Mitigation of Multipath-induced Errors in Satellite Navigation

Master Thesis

Ioana Gulie

Aalborg University
Department of Electronic Systems
Fredrik Bajers Vej 7B
9220 Aalborg, Denmark

Astrium GmbH
Department of Satellite Systems
Robert Koch Strasse 1
82024 Taufkirchen, Germany

Department of Electronic Systems
Fredrik Bajers Vej 7
DK-9220 Aalborg Ø
http://es.aau.dk

AALBORG UNIVERSITY
DENMARK

Title:
Mitigation of Multipath-induced Errors in Satellite Navigation

Theme:
Signal Processing

Project Period:
Autumn Semester 2013 and Spring Semester 2014

Project Group:
14GRP1071

Participant(s):
Ioana Gulie

Supervisor(s):
Carles Navarro Manchon (AAU)
Fleury Henri Bernard (AAU)
Jan Wendel (Astrium)
Frank Schubert (Astrium)

Copies: 6

Page Numbers: 81

Date of Completion:
10.06.2014

Abstract:

This master thesis is a research project, in the field of Global Navigation Satellite Systems, focused on joint estimation of time of arrival of direct and reflected signal, for improving the user positioning. The purpose is to investigate if, and the degree in which, different joint estimation methods can outperform the tracking loops already implemented in the conventional receivers. The maximum likelihood estimator will be investigated and implemented. Due to the computational complexity of the maximum likelihood estimation, signal compression techniques will be used. Different constraints will be proposed and developed.
For evaluation of the solutions, a simulation platform will be developed, which will be used for generating the direct and reflected signals and also for implementing the maximum likelihood algorithm. The performance of estimator in multipath environments will be compared with the classical tracking loops.

Preface

This thesis is submitted to the Aalborg University, Denmark, in fulfillment of the requirements for the Master of Science in Engineering. It has been partially supported by Astrium GmbH, Department of Satellite Systems. The work has been carried out during the period spanning from October 2013 to June 2014 at the Faculty of Engineering and Science, Aalborg University, Denamrk and at Astrium GmbH, Taufkirchen, Germany.

I would like to express my gratitude for the excellent guidance and support provided to my supervisors from Astrium, Dr.-Ing. habil. Jan Wendel and Dr. Frank Schubert and to my supervisors from Aalborg, Assistant Professor Carles Navarro Manchón and Professor Bernard Henri Fleury.

Contents

Preface iii

Acronyms xi

1 Introduction 1
1.1 Objectives 2
1.2 Thesis Outline 2

2 Global Navigation Satellite systems 3
2.1 GNSS introduction 3
2.2 Type of errors 4
2.3 GNSS signals 5
2.4 GNSS user equipment 8
2.5 Delay Lock Loop and the effects of multipath propagation 11
2.6 Multipath mitigation in conventional receivers 13
2.7 Signal to noise ratio in GNSS 14

3 Maximum Likelihood Estimation 17
3.1 Signal model 18
3.2 Cost Function 19
3.3 Parameters Estimation 21
3.3.1 Amplitudes determination 22
3.3.2 Delays estimation 23
3.3.3 Algorithm overview 23
3.4 Parameters Estimation with constrained amplitudes 25
3.4.1 Amplitude constraints 25
3.4.2 Constrained minimization 26
3.5 Computational complexity 30

4 Compression 33
4.1 Compression in general 33
4.2 Compression based on code matched correlators 35
4.3 Compression based on signal matched correlators 38

5 Numerical Assessment 41
5.1 Assumptions about the system 41
5.2 Discrete delays 41
5.2.1 No multipath component 42

5.2.2 One multipath component 44
5.2.3 Two multipath components 48
5.3 Continuous delays 49
5.3.1 Search algorithm 50
5.3.2 No Multipath Component 50
5.3.3 One multipath component 51
5.3.4 Two multipath components - Noiseless channel 57

6 Conclusion 59

Bibliography 61

A SNR in DLL 63

B Amplitudes derivation without constraints 65

C Amplitudes derivation with Lagrange constraint 67

D Cost Function minimization following Weill [15] 71
D.1 Amplitudes derivations 73

E Additional simulation figures 79
E.1 Grid delays, LOS and 1 MP, No Noise 79
E.2 Continuous delays, LOS and 1 MP, No Noise 79

Todo list

SPC10-14GRP1071

Aalborg University & Astrium GmbH, June 11, 2014

Ioana Gulie
<igulie12@student.aau.dk>

Acronyms

GNSS	global navigation satellite systems
GPS	Global Positioning System
GLONASS	Global Navigation Satellite System
TOA	Time of arrival
ML	Maximum Likelihood
BPSK	Binary phase shift key
PRN	Pseudo-random noise
BOC	Binary offset carrier
CDMA	Code Division Multiple Access
QPSK	Quadrature Phase Shift Keying
IF	Intermediate Frequency
BB	Base Band
NCO	Numerical Controlled Oscillator
E	Early
P	Prompt
L	Late
DLL	Delay Lock Loop
CRLB	Cramér-Rao lower bound
PDF	probability density function
CMC	Code Matched Correlators
SMC	Signal Matched Correlators
LOS	Line of sight
SNR	Signal to noise ratio

Chapter 1

Introduction

In global navigation satellite systems (GNSS), like Global Positioning System (GPS), operated by U.S., Galileo, under development by the European Union and partners, or Global Navigation Satellite System (GLONASS), operated by Russia, the user position is determined from the measurements of time of flight of radio signals transmitted by orbiting satellites, technique also known as passive ranging. The GNSS signals can suffer interferences, blockage, attenuations and reflections until they reach the user equipment, and this is affecting the position measurement.[5] While most of the errors can be overcome, the ones caused by reflections, also known as multipath errors, are challenging. The reflections on the surrounding buildings and terrain are added as delayed multipath components to the direct radio signal, introducing a bias in the determination of time of flight. The reflections have a random behaviour for the user equipment, making it difficult to impossible to remove completely.

The user equipment, also known as GNSS receiver, is designed to reduce the multipath errors. Different types of receivers have different performances in terms of multipath mitigation. Widely used industry standard receivers are the narrow correlator and double delta correlator. In those two cases the DLL discriminator in the receiver is the one that reduces the multipath errors. The antenna of the user equipment can also be modified to mitigate the multipath, either hardware, like the choke ring antennas, or hardware and software, like the Beam-forming for the array antennas.

New multipath mitigation methods are using estimation theory to detect the time of flight of the direct component as well as of the multipath components. Those techniques can be implemented using the DLL discriminator of the receiver and the range processor, as it will be shown later in the paper. At the end of the estimation process, only the information about the direct component is used in the range measurement. This is known as joint estimation of the time of flight of direct and reflected signals and will be the research subject of this thesis.

1.1 Objectives

The objective of this thesis is to investigate different approaches for the joint estimation of the time of flight of the direct signal and the reflected signals and to compared them regarding their multipath mitigation capabilities by means of numerical simulations. The goal is to investigate if, and the degree in which, different joint estimation methods can outperform the tracking loops already implemented in the conventional receivers.

In this thesis, the maximum likelihood joint estimation of the time of flight will be investigated. Due to the computational complexity of the maximum likelihood estimation, signal compression techniques will be used. Different constraints will be proposed and developed.

For evaluation of the solutions, a simulation platform has to be developed, which will be used for generating the direct and reflected signals and also for implementing the proposed algorithms. The performance of estimation methods in multipath environments will be compared with the classical tracking loops.

1.2 Thesis Outline

The structure of the thesis is as follows.

Chapter 1 states the problem of multipath mitigation and presents the objectives of the thesis.

Chapter 2 provides an overview of the GNSS fundamentals, which are a necessary background when working with GNSS signals and multipath propagation.

Chapter 3 is proposing and developing a maximum likelihood estimation algorithm for the delays of the line of sight and of the multipath signals. For a higher efficiency, constrains are proposed on the amplitudes.

Chapter 4 presents two different compression methods, which are based on signal matched correlators and code matched correlators.

Chapter 5 evaluates the performance of the proposed maximum likelihood estimation algorithm, for both compression methods presented, and compares it with the classical DLL with 0.1 and 0.3 early-late spacing. One ideal scenario is considered, with delays on the sample grid, in which the general behaviour of the algorithm is illustrated, and a realistic scenario, with delays that can take any value in between +/- one chip.

Chapter 6 presents the achievements of this thesis and future directions of development of the work presented.

Chapter 2

Global Navigation Satellite systems

This chapter will cover the basic aspects of GNSS that are used in the next chapters. First, an introduction to GNSS will be given and the main types of errors that can occur will be described briefly. After that, the GNSS signal will be discussed. The chapter will end with a presentation of the GNSS user equipment, the tracking loops and the most used receivers for multipath mitigation.

2.1 GNSS introduction

According to Groves in [5], GNSS refers to *those navigation systems that provide the user with a three-dimensional positioning solution by passive ranging using radio signals transmitted by orbiting satellites.* Passive ranging means that the user is receiving signals without transmitting anything back. The satellite is emitting a signal from which the user can determine the range, i.e. its distance from the satellite. The range measurement from several satellites, combined with the information about the satellite position contained in the signal, allows the user equipment to compute the user position.

The range ρ, can be expressed as a function of the transmission time t_t, the arrival time t_a, and the speed of signal propagation, given by the speed of light, c, [5]

$$\rho_j = (t_a - t_{t,j})\, c, \tag{2.1}$$

where j is the index for a certain satellite.

In an error free environment, only three satellites are needed for the determination of the user position. In the three-dimensional space, with just one satellite emitting, the user could be anywhere on the sphere centred on the satellite and with the radius given by the range. With two satellites emitting, the user could be on the intersection circle between the two spheres. With three satellites, the solution is on one of the two points where the three spheres intersect, and with the Earth as a fourth sphere, only one point is selected.

The three-dimensional user position, at a certain arrival time, $\boldsymbol{r_u}(t_a)$, is computed from a set of three equations, corresponding to three different satellites. The equation for satellite j is

$$\rho_j = \sqrt{(\boldsymbol{r_{s,j}}(t_{t,j}) - \boldsymbol{r_u}(t_a))^T(\boldsymbol{r_{s,j}}(t_{t,j}) - \boldsymbol{r_u}(t_a))}, \tag{2.2}$$

where $\boldsymbol{r_{s,j}}(t_{t,j})$ is the position of satellite j at the transmission time $t_{t,j}$ and because of the data contained in the GNSS signal, it is known to the receiver for any given moment of time. The range ρ_j was determined according to equation (2.1). So the only unknown in this equation is the user position $\boldsymbol{r_u}(t_a)$.

The satellites have atomic clocks which give precise clock measurements. All the satellites are synchronised with each other. Equations (2.1) and (2.2) are for the ideal case, when the receiver clock is also synchronized with the satellites clocks. But in practice this does not happen. The satellites and receiver clocks are not perfectly synchronised and this affects the range measurement. Having just one receiver involved, the error will be the same for all the ranges determined by it and it can be overcome by considering the range measurement from a fourth satellite. The only change will be that in equation (2.2) another unknown parameter will be added, representing the error, and four equations will be needed. To distinguish between the ideal ranges measured without clock errors and the true ones measured with clock errors, the term of pseudo-range will be used for the latter ones. One of the equations, for satellite j, will be

$$\widetilde{\rho_j} = \sqrt{(\boldsymbol{r_{s,j}}(t_{t,j}) - \boldsymbol{r_u}(t_a))^T(\boldsymbol{r_{s,j}}(t_{t,j}) - \boldsymbol{r_u}(t_a))} + e_c(t_a), \tag{2.3}$$

where $\widetilde{\rho_j}$ is the pseudo-range and e_c is the error due to receiver clock.

2.2 Type of errors

For the range measurement, it is important to know the sources of errors and to try to model them. The following are the most important ones.

Errors in the data broadcast by the satellite: The satellite transmits orbit data, known as ephemeris data, which is a set of parameters used to compute satellite position at a certain moment of time. The information about the transmission time is controlled by the internal clock of the satellite which is very precise, but sometimes errors can occur. Both the ephemeris data and satellite clock data are analysed by a network of monitor stations, at precise locations, with highly accurate synchronized clocks, which are part of the Control Segment, also known as Ground Segment, of the GNSS system. The control segment includes also control stations and uplink stations. The control stations detect if there are errors in the ephemeris data or satellite clock data and can use the uplink stations to transmit correcting information to the satellite station. [5] Even if there are ephemeris errors in the signal, those are small and the resulting user position error is not significant, with values below 2 meters [1].

Atmospheric errors: Due to the free electrons in the ionosphere and the gases in the troposphere, the signals are suffering delays. Both ionospheric and tropospheric delays are dependent on the elevation angle of the satellite, being much higher for low angles. The elevation angle is the angle between the line of sight from the satellite to the receiver and the horizontal plane. Most GNSS user equipment have a threshold for elevation angle, ignoring the signals that are received from angles below 5 or 10 degrees. Ionospheric delays vary with solar radiation, meaning that the delays are different depending on the time of the day. Also they vary with frequency and that is why, if the ranging measurement is conducted on more frequencies, the ionospheric delay can be determined. The tropospheric delays vary with weather conditions. For both ionospheric and tropospheric delays, models can be used to partially correct them. [5]

Receiver measurement errors: The range measurement in the receiver can be affected by the thermal noise, radio-frequency interferences and other inaccuracies. This error will be different for each pseudo-range determined by the receiver. It can be added as an extra component to the equation (2.3), but is basically impossible to determine it. However, the resulting position error will not be larger than 1.5 meters.

$$\widetilde{\rho_j} = \sqrt{(\boldsymbol{r_{s,j}}(t_{t,j}) - \boldsymbol{r_u}(t_a))^T(\boldsymbol{r_{s,j}}(t_{t,j}) - \boldsymbol{r_u}(t_a))} + e_c(t_a) + e_{n,j}(t_a), \tag{2.4}$$

where $e_{n,j}(t_a)$ represents the noise error for one pseudo-range determined.

Signal blockage: Depending on the elevation angle, the surrounding buildings and terrain, the signal from a certain satellite can be blocked. The signal can be blocked also by the user vehicle. In mountain areas or streets with high buildings, also known as urban canyons, mostly for low elevation satellites, signal blockage can be severe and the user equipment might receive signals from less then four satellites. If this is for short periods of time, one of the four unknown parameters used to determine the user position, as presented in equation (2.3), can be assumed to be known, for example the user height or the receiver clock error, equal to the previous determined values. [5]

Multipath errors: The signal received by the user equipment in a certain observation interval is usually not just the combination of direct signals from the satellites in view. It also contains reflected or diffracted components of the same signals. The delayed components of a certain satellite are attenuated, phase shifted with respect to the direct component and delayed. More details about the multipath components will be given in the next chapters. As it will be explained, when the delays are high, the user equipment can ignore them when computing the range. But if the delays are small, also known as close multipath, they introduce a bias in the time of flight calculation. The multipath errors are a big challenge, because they are difficult to model and reduce. They are dependent on the outside environment, and for the close multipath this is changing randomly and is hard to predict.

2.3 GNSS signals

This section will give a brief overview about the GNSS signals common characteristics and some details which are not of interest for this thesis will be skipped.

Each GNSS signal has three important components, the navigation message data, the PRN code and the carrier component. Each of them will be presented next.

Each satellite sends data messages, in the form of a baseband signal, with the frequency spectrum around 0Hz, and with values of either +1 or −1 [6]. The data message rate is usually between 50 and 500 symbols per second [5].

