ANR, Appel à Projets Générique (AAPG 2019)

QCSP Project (ANR-19-CE25-0013-02)

Deliverable D2.3a
Blind Synchronization Algorithm For QCSP

Editor:	Lab-STICC/UBS
Deliverable nature:	Internal (scope: Consortium and ANR)
Due date:	November 2020
Delivery date:	November 2020
Version:	1.0
Total number of pages:	29 pages
Keywords:	CCSK modulation, symbol over-modulation, chip synchronization

Abstract
This deliverable reports on the specifications of achieving Blind synchronization for the Quasi Cyclic Short Packet (QCSP).

List of Authors

Partner	Author
LAB-STICC/UBS	Kassem Saied (kassem.saied@univ-ubs.fr)
LAB-STICC/UBS	Emmanuel Boutillon (Emmanuel.Boutillon@univ-ubs.fr)

Contents

Executive Summary

Work Package 2 (WP2) of the QCSP project is concerned with the proposition and evaluation of algorithms to achieve an efficient detection and synchronization of the CCSK-modulated Frame using the soft output provided to the decoder and exploiting the particular structure of the frame.

The goal of **Task 2.3** is to propose new synchronization algorithm with all the corresponding time and frequency specifications that match the Scenario of IoT systems. The Synchronization performance will be assessed through Monte-Carlo simulations.

Figure 1: Gantt diagram of WPx

This deliverable is organized as follows:

Chapter 1 describes the overall communication chain principle and discusses the decomposition of the processing, carried at the receiver, into two dimensions: frequency and time, to perform blind detection and synchronization.

Chapter 2 illustrates in details with examples the steps of the blind synchronization process.

Chapter 3 gives a brief result for all SNRs with a conclusion. And Chapter 4 is the appendix for more details.

1 System Model

This chapter describes in details the overall Communication principle of the QCSP system model. It also defines the synchronization after the reception of the frame through asynchronous AWGN channel. Fig. 1.1 shows a simplified description of the communication link.

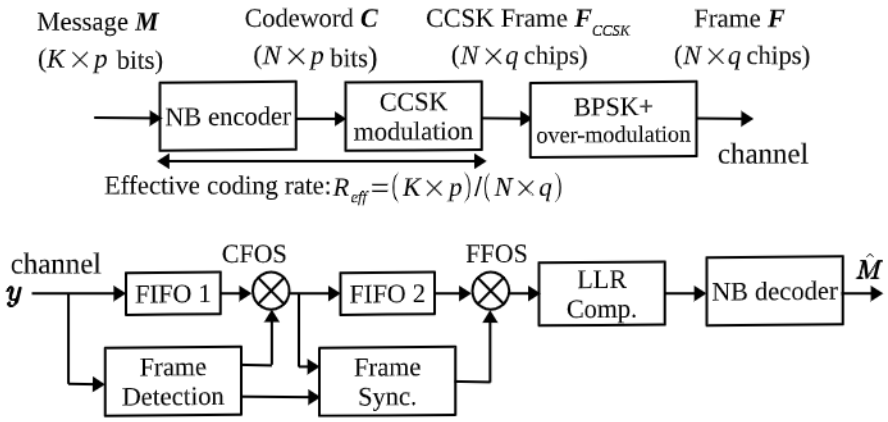

Figure 1.1: Overall Communication Principle (CFOS: Coarse Frequency Offset suppression, FFOS: Fine Frequency Offset suppression)

1.1 Transmitter side

We consider a NB code defined over the Galois field with q elements, denoted by GF(q). The input of the NB-code is a binary message $\boldsymbol{M}$ of size $m = K \times p$ information bits, equivalently K GF(q) symbols. The NB-encoder generates a codeword $\boldsymbol{C}$ of N GF(q) symbols, i.e., $\boldsymbol{C} = [c_0, c_1, \ldots, c_{N-1}]$, with $c_k \in \text{GF}(q)$, $k = 0, 1, \ldots, N-1$. Let $\boldsymbol{H}$ be the parity check matrix associated to the NB code, then the codeword C verifies $\boldsymbol{HC} = 0$. The CCSK modulation uses a pseudo-random binary sequence $\boldsymbol{P}_0 = \{P_0(i)\}_{i=0,\ldots,q-1}$ of length q, where $P_0(i) \in \{0, 1\}$. The CCSK modulation maps an element c of GF(q) to the sequence $\boldsymbol{P}_c$ defined as the circular right shift of $\boldsymbol{P}_0$ by c positions, i.e., $\boldsymbol{P}_c = \{P_0(i - c \bmod q)\}_{i=0,\ldots,q-1}$. The link between the element c of GF(q) and the integer shift value c is done by considering the element of GF(q) as polynomial over GF(2)$[X]/P[X]$. The binary representation of this polynomial gives the binary representation of the integer value c.

So $\boldsymbol{P}_{c_k}$ is the circularly right shifted sequence of $\boldsymbol{P}_0$ by c_n positions. The CCSK frame $\boldsymbol{F}$ is thus defined as the concatenation of N CCSK symbols, i.e., $\boldsymbol{F} = [\boldsymbol{P}_{c_0}, \boldsymbol{P}_{c_1}, \ldots, \boldsymbol{P}_{c_{N-1}}] = \coprod_{k=0}^{N-1} \boldsymbol{P}_{c_k}$, where $\coprod$ represents the concatenation operation.

In a very noisy environment, determining when the sequence starts at the exact symbol level is not a trivial task. To help the synchronization, a symbol over-modulation was added to the CCSK symbols. So instead of transmitting $\boldsymbol{F}$ as defined before, we sent the over-modulated QCSP frame $\boldsymbol{F}$ such that:

$$\begin{aligned} \boldsymbol{F} &= [(-1)^{b_0}\boldsymbol{P}_{c_0}, (-1)^{b_1}\boldsymbol{P}_{c_1}, \ldots, (-1)^{b_{N-1}}\boldsymbol{P}_{c_{N-1}}] \\ &= \coprod_{k=0}^{N-1} (-1)^{b_k}\boldsymbol{P}_{c_k}, \end{aligned} \tag{1.1}$$

where $\boldsymbol{B} = [b_0, b_1, ..., b_{N-1}]$ with $b_k \in \{0, 1\}$ is a sequence with good auto-correlation properties. With this additional modulation, the phase of the the maximum values found at the output of the correlation block at the receiver side can be used to determine the exact window of arrival of the incoming frame. This preliminary idea will be discussed in details in section 2.2.

Before transmission, the generated frame $\boldsymbol{F}$ is composed of $N \times q$ BPSK symbols, which is then shaped by a half raised cosine filter with a roll-off factor equal to 0.35.

1.2 Channel model

We assumed the up-link channel where low cost sensors sporadically transmit small messages to the base station in an ALOHA protocol, i.e. without prior time and frequency synchronization at the receiver. The model of the channel is given at chip level in the sequel. The derivation between sample level and chip level is given in the appendix (Appendix 4.1). The QCSP message is thus received with unknown time offset t_0 (t_0 encompasses both the unknown time of emission and the unknown time of propagation between the emitter and the receiver). The message is also affected by an unknown (but limited) frequency offset $f_0 \in [-F_m, F_m]$ and an initial phase offset $\phi_0 \in [0, 2\pi[$.

$$\begin{aligned} y(n) &= e^{j(2\pi f_0 n + \phi_0)} \boldsymbol{F}(n - t_0) + z(n), \text{ if } n \in [\![t_0, t_0 + Nq - 1]\!] \\ &= z(n), \text{ otherwise} \end{aligned} \tag{1.2}$$

Without any prior information, f_0 and ϕ_0 are supposed to be uniformly distributed in their respective interval ranges.

One should note that a frequency offset f_0 generates qf_0 full rotations of the signal during the reception of each CCSK frames. The $z(n)$ are realisation of a complex white Gaussian Noise $\mathcal{CN}(\mu, \sigma)$, with mean $\mu = 0$ and standard deviation $\sigma = \sqrt{10^{\frac{-SNR}{10}}}$, where SNR represents the signal to noise ratio of the transmission. The whole received frame will be denoted $\boldsymbol{y}$

1.3 Receiver Side

Let us call $\mathcal{H}(t, f)$ the hypothesis "a frame is arrived at time t with a frequency offset f". According to D2.2a, this hypothesis can be tested by computing a score function that is proportional to the likelihood of this hypothesis and by comparing the score function to a given threshold. It should be noticed that, in this report, t indicates the index. It is thus necessary to receive $qN - 1$ more chips to get the whole frame before being able to evaluating the hypothesis $\mathcal{H}(t, f)$.

Let $s_t^f(\boldsymbol{y})$ be the score function associated to hypothesis $\mathcal{H}(t, f)$. The score defined as

$$s_t^f(\boldsymbol{y}) = \sum_{n=0}^{N-1} m_{t+kq}^f, \tag{1.3}$$

where

$$m_t^f = \max\{|L_t^f(i)|, i = 0, 1, \dots q - 1\}, \tag{1.4}$$

and L_t^f is the correlation vector between the received block $\boldsymbol{Y}_t^f$ defined as

$$\boldsymbol{Y}_t^f = (y(t + i)e^{-j2\pi f(t+i)})_{i=0,1,\dots,q-1}, \tag{1.5}$$

and the q CCSK symbols, i.e.

$$\boldsymbol{L}_t^f = \text{IFFT}(\text{FFT}(\boldsymbol{Y}_t^f)^* \odot \text{FFT}(\boldsymbol{P}_0)). \tag{1.6}$$

As shown in deliverable D2.2a, when the couple (t, f) is closed enough to the real time of arrival and frequency offset couple (t_0, f_0), then the hypothesis $\mathcal{H}(t, f)$ appears to be verified if the level of noise is not too high. Fig. 1.2 shows in 3D the values of $s_t^f(\boldsymbol{y})$ for a frame of length $N = 60$ without noise and with a CCSK sequence of length $q = 64$, affected by $t_0 = 20$ chips and $f_0 = 0.00875$ Hz. The maximum value of the score function $s_t^f(\boldsymbol{y})$ appears in this noiseless channel exactly for the hypothesis $\mathcal{H}(t = t_0, f = f_0)$ (cyan colored point).

