Feasibility study of 5G low-latency packet radio communications without preambles

A Master's Thesis

Submitted to the Faculty of the

Escola Tècnica d'Enginyeria de Telecomunicació de Barcelona

Universitat Politècnica de Catalunya

by

Somayeh Jafari Dinani

In partial fulfilment

of the requirements for the degree of

MASTER IN TELECOMMUNICATIONS ENGINEERING

Advisor: Nemesio Javier Villares Piera

Barcelona, September 2019

Title of the thesis: Feasibility study of 5G low-latency packet radio communications without preambles

Author: Somayeh Jafari Dinani

Advisor: Nemesio Javier Villares Piera

Abstract

This thesis deals with the feasibility of having lower latency for radio communication of short packets, which is the major traffic in the fifth generation (5G) of cellular systems. We will examine the possibility of using turbo synchronization instead of using a long preamble, which is needed for Data-Aided (DA) synchronization. The idea behind this is that short packets are required in low-latency applications. The overhead of preambles is very significant in case of short packets. Turbo synchronization allows to work with short or null preambles. The simulations will be run for a turbo synchronizer which has been implemented according to the Expectation Maximization (EM) formulation of the problem. The simulation results show that the implemented turbo synchronizer outperforms or attains the DA synchronizer in terms of reliability, accuracy and acquisition range for carrier phase synchronization. It means that the idea of eliminating the preamble from the short packet seems practical. The only downward is that there is a packet size limitation for the effective functionality of turbo synchronizer. Simulations indicate that the number of transmitted symbols should be higher than 128 coded symbols.

Dedication: To those who inspire me, especially my mom Zahra and my dad Bahram.

Acknowledgements

I would first like to thank my thesis advisor Prof. Nemesio Javier Villares Piera of the department of Signal Theory and Communications at Technical University of Catalonia. He was always available and receptive whenever I had a question about my research or got into trouble.
I would also like to thank the experts who were involved in the validation survey for this research project: Prof. Meritxell Lamarca Orozco and Prof. Luis Javier de la Cruz Llopis.
Last but not least, I would like to thank the department of Signal Processing and Communications for allowing me to access to the TSC Calcula Cluster to run my simulations.

Revision history and approval record

Revision	Date	Purpose
0	20/06/2018	Document creation
1	31/05/2019	Document revision
2	21/09/2019	Document revision

Written by:		Reviewed and approved by:	
Date	18/09/2019	Date	29/10/2019
Name	Somayeh Jafari Dinani	Name	Javier Villares
Position	Project Author	Position	Project Supervisor

Table of contents

List of Figures

List of Tables

1. Introduction

The concept of Internet of things is supposed to bring wireless connectivity to anything which is capable to be connected, which ranges from static tiny sensors to drones and cars. In this scenario, the number of devices which will be connected is huge. Also, there are some critical applications (e.g., autonomous driving, industrial automation, etc.) that require lower latency and higher reliability. So far, each new generation of cellular system has been designed to surpass the previous generation in terms of data rate. But 5G departs from this scheme. By the best of our knowledge, the design of 5G wireless system standard aimed at addressing all the aforementioned challenges globally.
At the first step, according to its typical scenarios or applications, the researchers considered three dimensions performance metric cube based on throughput, number of links and delay. Then, ITU IMT-2020 has divided the 5G network services into three categories which are shown in Figure 1.1:

1. Enhanced mobile broadband (eMBB)
2. Ultra-reliable low-latency communications (URLLC)
3. Massive machine type communications (mMTC)

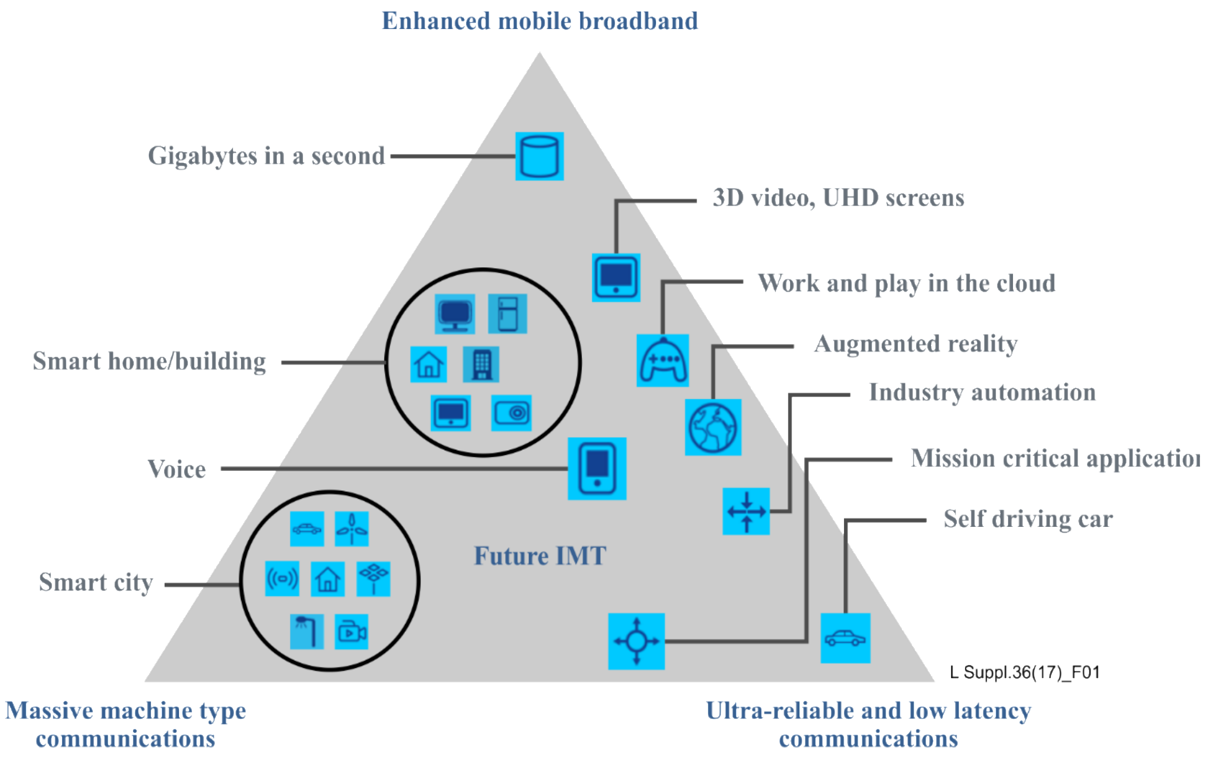

Enhanced mobile broadband
Gigabytes in a second
3D video, UHD screens
Work and play in the cloud
Augmented reality
Industry automation
Mission critical applicatio
Self driving car
Smart home/building
Voice
Smart city
Future IMT
L Suppl.36(17)_F01
Massive machine type communications
Ultra-reliable and low latency communications

Figure 1. 1. 5G services from ITU-T L.1310 – Study on methods and metrics to evaluate energy efficiency for future 5G systems [1]

To make these services a reality, three potential research directions have been defined according to the typical layered protocol architecture:

1. **Network Architecture:** in which both the information technology (IT) and the telecom industries have promoted the concepts of “Software Defined Networks (SDN) [2] and “Network Functionality Virtualization” (NFV) [3] in order to provide more flexible network control and also to reduce the network capital and operational

expenditures (CAPEX) for these new complex network functionalities and inter-connection relations. Besides, they proposed the decoupling of the control plane signalling [4] from the traditional transmission approaches to alleviate the "signalling storm" problem.

2. **MAC mechanism:** for which they recommended a flexible MAC protocol design with dynamic scheduling and variable packet length characteristic. In order to achieve that, two important research directions have been introduced. One way is to incorporate the software defined MAC concept with local caching capabilities. In software defined MAC concept, different data transmission requirements will be allocated to different types of base stations. Another way is to consider the "user-centric" scheduling where the radio resource management and the transmission mode adaptation are moved from the base station to the user terminal side.
3. **Physical layer schemes:** In order to perfectly match the physical layer design with the flexible MAC mechanism, it is expected to provide a tuneable air interface design with adaptable frame structures, waveforms and other transmission technologies.

In this work, the focus will be on the physical layer scheme, especially for the URLLC category. It is known that the stringent requirements of URLLC services, i.e. ultra- high reliability and low latency, are the most challenging feature of the fifth generation of mobile networks. The problem becomes even more challenging for those services beyond the 5G horizon such as tele-surgery and factory automation which require latencies less than 1ms and packet error rates as low as 10^{-9}. The reason is that ultra-high reliability and low latency are two conflicting requirements. "One could use short packets to reduce latency which in turn causes a severe loss in coding gain" [8]. On the other hand, in order to enhance reliability, it is needed to use strong channel codes eventually paired with retransmission techniques which indeed increase the latency [8]. According to [5], four key metrics, requirements and performance benchmarks are considered for designing the physical layer of URLLC which are:

1) **Latency:** In the physical layer, the main focus is on the user plane latency, which is defined as the time to successfully deliver a data block from the transmitter to the receiver via radio interface in both uplink and downlink directions. The user plane latency consists of four major components: the time-to-transmit latency, the propagation delay, the processing latency and finally the retransmission time. The time-to-transmit latency corresponding to the time to transmit a packet, which is required to be in the order of a hundred microseconds. It is much less than 1ms which is currently considered in 5G. The propagation delay is typically defined as the delay of propagation through the transmission medium, and it depends on the distance between the transmitter and the receiver. The processing latency is the time to perform the encoding and decoding and also the channel estimation and synchronization in the initial transmission.
"The general vision of URLLC requirement by 3GPP is that the user plane average latency should be 0.5 ms for both uplink and downlink, without an associated reliability value [41], [14]".
2) **Reliability:** The reliability is defined as the probability of receiving correctly a frame of K information bits within the admitted latency at a certain channel quality (e.g., 5-percentile signal-to-interference-plus-noise ratio (SINR)) [41]. Sources of failure

from higher layer perspective are: when the packet is lost, or it is received late, or it has residual errors. It is essential to maximize the reliability of every packet, meaning to minimize the error rate, leading to less retransmissions. In this thesis, Packet-Error-Rate (PER) and Bit-Error-Rate (BER) have been used as two metrics in order to compare different methods in terms of reliability.

"The general URLLC requirement according to 3GPP is that the reliability of a transmission of one packet of 32 bytes should be $(1 - 10^{-5})$*, within a user plane latency of 1ms (with or without HARQ) [5], [14]".*

3) **Flexibility:** The flexibility of the channel coding scheme is an important aspect along with the evaluation of the coding performance. Bit-level granularity of the codeword size is desired for URLLC. Therefore, the single–bit information block length granularity should be supported by lifting, puncturing and shortening. In other words, the actual coding rate used for transmission should not be restricted and optimized for specific ranges. Thus, the channel codes will be sufficiently flexible to enable hybrid automatic repeat request (HARQ). However, the number of retransmissions needs to be kept as low as possible to improve latency.[6]
4) **Performance Benchmark:** There are two effects which should be distinguished here to understand better the code design problem for short packets. The first one is that if we decrease the block length, the coding gain will be reduced and the gap between the code performance and the Shannon's limit will grow. The problem is not the code design but it is mainly due to the reduction in channel observations that comes with finite block lengths. It means Ergodic and outage capacity are not applicable. The second effect is that if we decrease the block length, the gap between modern codes such as low density parity check (LDPC) codes or Turbo codes, and the finite length performance bounds will widen significantly. This is often due to the suboptimal decoding algorithms.

In the first part of the thesis, introduction part, the rationale behind this thesis is explained. The second part is devoted to the state of the art of the contributed techniques. In the third part, we address some background information about digital communications and then explain the implemented techniques in detail. In chapter four, first the short packet transmissions will be explained and then there is some explanation about encountered synchronization issue in short packet transmissions. Afterwards the solution used in this work will be explicated. Finally, some metrics will be considered for the assessment of the aforementioned solution. Then, in the fifth part, the simulation results are examined. Eventually, in the conclusion part, the overall result of the work will be mentioned and some future ideas will be proposed for the improvement of the synchronization.

2. State of the art of the technology used or applied in this thesis

One of the key requirements of URLLC is to fulfil low latency. In order to get lower latency, it is needed to support short packet transmissions. The physical layer design rules for short packet transmissions are quite different from classic asymptotic long packet transmissions. One of the reasons is that the use of preamble symbols, which are inserted into each transmitted block to facilitate the receiver synchronization, can incur significant performance losses in short packet transmission. It is shown in [7] that inserting pilots for channel estimation leads to significant rate loss when the coherence block is short. In [8] and [9], they recommend to decrease the preamble size according to the data information size not keeping the size of preamble fixed. In [10], a synchronization word is superimposed to the data symbols instead of being embedded in a frame header. The advantage is that the synchronization word length is as large as the frame itself. In [11] and [20], it is proposed to use the reliable soft data symbols called virtual pilots (VPS) among all possible soft data symbols in order to improve the channel estimation quality and detection and decoding quality. It is observed that the gain obtained by the virtual pilot signals diminishes as the number of virtual pilot symbols increases (saturated) [11]. In [13], the following two short packet structures are compared in terms of detecting and decoding performance: 1) time-multiplexed structure in which a fraction of degrees of freedom (DoFs) is used as a preamble for detection and the remaining for data transmission and 2) the superimposed structure in which all DoFs are used for both packet detection and data decoding. It shows that for delay-constrained data and ACK exchange, there is a trade-off between the DoFs spent for detection and for decoding. In the second structure, the optimal PER has been achieved for higher detection overhead. As a result, PER is higher than in the preamble case. But this structure is advantageous due to its flexibility to achieve optimal operation without the need to use multiple codebooks.

Based on [41], the packet size for control-oriented applications is 20 bytes or smaller.

Generally, in order to have highly reliable transmission in URLLC, a channel code with low code rate is used. Several channel code candidates such as LDPC codes [15], Polar codes [16], tail-biting convolutional codes (TBCC) and Turbo codes were considered for both the eMBB and URLLC data channels. It was recognized that while LDPC codes, Turbo codes and Polar codes have similar performance at large block sizes, they have considerable performance differences at small block sizes.

Although turbo codes are considered as powerful codes, they are not recommended to use for short packet transmission. Turbo codes are very sensitive to synchronization errors. That is, even small mismatches between the transmitter's local oscillator and receiver's local oscillator and small phase shifts introduced by the wireless channel can lead to severe performance degradation [8].

According to [5], LDPC codes have been already selected for the eMBB data channel. But recent investigations show that there exists some error floors for LDPC codes which are constructed by using base graph (BG) 2, which is considered for short block low rate scenarios [5]. However, if one focuses on a low signal-to-noise ratio (SNR), LDPC codes achieve a lower PER according to [17] and [18].

Polar codes, based on [5], have been selected for the eMBB control channel. They outperform LDPC codes without any sign of error floor. According to [20], they bring nearly one dB coding gain in terms of PER over convolutional codes. Also, based on [8], they outperform TBCC for code rate lower than 1/3. In addition, they outperform other codes such as TBCC, LDPC codes and turbo codes in terms of complexity.

Therefore, it has been recommended to use polar codes or LDPC codes for low code rates.

Regarding convolutional codes, the result of [9] is that the shorter the block length, the more favorable are convolutional codes in general. They are not able to outperform the fundamental lower block-code bounds (sphere packing bound (SPB), the improved sphere packing bound (ISPB), and converse bound) in terms of PER but they offer more flexibility in terms of block size than comparable block codes. Also, it is concluded that the gap between the block-code lower bound and the convolutional code performance increases with an increment in the code rate. Furthermore, it is proved that reducing the code rate results in less structural delay, which is the delay that occurs due to the fact that encoder and decoder can only perform their operations once a certain number of symbols is available. For very low structural delay, convolutional codes are able to outperform even lower block code limits for the BER, hence show a general superiority. This superiority shrinks with increasing code rate. In [17], [18], it has been shown that convolutional codes are still the first choice for applications which require a very low data delay and consider the BER as performance criterion. Based on [19], it is concluded that using convolutional codes in Internet of Thing (IoT) systems is preferable over turbo codes or LDPC codes.

Based on [8], although for short packet transmission, terminated convolutional code seems useful, it is not recommended due to its rate loss which is introduced by a zero tail termination. According to [22], TBCC outperforms tail-terminating convolutional code (TTCC) for QPSK modulated signal corrupted by AWGN. It is also seen that TTCC with memory of 6 has worse performance than TTCC with 8 memory and TBCC with 6 memory for all code rates. The lower the code rate the better the block-error-rate performance. TBCC is currently considered within 5G standardization for URLLC [8]. “*The LTE (L1/L2) downlink control channel uses tail-biting 64-state convolutional codes” [25]*. Also, according to [8], it is shown that LTE TBCC outperforms LTE turbo codes for short packet lengths (e.g., 40 bits information block length). They also surpass polar codes for code rate greater than 1/3 in terms of block-error rate. However, it is illustrated that for code rate 1/2, and codeword length 128, the extended Bose-Chaudhuri-Hocquenghem (BCH) code outperforms TBCC with memory of 14 and 11, LDPC codes and polar code in terms of complexity.

It has been shown in [8] that higher modulation order leads to more loss in the spectral efficiency when we reduce the preamble size (the loss increases from 9.8% to 18% when the modulation changes from QPSK to 16-QAM). According to [21], which has performed analyses for different modulation schemes and has compared them to random coding bound (RCB), it is seen that 16-ary and 32-ary non-coherent orthogonal modulation schemes provide reasonable balance between energy and bandwidth efficiency and also lead to better estimate of parameters. It has also been proposed to use non-coherent orthogonal modulations rather than differentially encoded phase shift keying (PSK) which is limited in achieving a high level of energy efficiency despite excellent bandwidth efficiency.

Regarding the TBCC in LTE, although it is seen in [25] and [26] that 2x wrap-around (2 Viterbi decoding iterations) is sufficient to get an acceptable decoding performance, it is not acceptable in terms of latency, especially for short packet transmission. However some lower latency decoders, which are based on the extension of the decoder trellis but not on its full trellis, some fraction of it [22], have been used. There, the ending state does not have to be the same as the starting state on the circular trellis, as in TBCC. The decoded bits are taken from the middle of the trace-back path and a circular shift is applied to restore the right ordering of the decoded bits. It is worth mentioning that the level of trellis extension depends on the code rate. The lower the code rate the less trellis extension, the less decoding complexity and latency. In [17] and [18], the BCJR algorithm and Viterbi decoder provide the best results for very short latencies while the sequential decoding approach performs the best for slightly longer delays. Also, based on [18], it is seen that the stack sequential decoding is more practical than BCJR decoding and Viterbi algorithm when using large memory convolutional codes (punctured).

3. Methodology / project development

In this chapter, first we will explain the background of the digital communication system in brief. Then, the utilized techniques for each of the digital communication system components will be explained in detail.

3.1. Theoretical Background/ Digital Communication System

A basic digital communication system is made of several components, as seen in Figure 3.1.

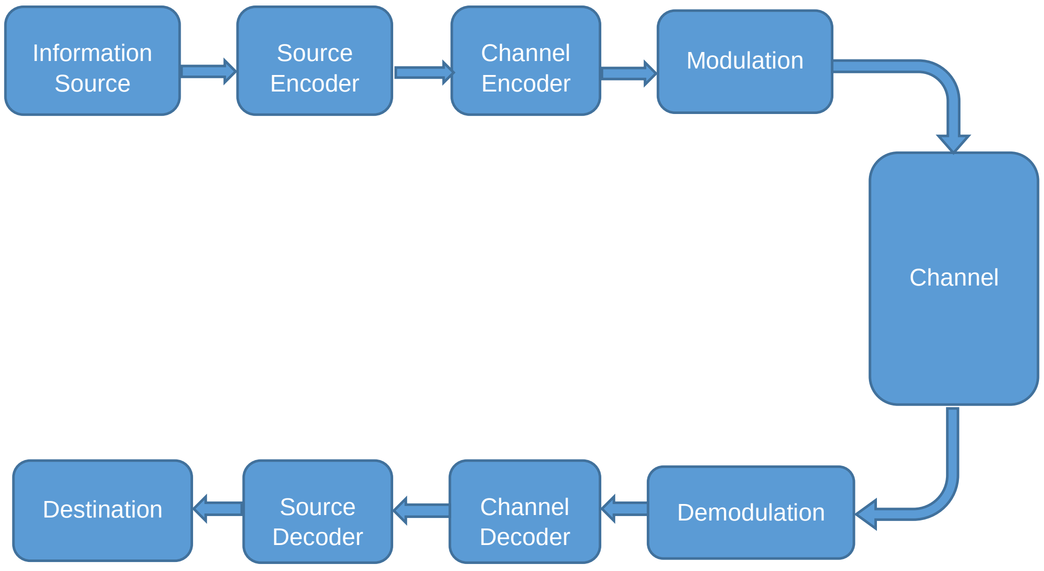


Figure 3. 1. Basic elements of digital communication system [27]

An information source can be any source of data, a continuous waveform or discrete symbols. Then a source encoder converts the source output into a sequence of binary digits with little or no redundancy. This is because of the fact that ideally the source output is expected to be represented by as few binary digits as possible. After, the output of source encoder, which is now called the information sequence, is passed to a channel encoder to add some redundancy to it in a controlled way in order to mitigate the effects of noise and interference of the channel at the receiver side. So in this way, it is possible to increase the reliability of the received signal. The output of this step called a codeword. Afterwards, the codeword goes through a modulator which is an interface to a communication channel and its primary duty is to map the codeword into signal waveforms.

The communication channel is a physical medium that is used to send the signal from the transmitter to the receiver. The essential feature of it is that the transmitted signal is corrupted in a random manner by a variety of possible mechanism such as additive thermal noise.

At the other side of the channel, a demodulator processes the channel-corrupted transmitted waveform and converts the waveforms into a sequence of numbers representing estimates of the transmitted data symbols. Thereafter, it is passed to a channel decoder to reconstruct the original information sequence. This is possible because of the knowledge of the code used by the channel encoder and the redundancy contained in the received data. It is notable to mention that the channel decoder output is not equal to the original information because of some decoding errors caused due to the noise.

At the end of the system, the source decoder reconstructs the original source signal by using the knowledge of the source encoder.

3.2. System Model

In this section, we will explain the different parts of a digital communication system which have been designed in this thesis, and the reasons why each technique has been chosen.

3.2.1. Coding

In late 1940s Shannon showed that it is possible to get arbitrarily low error probability by using coding techniques, provided that the bit rate is below a threshold called "channel capacity".[28]

As a result of this fact, coding is considered for enhancing the performance of the digital communication system in the presence of additive noise. There are two main classical types of channel coding: block codes and convolutional codes.

In this work, based on the following facts, the focus will be on convolutional codes. The first reason is that soft decoding may lead to better performance than hard decoding and performing economical soft decision decoding on convolutional codes is possible. The second is that convolutional codes have the block length and the code rate flexibility which is needed for 5G scenarios. Another fact is that according to [22], although the extended BCH codes, which are circular block codes, perform very close to the normal approximation benchmark given in [23], they are not recommended to be used because of their very complex decoding. The next reason is that based on [21], which is about coding for short packet transmission, convolutional codes have excellent performance in terms of energy efficiency and relative simplicity in comparison with BCH codes and reasonable robustness to imperfections such as estimation errors. And the final reason is that based on [7], capacity achieving codes like LDPC codes and turbo codes do not perform as well as their lengths are reduced. Furthermore, for short block length, less than 1000 symbols, convolutional codes outperform turbo codes.

