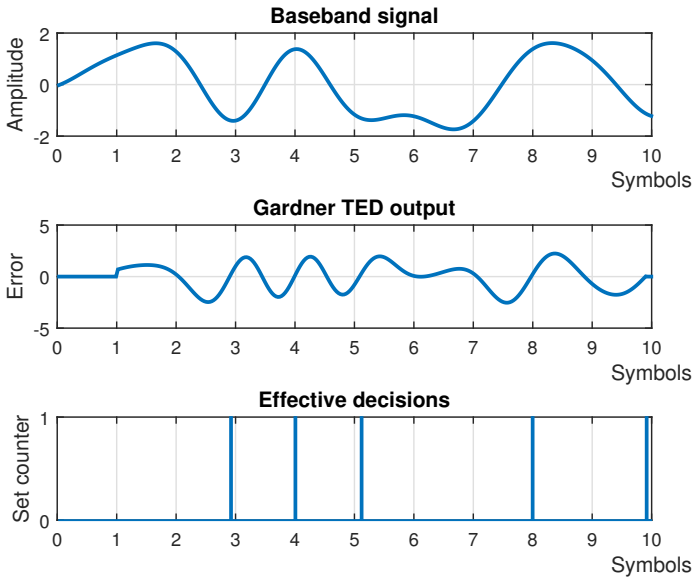
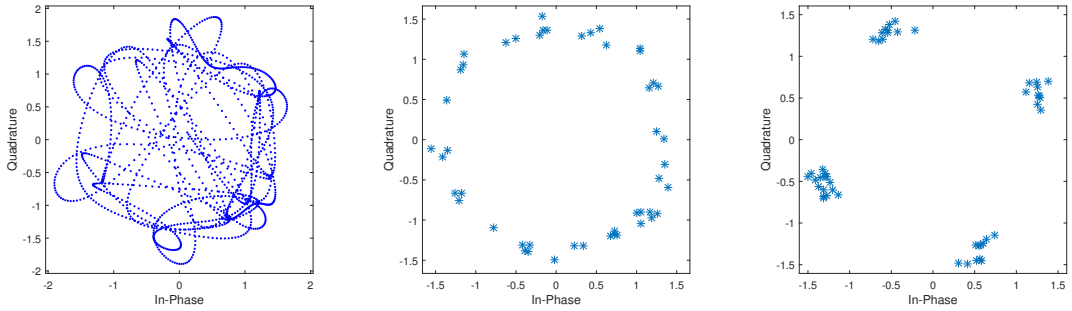


Figure 2.4: Baseband signal, Gardner TED's output and corresponding sampling decisions

The baseband signal before synchronization suffers from undesired phase and amplitude modulation. The purpose of symbol timing is to define the ideal sampling time to minimize the phase and amplitude variation. Moreover, if a carrier frequency offset is introduced, the scatter plot rotates. Fig. 2.5 shows the scatter plots at the different stages of the synchronizer.



(a) Before down-sampling (b) After down-sampling (c) After Carrier recovery

Figure 2.5: Representations of a typical scatter plot at different stages of the synchronizer.

In Fig. 2.5(a), the signal before down-sampling rotates, and is subject to variations in amplitude. Fig. 2.5(b) shows that after timing recovery, only samples at the middle of the symbols left, and they are distributed around a rotating constellation with nearly same amplitudes. The last step, carrier frequency recovery is shown in Fig. 2.5(c). The samples are aggregated near four constellation points. Note that rotation due to frequency offset is removed, and the signal is only subject to a phase offset, and the remaining error can be modelled as additive Gaussian noise. Also, the estimation error of the carrier frequency offset may accumulate after a number of symbols if it is not monitored. Nonetheless, these results can still prove that the proposed implementation can compensate for the symbol period scaling and the carrier frequency shift introduced by the Doppler effect.

2.3 Synchronizer Performance in Realistic Conditions

A set of sea trials were conducted to test underwater communication algorithms in the Summer 2017. The overarching goal is to demonstrate the communication performance of new algorithms in controlled underwater acoustic environmental conditions.

A preliminary trial took place in Northwest Arm near Halifax, while the final set of experiments were run 10 km off the coast of St. Margaret's Bay, NS, Canada in July 2017. The experiment provided an opportunity to evaluate a variety of synchronization algorithms for underwater communications.

A transmission frame was divided into pilot and data blocks. The pilot block was a wideband, linear frequency modulated chirp signal for frame synchronization. The data block was divided into four sections, and each section had a symbol rate of 240, 80, 60, 15 and 5 baud respectively. The data consisted of a pseudo-random noise (PN) sequence that had a length of 512 chips, and that was repeated 16 times. A QPSK modulation scheme was chosen, and the transmitted waveform was pulse shaped using a RRC filter with a rolloff factor 0.5. The centre frequency was 2.048 kHz.

2.3.1 Synchronization in Benign Conditions

During the test in Northwest Arm, the depth is reported to be approximately 10-13 m, and the receiver's deployment location is around 220 m away from the transmitter.

There was mild current observed during the experiments, and the sea state is generally favourable to the test. The acoustic propagation properties can also be observed using the channel impulse response in Fig. 2.6, where the color represents the intensity of the arrival signal at a certain path delay. It can be found that there is a strong and stable direct path of arrival and very few multipath interference.

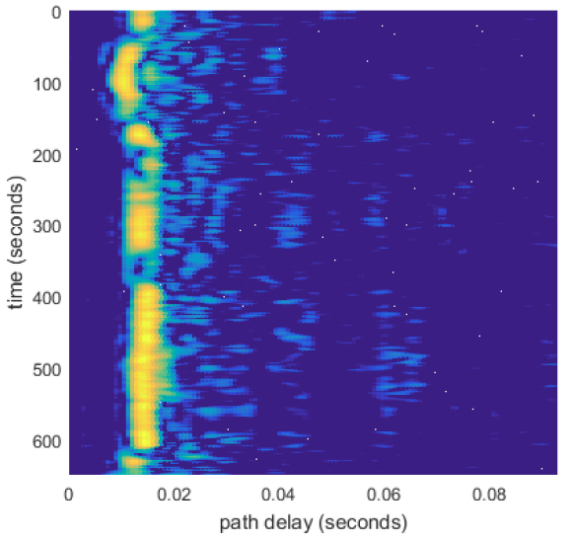
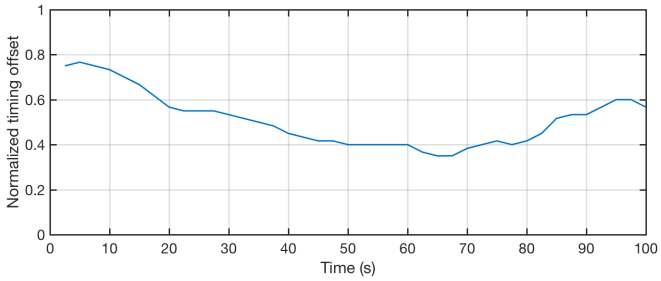


Figure 2.6: Channel impulse response as a function of time.

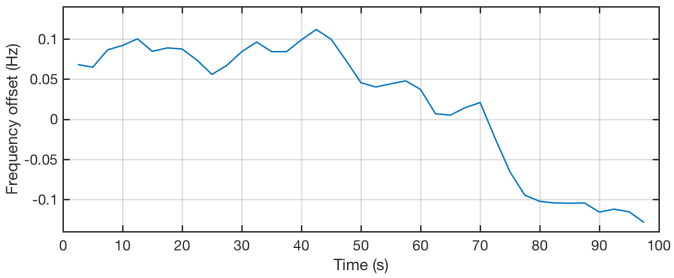
At the receiver, all signals are sampled at 44.1 kHz. The frame is first coarse synchronized with the pilot block so that the data block can be extracted. Down conversion and a matched filter are applied to the data block to generate the baseband signal. If the baseband signal is directly down-sampled and demodulated, the bit error rate (BER) is 0.49, which means no information is recovered.

To recover the data, the feedforward synchronizer discussed in Section 2.1.2 and 2.1.3 is used before demodulation. For demonstration purposes, results from the 80 baud data are presented here. In each iteration, 200 symbols are fed into the synchronizer to estimate the timing and carrier frequency offsets. The synchronizer outputs are plotted in Fig. 2.7.

Several observations can be made from Fig. 2.7. There are 8192 QPSK symbols for a total duration of 102.4 s. Both symbol timing and carrier frequency offset are varying with time. One possible cause is the Doppler effect due to the relative motion between the transmitter and the receiver. After demodulation, the BER is 4.1×10^{-3}.



(a) Symbol timing offset as a function of time



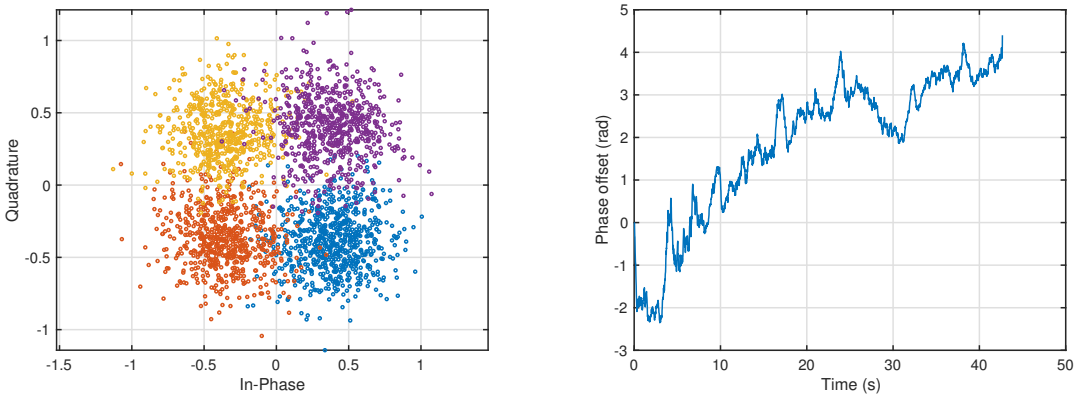
(b) Carrier frequency offset as a function of time

Figure 2.7: Symbol timing and frequency offset estimation results.

2.3.2 Synchronization in Highly Dispersive Conditions

For the trial in St. Margaret's Bay, the channel conditions are much more dispersive than that in Northwest Arm. The transmission distance tested in the trial is between 1 km and 10 km, and moderate current introduced drift is experienced. Compared to the channel in Northwest Arm, this is a typical underwater acoustic propagation channel with strong time variation, Doppler and multipath conditions.

To mitigate the channel conditions, a case of a slow symbol rate at 60 baud with 1 km transmission distance is studied. The transmitted signal is a QPSK modulated PN sequence with length of a 512 symbols and repeated 5 times. A coarse rescaling is applied using pilot blocks before any synchronization process [27]. To be specific, the duration between two consecutive chirp signals is known and can be restored by resampling. As such, the Doppler introduced linear, time domain scaling is compensated coarsely. The received signal is down-converted to baseband and the oversampling rate is 10. The symbol timing recovery is applied first using the O&M algorithm. For demonstration purpose, a DA PLL is used for carrier frequency recovery. The resulting constellation diagram and the phase offset are shown in Fig. 2.8.



(a) Constellation diagram after synchronization, symbols are colored according to their true value

(b) Time varying phase offset

Figure 2.8: Synchronization using a data-aided PLL.

It can be observed that in Fig. 2.8(b), the phase varies significantly with both a low frequency trend and high frequency random components. It illustrates that an effective frequency recovery is essential to the underwater acoustic communications. Note that the phase offset is fully recovered using a DA algorithm, which is not practical. Also, even with good synchronization, the overall symbol error rate (SER) is 7.3×10^{-2}. The resulting constellation for these conditions can be observed in Fig. 2.8(a), where the color coding identifies the expected quadrant for each symbol.

The practical receiver algorithm is developed based on a standard coherent underwater acoustic receiver proposed by M. Stojanovic [50]. A fractionally spaced decision feedback equalizer (FS-DFE) is used to mitigate the ISI introduced by the multipath propagation. The FS-DFE uses sampling interval smaller than the symbol period (half symbol period in this case) and is insensitive to the timing phase of the incoming signal. Channel tracking is accomplished through the use of an adaptive algorithm which is a combination of recursive least squares (RLS) and a second-order digital PLL. The RLS algorithm is known for its fast equalizer convergence speed. The PLL is not commonly found in an equalizer, since the equalizer itself can track and compensate for small carrier frequency offset. However, the frequency offset in the measured conditions is too strong to be compensated by the equalizer alone.

The receiver structure is shown in Fig. 2.9. Note that the frequency compensation

in the original structure is relocated to the front of the feedforward filter to achieve a faster convergence. The equalizer's output is given by

$$\hat{\boldsymbol{d}} = \boldsymbol{a}\boldsymbol{x}e^{-j\hat{\theta}} + \boldsymbol{b}\tilde{\boldsymbol{d}} \tag{2.26}$$

Note that in (2.26), the input $\boldsymbol{x}$, decision $\tilde{\boldsymbol{d}}$, equalizer filter weights $\boldsymbol{a}$ and $\boldsymbol{b}$ are vectors. In this case, the number of taps for the feedforward and feedback filters are 8 and 3 respectively. Detailed Matlab code can be found in Appendix A. At the beginning, the equalizer works in training mode, and the number of training symbols is 1000. After training, the equalizer switches to the decision directed mode. The final SER is 9×10^{-3}, and this result is not sensitive to the symbol timing.

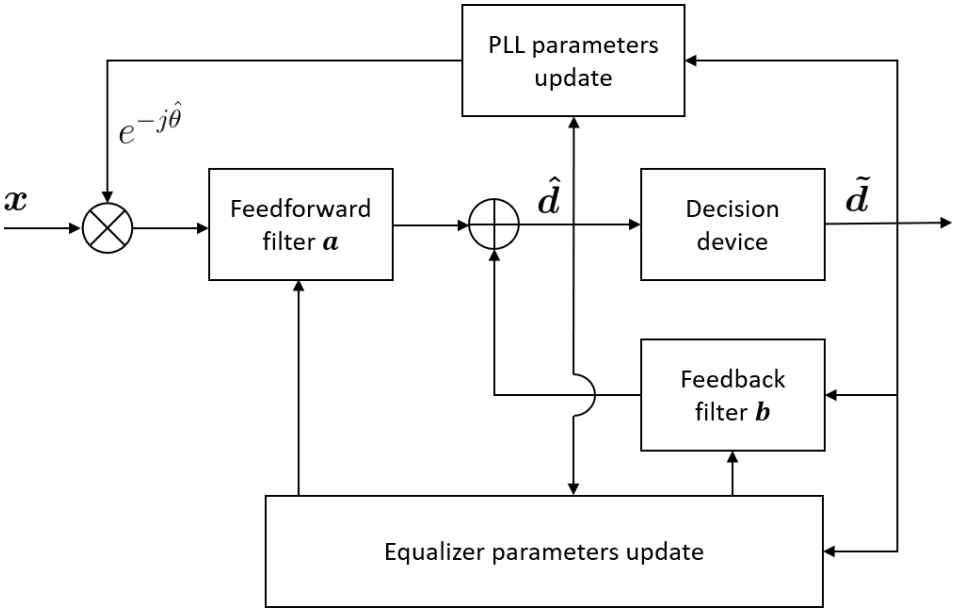


Figure 2.9: The structure of the FS-DFE with PLL

As a comparison to this receiver structure, three other designs are also tested.

1. The symbol timing is recovered first and a *symbol spaced* DFE is applied *without a PLL*. The resulting SER is 4.29×10^{-2}.

2. The symbol timing is recovered first and a *symbol spaced* DFE is applied *with a PLL* . The resulting SER is 1.09×10^{-2}.

3. Finally, an FS-DFE *without a PLL* is also tested. The resulting SER is 5.83×10^{-2}.

Note that the performance of a symbol spaced DFE is very sensitive to the timing recovery result. An inaccurate symbol timing can easily lead to the equalizer convergence failure. Although the fractionally spaced equalizer reduces the burden of symbol timing recovery, it also increased the complexity of the equalizer. Also, it can be found that a build-in PLL can significantly improve the SER performance of an equalizer.

2.4 Summary

In this chapter, three topics have been discussed. The first is a description of the standard symbol timing and carrier frequency recovery techniques. Both have similar feedforward and feedback configurations and consist of an error detector or estimator. Several popular detector/estimator algorithms are described and compared.

Next, a feedforward configuration with a Gardner detector that can compensate for the Doppler effect introduced symbol timing and carrier frequency offset is detailed. In this design, a symbol error detector is used in a feedforward manner such that it can achieve a faster convergence rate and keep a good tracking ability at the same time.

The last section focuses on the underwater acoustic communication in realistic conditions. The processed data is acquired in a sea trial, where strong Doppler and time varying fading can be observed. For a benign propagation channel, the synchronization can be easily realized with standard techniques, while for complex channel conditions, a more sophisticated receiver structure is required. The classic receiver that consists of a FS-DFE and a built-in PLL is adopted, and shows good performance in such complex channel conditions. The performance of this receiver structure and its variants show that the synchronization is critical for coherent underwater acoustic receivers.