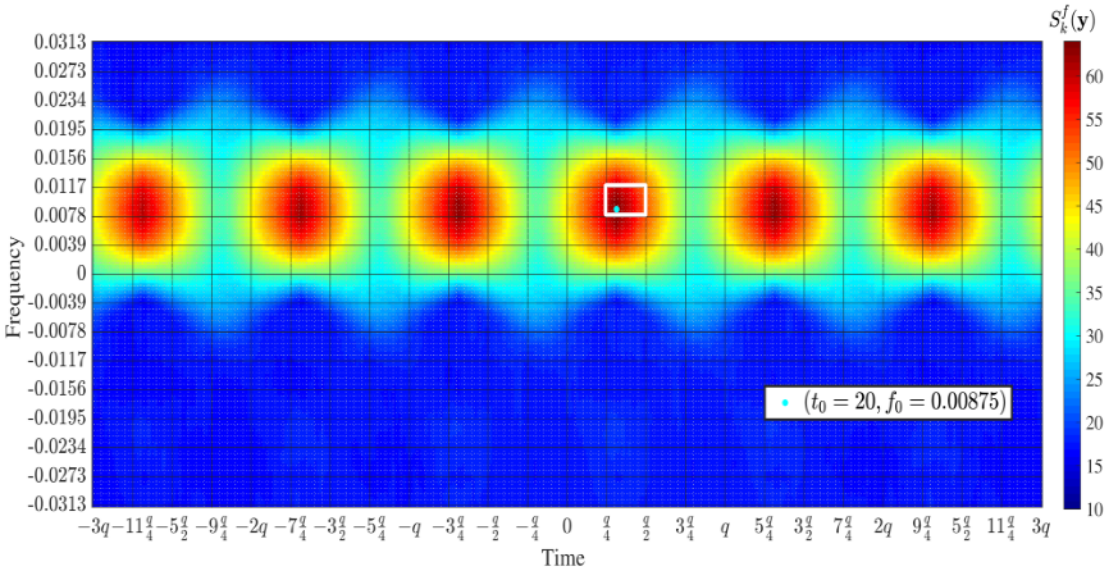

Figure 1.2: Values of $s_t^f(\boldsymbol{y})$ for an arriving QCSP frame in a noiseless channel

This property is first used by the detection algorithm to assess or not the arrival of a new frame with limited computational resource. According to deliverable D2.2, the time and the frequency space are split into bins of size $(\Delta_t, \Delta_f) = (q/4, 1/(4q))$, as shown in the black lines decomposition of the grid in Fig. 1.2. The score in the (τ, φ) bin, $\tau \in \mathbb{N}$, $\varphi \in [\![-F_m/\Delta_f, +F_m/\Delta_f]\!]$ corresponds to the score of hypothesis $\mathcal{H}(\tau\Delta_t, \varphi\Delta_f)$. Note that when a bin is triggered (i.e., the associated score function is above a threshold), some bins of the time and frequency neighbourhood can also be triggered. The (τ, φ) index information related to the bin with the highest score function is used to generate the coarse time synchronisation $t_c = \tau\Delta_t$ and coarse frequency synchronisation $f_c = \varphi\Delta_f$. The couple (t_c, f_c) (the bin delimited by the white rectangle) is thus sent to the synchronisation process. Since they may have some time or frequency ambiguity, the received samples in a window of size $2N$ around the estimated time of arrive t_c is sent to the synchronisation unit. More specifically, the samples $y(t_c - Nq/2), y(t_c - Nq/2 + 1), \ldots, y(t_c + qN + qN/2)$ are sent to the synchronisation unit.

1.4 Problem Statement

In a noisy channel, with very low SNR, the maximum value of the score function $s_t^f(\boldsymbol{y})$ will not simply matches the hypothesis $\mathcal{H}(t = t_0, f = f_0)$ which represents the synchronization parameters. Fig. 1.3 shows in 3D the values of $s_t^f(\boldsymbol{y})$ for the same previous frame of length $N = 60$ with a CCSK sequence of length $q = 64$ but at SNR $= -9.75$ Also it is affected by a delay $t_0 = 20$ chips and frequency offset $f_0 = 0.0875$ Hz. In this figure, t is limit to the interval $[\![t_0 - \frac{N}{2}q, t_0 + \frac{N}{2}q]\!]$ and f to the interval $[f_0 - 2/q, f_0 - 2/q]$. The figure is shown in different scales and from different angles of views to be more illustrative. In this example, the pink dot is the maximum value, and it is clear that it doesn't fit the real time and frequency shifts which is presented at the cyan dot. To be noted also, this maximum value could be one of different values of the peaks shown in the figure (the dark red). In order to alleviate that ambiguity, the synchronization block will be fed by a window of size $2N$ symbols around the estimated time of arrival as shown in the black rectangle in Fig. 1.3-(d).

For more illustration, Fig. 1.4 shows the same of the previous results of $s_t^f(\boldsymbol{y})$ but with a zoomed figure, with $t \in [\![t_0 - 3q, t_0 + 3q]\!]$ and $f \in [f_0 - \frac{1}{4q}, f_0 - \frac{1}{4q}]$, and Fig. 1.5 for $t \in [\![t_0 - q/2, t_0 + q/2]\!]$ and $f \in [f_0 - \frac{1}{4q}, f_0 - \frac{1}{4q}]$.

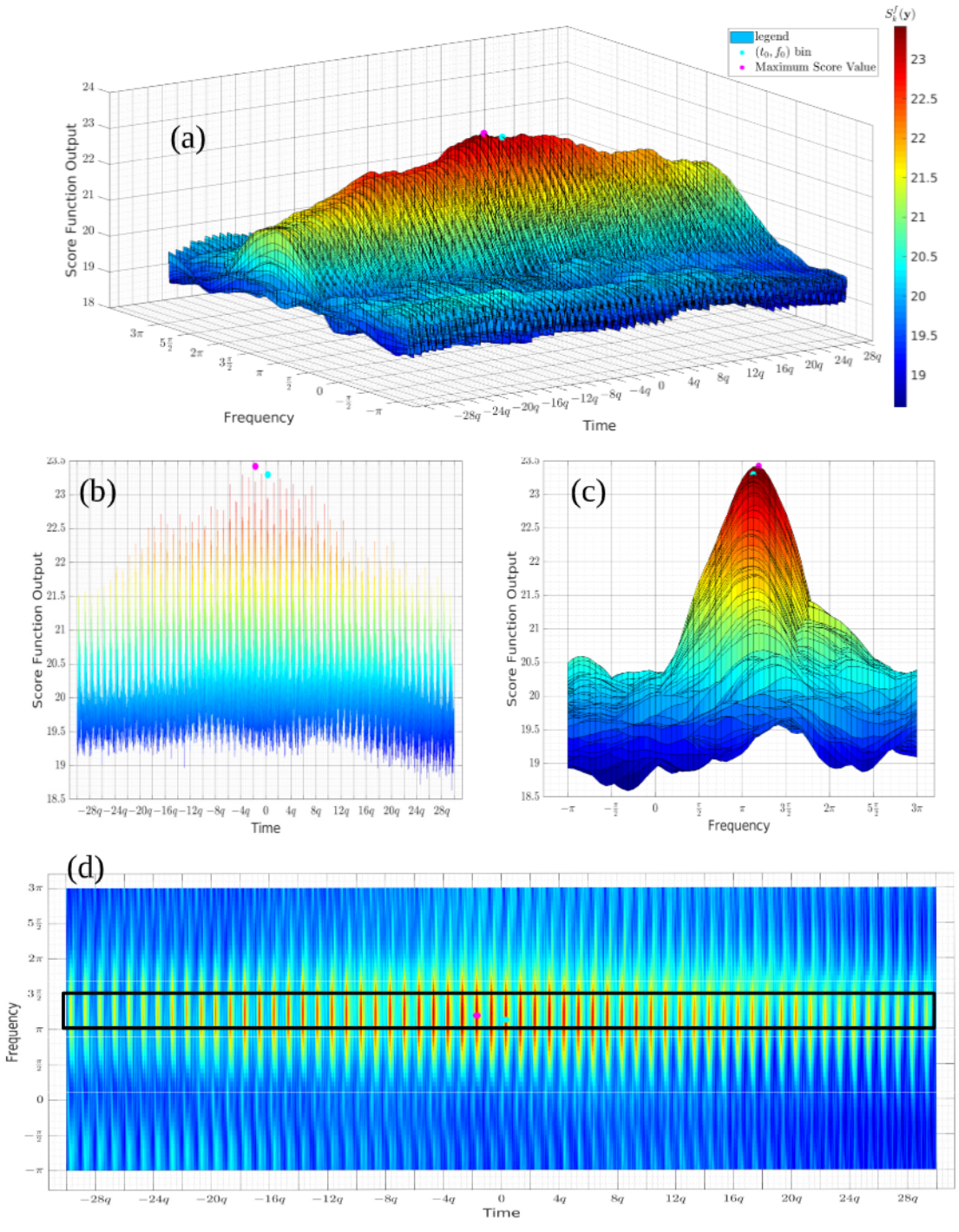

Figure 1.3: Values of $s_t^f(\boldsymbol{y})$ for an arriving QCSP frame in a very Low SNR channel

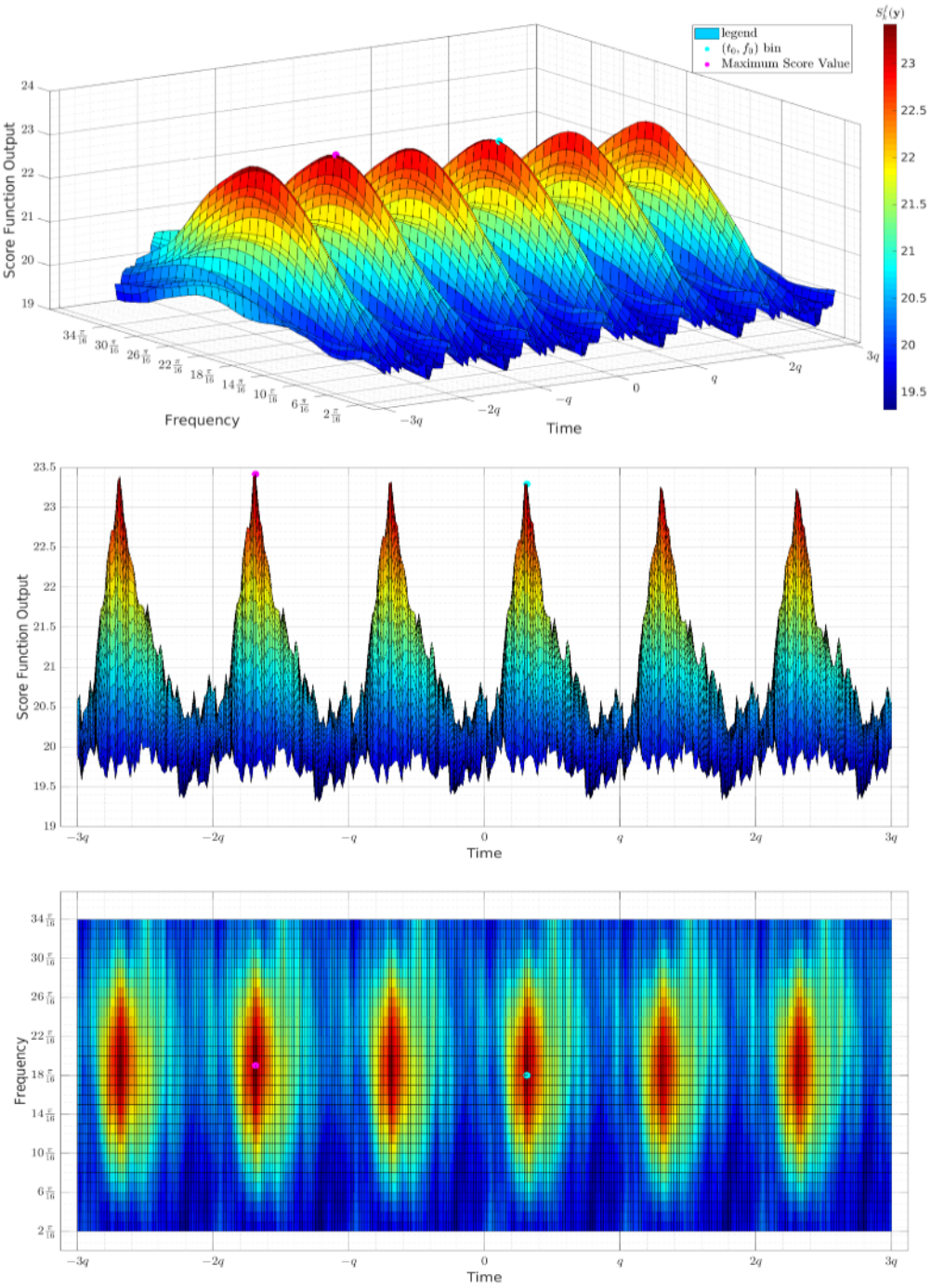

Figure 1.4: Values of $s_t^f(\boldsymbol{y})$ for an arriving QCSP frame in a very Low SNR channel, with $t \in [\![t_0 - 3q, t_0 + 3q]\!]$ and $f \in [f_0 - \frac{1}{4q}, f_0 - \frac{1}{4q}]$