3.2.1.1. Convolutional Code

Convolutional coding is one of the powerful, effective and widely used error-correcting codes. Convolutional codes are different from block codes because of the existence of memory (m) in the coding scheme. k' information bits enter the encoder at each time and they produce n' binary symbols at the output of encoder and change the state of the encoder. The output depends not only on the recent k' bits that just entered the encoder, but also on the $(L-1)k'$ previous content of the encoder that constitutes its state. The quantity L is defined as the constraint length of convolutional code. The output dependence on the previous information bits causes the encoder to be described as a finite-state machine. The number of states of convolutional code is equal to $2^{(L-1)k'}$. Convolutional codes can also be represented by a shift register of length $k'L$ as shown in Figure 3. 2. It is seen that n' bits at the output of encoder are linear combination of various shift-register bits. The base code rate of convolutional code is defined as $R_c = {k'}/{n'}$ whose unit is information bits per transmission. Because $0 < k' < n'$ are arbitrary integers, $R_c < 1$. Note that inputs and outputs for convolutional codes are binary and convolutional codes are often characterized by the base code rate and the trellis depth $[R_c, 2^{(L-1)k'}]$.

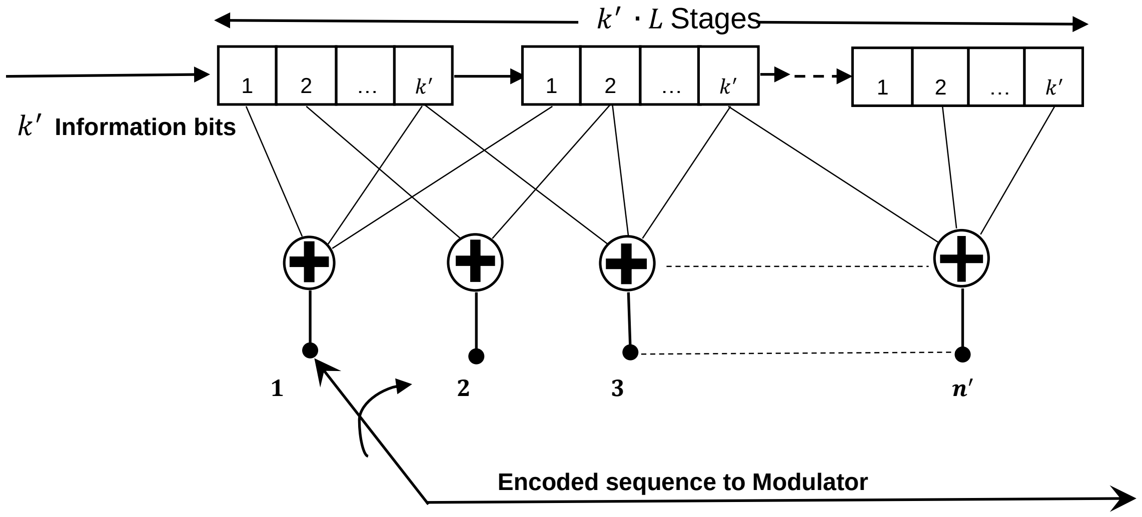


Figure 3. 2. A convolutional encoder

There are three alternative methods that are often used to describe a convolutional code i.e. tree diagram, trellis diagram and state diagram. Depending on the application, the error correction capabilities of convolutional codes can be increased or decreased by decreasing or increasing code rate. The error correction capability depends on constraint length, code rate and generator polynomials.

There are three different termination methods for convolutional codes: truncated, terminated (zero tail termination) and circular (tail-biting). According to [36], the drawback of the truncated convolutional codes is that the block error probability rises in the last bits. Although terminated convolutional codes are expected to be beneficial for short packet transmission due to an improvement in block error probability, they are not recommended because of their rate loss introduced by the zero tail termination. Therefore, tail-biting convolutional codes, which eliminate this rate loss, deserve special attention.

3.2.1.1.1. Tail-biting Convolutional Code

Tail-biting convolutional codes always start from and end at the same state, but starting and ending states are not necessary the all-zero state. Also, this equality of initial state and final state makes the codeword trellis circular as shown in Figure 3. 3. Because the starting and ending states are unknown, the decoding complexity is slightly increased.

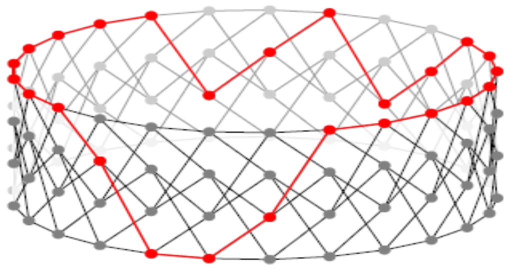


Figure 3. 3. Trellis of tail-biting convolutional code [29]

3.2.2. Modulation

The modulation is the process of converting an input sequence of bits into a waveform which is suitable for transmission over the communication channel. The basic parts of a modulator are two: a procedure for mapping a sequence of binary digits into a sequence of real or complex numbers, followed by an approach for mapping a sequence of numbers into a waveform. Hereinafter, square-QAM (quadrature amplitude modulation), which involves the following three stages, will be adopted; first, mapping a sequence of bits to a sequence of complex symbols; then, mapping the complex symbols to a complex baseband waveform; and finally mapping the complex baseband waveform to a real passband waveform. In this thesis, among other reasons, QAM is selected because it allows the analytical calculation of Code-Aided Cramer Rao bound for the estimation of the carrier phase.

3.2.2.1. Mapping

In QAM, the bit sequence is segmented into m'-tuples. Then, there is a mapping from binary m'-tuples to a set of $M = 2^{m'}$ complex numbers. The set of $2^{m'}$ possible complex numbers resulting from the mapping is called the constellation. Here, the focus will be on 4-QAM (QPSK) because of its simplicity and its lower loss in the spectral efficiency with a short preamble [8].

Suppose that $\{c'_1 c'_2 \dots\}$ denote the incoming binary sequence from the encoding step, where each c'_n is ± 1 (rather than the traditional $0/1$). Then, that sequence is segmented into 2-tuples and converted into a sequence of complex signals a_k chosen from the constellation $\mathcal{A} = \{(1+j), (-1+j), (-1-j), (1-j)\}$ of size $M = |\mathcal{A}| = 2^{m'} = 2^2 = 4$ for QPSK.

3.2.2.2. Pulse Shaping

It is known that the baseband filters in a communication system should satisfy the Nyquist criterion so that symbols can be transmitted over the channel with flat response within a limited frequency band, without ISI.

A QAM baseband modulator is determined by the symbol period T and a waveform $p(t)$. The discrete-time sequence $\{a_k\}$ modulates the amplitude of a sequence of time shifts $\{p(t-kT)\}$ of the basic pulse $p(t)$ to create a complex transmitted signal $a(t)$ as follows:

$$a(t) = \sum_{k \in \mathbb{Z}} a_k \, p(t-kT) \qquad (3.1)$$

As mentioned before, $p(t)$ could be chosen in a way to decay fast with increasing t. This means that $P(f)$ should be a continuous function that goes to zero rapidly but not instantaneously as f increases beyond $W_{m'} = {}^{1}/_{2T}$, where $W_{m'}$ is defined as the nominal baseband bandwidth of the QAM modulator (the Nyquist bandwidth) and $B_{m'}$ is the actual design bandwidth which is slightly larger than $W_{m'}$.

One of the examples of such baseband filters is the raised-cosine filter. To improve noise cancellation, this filter is usually split into two parts, two square-root-raised–cosine filters: one at the transmitter side and the other at the receiver side. The transfer function of the square-root-raised-cosine filter for any given rolloff factor $\alpha' = (\frac{B_{m'}}{W_{m'}} - 1)$, which is between 0 and 1, is:

$$H_{SRRC}(f) = \begin{cases} \sqrt{T} & for\ \ 0 \leq |f| \leq \frac{1-\alpha'}{2T} \\ \sqrt{\frac{T}{2}} \cdot \sqrt{\left(1 + cos\left[\frac{\pi T}{\alpha'}\left(|f| - \frac{1-\alpha'}{2T}\right)\right]\right)} & for\ \frac{1-\alpha'}{2T} < |f| < \frac{1+\alpha'}{2T} \\ 0 & for\ |f| > \frac{1+\alpha'}{2T} \end{cases} \tag{3.2}$$

And its impulse response is:

$$h_{SRRC}(t) = \frac{1}{\sqrt{T}} \frac{\sin\left((1-\alpha')\pi\frac{t}{T}\right) + 4\alpha'\frac{t}{T}\cos\left((1+\alpha')\pi\frac{t}{T}\right)}{\pi\frac{t}{T}\left(1 - \left(4\alpha'\frac{t}{T}\right)^2\right)} \tag{3.3}$$

3.2.2.3. Baseband to Passband (frequency conversion)

In QAM, the complex baseband waveform $a(t)$ is shifted up to passband as $a(t)e^{j2\pi f_c t}$. This waveform is complex and is converted into a real waveform for transmission through the channel by adding its complex conjugate. The resulting real passband waveform is then:

$$x(t) = a(t)e^{j2\pi f_c t} + a^*(t)e^{-j2\pi f_c t} \tag{3.4}$$

The passband $x(t)$ can also be written in the following equivalent ways:

$$x(t) = 2\Re\{\, a(t)e^{j2\pi f_c t}\} = 2\Re\{\, a(t)\}\cos(2\pi f_c t) - 2\Im\{\, a(t)\}\sin(2\pi f_c t) \tag{3.5}$$

3.2.3. The Channel Effects

The most common form of signal degradation comes in the form of additive noise which is generated physically from electronic components and amplifiers at the receiver of the communication system, or interference encountered in transmission, as in the case of radio signal transmission. The first one, which is called thermal noise, is characterized statistically as a Gaussian noise process and has dominant effect on many communication systems.

3.2.3.1. The Effect of AWGN

The simplest mathematical model for a communication channel is the additive noise channel, illustrated in Figure 3.4. In this model, the transmitted signal $x(t)$ is corrupted by an additive random noise process $w(t)$. It is known that for additive noise, the assumption is that the transmitted signal and noise are statistically independent. Mostly, the additive noise is modelled as a Gaussian process whose probability density function is:

$$f_N(n_1) = \frac{1}{\sqrt{2\pi\sigma_N^2}} \exp\left[\frac{-(n_1-\mu)^2}{2\sigma_N^2}\right] \tag{3.6}$$

where $\mu = 0$ is the mean of n_1, and σ_N^2 is its variance.

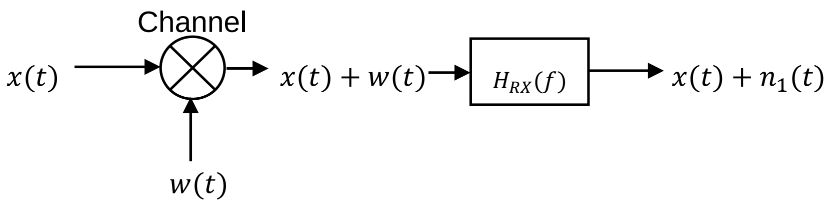


Figure 3. 4. The additive noise channel

It is worth mentioning that $w(t)$ is white and has infinite variance. Therefore, there is a need to filter the received signal so that only the in-band noise, $n_1(t)$, which has finite variance σ_N^2, is considered.

3.2.3.2. Carrier Synchronization Error

In practice, besides the noise, there is uncertainty because of the randomness of certain signal parameters. This randomness is the result of the propagation delay in the transmitted signal. One of the most common unknown signal parameter, which is considered in this work, is the carrier phase offset φ. Taking into account the carrier phase error φ, the lowpass equivalent of the received signal will be:

$$r(t) = a(t)e^{j\varphi} + n_2(t) \tag{3.7}$$

where $n_2(t)$ stands for the lowpass equivalent complex noise at the input of matched filter.

3.2.4. Demodulation

At the receiver side, the demodulator performs the inverse of the modulation operations in the reverse order; first mapping the received passband waveform into a baseband waveform, then recovering the sequence of complex symbols, and finally recovering the binary digits.

3.2.4.1. Passband to Baseband

The received signal can be demodulated to baseband by the reverse of the two steps used in 3.2.2.3. Next the involved operation are explained.

3.2.4.2. Matched Filter

Matched filter is the optimal implementation of the receiver in the presence of AWGN. First it is proved in [37] that the matched filter maximizes the output SNR if the transmitted signal is corrupted by AWGN.

The baseband demodulator is determined by the interval T (the same as at the modulator) and a waveform $q(t) = p(T-t)$. The demodulator filters $r(t)$ by $q(t)$ and samples the output at T-spaced sample times. Denoting the filtered output by

$$z(t) = r(t) * q(t) = \int_{-\infty}^{\infty} r(\tau)q(t-\tau)d\tau = \int_{-\infty}^{\infty} r(\tau)p(T-t+\tau)d\tau \tag{3.8}$$

As a result, it is possible to represent the output $z(t)$ as

$$z(t) = \sum_k a_k e^{j\varphi}\, g(t-kT) + n(t) \tag{3.9}$$

where $g(t)$ is the convolution of $p(t)$ and $q(t)$, and $n(t)$ is the complex noise at the matched filter output. The received complex $z(T), z(2T), \ldots$ are obtained by sampling the signal $z(t)$ at multiples of the symbol period T.

3.2.4.3. De-mapping

This step is the reverse of mapping. It maps a sequence of complex numbers to a sequence of real numbers. Theses real numbers are soft decisions on the sequence of noisy received symbols at the input of the decoder. This step is considered as a part of the demodulation process and, it will be done after synchronization.

3.2.5. Synchronization

Synchronization, from the Greek synchronous [i.e., syn (together) + chronos (time)], denotes the function of making two systems or two signals running exactly together at the same pace. It is a fundamental function in digital communication systems because without initial synchronization, the codewords will not be able to be decoded correctly. Its mission is to estimate certain signal parameters, such as the carrier phase offset, which are necessary in the demodulation and data detection processes. Because the coherent detection scheme is used, it is necessary to estimate and compensate the carrier phase shift before proceeding with data decoding. As such, the synchronization parameters are estimated directly from the received samples at the output of the matched filter.

Traditionally, synchronizers used to operate in either data-aided (DA) or non-data-aided (NDA) modes. Then, with recent advent of powerful coding techniques, these conventional modes have been shown to be unable to properly synchronize state-of-the-art receivers. Therefore, a new family of synchronizers referred to as code-aided (CA) synchronizers were introduced. In the following, a brief explanation for each of these three types of synchronizer has been given.

3.2.5.1. Data-Aided Synchronization (Pilot-Aided), in which a preamble is transmitted along with the data-bearing signal. The preamble is considered completely reliable, meaning that the training sequence, say $\breve{\boldsymbol{a}}$, is perfectly known. The received preamble contains information about the carrier and symbol timing, which is extracted by processing appropriately the received signal. In DA method, the estimation is only based on the first K_{pre} symbols at the matched filter output. Therefore, this method is suboptimal because it does not use the information data symbols to estimate the synchronization parameters. The performance of DA technique can be improved by inserting more preamble symbols or increasing the transmitted power. This leads to unacceptable losses in terms of power and spectral efficiency and impinges directly on the whole throughput of the system and decreases its effective transmission rate.

This method is commonly used in wireless communications where the goal is to minimize the time required to synchronize the receiver with the transmitter. According to [34], the DA phase estimator is:

$$\hat{\varphi}_{DA} = arg\left\{\sum_{k=0}^{K_{pre}-1} \breve{a}_k^* z_k(\hat{v}_{DA}, \hat{\tau}_{DA})\right\} \tag{3.10}$$

where $z_k(v,\tau)$ denotes the matched filter output at the time $kT+\tau$, v is carrier frequency offset and $\breve{a}_k$ is the value of the preamble symbol at time k. In the following, we will assume that the timing error τ and carrier frequency offset v are perfectly known, i.e., $\hat{\tau}_{DA} = \tau$, and $\hat{v}_{DA} = v$.

3.2.5.2. **Non-Data-Aided Synchronization (Non-Pilot-Aided and Non-Code-Aided),** in which the receiver does the synchronization task by extracting the necessary information from the modulated signal $z(t)$ (the output of matched filter). It is assumed that all possible transmitted sequences are a priori equiprobable. As a result, the prior does not provide any information about the transmitted sequence. Because of the two aforementioned reasons, compared to DA synchronization, the downsides of NDA synchronization are: taking more time to establish synchronization and having very poor results at low SNR. The performance of NDA technique can be improved by increasing the observation window size which is limited by the delay tolerated by the system.

According to the Viterbi &Viterbi carrier phase synchronizer, the NDA phase estimator is:

$$\hat{\varphi}_{VV} = \frac{1}{M}\arg\left\{\sum_{k} z_k^M\,(\hat{v},\hat{\tau})\right\} \tag{3.11}$$

where $2\pi/M$ is the minimum angle for which the constellation is rotationally-invariant.

It is worth mentioning that, because the above arg-function only delivers values between $-\pi$ to π, the Viterbi &Viterbi synchronizer is only able to recover phase offsets belonging to the interval $\left[-\frac{\pi}{M}, \frac{\pi}{M}\right]$. Therefore, the Viterbi & Viterbi synchronizer suffers from M-fold ambiguity because of the NDA nature of the synchronizer.

3.2.5.3. Code-Aided Synchronization

The idea behind CA synchronization is to get benefit from the structure of the code, which is used to protect the data, to improve the estimation quality achieved by the synchronizers. The CA synchronizer uses the channel code structure and properties to perform good non-pilot-aided synchronization. Although the CA synchronization may seem as a natural solution for improving synchronization functions, depending on its implementation, its results may differ. There are two main distinguishable approaches among the implementation algorithms [12]. “The first approach consists in modifying the detection /decoding device in order to embed parameter estimation. For example, combined iterative decoding and estimation is performed by modifying the decoder using a sort of pre-survivor estimation technique” [12]. The second one is based on the estimation of the synchronization parameters from some information outputs provided by the decoder. The latter consists of two approaches: one using hard symbol decisions, implying a loss of information about the decision reliability, and the other which uses soft symbol decisions (the symbol hard decision and its reliability).

3.2.6. Decoding

Convolutional codes decoding is based on detection theory and chooses the most likely transmitted codeword. There are basically two criteria that are widely applied to convolutional codes decoding: the maximum-likelihood (ML) criterion in which the codeword (sequence) with the maximum likelihood among all the possible codewords, is selected and the maximum a posteriori probability (MAP) criterion in which the transmitted symbol with the maximum a posteriori probability among all the possible symbols, is selected.

Usually, it is assumed that the incoming binary digits are i.i.d and equiprobable. This means that all codewords are equally likely, which then justifies maximum likelihood decoding. The detector selects the codeword that has the maximum likelihood among all the possible codewords. Therefore, the Viterbi decoder, which implements the ML criterion and is optimal for determining the maximum likelihood sequence (codeword), is used.

It is known that Wrap-Around Viterbi Algorithm (WAVA), which is a circular Viterbi algorithm, is the optimum decoder for decoding of TBCCs. According to [25] and [26], the complexity of using TBCC is very high when the WAVA is used for decoding, and it is mainly dominated by the memory order, especially for short packet transmission [8]. The memory order should be usually large to guarantee good performance in case of short codewords, and the complexity increases exponentially with the memory. As a result, the WAVA is impractical. In addition, if the receiver complexity is limited, the performance of WAVA is not acceptable in terms of latency.
Although, according to [17] and [18], the performance of the BCJR algorithm is the same as Viterbi decoder in terms of the complexity and the impact of the memory, the BCJR algorithm will be used in this thesis because it provides soft decisions, which are required to implement CA synchronization. Additionally, it is the optimal soft-input soft-output (SISO) algorithm in the sense of minimum BER [24].

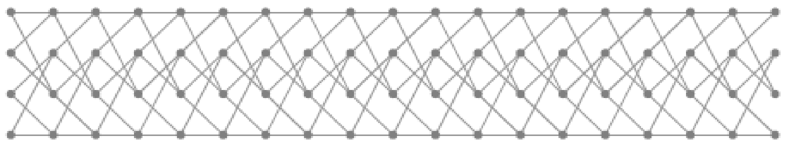
3.2.6.1. BCJR Algorithm

Decoding method of Bahl, Cocke, Jelinek, and Raviv (BCJR), introduced in 1974, is a decoding scheme which is based on the maximum a posteriori (MAP) algorithm [27]. The BCJR applies to Markov data source. It makes decisions on a symbol-by-symbol basis, but each symbol decision is based on an observation of the whole received signal sequence. The BCJR algorithm, which is a trellis-based decoding algorithm, generates a posteriori probability for each received bit/symbol and can be used in an iterative decoding scheme.

It provides a hard decision on each received bit/symbol and the a posteriori probability metric serves as a measure for the reliability of the hard decision.

In order to apply this algorithm for decoding of a tail-biting convolutionally coded sequence, first we need to unroll the circular tail-biting trellis as shown in Figure 3.5. Therefore, the trellis will have the same initial and final states.

Figure 3. 5. The unrolled trellis [29]



We know that convolutional codes are finite memory encoders in which the output and the next state depend on the current state and the input. If $\boldsymbol{b} = (b_1 b_2 \ldots b_K)$ is the information sequence of length K where the length of b_k is $k' = 1$ and $b_k \in \{0,1\}$, $\boldsymbol{c} = (\boldsymbol{c_1 c_2} \ldots \boldsymbol{c_K})$ is the corresponding encoded sequence where the length of $\boldsymbol{c_k}$ is n' and σ_k is the encoder state at time k, we will have for $1 \leq k \leq K$:

$$\boldsymbol{c_k} = f_{co}(b_k, \sigma_{k-1}) \tag{3.12}$$

$$\sigma_k = f_s(b_k, \sigma_{k-1}) \tag{3.13}$$

where functions f_{co} and f_s define the codeword and the new state as functions of the input and the previous state. It is clear that any pair of states (σ_k, σ_{k-1}) that satisfies equation (3.13), corresponds either to $b_k = 0$ or to $b_k = 1$. Therefore, it is possible to partition the set of all pairs of states (σ_k, σ_{k-1}), which correspond to all possible transitions, into two subsets S_0 and S_1 corresponding to $b_k = 0$ and $b_k = 1$, respectively.

If $\boldsymbol{z}' = (\boldsymbol{z}'_1, \boldsymbol{z}'_2, \ldots, \boldsymbol{z}'_K)$ is the received vector at the output of de-mapper, decisions on the transmitted information bit b_k are based on the observation $\boldsymbol{z}'$ and are computed using the MAP rule as follows:

$$\hat{b}_k = arg \max_{b_k \in \{0,1\}} P(b_k | \boldsymbol{z}') = arg \max_{b_k \in \{0,1\}} \frac{P(b_k, \boldsymbol{z}')}{P(\boldsymbol{z}')} = arg \max_{b_k \in \{0,1\}} P(b_k, \boldsymbol{z}') \tag{3.14}$$

$$= arg \max_{l \in \{0,1\}} \sum_{(\sigma_{k-1}, \sigma_k) \in S_l} P(\ \sigma_{k-1}, \sigma_k, \boldsymbol{z}') \tag{3.15}$$

where the last equality is concluded from the fact that $b_k = l$ corresponds to all pairs of state $(\sigma_{k-1}, \sigma_k) \in S_l$ for $l = 0,1$.

To solve this, we partition $\boldsymbol{z}'$ into three parts as following:

$$\boldsymbol{z}' = \left(\boldsymbol{z}'^{(\boldsymbol{k-1})}_{\boldsymbol{1}}, \boldsymbol{z}'_{\boldsymbol{k}}, \boldsymbol{z}'^{(\boldsymbol{K})}_{\boldsymbol{k+1}}\right) \tag{3.16}$$

where $\boldsymbol{z}'^{(\boldsymbol{k-1})}_{\boldsymbol{1}} = (\boldsymbol{z}'_1, \boldsymbol{z}'_2, \ldots, \boldsymbol{z}'_{k-1})$ and $\boldsymbol{z}'^{(\boldsymbol{K})}_{\boldsymbol{k+1}} = (\boldsymbol{z}'_{k+1}, \boldsymbol{z}'_{k+2}, \ldots, \boldsymbol{z}'_K)$.