Further, another component is added, the spreading or pseudo-random noise (PRN) signal. The PRN signal is similar to the data message signal, with values of either +1 or −1, but with a higher symbol rate, between 0.511 and 10.23 Mchips per second. To distinguish between the PRN and data, in the PRN case, one symbol is called chip and the PRN symbol rate is measured in chips per second and is called chipping rate, even if mathematically symbols and chips are the same. [5] The bandwidth of the modulated signal is in general proportional to the chipping rate [6]. The interval of time in which a chip is transmitted is the chip period.

The PRN signal is periodic and one period represents the PRN code. Depending on the type of signal, the PRN codes have different lengths and characteristics. One satellite broadcasts several PRN codes, corresponding to different access and performance levels. All PRN codes are pseudo-random sequences, predefined, known a priori by the receiver and distinct between satellites. Except for the fact that is completely known at the receiver, a PRN code has the same properties as a random sequence of the same length. That is why it does not correlate with other PRN codes, or with itself if it is delayed with more than one chip. Because of those properties, the PRN codes play an important role in GNSS, in signal transmission and in multipath mitigation techniques.

The PRN and data message signals could be expressed as the convolution between a chipping or data sequence, and a pulse. The pulse shape can be a rectangular window, as is the case in figure 2.1, or other shapes, like the Root Raised Cosine [11]. In practice, the rectangular window pulse shape is widely use, the others being to expensive to implement comparing to the performance gain.

Before transmission, the data signal is modulated on higher frequencies, on a carrier frequency between 1 and 2 GHz, resulting in a bandpass signal. This is called Binary Phase Shift Keying (BPSK) modulation. This type of modulation is represented in figure 2.1 and can be expressed as in equation (2.5).

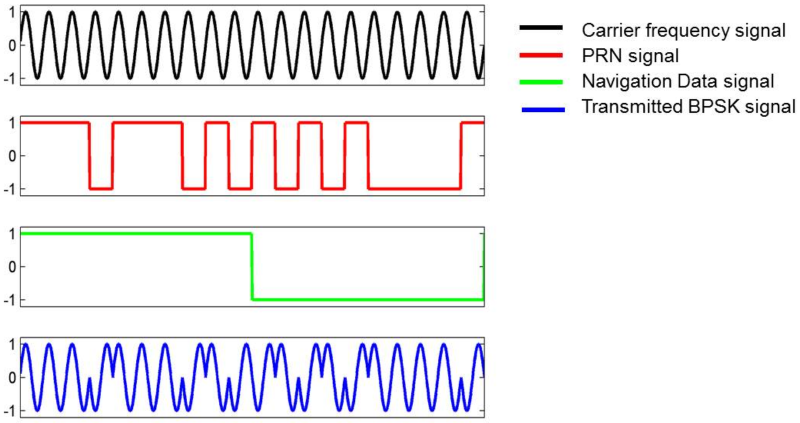


Figure 2.1: The signals contributing to the BPSK signal.

$$s_{BPSK}(t) = ac(t)d(t)cos(2\pi f_c t + \phi_c), \tag{2.5}$$

where s_{BPSK} is the BPSK modulated signal emitted by a certain satellite, a is the amplitude, c is the PRN signal, d is the data signal and f_c and ϕ_c are the carrier frequency and phase offset.

Another type of modulation, specific to many of the new GNSS signals, is the Binary Offset Carrier (BOC) modulation. It is more complex but gives better performance for the signal in terms of tracking capabilities and multipath resilience [2]. The modulated signal can be expressed as

$$s_{BOC}(t) = ab(t)c(t)d(t)cos(2\pi f_c t + \phi_c), \tag{2.6}$$

where s_{BOC} is the BOC modulated signal emitted by a certain satellite, b is the extra component added, called sub-carrier function, which is usually a periodic signal with a rectangular window shape, with amplitudes of $+1$ and -1. The rest of the parameters are the same as in (2.5). Compared to the BPSK signals, the center of the frequency spectrum of BOC signals is moved from f_c to $f_c \pm f_s$, where f_s is the frequency rate of $b(t)$.

Now, that the main components of the GNSS signal have been defined, the role of the PRN code in signal transmission and multipath mitigation will be detailed.

In GNSS signal transmission, one PRN code corresponds to one signal. The receiver identifies very easy each signal, with the help of the PRN code. Different satellites can transmit in the same time, sharing the same carrier frequency, due to the separation offered by the PRN codes. This technique is known as Code Division Multiple Access (CDMA).

Usually, one satellite station needs to transmit simultaneously more signals, sharing the same carrier frequency. One popular technique is Quadrature Phase Shift Keying (QPSK). If two BPSK signals are multiplexed together using QPSK, the resulting signal is

$$s_{QPSK}(t) = a_I c_I(t) d_I(t) cos(2\pi f_c t + \phi_c) - a_Q c_Q(t) d_Q(t) sin(2\pi f_c t + \phi_c), \tag{2.7}$$

where the subscripts I and Q denote the in-phase and quadrature-phase components.

The PRN code is important in multipath mitigation techniques because of its autocorrelation function and it will be explained in the next sections how those properties are used in multipath mitigation. For now just the autocorrelation properties of the PRN code will be presented. The autocorrelation function of the PRN code is almost zero for delays larger than one chip and maximum when the delay is zero. This can be explained intuitively by considering the pseudo-random characteristic of a rectangular shape PRN code with amplitude $+1$ and -1. If two identical such codes are out of sync with more then one chip, at a random moment of time, the multiplication of the two codes could take random values of $+1$ and -1, and the summation of those values over an accumulation interval will be approximately 0. Following the same approach, if the two identical codes are out of sync with less then one chip then the probability that their multiplication value at a certain moment of time is $+1$ is proportional with the overlapping region. This can be expressed as follows.

The autocorrelation function for a continuous PRN signal, $c(t)$, is given by the relation

$$R_{cc}(\tau) = \lim_{T \to \infty} \frac{1}{T} \int_0^T c(t-\tau) c(t) dt, \tag{2.8}$$

where τ is the delay and T is the accumulation interval.

In GNSS, the accumulation interval is at least 1 ms [5] and the sampling rates are high, resulting in high enough number of samples N, to be able to make the following approximation for the autocorrelation function of the discrete PRN signal.

$$R_{cc}(i) = \lim_{N \to \infty} \frac{1}{N} \sum_{n=1}^{N} c_{n-i} c_n \tag{2.9a}$$

$$\approx \frac{1}{N} \sum_{n=1}^{N} c_{n-i} c_n \tag{2.9b}$$

where i represents the number of samples corresponding to the delay τ, and c_n is one sample of the PRN signal, from a vector of samples indexed after n.

For the PRN code with rectangular shape and amplitude 1, and for an accumulation interval less than the length of the PRN code, the resulting expression for the autocorrelation function is as given in (2.10) for the discrete case, and in (2.11) for the continuous case [6].

$$R_{cc}(i) = \begin{cases} 1 - \frac{|i|}{N_c} & \text{for} \quad |i| \leq N_c \\ 0 & \quad\quad \text{otherwise} \end{cases} \tag{2.10}$$

where N_c is the number of samples in one chip period. In terms of continuous delay, this autocorrelation can be expressed as

$$R_{cc}(\tau) = \begin{cases} 1 - \frac{|\tau|}{T_c} & \text{for} \quad |\tau| \leq T_c \\ 0 & \quad\quad \text{otherwise} \end{cases} \tag{2.11}$$

where T_c represents one chip period.

The autocorrelation function from equation (2.11) is represented in figure 2.2.

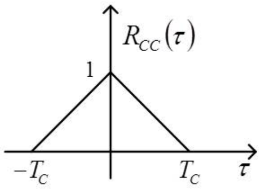


Figure 2.2: The autocorrelation function for a PRN code with rectangular shape.

It will be presented in the next sections how the autocorrelation function is used in the receiver.

2.4 GNSS user equipment

In figure 2.3 is presenting a block diagram of the user equipment. The receiver is not clearly separated in the user equipment. The ranging processor block is a software processing device and not everyone consider it part of the receiver. Some authors see it as another component of the user equipment which functions in a loop with the receiver. In this thesis, the ranging processor is considered part of the receiver.

The only two blocks, which are of interest for the multipath mitigation techniques analysed in the next chapters, are the intermediate frequency (IF)/base band (BB) signal processing and the ranging processor blocks, which are marked with a blue doted line in figure 2.3. All the rest will be described briefly, for putting the mentioned blocks in the context.

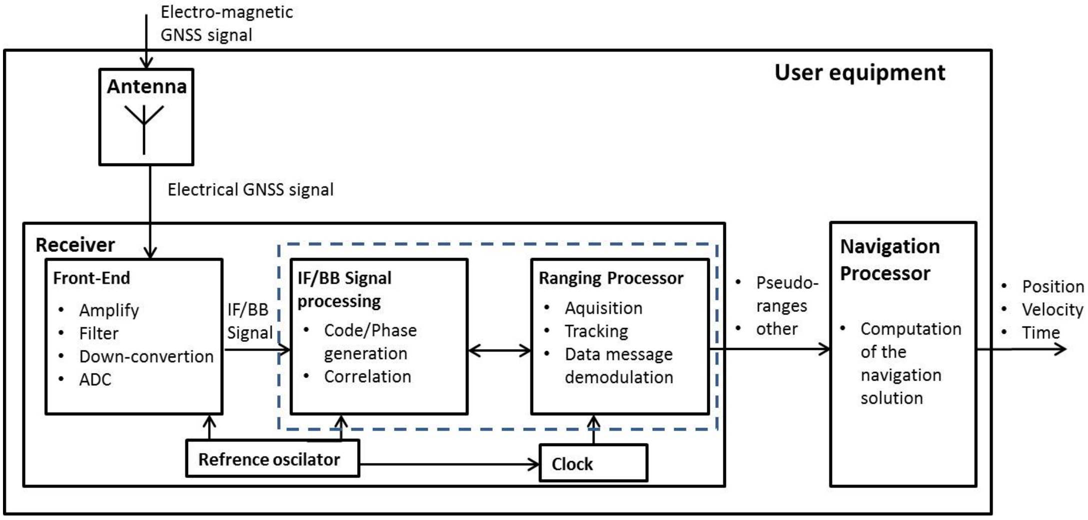


Figure 2.3: Block diagram showing the GNSS user equipment.

The antenna is converting an electromagnetic signal into an electric signal [5]. The antenna can be designed to mitigate multipath, either at the hardware level, like the Choke Ring Antenna, or software, like the Beam-forming, as already mentioned in the introduction of this thesis. The output from the antenna is directed to the front-end of the receiver.

One satellite broadcasts different signals in different frequency bands. In GPS, there are 10 different types of navigation signals, like C/A, P(Y) and others, corresponding to different applications and with different access levels, each allocated to one of the three bands available, known as L1, L2 and L5. Similarly, in Galileo there are other 10 types of signals allocated across three frequency bands. More about the different GNSS signals and frequency bands can be found in [5].

In the front-end, the signals are processed separately, if belonging to different frequency bands. The received signal is processed in the analogue domain first and then converted to the digital domain. Different receiver types process the signal differently, but usually it is amplified, band-limited and down-converted. The down-conversion is done in two stages, and in the second stage the signal can be converted to base band (BB) or to a frequency called intermediate frequency (IF), depending on the receiver type. In this thesis the BB will be used, but the IF will be presented for giving a general picture.

In the IF signal, the carrier frequency is denoted by f_{IF}. If the transmitted signal is a QPSK signal, then the output IF signal, s_{IF}, is given by the relation

$$s_{IF}(t) = Re\{s(t)e^{j(2\pi(f_{IF}+f_{DOP})t+\phi)}\}, \tag{2.12}$$

where

$$s(t) = a_I c_I(t) d_I(t) + j a_Q c_Q(t) d_Q(t). \tag{2.13}$$

The same notations as in (2.7) are being used, with the specification that the amplitudes a_I and a_Q are after the transformations in the front-end. The Doppler frequency, f_{DOP}, is due to the relative movement between the user and the satellite, during the signal receiving process. More details about the Doppler frequency can be found in literature ([8],[6]). If the expression in (2.12) is extended, it can be observed that the s_{IF} signal is similar to the s_{QPSK} signal from (2.7), with changes in frequency, phase and amplitude.

The BB signal is also the result of a down-conversion of the received signal, but to a carrier frequency equal to 0 Hz. The resulted signal is a complex signal and can be expressed as

$$s_{BB}(t) = s(t)e^{j(2\pi f_{DOP}t+\phi)}, \tag{2.14}$$

with the same notations as in (2.12) and with s given by (2.13).

In the BB signal processing block, each type of signal broadcast by each satellite is identified, based on the PRN code. For this to be possible, one channel is allocated to each type of signal and to each satellite in view. The total number of channels in a receiver, $n_{channels}$, can be expressed as

$$n_{channels} = n_{types} * n_{satellites}, \tag{2.15}$$

where n_{types} indicated how many type of signals a certain receiver can decode and $n_{satellites}$ represents how many satellites are in the view of the receiver. Typically, $n_{satellites}$ value is between 10 and 12, and n_{types} value can be equal to or greater than 1.

The main task of the BB signal processing and the ranging processor blocks is, for each channel, to find the corresponding pseudo-ranges and the satellite positions. The difference between the time of transmission and the time of arrival, known as time of flight, multiplied by the speed of light, gives the pseudo-range, as explained in equation (2.1). The pseudo-ranges, satellite positions and other data, from each channel, are sent to the navigation processor, which computes the user position, as shown in equation (2.3).

Timing parameters and the ephemeris data can be retrieved from the message data, after demodulation. If the exact time of transmission is known, the satellite position is calculated from the ephemeris data. To calculate the time of flight, three different stages are needed: acquisition, tracking and message data demodulation. The time of flight, t_{flight}, can be expressed as

$$t_{flight} = T_c(x\ N_{PRN} + y), \tag{2.16}$$

where T_c is one chip period, N_{PRN} is the PRN code length in chips, x is an integer retrieved from the message data and y is computed in the acquisition and tracking stage and with the property that $y < N_{PRN}$. The value given by yT_c is known as code phase.

The BB signal processing and the ranging processor blocks, for one channel, are detailed in figure 2.4.

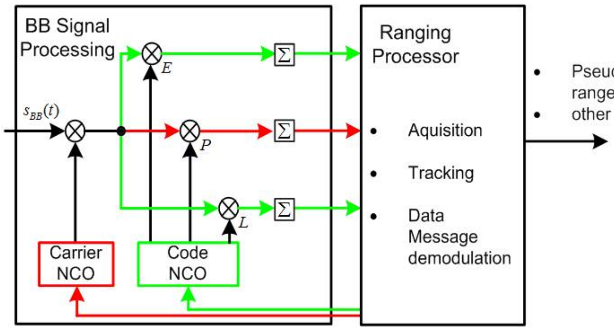


Figure 2.4: Block diagram showing the tracking loops for one channel.

To determine the Doppler offset, given by f_{DOP} in formula (2.14), and the phase of the PRN code, yT_c from (2.16), two loops are working together, the carrier loop and the code loop, in two stages, the acquisition and tracking stage. The acquisition gives the coarse estimates and the tracking is refining the estimates. The red lines in figure 2.4 indicate the carrier loop and the green lines the code loop. Each loop has a numerical controlled oscillator (NCO), controlled by the ranging processor and driven by the reference oscillator [5], which generates local replicas of the carrier frequency or the PRN code.

In a classical code loop, the Code NCO generates three local replicas of the PRN code, known as early (E), prompt(P) and late(L) replicas, which are the same PRN code with different phases. In the acquisition stage only the prompt replica is used to determine the coarse estimate of the code phase. In the tracking stage, all three local replicas are used, for determining the refined code phase component. This type of implementation of the code loop in the tracking stage is known as delay lock loop and it will be explained in details in the next section. The code tracking loop estimate is the one affected by multipath errors.