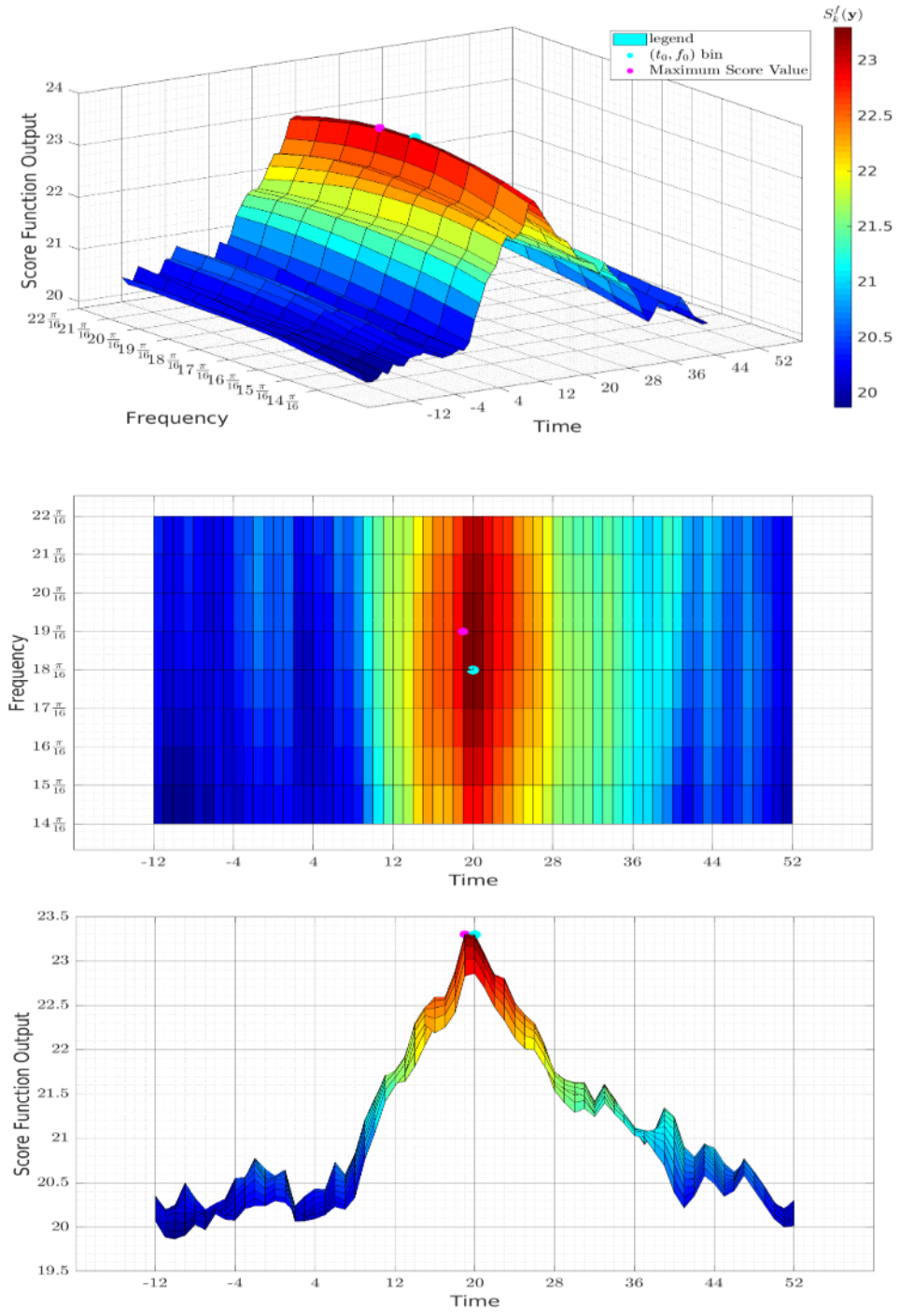

Figure 1.5: Values of $s_t^f(\boldsymbol{y})$ for an arriving QCSP frame in a very Low SNR channel, with $t \in [\![t_0 - q/2, t_0 + q/2]\!]$ and $f \in [f_0 - \frac{1}{4q}, f_0 - \frac{1}{4q}]$

2 Algorithm Specifications

This section presents the successive steps of the Blind Synchronization algorithm. The input of the algorithm is a set of $2N$ successive received samples that are assumed to contain a frame, as well as the coarse time offset t_c and the coarse frequency offset f_c. The output of the algorithm is the updated estimation of the correct time of arrival $\bar{t}_0$, the frequency offset $\bar{f}_0$ and the initial phase $\bar{\phi}_0$. Then, from the estimated parameters $(\bar{t}_0, \bar{f}_0, \bar{\phi}_0)$, a coherent demodulation is performed to generate the Log Likelihood Ratio (LLR) that are sent to the NB decoder.

The synchronization process is divided into several steps (or tasks) performed sequentially. Each step adds some additional knowledge that allows a coherent demodulation of the received frame. Those steps are:

1. Coarse modulo q chip synchronization (from t_c to $\hat{t}_0$) to reduce the error on chip synchronization down to few unit modulo q.
2. Symbol synchronization (from $\hat{t}_0$ to $\tilde{t}_0$) to reduce the error on chip synchronization to few unit thanks to the over-modulation sequence.
3. Fine frequency synchronization (from f_c to $\hat{f}_0$, then to $\bar{f}_0$) to suppress the remaining frequency offset error
4. Exact chip synchronization (from $\tilde{t}_0$ to $\bar{t}_0$) using the Non Binary code properties.
5. Phase mitigation (estimation of $\bar{\phi}_0$), to estimate the initial phase error ϕ.
6. LLR computation thanks to the coherent demodulation based on parameters $(\bar{t}_0, \bar{f}_0, \bar{\phi}_0)$.

Figure 2.1 shows the different steps of the successful synchronization process, from the initial coarse time/frequency estimation (t_c, f_c) to the final correct time/frequency estimation $(\bar{t}_0, \bar{f}_0) = (t_0, f_0)$.

Note that some items of the above list can be permuted or eventually suppressed. Our objective here is to indicate and illustrate a first effective synchronization method that can be optimized later. So far, there is no guaranty that the method is optimal, neither in terms of performance, nor in terms of complexity.

To illustrate the different phases of the synchronization process, performance results are given at each step of the synchronization process, after transmitting 10,000 frames of length $m = 120$ bit of information, a NB-Code of code rate 1/3 over GF(64) (i.e. 60 CCSK symbols). The frames are transmitted in an AWGN channel with an SNR of -9.75 dB, a delay t_0 randomly selected between $-q/2$ and $q/2$ and a frequency offset f_o with uniform distribution between $[-F_m, F_m]$.

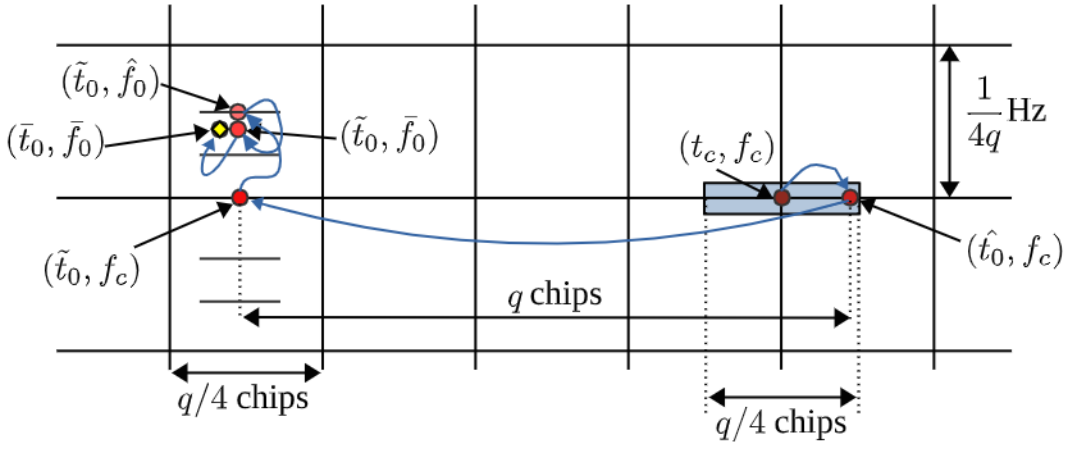

Figure 2.1: Example of a successful synchronization process from step 1 to step 4

2.1 Step 1: Coarse modulo q chip Synchronization

The detection algorithm performs the time detection with a granularity of $q/4$ chips. The coarse chip synchronization perform the same operation but with a granularity of one chip in a time window $[\![t_c - q/8, t_c + q/8]\!]$ around chip index t_c.

$$\hat{t}_0 = \arg\max\{s_t^{f_c}(\boldsymbol{y}), t \in [\![t_c - q/8, t_c + q/8]\!]\}; \tag{2.1}$$

with f_c, the coarse frequency estimation equal to $f_c = \varphi/(4q)$.

Figure 2.2 shows the histogram of the chip synchronization error $\hat{e}_0 = t_0 - \hat{t}_0$ obtained after transmitting 10,000 of the predefined QCSP frame.

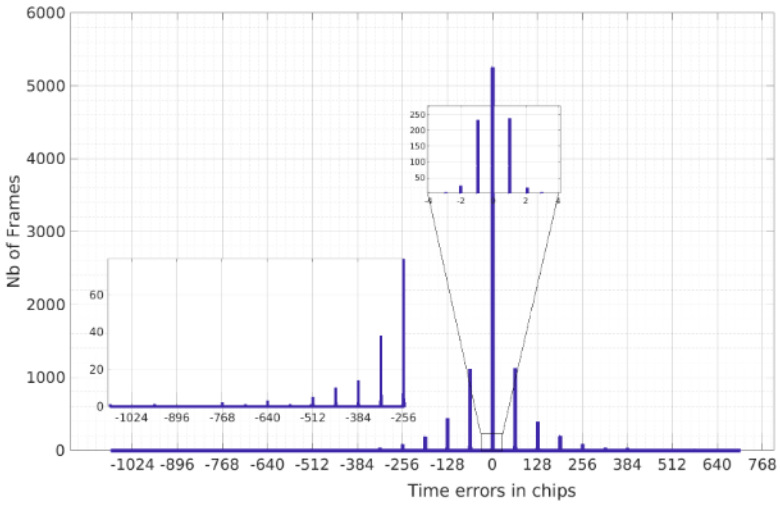

Figure 2.2: Histogram of coarse chip synchronization error with 10,000 transmitted frames

We can figure out from the first step of synchronization that we have more than 4,000 frames out of 10,000 that are not correctly synchronised at the chip level. By analysing the probability distribution in the figure we can distinguish that there exist two different types of errors. First we have the chip errors that are roughly multiple of q chips (i.e. the the size of a CCSK symbol in the QCSP frame). synchronization error of up to 17 CCSK symbols are observed in this simulation. The second type of chip synchronization error are small chip offset (between -4 and 4 chips, typically) around each q peak, which are occurring at indices multiple of q (i.e., sq with s is an integer $\in [\![-N/2, N/2]\!]$). From the first type of time synchronization errors, we can deduce that there exist errors at the symbol level and that the coarse time estimation is not sufficient by it own to make correct symbol synchronization at this very low SNR. The symbol ambiguity is solved in the second step of the synchronization process by taking profit of the over modulation.

2.2 Step 2: Symbol synchronization

In a very noisy environment, symbol over-modulation is used to help correcting the symbol ambiguity at the receiver side. This is based on the over-modulation given in equation (1.1) using the phase information variation given by the binary sequence $\boldsymbol{B}$. In this section, we will neglect the small chip synchronization error and focus on the errors at the symbol level. Thus, we assume that $\hat{t}_0 = t_0 + s.q$, with $s \in [\![-N/2, N/2]\!]$.