Chapter 3

Synchronization Based on Entropy Minimization

3.1 Introduction

In coherent wireless communications systems, synchronization is a key operation at the receiver; it is usually realized between the matched filter and the equalizer. The two main functions of the synchronizer are symbol timing and carrier recovery. The purpose of symbol timing recovery is to recover the symbol clock from the modulated waveform, so that it can down-sample the waveform with the correct symbol timing offset (STO). Hence, at the output of a matched filter, a signal that is sampled at the ideal instant can have maximum signal to noise ratio (SNR) and no intersymbol interference (ISI) [33]. Also, to recover the information embedded in the phase modulation in coherent communication systems, the down converter must have exactly the same frequency as the carrier of the signal. However, in practice, the local oscillator frequency deviates from the input signal's carrier frequency. As such, the carrier frequency offset (CFO) has to be compensated. Moreover, in some applications, such as underwater acoustic communications, the channel may constantly change due to the time-variant environment or Doppler effect. Therefore a continuous and fast estimation and compensation of the STO and CFO is essential to maintain the link reliability.

Various synchronization algorithms have been described in the literature. While data-aided (DA) synchronization offers a superior performance, in this work, non-data-aided (NDA) schemes are approached to maintain a high spectral efficiency. A feedforward timing correction architecture is described for fast convergence rate.

Most synchronization algorithms follow the maximum likelihood (ML) criterion or its approximation. For example, the Oerder and Meyr (O&M) algorithm [38] is a square-law nonlinearity (SLN) estimator exploiting the cyclostationary properties of the modulated signal, and is one of the most commonly used NDA feedforward

STO estimator. It has been proven that this algorithm and its variants [25] can be asymptotically interpreted as an ML estimator [32, 57]. For feedback configurations, the NDA Gardner timing error detector and its DA counterpart, the zero-crossing detector can also be derived from the ML criterion [33]. The NDA feedforward CFO estimator proposed in [56] employs the fourth-order cyclostationary property, and still follows the ML criterion. Also, various DA feedforward CFO estimators attempt to maximize the inner product between the training sequence and the data samples. All these CFO estimators can be treated as generalized ML estimators [33].

The primary contribution of this work consists in the definition of a unified synchronization criterion, relying on entropy minimization (EM) as an alternative to the ML criterion. Specifically, the entropy of the eye diagram is evaluated for symbol timing recovery, and the entropy of the constellation diagram is measured for carrier frequency recovery. For both applications, the synchronization parameter that leads to a minimum entropy value is considered to be optimum. A similar concept, the minimum error entropy which is an important and highly effective optimization criterion in information theoretic learning can be found in [5]. Note that it has been shown in [41] that the CFO of phase-shift keying (PSK) and quadrature amplitude modulation (QAM) modulated signals can be recovered by minimizing the entropy of the instantaneous phase probability density function (PDF), and the phase entropy is obtained by linear search with high computational complexity. In comparison, in the proposed work, a more generic entropy criterion is employed, where both the phase and amplitude components are considered. This concept has been briefly introduced in our previous work [30]. However, to the best of the authors' knowledge, this is the first work that provides a non-ML, unified criterion for both symbol timing and carrier recovery and that also includes an extensive conceptual analysis.

For demonstration purposes, a custom estimation algorithm is provided to evaluate the entropy of the signal eye diagram and constellation. Many practical issues can be encountered when implementing EM based synchronization algorithms. These issues include undesired local minima, insufficient oversampling rate and the vanishing of the gradient. The next contribution of this work is a practical implementation that addresses these issues.

The performance of the proposed symbol timing and carrier frequency recovery

criterion is also evaluated in controlled conditions. The effects of various channel impairments, including noise and multipath, are analyzed on the system performance. The EM based algorithms are found to have improved performance when the pulse shaping filter has a small excess bandwidth, so that it allows better spectral efficiency. It is also demonstrated that the EM criterion provides a higher SNR than the ML criterion for timing recovery in multipath channels.

This chapter is organized as follows. In Section 3.2, an entropy minimization criterion for synchronization is introduced. A customized entropy estimation algorithm, and its implementation issues are discussed in Section 3.3. In Section 3.4, the performance of the algorithm is evaluated in controlled conditions. Finally, conclusions are drawn in Section 3.5.

3.2 Entropy Minimization based Synchronization

In this section the entropy minimization criterion is presented. Specifically, the signal model adopted in this paper is presented in Section 3.2.1. The standard ML criterion is briefly introduced in comparison to the EM criterion in Section 3.2.2. Then, the application of the EM criterion to symbol timing using the eye diagram entropy is explained in Section 3.2.3, while the constellation diagram entropy used for carrier frequency recovery is explained in Section 3.2.4.

3.2.1 Signal Model

In this work, it is assumed that the signal is transmitted using coherent modulation schemes with an alphabet size of M, where M is a power of two. In the following discussion, quadrature phase shift keying (QPSK) modulation is used to validate the performance, but the application is not limited to low order modulation schemes. In fact, the discussion can be easily extended to other PSK or QAM modulations. The received binary information follows an independent identical distribution (i.i.d.). The modulated data is pulse shaped to limit the bandwidth occupancy. A standard pulse shaping filter is used at the transmitter, which is designed under the Nyquist criterion, such that there is no ISI at the ideal sampling instants.

At the receiver, the over-sampled passband signal goes through a down converter

and a matched filter first. Then, a timing recovery is applied, and it is followed by a carrier frequency recovery. Such a design is based on the idea that the decimated samples are sufficient for the CFO estimation, and the estimation has less computational burden because it operates at a reduced rate.

At baseband, the i-th data sample x_i after timing and carrier frequency recovery can be expressed as [60]

$$x_i[\tau, f_\Delta] = y(iT + \tau, f_\Delta)e^{-j2\pi f_\Delta iT}, \tag{3.1}$$

where y is the output of the matched filter, τ and f_Δ are the STO and the CFO respectively, and T is the symbol period. To estimate the STO and CFO, the EM criterion will be presented in Section 3.2.3 and Section 3.2.4 respectively.

3.2.2 Maximum Likelihood Versus Entropy Minimization

The estimation of the STO and CFO can be considered to be an optimization problem. Except for a few heuristic methods, most algorithms are based on maximizing a likelihood function. In [33], for the NDA symbol timing recovery, this criterion yields the objective function $\Lambda(\tau)$, which is equal to

$$\Lambda(\tau) = \sum_{i=1}^{N} |x_i(\tau)|^2, \tag{3.2}$$

while the DA carrier frequency recovery often uses the objective function $\Lambda(\tau)$, defined as

$$\Lambda(f_\Delta) = \left| \sum_{i=1}^{N} c_i^* x_i(\tau, f_\Delta) \right|, \tag{3.3}$$

where c_i^* is the complex conjugate of the i-th training symbol. These two objective functions are actually aimed at maximizing the energy of the data sample set. This ML based estimation method uses second-order statistics of the samples and is suitable for linear channels with additive white Gaussian noise (AWGN). However, for channels that are dominated by ISI, and cannot be modelled with second-order statistics appropriately [16], a criterion that considers higher order statistics is a more reasonable approximation.

The entropy is a measure of randomness or uncertainty of a signal, and it is a function of the signal PDF. As explained in [45], the higher order statistics are

taken into consideration by measuring entropy. According to information theory, the minimum entropy of any type of communication signal is equal to the entropy of the transmitted information. It is understood that the purpose of synchronization is to remove the interference due to the STO and CFO. This specific interference introduces extra entropy to the received signal and, as such, the entropy can be used as a cost function towards synchronization. Unlike the ML criterion, the EM criterion does not model the interference using a statistical model, making it difficult to prove the EM criterion mathematically. However, it will be demonstrated through rigorous numerical simulations in the rest of the paper.

The Shannon entropy is the most commonly used measure of the quantity of information embedded in a signal. Assuming M possible observations of a one-dimensional discrete signal, with p_k representing the probability of the k-th occurrence, the Shannon information entropy H_S is expressed as [46]

$$H_S = -\sum_{k=1}^{M} p_k \log p_k, \tag{3.4}$$

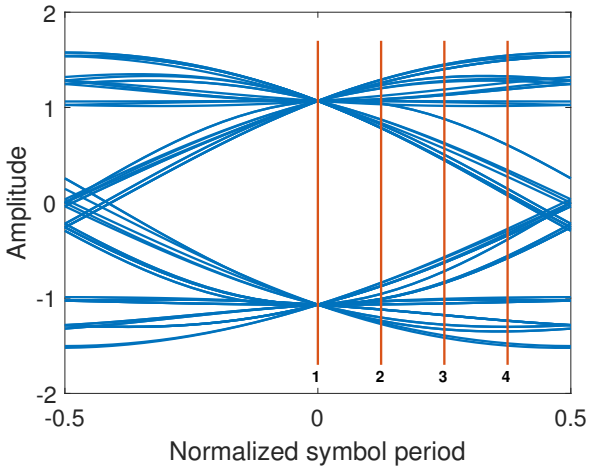
where the logarithmic function usually uses base two, and the corresponding entropy unit is expressed in *bits*. In the next two sections, the Shannon entropy is used as a metric to evaluate the eye diagram and constellation diagram.

3.2.3 Eye Diagram Entropy and Symbol Timing Recovery

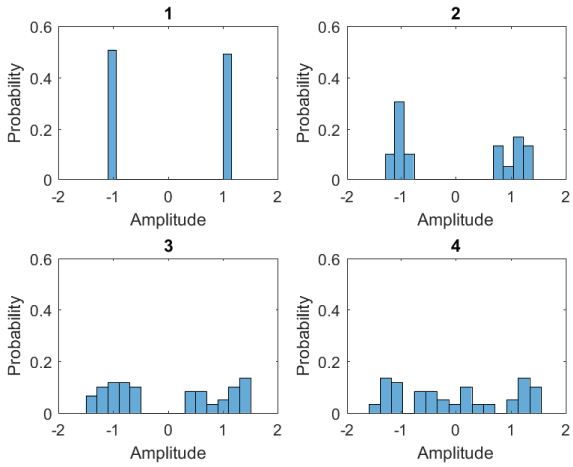
To recover the symbol timing, in this work the entropy of the eye diagram is utilized. Ideally, the down-sampling instant should be located in the middle of the eye diagram where the eye opening reaches its maximum. The symbol timing recovery can be then interpreted as an algorithm to adjust the timing instant with a proper STO on the eye diagram.

The eye diagram is composed of time domain signal traces that are periodically overlaid in a window with a length of one or two symbol periods. The *eye diagram entropy* is defined as the entropy of the signal that is distributed at a certain timing instant on the eye diagram. This can be further explained using Fig. 3.1. The eye diagram in Fig. 3.1(a) is sliced vertically at four timing instants. The signal probability distributions at each timing instant are estimated using the histogram in

Fig. 3.1(b). The histogram is a simple visualization of the data distribution where bins are defined, and the number of data samples within each bin is tallied. Here the number of samples is normalized and presented as probabilities. Then, the eye diagram entropy can be found using (3.4).



(a) Timing instant on a typical eye diagram



(b) Histograms of signal probability distribution

Figure 3.1: Signal probability distribution on four timing instants on a typical eye diagram.

For example, a given modulation scheme with an alphabet size of M is implemented to transmit random data in an ideal channel. At the receiver, with perfect timing recovery, the signal samples can only be distributed equally within M possible symbols. The probability that the samples belong to the k-th symbol is $p_k = 1/M$.

Substituting p_k into (3.4), the minimum eye diagram entropy is defined as

$$\min H_{eye} = -\sum_{k=1}^{M} \frac{1}{M} \log_2 \frac{1}{M} = \log_2 M, \tag{3.5}$$

which is exactly the same as the amount of information carried by each symbol.

The minimum eye diagram entropy only exists when the perfect symbol timing is achieved, because according to the Nyquist criterion, there is zero ISI at these timing instants. For a timing instant that deviates from the middle of the eye, the Nyquist criterion is violated and the interference from adjacent symbols increases the randomness. An analytical relationship between the eye diagram entropy and the timing instant is difficult to demonstrate; however, it can be observed that as the timing instant shifts away from the centre, the signal energy from current symbol decays, and the interference energy grows, leading to increased randomness. Therefore, the entropy, being a measure of randomness, will also increase. Since each interference pulse from adjacent symbols carries the same amount of information, if the span of the pulse shaping filter is N_{span}, the maximum eye diagram entropy will be N_{span} times the amount of information carried by each symbol, which can be presented as

$$\max H_{eye} = N_{span} \log_2 M. \tag{3.6}$$

To summarize, the EM based symbol timing recovery seeks the timing instant with minimum eye diagram entropy. The eye diagram entropy can be seen as an indicator of the ISI that is introduced by adjacent symbols. Effectively, the desired timing instant with zero ISI is the instant with minimum eye diagram entropy.

3.2.4 Constellation Diagram Entropy and Carrier Frequency Recovery

The EM criterion can also be applied to recover the carrier frequency. When the passband signal is down converted to baseband, the complex data can be visualized as a constellation diagram. However, if the frequency of the local oscillator is different (even by a very small margin) from the carrier of the signal, the resulting constellation diagram rotates and cannot be demodulated reliably. The estimation of a CFO that is much smaller than the symbol rate is discussed in this section, since for a large CFO, a preceding coarse carrier recovery is usually required.

The randomness of the signal distribution on the constellation diagram can be quantitatively measured by the *constellation diagram entropy*. Similar to the eye diagram entropy, the histogram can be used for probability estimation. However, a 2D histogram is needed to present both in-phase and quadrature components (or phase and amplitude components when considering the polar coordinate system) on the constellation diagram. Examples of noise free QPSK constellation diagrams with zero, mild and strong CFO with their corresponding histograms are shown in Fig. 3.2. The constellation diagram entropy can be estimated with these 2D histograms. According to (3.4), the highest probability peaks in the histogram indicate the lowest entropy (as observed in Fig. 3.2(a)). The probability that a sample occupies a given bin can be roughly considered to be equal in all bins. Therefore, it can be approximated by $p_{const} \approx 1/n_{bin}$, where n_{bin} is the number of histogram bins loaded with signal samples. Thus, the constellation entropy is given by

$$H_{const} \approx -n_{bin} p_{const} \log_2 p_{const} \approx -\log_2 \frac{1}{n_{bin}}. \tag{3.7}$$

For example, when the CFO is zero, $\min n_{bin} = M$, and the minimum constellation entropy is

$$\min H_{const} = \log_2 M, \tag{3.8}$$

which is the same as (3.5). There is also an upper limit on the constellation entropy when the rotation of the constellation results in samples that are uniformly distributed along a circle such that separate clusters can no longer be distinguished. This phenomenon will be further discussed in Section 3.3.3. Note that an analytical discussion is provided in [41] and demonstrates that, for PSK modulation, the entropy has a global minimum and corresponds to a CFO equal to zero.

3.3 Implementation Entropy based of Symbol Timing and Carrier Frequency Estimation

In this section, a customized entropy estimation algorithm is provided. Then the EM criterion will be implemented for symbol timing and carrier frequency estimation. Practical issues are addressed, and the estimation algorithms are discussed in detail.

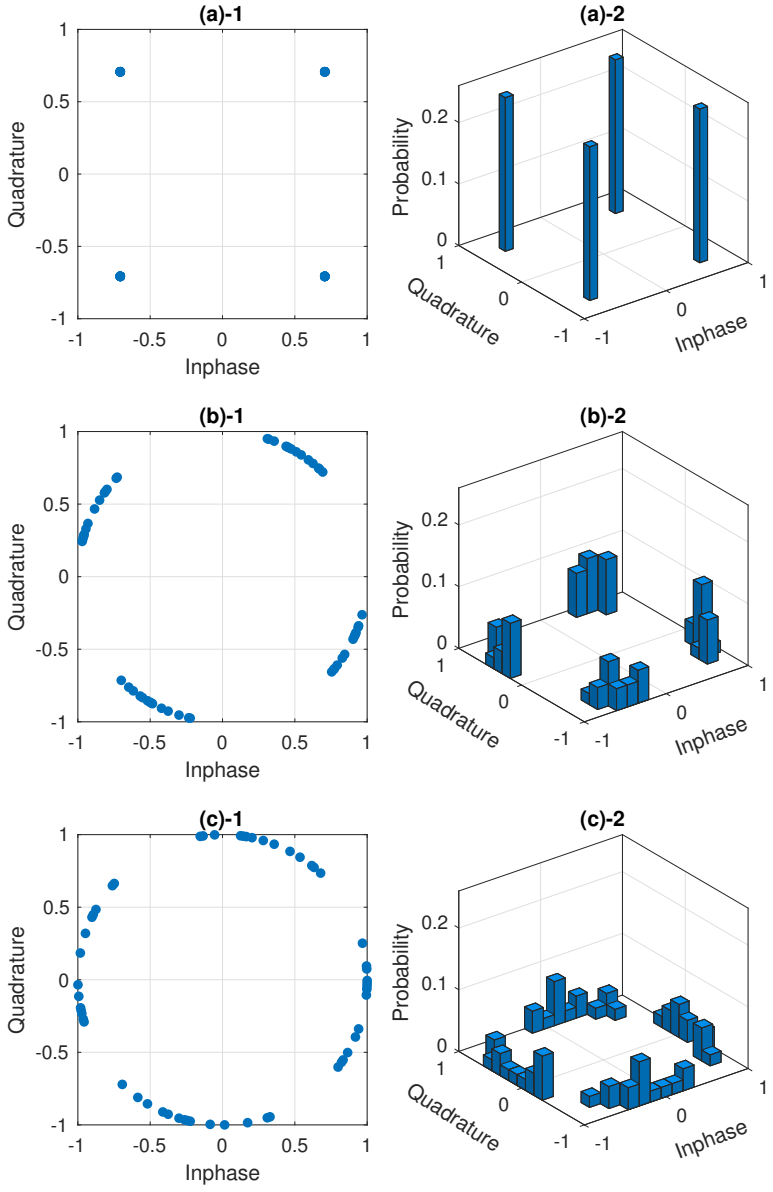


Figure 3.2: Constellation diagrams with different CFOs and the corresponding histograms of signal probability distribution.