2.5 Delay Lock Loop and the effects of multipath propagation

In the tracking stage, the PRN code is known. What is not known is the fine alignment between the transmitted PRN code and the locally generated code, which can be between $[-T_c, T_c]$. To determine it, the autocorrelation properties of the PRN code are used. It was shown in equation (2.11) that in a multipath error free environment, the autocorrelation function of the PRN code is symmetric, has maximum value when the signal is not delayed and is zero if the delay is larger then one chip period.

In the DLL, the correlation between two signals is achieved by multiplying the incoming samples and accumulating the results over an observation interval, similar to formula (2.9).

Three different replicas of the PRN code are generated, the early, prompt and late replicas, with different delays, and the DLL is trying to match the received code with the prompt replica. The delay between the early and the late replicas is a design parameter, known as chip correlator spacing, with values between 0.05 and 1 chips, depending on the receiver type. The prompt replica is halfway between them. [5] The DLL correlates the received signal with the each replica and compares the results. If the received PRN code is aligned with the prompt replica, then the correlation value between the received signal and the early replica is equal with the correlation value between the received signal and the late replica. The Code NCO is controlling and changing the delay of the prompt replica until the two mentioned correlation values are equal. This process is represented in figure 2.5.

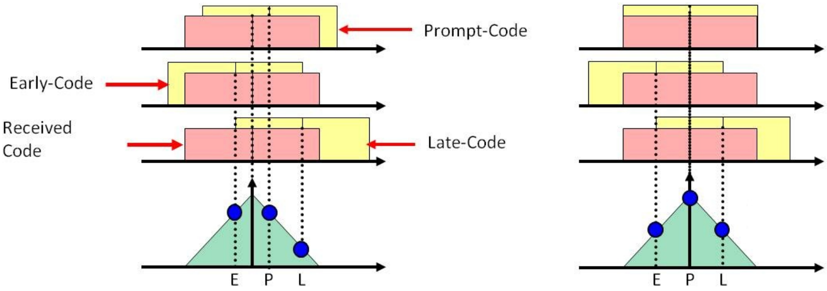


Figure 2.5: Representation of delay estimation principle in a DLL

In a multipath environment, the received signal is a sum of the direct and the reflected components. The reflected components are attenuated, delayed and phase shifted. The correlation function between a multipath affected signal and the locally generated replica is not symmetric and errors might occur in the determination of code phase.

For illustration, we consider the input to one of the BB processing channels, given by a multipath affected BPSK signal, which can be expressed as

$$s_{BB}(t_a) = \sum_{m=0}^{M} a_m c(t_t - \tau_m) d(t_t - \tau_m) e^{j(2\pi f_{DOP} t_a + \phi)} + w(t), \tag{2.17}$$

where w is the noise, M represents the number of reflected components, and a_m and τ_m are the amplitudes and delays corresponding to each component, with $a_0 > a_m$ and $\tau_0 < \tau_m$, for all $m = \{1...M\}$. Exceptions can occur in those relations, due to the propagation of the direct component through an environment that is delaying or attenuating, but we don't consider them here. The rest of the notations are the same as before.

The carrier loop removes the carrier components and the input to the DLL can be expressed as

$$s_{DLL}(t_a) = \sum_{m=0}^{M} a_m c(t_t - \tau_m) d(t_t - \tau_m) + w(t), \tag{2.18}$$

where a_m is a complex amplitude. For simplicity, the data message is assumed to be known or not existent, and it will ignored and the signal is

$$s_{DLL}(t_a) = \sum_{m=0}^{M} a_m c(t_t - \tau_m) + w(t). \tag{2.19}$$

If the noise is ignored, the correlation function between the received signal and a replica of the PRN code, c_{rep}, can be expressed as below

$$\begin{aligned} r_{sc}(\tau) &= \sum_{n=1}^{N} c_{rep}(t_n - \tau) s_{DLL}(t_n) \\ &= \sum_{n=1}^{N} c_{rep}(t_n - \tau)(\sum_{m=0}^{M} a_m c(t_n - \tau_m)) \\ &= \sum_{m=0}^{M} a_m \sum_{n=1}^{N} c_{rep}(t_n - \tau) c(t - \tau_m). \end{aligned} \tag{2.20}$$

Please note that discrete values have been used, as the signal is digital, and the term $1/N$ in front of the summation in the correlation was not used any more, to be aligned with the behaviour of the DLL.

From equation (2.20) it can be noticed that the correlation function of a signal with multipath components is a summation of the correlation functions corresponding to each component from the signal. For the case of $M = 1$, this is illustrated in the next figure.

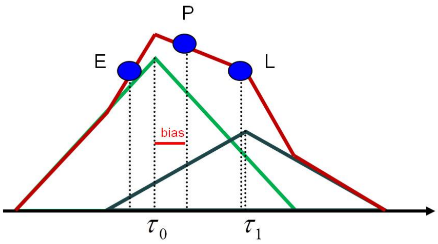


Figure 2.6: The correlation function between the replica PRN code and a signal with one multipath component.

It can be observed that when the correlation values between the received signal and the early and late replicas are equal, the prompt replica is not perfectly aligned with the direct component. This biasing effect is introduced by the multipath signal component, which distorts the correlation function corresponding to only the LOS component. Next, it will be explained how the conventional GNSS receivers are trying to overcome this bias.

2.6 Multipath mitigation in conventional receivers

In the GNSS industry today, there are two main types of receivers with adapted DLL architecture designed to mitigate multipath, known as narrow correlator and double delta correlator. Many variations of those exist, which will not be detailed in this thesis.

The narrow correlator is based on the observation that the bias introduced by the multipath is less sever if the chip correlator spacing is smaller. The band limiting of the signal in the front-end of the receiver is usually rounding the edges of the correlation function and this is limiting how small the chip correlator spacing can get. The narrow correlator has chip correlator spacing between 0.1 and 0.05 chips.

The double delta correlator is based on the observation that the delay of the peak of the correlation function is not affected my the multipath, if lower amplitudes are assumed for the reflected components. Two additional correlators are implemented in the receiver, known as very-early and very-late. The spacing between the very-early and very-late correlators is twice the spacing between the early and late correlators. The results from the early late correlators and from the very-early and very-late correlators are used to find the peak of the correlation function which also coincides with the true delay of the received signal.

In Chapter 5, the performance of DLL, with early late spacing of 0.3 and 0.1, will be compared with the performance of the proposed ML estimation algorithm.

2.7 Signal to noise ratio in GNSS

In GNSS, for expressing the signal power relative to noise, the carrier to noise density ratio (C/N_0) is specified. The carrier to noise density ratio is the ratio of the received carrier power to noise spectral density and is measured in dB Hz.

The terms of signal to noise ration and carrier to noise density ratio are sometimes used interchangeably, but they are not the same [9]. The carrier is the received signal, so both terms, signal/ carrier, can be used. But the difference is given by the noise density, N_0, which is the amount of noise power per unit of bandwidth and is expressed in W/Hz [3], or dB W/Hz. In GNSS, it is convenient to refer to the signal to noise density ratio, because it does not depend on the bandwidth.

The bandwidth B is included in the carrier to noise ratio (C/N), as below

$$\frac{C}{N} = \frac{C}{BN_0}. \tag{2.21}$$

In dB, the ratio becomes

$$\left(\frac{C}{N}\right)_{dB} = \left(\frac{C}{N_0}\right)_{dBHz} - 10\log_{10}(B). \tag{2.22}$$

Considering the sample frequency f_s equal with the bandwidth

$$\left(\frac{C}{N}\right)_{dB} = \left(\frac{C}{N_0}\right)_{dBHz} - 10\log_{10}(f_s). \tag{2.23}$$

The carrier power C is the power of the received signal, which, in the absence of multipath components (M = 0), is

$$\begin{aligned} C = P_{s_{DLL}} &= E[a_0 c(t-\tau_0)(a_0 c(t-\tau_0))^*] \\ &= |a_0|^2 E[c(t-\tau_0)c^*(t-\tau_0)] = |a_0|^2 E[1] \\ &= |a_0|^2, \end{aligned} \tag{2.24}$$

where $E[\cdot]$ is the expected value operator. We computed the power using the expected value, due to the pseudo-random characteristic of the PRN code.

For complex white noise, with standard deviation of the real and imaginary parts given by σ, the noise power is given by

$$P_{noise} = 2\sigma^2. \tag{2.25}$$

The SNR, for one signal component (M = 0), is given by

$$\text{SNR} = \frac{C}{N} = \frac{|a_0|^2}{2\sigma^2}, \tag{2.26}$$

where $2\sigma^2 = BN_0$

When multipath components are part of the signal ($M > 0$), the total SNR will change. As a convention in the thesis, the term SNR will be used for the SNR of the direct component. When this will not be the case, it will be clear from the text.

In GNSS, the SNR from equation (2.26) is negative. After the correlation of the received signal with the replica PRN code in the DLL, the SNR is increased to a positive level. The SNR after correlation will be equal with the SNR before the correlation multiplied with a factor of N, where N represents the total number of samples in the observation interval, as it is shown in appendix A.

Chapter 3

Maximum Likelihood Estimation

In this chapter, an estimation algorithm, based on the maximum likelihood estimator, will be proposed and developed for the estimation of the delays of the line of sight and the multipath components in the GNSS signals. The main difference between the classical DLL and the maximum likelihood estimation is that the DLL is trying to minimize the error induced by the multipath. The ML estimator proposed, however, performs joint estimation of the delays of all multipath components present in the signal, therefore avoiding the error introduced by multipath signal components which are not accounted for in the DLL.

The maximum likelihood estimation is a very popular method within statistical signal processing, due to its asymptotic optimality: as the number of observations used by the estimator tends to infinity, the estimator becomes unbiased and achieves the Cramér-Rao lower bound (CRLB)[7]. Asymptotically unbiased estimator means that the mean of the estimate resulting from a number of realizations tends to the true values of the parameters to be estimated, as the number of realizations goes to infinity. The fact that the estimate asymptotically achieves the CRLB means that the variance of the estimator achieves its minimum possible limit, the CRLB, as the number of observations tends to infinity. In addition, for estimation problems affected by noise, the CRLB can be achieved even for relatively small data sets if the SNR is large enough [7]. Weill has researched and documented the relation between the maximum likelihood estimator for GNSS signals and the CRLB, and more about it can be found in [13], [12] and [14]. The CRLB is not a topic for this thesis, so it will not be discussed further. The goal of the this thesis is to asses if, and when, the maximum likelihood estimator performs better than the classical DLL and and this will be detailed in Chapter 5.

Two important directions of research for the maximum likelihood estimator in GNSS can be found in literature, the one from Selva Vera, in [11], and the one from Weill, in [15], and they are the main reference for this chapter and chapter 4. Similar to Selva [11], we are going to use the vector form of the signals, as we consider it straightforward and easy to generalize for different number of multipath components. Similar to Weill [15], and to the development of his work for the Vision Correlator [10], we will consider constraints on the amplitudes, as presented in section 3.4, based on the method proposed by Weill, with Lagrange multipliers, but also with a different method, developed in this thesis.

3.1 Signal model

The maximum likelihood principle states that the maximum likelihood estimator is the one that maximizes the likelihood function, i.e. the probability density function (PDF) of the observations considered as a function of the parameters to estimate [7]. In order to formulate the likelihood function, it is very important to asses a correct signal model and this is the subject of this section.

Before stating a signal model, some assumptions have to be made. The signal arriving at the receiver is affected by multipath propagation caused by the scatterers in the environment. We assume that the receiver and the environment are static during one observation time and, hence, so is the multipath channel response. Only one channel in the GNSS receiver is considered, meaning that only one type of signal from only one satellite is analysed. Furthermore, the carrier frequency and Doppler offset have been corrected by the carrier loop. For simplicity, the navigation data message is considered to be either known, removed or not existent, as in a pilot signal, and it will be ignored. In the case when the navigation message is not ignored and not known, the observation time is aligned so that a bit transition in the message data occurs at the border of an observation interval, not during the observation interval. Under this assumption, the message data only impacts the sign of the correlation result.

At the point in the receiver where the ML estimation is taking place, the signal has been converted from analogue to digital domain, so we will work with a discrete-time signal model. The signal samples are part of a data set, also known as time series, collected during an observation interval, and are denoted by $\{s_1, s_2, ..., s_N\}$, with one signal sample s_n given by

$$s_n \equiv s(t_n) \equiv s((n-1)T_s), \tag{3.1}$$

where t_n is the time instant corresponding to the n'th sample, equal with $(n-1)T_s$, where T_s is the sample period.

Assuming that M multipath components are reaching the receiver at the moment of observation, the signal model for the n'th sample can be formulated as

$$s(t_n) = \sum_{m=0}^{M} a_m c(t_n - \tau_m) + w(t_n), \tag{3.2}$$

where $s(t_n)$ is the received complex„ baseband signal, with the carrier frequency and the Doppler offset removed, $w(t_n)$ is a sample of a complex zero-mean Gaussian noise, $c(t_n)$ is sample of the PRN code, τ_0 and a_0 are the delay and amplitude of the line of sight component, τ_m and a_m, for $m > 0$, are the delays and amplitudes of the multipath components. The amplitudes a_m are complex and can be expressed in polar form as

$$a_m = A_m e^{j\phi_m}, \tag{3.3}$$

where A_m and ϕ_m are the magnitude and phase of the m'th signal component.

If the N samples from an observation interval are ordered in a vector, then the signal model can be expressed as below

$$\begin{bmatrix} s(0T_s) \\ s(1T_s) \\ \vdots \\ s((N-1)T_s) \end{bmatrix} = \begin{bmatrix} c(0T_s-\tau_0) & c(0T_s-\tau_1) & \dots & c(0T_s-\tau_M) \\ c(1T_s-\tau_0) & c(1T_s-\tau_1) & \dots & c(1T_s-\tau_M) \\ \vdots & \vdots & \ddots & \vdots \\ c((N-1)T_s-\tau_0) & c((N-1)T_s-\tau_1) & \dots & c((N-1)T_s-\tau_M) \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_M \end{bmatrix} + \begin{bmatrix} w(0T_s) \\ w(1T_s) \\ \vdots \\ w((N-1)T_s) \end{bmatrix} \tag{3.4}$$

and more compactly as

$$\boldsymbol{s} = \boldsymbol{C}(\boldsymbol{\tau})\boldsymbol{a} + \boldsymbol{w}, \tag{3.5}$$

where

$$\begin{aligned} &\boldsymbol{s} = [s_1\ s_2\ \dots\ s_N]^T, \\ &\boldsymbol{a} = [a_0\ a_1\ \dots\ a_M]^T, \\ &\boldsymbol{\tau} = [\tau_0\ \tau_1\ \dots\ \tau_M]^T, \\ &\boldsymbol{C}(\boldsymbol{\tau}) = [\boldsymbol{c}(\tau_0)\ \boldsymbol{c}(\tau_1)\ \dots\ \boldsymbol{c}(\tau_M)] \text{ with} \\ &\boldsymbol{c}(\tau_m) = [c_1(\tau_m)\ c_2(\tau_m)\ \dots\ c_N(\tau_m)]^T \text{ and } c_n(\tau_m) \equiv c((n-1)T_s - \tau_m), \\ &\boldsymbol{w} = [w_1\ w_2\ \dots\ w_N]^T \text{ with } w_n \equiv w((n-1)T_s). \end{aligned} \tag{3.6}$$

3.2 Cost Function

The signal model in (3.2) is a mathematical expression describing the observed signal $\boldsymbol{s}$, as a function of a set of parameters, which are either known or can be determined or estimated. In the present case, the delay of the line of sight τ_0 is the parameter of interest, while the rest of the delays τ_m, with $m > 0$, are just means to estimate the parameters of interest, being known as nuisance parameters. The PRN code $\boldsymbol{c}$ is known to the receiver, the noise $\boldsymbol{w}$ is assumed to be a complex zero-mean Gaussian noise with uncorrelated samples and the complex amplitudes a_m are unknown and need to be determined together with the delays. In practice, the number of multipath components is not known, and is difficult to approximate, but for now M is assumed to be known and later in the thesis it will be investigated how a bad approximation is affecting the estimation results.