Going back to (1.4), it is possible to add also the phase information to the maximum correlation value $m_t^{f_c}$. By defining $d_t^{f_c}$ as

$$d_t^{f_c} = \arg\max\{|L_t^{f_c}(i)|, i = 0, 1, \dots q-1\}, \tag{2.2}$$

we can defined $\gamma_t^{f_c}$ as $\gamma_t^{f_c} = L_t^{f_c}(d_t^{f_c})$. One should note that, by construction, $m_t^f = |\gamma_t^f|$. Let us determine the exact phase information of $\gamma_{t_0+kq}^{f_c}$ assuming no decoding error. The value of $\gamma_{t_0+kq}^{f_c}$ corresponds the scalar product between $\boldsymbol{Y}_{t_0+kq}^{f_c}$ and P_{c_n}, where $\boldsymbol{Y}_{t_0+kq}^{f_c} = ((-1)^{b_k} P_{c_k}(l) e^{j(2\pi(f_0-f_c)(t_0+kq+i)+\phi_0)} + z(t_0+kq+i))_{i=0,1,\dots,q-1}$. Thus,

$$\begin{aligned}
\gamma_{t_0+kq}^{f_c} &= (-1)^{b_k} e^{j(2\pi(f_0-f_c)(t_0+kq)+\phi_0)} \sum_{i=0}^{q-1} e^{j2\pi(f_0-f_c)i} + Z \\
&= (-1)^{b_k} e^{j(2\pi(f_0-f_c)(t_0+kq)+\phi_0)} \frac{1-e^{j2\pi(f_0-f_c)q}}{1-e^{j2\pi(f_0-f_c)}} + Z_k, \\
&= (-1)^{b_k} e^{j\Phi_0} e^{(2\pi(f_0-f_c)qn)} \frac{\sin(2\pi(f_0-f_c)q)}{\sin(2\pi(f_0-f_c))} + Z_i, \\
&= (-1)^{b_k} e^{j\Phi_0} e^{j\theta k} A + Z_n,
\end{aligned} \tag{2.3}$$

with $\Phi_0 = 2\pi(f_0-f_c)(t_0+(q-1)/2)+\phi_0$, $\theta = 2\pi(f_0-f_c)q$, $A = \frac{\sin(2\pi(f_0-f_c)q)}{\sin(2\pi(f_0-f_c))}$ and Z_n the summation of q independent samples of a complex Gaussian noise of standard deviation σ, i.e., a complex Gaussian noise of standard deviation $\sqrt{q}\sigma$. For simplicity shake, the f_c exponent is omitted in the rest of this section.

Since by hypothesis $\hat{t}_0 = t_0 + sq$, with s taking integer value between $-N/2$ and $N/2$ (not outside -17 and 17 in practice over the 10,000 simulated QCSP frames). Thus, $\gamma_{\hat{t}_0+kq} = \gamma_{t_0+(k+s)q}$ with the convention that $\gamma_{\hat{t}_0+kq}^{f_c} = Z_{k+s}$ only when $(k+s)$ is outside the range $[\![0, N-1]\!]$. We can thus defined the vector $\boldsymbol{\Gamma}_{\hat{t}_0}(s) = (\gamma_{\hat{t}_0+(k-s)q})_{k=0,\dots,N-1}$. Note that by construction, $\boldsymbol{\Gamma}_{\hat{t}_0}(\hat{s}) = \boldsymbol{\Gamma}_{t_0}(0)$. Knowing that $\boldsymbol{B} = ((-1)^{b_0}, (-1)^{b_1}, \dots, (-1)^{b_{N-1}})$ is the over-modulation sequence, we can note that $\boldsymbol{\Gamma}_{\hat{t}_0}(\hat{s}) \odot \boldsymbol{B}$ is a pure sinusoidal corrupted by noise. In fact, $\boldsymbol{\Gamma}_{\hat{t}_0}(\hat{s}) \odot \boldsymbol{B} = \boldsymbol{\Gamma}_{t_0}(0) \odot \boldsymbol{B} = (\gamma_{t_0+kq}^{f_c} \times (-1)^{b_k})_{k=0,1,\dots,N-1}$. Thus, according to (2.3),

$$\boldsymbol{\Gamma}_{t_0}(0) \odot \boldsymbol{B} = (e^{j\Phi_0} e^{j\theta k} A + (-1)^{b_k} Z_k)_{k=0,1,\dots,N-1}. \tag{2.4}$$

When $(\hat{t}_0 - t_0)/q \neq s$, the effect of the over-modulation is not suppressed since the value of the k^{th} component of $\boldsymbol{\Gamma}_{\hat{t}_0}(s) \odot \boldsymbol{B}$ is equal to thus $(-1)^{b_k+b_{k+s}} e^{j\Phi_0} e^{j\theta(k+s)} + Z_{k+s}(-1)^{b_k}$. In others words, the phase between two consecutive components of $\boldsymbol{\Gamma}_{\hat{t}_0}(s) \odot \boldsymbol{B}$ varies between θ or $\theta+\pi$, which means that the regular sinusoidal pattern is thus destroyed. In this Deliverable, we propose simply to decide to select the value s that maximizes the maximum module of the Fast Fourier Transform of the $\boldsymbol{\Gamma}_{\hat{t}_0}(s) \odot \boldsymbol{B}$ vector. In summary, the ambiguity on symbol position is solved by taking $\hat{s}$ as

$$\hat{s} = \arg\max\{\max\{|\mathrm{FFT}(\boldsymbol{\Gamma}_{\hat{t}_0+sq})|\}, s \in [\![-N/2, N/2]\!]\} \tag{2.5}$$

Figure 2.3 illustrates the use of equation (2.5) by applying it over 4 independent received QCSP frames defined in the introduction of this chapter. For-example the curve with blue color is perfectly symbol synchronized, while the other cases needs to be synchronized as mention in the figure, i.e., the black one with $\hat{s} = -8$, violet with $\hat{s} = 1$, orange with $\hat{s} = 6$ symbols and blue with $\hat{s} = 0$.

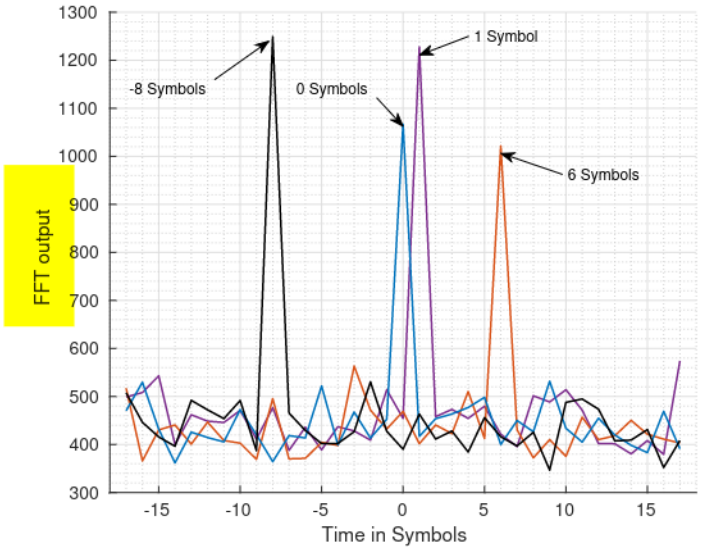

Figure 2.3: Illustration of equation (2.5) over 4 different cases of the detection of predefined QCSP frame

From $\hat{s}$, the estimated of the time of arrival of the frame is updated from $\hat{t}_0$ to $\bar{t}_0 = \hat{t}_0 - \hat{s}q$. Figure 2.4 shows the probability of distribution of chip synchronization error $\bar{e} = t_0 - \bar{t}_0$ over 10,000 frames of the predefined QCSP frame after applying the second step in the synchronization process, i.e. Symbol adaptation. The analysis of Fig. 2.4 shows that taking profit of the over-modulation allows to mitigate the time offset at the symbol level. At this end of the second step of the synchronization process, more than 90 % of the frame are correctly synchronised in time. The remaining 10 % suffers from a low amplitude error synchronization, with $\bar{e} \in [-4, 4]$. As mentioned before, this chip offsets will be solved by the help of the parity checks from the NB-Coding. This step will be presented and discussed in details in step 5 of the synchronization process.

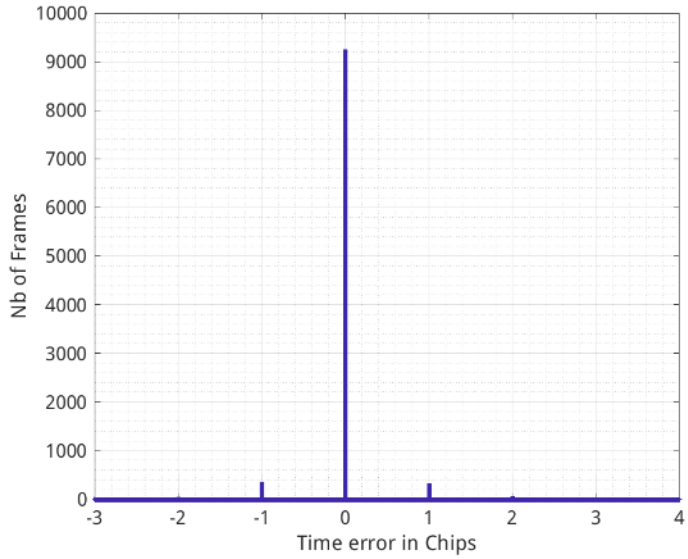

Figure 2.4: Probability of distribution of chip synchronization error $\hat{r}$ obtain with the transmission of 10,000 frames after step 1 and 2 of the synchronization process

To conclude, two hypothesis have been assumed to justify the mathematical model used to determine the parameter $\hat{s}$. The first hypothesis was the fact that $\hat{t}_0 = t_0 + \hat{s}q$, while in fact $\hat{t}_0 = t_0 + \hat{s}q + \hat{r}$,

with $\hat{r}$ having the distribution given in Fig. 2.4. The second hypothesis is not explicitly formulated but, due to the channel noise, not all the values of d_{t_0+nq}, are correctly estimated, leading to value of γ_{t_0+kq} different than the expression given in (2.3). Nevertheless, the simulation shows that, even if the hypothesis used to justify the mathematical model are not fully exact, the method remains efficient in practice.

2.3 Step 3: Fine Frequency Synchronization

At this stage of the synchronization process, the position of the first chip of the frame is known with a good accuracy. It is now time to refine the coarse frequency estimation f_c given by the detection algorithm. The first step of this method is just to perform the search of the optimal frequency with a grid of size $\frac{1}{32q}$ instead of $\frac{1}{4q}$ (It is worth noticing here that $\frac{1}{32q}$ can be decreased or increased based on the optimization between complexity and performance). Let $\tilde{t}_0 = \hat{t}_0 + \hat{s}q$, then $\hat{f}_0$ is estimated, according to (1.3) as

$$\hat{f}_0 = \arg\max\{s^f_{\tilde{t}_0}(\boldsymbol{y}), f \in \{f_c + \frac{t}{32q}\}_{t=-8,\ldots,8}\} \tag{2.6}$$

Figure 2.5 shows the distribution of the frequency error estimation $f_0 - \hat{f}_0$ over 10,000 frames same as the predefined QCSP frame.