Now, it is possible to rewrite $P(\ \sigma_{k-1}, \sigma_k, \boldsymbol{z}')$ as:

$$\begin{aligned}P(\sigma_{k-1},\sigma_k,\boldsymbol{z}') &= P\left(\sigma_{k-1},\sigma_k,\boldsymbol{z}'^{(\boldsymbol{k-1})}_{\boldsymbol{1}},\boldsymbol{z}'_{\boldsymbol{k}},\boldsymbol{z}'^{(\boldsymbol{K})}_{\boldsymbol{k+1}}\right)\\ &= P\left(\sigma_{k-1},\sigma_k,\boldsymbol{z}'^{(\boldsymbol{k-1})}_{\boldsymbol{1}},\boldsymbol{z}'_{\boldsymbol{k}}\right)P\left(\boldsymbol{z}'^{(\boldsymbol{K})}_{\boldsymbol{k+1}}|\,\sigma_{k-1},\sigma_k,\boldsymbol{z}'^{(\boldsymbol{k-1})}_{\boldsymbol{1}},\boldsymbol{z}'_{\boldsymbol{k}}\right)\\ &= P\left(\sigma_{k-1},\boldsymbol{z}'^{(\boldsymbol{k-1})}_{\boldsymbol{1}}\right)P\left(\sigma_k,\boldsymbol{z}'_{\boldsymbol{k}}\middle|\sigma_{k-1},\boldsymbol{z}'^{(\boldsymbol{k-1})}_{\boldsymbol{1}}\right)P\left(\boldsymbol{z}'^{(\boldsymbol{K})}_{\boldsymbol{k+1}}|\,\sigma_{k-1},\sigma_k,\boldsymbol{z}'^{(\boldsymbol{k-1})}_{\boldsymbol{1}},\boldsymbol{z}'_{\boldsymbol{k}}\right)\\ &= P\left(\sigma_{k-1},\boldsymbol{z}'^{(\boldsymbol{k-1})}_{\boldsymbol{1}}\right)P(\sigma_k,\boldsymbol{z}'_{\boldsymbol{k}}|\sigma_{k-1})P\left(\boldsymbol{z}'^{(\boldsymbol{K})}_{\boldsymbol{k+1}}|\,\sigma_k\right)\end{aligned} \tag{3.17}$$

where the first three steps follow from the chain rule and the last step follows from Markov properties of the state in a trellis.

At this point, let define $\alpha_{k-1}(\sigma_{k-1})$, $\beta_k(\sigma_k)$ and $\gamma_k(\sigma_{k-1},\sigma_k)$ as:

$$\alpha_{k-1}(\sigma_{k-1}) = P\left(\sigma_{k-1},\boldsymbol{z}'^{(\boldsymbol{k-1})}_{\boldsymbol{1}}\right)$$

$$\beta_k(\sigma_k) = P\left(\boldsymbol{z}'^{(\boldsymbol{K})}_{\boldsymbol{k+1}}|\,\sigma_k\right)$$

$$\gamma_k(\sigma_{k-1},\sigma_k) = P(\sigma_k,\boldsymbol{z}'_{\boldsymbol{k}}|\sigma_{k-1})$$

Using these definitions in equation (3.17), we have

$$P(\sigma_{k-1},\sigma_k,\boldsymbol{z}') = \alpha_{k-1}(\sigma_{k-1})\gamma_k(\sigma_{k-1},\sigma_k)\beta_k(\sigma_k) \tag{3.18}$$

Then, the equation (3.15) can be rewritten as

$$\hat{b}_k = arg\max_{l\in\{0,1\}}\sum_{(\sigma_{k-1},\sigma_k)\in S_l}\alpha_{k-1}(\sigma_{k-1})\gamma_k(\sigma_{k-1},\sigma_k)\beta_k(\sigma_k) \tag{3.19}$$

This equation indicates that calculation of the values of $\alpha_{k-1}(\sigma_{k-1})$, $\beta_k(\sigma_k)$ and $\gamma_k(\sigma_{k-1},\sigma_k)$ is needed for MAP decoding.

In order to facilitate the computation, there is a need to derive recursion relations for $\alpha_{k-1}(\sigma_{k-1})$ and $\beta_k(\sigma_k)$.

The Forward Recursion for $\boldsymbol{\alpha_k(\sigma_k)}$: $\alpha_k(\sigma_k)$ can be obtained by using a forward recursion of the following form as proved in [37].

$$\alpha_k(\sigma_k) = \sum_{\sigma_{k-1}\in\Sigma}\gamma_k(\sigma_{k-1},\sigma_k)\alpha_{k-1}(\sigma_{k-1}),\quad 1\le k\le K \tag{3.20}$$

where Σ denotes the set of all states. This relation means that if the values of $\gamma_k(\sigma_{k-1},\sigma_k)$ are given, it is possible to obtain $\alpha_k(\sigma_k)$ from $\alpha_{k-1}(\sigma_{k-1})$. Assuming that the trellis starts in the all-zero state, then the initial condition for the forward recursion becomes:

$$\alpha_0(\sigma_0) = P(\sigma_0) = \begin{cases}1 & \sigma_0 = 0\\ 0 & \sigma_0 \neq 0\end{cases} \tag{3.21}$$

Equations (3.20) and (3.21) provide a complete set of recursions for computing the values of α.

The Backward Recursion for $\boldsymbol{\beta_k(\sigma_k)}$: $\beta_{k-1}(\sigma_{k-1})$ can be obtained by using a backward recursion of the following form as proved in [37].

$$\beta_{k-1}(\sigma_{k-1}) = \sum_{\sigma_k \in \Sigma} \beta_k(\sigma_k)\gamma_k(\sigma_{k-1}, \sigma_k), \quad 1 \le k \le K \tag{3.22}$$

Assuming that the trellis is terminated in the all-zero state, the boundary condition for the backward recursion becomes:

$$\beta_K(\sigma_K) = \begin{cases} 1 & \sigma_K = 0 \\ 0 & \sigma_K \neq 0 \end{cases} \tag{3.23}$$

These two recursive relations together with their conditions provide the necessary equations to determine $\alpha's$ and $\beta's$ when $\gamma's$ are known.

Computing $\boldsymbol{\gamma_k(\sigma_{k-1}, \sigma_k)}$: It is possible to rewrite $\gamma_k(\sigma_{k-1}, \sigma_k)$ according to the conclusion of the equation (3.13) which mentions that there is a one-to-one mapping between a pair of states (σ_{k-1}, σ_k) and the input b_k as

$$\gamma_k(\sigma_{k-1}, \sigma_k) = P(\sigma_k, \boldsymbol{z}'_{\boldsymbol{k}}|\sigma_{k-1}) = P(\sigma_k|\sigma_{k-1})P(\boldsymbol{z}'_{\boldsymbol{k}}|\sigma_k, \sigma_{k-1}) = P(b_k)P(\boldsymbol{z}'_{\boldsymbol{k}}|b_k)$$

$$= P(b_k)P(\boldsymbol{z}'_{\boldsymbol{k}}|\boldsymbol{c_k}) \qquad 1 \le k \le K \tag{3.24}$$

It is seen that $\gamma_k(\sigma_{k-1}, \sigma_k)$ depends on $P(b_k)$ which is the prior probability of the information sequence at time k and $P(\boldsymbol{z}'_{\boldsymbol{k}}|\boldsymbol{c_k})$ which depends on the channel characteristics.

The equation (3.19) together with the forward equation, the backward equation and the equation (3.24) which are for calculating α, β and γ, respectively, are known as the BCJR algorithm.

Note the BCJR algorithm finds the most likely individual bits or symbols. The BCJR algorithm also provides the values of $P(b_k|\boldsymbol{z}')$. These values provide the level of certainty of the decoder about the value of b_k and are called soft outputs or soft values. Therefore, it is possible to find the a posteriori Log-likelihood ratio values (soft outputs) as:

$$LLR(b_k) = ln\frac{P(b_k = 1|\boldsymbol{z}')}{P(b_k = 0|\boldsymbol{z}')} = ln\frac{P(b_k = 1, \boldsymbol{z}')}{P(b_k = 0, \boldsymbol{z}')}$$

$$= ln\frac{\sum_{(\sigma_{k-1},\sigma_k)\in S_1} \alpha_{k-1}(\sigma_{k-1})\gamma_k(\sigma_{k-1}, \sigma_k)\beta_k(\sigma_k)}{\sum_{(\sigma_{k-1},\sigma_k)\in S_0} \alpha_{k-1}(\sigma_{k-1})\gamma_k(\sigma_{k-1}, \sigma_k)\beta_k(\sigma_k)} \tag{3.25}$$

A decoder such as the BCJR decoder that accepts soft inputs (the vector $\boldsymbol{z}'$) and generates soft outputs is called soft-input soft-output (SISO) decoder. Note that the decoding rule based on $LLR(b_k)$ soft values is given by:

$$\hat{b}_k = \begin{cases} 1 & LLR(b_k) \ge 0 \\ 0 & LLR(b_k) < 0 \end{cases} \tag{3.26}$$

According to [37], it is known that this version of the BCJR algorithm is not numerically stable, particularly if the trellis length is long. Therefore, an alternative to this algorithm, which is known as the Log-APP (logarithm of a posteriori probability) algorithm, is used.

In the Log-APP algorithm, instead of α, β and γ, their logarithms are defined as:

$$\tilde{\alpha}_k(\sigma_k) = \ln(\alpha_k(\sigma_k))$$

$$\tilde{\beta}_k(\sigma_k) = \ln(\beta_k(\sigma_k))$$

$$\tilde{\gamma}_k(\sigma_{k-1}, \sigma_k) = \ln(\gamma_k(\sigma_{k-1}, \sigma_k))$$

Therefore, the forward and the backward recursions have the following forms:

$$\tilde{\alpha}_k(\sigma_k) = ln\left(\sum_{\sigma_{k-1}\in\Sigma} \exp(\tilde{\gamma}_k(\sigma_{k-1}, \sigma_k) + \tilde{\alpha}_{k-1}(\sigma_{k-1}))\right) \tag{3.27}$$

$$\tilde{\beta}_{k-1}(\sigma_{k-1}) = ln\left(\sum_{\sigma_{k-1}\in\Sigma} \exp(\tilde{\gamma}_k(\sigma_{k-1}, \sigma_k) + \tilde{\beta}_k(\sigma_k))\right) \tag{3.28}$$

with the initial conditions in case of terminated trellis:

$$\tilde{\alpha}_0(\sigma_0) = \begin{cases} 0 & \sigma_0 = 0 \\ -\infty & \sigma_0 \neq 0 \end{cases} \qquad \tilde{\beta}_K(\sigma_K) = \begin{cases} 0 & \sigma_K = 0 \\ -\infty & \sigma_K \neq 0 \end{cases} \tag{3.29}$$

and the a posteriori LLR values are computed as:

$$\begin{aligned} LLR(b_k) = ln&\left[\sum_{(\sigma_{k-1},\sigma_k)\in S_1} \exp(\tilde{\gamma}_k(\sigma_{k-1}, \sigma_k) + \tilde{\alpha}_{k-1}(\sigma_{k-1}) + \tilde{\beta}_k(\sigma_k))\right] \\ &- ln\left[\sum_{(\sigma_{k-1},\sigma_k)\in S_0} \exp(\tilde{\gamma}_k(\sigma_{k-1}, \sigma_k) + \tilde{\alpha}_{k-1}(\sigma_{k-1}) + \tilde{\beta}_k(\sigma_k))\right] \end{aligned} \tag{3.30}$$

These relations are numerically more stable but are not computationally efficient. To improve the computational efficiency, the following notation is introduced:

$$max^*\{\ddot{x}, \ddot{y}\} \triangleq \ln\left(e^{\ddot{x}} + e^{\ddot{y}}\right) \tag{3.31}$$

$$max^*\{\ddot{x}, \ddot{y}, \ddot{z}\} \triangleq \ln\left(e^{\ddot{x}} + e^{\ddot{y}} + e^{\ddot{z}}\right) \tag{3.32}$$

Using these definitions, the recursion formulas are:

$$\tilde{\alpha}_k(\sigma_k) = \max_{\sigma_{k-1}\in\Sigma}{}^*\{\tilde{\gamma}_k(\sigma_{k-1}, \sigma_k) + \tilde{\alpha}_{k-1}(\sigma_{k-1})\} \tag{3.33}$$

$$\tilde{\beta}_{k-1}(\sigma_{k-1}) = \max_{\sigma_k\in\Sigma}{}^*\left\{\tilde{\gamma}_k(\sigma_{k-1}, \sigma_k) + \tilde{\beta}_k(\sigma_k)\right\} \tag{3.34}$$

where the initial conditions are given by equation (3.29). The a posteriori LLR values are given by:

$$LLR(b_k) = \max_{(\sigma_{k-1},\sigma_k)\in S_1}{}^{*}\{\tilde{\gamma}_k(\sigma_{k-1},\sigma_k) + \tilde{\alpha}_{k-1}(\sigma_{k-1}) + \tilde{\beta}_k(\sigma_k)\} - \max_{(\sigma_{k-1},\sigma_k)\in S_0}{}^{*}\{\tilde{\gamma}_k(\sigma_{k-1},\sigma_k) + \tilde{\alpha}_{k-1}(\sigma_{k-1}) + \tilde{\beta}_k(\sigma_k)\} \quad (3.35)$$

It is shown in the [37] that

$$max^{*}\{\ddot{x},\ddot{y}\} = \max\{\ddot{x},\ddot{y}\} + \ln\left(1 + e^{-|\ddot{x}-\ddot{y}|}\right) \quad (3.36)$$

$$max^{*}\{\ddot{x},\ddot{y},\ddot{z}\} = max^{*}\{max^{*}\{\ddot{x},\ddot{y}\},\ddot{z}\} \quad (3.37)$$

The term $\ln(1 + e^{-|\ddot{x}-\ddot{y}|})$ is small when $\ddot{x}$ and $\ddot{y}$ are not close. Its maximum occurs when $\ddot{x} = \ddot{y}$. If $\ddot{x}$ and $\ddot{y}$ are large or when they are not close, we can use the approximation

$$max^{*}\{\ddot{x},\ddot{y}\} \approx \max\{\ddot{x},\ddot{y}\} \quad (3.38)$$

$$max^{*}\{\ddot{x},\ddot{y},\ddot{z}\} \approx \max\{\ddot{x},\ddot{y},\ddot{z}\} \quad (3.39)$$

Theses approximations are valid when $\ddot{x}$, $\ddot{y}$ and $\ddot{z}$ are not close and generally would result in a small performance degradation. The resulting algorithm is a suboptimal implementation of the MAP algorithm and called Max-Log-APP algorithm which has been used in this work. The symbol estimator should be scaled according to the amplitude of the received QPSK symbols in this algorithm.

In the summary, the BCJR algorithm can be carried out as outlined next:

1. Initialize the algorithm with the boundary conditions $\tilde{\alpha}_0(\sigma_0) = \tilde{\beta}_K(\sigma_K)$ considering the trellis is circular as in the case of TBCC.
2. Given prior probabilities $P(b_k)$(usually, the decoding starts assuming equiprobable bit values), the channel characteristics metrics $(P(\boldsymbol{z}'_{\boldsymbol{k}}|\boldsymbol{c}_{\boldsymbol{k}}))$, and the received sequence $(\boldsymbol{z}')$, compute the $\boldsymbol{\gamma}$'s.
3. Use the forward recursion to compute the $\boldsymbol{\alpha}$'s.
4. Use the backward recursion to compute the $\boldsymbol{\beta}$'s.
5. Compute the posterior probabilities $(P(b_k|\boldsymbol{z}'))$, and make a decision.

Note that, in the BCJR algorithm, we have to pass through the trellis once in the forward direction and another in the backward direction. Therefore, the complexity of this algorithm is roughly twice the complexity of the Viterbi algorithm. Also, because in the forward pass the values of $\boldsymbol{\alpha}$ have to be stored, the storage requirements of the BCJR algorithm are more demanding.

It is also worth mentioning that the algorithm needs to know the distribution of the starting and ending trellis states or at least to be assigned a priori probability. For tail-biting convolutional codes, the initial/final state is unknown a priori. Therefore, it starts with equiprobable initial states which leads to approximately true a priori probabilities after recursion [39].

4. Short Packet Transmission

In modern wireless system, each transmitted packet consists of payload bits (data) and control information bits. Usually, the packets are long because of the following two reasons:
1. According to a fundamental result in information theory, when the packet is long, there exists channel codes for which the information payload can be reconstructed with high probability. Intuitively, when the packet is long, both thermal noise and the distortions introduced by the propagation channel are averaged out because of the law of large numbers.
2. Another reason is the fact that when the packet is long, the payload size contained in a packet is much larger than the control information associated with the packet. As a result, the existence of the control information does not deteriorate significantly the efficiency of the overall transmission.
On the contrary, 5G needs to support flexible packet size. We know from [1] that URLLC refers to communication services like reliable cloud connectivity where data packets are exchanged at moderately low throughput but with stringent requirements in terms of reliability and latency. mMTC refers to scenarios like industrial control where a massive number of devices need to be supported in a given area. In this case, reliability must be high to cope with critical events. It is also known that major traffic in mMTC is generated by short control messages. Because it is crucial to use the degrees of freedom (DoFs) in optimal way to send data and control information, it is essential for a 5G system to also support short packet transmission in favour of both mMTC and URLLC as a generic mode of the mMTC. Furthermore, short packets will take less time to transfer resulting into less collisions in the physical layer.
The physical layer design rules for short packet transmission are quite different from its classic asymptotic counterpart where block lengths are assumed asymptotically long. Conventionally, when the packet length is large, the coding and modulation schemes are used to adapt the transmission rate with constant control overhead. As a result of the coding limitations and the significant role of the control information, there is a rethinking necessity for the structure of short packets. The control information part consists of two parts: a preamble (pilot) and a header shown in Figure 4.1. The header is responsible to route a packet from a source to its intended destination by means of addresses and the preamble includes necessary information for packet detection, efficient synchronization (in time, phase and frequency), or estimation of channel state information (CSI), which are needed by the receiver to compensate for distortion of the transmitted signal introduced by the wireless channel.

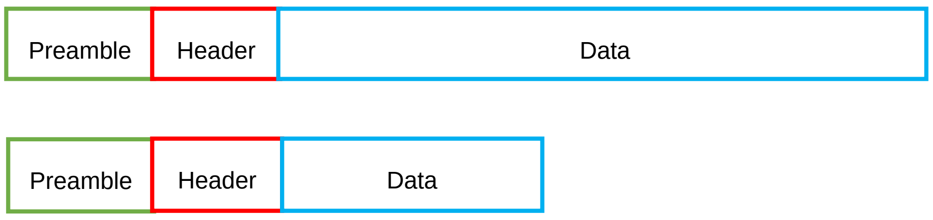
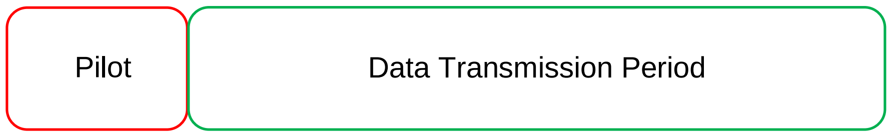


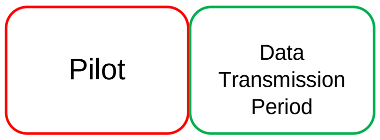
Figure 4. 1. Examples of Short and Long packet with payload and control information

The number of connected devices in 5G is expected to be larger because of IoT (mMTC). Therefore, the header size is not expected to decrease. As a result, in order to optimize the DoFs usage, it is necessary to rethink the preamble part. At first, the researchers decided to keep the size of preamble fixed when the data length decreases as shown in Figure 4.2.b. In this case, the preamble is no longer negligible in size compared to the payload. This results in a significant effect on the overall efficiency of the transmission. Then, they decided to decrease the preamble size proportionally to the data information size as shown in Figure 4.2.c. These two methods multiplex in time the detection sequence and the information data so that a fraction of the DoFs is used as a preamble for detection and the remaining for data transmission. Another idea is to superimpose the preamble word to the data symbols instead of embedding in a frame preamble [10], therefore, all DoFs will be used for both packet detection and data decoding. Finally, the last idea is to eliminate the preamble from the packet.

a) Long data transmission used in current system



b) Short data transmission when the training period is maintained



c) Short data transmission when the current ratio of training period to data transmission period is maintained.

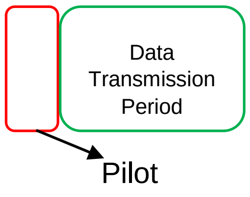


Figure 4. 2. Packet Structure

4.1. <u>Synchronization Problem</u>

As it is mentioned before, there are some upcoming applications which are expected to transmit critical information with low latency and ultra-high reliability in 5G system. Consequently, they should support short packet transmissions. It is also known that in order to be able to decode uniquely the codewords, initial synchronization is needed.
Conventionally, there is a trade-off between improving the performance of frame synchronization and improving the performance of information throughput. Meaning that, the receiver needs to know the channel to decode the information efficiently. For this purpose, the transmitter should insert more preamble symbols into each transmitted packet, which incurs significant performance losses.
If we consider a fixed preamble size, the throughput loss is more noticeable when the packet size is small. If the preamble size is shortened for having lower latency and better usage of DoFs, it is possible that the DA synchronizer would not be able to work properly

because the preamble size is too short to yield accurate estimates. As a result, we need to find a satisfactory synchronizer for this case.
It is notable to mention that the information theorists have mostly viewed the design of preamble as something outside their competence area. As a consequence, the transmission of preamble has been left to heuristic approaches. It means in practice, all the current protocols are based on an assumption that the preamble is perfectly reliable [1].

4.2. Proposed Solution

It is mentioned before that, if we decrease the preamble size proportionally to the data length, such preamble size may not be acceptable for short packets to estimate the synchronization errors. As a consequence, it is proposed to reduce or eliminate the preamble from the packet and consider turbo synchronization instead of DA synchronization in order to improve detection, decoding and also estimation of carrier synchronization errors.

The reason is that the focus of this work is on the physical layer. And in this layer, it is possible to control user plane latency. As it is mentioned in the introduction, the user plane latency consists of the time-to-transmit latency, the propagation delay, the processing latency and retransmission time. We know turbo synchronization works properly at low SNR and using preamble leads to rate inefficiency. We also know that turbo synchronization allows to work with short or null preambles, required for short packet transmission. This will lead to some decrease in the time-to-transmit latency.

But one question which arises here is that whether this solution leads to overall lower latency or not. Does turbo synchronization work properly in short packet transmission? Is there any limitation on the packet size reduction?

4.2.1. Turbo Synchronization

In contrast to DA synchronizers, which most times offer a simple closed-form expression for the ML estimator, CA and NDA synchronizers do not have such simple expression. This calls for an alternative solution based on iterative methods. There are two approaches for iterative methods: the decision-directed (DD) approach and the soft-decision-directed (SDD) approach. In both approaches, synchronization starts from an initial estimate of the synchronization parameters, then a decision is made about the transmitted symbols. After that, this decision is used to compute a new estimate of the synchronization parameters and so forth. The difference between DD and SDD are that SDD uses soft symbol decisions for estimation instead of hard decisions. Intuitively, it means that less reliable decisions should have less weight in the SDD synchronizer.

The concept of SDD synchronization is a key ingredient of turbo synchronization because it is based on the exchange of soft information, in agreement with the turbo principle. There are different methods for the implementation of turbo synchronization such as the EM algorithm, gradient method and the sum-product algorithm [34]. In order to implement turbo synchronization, in this paper, the well-known iterative expectation maximization algorithm (EM) has been used.