3.3.1 Customized Entropy Estimation Algorithm

According to the Shannon entropy defined by (3.4), the entropy is a function of the PDF of the observations. A common technique is to use the histogram for probability estimation as shown in Fig. 3.1 and Fig. 3.2. In this section, the Rényi entropy is utilized as an alternative customized entropy estimation algorithm for the purpose of conceptual verification.

The Rényi entropy is a generalization of the Shannon entropy. It is defined as [44]

$$H_R = \frac{1}{1-\beta} \log \left(\sum_{k=1}^{M} p_k^{\beta} \right), \tag{3.9}$$

where β is the order of the Rényi entropy. As explained in [3], when the order $\beta \rightarrow 1$, the Rényi entropy tends to be equal to the Shannon entropy. Following [45] the quadratic Rényi entropy ($\beta = 2$) is chosen in this work such that a further simplification can be made. Using $\beta = 2$ in (3.9), the quadratic Rényi entropy is given by

$$H_{R2} = -\log \left(\sum_{k=1}^{M} p_k^2 \right). \tag{3.10}$$

Note that in (3.10), the logarithm function is external to the sum of the quadratic probabilities. Because the logarithm function is monotonic, minimizing (3.10) is equivalent to maximizing its internal portion. Since the search for a minimum entropy relies on a relative value of H_{R2}, the logarithm function can be dropped out without affecting the estimation result. This can be attractive when implemented on elementary processors that cannot process advanced math functions. Recall that for the Shannon entropy given in (3.4), the sum of $p_k \log p_k$ is simplified to the sum of p_k^2 here.

As suggested in [42], the kernel density estimation (KDE) is used to evaluate the argument of the logarithm function in (3.10). As such, for a set of N samples, $i = 1, \ldots, N$, the sample probability at an observation ξ can be estimated by

$$p(\xi) = \frac{1}{N} \sum_{i=1}^{N} K_r \left(\xi - x_i \right), \tag{3.11}$$

where $K_r(\cdot)$ is a kernel function with a positive parameter r. Then, by substituting

(3.11) into (3.10) and after some simplification, we have

$$H_{R2} = -\log\left(\frac{1}{N^2}\sum_{i=1}^{N}\sum_{j=1}^{N} K_r\left(x_i - x_j\right)\right), \tag{3.12}$$

where p_k^2 in (3.10) is directly estimated by the kernel function. The simplest kernel function, the top-hat kernel is given by

$$K_r(x) = \begin{cases} 1, & |x| \leq r \\ 0, & \text{otherwise.} \end{cases} \tag{3.13}$$

where the threshold r is used to determine the quantization level in which samples are grouped for entropy estimation. The choice of r will be detailed later in this section. Using (3.12) and (3.13), the entropy can be estimated by measuring the distances between samples instead of using histograms.

From the discussion above, the customized entropy estimation algorithm is summarized using the following steps:

1. For a given set of observations with N samples, calculate all the distances d_{ij} between each sample pair x_i and x_j, where $1 \leq i < N$ and $i < j \leq N$. Then, the distance d_{ij} is given by

$$d_{ij} = \left\|x_i - x_j\right\|, \tag{3.14}$$

   where $\|\cdot\|$ represents the Euclidean norm.

2. Define a separation threshold r and count the number of d_{ij} that satisfy $d_{ij} > r$, and denote the number of separated sample pairs as H_{sp}.

3. After normalization, express the modified Rényi entropy (MRE) as

$$H_{MRE} = \frac{H_{sp}}{N(N-1)/2}. \tag{3.15}$$

The resulting H_{MRE} is the modified version of the quadratic Rényi entropy in which the logarithm function is dropped. Because of the normalization, the value of H_{MRE} is limited between 0 to 1 and is unitless. Intuitively, H_{MRE} is a measure of the amount of sample dispersion, since it counts the number of sample pairs with distances greater than the threshold r.

The choice of the threshold r (often referred as bandwidth in KDE [1]) exhibits a strong influence on the results. It can be derived arithmetically from sophisticated algorithms to achieve an optimal probability estimate [1]. Note that the absolute entropy value is not important, and the threshold r is empirically set to be equal to the root-mean-square value of the noise.

Another design parameter for MRE estimation is the number of samples, which is equal to a few hundred in our application. Entropy estimation is more accurate with a growing number of samples, but the number of distance calculations grows quadratically. Note that the computational complexity can be reduced by using other distance metrics presented in [4]. However, this will cause a reduced tracking ability, which means that the algorithm cannot adapt to the time varying channel effectively. As such, the choice for the number of samples depends both on the dynamic channel conditions and the hardware capabilities.

3.3.2 Symbol Timing Offset Estimation

STO estimation can be implemented by searching for the instant with minimum entropy in the eye diagram. In this section, the following practical issues are addressed: 1) resolving local minima in the entropy curve, 2) examining the timing recovery in presence of CFO, and 3) providing accurate STO estimation at low oversampling rate.

The entropy reaches a global minimum in the centre of the eye diagram, but in practice, when the timing instant is close to the symbol transition area, the entropy may decrease and create local minima. Local minima could result in false STO estimation especially at low SNR conditions, and particularly if gradient based search algorithms are used. Since the entropy local minima occur when the sample magnitude is small, an additional threshold can be introduced to eliminate these data samples in the entropy estimation.

The following steps summarize the MRE estimation algorithm with the additional threshold:

1. Define a threshold r_{mg} and build a new sample set where all samples with a magnitude greater than r_{mg} are included. The number of samples in the new set is denoted as N_{mg};

2. Find all the Euclidean distances d_{ij} between each sample pair x_i and x_j in the new sample set, where $1 \leq i < N_{mg}$ and $i < j \leq N_{mg}$;

3. Define an aggregation threshold r. Count the number of d_{ij} for which $d_{ij} < r$ (note this inequality is different from its counterpart in Section 3.3.1), and denote the number of aggregated sample pairs as H_{ag};

4. Express the bounded modified Rényi entropy (BMRE) as

$$H_{BMRE} = 1 - \frac{H_{ag}}{N(N-1)/2}. \tag{3.16}$$

A typical case is given here to show how the EM based STO estimation works. The received signal consists of a frame of QPSK modulated random symbols, which are pulse shaped to generate a baseband complex envelope. The oversampling rate is 40 to provide a better eye diagram resolution. An AWGN channel is assumed with $E_s/N_0 = 18$ dB. After the matched filter, the eye diagram of the real component of the signal is shown in the upper part of Fig. 3.3, and both the MRE and BMRE estimation results are shown in the lower part. The thresholds are $r = 0.25$ and $r_{mg} = 0.3$.

In Fig. 3.3, the entropy reaches a global minimum in the centre of the eye diagram, and its value is close to 0.75. This minimum entropy value can be derived as follows. Since the QPSK symbols are i.i.d, there are four clusters distributed on the constellation diagram, and each cluster consists of $N/4$ samples. The threshold r is designed such that the condition of $d_{ij} < r$ can only be satisfied between samples located in the same cluster. For 4 clusters, $H_{ag} \approx 4\left(N/4\right)^2/2$ (assuming $N \gg 1$). Thus, the minimum value of the BMRE is approximated by

$$\min H_{BMRE} \approx 1 - \frac{4\left(N/4\right)^2/2}{N^2/2} = 0.75. \tag{3.17}$$

The minimum value of MRE in the eye diagram centre can also be derived in a similar way to the BMRE. The entropy increases with the absolute value of STO, indicating more randomness introduced by ISI. When the STO is beyond ± 0.35, the MRE estimation decreases and creates a local minimum in the symbol transition. This phenomenon is also illustrated in the eye diagram, where the samples aggregate into three visible groups (with amplitudes of ± 1 and 0).

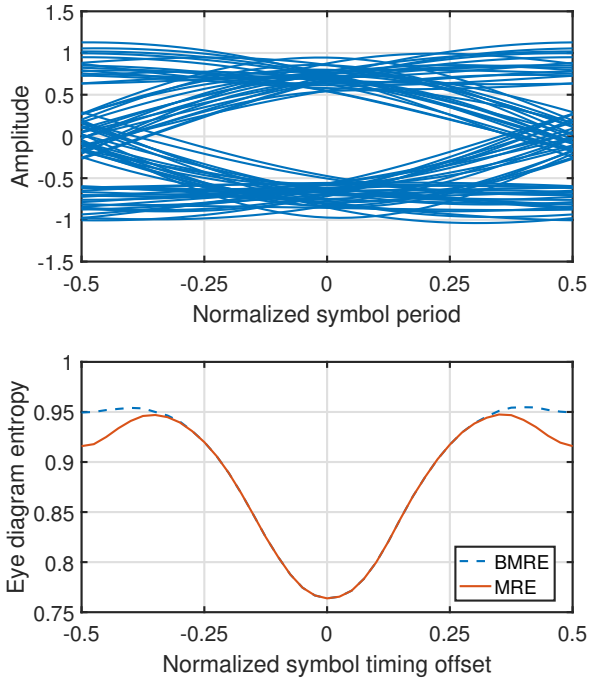


Figure 3.3: An typical eye diagram (upper) and the corresponding eye diagram entropy (lower).

The result of the BMRE algorithm coincides with that of the original MRE algorithm in most of the timing instants, but the local minima near ± 0.5 become flat. As such, with the BMRE estimation algorithm, the local entropy minima due to the symbol transitions are removed, and the STO can be estimated with higher accuracy.

Next, the impact of timing recovery in presence of uncompensated CFO will be evaluated. Theoretically, the previous analysis of eye diagram entropy still holds, but the CFO does introduce extra entropy in the estimation. To understand how the CFO affects the eye diagram entropy, another simulation that is similar to what was demonstrated in the early part of this section is conducted with an extra CFO at 1% of the symbol rate introduced. The eye diagram and the corresponding entropy are depicted in Fig. 3.4.

The eye diagram shows an eye that is completely closed: the centre of the eye or optimum timing instant cannot be identified by only observing the eye diagram. However, the optimum timing instant can be clearly identified with the entropy curve. Both the MRE and the BMRE algorithms have the same global minimum at zero

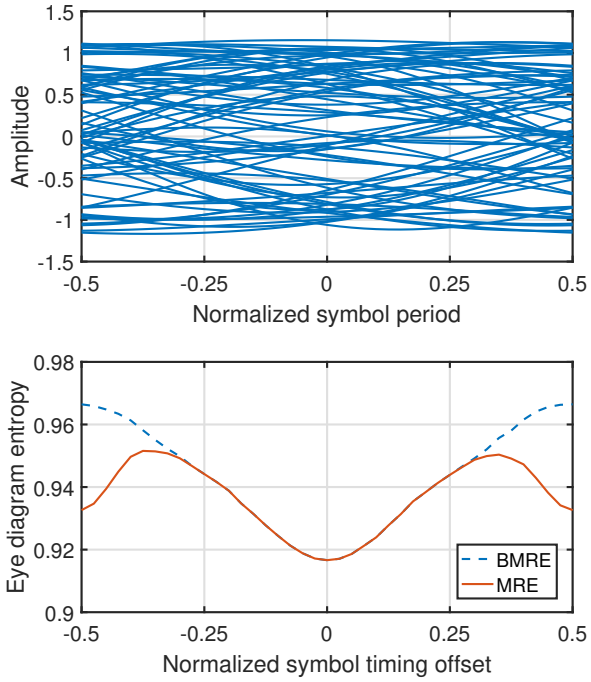


Figure 3.4: An eye diagram with carrier frequency offset (upper) and the corresponding eye diagram entropy (lower).

STO.

In Fig. 3.4, it is also interesting to observe the entropy plot at a symbol transition. With the MRE algorithm, the local minima are more noticeable and may lead to a false STO estimate. However, the BMRE algorithm shows superior performance: the curve is not flat anymore but continues growing with the same gradient as a function of timing offset. This feature shows a good adaptation of the BMRE algorithm and proves that the "symbol timing recovery before carrier recovery" receiver configuration is feasible.

In the previous discussion, a global search for STO with minimum eye diagram entropy requires a high oversampling rate (normally more than 10 samples per symbol). However, this is not always available in practice, especially for high speed communication. Recall that the O&M algorithm uses as low as 4 samples per symbol, and its STO is given by

$$\tau = \frac{T}{2\pi} \arg \left\{ \sum_{i=1}^{N} x_i^2 e^{-j2\pi(i-1)/N_{sps}} \right\}, \tag{3.18}$$

where T is the symbol period and N_{sps} is the oversampling rate. The operation $\arg\{\cdot\}$ returns the phase angles in radians. The guiding principle of the O&M algorithm is to apply a discrete Fourier transform (DFT) to the squared signal x_i^2, and the STO is extracted from the angle of the resulting spectrum line at the symbol rate.

The theory behind (3.18) is that the squared signal x_i^2 is a periodic signal with the same frequency and phase as the pulse shaped symbols due to the cyclostationary property of x_i. Thus, even with a low oversampling rate, the STO can still be estimated by DFT. The eye diagram entropy curve exhibits a similar property as x_i^2 within one period. Therefore, the same approach can be applied to the EM based symbol timing algorithm. To be specific, x_i^2 in (3.18) can be replaced with the eye diagram entropy H_i, such that the STO can be found with

$$\tau = \frac{T}{2\pi} \arg \left\{ \sum_{i=1}^{N_{sps}} H_i e^{-j2\pi(i-1)/N_{sps}} \right\}. \tag{3.19}$$

Note that (3.19) assumes that the entropy curve is symmetric to the centre of the eye diagram, but the symmetry may not be maintained at low SNR or in a multipath channel. In these conditions, the algorithm may result in large estimation variance.

Although the complexity of the STO estimation is reduced by using less samples (similar to the O&M algorithm), the EM based method still requires higher computational load than conventional algorithms, because of the nature of entropy estimation. The MRE and BMRE algorithms can relieve certain computational load by using approximation methods, but the complexity remains higher than the conventional methods. Nonetheless, the entropy analysis provides more insights on the signal eye diagram and helps locate the maximum eye opening, as will be shown in Section 3.4.

The STO estimation discussed in this section assumes that the symbol period is known to the receiver and that there is no time scaling during transmission. Thus, the symbol timing recovery discussed here is equivalent to locating a timing offset on the eye diagram. If this assumption does not hold, a symbol period estimation is required. Such an estimation can be done by searching for the symbol period that can minimize the whole eye diagram entropy.

3.3.3 Carrier Frequency Estimation

In this section, the implementation of the EM based CFO estimation algorithm is detailed. Similarly to the STO estimation discussed above, the proposed algorithm is also NDA. The constellation diagram entropy is measured by defining an adequate range for the trial CFO, and a global search is applied to find the minimum entropy. The characteristics of the entropy curve are shown first and then a method that can increase the global search efficiency is proposed.

In Section 3.2.4, it can be noted that the constellation entropy is almost flat when the CFO is greater than a maximum frequency. This maximum frequency can be considered as the effective search range for EM based CFO estimation. The entropy curve within the frequency limit has a V-shape "trough" (negative peak) and the entropy global minimum is located in the middle of the trough. For a given modulation scheme, the range of the trough is affected by the CFO (f_Δ), the symbol rate $1/T$, and the number of data samples N in the window. For example, the M-PSK modulated signal has a minimum constellation phase difference $2\pi/M$. The accumulated phase shift due to CFO is given by $2\pi f_\Delta NT$. The constellation entropy increases with the CFO until the accumulated phase shift is greater than the minimum constellation phase difference. Therefore, the CFO search range is given by

$$|f_\Delta| < \frac{1}{MNT}, \tag{3.20}$$

and when the CFO is larger than $1/MNT$, the entropy curve becomes flat.

Using (3.20), if N is equal to a few hundred samples, a given CFO that can fall into the entropy search range must be on the order of 0.1% of the symbol rate, which is relatively small compared to the CFO range that needs to be covered. The resulting entropy curve as a function of trial CFO is generally flat with the exception of a sharp trough. A similar result has also been reported in [41]. Consequently, the linear search requires very fine steps to achieve high frequency resolution, and the potential gradient descent algorithm may not converge due to lack of gradient. In other words, an efficient search algorithm cannot be applied.