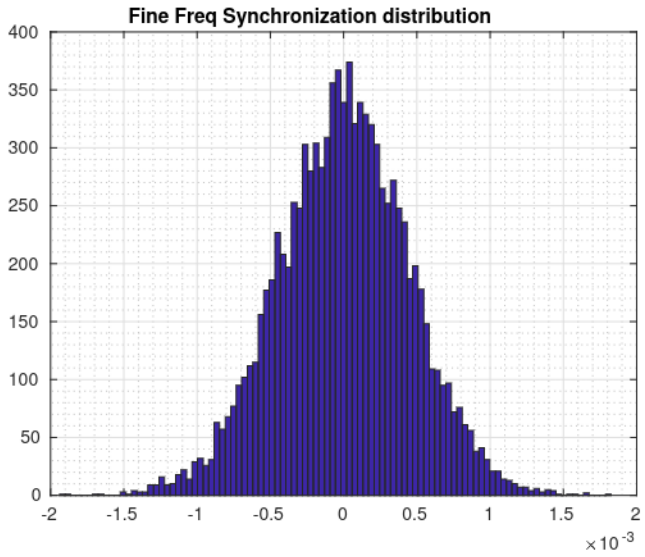

Figure 2.5: Distribution of frequency error estimation $f_0 - \hat{f}_0$ over 10,000 transmitted frames

We can figure out from Figure 2.5 that, in the majority of the cases, the frequency estimation error lies between $\pm 10^{-3}$. It correspond roughly to a full rotation every 8 CCSK symbols ($1/(q8) = 0.97 \times 10^{-4}$). This residual frequency offset is by far too high for a coherent demodulation, which show the limit of globally non-coherent frequency estimation. In fact, the absolute value of the scored function m^f_{t+kq} (locally coherent detection at symbol level) are summed in (1.3) (incoherent accumulation at frame level). In order to improve the estimation of f_0, coherent estimation on the whole frame is required.

2.4 Step 4: Coherent frequency estimation

The principle of coherent frequency estimation is based on (2.4). that is recalled here by replacing (f_c, t_0) by $(\hat{f}_0, \bar{t}_0)$.

$$\mathbf{\Gamma}_{\bar{t}_0}^{\hat{f}_0} \odot \boldsymbol{B} = (e^{j\Phi_0} e^{j\hat{\theta}k} A + (-1)^{b_k} Z_k)_{k=0,1,\ldots,N-1}. \tag{2.7}$$

With $\hat{\theta} = 2\pi(f_0 - \hat{f}_0)q$

This is a classical problem of pure frequency estimation. The near Maximum Likelihood estimator is given in Appendix 4.2 is proposed to perform the frequency estimation. The output will be called $\bar{f}_0$.

Figure 2.6 gives the histogram of the frequency error estimation $\bar{f}_0 - f_0$ based on the transmission of 10,000 predefined QCSP frames.

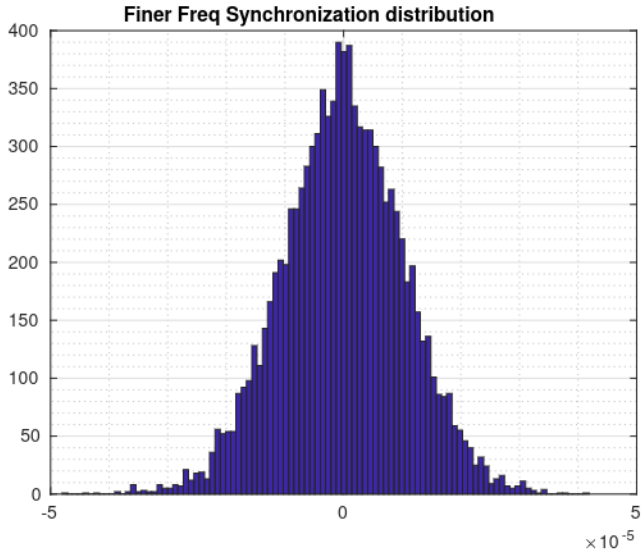

Figure 2.6: Histogram of the final frequency estimation error based on the transmission of 10,000

.

2.5 Step 5: Fine Chip Synchronization using Syndrome

The objective of the this synchronization stage is to solve the remained asynchronous frames due to chip estimation error $\hat{e} = t_0 - \bar{t}_0$. Again, to lighten notation, the exponent $\hat{f}_0$ is omitted in this section.

For each hypothesis of the value of $\hat{r}$, we generate from (2.2) the decision vector $D_{\bar{t}_0+e}$ defined as

$$D_{\bar{t}_0+e} = (d_{\bar{t}_0+e+kq})_{k=0,1,\ldots,N-1} \tag{2.8}$$

Since the CCSK symbols of the frame are linked by the NB code, it is possible to use this side information to determine the value of $\hat{e}$. Let $\boldsymbol{H}$ be the parity check matrix, then in case of correct synchronization without any decision error, the vector D_{t_0} verify all NB parity check equations of $\boldsymbol{H}$, i.e., $\boldsymbol{H}D_{t_0}$ is equal the null vector. Equivalently, the number of parity check satisfied of the parity check code is equal to the number of parity check of the matrix. If the vector $\boldsymbol{H}D_{t_0}$ contains some error, the number of satisfied parity checks decreases but should remain high enough to allow a successfully decoding processes by Non Binary decoder. In the contrary, with a synchronization error, the probability that a parity check is satisfied should be $1/q$. Since there is $\frac{2}{3N}$ parity checks on a GF(q) NB code of length N and coding rate 1/3, the total number of satisfied parity checks follows a Poison law of mean $\lambda = \frac{2N}{3q}$. In others word, when $e \neq \hat{e}$, the number of satisfied parity checks of $\boldsymbol{H}$ with vector $D_{\bar{t}_0+r}$ is very low (the mean value of a Poisson process is λ, i.e., $\frac{2\times 60}{3\times 64} = 0.625$ for the considered QCSP frames). Let $\mathcal{N}(x)$ be the function that returns the number of null element in vector x. The selection of $\hat{r}$ is performed as

$$\hat{r} = \arg\max\{\mathcal{N}(\boldsymbol{H}D_{\bar{t}_0+r}), r \in [\![-q/4, q/4]\!]\} \tag{2.9}$$

Figure 2.7 illustrates the use of equation (2.9) by applying it over 5 predefined QCSP frames including two frames with ±3 chip errors and 3 perfectly synchronized. We can see blue colored plot is perfectly symbol synchronized, while the other cases needs to be synchronized as mention in the figure, i.e., the green one with $\hat{r} = 3$ and orange with $\hat{r} = -3$ symbols.

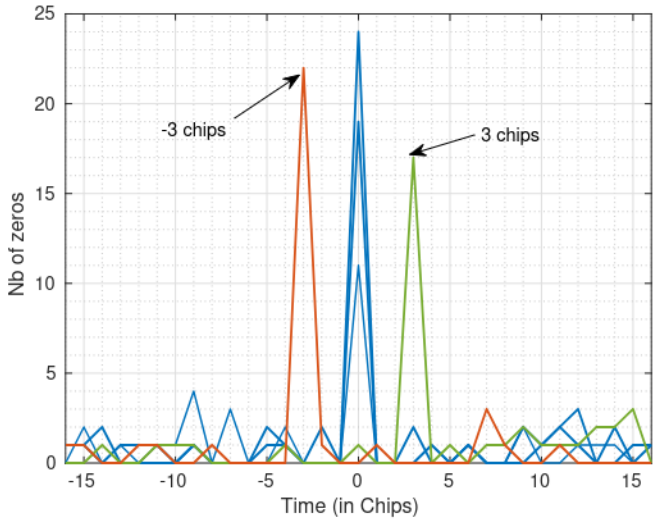

Figure 2.7: Illustration of equation (2.9) over 5 different cases

After the final step in synchronization process, we plot the probability of distribution of frame chip miss synchronization for the transmission of 10,000 frames of the predefined QCSP frame in Figure 2.8. We can notice that all the frames are perfectly synchronized now, at SNR = -9.75 dB, ant the blind synchronization algorithm is achieved.

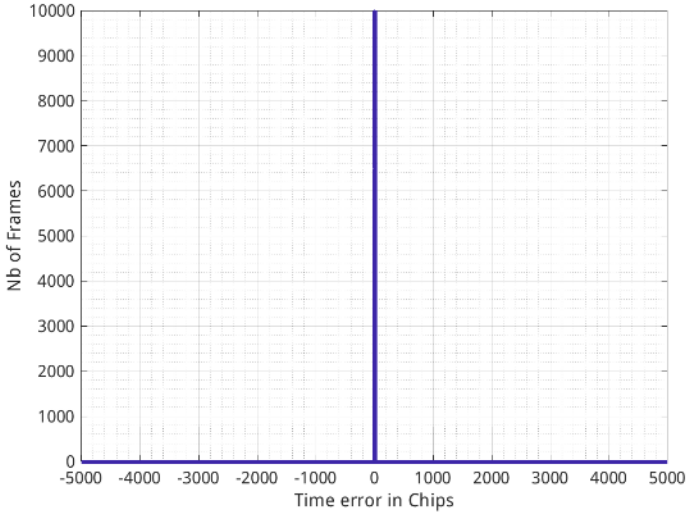

Figure 2.8: Probability of distribution of time miss synchronization for the transmission of 10000 frames after the Synchronization process

2.6 Step 6: Phase mitigation and LLR computation

This section starts by describing the phase estimation algorithm. Then the method to generate LLR is given. Finally, some theoretical study on performance degradation due to phase estimation error is presented.

2.6.1 Phase estimation algorithm

Let us assume that t_0 is accurately determined and that $\bar{f}_0$ is closed enough to f_0 so that the different is negligible, then from (2.7), we obtain,

$$\mathbf{\Gamma}_{t_0}^{f_0} \odot \boldsymbol{B} = (e^{j\Phi_0} A + (-1)^{b_k} Z_k)_{k=0,1,\ldots,N-1}. \tag{2.10}$$

Then, a good estimator of Φ_0 is given as:

$$\bar{\Phi}_0 = \text{Phase}\left(\sum_{k=0}^{N-1} \gamma_{t_0+kq}^{f_0}\right). \tag{2.11}$$

2.6.2 LLR computation

Once $\bar{\Phi}_0$ estimated, its impact should be mitigated by replacing the computation of $L_{t_0+kq}^{f_0}(i)$, $i = 0, 1, \ldots, q-1$ and $k = 0, 1, \ldots, N-1$ given in (1.6) by

$$\bar{L}_k(i) = \mathcal{R}(e^{-j\bar{\Phi}_0} L_{t_0+kq}^{\bar{f}_0}(i)). \tag{2.12}$$

where $\mathcal{R}(x)$ represents the real part of complex x. Then, from (2.12) the LLR sent to the decoder are normalized as:

$$L_k(i) = \frac{1}{\sigma^2}(\bar{m}_k - \bar{L}_k(i)) \tag{2.13}$$

with $\bar{m}_k$ defined as the maximum value of the vector $\bar{\boldsymbol{L}}_k$, i.e.,

$$\bar{m}_k = \max\{\bar{L}_k(i), i = 0, 1, \ldots, q-1\} \tag{2.14}$$

Once the LLR computed, the decoding process can start. The following section study the impact of phase estimation on the performance.

2.6.3 SNR degradation due to phase estimation errors

One should note that a phase estimation error of $e_\Phi = \Phi_0 - \bar{\Phi}_0$ generates a degradation of the signal amplitude by $\cos(e_\Phi)$ without affecting the amplitude of the noise. It thus generates a $10\log_{10}(\cos(e_\Phi))$ dB degradation of the SNR.

To illustrate the impact of phase estimation error, Fig. 2.9 shows the degradation of SNR as a function of angle θ in radian.