4.2.1.1. EM Algorithm [33]

The problem addressed in this section is to find the ML estimate $\hat{\varphi}$ of φ, that is to say the solution of

$$\hat{\varphi} = arg\max_{\tilde{\varphi}}\{\ln P(\boldsymbol{z}|\,\tilde{\varphi}\,)\} \tag{4.1}$$

where $\boldsymbol{z}$ denote a random vector obtained by expanding the received modulated-signal $z(t)$ onto a suitable basis, φ indicates the phase error to be estimated from the observation of the received vector $\boldsymbol{z}, \tilde{\varphi}$ is a trial value of φ and $P(\boldsymbol{z}|\tilde{\varphi}) = \int_{\boldsymbol{a}} P(\boldsymbol{z}|\boldsymbol{a}\,, \tilde{\varphi})P(\boldsymbol{a}\,)da$ where we have assumed that $\boldsymbol{z}$ also depends on a random discrete-valued nuisance parameter $\boldsymbol{a}$ independent of φ and with a priori probability density function $P(\boldsymbol{a})$.

Unfortunately, there is no analytical solution to such a problem. In this case one has to use iterative numerical methods in order to find the solution of (4.1). One of these iterative methods is referred to as the EM algorithm. The EM algorithm alleviates the problem by breaking down the global maximization problem (4.1) into a sequence of easier problems i.e., the M-steps (4.3), according to the choice of the complete data set. The algorithm proceeds in two steps: the expectation step (E-step) and the maximization step (M-step). At iteration (i), we have:

E-step: $Q\left(\tilde{\varphi}, \hat{\varphi}^{(i-1)}\right) = \int P\left(\check{\boldsymbol{z}}\middle|\boldsymbol{z}, \hat{\varphi}^{(i-1)}\right) \ln P(\check{\boldsymbol{z}}|\,\tilde{\varphi})\, d\check{\boldsymbol{z}}$ $\quad(4.2)$

M-step: $\hat{\varphi}^{(i)} = arg\max_{\tilde{\varphi}}\{Q\left(\tilde{\varphi}, \hat{\varphi}^{(i-1)}\right)\}$ $\quad(4.3)$

where $\hat{\varphi}^{(i)}$ is the phase estimate computed at the i-th iteration and $\check{\boldsymbol{z}}$ is related to $\boldsymbol{z}$ by $\boldsymbol{z} = f(\check{\boldsymbol{z}})$, with $f(.)$ denoting a many-to-one mapping. The actual observation set $\boldsymbol{z}$ and the extended observation set $\check{\boldsymbol{z}}$ are usually referred to as the incomplete and the complete data set, respectively. The so-called complete data set may be chosen in many different ways. $\check{\boldsymbol{z}}$ may often selected as an extended observation vector for which the corresponding ML problem is easy to solve. Here, we consider $\check{\boldsymbol{z}} \triangleq [\boldsymbol{z}, \boldsymbol{a}\,]$.

This algorithm converges under fairly general conditions [33] towards the ML estimate (4.1). By using the Bayes rule and the independence of $\boldsymbol{a}$ and φ, we may write:

$$P(\check{\boldsymbol{z}}|\tilde{\varphi}) = P(\boldsymbol{z}, \boldsymbol{a}|\tilde{\varphi}) = P(\boldsymbol{z}|\boldsymbol{a}, \tilde{\varphi})P(\boldsymbol{a}|\tilde{\varphi}) = P(\boldsymbol{z}|\boldsymbol{a}, \tilde{\varphi})P(\boldsymbol{a}) \tag{4.4}$$

After this, the substitution of this result into (4.2) yields

$$Q\left(\tilde{\varphi}, \hat{\varphi}^{(i-1)}\right) = \int P\left(\boldsymbol{a}\middle|\boldsymbol{z}, \hat{\varphi}^{(i-1)}\right) \ln P(\boldsymbol{z}|\boldsymbol{a}, \tilde{\varphi})\, d\boldsymbol{a} + \int P\left(\boldsymbol{a}\middle|\boldsymbol{z}, \hat{\varphi}^{(i-1)}\right) \ln P(\boldsymbol{a})\, d\boldsymbol{a} \tag{4.5}$$

The second term does not depend on $\tilde{\varphi}$ and does not affect the maximization operation in (4.3). Therefore, it can be dropped. As a result, the Q-function has the following structure for the particular case of phase estimation in the presence of an independent nuisance vector $\boldsymbol{a}$.

$$Q\left(\tilde{\varphi}, \hat{\varphi}^{(i-1)}\right) = \int P\left(\boldsymbol{a}\middle|\boldsymbol{z}, \hat{\varphi}^{(i-1)}\right) \ln P(\boldsymbol{z}|\boldsymbol{a}, \tilde{\varphi})\, d\boldsymbol{a} \tag{4.6}$$

It is seen that the estimation of synchronization parameter φ via the EM algorithm only requires the knowledge of posterior probabilities $P\left(\boldsymbol{a}|\boldsymbol{z}, \hat{\varphi}^{(i-1)}\right)$ and the log-likelihood function $\ln P(\boldsymbol{z}|\boldsymbol{a}, \tilde{\varphi})$.

Now, we will apply the general aforementioned framework to the particular case of carrier phase synchronization from the samples at the output of matched filter (3.9). In this context, the nuisance parameter vector $\boldsymbol{a}$ contains the values of the K unknown transmitted data symbols $(a_0, a_1, .., a_{K-1}) \in \mathcal{A}^K$, where $\mathcal{A}$ is the constellation alphabet.

If we neglect terms which are independent of φ, the log-likelihood function can be written as

$$\ln P(\boldsymbol{z}|\boldsymbol{a}, \tilde{\varphi}) = -2\Re\left\{\sum_{k=0}^{K-1} a_k^* z_k(v,\tau) e^{-j\tilde{\varphi}}\right\} + \sum_{k=0}^{K-1} |a_k|^2 \qquad (4.7)$$

where $z_k(v,\tau)$ is the matched filter output.

Let define for each transmitted symbol a_k

$$\eta_k\left(\boldsymbol{z}, \hat{\varphi}^{(i-1)}\right) \triangleq \int a_k P\left(\boldsymbol{a}\middle|\boldsymbol{z}, \hat{\varphi}^{(i-1)}\right) d\boldsymbol{a} = \sum_{a_{m_1} \in \mathcal{A}} a_k P\left(a_k = a_{m_1}\middle|\boldsymbol{z}, \hat{\varphi}^{(i-1)}\right) \qquad (4.8)$$

$$\rho_k\left(\boldsymbol{z}, \hat{\varphi}^{(i-1)}\right) \triangleq \int |a_k|^2 P\left(\boldsymbol{a}\middle|\boldsymbol{z}, \hat{\varphi}^{(i-1)}\right) d\boldsymbol{a} = \sum_{a_{m_1} \in \mathcal{A}} |a_k|^2 P\left(a_k = a_{m_1}\middle|\boldsymbol{z}, \hat{\varphi}^{(i-1)}\right) \qquad (4.9)$$

Using these definitions and replacing (4.7) in (4.6), we get

$$Q\left(\tilde{\varphi}, \hat{\varphi}^{(i-1)}\right) = -2\Re\left\{\sum_{k=0}^{K-1} \eta_k^*\left(\boldsymbol{z}, \hat{\varphi}^{(i-1)}\right) z_k(v,\tau) e^{-j\tilde{\varphi}}\right\} + \sum_{k=0}^{K-1} \rho_k\left(\boldsymbol{z}, \hat{\varphi}^{(i-1)}\right) \qquad (4.10)$$

Consequently, the solution of the maximization step is

$$\hat{\varphi}^{(i)} = arg\left\{\sum_{k=0}^{K-1} \eta_k^*\left(\boldsymbol{z}, \hat{\varphi}^{(i-1)}\right) z_k(v,\tau)\right\} \qquad (4.11)$$

Note that the a posteriori average values $\eta_k\left(\boldsymbol{z}, \hat{\varphi}^{(i-1)}\right)$ can be computed from the marginal posterior probabilities $P\left(a_k\middle|\boldsymbol{z}, \hat{\varphi}^{(i-1)}\right)$. In other words, for this particular case, the implementation of an iterative ML estimation algorithm only requires the evaluation of the marginal posterior probabilities $P\left(a_k\middle|\boldsymbol{z}, \hat{\varphi}^{(i-1)}\right)$. This makes synchronization via the EM algorithm and BCJR decoders complementary since the marginal probabilities needed by the first one can be provided by the second one. This leads to the turbo synchronization algorithm described in Algorithm (1) where Itr denotes the number of performed EM iterations. Step 2 means that the system will perform the BCJR decoding at each EM iteration. Step 3 is simply performed by resolving (4.11).

Algorithm (1)

```
1. $\hat{\varphi}^{(0)} = \varphi_0$;
   for $i = 1 \rightarrow i = Itr$ do
2.     Perform BCJR decoding ;
3.     Computation $\hat{\varphi}^{(i)}$ ;
   end
```

Table 4. 1. EM algorithm

It is known that the final convergence point of the EM algorithm depends on its initialization like most iterative algorithms. Depending on its initialization, the EM algorithm may converge either to a saddle point, or to a local or the global maximum. In practice, the optimal performance depends on how close is the initial estimate to the global maximum. Therefore, at the first step, it is needed to consider an initial phase estimation close to true phase error to start turbo synchronization. This initial step is generally referred to as phase acquisition.

4.2.1.1.1. Initial Phase Estimation (Phase Acquisition)

Usually, initialization of turbo synchronization is done by means of considering short preambles and solving formula (3.10) for computing φ_0 $(\varphi_0 = \hat{\varphi}_{DA})$. As it is mentioned before, since the preamble usage may not be efficient in short packet transmission, we do not consider the preamble here. Therefore, another method is considered to do phase acquisition. It has the following steps:

First the NDA phase estimate, $\hat{\varphi}_{VV}$, is calculated by formula (3.11) from previous chapter. Then, n_s initial phases are obtained from the interval $[\hat{\varphi}_{VV}, \hat{\varphi}_{VV} + 2\pi]$ by the following formula:

$$\varphi_0^{(l)} = \hat{\varphi}_{VV} + \frac{(l-1)2\pi}{n_s} \qquad l = 1,2,\dots,n_s \tag{4.12}$$

Then, each of these n_s phases is considered as an initial phase correction that is applied to the output of matched filter. Then, after phase correction and demapping, the sequence of numbers goes through the decoder. At this step, the BCJR decoder (SISO decoder) computes the (logarithmic) a posteriori probability ratio (LAPPR) of each transmitted (encoded) bit c_n, which constitutes a reliability metric on the decision of c_n. Then, LAPPRs are converted from bit level a posteriori probabilities to symbol level a posteriori probabilities. The LAPPR of n-th coded bit, c_n, is defined as

$$L_n \triangleq \ln \frac{P(c_n = 1|\boldsymbol{z})}{P(c_n = 0|\boldsymbol{z})} \qquad n = 1,2,\dots,N$$

Since we know that $P(c_n = 0|\boldsymbol{z}) + P(c_n = 1|\boldsymbol{z}) = 1$, it is possible to recover the a posteriori probabilities from L_n as follows:

$$\begin{cases} P(c_n = 0) = \frac{1}{1+e^{L_n}} \\ P(c_n = 1) = \frac{e^{L_n}}{1+e^{L_n}} \end{cases}$$

And, finally, we are able to calculate the symbol level a posteriori probabilities from these bit level a posteriori probabilities, as indicated next.

For a Gray-mapped QPSK constellation, the symbols $a_{m_1,k}(m_1 = 1, \dots, 4)$ can be represented as:

$$a_{m_1,k} = c'^{(1)}_{m_1,k} + jc'^{(2)}_{m_1,k}$$

where $c'^{(1)}_{m_1,k}$ and $c'^{(2)}_{m_1,k}$ belong to $\{-1,1\}$ and are associated to the pair of coded bits, that were mapped onto the symbol $a_{m_1,k}$. For QPSK, the symbol $a_{m_1,k}$ can take the following four values:

$$a_1 = -1 - j$$

$$a_2 = -1 + j$$

$$a_3 = +1 + j$$

$$a_4 = +1 - j$$

Consequently, if we have soft decisions about the coded bits $c'^{(1)}_{m_1,k}$ and $c'^{(2)}_{m_1,k}$, we can find the soft decision about $a_{m_1,k}$. Soft decisions about the coded bits can be derived by the BCJR decoder. At the output of the decoder, the following LAPPR $LLR^{(1)}_{m_1,k}$and $LLR^{(2)}_{m_1,k}$are available:

$$LLR^{(i')}_{m_1,k} = \log\left(\frac{P(c'^{(i')}_{m_1,k} = 1|\mathbf{z},\hat{\varphi})}{P(c'^{(i')}_{m_1,k} = -1|\mathbf{z},\hat{\varphi})}\right) for\ i' = 1,2$$

where $P\left(c'^{(i')}_{m_1,k} = 1\middle|\mathbf{z},\hat{\varphi}\right) + P\left(c'^{(i')}_{m_1,k} = -1\middle|\mathbf{z},\hat{\varphi}\right) = 1$. Then, we have

$$\begin{cases} P(c'^{(i')}_{m_1,k} = -1|\mathbf{z},\hat{\varphi}) = \dfrac{1}{1+e^{LLR^{(i')}_{m_1,k}}} \\ P(c'^{(i')}_{m_1,k} = 1|\mathbf{z},\hat{\varphi}) = \dfrac{e^{LLR^{(i')}_{m_1,k}}}{1+e^{LLR^{(i')}_{m_1,k}}} \end{cases}$$

Then, the symbol a posteriori probabilities will be:

$$P(a_{1,k}) = P\left(c'^{(1)}_{1,k} = -1\middle|\mathbf{z},\hat{\varphi}\right) \cdot P(c'^{(2)}_{1,k} = -1|\mathbf{z},\hat{\varphi})$$

$$P\left(a_{2,k}\right) = P\left(c'^{(1)}_{2,k} = -1\middle|\mathbf{z},\hat{\varphi}\right) \cdot P(c'^{(2)}_{2,k} = 1|\mathbf{z},\hat{\varphi})$$

$$P\left(a_{3,k}\right) = P(c'^{(1)}_{3,k} = 1|\mathbf{z},\hat{\varphi}) \cdot P(c'^{(2)}_{3,k} = 1|\mathbf{z},\hat{\varphi})$$

$$P(a_{4,k}) = P(c'^{(1)}_{4,k} = 1|\mathbf{z},\hat{\varphi}) \cdot P(c'^{(2)}_{4,k} = -1|\mathbf{z},\hat{\varphi})$$

From this soft information, the symbol a_k will be estimated as follows:

$$\hat{a}_k = \sum_{m_1=1}^{M=4} a_{m_{1,k}} . P(a_{m_1,k}) \qquad k = 1,2,\dots,K \tag{4.13}$$

It is seen that according to the formula, the estimated symbol is the weighted average of all the possible symbols values with $P(a_{m_{1,k}})$s being the weights. In this method, the soft detected symbol does not necessarily coincide with a symbol of the constellation. In fact, when the reliability is high, it is close to a constellation point whereas it approaches zero when the decision is not reliable.

At the last step, the symbol estimator is used to get the logarithm of likelihood function for each initial phase $\varphi_0^{(l)}$ $(l = 1,\dots,n_s)$:

$$LLF\left(\varphi_0^{(l)}\right) = \Re\left(\sum_{k=1}^{K} \hat{a}_k{}^* \cdot z_k(v,\tau) e^{-j\varphi_0^{(l)}}\right) \tag{4.14}$$

in which $\hat{a}_k$ is obtained as explained before with $\widehat{\varphi} = \varphi_0^{(l)}$.

Now, the initial phase estimator φ_0 will be the one for which the LLF is maximum. It is clear that if the number of points (n_s) increases, it is more possible to find a phase near the true phase value. In this way, it is more probable to get the zero phase error and to reach the global maximum of the likelihood function.

Now that we have computed φ_0, it is time to go through with the recursive structure of algorithm (1). Consequently, the maximization step formula (4.11) will be simplified to the following form:

$$\hat{\varphi}^{(i)} = arg\left(\sum_{k=1}^{K} \hat{a}_k{}^{(i-1)^*} \cdot z_k(v,\tau)\right) \tag{4.15}$$

where $\hat{a}_k{}^{(i)}$ is the symbol soft-decision computed at the i-th EM iteration.

It is notable that the LAPPR used in the i-th iteration, depends on the phase estimate obtained in the previous iteration by the symbol estimation.

4.3. <u>Assessment of the Solution</u>

Ideally, we would like to design communication systems for which R (data rate) is as large as possible while PER or BER is as small as possible at the same time.

In addition, when we estimate an unknown parameter, a natural question arising is what the ultimate accuracy of the estimator is. Usually, the quality of a signal parameter estimator is measured in terms of its bias and its variance. In order to define these terms, suppose that there is a sequence of observations $\boldsymbol{z} = (z_1, z_2, z_3, \dots, z_K)$ with the probability density function (PDF) $P(\boldsymbol{z}|\varphi)$, from which an estimator of a parameter φ is extracted. The bias of an estimator, say $\widehat{\varphi}(\boldsymbol{z})$, is defined as

$$bias = E[\hat{\varphi}(\boldsymbol{z})] - \varphi \tag{4.16}$$

where φ is the true value of the parameter. The estimator is unbiased when $E[\hat{\varphi}(\boldsymbol{z})] = \varphi$.

The variance of the estimator $\hat{\varphi}(\boldsymbol{z})$ is defined as

$$\sigma_{\hat{\varphi}}^2 = E\{[\hat{\varphi}(\boldsymbol{z})]^2\} - \{E[\hat{\varphi}(\boldsymbol{z})]\}^2 \tag{4.17}$$

Generally, the variance computation may be difficult. Therefore, another alternative called Cramer-Rao lower bound (CRLB) is used to assess the best quality that any signal parameter unbiased estimator can achieve.

Based on the aforementioned explanation, we will explain CRLB and BER in more detail as two assessment criteria.

4.3.1. Cramer-Rao Lower Bound

The Cramer-Rao lower bound is a bound on the variance of any unbiased estimator. At best, if an estimator attains the bound for all values of the unknown parameter, the estimator is the minimum variance unbiased estimator (MVU) and is said to be efficient in the sense that it uses effectively the available data. At worst, it provides a benchmark against which we can compare the performance of any unbiased estimator. Furthermore, it alerts us to physical impossibility of finding an unbiased estimator whose variance is less than the bound.

Since all the information is embodied in the observed data and the underlying PDF, the estimation accuracy depends directly on the PDF. When the PDF is viewed as a function of the unknown parameter ($\boldsymbol{z}$ is fixed), it is termed likelihood function. Intuitively, the sharpness of likelihood function determines how accurately we can estimate the unknown parameter. The sharpness is effectively measured by the negative of the second derivative of the logarithm of likelihood function at its peak. This is the curvature of log-likelihood function. Since in general, the second derivative depends on the random observation vector $\boldsymbol{z}$, it is necessary to compute

$$-E\left\{\frac{\partial^2}{\partial \varphi^2} lnP(\boldsymbol{z}|\varphi)\right\} \tag{4.18}$$

which measures the average curvature of log-likelihood function. The expectation is taken with respect to $P(\boldsymbol{z}|\varphi)$, resulting in a function of φ only. $P(\boldsymbol{z}|\varphi)$ is determined by

$$P(\boldsymbol{z}|\varphi) = \int_{-\infty}^{\infty} P(\boldsymbol{z}|\boldsymbol{a},\varphi)P(\boldsymbol{a})d\boldsymbol{a} \tag{4.19}$$

where $\boldsymbol{a}$ is a random vector having a known PDF $P(\boldsymbol{a})$ which does not depend on φ and consists of all the other nuisance parameters, including the data.

To compute CRLB, first it is assumed that the PDF satisfies the "regularity" condition (4.20).

$$E\left[\frac{\partial}{\partial \varphi} lnP(\boldsymbol{z}|\varphi)\right] = 0 \quad for\ all\ \varphi \tag{4.20}$$

where the expectation is taken with respect to $P(\boldsymbol{z}|\varphi)$. Then, the variance of any unbiased estimator $\hat{\varphi}$ must satisfy

$$\sigma_{\hat{\varphi}}^2 \geq \frac{1}{-E\left\{\frac{\partial^2}{\partial \varphi^2} lnP(\boldsymbol{z}|\varphi)\right\}} = \frac{1}{E_{\boldsymbol{z}}\left\{\left[\frac{\partial}{\partial \varphi} lnP(\boldsymbol{z}|\varphi)\right]^2\right\}} = CRLB(\varphi) \tag{4.21}$$

where the derivative is evaluated at the true value of φ and the expectation is taken with respect to $P(\boldsymbol{z}|\varphi)$. It is clear that the lager the quantity in (4.18), the smaller the variance of the estimator. Furthermore, an unbiased estimator may be found that attains the bound for all φ if and only if

$$\frac{\partial}{\partial\varphi} lnP(\boldsymbol{z}|\varphi) = I(\varphi)(y(\boldsymbol{z}) - \varphi) \qquad (4.22)$$

for some functions y and I. That estimator, which is the MVU estimator, is $\hat{\varphi} = y(\boldsymbol{z})$, and the minimum variance is ${}^{1}/_{I(\varphi)}$ where $I(\varphi)$ is called the Fisher information matrix (FIM), which is related with CRLB by

$$CRLB(\varphi) = I^{-1}(\varphi) \qquad (4.23)$$

Intuitively, the more information, the lower the bound.

4.3.1.1. Data-Aided CRLB and Modified CRLB

Unfortunately, in most practical cases, the computation of (4.21) is impossible because of either the integration in (4.19) is not able to be carried out analytically or the expectation in (4.21) poses insuperable obstacles. Therefore, it is recommended to resort to another bound which is a lower bound to the variance of any parameter estimator $\hat{\varphi}(\boldsymbol{z})$, called the Modified CRLB and calculated by (4.24). It is useful when besides the unknown parameter, the observed data also depends on other unwanted parameters.

$$MCRLB(\varphi) = \frac{1}{E_{\boldsymbol{z},\boldsymbol{a}}\left\{\left[\frac{\partial}{\partial\varphi} lnP(\boldsymbol{z}|\boldsymbol{a},\varphi)\right]^2\right\}} \qquad (4.24)$$

We know from equality (3.9) that the output of matched filter, which has a complex envelope, is: $z(t) = \sum_k a_k e^{i\varphi}\, g(t-kT) + n(t)$

where T is the symbol spacing, the data symbols $\{a_k\}$ are zero-mean and independent random variables, $g(t)$ is the real-valued signalling pulse and $n(t)$ represents complex-valued additive white Gaussian noise with two-side power spectral density $2N_0$. The only unknown parameter is the carrier phase error φ which is assumed to be deterministic and independent of the data. An exact representation of the observed waveform $z(t)$ would require infinite-dimensional vector spaces but, for the time being, we assume that a finite-dimensional vector $\boldsymbol{z}$ can be found to represent $z(t)$ with adequate accuracy within the observation interval. Therefore, assume that $\boldsymbol{z}$ has K components so that we have: $z_k = a_k e^{i\varphi} + n_k \quad k = 1,2,\dots,K$

where the $\{n_k\}$ complex samples are independent and equally distributed Gaussian random variables with zero-mean and variance σ_N^2 for both the real and imaginary components of the complex noise n_k.