In order to cover a large estimation range without intensive computation, an algorithm that can expand the width of the trough is required. To increase the trough width, a possible solution is to reduce N. However, the lack of samples will

lead to an inaccurate PDF and entropy estimation. Instead, a block average algorithm inspired by [21] is adopted to smooth the entropy curve. The data samples are equally segmented into L blocks, and the entropy of the i-th block is denoted as $H_i(f_\Delta)$. The block averaged constellation diagram entropy is given by

$$H_{const}(f_\Delta) = \frac{1}{L}\sum_{i=1}^{L} H_i(f_\Delta). \tag{3.21}$$

The CFO is assumed to be constant for the set of data samples. Thus, each block possesses the same entropy curve but with random fluctuation due to the lack of samples for probability estimation. By averaging the entropy curve using small blocks, a wide and smooth entropy trough is achieved. The new trough is L times wider than the original one.

With the same settings as in Section 3.3.2, the numerical simulation of the constellation diagram entropy is compared using 1) 400 samples, 2) 50 samples and 3) block averaged 400 samples with a block size of 50 samples ($L = 8$). The results are plotted as a function of the CFO in Fig. 3.5. Perfect symbol timing is assumed, and the CFO search range is swept within 1% of the symbol rate.

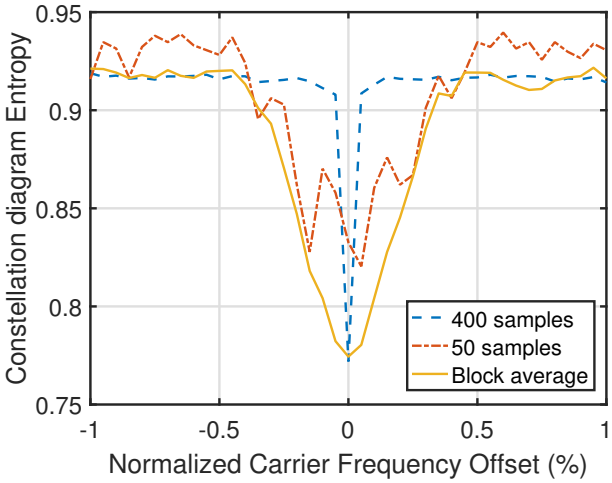


Figure 3.5: Constellation diagram entropy for carrier frequency recovery.

As discussed in Section 3.2.4, the entropy curve should reach the global minimum with the CFO equal to zero. In Fig. 3.5, the entropy curve using 400 samples has only one global minimum when the CFO is equal to zero and no other local minimum.

But its curve has a very narrow trough as predicted. The entropy estimated using 50 samples has a trough boundary at $\pm 0.5\%$ of the symbol rate, which agrees with (3.20). In fact, the trough range is expanded by 8 times, but many fluctuations and local minima appear. In comparison, the entropy curve using the block averaging has the same expanded trough as the entropy estimated using 50 samples, but the local minima are smoothed out, leaving only the global minimum at the zero CFO.

Given the expanded entropy trough, a more efficient two-step linear search, similar to the algorithm described in [41], can be readily applied. First, a coarse search through the frequency range of interest with step size equal to the half trough width can provide an approximate CFO estimate. A second search with a fine frequency step near the coarse estimation result can improve the accuracy of the CFO estimate.

Recall that the computation complexity grows quadratically with the number of samples, so breaking down the sample set into small blocks can significantly reduce the required computation. For example, if the 400 samples are equally divided into 8 blocks, it is easy to find that compared to the algorithm without block averaging, this algorithm requires eight times less number of Euclidean distance calculations (defined by (3.14)). This is a significant reduction in computational complexity.

Compared to the linear search algorithm proposed in [41], the block average method for CFO estimation reduces the computational complexity by calculating less sample pair distances. However, similar to the STO estimation using (3.19), it still more complex than the conventional ML based algorithms because of the estimation of entropy instead of energy. The EM based algorithms are not appropriate for computation sensitive tasks. On the other hand, note that since the computing of the entropy for each trial STO or CFO is independent, it is possible to perform in parallel mode (such as using a multi-core processor or FPGA) to accelerate the estimation.

3.4 Performance Evaluation

In this section, the performance of the symbol timing and carrier frequency estimation algorithms presented in the previous section are assessed in controlled conditions. The estimation error variance in AWGN channel is used as the major figure of merit. Also,

the effect of multipath impairment is analyzed on the system performance.

First, the performance of the symbol timing estimation algorithm (3.19) is examined in presence of AWGN. The O&M algorithm described by (3.18) is used here to represent the ML estimator for comparison.

The performance comparison in presence of AWGN is shown in Fig. 3.6. In this figure, the QPSK modulation is evaluated, but similar results are expected for other PSK or QAM modulation schemes. A pulse shaping filter is used to limit the bandwidth. The rolloff factor used in the evaluation is $\alpha = 0.25$ and 0.05 respectively. The small excess bandwidth conditions are chosen to accommodate a recent bandwidth efficient communication standard [36]. The choice is also motivated by the fact that using small rolloff factor represents the worst case with respect to the timing error variance.

AWGN is introduced at the receiver, such that the symbol energy to noise spectral density ratio (E_s/N_0) ranges from 5 to 40 dB. For each E_s/N_0 setting, the average of 500 Monte Carlo trials are taken. In each trial, a block size of 100 symbols are used to estimate the STO. The proposed algorithm (3.19) is used for the EM based STO estimation. After normalization by the symbol period, the variances of the timing error (also known as the jitter variance in some literature [13]) with respect to the symbol period are represented in Fig. 3.6. Following [33], the modified Cramér-Rao bound (MCRB) is also shown as the theoretical limit.

Several conclusions can be drawn from Fig. 3.6. Generally, the EM based symbol timing algorithm has lower error variance than the O&M algorithm for both rolloff factors and for a large range of E_s/N_0. With the O&M algorithm, larger rolloff factor generates smaller error variance. When the E_s/N_0 is greater than 25 dB, the variance value reaches a lower limit because of its strong self-noise [33]. In contrast, the timing error variance of the EM based algorithm changes less significantly with different rolloff factors. Also, it suffers less from the self-noise.

The O&M algorithm has less error variance when $E_s/N_0 < 9$ dB and $\alpha = 0.25$. This is because the O&M algorithm, as an example ML algorithm, is designed under AWGN assumption. In fact, the EM based algorithm is favoured in small excess bandwidth conditions, because the eye diagram entropy estimation can effectively measure the ISI. However, if a pulse shaping filter with larger excess bandwidth is

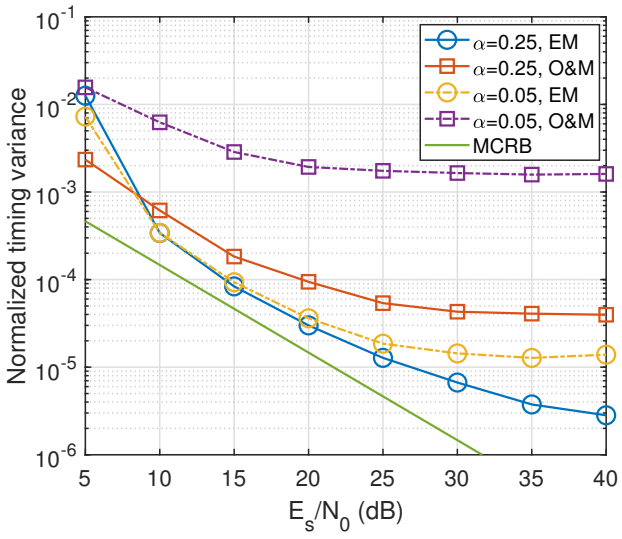


Figure 3.6: Timing error variance of two symbol timing algorithms with two rolloff factors.

used ($\alpha > 0.5$ for example), the O&M algorithm will have a performance very close to the MCRB [33].

As explained in Section 3.3.2, the EM based symbol timing estimation is insensitive to the CFO, but it is interesting to understand how the performance changes in the presence of CFO. The simulation settings are generally the same as for the last one, except that different modulation schemes, BPSK and QPSK, are evaluated, and the rolloff factor is set to 0.25. The introduced CFO is 1% of the symbol rate, and the timing variances are plotted in Fig. 3.7.

For BPSK modulation, the EM based algorithm shows a good performance that is close to the MCRB when the E_s/N_0 is below 30 dB. It has the highest performance improvement compared to the O&M algorithm. However, for QPSK modulation, the performance improvement is marginal. This is because, for the EM based algorithm in (3.19), it is assumed that the eye diagram entropy curve is symmetrical to the centre of the eye diagram, and this only stands with low modulation order and low noise level. Another example of an asymmetrical eye diagram entropy condition will be analyzed in the following simulation.

A key issue that coherent communication systems are facing is the multipath channel impairment. Unpredictable channel impulse responses violate the Nyquist

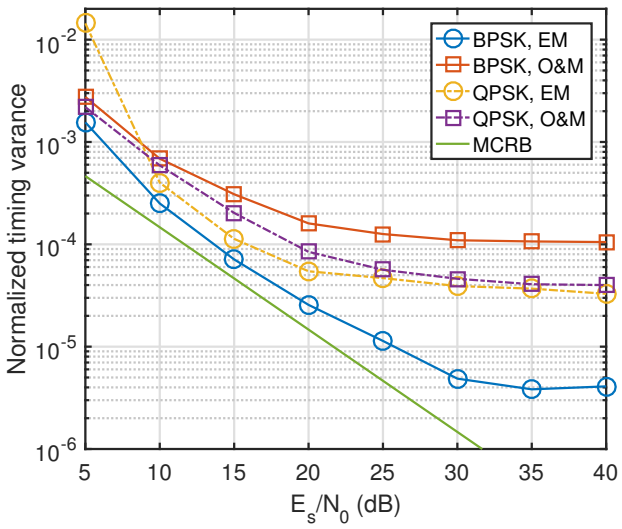


Figure 3.7: Performance of two symbol timing estimation algorithms in the presence of CFO.

ISI criterion, and the communication performance is compromised. The nature of EM based symbol timing estimation is to search for the timing instant with minimum ISI, which makes it more suitable for these conditions than ML based algorithms. To demonstrate this, a set of BPSK modulated symbols with $\alpha = 0.5$ is transmitted. For simplicity, a multipath channel with impulse response $h(t)$ is given by

$$h(t) = \delta(t) + 0.5\delta(t - 1.4T) + 0.2\delta(t - 3.5T). \tag{3.22}$$

At the receiver, the eye diagram and the timing instants estimated by both O&M and EM algorithms are plotted in Fig. 3.8 for E_s/N_0 equal to 15 dB.

The eye diagram in Fig. 3.8 is almost closed and shifted from the centre due to ISI. The bit error rates (BER) of demodulated samples recovered by the two algorithms are compared without equalization. The EM algorithm can find the maximum eye opening and achieves a BER of 1.2%. In contrast, the BER is 5.9% if using the O&M algorithm. As one can observe in Fig. 3.8, the samples recovered by the O&M algorithm have the highest energy output, but with strong ISI.

The results from the channel given by (3.22) is not a special case. To evaluate the timing recovery algorithms in realistic multipath channel conditions, a measured impulse response of an underwater acoustic channel in Fig. 3.9 is considered. The

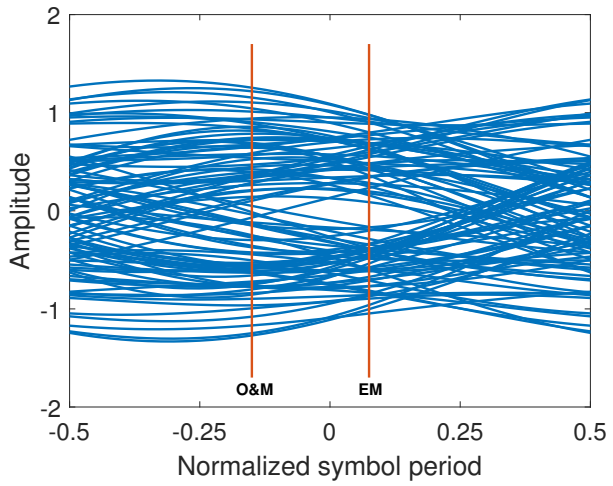


Figure 3.8: An eye diagram after a multipath channel and the corresponding timing estimation results.

multipath channel models utilize the amplitudes and delays of the five greatest impulses from the measured data. Both Rayleigh and Rician fading are tested, where the Rician K-factor is set to 3. The channel model is quasi-static, such that the channel parameters are constant within each trial. The final results are the average of 500 Monte Carlo trials.

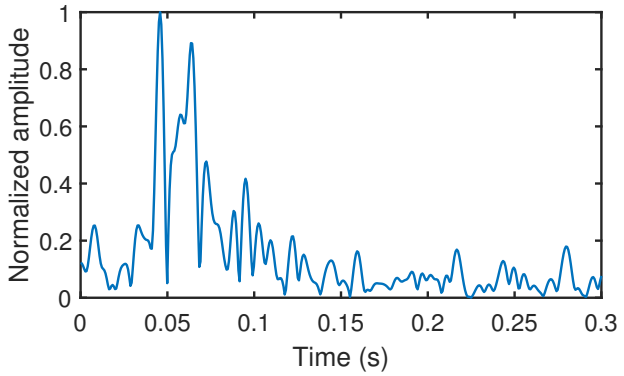


Figure 3.9: Impulse response of a multipath channel.

The transmit signal are QPSK modulated, and 3000 random symbols are sent in each trial with a symbol rate of 240 Bd. At the receiver side, a symbol-spaced decision feedback equalizer is placed after the timing recovery to compensate for the multipath channel. Both the equalizer's feedforward and feedback filters have 6 complex weights. The adaptive algorithm used in the equalizer is a recursive least square (RLS) with

Table 3.1: Symbol error rate (%) in multipath fading channels

Fading	**Rolloff**	**Mid**	**O&M**	**EM**
Rayleigh	0.75	0.36	13.35	1.59
	0.5	1.10	16.02	2.67
	0.25	6.27	16.43	5.85
	0.05	20.38	26.34	14.04
Rician	0.75	0.17	11.92	1.27
	0.5	0.58	11.47	1.64
	0.25	3.64	12.57	3.59
	0.05	11.92	19.45	8.38

a forgetting factor of 0.95. The first 1000 symbols are the training signal for the equalizer, and the symbol error rate (SER) is calculated with the equalizer's output for the rest 2000 symbols.

The simulation results are summarized in Table 3.1. The SER when synchronizing to the middle of the first arrival is also calculated as a reference. Note that if the equalizer converges during the training, the SER of a single trial is negligible. However, if it fails to converge, the SER can be up to around 75% for QPSK modulation. Therefore, the averaged SER listed in the table is a indicator of how often the equalizer fails to converge with the given timing recovered samples.

As can be observed, the SER of the EM algorithm significantly outperforms that of the O&M algorithm for all rolloff factor settings and fading conditions. As such, the equalizer is much more likely to converge using the EM based timing recovery algorithm. It can be observed in Table 3.1 that in small rolloff factor conditions, using the first arrival does not always provide optimum down-sampling positions, since the maximum eye opening is shifted due to ISI as shown in Fig. 3.8. In fact, it is clear that the EM algorithm provides better SER when the rolloff factor is small.

Next, the performance of CFO estimation in presence of AWGN is evaluated. For the carrier frequency recovery test, perfect symbol timing is assumed. The signal is QPSK modulated with a CFO equal to 1% of the symbol rate. Three algorithms are compared for CFO estimation: the EM based algorithm, the open loop and the classic ML algorithm.

The open loop algorithm proposed in [7] estimates the CFO by averaging the differential phase error over the window. For QPSK modulation, the CFO is given

by [33]

$$f_\Delta = \frac{1}{8\pi T} \arg \left\{ \sum_{i=2}^{N} \left(x_i x_{i-1}^* \right)^4 \right\}. \tag{3.23}$$

The classic ML algorithm uses the same global search method as the EM algorithm. The objective function is given by

$$\Lambda(f_\Delta) = \left| \sum_{i=1}^{N} x_i^4 e^{-j8\pi f_\Delta i T} \right|^2. \tag{3.24}$$

In (3.24), the CFO is estimated by searching for the trial CFO that yields the highest energy or alternatively by applying a computationally efficient FFT based implementation [56]. Note that both (3.23) and (3.24) are NDA algorithms and a power of 4 is applied to the signal to remove the modulation. This is not necessary in the EM algorithm. The CFO that is estimated using the three algorithms is normalized by the symbol rate, and for each E_s/N_0 condition, 500 trials are conducted to compute the variance. The results are shown in Fig. 3.10, where the MCRB is also included as a reference.

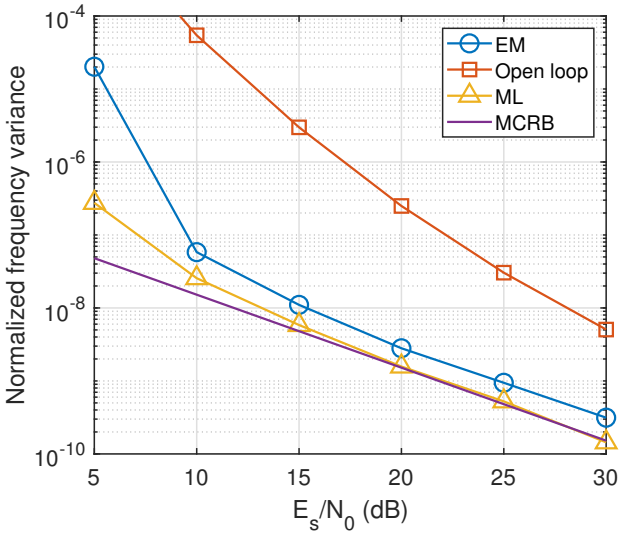


Figure 3.10: Performance of three carrier frequency recovery algorithms.