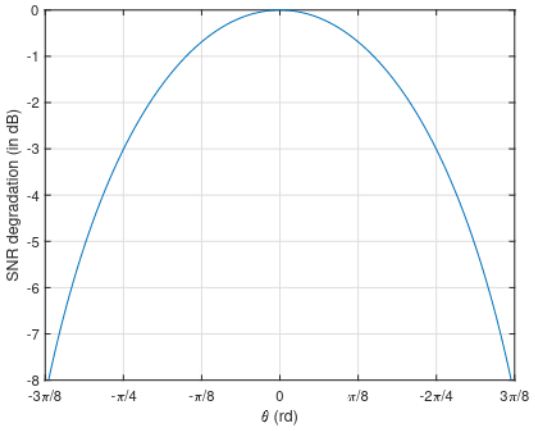

Figure 2.9: Degradation of SNR (in dB) due to phase estimation error e_Φ.

It is also possible to express the angle θ as the function of SNR, $e_\Phi(\text{snr}) = \arccos\sqrt{10^{\text{SNR}/10}}$, i.e., to determine the angle required for a given SNR degradation, as shown in Fig. 2.10.

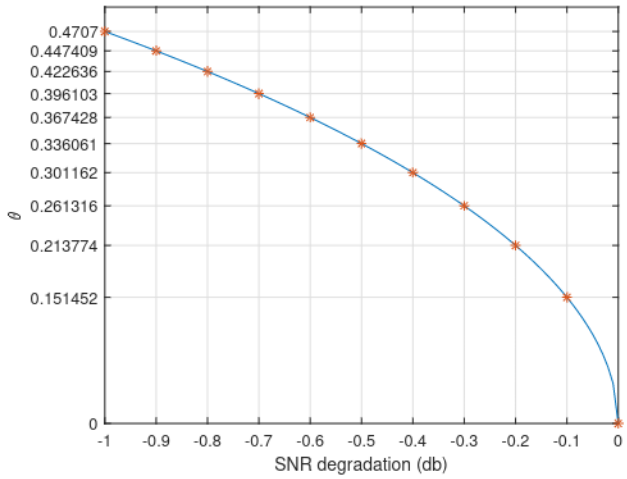

Figure 2.10: Error phase required for a given level of SNR degradation.

From Fig. 2.10, we can deduce that an angle estimation error between $[-0.0482\pi, 0.0482\pi]$ generates a SNR degradation bellow 0.1 dB and that an angle estimation error above 0.1498π (in absolute value) generates a degradation greater than 1 dB.

The Matlab code given in Appendix 4.3 estimates through Monte-Carlo simulation the probability that the degradation in SNR introduced by the phase estimation error is greater than 0.1 dB, 0.5 dB, 1 dB and 3 dB, respectively. Fig. 2.11 gives the result obtained with 100,000 transmission of the reference QCSP frame. Two types of curves are given. The one using directly (2.11), in plain line and the one "Genius Aided" (GA, dotted line) where only correct decision are taken into account. One can note that GA phase estimation is far more reliable than direct phase estimation. The GA case may

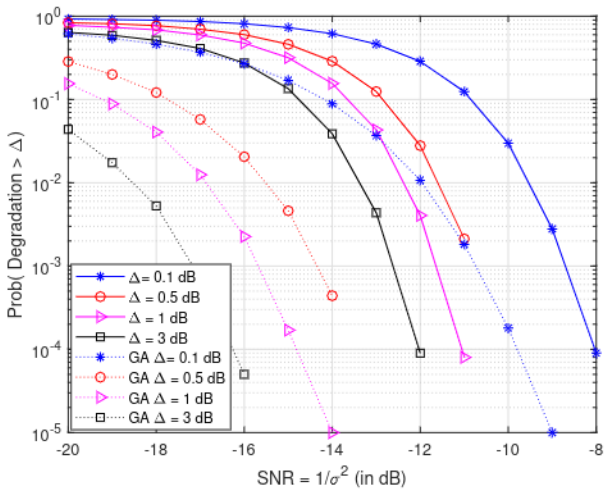

Figure 2.11: Probability to have a SNR degradation above a given level as a function of the channel SNR'.

be approached using the parity check of the code and giving a weight to the $\gamma^{f_0}_{t_0+kq}$ values of (2.11) according to the number of the parity check fulfilled by the k^{th} variable of the code decision vector $D^{f_0}_{t_0}$ defined in (2.8). This point will be studied in the sequel of the project.

3 Results and Conclusion

In this section, we show the probabilities of miss reception at the receiver of a QCSP system in an AWGN channel and without any prior information about time or frequency offsets. In Fig. 3.1, the probability of miss detection $\mathcal{P}_{md}$, miss synchronization $\mathcal{P}_{ms}$ and the joint probability of miss detection and miss synchronization $\mathcal{P}_e$ are shown using the proposed blind detection and synchronization algorithm over a QCSP frame of the following parameter, $m = 120$ bits, $R_c = 1/3$ and $q = 64$ chips, where the time and frequency shifts are uniform randomly distributed. As can be seen, $\mathcal{P}_e$ is directly effected by each of detection and synchronization performance, and mainly impacted by $\mathcal{P}_{md}$ at SNRs bellow -10 dB, then by $\mathcal{P}_{ms}$ at SNRs higher than -10 dB. Also, starting from SNR = -9.75, the detector and synchronization block will fed the decoder with frames almost perfectly detected and synchronized ($\mathcal{P}_e = 10^{-4}$) for a QCSP frame of the previous defined parameter.

As a practical case study, probability of error correction $\mathcal{P}_\epsilon$ for GF(64)-LDPC code of rate 1/3, $m = 120$ bits is found as in defined in [1]. Two types of decoding algorithm are used: the Belief Propagation (BP) [2] with 50 decoding iterations and the Extended Mean Sum (EMS) with 30 decoding iterations and $n_m = 20$ (see [3] for the definition of the EMS algorithm). The resulting probabilities of error $\mathcal{P}_\epsilon$ with a perfect synchronization are given in Fig. 3.1.

Finally we can conclude that blind detection and synchronization algorithm without the use of the preamble can be achieved and put on the road for short packets wireless transmission.

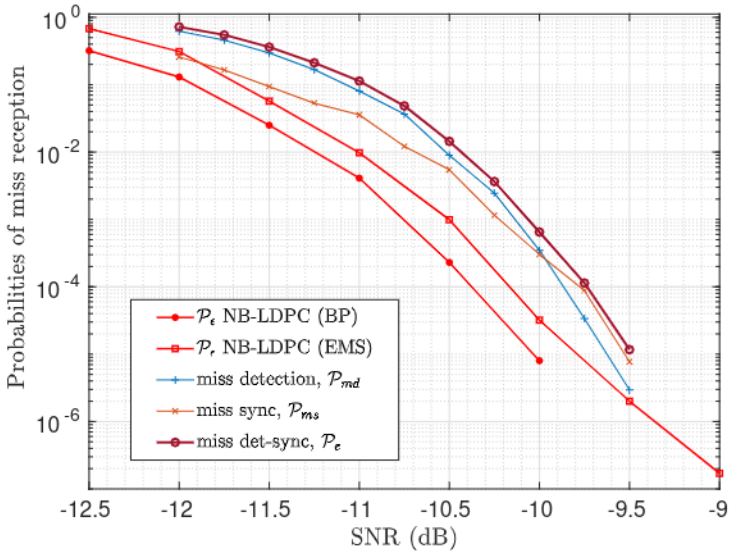

Figure 3.1: Probabilities of miss reception as function of SNR for random asynchronous QCSP frame

4 Appendices and Matlab Code

4.1 Sample versus Chip synchronisation

4.1.1 System model

Let T_c and T (in second) be the duration of a chip and a CCSK symbol respectively, such that $T = q \times T_c$. The receiver will over-sample the incoming signal with O samples per chip. In other words, the clock frequency F_e of the receiver Analog Digital Converter (ADC) is equal to $F_e = O/T_c$, with O the over-sampling factor (typically between 4 up to 8). Indexing the time by duration T_c of a chip (i.e. O clock cycles), it is possible to determine the time of arrival t_a as a real $x_a = t_a/T_c$ and by decomposing x_a as

$$x_a = k_a + r_a/O + \epsilon_a, \tag{4.1}$$

where $k_a = \lfloor x_a \rfloor$, the integer part of x_a represent the time in number of chips, r_a the closest index of the clock cycle within a chip ($r_a \in \{0, 1, \ldots, O-1\}$) and ϵ_a is the residual timing synchronization error (with $\epsilon_a \in [-\frac{1}{2O}, \frac{1}{2O}]$).

In the sequel, it is considered that the oversampling factor is high enough so ϵ_a is negligible and can be considered equal to 0. Let us consider the two alternatives to suppress the sample level ambiguity.

4.1.2 Parallel filters

Moreover, we will also assume that by testing in parallel all the O hypothesis of the r_a value, we can always manage to set r_a equals to 0. In summary, the frame will be received at chip index k_a and affected by frequency offset f_o.

All the developments presented in the report are based on this method with the best of the O hypothesis.

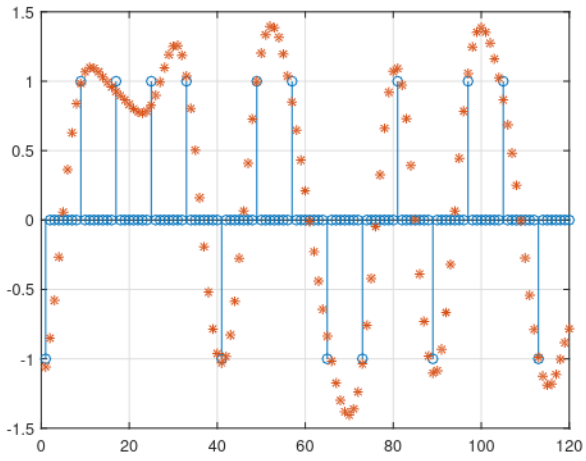

Figure 4.1: Example of a sequence of chip and the corresponding sample with $O = 8$

4.1.3 Single extended filter

It is also possible to perform the time analysis on a sample level, instead of a chip level. In that case, both $\boldsymbol{P}_0$ and $\boldsymbol{Y}_t^f$ are vectors of size Oq. The vector $\boldsymbol{P}_0$ thus include the effect of the cosine rise filter at the emitter level. Fig. 4.1 shows a sequence of chips and the corresponding samples for $O = 8$

Fig. 4.2 shows a value of $S^{f_0}_{y_0+s/O}$ with s the index of the sample around the optimal time t_0. It is clear from this figure that a regular amplitude oscillation of length $O = 8$ appears. The maximum values of the oscillation are spaced of 8 samples. It corresponds to the optimal time synchronisation.

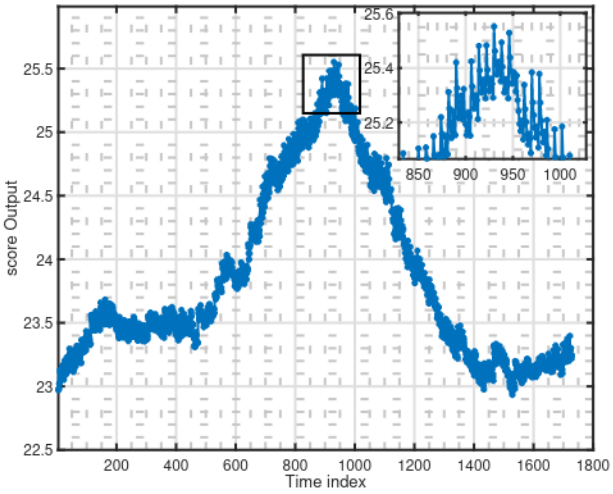

Figure 4.2:

4.2 Frequency estimation

Frequency estimation of the a pure sinusoidal signal corrupted with Additive White Gaussian Noise has been studied since the seventies. Assuming N received sampled, the proposed method is based firstly on a Fast Fourier Transform (FFT), and secondly, on the search of the index that maximizes the module of the FFT, and thirdly, on an novel post-processing of the FFT output using non-linear interpolation to obtain an accurate estimate of the frequency. The proposed method dramatically outperforms all state of the art solutions of the same complexity for a large set of useful configurations ($N \geq 32$, large range of SNR).