Now, we are able to define $P(\boldsymbol{z}|\boldsymbol{a},\varphi)$ as the following:

$$P(\boldsymbol{z}|\boldsymbol{a},\varphi) = \frac{1}{(2\pi\sigma_N^2)^{\frac{K}{2}}} exp\left\{-\frac{1}{2\sigma_N^2}\sum_{k=1}^{K}\left[z_k - a_k e^{i\varphi}\right]^2\right\} \qquad (4.25)$$

If the number of dimensions of $\boldsymbol{z}$ tends to infinity, to use (4.24), $P(\boldsymbol{z}|\boldsymbol{a},\varphi)$ is replaced by another likelihood function $\Lambda(\varphi,\boldsymbol{a})$ which is defined as:

$$\Lambda(\varphi,\boldsymbol{a}) = exp\left[-\frac{1}{2N_0}\int_T \left|z(t)-a(t)e^{i\varphi}\right|^2 dt\right] \tag{4.26}$$

Note that the expectation over $\boldsymbol{z}$ in (4.24) can be replaced by the expectation over the noise process $n(t)$.

$$MCRLB(\varphi) = \frac{1}{E_{n,\boldsymbol{a}}\left\{\left[\frac{\partial}{\partial\varphi}ln\Lambda(\varphi,\boldsymbol{a})\right]^2\right\}} \tag{4.27}$$

Then, if we solved this expectation according to [30], we have

$$\sigma_{\hat{\varphi}}^2 \geq MCRLB(\varphi) = \frac{1}{2K\rho} \tag{4.28}$$

where ρ is the energy per transmitted symbol over the noise spectral density N_0 at the receiver input, K is the number of transmitted symbols.

Now, assume that $\boldsymbol{a}$ is a known deterministic quantity, then, $P(\boldsymbol{a})$ can be expressed using Dirac's deltas as follows:

$$P(\boldsymbol{a}) = \prod_{k=0}^{K_{pre}-1}\delta(a_k-\breve{a}_k) \tag{4.29}$$

where $\breve{a}_k$ is the value of the preamble symbol at time k. As a result, $P(\boldsymbol{z}|\varphi) = P(\boldsymbol{z}|\breve{\boldsymbol{a}},\varphi)$.

This DA assumption enables the derivation of a closed-form expression for the DA CRLB which is:

$$CRLB_{DA}(\varphi) = \frac{1}{2K_{pre}\rho} \tag{4.30}$$

where K_{pre} is the number of preamble symbols.

In computing MCRLB, the assumption is that essentially no information is available on the unwanted parameters. In this hypothesis, the MCRLB is looser than the DA CRLB in general ($CRLB_{DA}(\varphi) \geq MCRLB(\varphi)$) because the other parameters and data are known in the DA case. In this way, the calculation of the DA CRLB is simpler. The MCRLB equals to the DA CRLB if all the data symbols were known to the receiver, i.e., for large SNR.

Since usually no estimator can provide a lower variance than that obtained by the exact CRLB, it may be asked whether it is possible to attain the MCRLB. According to [30], if enough information is provided, the MCRLB can actually be attained. For example, if it is assumed that frequency offset, timing and the data are available. As a consequence, the estimation process relies on a perfectly known transmitted sequence.

It is known that computing the CRLB for NDA and CA is more complex than for DA due to the summation over $\boldsymbol{a}$. Therefore, the modified CRLB and DA CRLB are the first options to be used.

4.3.1.2. Non-Data-Aided CRLB

It was mentioned before that, unfortunately, the evaluation of the exact CRLB is mathematically quite difficult when the observed signal contains random discrete data and random noise in addition to the parameter to be estimated. And also it is stated that the first option was to compute MCRLB (or DA CRLB). Another option is to apply a blind estimation process that only assumes statistical knowledge on the random transmitted symbols. This leads to another bound called non-data-aided (NDA) CRLB. NDA CRLB is the exact CRLB if the information symbols are unknown and are viewed as nuisance parameters. It is also known that this method has very poor estimation performance, especially in the presence of short data record in the low SNR region [31].

In this thesis, the observations are the noisy linearly modulated waveforms that are a function of deterministic parameters like the carrier phase as well as the data symbol sequence. In order to calculate NDA CRLB, the assumption is that there is no a priori information about an unknown parameter φ, i.e. $P(\varphi)$ is uniform and the transmitted symbols are usually assumed to be equally likely, i.e.

$$\Pr\left[a_k = a_{m_1}\right] = \frac{1}{|\mathcal{A}|} = \frac{1}{M} \qquad for\ \ k = k_0, k_0+1, \dots, k_0+K-1\ \ and\ m_1 = 1, \dots, M \tag{4.31}$$

where K is the total number of recorded data and k_0 refers to the time instant of the first observed sample.

Since $\{a_k\}$ is a sequence of independent identically distributed (i.i.d) data symbols, we have

$$P(\boldsymbol{a}) = \frac{1}{|\mathcal{A}|^K} \ \ \forall \boldsymbol{a} \in \mathcal{A}^K \tag{4.32}$$

We know that the observation vector $\boldsymbol{z}$ which is obtained at the output of the matched filter is:

$$z_k = a_k e^{j\varphi} + n_k \quad for\ k = k_0 + \cdots + k_0 + K - 1 \tag{4.33}$$

where the unknown parameter φ is deterministic. The sequence $\{n_k\}$ consists of i.i.d zero-mean complex Gaussian noise random variables with variance$\sigma_N^2 \triangleq E|n_k|^2$. The symbols a_k are assumed to be independent from n_k. If the sequence of symbols can be deemed independent, z_ks are independently identically distributed according to the following mixture of Gaussian distribution.

$$P(z_k|\varphi) = \frac{1}{M 2\pi\sigma_N^2} \sum_{m_1=1}^{M} exp\left(-\frac{\left|z_k - a_{m_1}e^{j\varphi}\right|^2}{2\sigma_N^2}\right) \tag{4.34}$$

Using the independence of the random variables z_k, the FIM is given by:

$$I(\varphi) = -\sum_{k=k_0}^{k_0+K-1} E\left(\frac{\partial^2 lnP(z_k|\varphi)}{\partial\varphi^2}\right) \tag{4.35}$$

Then, according to [31], the CRLB for QPSK modulation is:

$$CRLB_{NDA}(\varphi) = MCRLB(\varphi)\left(\frac{1}{1-(1+\rho)f_1\left(\frac{\rho}{2}\right)}\right) \tag{4.36}$$

where f_1 is the following decreasing function of ρ:

$$f_1(\rho) \overset{\text{def}}{=} \frac{2e^{-\rho}}{\sqrt{2\pi}} \int_0^{+\infty} \frac{e^{\frac{-u^2}{2}}}{cosh\left(u\sqrt{2\rho}\right)} du \tag{4.37}$$

Note that the hypothesis of independence of $\Re(a_k)$ and $\Im(a_k)$ is taken into account for QPSK.

4.3.1.3. Code-Aided CRLB

When there are unknown and random transmitted symbols besides the synchronization parameters, another option is CA CRLB. The only needed quantity to evaluate this newly CRLB is the extrinsic information outputted by the MAP SISO decoder.

Note that having the APPs at hand, it is possible to evaluate all the quantities involved in the expressions of the CRLB.

The received samples at the output of the matched filter are modelled like (4.33) where K is the total number of recorded data and k_0 refers to the time instant of the first observed sample. The unknown parameter φ is deterministic. The noise components, $\{n_k\}$, are modeled as zero mean complex Gaussian random variables with independent real and imaginary parts, each of variance σ_N^2. Without loss of generality, it is also assumed that the energy of the transmitted symbols is normalized to one (i.e.$E\{|a_k|^2\} = 1$). So that the average SNR of the system is given by $\rho = \frac{E\{|a_k|^2\}}{2\sigma_N^2} = \frac{1}{2\sigma_N^2}$ (the channel gain is considered here equal to one).

Based on the definition of CRLB, the first step is to find an explicit expression for the LLF or, equivalently, the PDF $P(\boldsymbol{z}|\varphi)$. Since the coded bits are assumed to be statistically independent (assuming a long interleaver), the transmitted symbols, which are simply some real-valued representations for different blocks of these bits, can also be considered independent, leading to:

$$P(\boldsymbol{z}|\varphi) = \prod_{k=k_0}^{k_0+K-1} P(z_k|\,\varphi) \tag{4.38}$$

Consequently, the LLF is:

$$ln\, P(\boldsymbol{z}|\varphi) = \sum_{k=k_0}^{k_0+K-1} ln\, P(z_k|\varphi) \tag{4.39}$$

The PDF of each received sample z_k, which is parameterized by the unknown parameter vector φ , is given by:

$$\begin{aligned} P(z_k|\varphi) &= \sum_{a_{m_1}\in\mathcal{A}} \Pr(a_k = a_{m_1}) P\left(z_k \middle| a_k = a_{m_1}, \varphi\right) \\ &= \frac{1}{2\pi\sigma_N^2} \sum_{a_{m_1}\in\mathcal{A}} \Pr(a_k = a_{m_1}) exp\left\{-\frac{\left|z_k - a_{m_1} e^{j\varphi}\right|^2}{2\sigma_N^2}\right\} \end{aligned} \tag{4.40}$$

Now, the APPs, $\Pr(a_k = a_{m_1})$, of the transmitted symbols involved in (4.33) must be found. In CA estimation, the APPs of the symbols given by the soft decoder must be used to enhance the estimation performance.

In practice, the information about the LAPPRs of the conveyed bits is acquired from the output of the MAP SISO decoder at its convergence (here, at the convergence of the BCJR algorithm).

In case of QPSK or bit-interleaved QAM constellations, the coded bits that are embedded in each channel symbol a_k are statistically independent assuming that the receiver carrier phase is synchronized. Therefore, the symbol APPs can be factorized in terms of the APPs of the coded bits.

$$\Pr(a_k = a_{m_1}) = \Pr\left[b_1^k = \bar{b}_1^{m_1}, b_2^k = \bar{b}_2^{m_1}, \dots, b_{\log_2 M}^k = \bar{b}_{\log_2 M}^{m_1}\right] = \prod_{l_1=1}^{\log_2 M} \Pr\left[b_{l_1}^k = \bar{b}_{l_1}^{m_1}\right] \quad (4.41)$$

It is also defined that the LAPPR of the l_1-th coded bit, $b_{l_1}^k$, conveyed by the transmission of the symbol, a_k, as follows:

$$LLR_{l_1}(k) \triangleq \ln\left(\frac{\Pr\left[b_{l_1}^k = 1\right]}{\Pr\left[b_{l_1}^k = 0\right]}\right) \quad (4.42)$$

Using $(4.43)$and the fact that $\Pr\left[b_{l_1}^k = 1\right] + \Pr\left[b_{l_1}^k = 0\right] = 1,$ it can be shown that:

$\Pr\left[b_{l_1}^k = 1\right] = \frac{e^{LLR_{l_1}(k)}}{1+e^{LLR_{l_1}(k)}}$ and $\Pr\left[b_{l_1}^k = 0\right] = \frac{1}{1+e^{LLR_{l_1}(k)}}$. So for every a_{m_1} in $\mathcal{A}$, if $a_k = a_{m_1}$, then, we can write a generic expression:

$$\Pr\left[b_{l_1}^k = \bar{b}_{l_1}^{m_1}\right] = \frac{1}{2\cosh(LLR_{l_1}(k)/2)} e^{\left(\bar{b}_{l_1}^{m_1}-1\right)\frac{LLR_{l_1}(k)}{2}}$$

By solving the above equations, the probabilities $\Pr(a_k = a_{m_1})$ can be computed. Then, if the methods in [24] are followed, the code-aided CRLB is calculated by

$$CRLB_{CA}(\varphi) = \left(\sum\nolimits_{k=k_0}^{k_0+K-1} \Omega_{p,k}(\rho)\right)^{-1} \quad (4.43)$$

where

$$E\left\{\frac{\partial^2 lnP(z|\varphi)}{\partial\varphi^2}\right\} = \sum\nolimits_{k=k_0}^{k_0+K-1} \Omega_{p,k}(\rho) \quad (4.44)$$

To know more about it, you can refer to [24].

Since the values of all the obtained LAPPRs depend on the underlying noise realization, the new analytical CA CRLB are averaged over a small number of noise realization at every SNR point in order to smooth the curve. Typically, they were smoothed over 20 realizations.

Moreover, the obtained CA CRLB does not depend on the true value of the unknown phase shift. Hence, it holds the same value for all the possible values of synchronization parameter. Thus, in our simulation, we evaluate the CA CRLB for phase error equals to zero (perfect synchronization). This provides the most reliable values for LAPPRs. Note that, in the presence of non-zero values for φ, the derotated samples, $z_k e^{-j\varphi}$, should be fed to the decoder. Anyway, the CRLB is always evaluated at the true value of the unknown parameter so that it predicts the performance of turbo synchronization when, after convergence, it is tracking correctly the carrier phase. Like turbo synchronizers, the CA CRLB uses the soft outputs computed by the SISO decoder.

4.3.2. Bit Error Probability

The bit-error-rate performance of a receiver is a figure of merit that allows different designs to be compared in a fair manner. Bit-error-rate performance is usually depicted on a two dimensional graph. The abscissa is the normalized signal-to-noise (SNR) which is expressed as E_b/N_0: the energy-per-bit divided by the one-sided power spectral density of the noise, and is expressed in decibels (dB). The ordinate is the bit-error-rate, which is a dimensionless quantity and is usually expressed in powers of ten.

Since the ultimate goal of the receiver is to minimize the bit error rate (or the packet error rate), these two metrics should be considered as the criterion for deriving the synchronization algorithms. This solution is generally time-consuming, especially for long packet transmission, large constraint length (convolutional codes) or, in general, powerful codes.

5. Simulation Results

In this section, the simulation results will be illustrated to examine the performance of turbo synchronization for short packets without preambles in terms of

I. The accuracy, i.e., the variance it can achieve.
II. The acquisition range, i.e., the range of phase error for which it properly recover synchronization.
III. The reliability, i.e., the BER performance and, the PER performance.

For simplicity, a Gray-mapped QPSK convolutionally coded transmission has been considered. The termination method of the convolutional code is tail-biting. Two square-root-raised-cosine filters with rolloff factor 0.3 have been implemented as baseband filters: one at the transmitter side and the other at the receiver side, both satisfying the Nyquist criterion. The filter is truncated to 20 symbols and each symbol period contains 4 samples. The considered noise is a complex additive white Gaussian noise with random variance σ_N^2 for both real and imaginary parts. To synchronize the system, the turbo synchronization is implemented by means of the EM algorithm. The number of turbo iterations (Itr) is 10. The phase error is considered as an unknown random variable. We consider perfect time and frequency synchronization. The decoder is the BCJR decoder which is implemented by means of Max-Log-MAP algorithm plus a correction term (Eq. 3. 36) (Max* algorithm in the Matlab Communications System Toolbox).

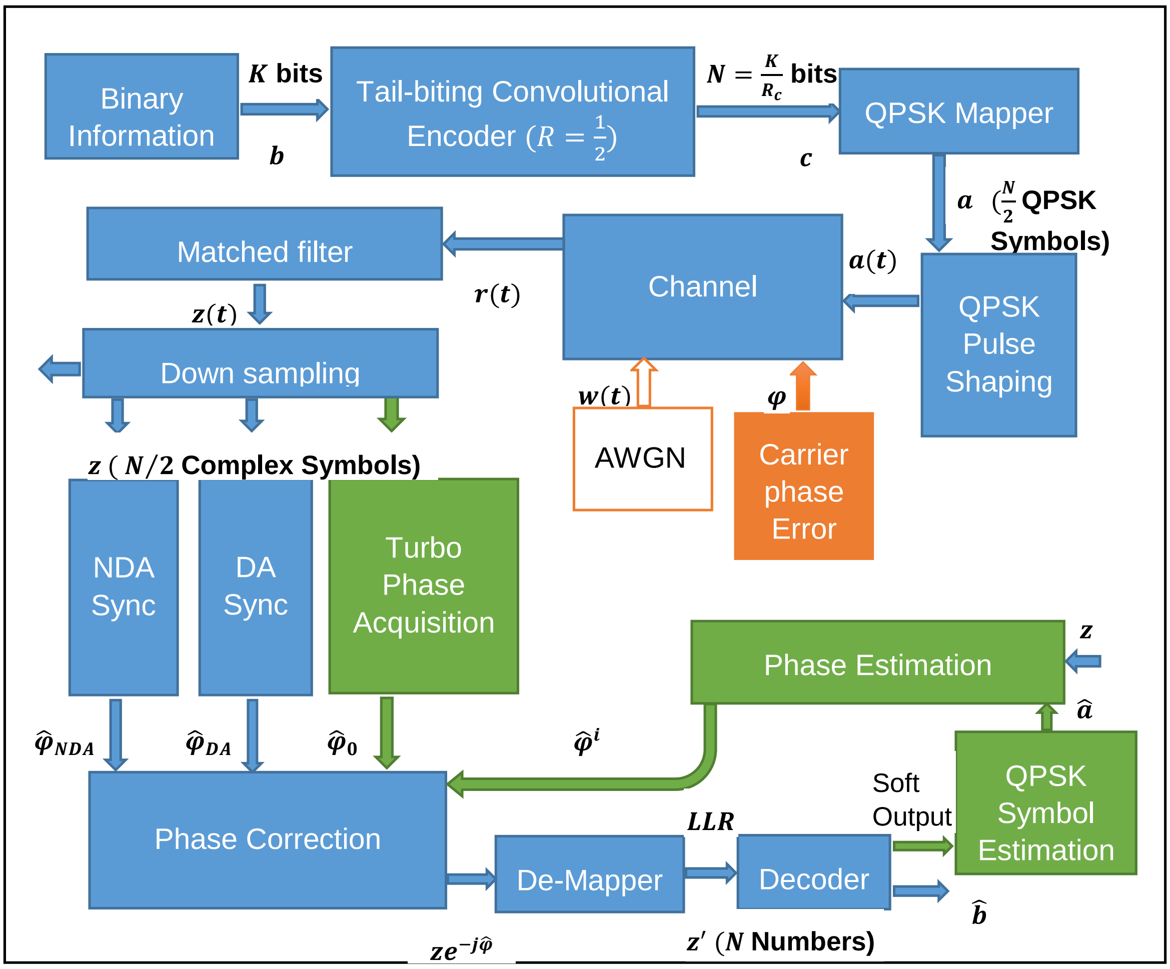


Figure 5. 1. The implemented system schematic

5.1. Accuracy Evaluation

In this section, we will assess the ultimate accuracy of carrier phase estimators by comparing their variance to the related CRLB.

To do that, we plot four types of CRLB: MCRB, DA CRLB, NDA CRLB and CA CRLB. In addition, the variance for DA, CA and NDA modes are plotted. We consider two different packet lengths for the transmitted frames: the short length of 64 QPSK symbols and the medium length of 256 QPSK symbols.

It is mentioned that the error correction capability of the convolutional codes depends on its constraint length, code rate and the selected generator polynomials. It is also known that the traceback depth of the decoder influences also the decoding reliability and delay. In general, the traceback depth must be less than or equal to the number of input symbols. Hereinafter, the generally used code rate is 1/2, and, therefore, the traceback should be greater than, approximately, $5(L-1)$ information bits (Matlab Communications System Toolbox). Because the packet size is 64 symbols, the constraint length should be less than 13 to decode packets optimally. As a result, we consider short constraint length 3, medium one 8, and the largest acceptable one 12 with generator polynomials $[5\ 7]_3$, $[247\ 371]_8$ and $[4335\ \ 5723]_{12}$, respectively. Furthermore, we did simulations for codes rates 1/4 and 1/8, with generator polynomials $[235\ 275\ 313\ 357]_8$ and $[275\ 275\ 253\ 371\ 331\ 235\ 313\ 357]_8$, respectively. The turbo synchronizer is initialized by the initialization method explained in section 4.2.1.1.1 and considering $n_s = 8$. The CA CRLB curves are averaged over 20 realizations. The estimator variances are obtained from 1000 independent realizations.

Simulation are carried out considering that no termination bits are used to return the encoder to a known state. (This corresponds to the "truncated mode of the Matlab BCJR decoding function in the Communication Toolbox.)

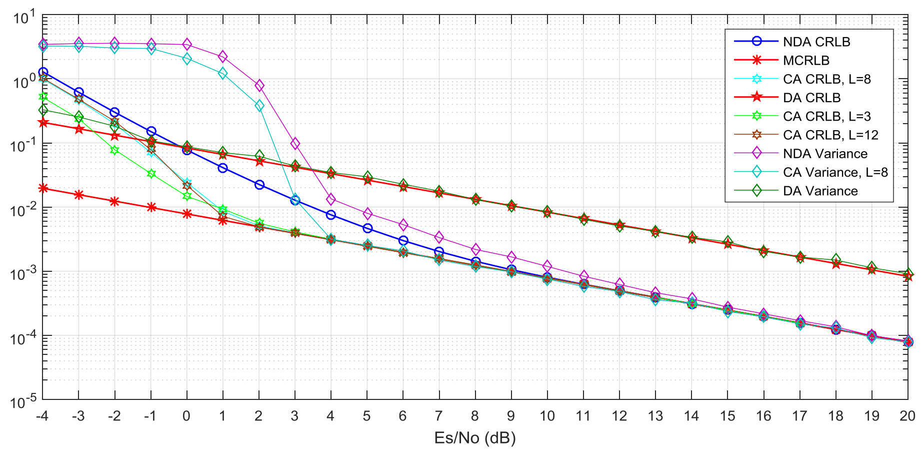


Figure 5. 2. The Variance of phase offset estimation for NDA, DA and CA scenarios for $L = 8$ and the MCRLB, NDA CRLB, DA CRLB for $K_{pre} = 6$ symbols and CA CRLB for different constraint lenghts 3,8 and 12. The packet length is 64 symbols.

Figure 5.2 depicts the variance of carrier phase estimation achieved by different synchronizers: DA with the preamble size of 6 symbols (approximately 10% of the packet size), NDA synchronizer and CA with different constraint lengths 3, 8 and 12 versus different SNR values. It is seen that the variance of each synchronizer attains its related CRLB when the SNR value is high enough.

The figure shows that the MCRLB is looser than other bounds at low SNR. At high SNR values, the NDA and CA bounds attain the MCRLB. This result is compatible with [32]. According to the formulas (4.28) and(4.30), the MCRLB and the DA CRLB are identical except for a constant factor equal to $6/64$, which is the ratio of observation lengths considered in each case. The DA estimator variance attains the DA CRLB at SNR values greater than -1dB.

We consider the conventional NDA Viterbi & Viterbi carrier phase synchronizer for the NDA case. As it is shown in the figure, the phase offset variance approaches the related CRLB when the value of the signal-to-noise ratio is equal to 4dB and touches the CRLB when the signal-to-noise ratio value is equal to 15dB. It is also clear in the figure that the NDA CRLB decreases exponentially with increasing SNR values and reaches to MCRLB at the SNR value of 9dB.

The CA CRLB lies between the MCRLB and the NDA CRLB in the low-to-medium SNR region. This highlights the performance improvement that can be achieved by a coded system over the uncoded one and also shows that the soft information provided by the BCJR decoder which is exploited during estimation causes an enhancement in the synchronization accuracy. In addition, the CA CRLB decreases rapidly and reach the MCRLB, which is the ideal bound that would be obtained if all the transmitted symbols were perfectly known to the receiver, beyond a relatively small SNR threshold. Surprisingly, it improves at low SNR when the constraint length decreases form 8 to 3 and it is nearly the same for constraint lengths 8 and 12 for any SNR. The CA variance is the variance of the phase offset error obtained by the EM turbo synchronizer after convergence. It follows the same shape as the NDA variance curve. The only difference is that it attains MCRLB and CA CRLB at a lower SNR value, which is approximately 4dB. This is a direct consequence of the fact that the turbo synchronization is an iterative implementation of ML CA estimator and the Viterbi & Viterbi synchronizer is only an approximation at low SNR of the ML NDA estimator.

Therefore, the reason for the gap between the performance achieved by the turbo synchronizer and the conventional Viterbi &Viterbi synchronizer is twofold: $i)$ the latter operates in NDA mode and drops important statistical information about the transmitted sequences, and $ii)$ the latter does not implement the actual ML solution and is based on an approximation.