For the ease of mathematical representation, the unknown parameters are included in the vector $\boldsymbol{\theta}$, as follows

$$\boldsymbol{\theta} = [\boldsymbol{a}^T, \boldsymbol{\tau}^T] = [a_0,\ \dots\ ,\ a_M, \tau_0,\ \dots\ , \tau_M]. \tag{3.7}$$

In ML estimation, the estimate $\hat{\boldsymbol{\theta}}_{ML}$ is obtained by maximizing the likelihood function, given by the probability density function (PDF) of $\boldsymbol{s}$, as a function of unknown parameters in $\boldsymbol{\theta}$. For a Gaussian distribution, the PDF for one complex sample is

$$p(s_n) = \left(\frac{1}{\pi\sigma_s^2}\right) \exp\left(-\frac{1}{\sigma_s^2}\left|s_n - \mu_s\right|^2\right) \tag{3.8}$$

where σ_s is the standard deviation of s_n and μ_s is the mean, or expected value of s_n.

According to (3.2), the signal s_n is composed of a deterministic part, given by

$$\sum_{m=0}^{M} a_m c_n(\tau_m),$$

and a stochastic part, given by the noise component, with variance σ^2 and the mean 0. The mean of one sample s_n will be given by

$$\mu_s = E[s_n] = E[\sum_{m=0}^{M} a_m c_n(\tau_m) + w_n] \tag{3.9a}$$

$$= E[\sum_{m=0}^{M} a_m c_n(\tau_m)] + E[w_n] \tag{3.9b}$$

$$= \sum_{m=0}^{M} a_m c_n(\tau_m) + 0, \tag{3.9c}$$

where $E[\cdot]$ denotes the expected value operator.

And the variance of s_n will be

$$\sigma_n^2 = \mathrm{var}(s_n) = \mathrm{var}(\sum_{m=0}^{M} a_m c_n(\tau_m) + w_n) \tag{3.10a}$$

$$= \mathrm{var}(\sum_{m=0}^{M} a_m c_n(\tau_m)) + \mathrm{var}(w_n) \tag{3.10b}$$

$$= 0 + \sigma^2, \tag{3.10c}$$

The expression for $p(s_n)$ becomes

$$p(s_n) = \left(\frac{1}{\sqrt{\pi}\sigma}\right)^2 \exp\left(-\frac{1}{\sigma^2}\left|s - \sum_{m=0}^{M} a_m c(\tau_m)\right|^2\right) \tag{3.11}$$

The PDF of the vector $\boldsymbol{s}$ is given by the product of the PDF's for each sample, because the noise samples are uncorrelated. And because the likelihood function is equal to the PDF of $\boldsymbol{s}$, the likelohood function is

$$L(\boldsymbol{s};\boldsymbol{\theta}) = \left(\frac{1}{\sqrt{\pi}\sigma}\right)^{2N} \exp\left(-\frac{1}{\sigma^2}\sum_{n=1}^{N}\left|s_n - \sum_{m=0}^{M} a_m c_n(\tau_m)\right|^2\right) \tag{3.12}$$

Thus, the ML estimate of $\boldsymbol{\theta}$ is given by

$$\hat{\boldsymbol{\theta}}_{ML} = \arg\max_{\boldsymbol{\theta}} L(\boldsymbol{s};\boldsymbol{\theta}). \tag{3.13}$$

By combining (3.12) and (3.13) the expression for $\hat{\boldsymbol{\theta}}_{ML}$ is

$$\hat{\boldsymbol{\theta}}_{ML} = \arg\max_{\boldsymbol{\theta}} \exp\left(-\frac{1}{\sigma^2}\sum_{n=1}^{N}\left|s_n - \sum_{m=0}^{M} a_m c_n(\tau_m)\right|^2\right). \tag{3.14}$$

The term in front of the exponent from (3.12) was dropped because it does not depend on $\boldsymbol{\theta}$. As the logarithm is a monotonically increasing function, it can be applied to the argument of the argmax operation, without changing the location of the extremum, leading to

$$\hat{\boldsymbol{\theta}}_{ML} = \arg\max_{\boldsymbol{\theta}} \left(-\frac{1}{\sigma^2} \sum_{n=1}^{N} \left| s_n - \sum_{m=0}^{M} a_m c_n(\tau_m) \right|^2 \right) \tag{3.15a}$$

$$= \arg\min_{\boldsymbol{\theta}} \sum_{n=1}^{N} \left| s_n - \sum_{m=0}^{M} a_m c_n(\tau_m) \right|^2 , \tag{3.15b}$$

which can also be expressed as

$$\hat{\boldsymbol{\theta}}_{ML} = \arg\min_{\boldsymbol{\theta}} \,||\, \boldsymbol{s} - \boldsymbol{C}(\boldsymbol{\tau})\boldsymbol{a} \,||^2 \, . \tag{3.16}$$

The term that is minimized in (3.16) is defined as the cost function for the parameters $\boldsymbol{\theta}$, i.e.

$$J(\boldsymbol{\theta}) =||\, \boldsymbol{s} - \boldsymbol{C}(\boldsymbol{\tau})\boldsymbol{a} \,||^2 \, . \tag{3.17}$$

It can be noticed that the ML minimization has become a Least Squares minimization problem. This is always the case in the ML estimation problems where the noise has Gaussian distribution $\mathcal{N}(\boldsymbol{0}, \sigma^2 \boldsymbol{I})$ [7]. How the minimization can be interpreted now is that the solution will give the parameters that best fit the noiseless signal model to the received signal. And that the squared error between each point from the received signal and the fitted model is minimized as much as possible.

The minimization of J with respect to $\boldsymbol{\theta}$ is $2M$ dimensional, has a quadratic dependence on $\boldsymbol{a}$ and a non-linear dependence on the delay parameters in $\boldsymbol{\tau}$. The minimization is challenging because of the high dimensions and because of the non-linear dependence on $\boldsymbol{\tau}$. But the quadratic dependence on the complex amplitudes in $\boldsymbol{a}$ can be exploited and a close form expression for $\boldsymbol{a}$ can be derived. This will reduce the problem from $2M$ dimensional to M dimensional, depending only on the delays parameters vector $\boldsymbol{\tau}$. For the estimation of $\boldsymbol{\tau}$, a grid search can be performed, at a resolution high enough. Or it can be estimated in two stages, first over a course grid and after that improve the estimation by performing an iterative search around the initial solution.

3.3 Parameters Estimation

In this section it will be detailed how the parameters in $\hat{\boldsymbol{\theta}}_{ML}$ can be estimated, by solving

$$\hat{\boldsymbol{\theta}}_{ML} = \arg\min_{\boldsymbol{\theta}} J(\boldsymbol{\theta}). \tag{3.18}$$

The norm 2 in (3.17) can be expressed as $||\, \boldsymbol{x} \,||^2 = \boldsymbol{x}^H \boldsymbol{x}$, where $\boldsymbol{x}$ is a complex vector and $(\cdot)^H$ denotes conjugate transposition. Using this, and by keeping the matrix, vector notation, the cost function in (3.17) can be extended as

$$\begin{aligned} J(\boldsymbol{\theta}) &= \left(\boldsymbol{s}^H - \boldsymbol{a}^H \boldsymbol{C}(\boldsymbol{\tau})^H\right)\left(\boldsymbol{s} - \boldsymbol{C}(\boldsymbol{\tau})\boldsymbol{a}\right) \\ &= \boldsymbol{s}^H\boldsymbol{s} - \boldsymbol{s}^H\boldsymbol{C}(\boldsymbol{\tau})\boldsymbol{a} - \boldsymbol{a}^H\boldsymbol{C}(\boldsymbol{\tau})^H\boldsymbol{s} + \boldsymbol{a}^H\boldsymbol{C}(\boldsymbol{\tau})^H\boldsymbol{C}(\boldsymbol{\tau})\boldsymbol{a}. \end{aligned} \tag{3.19}$$

By dropping $\boldsymbol{s}^H\boldsymbol{s}$ due to its independence of $\boldsymbol{\theta}$, the cost function can expressed

$$J(\boldsymbol{\theta}) \propto J_1(\boldsymbol{\theta}) = -\boldsymbol{s}^H \boldsymbol{C}(\boldsymbol{\tau})\boldsymbol{a} - \boldsymbol{a}^H \boldsymbol{C}(\boldsymbol{\tau})^H \boldsymbol{s} + \boldsymbol{a}^H \boldsymbol{C}(\boldsymbol{\tau})^H \boldsymbol{C}(\boldsymbol{\tau})\boldsymbol{a}, \tag{3.20}$$

where $\propto$ indicates proportionality.

3.3.1 Amplitudes determination

In order to determine a close form expression for the parameters in $\boldsymbol{a}$, the rest of the parameters in $\hat{\boldsymbol{\theta}}_{ML}$ will be considered to be known. Due to the quadratic relation between J and $\boldsymbol{a}$, for any fixed value of $\boldsymbol{\tau}$, the value of $\boldsymbol{a}$ that minimizes the cost function can be found, by taking the gradient of J with respect to $\boldsymbol{a}$ and setting it to 0.

Because $\boldsymbol{a}$ is complex, the derivative needs to be computed as below

$$\nabla_{\boldsymbol{a}} J = \begin{bmatrix} \frac{\partial J}{\partial \boldsymbol{a}} & \frac{\partial J}{\partial \boldsymbol{a}^*} \end{bmatrix}, \tag{3.21}$$

where $(\cdot)^*$ denotes conjugate.

By solving $\nabla_{\boldsymbol{a}} J(\boldsymbol{a}_{ML}, \boldsymbol{\tau}) = \boldsymbol{0}$, two sets of equations are obtained, both resulting in the same equation, as it is shown in appendix B,

$$\boldsymbol{C}(\boldsymbol{\tau})^H \boldsymbol{C}(\boldsymbol{\tau})\boldsymbol{a}_{ML} = \boldsymbol{C}(\boldsymbol{\tau})^H \boldsymbol{s}. \tag{3.22}$$

If $\boldsymbol{C}(\boldsymbol{\tau})^H \boldsymbol{C}(\boldsymbol{\tau})$ is invertible, the solution for the complex amplitude vector, $\boldsymbol{a}_{ML}$, has the following form

$$\boldsymbol{a}_{ML}(\boldsymbol{\tau}) = \left(\boldsymbol{C}(\boldsymbol{\tau})^H \boldsymbol{C}(\boldsymbol{\tau})\right)^{-1} \boldsymbol{C}(\boldsymbol{\tau})^H \boldsymbol{s}. \tag{3.23}$$

One remark needs to be made. In the estimation process, the amplitudes determined are complex and if needed, the magnitude and the phase can be calculated, by using the below relations [15]

$$a = a^R + ja^I, \tag{3.24a}$$
$$A = |a| = \sqrt{(a^R)^2 + (a^I)^2}, \tag{3.24b}$$
$$\phi = \operatorname{atan2}(a^R, a^I), \tag{3.24c}$$

where the superscripts R and I are indicating the real and imaginary parts, and a is any of the complex amplitudes from the vector $\boldsymbol{a}_{ML}(\boldsymbol{\tau})$.

If expression (3.23) is used in (3.17), the cost function becomes

$$J(\boldsymbol{\tau})|_{\boldsymbol{a}=\boldsymbol{a}_{ML}} =|| \ \boldsymbol{s} - \boldsymbol{C}(\boldsymbol{\tau}) \left(\boldsymbol{C}(\boldsymbol{\tau})^H \boldsymbol{C}(\boldsymbol{\tau})\right)^{-1} \boldsymbol{C}(\boldsymbol{\tau})^H \boldsymbol{s} \ ||^2 \tag{3.25}$$

Or, if (3.23) is used in the extended expression from (3.20), then a simplified form of the cost function is obtained, as follows

$$\begin{aligned} J_1(\boldsymbol{\tau})|_{\boldsymbol{a}=\boldsymbol{a}_{ML}} =& - \boldsymbol{s}^H \boldsymbol{C}(\boldsymbol{\tau}) \left(\boldsymbol{C}(\boldsymbol{\tau})^H \boldsymbol{C}(\boldsymbol{\tau})\right)^{-1} \boldsymbol{C}(\boldsymbol{\tau})^H \boldsymbol{s} - \boldsymbol{s}^H \boldsymbol{C}(\boldsymbol{\tau}) \left(\boldsymbol{C}(\boldsymbol{\tau})^H \boldsymbol{C}(\boldsymbol{\tau})\right)^{-1} \boldsymbol{C}(\boldsymbol{\tau})^H \boldsymbol{s} \\ &+ \boldsymbol{s}^H \boldsymbol{C}(\boldsymbol{\tau}) \left(\boldsymbol{C}(\boldsymbol{\tau})^H \boldsymbol{C}(\boldsymbol{\tau})\right)^{-1} \boldsymbol{C}(\boldsymbol{\tau})^H \boldsymbol{C}(\boldsymbol{\tau}) \left(\boldsymbol{C}(\boldsymbol{\tau})^H \boldsymbol{C}(\boldsymbol{\tau})\right)^{-1} \boldsymbol{C}(\boldsymbol{\tau})^H \boldsymbol{s}, \end{aligned} \tag{3.26a}$$

$$= -2\boldsymbol{s}^H \boldsymbol{C}(\boldsymbol{\tau}) \left(\boldsymbol{C}(\boldsymbol{\tau})^H \boldsymbol{C}(\boldsymbol{\tau})\right)^{-1} \boldsymbol{C}(\boldsymbol{\tau})^H \boldsymbol{s} + \boldsymbol{s}^H \boldsymbol{C}(\boldsymbol{\tau}) \left(\boldsymbol{C}(\boldsymbol{\tau})^H \boldsymbol{C}(\boldsymbol{\tau})\right)^{-1} \boldsymbol{C}(\boldsymbol{\tau})^H \boldsymbol{s}, \tag{3.26b}$$

resulting in

$$J_1(\boldsymbol{\tau})|_{\boldsymbol{a}=\boldsymbol{a}_{ML}} = -\boldsymbol{s}^H \boldsymbol{C}(\boldsymbol{\tau}) \left(\boldsymbol{C}(\boldsymbol{\tau})^H \boldsymbol{C}(\boldsymbol{\tau})\right)^{-1} \boldsymbol{C}(\boldsymbol{\tau})^H \boldsymbol{s}, \tag{3.27a}$$

which can also be expressed as

$$J_1(\boldsymbol{\tau})|_{\boldsymbol{a}=\boldsymbol{a}_{ML}} = -\boldsymbol{s}^H \boldsymbol{C}(\boldsymbol{\tau}) \boldsymbol{a}_{ML}(\boldsymbol{\tau}). \tag{3.27b}$$

The expression for the cost function does not depend on the amplitudes any more. This means that to a certain value of $\boldsymbol{\tau}$ corresponds a certain value of J and J_1.

3.3.2 Delays estimation

In order the find the optimum vector of delays $\hat{\boldsymbol{\tau}}_{ML}$, the most straightforward approach is to perform a grid search in the region where the delays are expected to be. To increase the accuracy of the estimation, the resolution of the grid can be increased, or an iterative search can be performed around the initial result. But for now, we will focus only on the grid search, performed at a resolution corresponding to the sampling period T_s.