In Fig. 3.10, the EM algorithm shows a much smaller frequency variance than the open loop algorithm. This demonstrates its robustness for CFO estimation. However, the performance of the classic ML algorithm is mostly the same as the MCRB,

making it slightly better than the EM algorithm. It is not surprising that the classic ML algorithm provides a smaller variance than the EM algorithm in AWGN, since theoretically it is the optimum solution in these conditions.

The CFO estimation performance for both the EM and classic ML algorithms in multipath channels has also been examined. The performance of the two algorithms has no significant difference in terms of SER if the same timing recovery is given. This is because no ISI gain can be provided to the EM based algorithm in contrast to its gain for the STO estimation. The EM algorithm has an estimation variance that is slightly larger than that of the classic ML algorithm, similar to the results observed in AWGN channel. Nonetheless, this demonstrates the usefulness of the proposed estimator as a universal timing recovery algorithm.

3.5 Summary

In this chapter, entropy minimization has been proposed as a synchronization criterion for wireless coherent receivers. It is an alternative to the maximum likelihood criterion, which is the foundation of most standard synchronization algorithms. The symbol timing and carrier frequency offset estimation are implemented by measuring the entropy of the eye diagram and the constellation diagram. The optimum timing delay is found by searching the timing instant with minimum eye diagram entropy, while the carrier frequency offset is estimated by searching through a range of frequencies to minimize the constellation diagram entropy.

Implementation constraints have also been presented. A modified version of the quadratic Rényi entropy and the kernel density estimation method are employed to estimate the probabilities. The proposed method is insensitive to the local minima and the carrier frequency offset. Also, it can extract the timing delay without the need for a high oversampling rate. The carrier frequency recovery algorithm uses block averaging to expand the estimation range without compromising its accuracy.

Although the estimation requires more computational load than the conventional ML based algorithms, it can provide more insights on the signal eye diagram and constellation diagram as well as better estimation performances. The performance of the proposed timing and frequency recovery algorithms is compared with that of

standard approaches by running a set of numerical simulations. It is shown that the entropy minimization has great potential and offers certain advantages for synchronization. Particularly, in multipath fading and small excess bandwidth conditions, the timing recovery using the EM algorithm can significantly improve the equalizer's convergence, and its symbol error rate outperforms that of the ML algorithm, by at least a factor of two.

Chapter 4

Application of Deep Learning Techniques for Wireless Receivers

4.1 Introduction

In the recent years, the signal processing algorithms for coherent underwater acoustic receivers are getting much more sophisticated than before. One reason is that the design and analysis of communication systems typically rely on the development of mathematical models that describe the underlying communication channel, which dictates the relationship between the transmitted and the received signals. Hence, researchers are building more complex underwater propagation models, hoping to obtain a comprehensive understanding of the channel properties [51]. In the same time, data processing algorithms are developed based on these channel models, such that the algorithms can eliminate the interference properly. Consequently, the performance of sophisticated designed algorithms rely on the accuracy of the channel models. As discussed before, the acoustic propagation channel is highly spatially and temporally variable. therefore, it is impossible to exhaust every possibility, but only a few approximate statistical models.

In the field of perception signal processing, researchers are facing similar problems, such as image recognition and classification. The challenge is to design an algorithm that can recognize a certain item. There is an excessive number of parameters that need to be taken into account to build the statistical model. Therefore, the essential features of an object used to be engineered manually for building recognition algorithms. Using this methodology, it is difficult to generalize algorithms to recognize other objects.

In Section 4.2, the state-of-the-art deep learning technique for perception signal processing is introduced. It is the most promising and popular subset of machine learning. Instead of engineering the features, deep learning technique can learn the

features from the dataset automatically. The deep learning models after training have surprisingly outstanding performance and their cognition capabilities have already outperformed human perception in various of areas.

Because of the remarkable performance of deep learning, researchers are applying it to broader areas, such as medical diagnosis, self-driving, language translation and playing games. However, there are very few study in the area of communications. Deep learning applications to wireless communication that can be found in the literature are still in the exploration phase [39, 40, 58].

There are two contributions in this chapter: the first is the implementation of CNN models for both symbol timing and carrier frequency offset estimation; the second contribution is a new CNN based coherent wireless receiver. This model can compensate for small symbol timing and carrier frequency offset. Since this chapter does not provide rigorous analytic demonstration of the performance, the results of this chapter can be used as a proof of concept. Nonetheless, it is demonstrated that deep learning techniques can be applied to wireless communications in various ways and are expected to have great potential.

4.2 Basics of Deep Neural Networks

The artificial neural network (ANN) or multilayer perceptron (MPL) shares many concepts with the modern deep neural networks. This section provides an introduction on the fundamental components of deep neural networks, including the layers, the loss functions and optimizers. Also, the iterative process that trains a model to fit a certain application is covered. Then, the convolutional layer, a type of neural layer that makes the training of a deep learning model possible, is detailed. Finally, a few deep learning frameworks that essentially enable the democratization of deep learning are introduced.

Neural Network Layers

A feedforward neural network consists of multiple layers that are connected in series. Mathematically, the l-th layer describes a mapping function f_l of an input vector $\boldsymbol{r}_{l-1}$

to an output vector $\mathbf{r}_l$ [39]:

$$\mathbf{r}_l = f_l(\mathbf{r}_{l-1}). \tag{4.1}$$

The output vector depends on the input vector $\mathbf{r}_{l-1}$, which is the output from the previous layer f_{l-1}. Moreover, it also depends on a set of layer parameters θ_l. The classic neural networks use *fully-connected* layers which are also known to be *dense* or *linear* layers, , and the output $\mathbf{r}_l$ is related to the input $\mathbf{r}_{l-1}$ by

$$\mathbf{r}_l = f_l(\mathbf{r}_{l-1}) = \mathbf{w}_l\mathbf{r}_{l-1} + \mathbf{b}_l, \tag{4.2}$$

where $\mathbf{w}_l$ is the weight vector and $\mathbf{b}_l$ is the bias vector. With an input layer, an output layer, and two fully-connected layers in between, a classic neural network is shown in Fig. 4.1, where the circles represent the input and output vectors $\mathbf{r}$, and the arrow lines represent the multiplications with weight vectors $\mathbf{w}$. Note that the bias $\mathbf{b}$ are not plotted. Other than the input and output layers, all other layers are invisible to the user, so they are often referred as hidden layers.

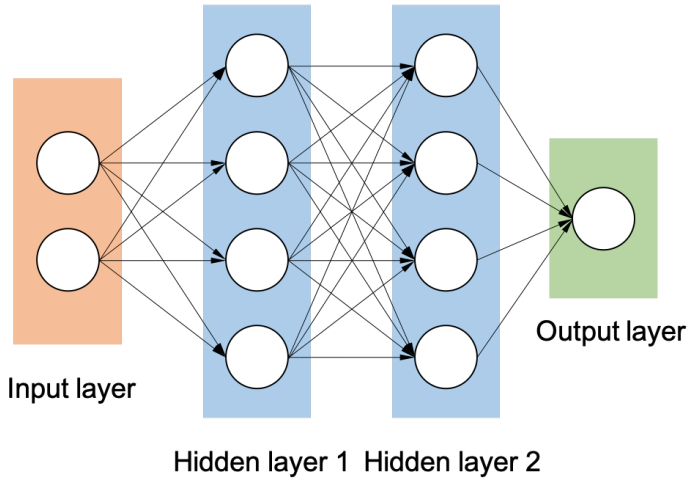


Figure 4.1: A typical artificial neural network.

One reason that an ANN usually has very few hidden layers is that the stack of fully connected layers is simply a linear function. Linear equations are easy to solve mathematically, but they have limited power to learn complicated functional mappings from data. Adding more layers does not improve the capability of the model, but increases the complexity of the network itself. To learn from complicated, high dimensional, non-linear dataset, non-linear layers are required. Such layers are

known as the *activation functions*. To be specific, in a deep neural network, multiple (sometimes tens even hundreds of) fully-connected layers are stacked on top of each other. The activation functions are usually directly connected at the output of a layer, such that the expressive power of a neural network can be improved [17]. With an activation function σ, (4.2) is rewritten as

$$f_l(\mathbf{r}_{l-1}) = \sigma(\mathbf{w}_l \mathbf{r}_{l-1} + \mathbf{b}_l). \tag{4.3}$$

Three popular activation functions are given as:

1. Sigmoid or Logistic:
$$\text{Sigmoid}(x) = \frac{1}{1+e^{-x}} \tag{4.4}$$
2. Tanh (Hyperbolic tangent):
$$\text{Tanh}(x) = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}} \tag{4.5}$$
3. ReLu (Rectified linear units):
$$\text{ReLU}(x) = \max(0, x) \tag{4.6}$$

The corresponding plots are shown in Fig. 4.2.

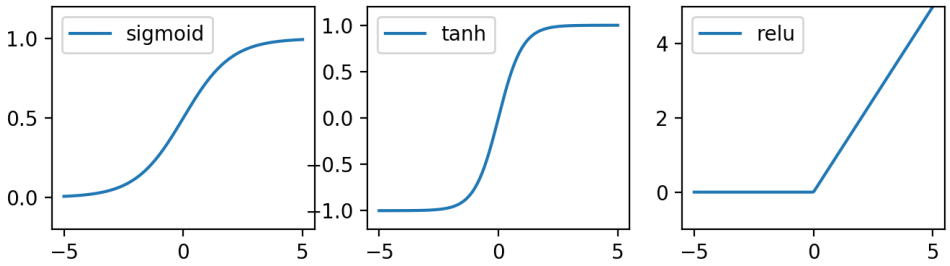


Figure 4.2: Plots of the three activation functions.

The detailed comparison of different activation functions can be found in many references. Although the Sigmoid and Tanh have been used in the past, their usage is avoided nowadays. Since the ReLU avoids and rectifies the vanishing gradient problem, almost all modern deep learning models use ReLU.

There are also many other types of layers that are commonly used in deep neural networks. Some of them are:.

- Pooling: layers that can down down-sample or compress the input data.
- Normalization: layers that normalize the input data.
- Dropout: layers that randomly zero some of the elements of the input data. This has proven to be an effective technique for regularization and preventing the co-adaptation of neurons as described in the paper [20].

Loss Function and Optimizer

With the layers introduced above, a basic deep learning model can be built. Initially, the weights and the bias of the layers are filled with small random values (a step called random initialization). The resulting representation of the function is meaningless, Next, the parameters are incrementally adjusted based on a feedback signal. This gradual adjustment is also called the process of training or learning. The training process applies the following process:

- **Step 1**: select a *batch* of training samples $\mathbf{x}$ and corresponding targets $\mathbf{y}$.
- **Step 2**: apply x to the network and obtain the prediction $\mathbf{y_{pred}}$.
- **Step 3**: measure the mismatch between $\mathbf{y_{pred}}$ and $\mathbf{y}$ using a loss function.
- **Step 4**: Using an optimizer, update the weights of the network to reduce the output mismatch on this batch of data.

The training loop stops when a low mismatch between predictions y_{pred} and expected targets y is obtained.

In Step 3, the mismatch is measured with a loss function, which is a measure of the error that needs to be minimized. Examples of loss functions are:

- The mean absolute error (L1 loss), which is the simplest loss function;
- The mean squared error (MSE or L2 loss), that is the most widely used loss function;
- The cross entropy loss, which is useful in training a classification problem.

The most difficult step in the training loop is Step 4, where the network weights are updated. A practical approach is to compute the gradient of the loss with respect to the network's coefficients. Such a method is known as *back propagation*. Popular optimization algorithms include:

- The stochastic gradient descent (SGD) algorithm [52] and its variants, which are a group of classical optimization algorithms with optional momentum to improve performance;
- The Adagrad algorithm [11] and its modification [62];
- Adam algorithm [26], which is a widely used highly efficient optimization algorithm.

Convolutional Layers

Originally, the convolutional neural networks (convnets) are designed to provide an efficient learning method for image classification. It has become the most important branch in the modern era of deep learning research. As suggested by its name, it features convolutional layers which can learn local patterns instead of global patterns as fully-connected layers do. In the case of images, patterns are found in small 2D windows of the inputs. In the case of time series data, patterns would be 1D windows of the samples. An illustration of a 2D convnet is shown in Fig. 4.3. The image spatial features are extracted by *translating* the pattern (the small orange square on top of the blue image) on the image and find the *dot product* between the pattern and a patch of the image. Therefore, its output is the *convolution* between the image and the pattern. When there are multiple features are extracted, the output is a stack of feature maps (the orange layer). Its features will be extracted again by convolving with patterns (the small green square on top of the orange feature map) and propagate to the next layer (the green feature map). In other words, each layer is a collection of the extracted features of its previous layer.

From the perspective of digital signal processing, such patterns in convnets can be considered as finite impulse response (FIR) filters. Specifically, the weights of the learned patterns are equivalent to the filter coefficients. The pattern weights are

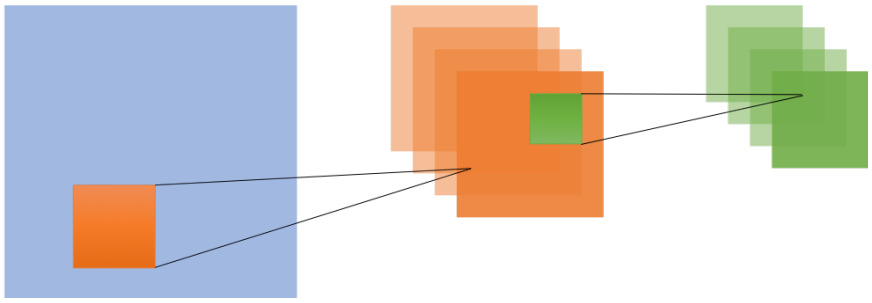


Figure 4.3: Three convolutional layers for 2D image processing.

translation (in the sense of geometry) invariant, while the filter coefficients are kept constant during the filtering. The mathematics behind both the pattern translation and the filtering are convolution operations. However, convnets can learn spatial hierarchies of patterns, and there is no counterpart in FIR filters.

Note that compared to fully-connected layers, the translation invariant patterns also reduce the number of free parameters (weights and bias). This feature reduces the converge time of training deep neural networks significantly.

Deep Learning Frameworks

One key factor that drives the development of deep learning is the existence of many engineering tools. Although there are deep learning codes written in Matlab, C++ and Java, the most popular programming language is Python. Nowadays, few researchers develop their own deep learning frameworks or libraries, because of the availability of mature tensor manipulation frameworks for Python. Some of the most widely used frameworks are listed below.

- **Theano** has been powering scientific deep learning investigations and education since 2007. It is primarily developed by Montreal Institute for Learning Algorithms (MILA) at the Université de Montréal. However, the development ceased in late 2017.

- **TensorFlow** is developed by Google since late 2015. It is used for both research and production and has become the most widely used and supported deep learning library.

- **Keras** is not technically a deep learning framework, but a high-level API, capable of running on top of TensorFlow or Theano. Because of its user friendliness, it allows for easy and fast prototyping. Keras development is backed primarily by Google.

- **Caffe** was originally developed at the University of California, Berkeley. It is known for its expressive architecture and high speed. Caffe is backed primarily by Facebook.

- **PyTorch** is a relative new deep learning framework with its first release in 2017. However, its growth is fast and its support in the industry is increasing. It is also backed by Facebook and will merge Caffe as its backend in a future release.

In this research, most of the codes are written in the latest PyTorch release (0.4.1).

4.3 Synchronization Using Deep Convolutional Networks

In this section, the deep convnet is used to estimate both the symbol timing offset (STO) and the carrier frequency offset (CFO). To be specific, the STO and CFO estimation are approached with CNN models, where synthesized data are used for training. In practice, the measurement data are preferred. However, the purpose of this section is to demonstrate the feasibility that CNN models can be implemented for synchronization applications for wireless communications. In addition, by modifying network parameters and training dataset, the resulting networks can adapt to various modulation, pulse shaping or channel conditions.

4.3.1 Symbol Timing Offset Estimation

Since the STO is a continuous value, it is good practice to model the STO estimation as a *regression* problem. As such, the function of the network is defined such that it can produce the estimate of the STO utilizing the input baseband coherent modulated signal samples. The design objectives of the network are: 1) insensitivity to the CFO in the signal; 2) capability to compensate for error in the sample rate; 3) capability to correct the STO for different modulation schemes and shaping pulses.

Network Model

The implemented network model has a standard structure. The first few layers are convolutional layers incorporated with ReLU and pooling layers. Two fully-connected layers are attached after the convolutional layers to generate a single output representing the STO estimate. The network structure is shown in Fig. 4.4. The input of the network are one-dimensional baseband samples with STO. These samples are obtained from the *magnitude* of the complex-valued baseband signal. There are several reasons that the magnitude of the baseband signal is used as the network instead of the complex-valued baseband signal. A first reason is that such operation can avoid the CFO interference. Also, the PyTorch deep learning framework as well as other popular frameworks do not support complex-value input.