Finally, it is obvious that the turbo synchronizer variance outperforms the DA synchronizer variance for SNR values which are greater than 2.5dB.

It is known that the ML estimator is asymptotically efficient, i.e., it reaches the CRLB for sufficiently long frame lengths. For the turbo synchronizer, if it is properly initialized, it is expected to perform in this way. Figure 5.3 illustrates that with an increase in the packet length from 64 symbols in figure 5.2 to 250 symbols in figure 5.3, the CA CRLB coincides with the MCRLB at SNR values greater than 2dB. In addition, the NDA variance approaches its related CRLB and the CA variance attains the CA CRLB at SNR value of 2dB compared to 4dB in figure 5.2 where the packet length was shorter.

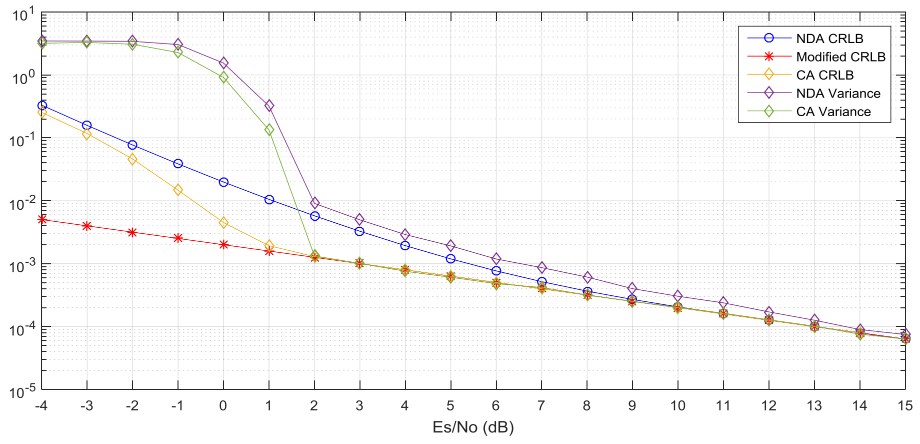


Figure 5. 3. The variance of phase offset estimators for NDA and CA scenarios and the MCRLB, NDA and CA CRLB for $L = 8$ and the packet length of 250 symbols.

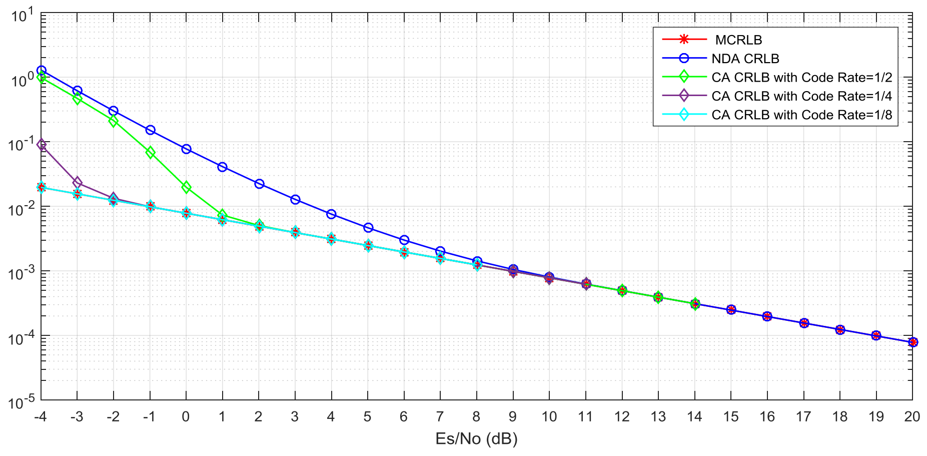


Figure 5. 4. The CA CRLB for different code rates: 1/2, 1/4 and 1/8, $L = 8$ and the packet length of 64 symbols.

Figure 5.4 shows the effect of the coding rate on the turbo synchronization performance for different SNR values and compares these different CA CRLBs with the MCRLB and NDA CRLB. It has been demonstrated that the CA CRLB improves by decreasing the overall coding rate.

While the curve of the code rate 1/2 approaches NDA CRLB when the SNR value decreases, the curve of the code rate 1/8 attains the MCRLB for all the simulated SNR values and the one for code rate 1/4 for any SNR greater than -2dB . This is due to the fact that more redundancy is provided by the encoder when the code rate is reduced. Consequently, the decoder is more likely able to correctly detect the transmitted bits and enhance the synchronizer performance. This effect is more apparent in the low SNR regime where more redundancy implies more reliable decoding and therefore better synchronization.

Eventually, we conclude that the implementation of the turbo synchronizer is beneficial in terms of the phase offset estimation accuracy, especially when the code rate is small enough or the packet length is not too short. (We will find the packet length threshold in the next section)

5.2. Acquisition range

In this section, the acquisition range, which is the range of phase errors for which the receiver is able to properly correct the carrier synchronization error, will be evaluated as another important feature of any synchronizer.

To do that, in the following 4 figures, we plot: 1) the average of the logarithm of likelihood function (LLF) versus different phase error values; 2) the ratio of the LLF of the 180° phase error to the LLF of the zero-phase error for different constraint lengths; 3) the probability of wrong acquisition versus different SNR values; and 4) the BER for different initial phase errors. In nearly all simulations, we consider that the packet length is 64 symbols, n_s is equal to 8 and the code rate is equal to 1/2.

It is known that the QPSK modulation has a phase ambiguity of $\pi/2$ radians, which is inherent to any NDA scheme. This ambiguity is insoluble using NDA synchronizers. As a result, the maximum acquisition range, which is achievable by a NDA estimator, is $\pm\pi/4$.

In order to see how the phase acquisition works, it is needed to plot the average of the logarithm of likelihood function (LLF) curve.

Figure 5. 5. The average LLF for different constraint lengths, K=64, code rate 1/2 and EbNo=1dB

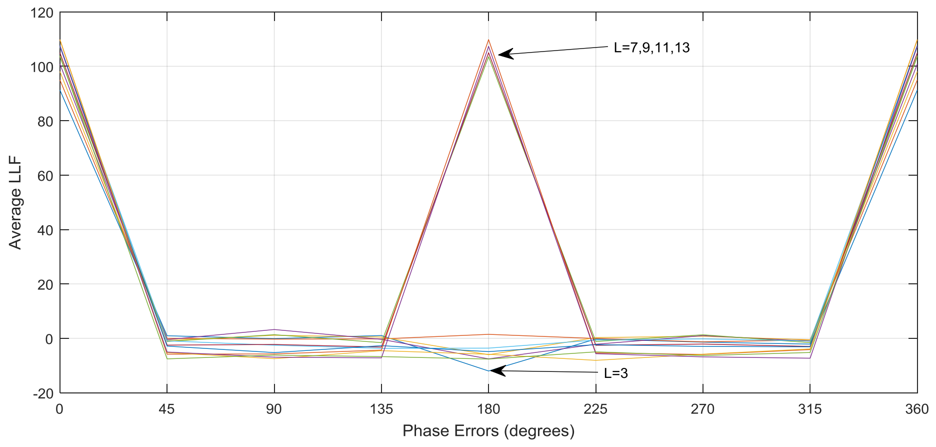


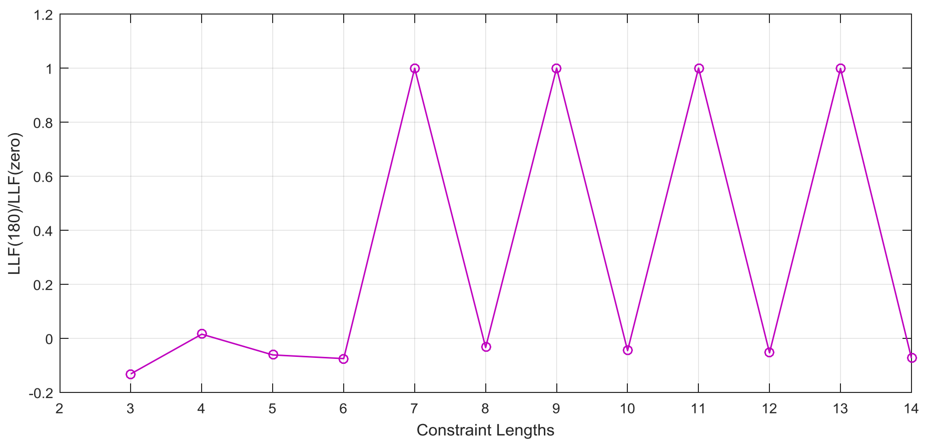
Figure 5.5 illustrates the average of the logarithm of likelihood function for different phase errors and different constraint lengths from 3 to 14 with the generator chosen from table 5.1. It is noticeable that the LLF has a 180^{o} symmetry for odd constraint lengths larger than 7.

To see how great the maximum is at 180^{o} phase offset, we plot the curve for the ratio of the average LLF at 180^{o} to the average LLF at zero degrees for different constraint lengths.

Figure 5.6 depicts that the average LLF has a 180^{o} symmetry for odd constraint lengths larger than 7 because the ratio is exactly one at these constraint lengths. This symmetry implicitly means that the turbo synchronizer will not be able to distinguish between two phase offsets that are π apart. Therefore, the EM turbo synchronizer is unable to properly synchronize the system and the hill-climbing nature of iterative methods on which the EM turbo synchronizer rely will lead to convergence to the wrong maximum when the absolute value of the phase offset is larger than $\pi/2$. The justification of such behaviour is that the turbo synchronizer acquisition range is limited due to the convergence failure. At low SNR, since the symbol decisions are not reliable, the CA turbo synchronizer acts like a NDA synchronizer. Therefore, it can be said that the same as NDA synchronizer, the maximum acceptable phase offset intrinsically limited by the constellation symmetry.

It is noticeable to mention that we have also done the above simulations (which are not attached here) for the convolutional encoder with truncated termination. In that case, there is a great local maximum at 180^{o} , especially for constraint lengths 7, but the average LLF curves always have a global maximum for all constraint lengths.

Figure 5. 6. The average LLF ($\mathbf{180}^{\circ}$)/ average LLF ($\mathbf{0}^{\circ}$) for different constraint lengths with the generator polynomials according to table 5. 1 and EbNo=1dB



Rate ½ Maximum Free Distance Codes				
Constraint Length L	Generators in Octal		d_{free}	Upper Bound on d_{free}
3	5	7	5	5
4	15	17	6	6
5	23	35	7	8
6	53	75	8	8
7	133	171	10	10
8	247	371	10	11
9	561	753	12	12
10	1167	1545	12	13
11	2335	3661	14	14
12	4335	5723	15	15
13	10533	17661	16	16
14	21675	27123	16	17

Sources: Odenwalder (1970) and Larsen (1973).

Table 5. 1. Convolutional code polynomial

In such cases, ambiguity resolution methods should be considered. As a direct consequence, the turbo synchronizer is unable to attain synchronization for phase offsets greater than $\pm\, {}^{\pi}/_{2}$ unless the ambiguity is somehow previously solved.

As an attempt to avoid the 180^{o} ambiguity, it is tried to substitute the generator polynomials in table 5.1 with the ones presented in table 5.2. As it is seen in figures 5.7 and 5.8, the ambiguity is solved for odd constraint lengths at the SNR value of 1dB. As a result, there is no ambiguity for all the simulated constraint lengths because there is a global maximum in the range$[0^{o}\ ,360^{o}]$ at the aforementioned SNR value.

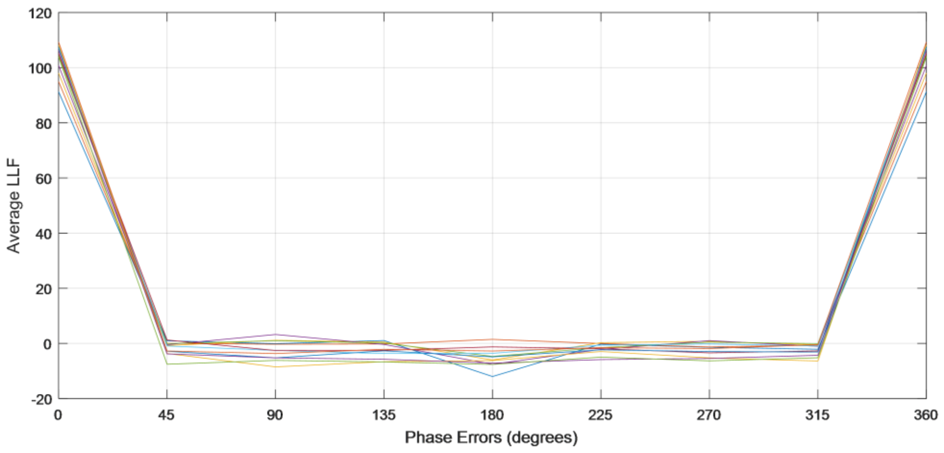


Figure 5. 7. The average LLF for different constraint lengths with the generator polynomials according to table 5. 2 and EbNo=1dB

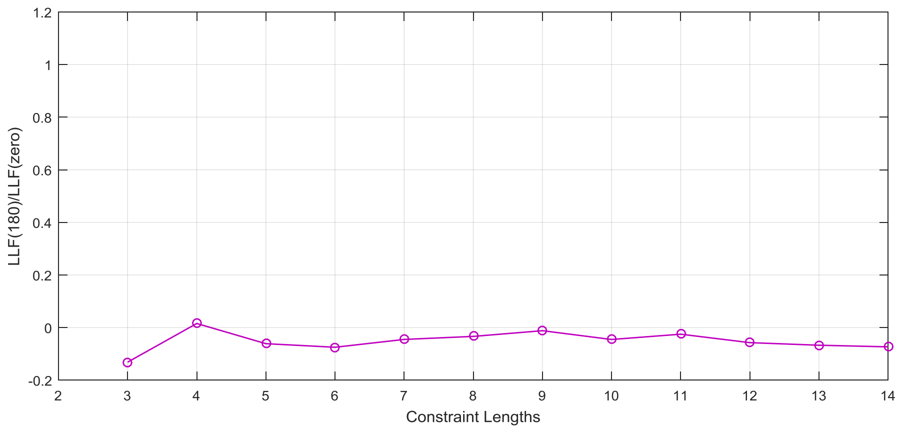


Figure 5. 8. The average LLF ($\mathbf{180}^{\circ}$)/ average LLF ($\mathbf{0}^{\circ}$) for different constraint lengths with the generator polynomials according to table 5. 2 and EbNo=1dB

Surprisingly, when we did the simulations (which are not attached here) for BER and PER, there was not a significant improvement with these new generator polynomials respect to the generators of table 5.1 at high SNR values. As a direct consequence, we plot another curve which shows the impact of the SNR value on the ratio of the average LLF at 180^{o} to the average LLF at zero degree phase error.

Rate ½ Maximum Free Distance Codes				
Constraint Length L	Optimal Generators in Octal		d_{free}	Upper Bound on d_{free}
3[1]	5	7	5	5
4[1]	15	17	6	6
5[1]	23	35	7	8
6[1]	53	75	8	8
7*[3]	133	161	9	-
8[1]	247	371	10	11
9*[2]	515	677	12	-
10[1]	1167	1545	12	13
11*[4]	2335	3461	-	-
12*[2]	5537	6131	14	-
13*[4]	10533	15661	-	-
14[1]	21675	27123	16	17

Sources: [1] Odenwalder (1970) and Larsen (1973) [2] Ref. [29]
[3] Ref. [40] [4] Determined by author

Note those with asterisk are the ones that have been changed.

Table 5. 2. Convolutional code polynomials

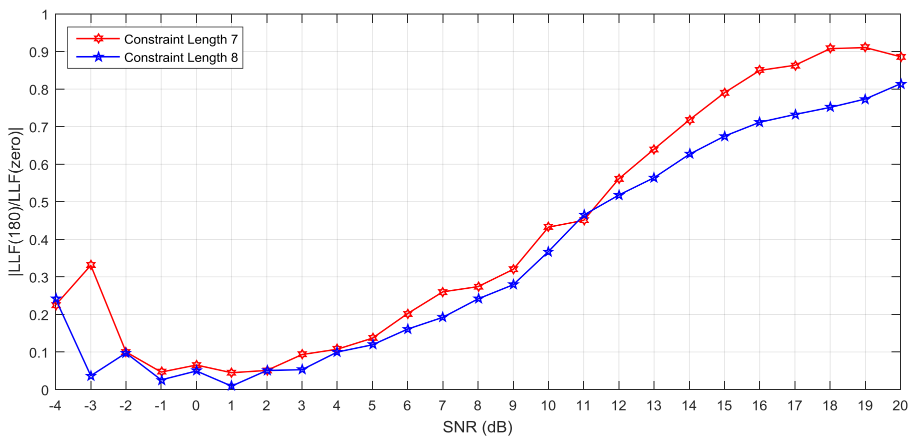


Figure 5. 9. The absolute value of the ratio of the average LLF at 180 phase error to the average LLF at zero phase error for different SNR values, packet length=64, L=7 and 8,ns=8, R=1/2

Figure 5.9 shows the absolute value of the ratio of the average LLF at 180 degrees phase error to the average LLF at zero degree phase error for different SNR values and constraint lengths of 7 and 8 with generator polynomials in table 5.2. It is remarkable that an increase in the SNR value leads to an increment in the ratio. Consequently, although the substitution of the generator polynomials seems beneficial at low SNR, it is not advantageous at high SNR. However this increment is not as dramatic as the ones obtained with the generator polynomials in table 5.1 at low SNR. To conclude, the substitution of the generator polynomials is unable to eliminate the phase ambiguity.

Now that the effect of the code on the average LLF was examined it is time to inspect the impact of the packet length on the average LLF.

Figure 5. 10. The average LLF for different packet lengths with constraint length 8 and EbNo=1dB.

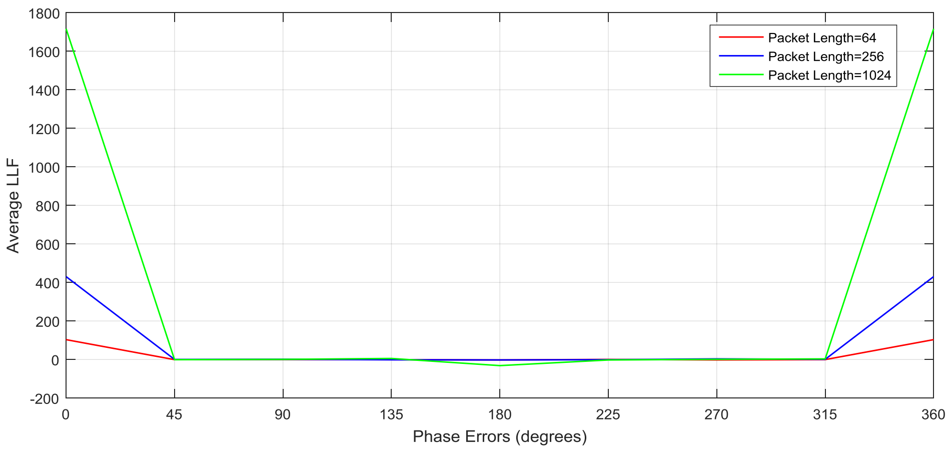


Figure 5.10 shows the impact of the packet length on the average LLF for constraint length of 8 at the SNR value which is equal to 1dB. It is noticed that the maximum of the average LLF increases with increasing of the packet lengths. But it does not affect the phase ambiguity at EbNo of 1dB according to the other simulations which are not attached here. It means that the phase acquisition is not influenced by the packet size at low SNR values.

It is apparent that if we partition the phase interval $[0, 360^{\circ}]$ into more sections, it is more probable to find a point near the global maximum of the LLF. Therefore, the more the sections the more effective the phase acquisition. Therefore, we present figure 5.11, which illustrates this tangible result.

The figure depicts the impact of the SNR value and the number of the sections in the interval $[0, 360^{\circ}]$ on the probability of wrong acquisition. The probability of wrong acquisition counts the number of packets whose phase error is outside of the acquisition range of $\pm\pi/4$ over the total number of the simulated packets.

It is clear that the probability of wrong acquisition decreases as the SNR value increases because of higher reliability of likelihood function at high SNR. Also, it proves the aforementioned idea about the number of the sections at the phase interval.

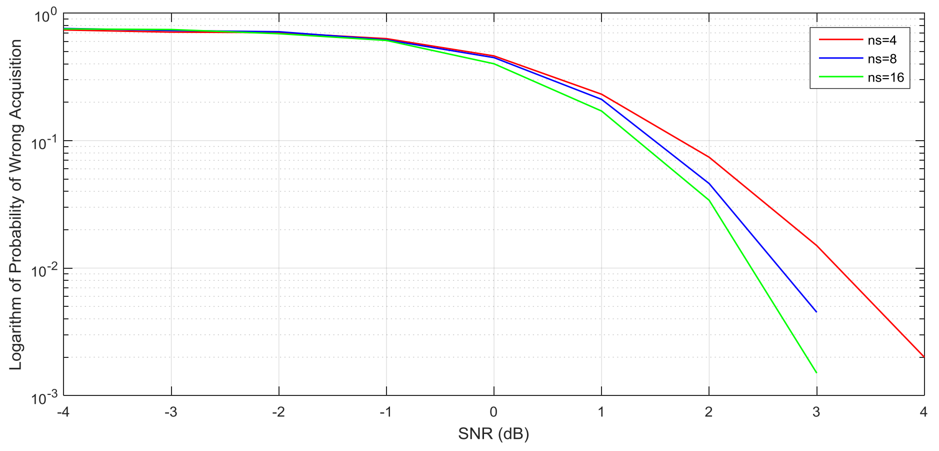


Figure 5. 11. The logarithm of probability of wrong acquisition for different SNR values, $K = 64$, $L = 8$, $R_c = 1/2$, $n_s = 4, 8\ and\ 16$. Here, the number of the simulated packet is 1000.

Now, it is time to see that how the turbo synchronizer converges. To do that, we calculate the BER for different initial phase errors and different number of iterations.

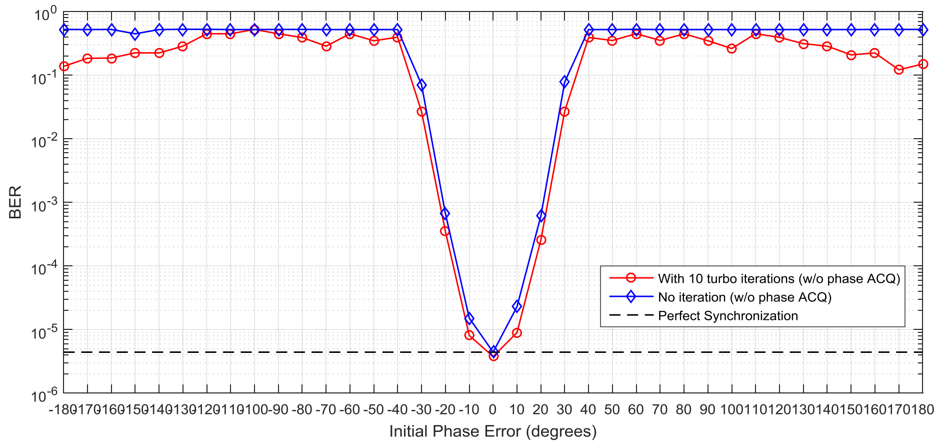


Figure 5. 12. BER versus different phase errors for $K = 64$, $L = 8$, $SNR = 4dB$ and $R_c = 1/2$

Figure 5.12 illustrates the acquisition and the convergence of the implemented turbo synchronizer for the packet length of 64 symbols at SNR value of 4dB. The curves show the BER at the receiver with no iteration and for 10 turbo iterations of the turbo synchronizer (both without phase acquisition). Furthermore, it compare these two cases with a system which is perfectly synchronized.