The optimum $\hat{\boldsymbol{\tau}}_{ML}$ is the one that minimizes the below expression,

$$\hat{\boldsymbol{\tau}}_{ML} = \arg\min_{\boldsymbol{\tau}} J_1(\boldsymbol{\tau})|_{\boldsymbol{a}=\boldsymbol{a}_{ML}}. \tag{3.28}$$

If we define the set $\mathcal{T}$, containing all possible combination of delays in the search grid, the approximate ML estimate can be defined as:

$$\hat{\boldsymbol{\tau}}_{ML,grid} = \arg\min_{\boldsymbol{\tau}\in\mathcal{T}} J_1(\boldsymbol{\tau})|_{\boldsymbol{a}=\boldsymbol{a}_{ML}}. \tag{3.29}$$

In the above expression, we simply restrict the values that $\boldsymbol{\tau}$ can take in the argmin operation to those present in the set $\mathcal{T}$, i.e., to the points in the grid. The ML estimation takes place in the receiver in the tracking stage. And in the tracking stage the incoming signal is already coarsely aligned with the PRN replica code, with an uncertainty of $+/-$ one chip. The search interval for all delays is $[-T_c, T_c]$, where T_c is the chip period.

Another restriction on the delays is that the line of sight delay is the smallest, and the multipath delays follow the below relation.

$$\tau_0 < \tau_1 < ... < \tau_M. \tag{3.30}$$

The constraints in (3.30) are introduced in the elements of the set $\mathcal{T}$.

3.3.3 Algorithm overview

As an overview of the estimation algorithm, a pseudo-code is generated.

Pseudo-code:

```
1 : Create a set $\mathcal{T}$ of all possible combinations of delays, $\boldsymbol{\tau}_i$, which are in the interval $[-T_c, T_c]$ and respect the condition $\tau_0 < \tau_1 < ... < \tau_M$. The index $i$ indicates the position in the set.
# For each $\boldsymbol{\tau}_i \in \mathcal{T}$
```

```
2 : Compute $\boldsymbol{a}_{ML}(\boldsymbol{\tau}_i) = \left(\boldsymbol{C}(\boldsymbol{\tau}_i)^H\boldsymbol{C}(\boldsymbol{\tau}_i)\right)^{-1}\boldsymbol{C}(\boldsymbol{\tau}_i)^H\boldsymbol{s}$
3 : Compute $J_1(\boldsymbol{\tau}_i) = -\boldsymbol{s}^H\boldsymbol{C}(\boldsymbol{\tau}_i)\boldsymbol{a}_{ML}(\boldsymbol{\tau}_i)$.
# Minimization
4 : Find the index $k$ yielding the lowest value $J_1(\boldsymbol{\tau}_i)$
5 : $\hat{\boldsymbol{\tau}}_{ML} = \boldsymbol{\tau}_k$
```

Assuming that the number of multipath components is known and the environment is not affected by noise, the ML estimation should always find the good solution, due to the optimality of the ML estimator. But because some multipath delays might be much closer to the line of sight delay comparing to the resolution of the search space and because the estimation is not allowed by the relation in (3.30) to search for equal delays, some errors are to be expected.

If the situation of $M = 1$ is considered, in which the delay of the multipath is so close to the delay of the line of sight that can be considered equal, and for the particular case in which $\tau_0 = \tau_1 = 0$ chips, the normalized cost function looks like in figure 3.1.

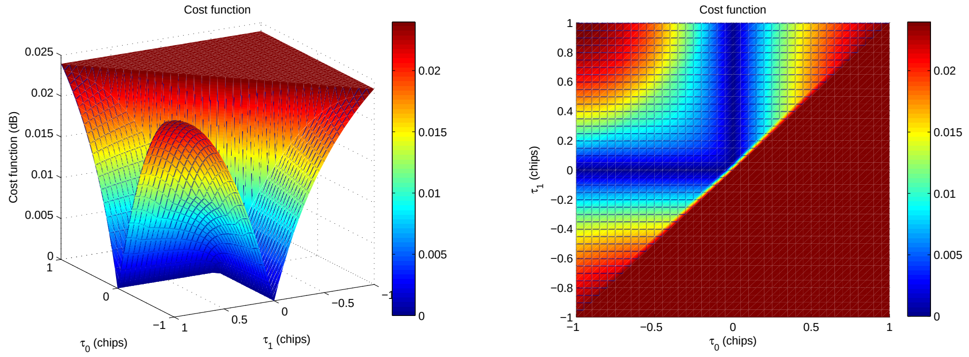


Figure 3.1: The normalized cost function, in the case when $\tau_0 = \tau_1 = 0$ chips, from two different view angles.

From the estimation point of view, all the delay combinations, in which one of the delays is on the true value and the other one is in a region that respects the condition in (3.30), have equal values of the cost function. For the special case in which $\tau_0 = \tau_1 = 0$ chips, the cost function has equal values, which are also the minimum value, for τ_0 in $[-Tc, 0 - Ts]$ and $\tau_1 = 0$ chips and for $\tau_0 = 0$ chips and τ_1 in $[0 + T_s, Tc]$. And this can be extended for more multipath delays very close to each other, with the same conditions: only one of the delays will be assigned to the true value. The process that is taking place in order for this to happen is that the estimation is assigning a high value to the amplitude corresponding to the component which is on the true value and a very low value to the rest of the amplitudes. When the components are summed up, because of the difference in amplitudes, so in power, the component that is on the true delay is almost masking the the rest of the components.

In practice, the values that the multipath delays get from the estimation are not important. What is important is that the line of sight delay is on the true value. The cost function can be adjusted, so that the LOS component is always the component with larger amplitude. This can be done by imposing some constraints on the amplitudes and it will be discussed in the next section.

3.4 Parameters Estimation with constrained amplitudes

3.4.1 Amplitude constraints

A simple restriction on the amplitudes would be given by the relation $A_0 > A_m$, with $m > 0$, considering that the direct component is usually the strongest. In practice, there can be exceptions, which will be ignored in the thesis. With some prior knowledge about the relation between the amplitudes, this restriction can be made more strict.

In GNSS, the power of line of sight component is at least $3 - 4$ dB higher than the power of the multipath components. Usually the difference is between 6 and 10 dB. Considering the signal model in (3.2), the line of sight component is $a_0 c_n(\tau_0)$ and its power is given by

$$P_0 = A_0^2,$$

as it was shown in (2.24).

The same applies to the first multipath and its power will be $P_1 = A_1^2$.

The ratio between the real part of the amplitude of the first multipath and the amplitude of the line of sight is

$$\frac{A_1}{A_0} = \sqrt{\frac{P_1}{P_0}} = \frac{1}{\sqrt{10^{(\frac{\Delta}{10})}}}, \tag{3.31}$$

where Δ represents the difference in dB between the power of the line of sight and the power of the first multipath component.

Figure 3.2 shows how the amplitudes ratio changes with the value of Δ.

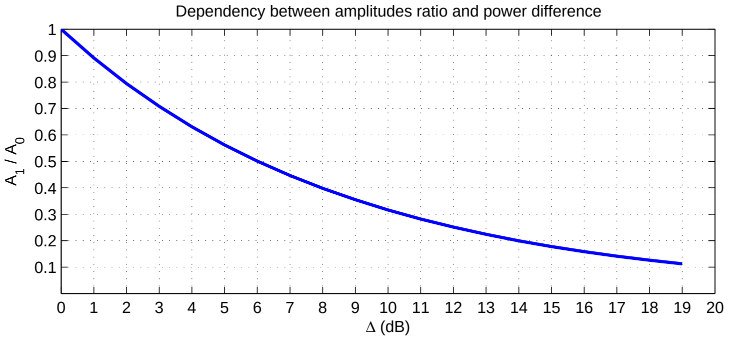


Figure 3.2: Dependency between amplitudes ratio and power difference.

In order to make the constraint on the amplitudes more stringent, it can be enforced that

$$\frac{A_1}{A_0} \leq \alpha, \tag{3.32}$$

where the value of α is a tuning parameter, but in general it is safe to set $\alpha = 0.7$ [15], which corresponds (roughly) to $\Delta = 3$ dB.

In order to use the amplitudes constraint in the estimation process, a relation between the complex amplitudes is needed. From equation (3.24b) results that $A^2 = |a|^2 = (a^R)^2 + (a^I)^2$, and (3.32) can be reformulated as

$$\frac{|a_1|^2}{|a_0|^2} \leq \alpha^2. \tag{3.33}$$

Considering (3.33), the new relation between the amplitudes is

$$\alpha^2|a_0|^2 \geq |a_1|^2, \tag{3.34a}$$

$$...$$

$$\alpha^2|a_0|^2 \geq |a_M|^2. \tag{3.34b}$$

In vector form, this constraint can be expressed as

$$\boldsymbol{a}^H\boldsymbol{\Omega}_1\boldsymbol{a} \geq 0, \tag{3.35a}$$

$$...$$

$$\boldsymbol{a}^H\boldsymbol{\Omega}_M\boldsymbol{a} \geq 0, \tag{3.35b}$$

where $\boldsymbol{\Omega}_1$, ... $\boldsymbol{\Omega}_M$ are diagonal matrices given by

$$\boldsymbol{\Omega}_1 = \begin{bmatrix} \alpha^2 & 0 & \cdots & 0 \\ 0 & -1 & \cdots & 0 \\ \vdots & \vdots & 0\ddots & \vdots \\ 0 & 0 & \cdots & 0 \end{bmatrix}, ... \,\boldsymbol{\Omega}_M = \begin{bmatrix} \alpha^2 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots 0 & \vdots \\ 0 & 0 & \cdots & -1 \end{bmatrix} \tag{3.35c}$$

A relaxation can be achieved by summing up the subequations in (3.34), resulting in

$$M\alpha^2|a_0|^2 \geq |a_1|^2 + ... + |a_M|^2, \tag{3.36}$$

and in vector form can be expressed as

$$\boldsymbol{a}^H\boldsymbol{\Omega}\boldsymbol{a} \geq 0, \tag{3.37a}$$

where $\boldsymbol{\Omega}$ is a diagonal matrix given by

$$\boldsymbol{\Omega} = \begin{bmatrix} M\alpha^2 & 0 & \cdots & 0 \\ 0 & -1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & -1 \end{bmatrix} \tag{3.37b}$$

Depending on how strict the constraint needs to be, one of the constraints presented, (3.34) or (3.36), can be used. For the case of equal delays, which was presented in subsection 3.3.3, the constrain in (3.36) is enough to make the difference and the estimation to give the correct result for the line of sight delay.

3.4.2 Constrained minimization

Two methods to compute the ML cost function, subject to the amplitude constraints, will be presented in this subsection, and in chapter 5 those methods will be compared for efficiency.

Brut Force Constraints This is a straightforward method and it is based on the idea that the values of the cost function, in the regions where the amplitudes do not respect the constraints, are not needed. More detailed, that when the amplitudes produced do not respect the constraints in (3.34), it is for those combinations of delays in which the true value of the line of sight delay is closer to another component than to the line of sight component. To illustrate the method, the pseudo-code is presented.

Pseudo-code:

```
1 : Create a set $\mathcal{T}$ of all possible combinations of delays, $\boldsymbol{\tau}_i$, which are in the interval $[-T_c, T_c]$ and respect the condition $\tau_0 < \tau_1 < ... < \tau_M$. The index $i$ indicates the position in the set.
# For each $\boldsymbol{\tau}_i$
2 : Compute $\boldsymbol{a}_{ML}(\boldsymbol{\tau}_i) = \left(\boldsymbol{C}(\boldsymbol{\tau}_i)^H \boldsymbol{C}(\boldsymbol{\tau}_i)\right)^{-1} \boldsymbol{C}(\boldsymbol{\tau}_i)^H \boldsymbol{s}$
3 : Check if $\left\{\boldsymbol{a}_{ML}(\boldsymbol{\tau}_i)^H \boldsymbol{\Omega}_1 \boldsymbol{a}_{ML}(i), ... \, \boldsymbol{a}_{ML}(i)^H \boldsymbol{\Omega}_M \boldsymbol{a}_{ML}(i)\right\} \geq 0$
4 : If yes, compute $J_1(\boldsymbol{\tau}_i) = -\boldsymbol{s}^H \boldsymbol{C}(\boldsymbol{\tau}_i) \boldsymbol{a}_{ML}(\boldsymbol{\tau}_i)$
5 : If not, set $J_1(\boldsymbol{\tau}_i) = 0$.
# Minimization
6 : Find the index $k$ yielding the lowest value $J_1(\boldsymbol{\tau}_i)$
7 : $\hat{\boldsymbol{\tau}}_{ML} = \boldsymbol{\tau}_k$
```

Lagrange constraints The cost function which will be considered is the one from (3.20), restated here for convenience

$$J_1(\boldsymbol{\theta}) = -\boldsymbol{s}^H \boldsymbol{C}(\boldsymbol{\tau})\boldsymbol{a} - \boldsymbol{a}^H \boldsymbol{C}(\boldsymbol{\tau})^H \boldsymbol{s} + \boldsymbol{a}^H \boldsymbol{C}(\boldsymbol{\tau})^H \boldsymbol{C}(\boldsymbol{\tau})\boldsymbol{a}. \tag{3.38}$$

With Lagrange, implementing the constraint from (3.35) is much complicated than implementing the constraint from (3.37). In order to exemplify the procedure to be followed, we choose the constraint that is simple to treat analytically, the one in (3.37).

$$h(\boldsymbol{a}) = \boldsymbol{a}^H \boldsymbol{\Omega} \boldsymbol{a}, \tag{3.39}$$

with $\boldsymbol{\Omega}$ as in (3.37b), then the ML optimization with Lagrange constraints can be formulated as

$$\begin{aligned} \hat{\boldsymbol{\theta}}_{ML,L} = &\arg\min_{\boldsymbol{\theta}} J_1(\boldsymbol{\theta}), \\ &\text{subject to } h(\boldsymbol{a}_{ML,L}) \geq 0, \end{aligned} \tag{3.40}$$

where $\hat{\boldsymbol{\theta}}_{ML,L} = [\hat{\boldsymbol{\tau}}_{ML,L},\ \boldsymbol{a}_{ML,L}]$.

The constrain can be incorporated in an objective function, called the Lagrangian, as below

$$L(\boldsymbol{\theta}, \lambda) = J_1(\boldsymbol{\theta}) - \lambda h(\boldsymbol{a}), \tag{3.41}$$

with $\lambda \geq 0$.

The solution is given by

$$\hat{\boldsymbol{\theta}}_{ML,L} = \arg\min_{\boldsymbol{\theta}} L(\boldsymbol{\theta}, \lambda_L), \tag{3.42}$$

where λ_L is the optimal value for λ.

The problem in (3.42) has a nonlinear dependency on $\boldsymbol{\tau}$ and a quadratic dependency on $\boldsymbol{a}$, and the same as in minimization with unconstrained amplitudes, we can freeze $\boldsymbol{\tau}$ and find a close form expression for $\boldsymbol{a}$. For a fixed $\boldsymbol{\tau}$, the minimization becomes

$$\boldsymbol{a}_{ML,L} = \arg\min_{\boldsymbol{a}} L(\boldsymbol{\theta}, \lambda_L). \tag{3.43}$$

The challenge is that in addition to $\boldsymbol{a}_{ML,L}$, the optimum λ_L has to be determined also. This optimization can be carried out by solving the system of equations resulting from the Karush-Kuhn-Tucker (KKT) conditions. More about the KKT conditions can be found in literature, as in [4], and we will not go into details about them in this thesis.