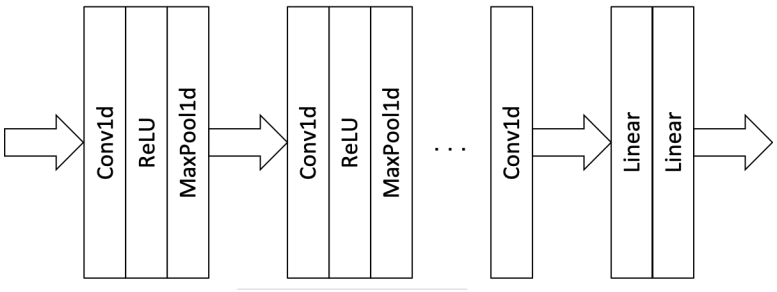


Figure 4.4: CNN based symbol timing offset estimation.

Model Training and Performance

The training dataset are synthesized with random STOs. The network output as the estimate of STO is compared with the true STO directly by applying an L1 loss function. The weights of the network are updated by the Adam optimization algorithm.

Some training parameters are reported here. The signal is originally generated by an oversampling rate of 16 and pulse shaping filter rolloff factor of 0.4. Complex-valued Gaussian noise is added with $E_s/N_0 = 20$ dB. Symbol timing offset is applied by randomly shifting the signal samples and then down-sampling by a factor of 4. As such, the oversampling rate of the input signal is 4. The rolloff factor and the

oversampling rate can be learned by the network. Therefore, if a signal has a different set of parameters, the network can adapt to it with minimum modification and retraining.

Each time, 16 symbols (64 samples) are fed into the network. The network consists of 5 convolutional layers and 2 linear layers. Note that these parameters may not be the optimum candidates. In fact, choosing appropriate hyperparameters (the number of layers and filters, etc.) is not an easy task in the study of deep learning, and it is beyond the scope of this section.

The loss as a function of training epoch is plotted in Fig. 4.5. It can be found that the designed network does converge as the training epoch increases. The convergence is relative fast: after around 250 iterations of training, the normalized STO (with respect to the symbol period) loss has been reduced to about 4%. Note that the training signal and the STO are synthesized randomly, so there is no concern of overfitting (learning features only from the training set, and cannot be generalized) due to limited number of samples in the dataset.

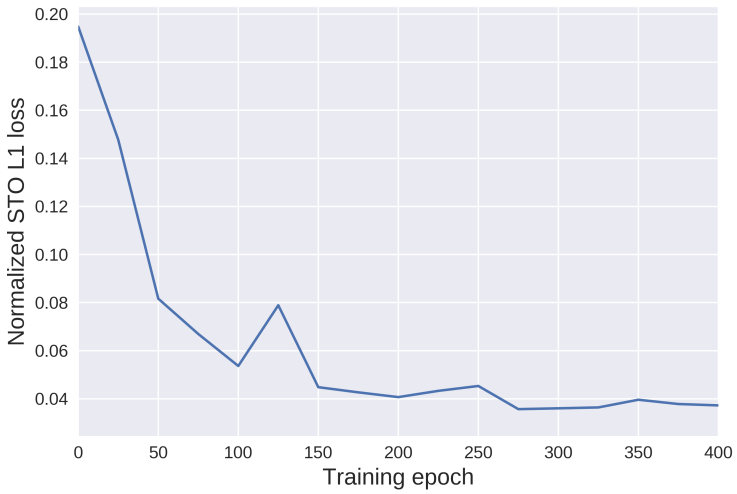


Figure 4.5: The training of CNN based STO estimation.

4.3.2 Carrier Frequency Offset Estimation

After the symbol clock is recovered, the next task of synchronization is to estimate the CFO. A feedforward convnet, similar to the STO estimation is implemented.

The network structure is almost the same as the network used for STO estimation in Fig. 4.4. It consists of a few convolutional layers for the first stage and fully-connected layers for the last stage. There are two major differences with the network in Fig. 4.4. The input samples are down-sampled signals, that are decimated to the symbol rate. In addition, the complex-valued samples cannot be fed into the network directly and the absolute operation cannot be applied, because it will erase the phase information. Instead of the magnitude, the signal *phase* component is taken as the input. It is a reasonable operation with the PSK modulated signal, since the amplitude component does not carry information.

Model Training and Performance

The training data are synthesized QPSK symbols with random CFOs. Because the down-sampling has been done beforehand, the vector size of network input is significantly reduced to 16. Consequently, the network has a reduced complexity, therefore, it requires less convolutional and fully-connected layers or less filters in the convolutional layers. Alternatively, if the CFO needs to be recovered before the STO, a larger network scale is required.

The same loss function and the optimization algorithm as the STO estimation are used. The loss is plotted as a function of training iteration in Fig. 4.6. One distinction in this training process is that the step size used in the optimizer is much smaller than that in the STO estimation. It provides a smoother loss curve but takes more iterations to converge to its minimum error.

A set of collected sea trial data that has been discussed in Section 2.3.1 is processed with the proposed CNN. In Fig. 4.7, the constellation of the down-sampled symbols are plotted. Also, the symbol phases before and after CFO recovery are plotted as a function of time. Each color represents a unique symbol in the QPSK modulation. It is shown that, before the CNN, the symbol phases are highly distorted and the CFO is time variant.

The feedforward CFO recovery requires that the signal samples are separated into small segments with a length of 16 samples with overlaps. Each segment is processed by the network to estimate its local CFO and compensate for it. Before moving to

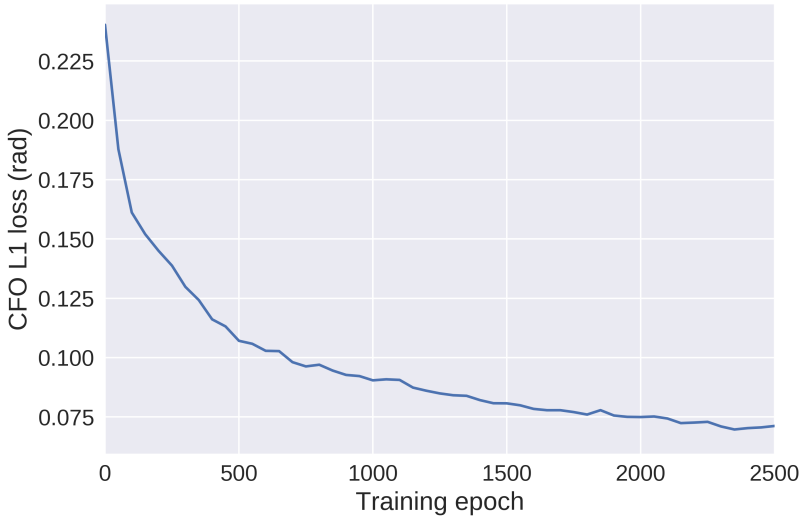


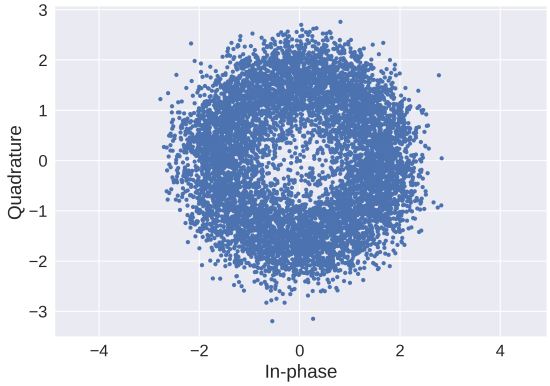
Figure 4.6: The loss in the training of CFO estimation.

the next segment, an initial phase correction that is inherited from all the previous segments is also applied. This step is critical because the symbol phase error is essentially the integration of frequency error through time. More discussion can also be found in the next section. This loop iterates until the end of the symbol sequence, and in Fig. 4.7(c), the symbol phases have been restored successfully.

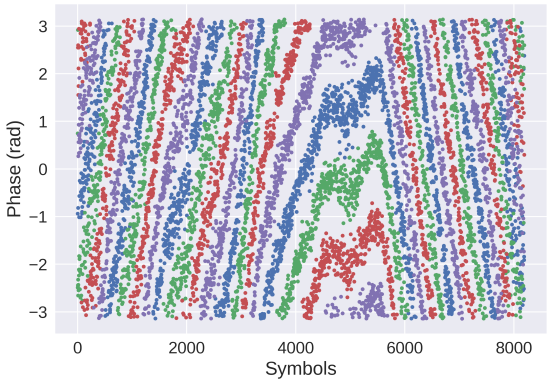
4.4 Passband Receiver Based on Deep Convolutional Networks

In the previous sections, it is shown that the STO and CFO can be estimated by applying trained CNN models. However, the structure of these CNN models is not very different from standard CNN models built with general purpose. In fact, one distinguishing feature of deep neural networks is that despite different scales and applications, they all share very similar components and structures. In other words, various problems that are traditionally solved by sophisticated and customized algorithms can be solved with a standardized structure. As such, algorithms discussed in Chapter 2 can be replaced by trained deep neural networks. This motivates us to design a complete wireless receiver with a CNN. A related work for OFDM receiver can be found in [63].

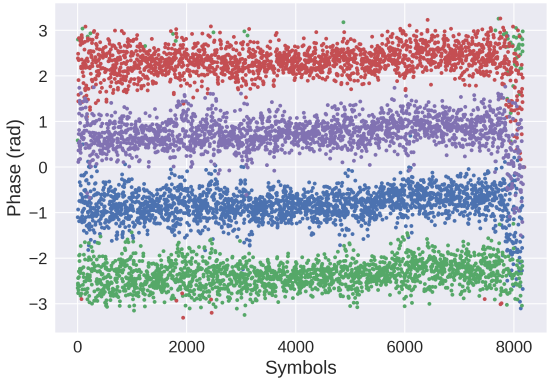
To this end, in this section, a CNN model that can replace a classic coherent



(a) Rotating constellation of the received symbols.



(b) Symbol phases before CFO recovery.



(c) Symbol phases after CFO recovery.

Figure 4.7: CFO recovery with CNN based algorithm.

wireless receiver is designed. To the best of the authors knowledge, there is no related study available. This research is rather experimental and considered to be a proof of concept.

The major function of the network is to convert the passband, over-sampled PSK signals to symbols. Specifically, the design has the following functions: 1) down-convert passband signal into baseband; 2) down-sample signal to symbol rate; 4) provide insensitivity to timing offset; 5) recover carrier frequency offset; 3) produce the symbol phases as output.

Commonly, the signal processing in a wireless receiver is at baseband. In this work, the signal is processed at passband. The most important reason is that the baseband signal is complex-valued, and the complex convolutions can not be simply replaced by two independent convolutions representing the real and imaginary components. Also, there is no well accepted deep learning framework that directly supports complex-valued data, but only a few experimental projects [54]. Moreover, the passband signal preserves all the information, while the down-conversion can be considered as a way for feature extraction, which may not be necessary for deep learning. One obvious drawback of processing at passband is that, the sample rate is usually much higher than the baseband signal. However, for research purpose and considering the low frequency characteristic of underwater acoustic communications, this drawback is acceptable.

Network Structure

The overall CNN based receiver structure is shown in Fig. 4.8. The network consists of two major modules: the demodulation module (Module 1) and the carrier frequency recovery module (Module 2). Note that the output of Module 2 is added with the output of Module 1 to compensate for the carrier frequency offset. This structure resembles the famous Deep Residual Network [19].

To build an actual CNN model, the first task is to prepare the training data set, as well as the input and the output of the model. The input is a segment of over-sampled passband signal. With such an arrangement, one can take advantage of the translation invariant property of the convolutional layers, so that the designed

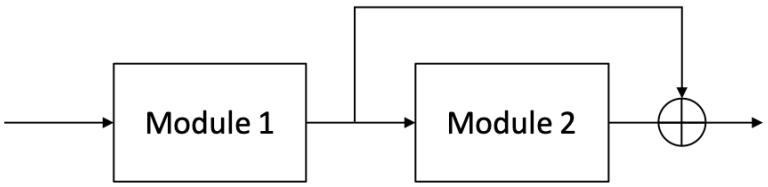


Figure 4.8: CNN based receiver design, top level.

CNN model can tolerate mild symbol timing offsets. If the transmitted signal is M-PSK modulated, the desired outputs from the receiver are symbols representing $0 \sim (M-1)$. Apparently, it suggests that the neural network model generates M unique classes as the output. However, the M discrete symbols are actually extracted from continuous symbol phases ranging from 0 to 2π. Also, considering the need of phase compensation due to carrier frequency offset, it is reasonable to output symbol phases instead of symbol classes.

One modification in this design that is different from standard CNN model is the output layer. As discussed above, there is a single output in this layer representing the symbol phase. This layer has a two-channel input, representing the baseband IQ components. As such, this layer is essentially an arc tangent operation. This receiver module (Module 1) is summarized in Fig. 4.9, where the total number of convolutional layers is adjustable.

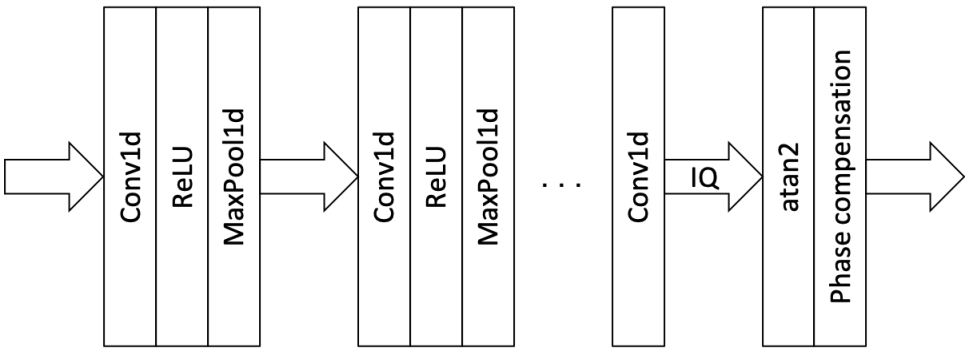


Figure 4.9: CNN based receiver design, Module 1.

The module output is the phase of the passband signal for each symbol. Note that this phase output is the combination of carrier phase and the embedded symbol phase. If the carrier frequency is an integer multiple of the symbol rate, the carrier phase is constant for each symbol. In such condition, the network can easily learn

from the training data set. On the other hand, if the assumption does not stand, the carrier introduced phase component for each symbol is a function of time, such that the network cannot learn from the training set. The solution is to explicitly compensate this carrier introduced phase offset in the output of the Module 1. In fact, the phase difference between two consecutive symbols can be found by:

$$\phi_{diff} = 2\pi \frac{f_c}{f_{sym}}, \tag{4.7}$$

where f_c and f_{sym} are the carrier frequency and the symbol rate respectively.

In Module 1, the carrier phase compensation can not recover the CFO due to the uncertainty of carrier frequency. Therefore, Module 2 is introduced for CFO estimation and its structure is shown in Fig. 4.10.

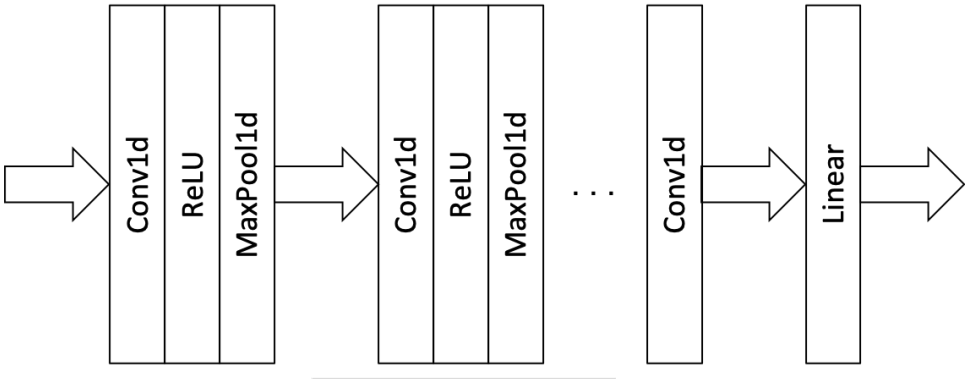


Figure 4.10: CNN based receiver design, Module 2.

Module 2 can be directly borrowed from Section 4.3.2. The input is the symbol phase sequence that is generated by Module 1. After several convolutional layers and fully-connected layers, the output is the CFO estimate. Module 2 can be considered as an CNN implementation of feedforward, non-data-aided CFO estimator. Then, the CFO compensation is applied to the output of Module 1 by adding the phase offset and applying modulus operation to prevent results exceeding $0 \sim 2\pi$.

Training and Performance of the Network

Choosing a right loss function is also critical in building a deep neural network. One reason for using symbol phase as the output is that it is good for designing a loss function. For classification problems, a cross-entropy loss function is the common

choice. However, it dose not reflect the relationship between symbols in terms of the phase. In addition, the standard loss functions for regression, such as MSE loss or L1 loss are not suitable either due to the circularity of the phase representation. For example, if the target phase is 0 and the model output is 1.99π, they are fairly close in the polar coordinate, but the MSE or L1 loss functions will give a large loss that should be minimized. The solution is straightforward. If the absolute difference is greater than π, it will be subtracted by 2π. This modified L1 loss function is given as:

$$loss = \begin{cases} |x-y| & \text{if } |x-y| \leq \pi, \\ 2\pi - |x-y| & \text{if } |x-y| > \pi, \end{cases} \tag{4.8}$$

where x and y represent the network output and the target value respectively.