It is depicted that the acquisition range is 80°degrees, ranges from -40° to 40°. As a result, you can see that the acquisition region in the considered system is not greater than the maximum acquisition range which is achievable by a NDA estimator, even if the algorithm takes the code structure into account. Such feature seems actually to be a function of the used mapping.

Moreover, it is expected that when the initial phase error is at the interval $[-40^{\circ}, 40^{\circ}]$, the BER converges to the perfectly-synchronized system ML performance after some iterations, if the turbo synchronizer works effectively. But this has not happened here and the turbo synchronizer stopped working.

Now, it is important to know when the turbo synchronization stops working. To paraphrase, what is the ultimate limit of the packet length for which turbo synchronization is functional.

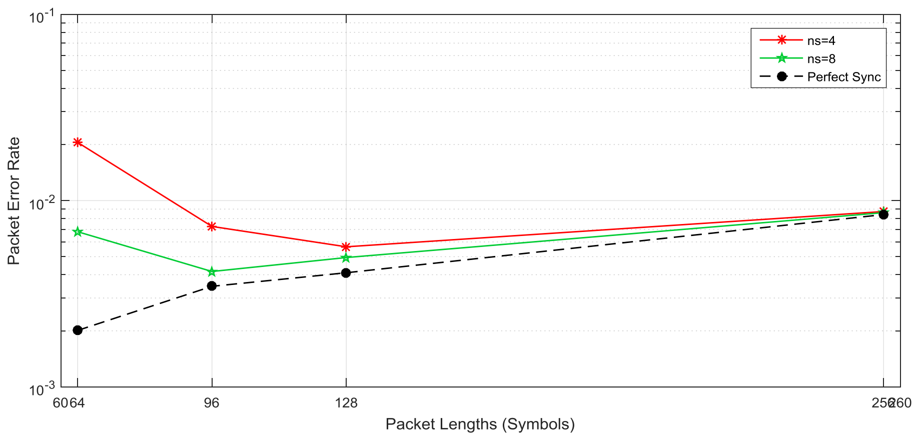


Figure 5. 13. The packet length limit of turbo synchronizer for $SNR = 3dB$ and $L = 8$

Figure 5.13 examines the convergence of the turbo synchronizer for different packet lengths. In the phase acquisition part, we consider two different set of initialization points ($n_s = 4\ and\ 8$) for the LLF. It is predicted that the receiver will have similar PER for both phase acquisitions, if the turbo part works in an effective way.

It is seen that the PER is approximately the same for both n_s when the packet lengths are larger than 128 symbols. They are also near the perfectly-synchronized PER. When the packet size ranges between 96 symbols to 128 symbol, although the receiver with $n_s = 8$ is able to converge to the ML performance because of proper phase acquisition, the turbo synchronizer stops working because PER values for $n_s = 8$ and $n_s = 4$ do not converge to the perfectly-synchronized ones.

To conclude, the turbo synchronization does not work when the packet length is nearly shorter than 128 symbols.

For comparison, we will plot again figure 5.12 but now for a packet length of 128 symbols to study so the convergence for not so short packets.

Figure 5.14 shows the turbo synchronization convergence. The BER curves without iteration, and for 10 turbo iterations are depicted for different initial phase offsets when the packet length is 128 symbols, the constraint length is 8, and the SNR value of 4dB.

It is seen that the turbo synchronizer works at this packet length, and the BER curve converges to the perfectly-synchronized curve after 10 turbo iterations when the initial phase error is less than $\pm 20^{\circ}$.

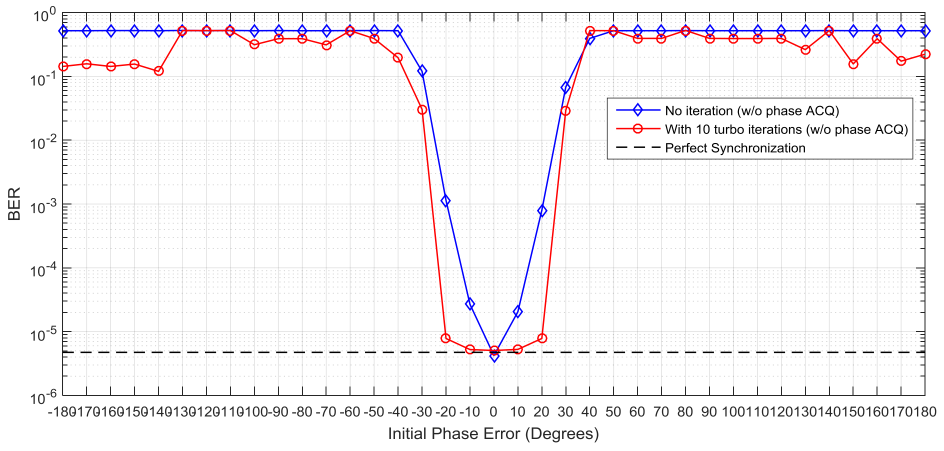


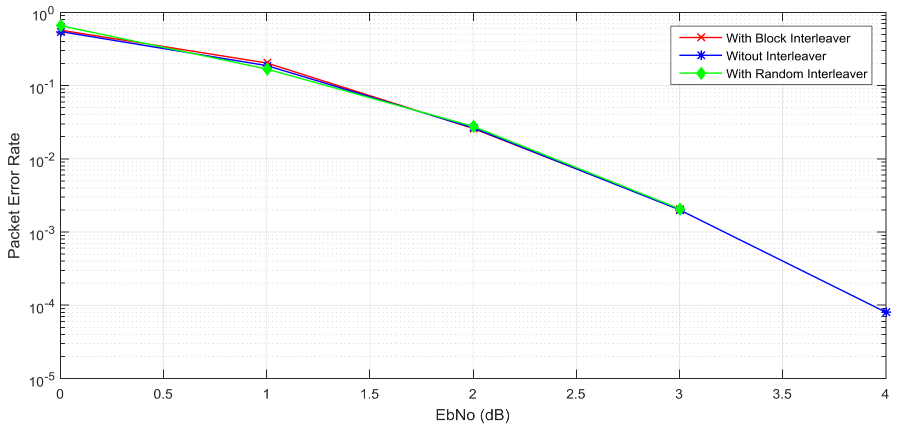
Figure 5. 14. BER versus different phase errors for $K = 128$, $L = 8$, $SNR = 4dB$ and $R_c = 1/2$

5.3. The reliability

In this section, the impact of EM turbo synchronization algorithm on the reliability of the short packet reception without preamble will be analysed. To compare different receivers, the bit-error-rate (BER) and the packet-error-rate (PER) are used as metrics. The BER graphs have been created for 200 erroneous bits and the graphs of PER has been plotted for 100 erroneous packets.
To do that, we perform some simulations for different packet length, different n_s values and different constraint lengths.

Before we run our simulations, the necessity of implementing an interleaver is examined. Usually, the interleaver is implemented to improve the reliability of the system. It is expected from the Bit-Interleaved Coded Modulation (BICM) architecture that the bit-interleaver is only needed for high-order modulations but not in the QPSK case.

Figure 5. 15. PER versus different EbNo (dB) for $K = 64$, $L = 8$, $Itr = 10$ and $R_c = 1/2$



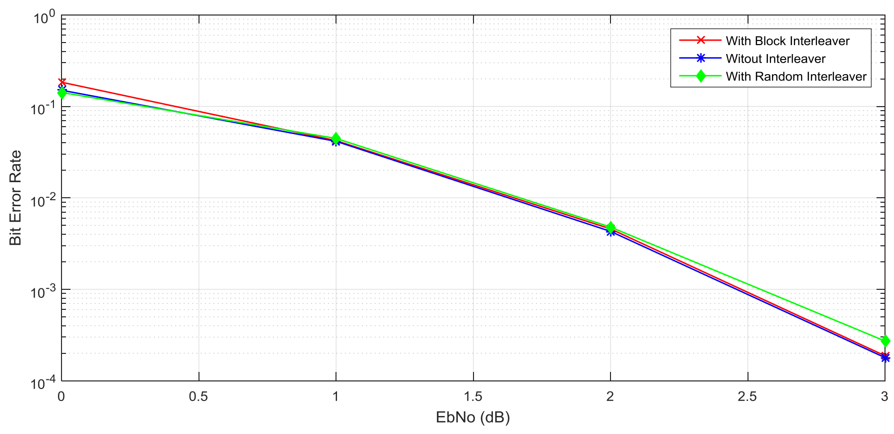


Figure 5. 16. BER versus different EbNo (dB) for $K = 64$, $L = 8$, $Itr = 10$ and $R_c = 1/2$

Figures 5.15 and 5.16 show the impact of the interleaver on the reception reliability. Three different scenarios are considered: with the block interleaver, with the random interleaver and without the interleaver. Both graphs indicate that it is not essential to use bit interleaving between the encoder and the mapper for the QPSK modulation. These results are compatible with the design rules of BICM.

In order to improve the functionality of the EM turbo synchronizer, it is known that proper initial estimation leads to better performance. Therefore, we plot BER and PER for three different partitioning of the interval $[0, 2\pi]$ which lead to different phase acquisition designs.

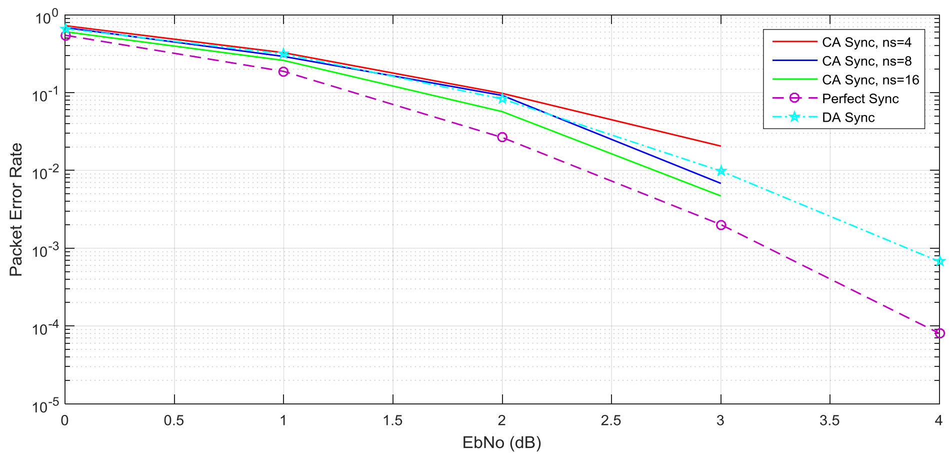


Figure 5. 17. The packet-error-rate versus EbNo (dB) for $n_s = 4, 8 \; and \; 16$, the packet size of 64 symbols, the constraint length of 8 and the preamble size of 10 symbols (0.16 of the packet size)

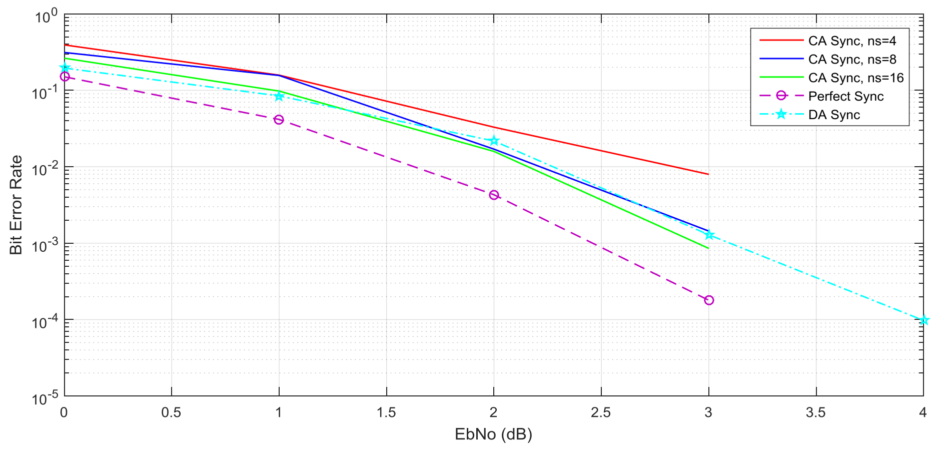


Figure 5. 18. The bit-error-rate versus EbNo (dB) for $n_s = 4, 8\ and\ 16$, the packet size of 64 symbols, the constraint length of 8 and preamble size of 10 symbols (0.16 of the packet size))

Figure 5.17 and 5.18 demonstrate the reliability of the system when we consider different phase acquisition precisions. In addition, they compare those curves with the ones for DA synchronizer with 10-symbols preamble size and the perfect synchronizer.

It is known from the previous acquisition analysis that the turbo synchronizer does not work at this packet size. Although it is expected that BER and PER have the same results for different n_s, it is seen that there is some degradation for $n_s = 4$ and the results of the others are near to DA synchronizer with the $0.16K$ preamble size. It means that, even without turbo iterations, PER and BER for n_s equal to 8 and 16 obtain acceptable results. To paraphrase, CA synchronizer performs better than DA synchronizer but is not able to attain perfect synchronizer reliability.

Then, the impact of the constraint length of convolutional code on the reliability performance is measured. The expectation is that when the constraint length increases, the reliability of the reception increases.

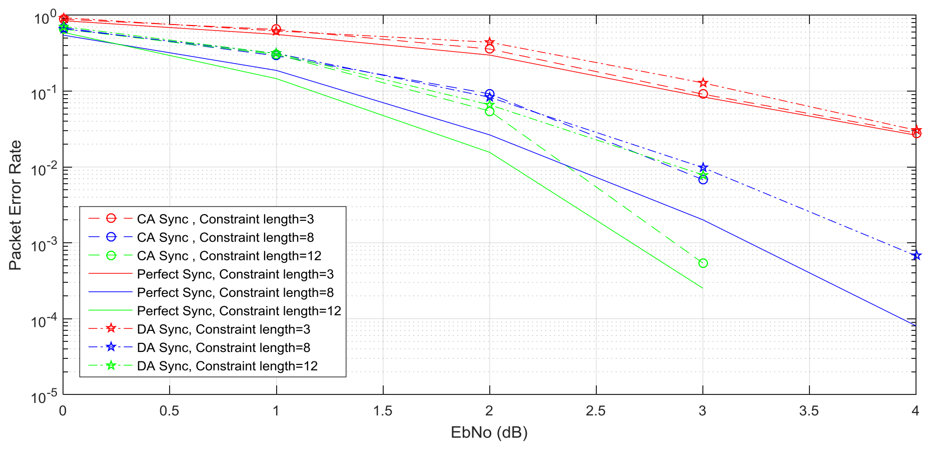


Figure 5. 19. PER versus different EbNo (dB) for $K = 64$ and different constraint lengths. In case of turbo synchronization, the number of iterations is 10 and $n_s = 8$. In data-aided case, the size of preamble is considered 0.16 times the packet length.

Figure 5.19 depicts the packet error rate for different constraint lengths and different degrees of synchronization. We consider $n_s = 8$ for CA synchronization and the preamble size of $0.16K$ for DA synchronization.

In perfect synchronization, as it is expected an increment in constraint length causes an improvement in the packet error rate and the receiver reliability. This result is compatible with the one for long packets

In DA synchronization, although there is an improvement when the constraint length ranges from 3 to 8, the reliability of the constraint length of 12 does not surpass the one of the constraint length of 8 and they coincide each other. The reason is that larger constraint-length codes require better phase synchronization to take benefit for this larger coding gain.

In turbo synchronization, or better to say CA synchronization, there is an improvement when the constraint length rises, especially for high SNR values. Anyway, turbo iteration does not work and the resulting BER/PER only depends on the phase acquisition precision.

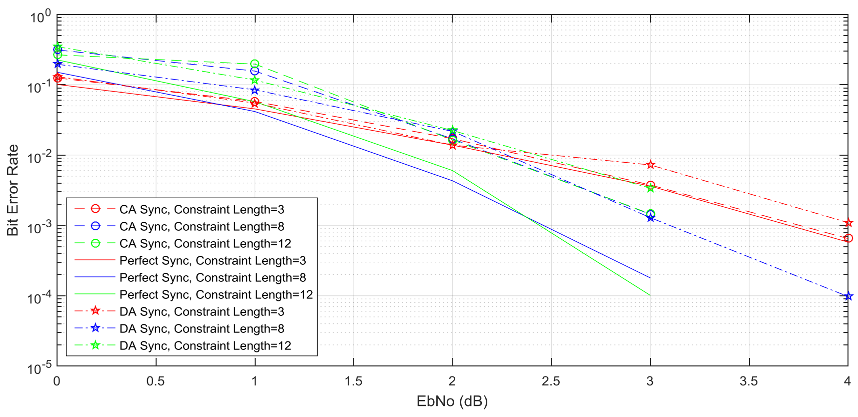


Figure 5. 20. BER for different EbNo (dB) for $K = 64$ and different constraint lengths. In case of turbo synchronization, the number of iteration is 10 and $n_s = 8$. In data-aided case, the size of preamble is considered 0.16 times the packet length.

Figure 5.20 represents the reliability achieved by the receiver in terms of BER for different E_b/N_o-ratio, constraint lengths and synchronization methods. We consider $n_s = 8$ for CA synchronization and the preamble size which is equal to $0.16K$ for DA synchronization.

In CA synchronization, the performance improves by increasing the constraint length from 3 to 8. But the results are the same for constraint lengths 8 and 12. The same behaviour is observed for perfect synchronization. This is because the packet length is too short and the maximum effective memory of the encoder, about 5 times the constraint length, is limited by the codeword length itself. These results are not incompatible with [34] in which an increment in the constraint length is shown to deteriorate the BER for packets whose size is larger than 25 symbols. Simulations were done for constraint lengths 3, 5, and 7. Now we know that they are not suitable selections for proper phase acquisition.

Note that, in the DA case, the BER increases when we increase the constraint length from 8 to 12. Again, the reason is that the variance of the adopted DA estimator is not sufficiently low for this large constraint length value.

Finally, the simulations have been run for different frame lengths to demonstrate its effect on the reliability performance.

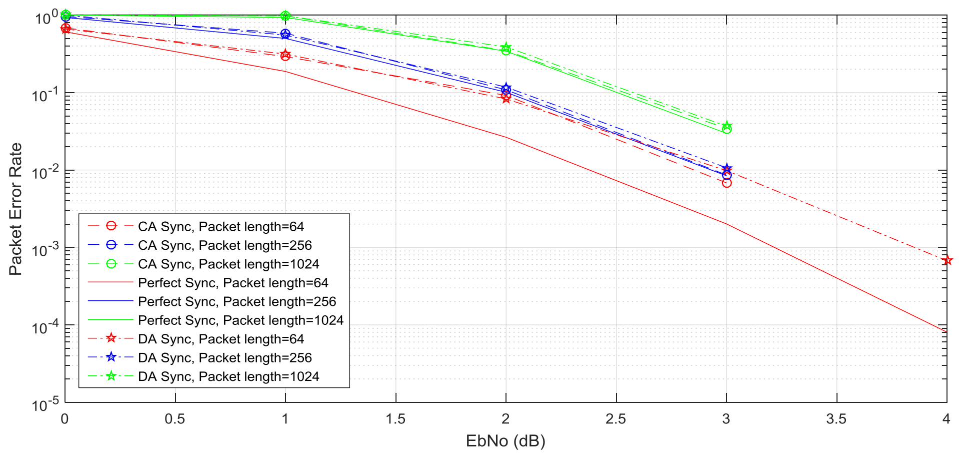


Figure 5. 21. PER versus EbNo (dB) for different packet lengths and $L = 8$. In case of code-aided synchronization, the number of iteration is 10 and $n_s = 8$. In data-aided case, the size of preamble is considered 0.16 of the packet length.

Figure 5.21 shows how the packet length affects the reliability of the receiver. The simulation are performed for three different packet lengths: 64, 256 and 1024 symbols with preamble sizes of 10, 38 and 154 symbols for DA case, respectively. The number of turbo iterations is 10 and the phase acquisition is made with $n_s = 8$. The constraint length is set to 8.It is clear that the larger the packet size, the worse the PER. It may be justified that when the size of packet increases, the time spent for transmitting the packet increases. Therefore, it is more probable that the packet is corrupted by the noise. Furthermore, it proves that for a fixed convolutional encoder memory, performance does not improve with the packet length.

For packet lengths 256 and1024 symbols, it is seen that the PER curves for the three types of synchronization coincide each other. The reason is that the turbo synchronizer converges to ML performance at this packet length according to the result of Figure 5.13 and, in the DA case, the size of the preambles are noticeable at this length. It means that the larger the preamble, the more precise the phase estimation.

For packet length 64 symbols, because the turbo part of the synchronizer does not work and the preamble size is too short, DA and CA curves do not touch the curve of the perfectly synchronized receiver.

Figure 5.22 indicates the impact of the packet length on the BER. The curves are formed for three different packet lengths: 64, 256 and 1024 symbols with the preamble sizes of 10, 38 and 154 symbols for DA case, respectively. The number of turbo iterations is 10 and the phase acquisition is made with $n_s = 8$.The constraint length is considered 8.

It is noticeable that the BER does not depend on the packet length. It only depends on the constraint length. Provided that codeword length larger than approximately $5.L$. Therefore, it is logical that PER increases with the packet length if BER does not change for length greater than $5.L$.

For packet lengths of 256 and 1024 symbols, the BER results for the DA synchronizer and turbo synchronizer coincide with the perfect synchronization results. But it is seen that a significant degradation occurs for the packet length of 64 symbols due to the failure of turbo synchronization in the CA case and insufficient preamble size in the DA case.

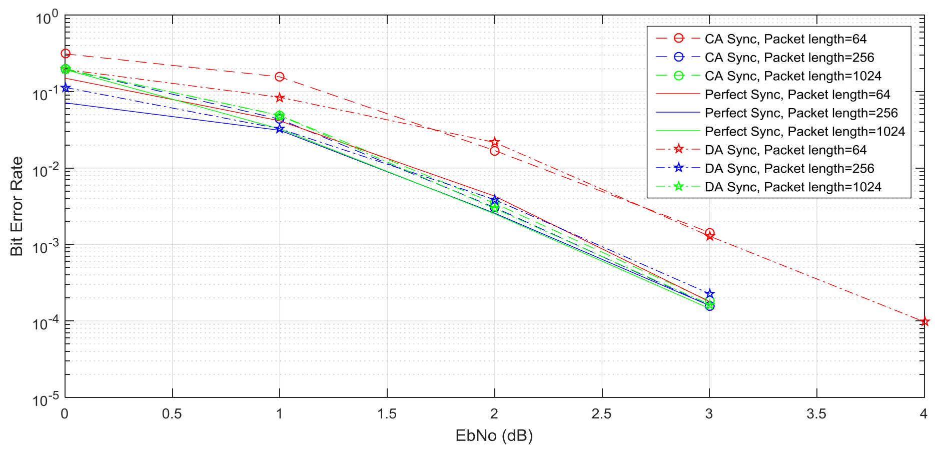


Figure 5. 22. BER versus EbNo (dB) for different packet lengths and $L = 8$. In case of code-aided synchronization, the number of iteration is 10 and $n_s = 8$. In data-aided case, the size of preamble is considered 0.16 of the packet length.

6. Conclusions and future development

After the advent of the fifth generation of cellular system, according to its applications, its services are divided into three categories. One of these categories is Ultra reliable Low Latency Communications. To make this category a reality, there are three potential research directions based on the layered structure. One of them is physical layer schemes with reduced latency. The user plane latency consists of the time-to-transmit latency, the propagation delay, the processing latency and retransmission time. The focus of this thesis is to decrease latency by reducing the time-to-transmit using short packets.

As a direct consequence of Shannon masterpiece in information theory [28], coding is used to enhance the performance of the digital communication system. In order to decode the codewords uniquely at the receiver side, it is necessary to do initial synchronization. Conventionally, synchronization was done in data-aided and non-data-aided modes. Then, code-aided synchronization, which exploits the channel code structure and its properties to estimate the synchronization parameters, was introduced.