The KKT conditions are

$$\nabla_{\boldsymbol{a}} L(\boldsymbol{a}_{ML,L}, \boldsymbol{\tau}, \lambda_L) = 0, \tag{3.44a}$$
$$h(\boldsymbol{a}_{ML,L}) \geq 0, \tag{3.44b}$$
$$\lambda_L \geq 0, \tag{3.44c}$$
$$\lambda_L h(\boldsymbol{a}_{ML,L}) = 0. \tag{3.44d}$$

Subequations (3.44b), (3.44c) and (3.44d) imply that

$$\text{when } \lambda_L = 0,\ h(\boldsymbol{a}_{ML,L}) > 0, \tag{3.45a}$$
$$\text{and when } \lambda_L > 0,\ h(\boldsymbol{a}_{ML,L}) = 0. \tag{3.45b}$$

A way to solve the constraint optimization is to consider first the case (3.45a), in which

$$\lambda_L = 0.$$

$L(\boldsymbol{\theta}, \lambda_L)$ becomes equal with $J_1(\boldsymbol{\theta})$, and by solving (3.44a)

$$\nabla_{\boldsymbol{a}} L(\boldsymbol{a}_{ML,L}, \boldsymbol{\tau}, \lambda_L) = \nabla_{\boldsymbol{a}} J_1(\boldsymbol{a}_{ML,L}, \boldsymbol{\tau}) = 0$$

results the same solution as in the unconstrained case in (3.23),

$$\boldsymbol{a}_{ML,L} = \boldsymbol{a}_{ML} = \left(\boldsymbol{C}(\boldsymbol{\tau})^H \boldsymbol{C}(\boldsymbol{\tau})\right)^{-1} \boldsymbol{C}(\boldsymbol{\tau})^H \boldsymbol{s}.$$

For this solution, the constraint needs to hold

$$h(\boldsymbol{a}_{ML,L}) > 0. \tag{3.46}$$

If (3.46) holds, then (3.23) is the expression of $\boldsymbol{a}_{ML,L}$.

This case is exactly equal to the first method, when the constraint is fulfilled. Hence, we can conclude that for the vectors of delays for which the constraint is inactive and fulfilled, then both methods yield the same result.

If (3.46) does no hold, the next case, (3.45b), is considered, with

$$\lambda_L > 0.$$

Expression (3.44a) becomes

$$\nabla_{\boldsymbol{a}} L(\boldsymbol{a}_{ML,L}, \boldsymbol{\tau}, \lambda_L) = \nabla_{\boldsymbol{a}} J_1(\boldsymbol{a}_{ML,L}, \boldsymbol{\tau}) - \lambda_L \nabla_{\boldsymbol{a}} h(\boldsymbol{a}_{ML,L}) = \mathbf{0}. \tag{3.47}$$

By solving

$$\nabla_{\boldsymbol{a}} L(\boldsymbol{a}_{ML,L}, \boldsymbol{\tau}, \lambda_L) = \begin{bmatrix} \frac{\partial L}{\partial \boldsymbol{a}} & \frac{\partial L}{\partial \boldsymbol{a}^*} \end{bmatrix} = \mathbf{0},$$

two sets of equations are obtained, both resulting in the same expression, as shown in appendix C,

$$\left(\boldsymbol{C}(\boldsymbol{\tau})^H \boldsymbol{C}(\boldsymbol{\tau}) - \lambda_L \boldsymbol{\Omega}\right) \boldsymbol{a}_{ML,L} = \boldsymbol{C}(\boldsymbol{\tau})^H \boldsymbol{s}. \tag{3.48}$$

If $\left(\boldsymbol{C}(\boldsymbol{\tau})^H \boldsymbol{C}(\boldsymbol{\tau}) - \lambda_L \boldsymbol{\Omega}\right)$ is invertible, the expression for $\boldsymbol{a}_{ML,L}$ is

$$\boldsymbol{a}_{ML,L} = \left(\boldsymbol{C}(\boldsymbol{\tau})^H \boldsymbol{C}(\boldsymbol{\tau}) - \lambda_L \boldsymbol{\Omega}\right)^{(-1)} \boldsymbol{C}(\boldsymbol{\tau})^H \boldsymbol{s}. \tag{3.49}$$

In order to find λ_L, the new expression for $\boldsymbol{a}_{ML,L}$ is used in

$$h(\boldsymbol{a}_{ML,L}) = 0, \tag{3.50}$$

resulting in the below equation, as shown in appendix C,

$$\left(\boldsymbol{C}(\boldsymbol{\tau})^H \boldsymbol{s}\right)^H \left(\boldsymbol{C}(\boldsymbol{\tau})^H \boldsymbol{C}(\boldsymbol{\tau}) - \lambda_L \boldsymbol{\Omega}\right)^{(-1)} \boldsymbol{\Omega} \left(\boldsymbol{C}(\boldsymbol{\tau})^H \boldsymbol{C}(\boldsymbol{\tau}) - \lambda_L \boldsymbol{\Omega}\right)^{(-1)} \boldsymbol{C}(\boldsymbol{\tau})^H \boldsymbol{s} = 0, \tag{3.51}$$

The expression for λ_L can be deduced from equation 3.51. The values obtained should be checked to hold the constraints for $\lambda_L > 0$ and for $\left(\boldsymbol{C}(\boldsymbol{\tau})^H \boldsymbol{C}(\boldsymbol{\tau}) - \lambda_L \boldsymbol{\Omega}\right)$ invertible.

Solving the equation (3.51) for any number of multipath components, so for arbitrary size of $\boldsymbol{C}(\boldsymbol{\tau})^H \boldsymbol{C}(\boldsymbol{\tau})$, is challenging, due to the matrix inversion. We restrict our analysis to the case of just one multipath component.

The following notations are used, to simplify the readability of the expression.

$$\boldsymbol{R}_{cc}(\boldsymbol{\tau}) = \boldsymbol{C}(\boldsymbol{\tau})^H \boldsymbol{C}(\boldsymbol{\tau}), \tag{3.52}$$

where $\boldsymbol{R}_{cc}$ can be seen as an auto-correlation matrix, of dimensions $(M+1, M+1)$.

$$\boldsymbol{r}_{sc}(\boldsymbol{\tau}) = \boldsymbol{C}(\boldsymbol{\tau})^H \boldsymbol{s}, \tag{3.53}$$

where $\boldsymbol{r}_{sc}$ can be seen as a cross-correlation vector, of dimensions $(M+1, 1)$.

For the case of $M = 1$, as shown in appendix C, the expression for λ_L is

$$\lambda_L = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}, \text{ subject to } \lambda_L > 0 \text{ and } \lambda_L \neq \frac{d}{e}, \tag{3.54}$$

where

$$\begin{aligned} a &= \alpha^4 \boldsymbol{r}_{sc}(\boldsymbol{\tau})^H \boldsymbol{\Omega}^{(-1)} \boldsymbol{r}_{sc}(\boldsymbol{\tau}), \\ b &= 2\alpha^2 \det(R_{cc}) \boldsymbol{r}_{sc}(\boldsymbol{\tau})^H \boldsymbol{R}_{cc}(\boldsymbol{\tau})^{(-1)} \boldsymbol{r}_{sc}(\boldsymbol{\tau}), \\ c &= \det(R_{cc})^2 \boldsymbol{r}_{sc}(\boldsymbol{\tau})^H \boldsymbol{R}_{cc}(\boldsymbol{\tau})^{(-1)} \boldsymbol{\Omega} \boldsymbol{R}_{cc}(\boldsymbol{\tau})^{(-1)} \boldsymbol{r}_{sc}(\boldsymbol{\tau}), \\ d &= \det(R_{cc}), \\ e &= \alpha^2 R_{cc}(\boldsymbol{\tau})_{2,2} - R_{cc}(\boldsymbol{\tau})_{1,1}, \\ \det(R_{cc}) &= R_{cc}(\boldsymbol{\tau})_{1,1} R_{cc}(\boldsymbol{\tau})_{2,2} - R_{cc}(\boldsymbol{\tau})_{1,2} R_{cc}(\boldsymbol{\tau})_{2,1}. \end{aligned} \tag{3.55}$$

The constrained expression for $\boldsymbol{a}_{ML,L}$ is used in the cost function, giving

$$\begin{aligned} J_1(\boldsymbol{\theta})|_{\boldsymbol{a}=\boldsymbol{a}_{ML,L}} &= -2\boldsymbol{s}^H\boldsymbol{C}(\boldsymbol{\tau})\big(\boldsymbol{C}(\boldsymbol{\tau})^H\boldsymbol{C}(\boldsymbol{\tau})-\lambda_L\boldsymbol{\Omega}\big)^{(-1)}\boldsymbol{C}(\boldsymbol{\tau})^H\boldsymbol{s} \\ &+\boldsymbol{s}^H\boldsymbol{C}(\boldsymbol{\tau})\big(\boldsymbol{C}(\boldsymbol{\tau})^H\boldsymbol{C}(\boldsymbol{\tau})-\lambda_L\boldsymbol{\Omega}\big)^{(-1)}\boldsymbol{C}(\boldsymbol{\tau})^H\boldsymbol{C}(\boldsymbol{\tau})\big(\boldsymbol{C}(\boldsymbol{\tau})^H\boldsymbol{C}(\boldsymbol{\tau})-\lambda_L\boldsymbol{\Omega}\big)^{(-1)}\boldsymbol{C}(\boldsymbol{\tau})^H\boldsymbol{s}, \end{aligned} \tag{3.56a}$$

which is

$$J_1(\boldsymbol{\theta})|_{\boldsymbol{a}=\boldsymbol{a}_{ML,L}} = -2\boldsymbol{s}^H\boldsymbol{C}(\boldsymbol{\tau})\boldsymbol{a}_{ML,L}+\boldsymbol{a}_{ML,L}^H\boldsymbol{C}(\boldsymbol{\tau})^H\boldsymbol{C}(\boldsymbol{\tau})\boldsymbol{a}_{ML,L}. \tag{3.56b}$$

To illustrate the Lagrange optimization, the pseudo-code is presented next.
Pseudo-code:

```
1 : Create a set $\mathcal{T}$ of all possible combinations of delays, $\boldsymbol{\tau}_i$, which are in the interval $[-T_c, T_c]$ and respect the condition $\tau_0 < \tau_1 < ... < \tau_M$. The index $i$ indicates the position in the set.
# For each $\boldsymbol{\tau}_i$
2 : Compute $\boldsymbol{a}_{ML}(\boldsymbol{\tau}_i) = \big(\boldsymbol{C}(\boldsymbol{\tau}_i)^H\boldsymbol{C}(\boldsymbol{\tau}_i)\big)^{-1}\boldsymbol{C}(\boldsymbol{\tau}_i)^H\boldsymbol{s}$
3 : Check if $\boldsymbol{a}_{ML}(\boldsymbol{\tau}_i)^H\boldsymbol{\Omega}\boldsymbol{a}_{ML}(\boldsymbol{\tau}_i) \geq 0$
# If yes
4 : Compute $J_1(\boldsymbol{\tau}_i) = -\boldsymbol{s}^H\boldsymbol{C}(\boldsymbol{\tau}_i)\boldsymbol{a}_{ML}(\boldsymbol{\tau}_i)$
# If not
5 : Compute $\lambda_L(\boldsymbol{\tau}_i)$ given by (3.54)
6 : Compute $\boldsymbol{a}_{ML,L}(\boldsymbol{\tau}_i) = \big(\boldsymbol{C}(\boldsymbol{\tau}_i)^H\boldsymbol{C}(\boldsymbol{\tau}_i)-\lambda_L(\boldsymbol{\tau}_i)\boldsymbol{\Omega}\big)^{(-1)}\boldsymbol{C}(\boldsymbol{\tau}_i)^H\boldsymbol{s}$
7 : Compute $J_1(\boldsymbol{\tau}_i) = -2\boldsymbol{s}^H\boldsymbol{C}(\boldsymbol{\tau}_i)\boldsymbol{a}_{ML,L}(\boldsymbol{\tau}_i)+\boldsymbol{a}_{ML,L}(\boldsymbol{\tau}_i)^H\boldsymbol{C}(\boldsymbol{\tau}_i)^H\boldsymbol{C}(\boldsymbol{\tau}_i)\boldsymbol{a}_{ML,L}(\boldsymbol{\tau}_i)$
# Minimization
8 : Find the index $k$ yielding the lowest value $J_1(\boldsymbol{\tau}_i)$
9 : $\hat{\boldsymbol{\tau}}_{ML} = \boldsymbol{\tau}_k$
```

For those delay vectors in which the unconstrained amplitudes already fulfil the constraint, both methods lead to the same estimate of the amplitudes. The difference lies in those delay vectors for which the constraint is active. While in the first method we simply discard those delays by setting the cost function to zero, in the constrained optimization method we try to find the best vector of amplitudes that still fulfils the constraints and in this case, the constraint will be fulfilled with equality.

In the end, the methods would give different solutions only if the optimum of the cost function found by the constrained optimization method corresponds to a vector of delays in which the amplitude constraint will be active. Otherwise, both methods will give the same solution, i.e. vector of delays.

3.5 Computational complexity

A drawback of the ML estimator is the computational complexity. In order to asses the computational complexity, the ML estimator without constrained amplitudes will be considered. We considered that the computational complexity of the other two approached with constrained amplitudes is not significantly different.

If the algorithm is not optimized to reduce the computational burden, the cost function in 3.27a is calculated at each observation interval, for each $\boldsymbol{\tau}_i \in \mathcal{T}$. For visualisation, the dimensions of each component in the cost function will be stated.

$$J_1(\boldsymbol{\tau}_i)|_{\boldsymbol{a}=\boldsymbol{a}_{ML}} = -\boldsymbol{s}_{1,N}^H \boldsymbol{C}(\boldsymbol{\tau}_i)_{N,M+1} \left(\boldsymbol{C}(\boldsymbol{\tau}_i)_{M+1,N}^H \boldsymbol{C}(\boldsymbol{\tau}_i)_{N,M+1}\right)^{-1} \boldsymbol{C}(\boldsymbol{\tau}_i)_{M+1,N}^H \boldsymbol{s}_{N,1} \quad (3.57)$$

To increase the computational efficiency, the product $\boldsymbol{C}(\boldsymbol{\tau}_{all})^H \boldsymbol{C}(\boldsymbol{\tau}_{all})$ can be precomputed for all the possible values of τ and stored in the receiver. Here, $\boldsymbol{\tau}_{all}$ denotes the vector of all the values of τ on the search grid. During the estimation, for any $\boldsymbol{C}(\boldsymbol{\tau}_i)^H \boldsymbol{C}(\boldsymbol{\tau}_i)$, the corresponding values can be retrieved from the stored matrix. By doing so, only the (M+1, M+1) matrix inversion is needed at each step in the search.

For each observation interval, the signal can be multiplied with the matrix of all the delayed PRN replicas, and during the estimation, for any $\boldsymbol{\tau}_i$, select the necessary values from the precomputed vector $\boldsymbol{C}(\boldsymbol{\tau}_{all})^H \boldsymbol{s}$.

We use N_{grid} to refer to the dimension of the vector $\boldsymbol{\tau}_{all}$. The value of N_{grid} depends on the implementation and on the resolution of the grid.

For the approach presented, the computational complexity is driven by the product $\boldsymbol{C}(\boldsymbol{\tau}_{all})^H \boldsymbol{s}$ and is equal with $\mathcal{O}(N_{grid}N)$.

In the receiver, the computational complexity might differ, depending on the specific implementation. In order for the ML estimator to be suitable for implementation in the GNSS receiver, the computational burden needs to be reduced. In chapter 4, the compression of the received signal will be presented.