During the training, L1 loss provides an improved training result compared to the L2 loss in our application. The optimizer is using the Adam algorithm with a learning rate of 0.001. For demonstration purpose, the input data is a synthesized QPSK signal. The symbol rate is set to 1.1 kHz, and the carrier frequency is set to 5 kHz such that their ratio is not an integer. Each input consists of 32 over-sampled symbols. A deep learning model is usually trained in batch to accelerate the convergence. In our case, the number of signals in a batch is 512. Note that the CFOs for 512 signals in a batch are randomly assigned, between ± 10 Hz. No other channel introduced interference is considered except for white Gaussian noise. Detailed code can be found in the Appendix A.

The training results are shown in Fig. 4.11. It can be found that the average loss starts from 90° and decreases with the training epoch. After 2000 epochs, the average loss is 8.4°. Note that unlike training with a limited data set, there is no concern of over fitting when training with randomly synthesized data. So, a verification or test set may not be necessary to examine the network performance: the decrease of loss itself is the figure of merit.

It is also interesting to verify that Module 2 can actually estimate the CFO effectively. The standard deviation of the loss with and without the CFO compensation is plotted in Fig. 4.12. The deviation is calculated with 512 symbols in a batch. In a sequence of 32 symbols, the CFO introduced symbol phase offset increases with time. Therefore, the loss deviation increases with time when no CFO compensation

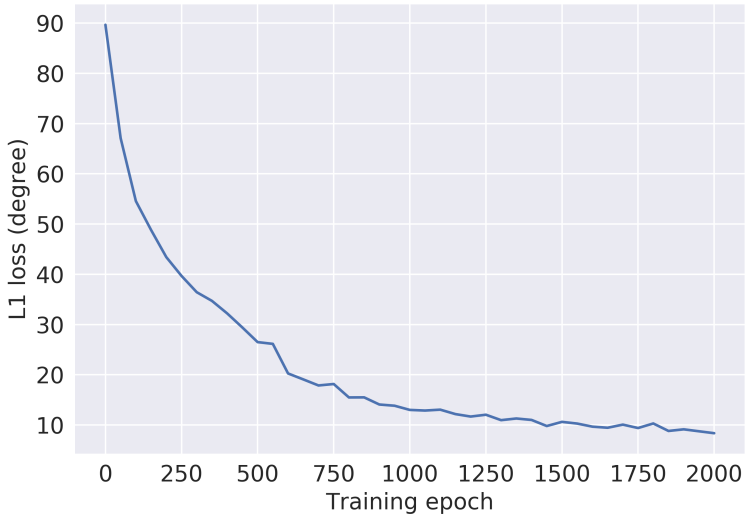


Figure 4.11: Loss decreasing as the training iteration.

is applied. On the other hand, the deviation with CFO compensation keeps a relative low value for all 32 symbols.

4.5 Summary and Future Work

In this chapter, the latest deep learning techniques are implemented to estimate the symbol timing and carrier frequency offset. Additionally, a coherent wireless receiver which is entirely based on a CNN is proposed.

The first section in this chapter is a brief introduction of deep learning techniques. The essential components of a deep learning model for both building and training are introduced, as well as several popular deep learning frameworks.

The second section studies the application of CNN based deep learning models for synchronization tasks. For the estimation of STO, the proposed CNN model is insensitive to the carrier frequency offset, and can adapt to the sampling rate, modulation schemes and the rolloff factors. The neural network proposed for CFO estimation has a similar structure with that used for STO estimation. It has been tested with real sea trial data and good results are produced.

The third section proposes a complete passband coherent receiver based on a CNN

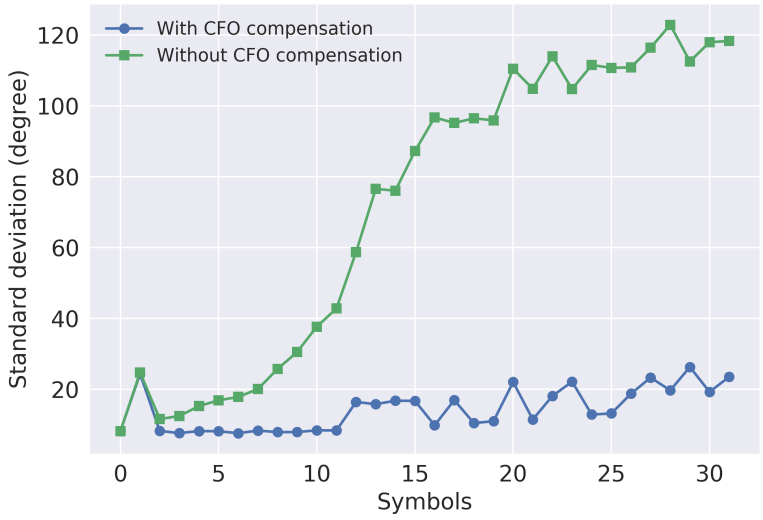


Figure 4.12: Error deviation with and without CFO compensation.

model. There is a significant difference between this model and the conventional receivers in terms of the structure. Except for the down-conversion and down-sampling functions, it also recovers both STO and CFO.

As a proof of concept, the performance of the proposed deep learning models for wireless receivers are not compared with that of the standard synchronizers or receivers. However, because of their strong capability and adaptation, deep neural networks can be fit to complex nonlinear functions. It makes the deep neural network a potential solution for wireless communications in nonlinear or nonparametric channels.

To implement deep learning techniques for underwater acoustic communications, there are still many future works:

- A quantitative performance comparison between traditional solutions and deep learning based solutions.
- A large, comprehensive synthetic data base, and measurement data base for effective model training.
- Application specific model design (complex-valued neural network, etc.)

Chapter 5

Conclusion

The synchronization problem is one of the most important issues in coherent wireless receivers. To be specific, it includes both symbol and carrier recovery. Especially for underwater acoustic communications, due to the channel characteristics, such as the multipath and Doppler effect, the received signal shows spreading in both time and frequency domain. This condition makes the synchronization a critical issue to maintain a reliable underwater acoustic communication link.

This dissertation approaches the synchronization problem for underwater acoustic communications in three different ways: 1) standard synchronization algorithms that used in radio frequency applications, 2) information entropy based symbol timing and carrier frequency estimation, and 3) application of deep learning techniques for synchronization as well as a complete receiver.

The maximum likelihood principle is the foundation of most of the standard synchronization algorithms. In Chapter 2, a comprehensive review of the principle, algorithm and synchronizer structure is presented. Then, a feedforward synchronization implementation is proposed. It is used to compensate for the Doppler introduced symbol and carrier offset. The simulation results show that this structure can jointly synchronize symbol and carrier of the received baseband signal efficiently. To demonstrate the application in a real scenario, the synchronizer performance is tested at sea. The data is collected during sea trials in both benign and highly dispersive conditions. In the benign condition, a basic timing and frequency recovery is needed to demodulate the signal, while in the dispersive condition, strong multipath in combination with the Doppler effect requires a more sophisticated receiver structure. The fractionally spaced decision feedback equalizer is used to compensate for the multipath distortion, and intense frequency offset is recovered by an integrated phase lock loop.

It has been shown that in dispersive channel conditions, the standard maximum

likelihood based synchronizers cannot achieve an optimum solution. This is because their derivation usually assumes a parametric Gaussian channel model. However, this assumption does not hold in time varying, doubly spread channels, such as that in underwater acoustic communications. A unified principle aiming to minimize the information entropy of the received signal is proposed in Chapter 3. To be specific, the symbol timing estimation is achieved by minimizing the eye diagram, while the carrier is recovered by minimizing the constellation diagram. Its implementation and performance are studied extensively. It has been proven that in multipath fading channels, the entropy minimization principle can provide more reliable synchronization estimation results than the maximum likelihood.

In recent years, machine learning techniques are improving at very fast speed. Problems in various fields have been solved using deep learning, a powerful subset of machine learning. However, there are few studies of deep learning applications in wireless communications. In Chapter 4, the deep learning models are implemented as a synchronizer, as well as a complete coherent receiver. Note that unlike standard synchronizers and receivers, the deep learning models are all similar with respect to their structures. Their functions are actually realized by training with specific data set. Although it does not mean that the structure of a deep learning model is not important, this result show that deep learning models have powerful adaptation, such that they can be fit to non-linear functions. This characteristic indicates a potential advantage of replacing the standard receiver components with a deep learning model. If the channel is nonlinear or nonparametric, the carefully trained deep learning model will have superior performance. This inference has not been proved in this chapter, but it is a valuable topic for future research.

Appendix A

Source Code

A.1 Decision Feedback Equalizer with Build in PLL

This piece of Matlab code shows how the FS-DFE with PLL presented in Fig. 2.9 is implemented for sea trial data processing.

```
clc; clear; close all
h = commsrc.pn('GenPoly', [10 7 0],'NumBitsOut', 512);
pn_seq1 = generate(h)*2-1;
pn_seq2 = generate(h)*2-1;
pn_seq = 0.707*(pn_seq1+1j*pn_seq2);

load('data_for_em_60bps.mat') % offset = 2

% truncate the last half data with less distortion
rept = 5;
ref = repmat(pn_seq,rept, 1);
data_for_em = data_for_em(1+512*5*10:512*10*10);

nSym = length(ref);

% prepare the data, sps = 2
offset = 3;
rxTrainPayloadSamp = data_for_em(offset:5:end);
%% equalizer setup
hPSKMod   = comm.PSKModulator(4, ...
    'PhaseOffset',pi/4, ...
    'SymbolMapping','Binary');
hPSKDemod = comm.PSKDemodulator(4, ...
    'PhaseOffset',pi/4, ...
    'SymbolMapping','Binary');
```

```
PSKConstellation = constellation(hPSKMod).';
sampPerSymPostRx = 2;

nFwdWeights = 8;  % Number of feedforward equalizer weights
nFbkWeights = 3;  % Number of feedback filter weights
forgetFactor = 0.96;  % RLS algorithm forgetting factor (sensitive)
alg = rls(forgetFactor);  % RLS algorithm object
eqObj = dfe(nFwdWeights, nFbkWeights,alg, ...
    PSKConstellation,sampPerSymPostRx);
eqObj.ResetBeforeFiltering = 0; % important for iteration

%% PLL setup
Bn = 0.03; % NormalizedLoopBandwidth (sensitive)
xi = 1; % DampingFactor
Kp = 2; % for BPSK QPSK
K0 = 1; % SamplesPerSymbol
theta = Bn/(xi+1/(4*xi));
d = 1+2*xi*theta+theta^2;
g1 = 4*(theta^2/d)/(Kp*K0);
gp = 4*xi*(theta/d)/(Kp*K0);
lambda = 0;
phi = 0;
lambdaHis = zeros(1, nSym); % optional save the phase history

rxTrainPayloadSym = nan(nSym,1);
err = rxTrainPayloadSym;
nTrain = 1000;
xTrain = ref;

for n = 1:nSym
    if n ≤ nTrain % training
        [rxTrainPayloadSym(n), ¬, err(n)] = equalize(eqObj,...
            rxTrainPayloadSamp(n*sampPerSymPostRx-1: ...
            n*sampPerSymPostRx)*exp(-1i*lambda),xTrain(n));
        errPh = angle(rxTrainPayloadSym(n)*xTrain(n)'); % DA phase err
```

```
    else % decision directed
        [rxTrainPayloadSym(n), d, err(n)] = equalize(eqObj, ...
            rxTrainPayloadSamp(n*sampPerSymPostRx-1: ...
            n*sampPerSymPostRx)*exp(-1i*lambda));
        errPh = angle(rxTrainPayloadSym(n)*d'); % NDA phase err
    end
    % update PLL
    phi = g1*errPh + phi;

    % comment this line to disable the PLL
    lambda = (gp*errPh+phi)+lambda;

    lambdaHis(n) = lambda;
end

% calculate err for payload
rxPayloadSym = rxTrainPayloadSym(1+nTrain:end);
rxData = step(hPSKDemod, rxPayloadSym);
txData = step(hPSKDemod, ref(1+nTrain:end));
[¬, BER_DFE_PLL] = biterr(txData, rxData);
BER_DFE_PLL
figure
plot(rxPayloadSym(ref(1+nTrain:end)==ref(1)),'o'); hold
plot(rxPayloadSym(ref(1+nTrain:end)==ref(2)),'o');
plot(rxPayloadSym(ref(1+nTrain:end)==ref(4)),'o');
plot(rxPayloadSym(ref(1+nTrain:end)==ref(14)),'o');
xlim([-2, 2])
ylim([-2, 2])
hold off
```

A.2 Passband Receiver Based on Deep Convolutional Networks

This piece of Python code shows how to build a CNN based wireless coherent receiver.

```
import torch
from torch import nn
import numpy as np
import matplotlib.pyplot as plt


def rrcosdesign(beta, span, sps):
    n = sps*span
    rrc = np.zeros(n, dtype=float)

    for x in np.arange(n):
        t = (x-n/2)/sps
        if t == 0.0:
            rrc[x] = 1.0 - beta + (4*beta/np.pi)
        elif beta != 0 and t == 1/(4*beta):
            rrc[x] = ((beta/np.sqrt(2))*(((1+2/np.pi)*(np.sin(np.pi/(4*
                     beta))))+((1-2/np.pi)*(np.cos(np.pi/(4*beta))))))
        elif beta != 0 and t == -1/(4*beta):
            rrc[x] = ((beta/np.sqrt(2))*(((1+2/np.pi)*(np.sin(np.pi/(4*
                     beta))))+((1-2/np.pi)*(np.cos(np.pi/(4*beta))))))
        else:
            rrc[x] = ((np.sin(np.pi*t*(1-beta))+4*beta*(t)*np.cos(np.pi*
                     t*(1+beta)))/(np.pi*t*(1-(4*beta*t)*(4*beta*t))))

    return rrc


def upsample(x, n):
    zo = np.zeros((len(x), n), dtype=x.dtype)
    zo[:, 0] += x
    return zo.flatten()

# get the CNN, optimizer and loss func ready
```

```
CUDA = True
LR = 1e-3  # learning rate
dtype = torch.float


class NET(nn.Module):
    def __init__(self):
        super(NET, self).__init__()
        # find the IQ component of each symbol
        # down sample by 16
        self.cnn_1 = nn.Sequential(
            # in_channels, out_channels, kernel_size, stride, padding
            nn.Conv1d(1, 32, 3, 1, 1),
            nn.ReLU(),
            nn.MaxPool1d(2),
            nn.Conv1d(32, 16, 3, 1, 1),
            nn.ReLU(),
            nn.MaxPool1d(2),
            nn.Conv1d(16, 8, 3, 1, 1),
            nn.ReLU(),
            nn.MaxPool1d(2),
            nn.Conv1d(8, 4, 3, 1, 1),
            nn.ReLU(),
            nn.MaxPool1d(2),
            nn.Conv1d(4, 2, 3, 1, 1),
        )

        self.cnn_2 = nn.Sequential(
            nn.Conv1d(1, 16, 3, 1, 1),
            nn.ReLU(),
            nn.MaxPool1d(2),
            nn.Conv1d(16, 8, 3, 1, 1),
            nn.ReLU(),
            nn.MaxPool1d(2),
            nn.Conv1d(8, 8, 3, 1, 1),
            nn.ReLU(),
            nn.Conv1d(8, 4, 3, 1, 1),
        )
```

```
        self.lin_2 = nn.Linear(32, 1)

    def forward(self, x, carr_off, time_cfo):
        # down convert, output absolute symbol phase
        symb_IQ = self.cnn_1(x)  # [batch, 1, 32]
        # convert IQ to phase
        symb_phase = torch.atan2(symb_IQ[:, 0, :], symb_IQ[:, 1, :])
        # remove carrier introduced phase
        symb_phase = (symb_phase-carr_off-np.pi) % (2*np.pi)-np.pi
        # estimate CFO
        x2 = symb_phase.unsqueeze(1)  # [batch, 32] to [batch, 1, 32]
        cnn_out2 = self.cnn_2(x2)
        cfo = self.lin_2(cnn_out2.view(cnn_out2.size(0), -1))
        # apply CFO
        cfo_mat = cfo.mm(time_cfo)  # [batch, 1]*[1, 32]=[batch, 32]
        symb_phase_comp = (symb_phase+cfo_mat-np.pi) % (2*np.pi)-np.pi

        return symb_phase, cfo, symb_phase_comp

net = NET()

if CUDA:
    device = torch.device('cuda:0')
    net.cuda()
else:
    device = torch.device('cpu')


optimizer = torch.optim.Adam(net.parameters(), lr=LR)


def circ_loss(input, target, cyc):
    dif = torch.abs(input-target)
    dif[dif > cyc/2] = dif[dif > cyc/2]-cyc
#     return torch.mean(dif**2)
    return dif.abs().mean()
```

```
M = 4  # QPSK
SYMB = 32  # num of symbols
BATCH = 512
beta, span, sps = 0.4, 6, 16
rrc = rrcosdesign(beta, span, sps)
fc = 5.0e3  # carrier freq
baud = 1.1e3

# note there is an assumption that timing is perfect
fs = baud*sps  # signal sample rate 16k>2*fc
# initialization
sig_batch = np.zeros((BATCH, 1, SYMB*sps))  # (128, 1, 512)
msg_batch = np.zeros((BATCH, SYMB))

carr_off = 2*np.pi*fc/baud*torch.arange(0, SYMB).float()
time_cfo = torch.arange(0, SYMB).float().view(1, -1)

if CUDA:
    carr_off = carr_off.cuda()
    time_cfo = time_cfo.cuda()

cyc = 2*np.pi
loss_log = []
for epoch in range(2001):
    # prepare the training input with numpy
    for batch in range(BATCH):
        msg = np.random.randint(0, M, SYMB)  # 0¬3
        sig_mod = np.exp(1j*(np.pi/M+msg*(2*np.pi/M)))  # k/4*pi k=0¬3
        sig_phase = np.angle(sig_mod)
        # pluse shaping
        sig_up = upsample(sig_mod, sps)  # SYMB*sps (512,)
        sig_pulse = np.convolve(sig_up, rrc)  # (607,)
        # up convert
        samp_len = len(sig_pulse)
        time_carr = np.arange(samp_len)/fs

        # include CFO
#         cfo = 0
```

```
        cfo = np.random.rand()*20-10
        carrier = np.exp(1j*(2*np.pi*(fc+cfo)*time_carr))
        sig_pass = np.real(sig_pulse*carrier)  # (607,)
        noise = 0.04*np.random.randn(sig_pass.size)
        sig_pass = sig_pass+noise
        sig_batch[batch, 0, :] = ...
            sig_pass[int(rrc.size/2):int(-rrc.size/2+1)]
        msg_batch[batch, :] = sig_phase

    # numpy to torch tensor
    sig_input = torch.tensor(sig_batch, dtype=dtype, device=device)
    target = torch.tensor(msg_batch, dtype=dtype, device=device)  # ...
        [128, 32]

    # training
    symb_phase, cfo_est, symb_phase_comp = net(sig_input, carr_off, ...
        time_cfo)
    phase_est = symb_phase_comp

    loss = circ_loss(phase_est, target, cyc)
    optimizer.zero_grad()
    loss.backward()
    optimizer.step()
    if epoch % 50 == 0:
        degree = loss.item()/np.pi*180
#         degree = loss.item()**0.5/np.pi*180
        print("epoch = ", epoch, "; loss = %7.4f" % degree)
        loss_log.append(degree)

x = np.arange(0, 2001, 50)
ax = plt.plot(x, loss_log)
plt.xlabel('Training epoch', fontsize=15)
plt.ylabel('L1 loss (degree)', fontsize=15)
plt.xticks(fontsize=15)
plt.yticks(fontsize=15)
plt.show()

x = np.arange(0, 32)
```

```
indx = 35
print(cfo)
y1 = phase_est[indx].cpu().detach().numpy()/np.pi*180
y3 = symb_phase[indx].cpu().detach().numpy()/np.pi*180
y2 = target[indx].cpu().detach().numpy()/np.pi*180
plt.plot(x, y1, label='symbol phase with CFO compensation')
plt.plot(x, y2, label='real symbol phase')
plt.plot(x, y3, label='symbol phase without CFO compensation')
plt.legend(loc='upper left')
plt.show()

std1 = ((phase_est-target)/3.1416*180).std(dim=0)
std2 = ((symb_phase-target)/3.1416*180).std(dim=0)
plt.plot(std1.cpu().detach().numpy(),
         label='With CFO compensation', marker="o")
plt.plot(std2.cpu().detach().numpy(),
         label='Without CFO compensation', marker='s')
plt.legend(loc='upper left', fontsize=12)

plt.xlabel('Symbols', fontsize=15)
plt.ylabel('Standard deviation (degree)', fontsize=15)
plt.xticks(fontsize=15)
plt.yticks(fontsize=15)
plt.show()
```