When the goal is to minimize the time required to synchronize the receiver with the transmitter, usually DA synchronization is used. It synchronizes the system with the help of a preamble. In URLLC, short packet transmission is considered as the major traffic. In short packet transmission, the size of the preamble is reduced proportionally to the information size. When the packet length is too short, the preamble length can be insufficient to synchronize satisfactorily the receiver. Therefore, it is proposed to skip or directly eliminate the preamble and do the synchronization task by means of turbo synchronization.

Turbo synchronization starts from an initial estimate of the synchronization parameters, then a decision is made about the transmitted symbols. After that, this decision is used to compute a new estimate of the synchronization parameters and so forth. The proposed turbo synchronizer is implemented by means of an EM algorithm whose initial estimate is based on the partitioning of the phase interval $[0,2\pi]$. Then, the soft information related to each of these phase correction points is obtained at the output of the decoder, which is the BCJR decoder in this work. This decoder is used as the optimal soft-input soft-output decoder algorithm in terms of minimum bit error rate and uses the MAP criterion to estimate the a posteriori probability of the received symbols. The phase whose likelihood is maximum is the initial phase estimate.

In order to assess the performance of the turbo synchronization for short packets without preambles, some simulations have been run. The assessments are in terms of accuracy, acquisition range and reliability.

In the accuracy section, the phase variance of three types of synchronizers, DA, NDA and the turbo synchronizer, are plotted and compared to their related CRLBs. It is seen that the turbo synchronizer variance outperforms the DA synchronizer variance for SNR values which are greater than 2.5dB for packet length of 64 symbols (preamble length 6 symbols) and attains the MCRLB for SNR values higher than 4dB. Furthermore, with such packet length, the NDA estimator variance even surpasses the DA estimator variance for SNR higher than 3.5dB. The accuracy enhances by reducing constraint length or decreasing the TBCC code rate, or increasing the packet length.

In the acquisition range part, it is shown that phase acquisition precision depends on the constraint lengths and their related generator polynomials. For odd constraint lengths which are larger than 7, there is 180^o ambiguity in the average LLF. This symmetry implicitly means that the turbo synchronizer will not be able to distinguish between two phase offsets that are π apart. The substitution of the generator polynomials can solve this ambiguity at low SNR values. As a result, the maximum achievable phase acquisition range improves from $\pm\pi/_4$ in NDA synchronization to $\pm\pi$ in CA case. It is expected that, when the phase error is in the range of $\pm\pi/_4$, the turbo synchronization performance converges to the perfect synchronization performance. But the turbo synchronizer does not work effectively for packet length of 64 symbols. Turbo synchronization works when the packet length is larger than 128 symbols and converges to the perfectly-synchronized system when the phase error ranges from -20° to 20° after 10 turbo iterations. It is worth mentioning that because the acquisition phase is usually near the true phase, the gain of EM-based turbo synchronization is not significant.

Finally, in the reliability section, we plotted BER and PER curves to evaluate the synchronizers performance. For short packet lengths, the BER and PER curves deteriorate by decreasing the number of initial phase estimates considered for phase acquisition due to the inactivity of the turbo synchronizer, although the original expectation was that they would achieve the same results. The BER curves improve when the constraint length increases, although they do not attain the perfect synchronization curves. They remain the same for different packet lengths expect for the 64-symbols packet size. The PER curves improve by increasing the constraint length, despite their huge distance to the perfectly-synchronized curves for higher constraint lengths when the packet length is limited to 64 symbols. And decreasing the packet length leads to an improvement in the PER curves, in spite of the huge distance to the perfectly-synchronized curve for the packet length of 64 symbols.

In all of these simulations, turbo synchronizer curves outperform or attain the DA curves. It means that the idea of eliminating the preamble from the short packet seems practical.

In future work, it is recommended to choose the code rate according to the packet length in such way that the transmitted symbols become higher than 128 symbols. For example, code rates lower than 1/4 seem a good choice for the packet length of 64 symbols. Since it is known that lower code rates lead to higher latency, it is not recommended to choose too low code rates. Also, the 16-QAM modulation seems good candidate to be implemented because it would allow to select code rate 1/4 without increasing the transmission latency.

Bibliography

[1] G. Durisi, T. Koch, P. Popovski. “Toward Massive, Ultrareliable, and Low-Latency Wireless Communication with Short Packets”. *Proceedings of the IEEE*, vol. 104, no. 9, pp. 1711-1726, September 2016. DOI: 10.1109/JPROC.2016.2537298

[2] D. Jacobs. “IETF SDN: I2RS uses traditional routing protocols in software networks”. TechTarget, 2010. [Online] Available: http://searchsdn.techtarget.com/tip/IETF-SDN-I2RSuses-traditional-routing-protocols-in-software-networks

[3] ETSI, Network Functions Virtualisation, ETSI, 2014. [Online] Available: https://www.etsi.org/technologies/nfv

[4] X. Xu, G. He, S. Zhang, Y. Chen, S. Xu. “On Functionality Separation for Green Mobile Networks: Concept Study over LTE”. *IEEE Communications Magazine*, vol. 51, no. 5, pp. 82-90, May 2013. DOI: 10.1109/MCOM.2013.6515050

[5] R1-1804849,“Remaining Issues on URLLC Data Channel Coding”, 3rd Generation Partnership Project, 3GPP TSG RAN WG1 Meeting #92bis-Discussion and Decision, April 2018. [Online] Available: https://www.3gpp.org/DynaReport/TDocExMtg--R1-92b--18780.htm

[6] R1-1608770, “Flexibility evaluation of channel coding schemes for NR”, 3rd Generation Partnership Project, 3GPP TSG RAN WG1 Meeting #86- Discussion and Decision, October 2016. [Online] Available: https://www.3gpp.org/DynaReport/TDocExMtg--R1-86b--31664.htm

[7] J. Ostman, G. Durisi, E. G. Strom. “Finite-blocklength bounds on the maximum coding rate of Rician fading channels with applications to pilot-assisted transmission*”. IEEE 18th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC), IEEE 2017*, 3-6 July 2017, Sapporo, Japan. pp. 1-5. doi: 10.1109/SPAWC.2017.8227650

[8] M. Shirvanimoghaddam, M. S. Mohammadi, R. Abbas, A. Minja, C. Yue, B. Matuz, G. Han, Z. Lin, W. Liu, Y. Li, S. Johnson, B. Vucetic. “Short Block-Length Codes for Ultra-Reliable Low Latency Communications”. *IEEE Communications Magazine*, vol. 57, no. 2, pp. 130-137, February 2019. DOI: 10.1109/MCOM.2018.1800181

[9] C. Rachinger, J. B. Huber, R. R. Muller. “Comparison of Convolutional and Block Codes for Low Structural Delay”. *IEEE Transactions on Communications*, vol. 63, no. 12, pp. 4629-4638, December 2015. DOI: 10.1109/TCOMM.2015.2488661

[10] A. T. P. Nguyen, R. L. Bidan, F. Guilloud. “Superimposed Frame Synchronization Optimization for Finite Blocklength Regime”. *IEEE Wireless Communications and Networking Conference (IEEE WCNC), IEEE 2019*, 15-19 April 2019, Marrakech, Morocco. pp. 1-6. doi: 10.1109/LCOMM.2019.2913363

[11] B. Lee, S. Park, D. J. Love, H. Ji, B. Shim. “Packet Structure and Receiver Design for Low-Latency Communications with Ultra-Small Packets”. *IEEE Global Communications Conference (GLOBECOM), IEEE 2016*, 4-8 December 2016, Washington, DC, USA. pp. 1-6. doi: 10.1109/GLOCOM.2016.7842122

[12] C. Herzet, N. Noels, V. Lottici, H. Wymeersch, M. Luise, M. Moeneclaey, L. Vandendorpe. “Code-Aided Turbo Synchronization*”. Proceeding of the IEEE*, vol. 95, no. 6, pp. 1255-1271, June 2007. DOI: 10.1109/JPROC.2007.896518

[13] A. –S. Bana, K. F. Trillingsgaard, P. Popovski, E. D. Carvalho. "Short Packet Structure for Ultra-Reliable Machine-Type Communication: Tradeoff between Detection and Decoding". *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE 2018*, 15-20 April 2018, Calgary, AB, Canada. pp. 6608-6612. doi: 10.1109/ICASSP.2018.8461650

[14] P. Popovski, J. J. Nielsen, C. Stefanovic, E. D. Carvalho, E. Strom, K. F. Trillingsgaard, A. –S. Bana, D. M. Kim, R. Kotaba, J. Park, R. B. Sorensen. "Wireless Access for Ultra-Reliable Low-Latency Communication (URLLC): Principles and Building Blocks". *IEEE Network*, vol. 32, no. 2, pp. 16-23, March-April 2018. DOI: 10.1109/MNET.2018.1700258

[15] L. M. Zhang, F. R. Kschischang. "Multi-Edge-Type Low-Density Parity-Check Codes for Bandwidth-Efficient Modulation". *IEEE Transactions on Communications*, vol. 61, no. 1, pp. 43-52, January 2013. DOI: 10.1109/TCOMM.2012.100512.120006

[16] S. B. Korada, E. Sasoglu, R. Urbanke. "Polar Codes: Characterization of Exponent, Bounds, and Constructions". *IEEE Transactions on Information Theory*, vol. 56, no. 12, pp. 6253-6264, December 2010. DOI: 10.1109/TIT.2010.2080990

[17] T. Hehn, J. B. Huber. "LDPC Codes and Convolutional Codes with equal structural delay: a comparison*". IEEE Transactions on communications*, vol. 57, no. 6, pp.1683-1692, June 2009. DOI: 10.1109/TCOMM.2009.06.080014

[18] S. V. Maiya, D. J. Costello, T. E. Fuja. "Low Latency Coding: Convolutional Codes vs. LDPC Codes". *IEEE Transactions on Communications*, vol. 60, no. 5, pp. 1215-1225, May 2012. DOI: 10.1109/TCOMM.2012.042712.110189

[19] N. A. Johansson, Y. -P. E. Wang, E. Eriksson, M. Hessler. "Radio access for ultra-reliable and low-latency 5G communications". *IEEE International Conference on Communication Workshop (ICCW), IEEE 2015*, 8-12 June 2015,London, UK .pp. 1184-1189. doi: 10.1109/ICCW.2015.7247338

[20] B. Lee, S. Park, D. J. Love, H. Ji, B. Shim. "Packet Structure and Receiver Design for Low Latency Wireless Communications With Ultra-Short Packets". *IEEE Transactions on Communications*, vol. 66, no. 2, pp. 796-807, February 2018. DOI: 10.1109/TCOMM.2017.2755012

[21] D. -S. Yoo, W. E. Stark, K. -P. Yar, S. -J. Oh. "Coding and Modulation for Short Packet Transmission". *IEEE Transactions on Vehicular technology*, vol. 59, no. 4, pp. 2104-2109, May 2010. DOI: 10.1109/TVT.2010.2041378

[22] S. Ashraf, Y. -P. E. Wang, S. Eldessoki, B. Holfeld, D. Parruca, M. Serror, J. Gross. "From Radio Design to System Evaluations for Ultra-Reliable and Low-Latency Communication*". European Wireless 2017; 23th European Wireless Conference*, 17-19 May 2017, Dresden, Germany. pp. 1-8.

[23] Y. Polyanskiy, H. V. Poor, S. Verdu. "Channel Coding Rate in the Finite Blocklength Regime". *IEEE Transactions on Information Theory*, vol. 56, no. 5, pp. 2307–2359, May 2010. DOI: 10.1109/TIT.2010.2043769

[24] F. Bellili, A. Methenni, S. Affes. "Closed-Form CRLBs for CFO and Phase Estimation from Turbo-Coded Square-QAM-Modulated Transmissions". *IEEE Transactions on Wireless Communications*, vol. 14, no. 5, pp. 2513-2531, May 2015. DOI: 10.1109/TWC.2014.2387855

[25] R1-071323 Motorola, “Performance of convolutional codes for the E-UTRA DL control channel”, 3rd Generation Partnership Project, 3GPP TSG RAN1 #48bis-Discussion and Decision, March 2006. [Online] available: https://www.3gpp.org/DynaReport/TDocExMtg--R1-48b--26242.htm

[26] R1-073033 Ericsson, “Complexity and performance improvement for convolutional coding”, 3rd Generation Partnership Project, 3GPP TSG-RAN WG1 #49bis- Discussion and Decision, June 2007. [Online] available: https://www.3gpp.org/DynaReport/TDocExMtg--R1-49b--26481.htm

[27] J. G. Proakis, M. Salehi. *Communication System Engineering*, 2nd ed. New Jersey, USA: Prentice-Hall Inc, 2002.

[28] C. E. Shannon. “A Mathematical Theory of Communications”. *The Bell System Technical Journal*, vol. 27, no. 3, pp. 379-423, July 1948. DOI:10.1002/j.1538-7305.1948.tb01338.x

[29] G. Liva, F. Steiner. “Channel Codes for Short Blocks: A Survey”. *11th International ITG Conference on Systems, Communications and Coding, SCC 2017*, 6-9 February 2017, Hamburg, Germany. pp. 1-6.

[30] A. N. D’Andrea, U. Mengali, R. Reggiannini. “The Modified Cramer-Rao bound and its application to synchronization problems”. *IEEE Transactions on Communications*, vol. 42, no. 234, pp. 1391-1399, February/March/April 1994. DOI: 10.1109/TCOMM.1994.580247

[31] J. -P. Delmas. “Closed-Form Expressions of the Exact Cramer-Rao Bound for Parameter Estimation of BPSK, MSK, or QPSK waveforms”. *IEEE Signal Processing Letters*, vol. 15, pp.405-408, April 2008, DOI: 10.1109/LSP.2008.921477

[32] F. Bellili, A. Methenni, S. Affes. “Closed-Form CRLBs for SNR Estimation from Turbo-Coded BPSK-, MSK-, and Square-QAM-Modulated Signals”. *IEEE Transactions on Signal Processing*, vol. 62, no. 15, pp. 4018-4033, August 2014. DOI: 10.1109/TSP.2014.2328328

[33] N. Noels, C. Herzet, A. Dejonghe, V. Lottici, H. Steendam, M. Moeneclaey, M. Luise, L. Vandendrope. “Turbo synchronization: an EM algorithm interpretation”. *IEEE International Conference on Communications (ICC03), IEEE 2003*, 11-15 May 2003, Anchorage, AK, USA. pp. 2933-2937. doi: 10.1109/ICC.2003.1204575

[34] C. Herzet. “Code-Aided Synchronization for Digital Burst Communications,” Ph.D. dissertation, Communications Laboratory, Universite catholique de Louvain, Louvain-la-Neuve, Belgium, 2006.

[35] L. Gaudio, T. Ninacs, T. Jerkovits, G. Liva. “On the Performance of Short Tail-Biting Convolutional Codes for Ultra-Reliable Communications”. *11th International ITG Conference on Systems, Communications and Coding, SCC 2017*, 6-9 February 2017, Hamburg, Germany. pp. 1-6.

[36] G. Durisi, G. Liva, F. Steiner. “Short-packet communications: fundamentals and practical coding schemes”. *IEEE Global Communications Conference, IEEE GLOBCOM 2018*,9-13 December 2018, Abu Dhabi, UAE. pp. 1-339.

[37] J. G. Proakis, M. Salehi. *Digital Communications*, 5th ed. New York, USA: McGraw-Hill Companies Inc, 2008.

[38] S. M. Kay. *Fundamentals of Statistical Signal Processing, Volume I: Estimation Theory*, 1st ed. New Jersey, USA: Prentice Hall, 1993.

[39] J. B. Anderson, S. M. Hladik. “Tailbiting MAP Decoders*”. IEEE Journal on Selected Areas in Communications*, vol. 16, no. 2, pp. 297-302 , February 1998. DOI:10.1109/49.661117

[40] P. J. Lee. “New Short Constraint Length, Rate 1/N Convolutional Codes Which Minimize the Required SNR for Given Desired Bit Error Rates”. *IEEE Transactions on Communications*, vol. 33, no. 2, pp. 171-177, February 1985.
DOI: 10.1109/TCOM.1985.1096259

[41] RP-151761, “*Study on scenarios and requirements for next generation access technologies”, technical specification group radio access network*. 3GPP, Tech. Rep. 38.913, October 2016. https://www.3gpp.org/ftp/Specs/archive/38_series/38.913/

Glossary and Notations

Ack	Acknowledgement
APP	A Priori Probability
AWGN	Additive White Gaussian Noise
BCH	Bose-Chaudhuri-Hocquenghem
BCJR	Bahl-Cocke-Jelinek-Raviv
BER	Bit Error Rate
BG	Base Graph
BICM	Bit-Interleaved Coded Modulation
CA	Code-Aided
CAPEX	Capital Expenditure (Capital Expense)
CRLB	Cramer Rao Lower Bound
CSI	Channel State Information
DA	Data-Aided
dB	Decibel
DD	Decision decoding
DoFs	Degrees of Freedom
EM	Expectation Maximization
eMBB	enhanced Mobile Broadband
5G	the fifth Generation
HARQ	Hybrid Automatic Repeat Request
i.i.d	Independent and Identically Distributed
IMT	International Mobile Telecommunications
IoT	Internet of Things
ISI	Inter-Symbol Interference
ISPB	Improved Sphere Packing Bound
IT	Information Technology
ITU	International Telecommunication Union
ITU-T	ITU-Telecommunication Standardization Sector
LAPPR	Logarithms of a Posteriori Probability Ratios
LDPC	Low-Density Parity Check
LLF	Logarithm of Likelihood Function
LLR	Logarithm of Likelihood Ratio
Log-APP	Logarithm of A Posteriori Probability

LTE	Long Term Evolution
MAC	Medium Access Control
MAP	Maximum A Posteriori Probability
Max-Log-APP	Maximum of Logarithm of A Posteriori Probability
MCRLB	Modified Cramer Rao Lower Bound
ML	Maximum Likelihood
mMTC	massive Machine Type Communication
ms	Millisecond
MVU	Minimum Variance Unbiased
NDA	Non-Data-Aided
NFV	Network Functions Virtualization
PDF	Probability Density Function
PER	Packet Error Rate
PSK	Phase Shift Keying
QAM	Quadrature Amplitude Modulation
QPSK	Quadrature Phase Shift Keying
RCB	Random Coding Bound
SDD	Soft Decision Decoding
SDN	Software Defined Networking
SINR	Signal-to-Interference-plus-Noise Ratio
SISO	Soft-Input Soft-Output
SNR	Signal-to-Noise Ratio
SPB	Sphere Packing Bound
TBCC	Tail-biting Convolutional Code
TTCC	Tail-terminating Convolutional Code
3D	Three Dimensional
3GPP	Third Generation Partnership Project
UHD	Ultra High Definition
URLLC	Ultra-Reliable Low-Latency Communication
VPS	Virtual pilots
VV	Viterbi & Viterbi
WAVA	Wrap-Around Viterbi Algorithm

$arg\{.\}$	The argument of
$argmax$	The argument that maximize
$E[.]$	The expectation of
$f(.)$	A many-to-one mapping
$I(.)$	The Fisher information matrix
$I^{-1}(.)$	The inverse of Fisher information matrix
$\Im\{.\}$	The imaginary part of
$LLF(.)$	The logarithm of likelihood function
$LLR(.)$	The soft output of decoder (The a posteriori LLR values)
$Q(.)$	The Q-function
$\Re\{.\}$	The real part of
$\Lambda(.)$	The likelihood function
$\in$	Belongs to
$\sum_{k}.$	The summation over k

$a(t)$	The complex transmitted signal (The complex baseband waveform)
$a^*(t)$	The complex conjugate of $a(t)$
a_k	The k-th element of the a sequence of complex signals, $\boldsymbol{a}$
$\breve{\boldsymbol{a}}$	The training sequence (the preamble)
$\breve{a}_k$	The value of the preamble symbol at time k
$\boldsymbol{a}$	The vector of random discrete-valued nuisance parameter
$\mathcal{A}$	The signal constellation
$\boldsymbol{b}$	The information sequence
$\hat{b}$	The decision on the transmitted information bit b
$B_{m'}$	The actual design bandwidth
$\boldsymbol{c}$	The encoded sequence
c'_n	The $n-th$ element of the twin of the codeword, $\acute{\boldsymbol{c}}$
E_b	Energy per bit
f_c	The central frequency of passband filter
$f_N(.)$	The probability density function of AWGN
f_{co}	The codeword as function of the input and the previous state
f_s	The new state as function of the input and the previous state
$g(t)$	The convolution of $p(t)$ and $q(t)$
$h_{SRRC}(t)$	The impulse response of the square-root-raised-cosine filter

$H_{SRRC}(f)$	The transfer function of the square-root-raised-cosine filter
$H(f)$	The frequency response of the matched filter
$H_{RX}(f)$	The frequency response of the receiver filter
i	The counter of turbo iterations
j	The imaginary number $j = \sqrt{-1}$
K	The total number of Information bits (The number of transmitted symbol)
k'	The number of information bits which enter the encoder at each time instance
K_{pre}	The number of preamble symbols
k_0	The time instant of the first observed sample
L	The constraint length of the convolutional code (the trellis depth)
m	Memory of Convolutional code
$M = \|\mathcal{A}\|$	The size of signal constellation (the modulation order)
m'	The number of segmentation of the bit sequence in the modulator
$n(t)$	The complex noise at the output of the matched filter
$n_1(t)$	The in-band noise
$n_2(t)$	The lowpass equivalent complex noise at the input of the matched filter
n'	The number of binary symbols at the output of the encoder
n_s	The number of phase obtained at the phase interval $[\hat{\varphi}_{VV}, \hat{\varphi}_{VV} + 2\pi]$
N_0	The power spectral density of noise
$p(t)$	The complex basic waveform satisfying Nyquist criterion (baseband filter)
$P(f)$	The frequency response of $p(t)$
$P(\boldsymbol{b})$	The prior probability of the information sequence
$P(\boldsymbol{a})$	Priori probability density function of $\boldsymbol{a}$
$q(t)$	The baseband filter at the receiver side (the baseband demodulator)
$r(t)$	The lowpass equivalent received signal
$R_c = \frac{k}{n}$	The base code rate of convolutional code
R	Data rate
S	The set of possible coded sequences
S_0	The subset of all pairs of states which corresponds to $b_k = 0$
S_1	The subset of all pairs of states which corresponds to $b_k = 1$
T	The symbol period
$w(t)$	The additive random noise process
$W_{m'}$	The nominal baseband bandwidth (the Nyquist bandwidth)

$x(t)$	The real passband transmitted waveform
$X(f)$	The transmitted signal spectrum
$\boldsymbol{z}$	A random vector obtained by expanding the received modulated-signal $z(t)$ onto suitable basis (The actual observation set or the incomplete data set)
$z(t)$	The output of the matched filter
$\check{\boldsymbol{z}}$	The extended observation set (the complete data set)
$\boldsymbol{z}'$	The received vector at the output of de-mapper
$\mathbb{Z}$	Integers
α'	The rolloff factor
α	The forward recursion
$\tilde{\alpha}$	The logarithm of α
β	The backward recursion
$\tilde{\beta}$	The logarithm of β
γ	
$\tilde{\gamma}$	The logarithm of γ
v	The carrier frequency offset
$\hat{v}$	The estimate of carrier frequency offset
μ	The mean value of noise
ρ	The energy per transmitted symbol over noise at the receiver input
σ_N^2	The variance of noise
σ_k	The encoder state at time k
τ	The timing error
$\hat{\tau}$	The estimate of timing error
φ	The carrier phase offset
$\tilde{\varphi}$	A trial value of φ
$\hat{\varphi}$	The estimate of carrier phase offset
$\hat{\varphi}^{(i)}$	The phase estimate computed at i-th iteration
Σ	The set of all states