Chapter 4

Compression

In chapter 3, the ML estimation method was presented for estimating the delays in the GNSS signals. Even if the method is optimal and achieves good results, it is not suitable for GNSS applications if the computations are too costly. A standard observation interval in GNSS is between 1 ms and 20 ms, and the sampling rates are high, leading to more than 10^5 samples for one observation. While the correlations between the PRN codes with different delays can be precomputed in advance and stored in the receiver, the correlations between the received signal and the delayed PRN codes have to be computed at each observation interval, and maybe for each possible delay, which may be infeasible to do in real-time computations. In this chapter, two different compression methods will be presented. The chapter is using as main reference the work of Selva Vera in [11].

First, it will be presented the idea of compression, what's expected from it and what are the evaluation criteria, and after that, the compression matrix will be derived.

4.1 Compression in general

The compression should achieve the reduction of the number of components of the observed vector $\boldsymbol{s}$, yet keeping all the information needed for the estimation process. It is the projection of the received signal into a subspace of dimension much smaller than the original signal vector. The parameter of interest in the ML estimation is the delay of the line of sight. It follows that the information needed for the ML estimation is the part from the signal that carries information about the delays of the LOS and multipath components. Considering the signal model in (3.5), restated here for convenience,

$$\boldsymbol{s} = \boldsymbol{C}(\boldsymbol{\tau})\boldsymbol{a} + \boldsymbol{w},$$

the component that needs to be preserved is the matrix $\boldsymbol{C}(\boldsymbol{\tau})$. Reformulating, all the information needed is contained in the subspace spanned by the columns of $\boldsymbol{C}(\boldsymbol{\tau})$. So, it can be said that the compression should project $\boldsymbol{s}$ into a smaller subspace, that contains $\boldsymbol{c}(\tau)$, for any τ.

Applying the compression matrix Q to the received signal, the following is obtained

$$\boldsymbol{Q}^H\boldsymbol{s} = \boldsymbol{Q}^H\big(\boldsymbol{C}(\boldsymbol{\tau})\boldsymbol{a} + \boldsymbol{w}\big), \tag{4.1}$$

The compressed signal model is

$$\boldsymbol{s_c} = \boldsymbol{C_c}(\boldsymbol{\tau})\boldsymbol{a} + \boldsymbol{w_c}, \tag{4.2}$$

where,

$$\boldsymbol{s_c} = \boldsymbol{Q}^H\boldsymbol{s}, \tag{4.3}$$

$$\boldsymbol{C_c}(\boldsymbol{\tau}) = \boldsymbol{Q}^H\boldsymbol{C}(\boldsymbol{\tau}) \tag{4.4}$$

$$\boldsymbol{w_c} = \boldsymbol{Q}^H\boldsymbol{w}. \tag{4.5}$$

The new cost function after compression will be

$$J_c = || \boldsymbol{s_c} - \boldsymbol{C_c}(\boldsymbol{\tau})\boldsymbol{a} \ ||^2, \tag{4.6}$$

which can be extended and approximate to

$$J_{1c} = -\boldsymbol{s_c}^H\boldsymbol{C_c}(\boldsymbol{\tau})\boldsymbol{a} - \boldsymbol{a}^H\boldsymbol{C_c}(\boldsymbol{\tau})^H\boldsymbol{s_c} + \boldsymbol{a}^H\boldsymbol{C_c}(\boldsymbol{\tau})^H\boldsymbol{C_c}(\boldsymbol{\tau})\boldsymbol{a}. \tag{4.7}$$

The unconstrained expression for the amplitudes is

$$\boldsymbol{a} = (\boldsymbol{C_c}(\boldsymbol{\tau})^H\boldsymbol{C_c}(\boldsymbol{\tau}))^{-1}\boldsymbol{C_c}(\boldsymbol{\tau})^H\boldsymbol{s_c}. \tag{4.8}$$

To achieve an efficient compression, the matrix $\boldsymbol{Q}$ should have a low number of columns, much lower then the number of rows.

The space of $\boldsymbol{Q}$ should contain the space of $\boldsymbol{c}(\tau)$ for any τ,

$$\boldsymbol{Q}\boldsymbol{Q}^H\boldsymbol{c}(\tau) = \boldsymbol{c}(\tau). \tag{4.9}$$

In order to keep the the noise samples of $\boldsymbol{s}$ uncorrelated after compression, the following condition needs to be fulfilled

$$\boldsymbol{Q}^H\boldsymbol{Q} = \boldsymbol{I}. \tag{4.10}$$

The relation in equation (4.9) is for the ideal case, but in practice some information is lost after compression. The loss incurred in compression for a given τ can quantified by [11]

$$H_{loss}(\tau) = \frac{|| \ \boldsymbol{c}(\tau) - \boldsymbol{Q}\boldsymbol{Q}^H\boldsymbol{c}(\tau) \ ||^2}{|| \ \boldsymbol{c}(\tau) \ ||^2}. \tag{4.11}$$

In order to find the optimum compression matrix, some criteria have to be considered: the number of columns in $\boldsymbol{Q}$ to be as small as possible, the hardware implementation to be easy to achieve and loss incurred in compression to be as low as possible, for all possible delays.

The canonical components compression method will be presented, with two different approaches, based on correlators matched to the code, and to the signal. This method has the advantage that it can be implemented in the receiver by modifying the DLL discriminator.

4.2 Compression based on code matched correlators

It will be shown that the observed signal can be compressed using a bank of code matched correlators. This is based on the PRN code structure.

As it was stated in section 2.3, the PRN code can be seen as the convolution between a sequence of Dirac deltas, with values of -1 and $+1$, called chips, and a pulse. Because the rectangular pulse shape is the most used in the GNSS receivers, only this one will be considered in this thesis. The distance between two chips is the chip period T_c, and most of the energy of the rectangular pulse shape is inside of an interval equal to the chip period.

$$c(t) = p(t) * g(t), \tag{4.12}$$

where $p(t)$ is the sequence of Dirac deltas and $g(t)$ is the pulse shape.

The components in equation (4.12) are represented in figure 4.1, for arbitrary values of the first four chips.

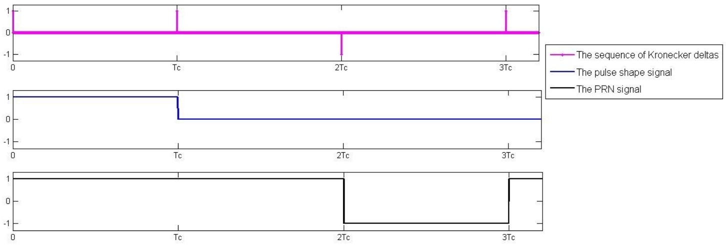


Figure 4.1: The signals contributing to the PRN signal.

The difference between the PRN code and the PRN chipping sequence is that the PRN chipping sequence is sparse, having only the first sample from each chip equal with $+1$ or -1, and the rest of the samples being 0.

In the digital domain, the convolution can be expressed as

$$c((n-1)T_s) = \sum_{k=-\infty}^{\infty} p((n-1-k)T_s)g(kT_s). \tag{4.13}$$

Considering that most of the energy of $g(t)$ is inside $[0, T_c]$, the expression can be approximated to

$$c((n-1)T_s) \approx \sum_{k=0}^{K-1} p((n-1-k)T_s)g(kT_s), \tag{4.14}$$

where K is the number of samples in one chip period.

By taking N samples and arranging them in a vector, the following expression can be stated

$$\begin{bmatrix} c(0T_s) \\ c(1T_s) \\ \vdots \\ c((N-1)T_s) \end{bmatrix} \approx \begin{bmatrix} p(0T_s - 0T_s) & p(0T_s - 1T_s) & \dots & p(0T_s - (K-1)T_s) \\ p(1T_s - 0T_s) & p(1T_s - 1T_s) & \dots & p(1T_s - (K-1)T_s \\ \vdots & \vdots & \ddots & \vdots \\ p((N-1)T_s - 0T_s) & p((N-1)T_s - 1T_s) & \dots & p((N-1)T_s - (K-1)T_s \end{bmatrix} \begin{bmatrix} g(0T_s) \\ g(1T_s) \\ \vdots \\ g((K-1)T_s) \end{bmatrix} \tag{4.15}$$

and more compact, the PRN code $\boldsymbol{c}$ is

$$\boldsymbol{c} \approx \boldsymbol{P}\boldsymbol{g}, \tag{4.16}$$

where

$$\begin{aligned}
&\boldsymbol{c} = [c_1\ c_2\ ...\ c_N]^T, \text{ with } c_n \equiv c((n-1)T_s), \\
&\boldsymbol{P} = [\boldsymbol{p}(0)\ \boldsymbol{p}(T_s)\ ...\ \boldsymbol{p}((K-1)T_s)] \text{ with} \\
&\boldsymbol{p}(kT_s) = [p_1(kT_s)\ p_2(kT_s)\ ...\ p_N(kT_s)]^T \text{ and } p_n(kT_s) \equiv p((n-1)T_s - kT_s), \\
&\boldsymbol{g} = [g_1\ g_2\ ...\ g_K]^T \text{ with } g_k \equiv g((k-1)T_s).
\end{aligned} \tag{4.17}$$

The delayed PRN code is $c(t-\tau)$, where τ is a random delay. Considering the delay τ a multiple of T_s, equal to iT_s, the discrete delayed PRN signal is

$$\begin{aligned}
c((n-1)T_s - \tau) &= c((n-i-1)T_s) \\
&\approx \sum_{k=0}^{K-1} p((n-i-1-k)T_s)g(kT_s) \\
&\approx \sum_{k=i}^{K-1+i} p((n-i-1-(k-i))T_s)g((k-i)T_s) \\
&\approx \sum_{k=i}^{K+i-1} p((n-1-k)T_s)g(kT_s - \tau)
\end{aligned} \tag{4.18}$$

The equation (4.18) we can conclude that for delaying the PRN code is enough to delay to pulse in equation (4.12), as below

$$c(t-\tau) = p(t) * g(t-\tau). \tag{4.19}$$

If τ can take values between $[-Tc, Tc]$, then i is between $[-K, K]$, and the discrete delayed PRN code, $c((n-1)T_s - \tau)$, for any possible τ, can be expressed as

$$c((n-1)T_s - \tau) \approx \sum_{k=-K}^{2K-1} p((n-1)T_s - kT_s)g(kT_s - \tau). \tag{4.20}$$

For N samples ordered in a vector, the expression becomes

$$\begin{bmatrix} c(0T_s-\tau) \\ c(1T_s-\tau) \\ \vdots \\ c((N-1)T_s-\tau) \end{bmatrix} \approx$$
$$\begin{bmatrix} p(0T_s+KT_s) & \dots & p(0T_s-0T_s) & \dots & p(0T_s-(2K-1)T_s) \\ p(1T_s+KT_s) & \dots & p(1T_s-0T_s) & \dots & p(1T_s-(2K-1)T_s \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ p((N-1)T_s+KT_s) & \dots & p((N-1)T_s-0T_s) & \dots & p((N-1)T_s-(2K-1)T_s \end{bmatrix} \begin{bmatrix} g(0T_s-\tau) \\ g(1T_s-\tau) \\ \vdots \\ g((3K-1)T_s-\tau) \end{bmatrix} \tag{4.21}$$

and in a more compact notation, the delayed PRN vector $\boldsymbol{c}(\tau)$ can be expressed as

$$\boldsymbol{c}(\tau) \approx \overline{\boldsymbol{P}}\boldsymbol{g}(\tau), \tag{4.22}$$

where $\boldsymbol{c}(\tau)$ is the same as in expression (3.6), and

$$\begin{aligned} &\overline{\boldsymbol{P}} = [\boldsymbol{p}(-KT_s) \; ...\boldsymbol{p}(0) \; ... \; \boldsymbol{p}((2K-1)T_s)] \text{ with } p_n(kT_s) \text{ the same as in equation (4.17)} \\ &\boldsymbol{g}(\tau) = [g_1(\tau) \; g_2(\tau) \; ... \; g_{3K}(\tau)]^T \text{ with } g_k \equiv g((k-1)T_s - \tau). \end{aligned} \tag{4.23}$$

The matrix $\overline{\boldsymbol{P}}$ is nothing more then a set of delayed replicas of the sampled PRN sequence of deltas, $p((n-1)T_s)$. The expression in (4.22) is very important because, according to it, it can be stated that $\boldsymbol{c}(\tau)$ is inside the span of $\overline{\boldsymbol{P}}$, for any τ [11].

According to the condition in equation (4.9), the space of $\boldsymbol{Q}$ should contain the space of $\boldsymbol{c}(\tau)$ for any τ. The conclusion is that $\boldsymbol{Q}$ can be chosen to have the same span as $\overline{\boldsymbol{P}}$.

We could choose $\boldsymbol{Q} = \overline{\boldsymbol{P}}$, but the condition

$$\boldsymbol{Q}^H\boldsymbol{Q} = \boldsymbol{I},$$

needs to be fulfilled. According to this condition, the columns of $\boldsymbol{Q}$ have to be orthonormal, meaning orthogonal and of norm 1. If any two different columns are multiplied the result should be 0.

Considering the sparse nature of the columns of $\overline{\boldsymbol{P}}$, and the pseudo-random characteristic of the PRN code, we can state that only the multiplication between columns spaced with a multiple of K will give a result different than zero. If one column is multiplied with itself, will give the result N/K and if the column from the position 1 will be multiplied with the column from position $K+1$ or $2K+1$, the result will close to zero, due to pseudo random PRN code. So the conclusion is that the matrix $\overline{\boldsymbol{P}}$ almost fulfils the conditions to be the compression matrix, but not perfectly. So we chose to include an additional matrix $\boldsymbol{R}$, which will be multiplied with $\overline{\boldsymbol{P}}$, and the result to be a matrix with orthonormal columns. The compression matrix can be expressed as [11]

$$\boldsymbol{Q} = \overline{\boldsymbol{P}}\boldsymbol{R}, \tag{4.24}$$

where $\boldsymbol{R}$ is obtained from the Cholesky decomposition of $(\overline{\boldsymbol{P}}^H\overline{\boldsymbol{P}})^{-1}$ [11], fulfilling the condition that

$$\boldsymbol{R}\boldsymbol{R}^H = (\overline{\boldsymbol{P}}^H\overline{\boldsymbol{P}})^{-1}, \tag{4.25}$$

With the compression matrix $\boldsymbol{Q}$ obtained as in (4.24), we make sure the conditions in (4.10) and (4.9) are fulfilled.

The compressed signal is

$$\boldsymbol{s}_c = (\overline{\boldsymbol{P}}\boldsymbol{R})^H\boldsymbol{s} = \boldsymbol{R}^H\overline{\boldsymbol{P}}^H\boldsymbol{C}(\boldsymbol{\tau})\boldsymbol{a} + \boldsymbol{R}^H\overline{\boldsymbol{P}}^H\boldsymbol{w} = \boldsymbol{C}_c(\boldsymbol{\tau})\boldsymbol{a} + \boldsymbol{w}_c, \tag{4.26}$$

The cost function in equation (4.6) becomes

$$J_c = || \; \boldsymbol{R}^H(\overline{\boldsymbol{P}}^H\boldsymbol{s}) - \boldsymbol{R}^H(\overline{\boldsymbol{P}}^H\boldsymbol{C}(\boldsymbol{\tau}))\boldsymbol{a} \; ||^2 \tag{4.27}$$

The term $\overline{\boldsymbol{P}}^H\boldsymbol{s}$ can be obtained after correlating the observed signal $\boldsymbol{s}$ with a bank of delayed replicas of $\boldsymbol{p}$. The number of correlators needed is equal to the number of columns in $\overline{\boldsymbol{P}}$,and it can be reduced, for higher compression levels. The rest of the components can be precomputed and stored in the receiver. The code matched correlators refer to the correlators matched to the delayed replicas of $\boldsymbol{p}$.