Bibliography

[1] Z. I. Botev, J. F. Grotowski, and D. P. Kroese. Kernel density estimation via diffusion. *The Annals of Statistics*, 38(5):2916–2957, 10 2010.

[2] Jean-franois Bousquet and Xiao Liu. Predicting the performance of a dual-band bi-directional transceiver for shallow water deployments. In *Proceedings of the 11th ACM International Conference on Underwater Networks & Systems - WUWNet '16*, pages 1–8, New York, New York, USA, 2016. ACM Press.

[3] P. A. Bromiley. Shannon entropy, Renyi entropy, and information. *Statistics and Information Series*, 1(2004):1–8, 2004.

[4] S. Cha. Comprehensive Survey on Distance / Similarity Measures between Probability Density Functions. *International Journal of Mathematical Models and Methods in Applied Sciences*, 1(4):300–307, 2007.

[5] Badong Chen, Lei Xing, Bin Xu, Haiquan Zhao, and Jose C. Principe. Insights Into the Robustness of Minimum Error Entropy Estimation. *IEEE Transactions on Neural Networks and Learning Systems*, 29(3):731–737, 3 2018.

[6] Mandar Chitre. A high-frequency warm shallow water acoustic communications channel model and measurements. *The Journal of the Acoustical Society of America*, 122(5):2580, 2007.

[7] J. Chuang and N. Sollenberger. Burst Coherent Demodulation with Combined Symbol Timing, Frequency Offset Estimation, and Diversity Selection. *IEEE Transactions on Communications*, 39(7):1157–1164, 1991.

[8] Aldo N. D'Andrea, Umberto Mengali, and Ruggero Reggiannini. The Modified Cramer-Rao Bound and its Application to Synchronization Problems. *IEEE Transactions on Communications*, 42(234):1391–1399, 1994.

[9] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *arXiv preprint*, 10 2018.

[10] Mari Carmen Domingo. Magnetic Induction for Underwater Wireless Communication Networks Using RF Electromagnetic Communications. *IEEE Transactions on Antennas and Propagation*, 60(6):2929–2939, 6 2012.

[11] John Duchi, Elad Hazan, and Yoram Singer. Adaptive Subgradient Methods for Online Learning and Stochastic Optimization. *J. Mach. Learn. Res.*, 12:2121–2159, 7 2011.

[12] M.P. Fitz. Planar filtered techniques for burst mode carrier synchronization. In *IEEE Global Telecommunications Conference GLOBECOM '91: Countdown to the New Millennium. Conference Record*, pages 365–369. IEEE, 1991.

[13] Wilfried Gappmair, Stefano Cioni, Giovanni E. Corazza, and Otto Koudelka. Jitter floor elimination for blind feedforward/feedback symbol-timing recovery exploiting the extended zero-crossing property. *International Journal of Satellite Communications and Networking*, 34(5):645–660, 9 2016.

[14] F. M. Gardner. A BPSK/QPSK Timing-Error Detector for Sampled Receivers. *IEEE Transactions on Communications*, 34(5):423–429, 5 1986.

[15] F. M. Gardner. *Demodulator Reference Recovery Techniques Suited for Digital Implementation*. Gardner Research Comp., 1988.

[16] O. Golani, D. Elson, D. Lavery, L. Galdino, R. Killey, P. Bayvel, and M. Shtaif. Experimental characterization of nonlinear interference noise as a process of intersymbol interference. *Optics Letters*, 43(5):1123, 3 2018.

[17] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT Press, 2016.

[18] Hongzhi Guo, Zhi Sun, and Pu Wang. Multiple Frequency Band Channel Modeling and Analysis for Magnetic Induction Communication in Practical Underwater Environments. *IEEE Transactions on Vehicular Technology*, 66(8):6619–6632, 8 2017.

[19] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. *arXiv preprint*, 2015.

[20] Geoffrey E. Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R. Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint*, 7 2012.

[21] Y. Huang and Y. Lu. Small carrier frequency difference detection based on the relative phase entropy. In *2007 International Symposium on Communications and Information Technologies*, pages 1417–1422. IEEE, 10 2007.

[22] M. Johnson, L. Freitag, and M. Stojanovic. Improved Doppler tracking and correction for underwater acoustic communications. In *1997 IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 1, pages 575–578. IEEE Comput. Soc. Press, 1997.

[23] Steven M. Kay. *Fundamentals of Statistical Signal Processing, Volume 1: Estimation Theory*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1993.

[24] D.B. Kilfoyle and A.B. Baggeroer. The state of the art in underwater acoustic telemetry. *IEEE Journal of Oceanic Engineering*, 25(1):4–27, 1 2000.

[25] Tae Kyoung Kim and Moonsik Min. Improved Non-Data-Aided Feedforward Symbol Timing Estimator for Low-Rate Sampling Systems. *IEEE Communications Letters*, 22(5):1010–1013, 5 2018.

[26] Diederik P Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. *arXiv preprint*, abs/1412.6, 2014.

[27] B. Li, S. Zhou, M. Stojanovic, L. Freitag, and P. Willett. Multicarrier communication over underwater acoustic channels with nonuniform Doppler shifts. *IEEE Journal of Oceanic Engineering*, 33(2):198–209, 2008.

[28] Chung-Yi Li, Hai-Han Lu, Wen-Shing Tsai, Zhen-Han Wang, Chung-Wei Hung, Chung-Wei Su, and Yi-Feng Lu. A 5 m/25 Gbps Underwater Wireless Optical Communication System. *IEEE Photonics Journal*, 10(3):1–9, 6 2018.

[29] Xiao Liu and Jean-Francois Bousquet. Symbol Timing and Carrier Frequency Recovery Based on Entropy Minimization. *IEEE Access*, pages 1–1, 2018.

[30] Xiao Liu and Jean-Franois Bousquet. Entropy Minimization Based Synchronization Algorithm for Underwater Acoustic Receivers. In *Proceedings of the International Conference on Underwater Networks & Systems - WUWNET'17*, pages 1–5, New York, New York, USA, 11 2017. ACM Press.

[31] Xiao Liu and Bousquet Jean-Francois. Acoustic Doppler Compensation using Feedforward Retiming for Underwater Coherent Transmission. In *OCEANS'15 MTS/IEEE Washington*, 2015.

[32] J.A. Lopez-Salcedo and G. Vazquez. Asymptotic equivalence between the unconditional maximum likelihood and the square-law nonlinearity symbol timing estimation. *IEEE Transactions on Signal Processing*, 54(1):244–257, 1 2006.

[33] Umberto Mengali and Aldo N. DAndrea. *Synchronization Techniques for Digital Receivers*. Applications of Communications Theory. Springer, 1997.

[34] Umberto Mengali and M. Morelli. Data-aided frequency estimation for burst digital transmission. *IEEE Transactions on Communications*, 45(1):23–25, 1997.

[35] H. Meyr, M. Moeneclaey, and S. A. Fechtel. *Digital Communication Receivers: Synchronization, Channel Estimation, and Signal Processing*. John Wiley & Sons, Inc., New York, USA, 10 2001.

[36] Alberto Morello and Vittoria Mignone. DVB-S2X: the new extensions to the second generation DVB satellite standard DVB-S2. *International Journal of Satellite Communications and Networking*, 34(3):323–325, 5 2016.

[37] Zhiqiang Nie, Shilian Wang, Tian Deng, and Dahai Chen. Research on low-loss and high-speed seabed propagation model for underwater Radio-Frequency-Electromagnetic communication. In *OCEANS 2017 - Aberdeen*, pages 1–8. IEEE, 6 2017.

[38] M. Oerder and H. Meyr. Digital filter and square timing recovery. *IEEE Transactions on Communications*, 36(5):605–612, 5 1988.

[39] Timothy O'Shea and Jakob Hoydis. An Introduction to Deep Learning for the Physical Layer. *IEEE Transactions on Cognitive Communications and Networking*, 3(4):563–575, 12 2017.

[40] Timothy J. O'Shea, Latha Pemula, Dhruv Batra, and T. Charles Clancy. Radio transformer networks: Attention models for learning to synchronize in wireless systems. In *2016 50th Asilomar Conference on Signals, Systems and Computers*, pages 662–666. IEEE, 11 2016.

[41] M. Pedzisz and A. Coatanhay. Minimum entropy approach for carrier frequency recovery. *IEEE Transactions on Wireless Communications*, 5(4):774–778, 4 2006.

[42] Jose C. Principe, Dongxin Xu, Qun Zhao, and John W. Fisher III. Learning from Examples with Information Theoretic Criteria. *The Journal of VLSI Signal Processing*, 26(1/2):61–77, 2000.

[43] P. Qarabaqi and M. Stojanovic. Statistical Characterization and Computationally Efficient Modeling of a Class of Underwater Acoustic Communication Channels. *IEEE Journal of Oceanic Engineering*, 38(4):701–717, 10 2013.

[44] A. Rényi. 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.

[45] I. Santamaría, D. Erdogmus, and J. C. Principe. Entropy minimization for supervised digital communications channel equalization. *IEEE Transactions on Signal Processing*, 50(5):1184–1192, 5 2002.

[46] C. E. Shannon. A Mathematical Theory of Communication. *Bell System Technical Journal*, 27(3):379–423, 1948.

[47] Bayan S. Sharif, Jeff Neasham, Oliver R. Hinton, and Alan E. Adams. A computationally efficient Doppler compensation system for underwater acoustic communications. *IEEE Journal of Oceanic Engineering*, 25(1):52–61, 2000.

[48] A. Shaw, A.i. Al-Shamma'a, S.R. Wylie, and D. Toal. Experimental Investigations of Electromagnetic Wave Propagation in Seawater. In *2006 European Microwave Conference*, pages 572–575. IEEE, 9 2006.

[49] Milica Stojanovic, Josko Catipovic, and John G Proakis. Coherent Communications over Long Range Underwater Acoustic Telemetry Channels. In Jos M F Moura and Isabel M G Lourtie, editors, *Acoustic Signal Processing for Ocean Exploration*, pages 607–612. Springer Netherlands, Dordrecht, 1993.

[50] Milica Stojanovic, Josko a. Catipovic, and John G. Proakis. Phase-coherent digital communications for underwater acoustic channels. *IEEE Journal of Oceanic Engineering*, 19(1):100–111, 1994.

[51] Milica Stojanovic and James Preisig. Underwater acoustic communication channels: Propagation models and statistical characterization. *IEEE Communications Magazine*, 47(1):84–89, 1 2009.

[52] Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton. On the Importance of Initialization and Momentum in Deep Learning. In *Proceedings of the 30th International Conference on International Conference on Machine Learning - Volume 28*, ICML'13, pages III–1139–III–1147. JMLR.org, 2013.

[53] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. Rethinking the Inception Architecture for Computer Vision. *arXiv preprint*, 12 2015.

[54] Chiheb Trabelsi, Olexa Bilaniuk, Ying Zhang, Dmitriy Serdyuk, Sandeep Subramanian, Joo Felipe Santos, Soroush Mehri, Negar Rostamzadeh, Yoshua Bengio, and Christopher J Pal. Deep Complex Networks. *arXiv preprint*, 2017.

[55] Carlos Uribe and Walter Grote. Radio Communication Model for Underwater WSN. In *2009 3rd International Conference on New Technologies, Mobility and Security*, pages 1–5. IEEE, 12 2009.

[56] Y. Wang, K. Shi, and E. Serpedin. Non-Data-Aided Feedforward Carrier Frequency Offset Estimators for QAM Constellations: A Nonlinear Least-Squares Approach. *EURASIP Journal on Advances in Signal Processing*, 2004(13):856139, 12 2004.

[57] Yan Wang, E. Serpedin, and P. Ciblat. Unified performance analysis of blind feedforward timing estimation. In *Conference Record of the Thirty-Sixth Asilomar Conference on Signals, Systems and Computers, 2002.*, pages 639–643, Pacific Grove, CA, USA, 2002. IEEE.

[58] Nathan E. West and Timothy J. O'Shea. Deep Architectures for Modulation Recognition. *2017 IEEE International Symposium on Dynamic Spectrum Access Networks, DySPAN 2017*, 3 2017.

[59] Tsai-Chen Wu, Yu-Chieh Chi, Huai-Yung Wang, Cheng-Ting Tsai, and Gong-Ru Lin. Blue Laser Diode Enables Underwater Communication at 12.4 Gbps. *Scientific Reports*, 7(1):40480, 12 2017.

[60] B. Xie, W. Qiu, and H. Minn. Exact Signal Model and New Carrier Frequency Offset Compensation Scheme for OFDM. *IEEE Transactions on Wireless Communications*, 11(2):550–555, 2 2012.

[61] Jing Xu, Aobo Lin, Xiangyu Yu, Yuhang Song, Meiwei Kong, Fengzhong Qu, Jun Han, Wei Jia, and Ning Deng. Underwater Laser Communication Using an OFDM-Modulated 520-nm Laser Diode. *IEEE Photonics Technology Letters*, 28(20):2133–2136, 10 2016.

[62] Matthew D Zeiler. ADADELTA: An Adaptive Learning Rate Method. *arXiv preprint*, abs/1212.5, 2012.

[63] Zhongyuan Zhao, Mehmet C. Vuran, Fujuan Guo, and Stephen Scott. DeepWaveform: A Learned OFDM Receiver Based on Deep Complex Convolutional Networks. *arXiv preprint*, 10 2018.