4.2.1 Introduction

The estimation of the frequency of a pure sinusoidal signal corrupted by Additive White Gaussian Noise (AWGN) is a very basic problem that has been studied for decades [4, 5]. Let us assume that N samples are available to estimate the frequency f_d from vector $\mathbf{r} = \{r(n)\}_{n=0..N-1}$, with

$$r\left(n\right) = e^{j(2\pi f_d n)} + w(n), \tag{4.2}$$

where n is the time index, f_d is the unknown frequency and $w(n)$ is a complex AWGN of variance σ^2. The f_d frequency can be assumed to be a random variable between -0.5 and 0.5, or being close to zero (for example, a residual fine frequency offset in a communication system).

The Cramer Rao Bound (CRB) defines the minimum standard deviation error σ_B of the best achievable estimator. It is given by

$$\sigma_B = \sqrt{\frac{6\sigma^2}{2\pi^2 N(N^2-1)}}. \tag{4.3}$$

The Maximum Likelihood frequency estimation is given by:

$$\bar{f}_d = \arg\max \left\{ \left| \sum_{n=0}^{N-1} r(n) e^{-2\pi n f} \right|^2, f \in [-0.5, 0.5] \right\}. \tag{4.4}$$

Fitz in [6] proposes an estimator based on the derivative of the periodogram (4.4) and setting it to zero. The resulting estimator is

$$\hat{f}_F(L) = \frac{\sum_{m=1}^{L} m \arg\left[\sum_{k=m}^{N-1} r(k)r(k-m)^*\right]}{2\pi L(L+1)(2L+1)}, \tag{4.5}$$

where L is a parameter of the algorithm that trades performance and complexity as far as $L|f_d|< 0.5$, which limits the applicability of the method for frequencies f_d bounded around 0 by $0.5/L$. The overall complexity of this method is $LN-(L^2-L)/2$ complex multiplications, L argument extractions, $L-1$ additions, and a division by a constant. In order to attain a complexity of the Fitz's method equivalent to that of a FFT of size N, L should be set to $L = \log_2(N)/2$. The resulting standard deviation of the estimation error using $L = \log_2(N)/2$ (respectively $L = \log_2(N)$) will be denoted by σ_F (respectively σ_{F2}) in the sequel.

There are many other methods in the literature: some of them use a FFT of size $N' > N$, using zero padding, to have a precision of the frequency in $1/N'$. This method can be further improved by deriving the maximum through interpolation. In particular, we will compare the new proposed method with a quadratic interpolation based on a FFT F of size of N' using [7]

$$\tilde{f}_d = \frac{1}{N'}\left(k_q + \frac{1}{2}\frac{|F(k_q-1)|-|F(k_q+1)|}{|F(k_q-1)|+|F(k_0+1)|-2|F(k_q)|}\right), \tag{4.6}$$

where k_q is the index maximizing the module of the Fast Fourier Transform F. The standard deviation of the estimation error obtained with this quadratic interpolation will be denoted by σ_Q for $N' = N$ and σ_{Q2} for $N' = 2N$. Finally, other methods improve the maximum selection by a dichotomous search around an initial estimation [8]. In all cases, the complexity of the methods is above $N\log_2(N)/2$ complex multiplications.

In the next section, we propose a method with a complexity $N\log_2(N)/2$ that gives an estimation error very close to the CRB in a very large range of Signal to Noise Ratio (SNR) and for any frequency f_d. Note that all the MATLAB codes used to generate the figures are available online in [9].

4.2.2 Proposed method

Let us first analyze the FFT of length N of the vector $\mathbf{r}$ received with no noise. Let $\mathbf{R}$ be the FFT of $\mathbf{r}$, and $k_0 \in \{-N/2+1,\ldots,N/2\}$ the index that maximizes $\mathbf{R}$

$$R(k) = \sum_{n=0}^{N-1} r(n)e^{-2\pi n\frac{k}{N}}, \;\; k \in \{-N/2+1,\ldots,N/2\}. \tag{4.7}$$

$$k_0 = \arg\max_k\{|R(k)|\}. \tag{4.8}$$

From k_0, a first estimate of f_d, given by $\hat{f}_d = \frac{k_0}{N}$, can be derived. Let $\delta \in]0.5,\ 0.5]$ be the fractional estimation error defined as $\delta = Nf_d - k_0$, thus

$$f_d = \frac{k_0+\delta}{N}. \tag{4.9}$$

By definition of the FFT[1],

$$R(k_0) = \sum_{n=0}^{N-1} r(n)e^{-\frac{2\pi j k_0 n}{N}} = \sum_{n=0}^{N-1} e^{\frac{2\pi j\delta n}{N}} = \frac{1-e^{2\pi j\delta}}{1-e^{\frac{2\pi j\delta}{N}}} \tag{4.10}$$

[1]The computation of the FFT can be bypassed if enough additional information is available on f_d. For example, if $|f_d|< 1/(2N)$, then $k_0 = 0$ and $R(0)$, $R(1)$ and $R(-1)$ can be computed directly.

Similarly, $R(k_0+1)$ is given by:

$$R(k_0+1) = \frac{1-e^{2\pi j(\delta-1)}}{1-e^{\frac{2\pi j(\delta-1)}{N}}} = \frac{1-e^{2\pi j\delta}}{1-e^{\frac{2\pi j(\delta-1)}{N}}} \tag{4.11}$$

and thus, the ratio $\alpha^+ = \frac{R(k_0+1)}{R(k_0)}$ is given by:

$$\alpha^+ = \frac{1-e^{\frac{2\pi j\delta}{N}}}{1-e^{\frac{2\pi j(\delta-1)}{N}}}. \tag{4.12}$$

For $\epsilon \ll 1$, $e^{j\epsilon}$ can be approximated by $1+j\epsilon+\frac{1}{2}\epsilon^2+j\frac{1}{6}\epsilon^3+\mathcal{O}(\epsilon^4)$, thus, setting $\epsilon = \frac{2\pi}{N}$, (4.12) gives

$$\alpha^+ = \frac{\frac{\epsilon^2\delta^2}{2}+j(-\epsilon\delta+\frac{\epsilon^3\delta^3}{6})+\mathcal{O}(\epsilon^4)}{\frac{\epsilon^2(\delta-1)^2}{2}+j(-\epsilon(\delta-1)+\frac{\epsilon^3(\delta-1)^3}{6})+\mathcal{O}(\epsilon^4)}. \tag{4.13}$$

Thus, the real part $\Re(\alpha^+)$ of α^+ is equal to:

$$\Re(\alpha^+) = \frac{\delta}{\delta-1}+\mathcal{O}(\epsilon^2). \tag{4.14}$$

Omitting the residual term $\mathcal{O}(\epsilon^2)$ and inverting (4.14), we can derive an estimation $\tilde{\delta}^+$ of δ from $\Re(\alpha^+)$ as

$$\tilde{\delta}^+ = \frac{\Re(\alpha^+)}{\Re(\alpha^+)-1}. \tag{4.15}$$

The estimation error $e(\delta) = \delta - \tilde{\delta}^+$ is plotted in Fig. 4.3 for $\epsilon = 2\pi/N$, $N = 64$, with δ varying between -0.5 and 0.5.

First we should note that the error curve is almost linear. It is thus possible to further improve the estimation using

$$\tilde{\delta}_i^+ = (1+\frac{\mu}{N^2})\tilde{\delta}^+, \tag{4.16}$$

where μ is selected to minimize the average quadratic error value $E[(\delta-\tilde{\delta}_i^+)^2]$, i.e. $\mu = 2.79$. The corresponding curve $e_i(\delta) = \delta - \tilde{\delta}_i^+$ for $N = 64$ is also shown in Fig. 4.3. In the sequel, the standard deviation of the frequency estimation error using $\tilde{\delta}$ (respectively $\tilde{\delta}_i$) will be denoted by σ_N^0 (respectively σ_N). So far, we have considered the ratio $\alpha^+ = \frac{R(k_0+1)}{R(k_0)}$ to estimate $\tilde{\delta}^+$. It is possible to derive the

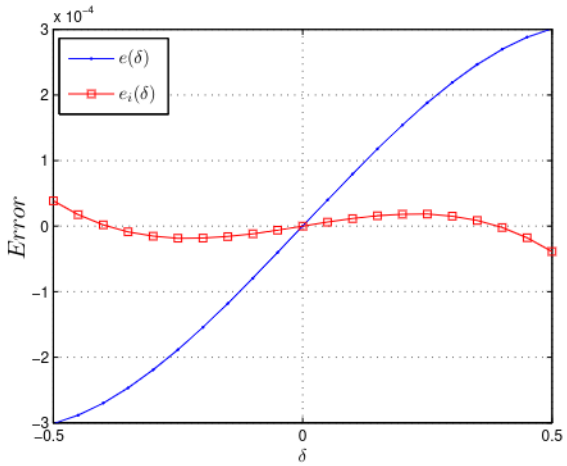

Figure 4.3: Error $e(\delta) = \delta - \tilde{\delta}$ and error $e_i(\delta) = \delta - \tilde{\delta}_i^+$

same computation with the ratio $\alpha^- = \frac{R(k_0-1)}{R(k_0)}$. In that case, the estimation $\tilde{\delta}^-$ is given as

$$\tilde{\delta}^- = \frac{\Re(\alpha^-)}{1 - \Re(\alpha^-)}. \tag{4.17}$$

In the absence of noise, both estimations give exactly the same result. Conversely, in the presence of noise, $\tilde{\delta}^+$ is more accurate than $\tilde{\delta}^-$ for $\delta \in [0, 0.5]$. Reciprocally, $\tilde{\delta}^-$ is more accurate than $\tilde{\delta}^+$ when $\delta \in]-0.5, 0]$. Thus, we propose to use the following estimation method: if $-\tilde{\delta}^+ < \tilde{\delta}^-$, $\tilde{\delta} = \tilde{\delta}^+$, else, $\tilde{\delta} = \tilde{\delta}^-$. The final estimation of $\tilde{f}_d$ is given by

$$\tilde{f}_d = \frac{1}{N}\left(k_0 + \tilde{\delta}\right). \tag{4.18}$$

Fig. 4.4 shows the standard deviation of the estimation error for $N = 64$ and an SNR of 0 dB. The standard deviations are estimated using 10000 runs. The x-axis represents the real δ value. In this figure, we also show the mean estimation $\tilde{\delta}_m$ defined as $\tilde{\delta}_m = \frac{\tilde{\delta}^+ + \tilde{\delta}^-}{2}$. The mean estimator $\tilde{\delta}_m$ is slightly more accurate than $\tilde{\delta}$ for a value of δ between -0.1 and 0.1. These properties can be used to slightly improve the performance.

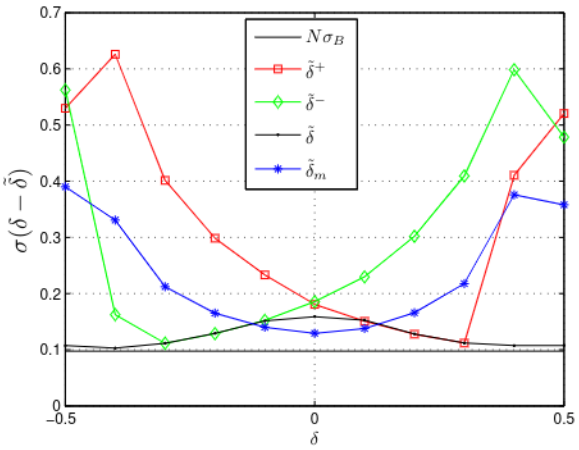

Figure 4.4: Standard deviation of error estimation as a function of δ for several methods with $N = 64$ and SNR = 0 dB

Let us assume that $R(k_0) = a + jb$, $R(k_0 + 1) = c + jd$ and $R(k_0 - 1) = e + jf$ then, after simplification, (4.15) and (4.17) can be expressed as

$$\tilde{\delta}^+ = \frac{ac + bd}{a(c - a) + b(d - b)}, \quad \tilde{\delta}^- = \frac{ae + bf}{a(a - e) + b(b - f)}. \tag{4.19}$$

Thus, the Non-Polynomial Interpolation (NPI) of $\tilde{\delta}^+$ and $\tilde{\delta}^-$ requires only 8 additions/subtractions, 8 multiplications and two divisions.

4.2.3 Performance comparison

In order to show the efficiency of the proposed method, we first fix $N = 64$. The frequency f_d is first assumed to be a random variable uniformly distributed between $[-1/N, 1/N]$. Fig. 4.5 shows the standard deviation of the frequency estimator based on the proposed methods using $\tilde{\delta}$ (with associated standard deviation σ_N^0) and $\tilde{\delta}_i$ (σ_N), the Fitz's method for the same complexity ($L = 3$, σ_F) and double complexity ($L = 6$, σ_{F2}) and the quadratic interpolation method with $N' = N$ (σ_Q) and $N' = 2N$ (σ_{Q2}). Moreover, the Cramer Rao Bound is also indicated (σ_B). Fig. 4.5 shows that σ_N^0 and σ_N differ only for a very high SNR, where the later flattens one decade below the former. Moreover, the standard deviation σ_N of the proposed method is closed to the CRB σ_B in a very large range of SNR. In fact, for SNRs between 0 dB and 65 dB, the ratio $\frac{\sigma_N}{\sigma_B}$ remains constant and equal to 1.28. The quadratic method almost reaches the CRB but for a shorter range of SNR and a double computational

cost. Finally, Table 4.1 extends the result of Fig. 4.5 for several values of N giving the minimum ratio $(\sigma/\sigma_B)_{\min}$ obtained and the approximate range of SNR where $\sigma_N/\sigma_B \leq (\sigma_N/\sigma_B)_{\min} + 0.1$. As shown in this table, $(\sigma_N/\sigma_B)_{\min}$ remains constant and is equal to 1.28. The Fitz method becomes less and less efficient when N increases (for $N = 4048$, $\sigma_F = 11.2\sigma_B$ while $\sigma_N = 1.28\sigma_B$!).

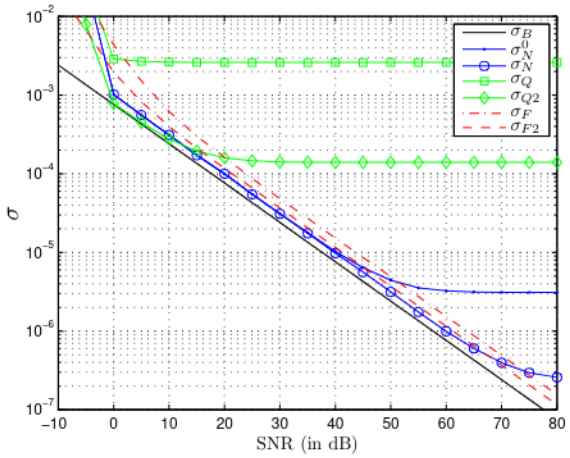

Figure 4.5: Variance of the frequency estimation error for $N = 64$ with several methods

Table 4.1: Performance of new algorithm

N	16	64	256	1024	4096
$(\sigma_F/\sigma_B)_{\min}$	1.32	2.0	3.5	6.1	11.2
$(\sigma_{F2}/\sigma_B)_{\min}$	1.07	1.5	2.5	4.4	7.9
$(\sigma_N/\sigma_B)_{\min}$	1.30	1.28	1.28	1.28	1.28
Range (dB)	[5, 55]	[0, 65]	[-5, 80]	[-10, 100]	[-15, 105]

Before concluding, it is worth mentioning that using $\bar{\mathbf{R}}$, the FFT of the normalized input vector $\bar{\mathbf{r}} = \{r(n)/|r(n)|\}_{n=0,\ldots,N-1}$, instead of $\mathbf{R}$ in the proposed method, reduces significantly the error variance for a high SNR. As shown in Fig. 4.6, the standard deviation can be reduced from $\sigma_N = 1.28\sigma_B$ down to $\bar{\sigma}_N = 1.08\sigma_B$ for a large range of SNRs.

4.2.4 Conclusions

In this annex, we have presented a Non-Polynomial Interpolation method to refine the precision of the frequency estimation based on FFT. The proposed method dramatically outperforms all state of the art solutions of the same complexity for a large set of useful configurations. For example, for $N = 4096$, the combination of FFT and NPI reduces the standard deviation of the frequency estimation error by a decade compared to the Fitz method of same complexity (i.e. with $L = 6$). The proposed NPI method can be applied in many applications, for example, the frequency offset estimation in a digital receiver.

4.3 Matlab code for phase estimation error

```
%=====================================
% Emmanuel Boutillon
%=====================================
% 15/10/2020
```

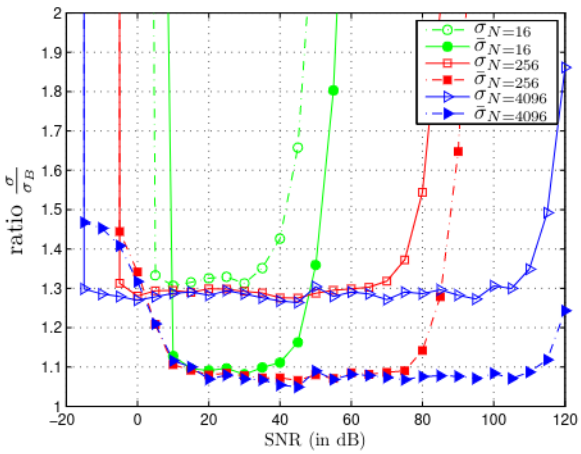

Figure 4.6: Ratio $\frac{\sigma_N}{\sigma_B}$ as a function of the SNR for $N = 16$, 256, and 4096

```
% Code to estimate phase error of a decoder.
%=====================================
%=== Input
% q: input size of CCSK sequence
% N: Size of the frame (i.e., number of CCSK symbols).
% s: PN sequence express a string.
% == Output
% snr_tab : range of SNR used
% std_v   : standard deviation of the estimation.
% std_v_GA : standard deviation of the estimation aided by a genius (Genius Aided
%                              that takes only into account max correlation corresponding
%                              to a correct decision.
% Prob_value : vector that give the probability that the absolute value of the error estimatio
% is bellow 0.1 dB, 0.5 dB, 1 dB and 3 dB, respectively
% Prob_value_GA : vector that give the probability that the absolute value
% of the error estimation aided by a genius (Genius Aided)
% is bellow 0.1 dB, 0.5 dB, 1 dB and 3 dB, respectively
%=========================================
function [snr_tab std_v, std_v_GA, Prob_value, Prob_value_GA] = estim_phase(q, N, s);

%=========================================================
% Fix Parameters of the simulation. Can be changed freely.
%=========================================================

Nb_samples     = 10^5;
angle_threshold = acos(sqrt(10.^(-[0.1, 0.5, 1, 3]/10)));
snr_tab        = -20:1:0;    % Table of SNR used for the simulation.

% Transformation of string sequence to BPSK sequence
p = 2*(s - '0')-1;

% Computation of FFT transform of CCSK sequence
FFTP = fft(p);
```

```
for snr_ind = 1:length(snr_tab)
    snr = snr_tab(snr_ind);
    sigma = sqrt(0.5*10^(-snr/10));

    phi = 2*pi*rand(1,Nb_samples) - pi;
    for i=1:Nb_samples
        a = floor(rand(1,N)*q)+1;
        sum_max = 0;  %sum_maximum
        sum_max_GA = 10^-8;  %sum_maximum with decision GA.
        for n = 1:N
            x = exp(1i*phi(i))*[p(a(n):end) p(1:a(n)-1)] + sigma*randn(1,q)+ 1i*sigma*randn(1,
            theta = ifft( conj(fft(x)).*FFTP);
            [A(n), I(n)] = max( abs(theta));
            phase(n) = angle(theta(I(n)));
            sum_max = sum_max + theta(I(n));
            if I(n) == a(n)            %Correct detection given by Genius.
                sum_max_GA = sum_max_GA + theta(I(n));
            end
        end
        sum_max = angle(sum_max);
        sum_max_GA = angle(sum_max_GA);
        erreur(i) = phi(i) + sum_max;
        if abs(erreur(i)) > pi
            erreur(i) = erreur(i) - sign(erreur(i))*2*pi;
        end
        erreur_GA(i) = phi(i) + sum_max_GA;
        if abs(erreur_GA(i)) > pi
            erreur_GA(i) = erreur_GA(i) - sign(erreur_GA(i))*2*pi;
        end
        % Computation of rougth error distribution
    end

    for k = 1:length(angle_threshold)
        Prob_value(snr_ind,k) = sum (abs(erreur) > angle_threshold(k))/Nb_samples;
        Prob_value_GA(snr_ind,k) = sum (abs(erreur_GA) > angle_threshold(k))/Nb_samples;
    end

    mean_v(snr_ind) = mean(erreur);
    std_v(snr_ind) = sqrt( var( erreur));
    std_v_GA(snr_ind) = sqrt( var( erreur_GA));
end
```

Bibliography

[1] (2020) Web site on Non-Binary LDPC. [Online]. Available: http://www-labsticc.univ-ubs.fr/nb˙ldpc/.

[2] M. C. Davey and D. J. C. MacKay, "Low density parity check codes over GF(q)," in *1998 Information Theory Workshop (Cat. No.98EX131)*, June 1998, pp. 70–71.

[3] E. Boutillon, L. Conde-Canencia, and A. Al Ghouwayel, "Design of a GF(64)-LDPC Decoder Based on the EMS Algorithm," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 60, no. 10, pp. 2644–2656, Oct 2013.

[4] D. Rife and R. Boorstyn, "Single tone parameter estimation from discrete-time observations," *Information Theory, IEEE Transactions on*, vol. 20, no. 5, pp. 591–598, Sept 1974.

[5] I. Perisa and J. Lindner, "Employing simple fft-interpolation for improved complex tone detection and fine estimation," in *Wireless Communication Systems, 2006. ISWCS '06. 3rd International Symposium on*, Sept 2006, pp. 744–748.

[6] M. Fitz, "Further results in the fast estimation of a single frequency," *Communications, IEEE Transactions on*, vol. 42, no. 234, pp. 862–864, Feb 1994.

[7] J. Smith and X. Serra, "Parshl: A program for the analysis/synthesis of inharmonic sounds based on a sinusoidal representation," in *Proceedings of the 1987 International Computer Music Conference (ICMC) ICMC'87, available at https://ccrma.stanford.edu/∼jos/parshl/parshl.pdf*, 1987.

[8] E. Aboutanios and B. Mulgrew, "Iterative frequency estimation by interpolation on fourier coefficients," *Signal Processing, IEEE Transactions on*, vol. 53, no. 4, pp. 1237–1242, April 2005.

[9] E. Boutillon. (2015, January). [Online]. Available: http://www-labsticc.univ-ubs.fr/$\sim$boutillon/Freq_estimation/Freq_estimation.html