4.3 Compression based on signal matched correlators

The compression based on signal matched correlators refers to correlators matched to the PRN signal, $c(t)$. It's actually much closer to the classical DLL structure than the code matched correlators. The considerations leading to this kind of compression are similar to those presented in section 4.2, but simplified.

Inspired by the proof in section 4.2, we force the formulation and say that a discrete sample of the PRN code is equal with

$$c((n-1)T_s) = \sum_{k=0}^{K-1} c((n-1-k)T_s) w_p(kT_s), \tag{4.28}$$

where $w_p(kT_s)$ is a sample in a vector $\boldsymbol{w_p}$, which has the component $w_p(0) = 1$, the rest the components equal to 0.

By taking N samples and arranging them in a vector, the following is obtained

$$\begin{bmatrix} c(0T_s) \\ c(1T_s) \\ \vdots \\ c((N-1)T_s) \end{bmatrix} \approx \begin{bmatrix} c(0T_s - 0T_s) & c(0T_s - 1T_s) & \dots & c(0T_s - (K-1)T_s) \\ c(1T_s - 0T_s) & c(1T_s - 1T_s) & \dots & c(1T_s - (K-1)T_s \\ \vdots & \vdots & \ddots & \vdots \\ c((N-1)T_s - 0T_s) & c((N-1)T_s - 1T_s) & \dots & c((N-1)T_s - (K-1)T_s \end{bmatrix} \begin{bmatrix} w_p(0T_s) \\ w_p(1T_s) \\ \vdots \\ w_p((K-1)T_s) \end{bmatrix} \tag{4.29}$$

and the PRN vector $\boldsymbol{c}$ becomes

$$\boldsymbol{c} \approx \boldsymbol{C_{cor}} \boldsymbol{w_p}, \tag{4.30}$$

where $\boldsymbol{c}$ is the same as in (4.17) and

$$\begin{aligned} &\boldsymbol{C_{cor}} = [\boldsymbol{c}(0)\ \boldsymbol{c}(T_s)\ ...\ \boldsymbol{c}((K-1)T_s)] \text{ with} \\ &\boldsymbol{c}(kT_s) = [c_1(kT_s)\ c_2(kT_s)\ ...\ c_N(kT_s)]^T \text{ and } c_n(kT_s) \equiv c((n-1)T_s - kT_s), \\ &\boldsymbol{w_p} = [(w_p)_1\ (w_p)_2\ ...\ (w_p)_K]^T \text{ with } (w_p)_k \equiv w_p((k-1)T_s). \end{aligned} \tag{4.31}$$

To delay the code c is enough to delay the window w_p, and in the discrete domain it can be stated that

$$c((n-1)T_s - \tau) \approx \sum_{k=-K}^{K-1} c((n-1)T_s - kT_s) w_p(kT_s - \tau). \tag{4.32}$$

By arranging N samples in a vector, the following results

$$\begin{bmatrix} c(0T_s - \tau) \\ c(1T_s - \tau) \\ \vdots \\ c((N-1)T_s - \tau) \end{bmatrix} \approx \\ \begin{bmatrix} c(0T_s + KT_s) & \dots & c(0T_s - 0T_s) & \dots & c(0T_s - (K-1)T_s) \\ c(1T_s + KT_s) & \dots & c(1T_s - 0T_s) & \dots & c(1T_s - (K-1)T_s \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ c((N-1)T_s + KT_s) & \dots & c((N-1)T_s - 0T_s) & \dots & c((N-1)T_s - (K-1)T_s \end{bmatrix} \begin{bmatrix} w_p(0T_s - \tau) \\ w_p(1T_s - \tau) \\ \vdots \\ w_p((2K-1)T_s - \tau) \end{bmatrix} \tag{4.33}$$

In a more compact notation,

$$\boldsymbol{c}(\tau) \approx \overline{\boldsymbol{C}}_{\boldsymbol{cor}} \boldsymbol{w}_{\boldsymbol{p}}(\tau), \tag{4.34}$$

where $\boldsymbol{c}(\tau)$ is the same as in (3.6), and

$$\begin{aligned} \overline{\boldsymbol{C}}_{\boldsymbol{cor}} &= [\boldsymbol{c}(-KT_s) \; ...\boldsymbol{c}(0) \; ... \; \boldsymbol{c}((K-1)T_s)] \text{ with } \boldsymbol{c}(kT_s) \text{ the same as in (4.31)} \\ \boldsymbol{w}_{\boldsymbol{p}}(\tau) &= [(w_p)_1(\tau) \; (w_p)_2(\tau) \; ... \; (w_p)_{2K}(\tau)]^T \text{ with } (w_p)_k \equiv w_p((k-1)T_s - \tau). \end{aligned} \tag{4.35}$$

It can be noticed from (3.6) and (4.35), that $\boldsymbol{c}(\tau)$ and $\boldsymbol{c}(kT_s)$ are the same when τ is equal with kT_s.

From expression (4.34) it can be concluded that $\boldsymbol{c}(\tau)$ is inside the span of $\overline{\boldsymbol{C}}_{\boldsymbol{cor}}$, for any τ [11]. Because $\boldsymbol{Q}$ is supposed to contain the space of $\boldsymbol{c}(\tau)$ for any τ, it can be deduced that $\boldsymbol{Q}$ and $\overline{\boldsymbol{C}}_{\boldsymbol{cor}}$ can have the same span. As in the previous method, taking into account that

$$\boldsymbol{Q}^H \boldsymbol{Q} = \boldsymbol{I},$$

there is a matrix $\boldsymbol{R}_{\boldsymbol{s}}$, for which [11]

$$\boldsymbol{Q} = \overline{\boldsymbol{C}}_{\boldsymbol{cor}} \boldsymbol{R}_{\boldsymbol{s}}. \tag{4.36}$$

The same as for code matched correlators, the matrix $\boldsymbol{R}_{\boldsymbol{s}}$ can be obtained from the Cholesky decomposition of $(\overline{\boldsymbol{C}}_{\boldsymbol{cor}}{}^H \overline{\boldsymbol{C}}_{\boldsymbol{cor}})^{-1}$, fulfilling the condition that

$$\boldsymbol{R}_{\boldsymbol{s}} \boldsymbol{R}_{\boldsymbol{s}}{}^H = (\overline{\boldsymbol{C}}_{\boldsymbol{cor}}{}^H \overline{\boldsymbol{C}}_{\boldsymbol{cor}})^{-1}, \tag{4.37}$$

The compressed signal is

$$\boldsymbol{s}_c = (\overline{\boldsymbol{C}}_{\boldsymbol{cor}} \boldsymbol{R}_s)^H \boldsymbol{s} = \boldsymbol{R}_s^H \overline{\boldsymbol{C}}_{\boldsymbol{cor}}{}^H \boldsymbol{C}(\boldsymbol{\tau}) \boldsymbol{a} + \boldsymbol{R}_s^H \overline{\boldsymbol{C}}_{\boldsymbol{cor}}{}^H \boldsymbol{w} = \boldsymbol{C}_c(\boldsymbol{\tau}) \boldsymbol{a} + \boldsymbol{w}_c, \tag{4.38}$$

The cost function in equation (4.6) becomes

$$J_c = || \; \boldsymbol{R}_s^H (\overline{\boldsymbol{C}}_{\boldsymbol{cor}}{}^H \boldsymbol{s}) - \boldsymbol{R}_s^H (\overline{\boldsymbol{C}}_{\boldsymbol{cor}}{}^H \boldsymbol{C}(\boldsymbol{\tau})) \boldsymbol{a} \; ||^2 \tag{4.39}$$

Similar to the previous method, the term $\overline{\boldsymbol{C}}_{\boldsymbol{cor}}{}^H \boldsymbol{s}$ can be obtained after correlating the observed signal $\boldsymbol{s}$ with a bank of delayed replicas of the PRN vector $\boldsymbol{c}$. The initial number of correlators can be reduced further, and the degree of compression is subject to further research. The rest of the terms required by the cost function can be precomputed and stored in the receiver.

The columns of $\overline{\boldsymbol{C}}_{\boldsymbol{cor}}$ and $\overline{\boldsymbol{P}}$ are called correlators. The maximum number of correlators in the case of signal matched correlators is equal with the number of samples spanning the interval where the delays could be, so the interval of two chip periods $[-T_c, Tc]$. The maximum number of correlators in the case of code matched correlators is a bit higher, equal with the number of samples spanning the interval where the delays could be plus the number of samples in one chip period, so three chip periods in total, from $[-T_c, 2Tc]$.

The role of the matrix $\boldsymbol{R}$ and $\boldsymbol{R}_s$ is to make sure that after compression, the noise components from the observed signal are kept uncorrelated. In the absence of noise, the role of this matrix in the compression should be negligible and to obtain good estimation results without $\boldsymbol{R}$ or $\boldsymbol{R}_s$ included in the compression. This will be investigated in chapter 5.

Chapter 5

Numerical Assessment

In this chapter, the numerical performance of the ML estimation will be evaluated. In section 5.2 a simple scenario will be considered, in which the delays are matching perfectly the search space and where the ML estimation is expected to give optimum results. In section 5.3, a more complicated scenario is considered, in which the delays can take any value, and errors, due to resolution and to numerical issues, are expected in the estimation. For this scenario, which we call from now on the *continuous delays scenario*, and fits better to what is expected in real-life situations, the performance of the ML estimation will be compared with the performance of one of the classical DLL's, the narrow correlator, with 0.1 and 0.3 early-late spacing.

Furthermore, we will explore in this chapter the effects that different types of compression will have on the performance of the ML estimator, compared to the case in which no compression is applied. For the DLL, the uncompressed signal is considered.

The compression is implemented with full number of correlators, 41 for Signal Matched Correlators (SMC) and 61 for Code Matched Correlators (CMC).

5.1 Assumptions about the system

In the thesis, when referring to signal to noise ration (SNR), we are sometimes referring to the carrier to noise density ratio (C/N_0), which is the ratio of the received carrier power to noise density and is measured in dB Hz. The difference can be seen from the units in which they are expressed. It is convenient to refer to C/N_0, because it is fixed for all the methods investigated, while the SNR is changed by the method used.

When multipath components are part of the signal ($M > 0$), the total SNR will change. As a convention in the thesis, the term SNR will be used for the SNR of the direct component. When this will not be the case, it will be clear from the text.

5.2 Discrete delays

In this section, an ideal case is considered, in which the delays can only be multiples of the sample period, and the search space for the delays is the grid spaced with T_s, in the interval $[-T_c, T_c]$.

To verify the correctness of the implementation and get obtain some insight on the estimator's behavior, we first assess the estimator's accuracy in a noise-free environment, where the estimator is expected to provide the correct result. After that, we evaluate it in a more realistic environment with AWGN by means of Monte Carlo simulations. For all simulations with noise, 1000 Monte Carlo trials will be run.

The compression methods will be compared and the effect of the whitening matrix on different compression methods will be investigated.

The performance of the estimator will be assessed also when the assumptions made on the signal model are not correct. We will restrict our investigation to only two situations, when one multipath component is present in the received signal, but not assumed by the signal model used for estimation, and the opposite, when the received signal has just the direct component, but the estimator assumes there is an additional multipath component.

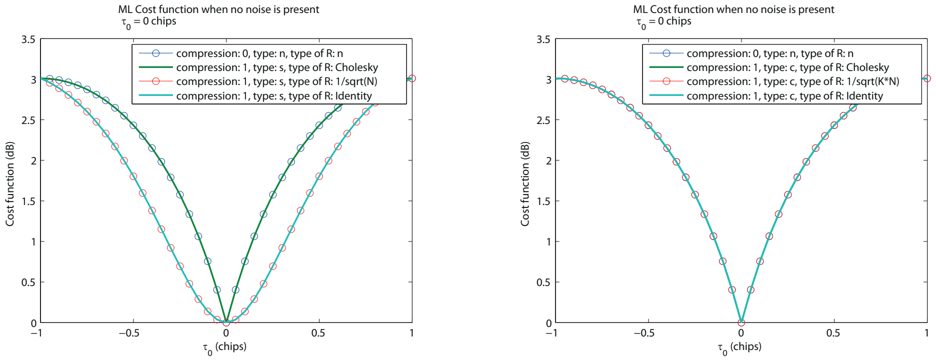
5.2.1 No multipath component

The scenario, in which the received signal has just the line of sight (LOS) component, and the signal model is assumed correctly, is the one that gives the best estimation results. In an ideal situation, with delays on the grid of the search, the estimation is perfect. Therefore, we use this scenario to explore how compression of the received signal may affect the estimator's accuracy. In fact, if the estimator cannot obtain accurate estimates with a given compression for this scenario, there is little hope that it will work well with more complicated signals.

In both compression methods discussed, SMC and CMC, the compression is achieved in two steps. First, the received signal is passed through a bank of correlators, and second, a whitening matrix $\boldsymbol{R}$ is applied. The role of the matrix $\boldsymbol{R}$ is to make sure that after compression, the noise components from the observed signal remain uncorrelated. The impact of $\boldsymbol{R}$ on the two compression methods is different, and it is expected to be negligible for the CMC, as the correlators of this method almost don't correlate the noise. We tested the performance of the algorithm without the matrix $\boldsymbol{R}$ obtained from a Cholesky decomposition, by replacing $\boldsymbol{R}$ with the identity matrix $\boldsymbol{I}$. An intermediate method was also tested, in which a weighted identity matrix was considered, $\kappa\boldsymbol{I}$, with κ given by $1/\sqrt{N}$ for SMC and $1/(\sqrt{KN})$ for CMC, where N is the number of samples in the observation interval and K is the ratio between the chip period T_c and the sample period T_s.

Noiseless channel

In order to test if the shape of the ML cost function in 3.20 is changed after compression, and what is the effect of the whitening matrix on it, the results in figure 5.1 were produced. The cost function for the uncompressed signal is compared with the cost function for the compressed signal. The expectation is, that in the absence of noise, the matrix $\boldsymbol{R}$ to have no effect on the estimation accuracy, i.e. where the minimum of the cost function is, but to change the shape of the cost function for some methods.



(a) No compression and compression with SMC.

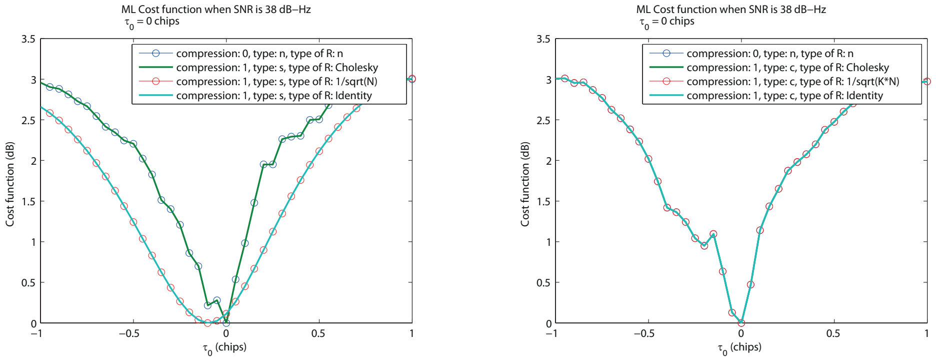
(b) No compression and compression with CMC.

Figure 5.1: Comparison between the cost function for compressed and uncompressed signals, and for different compression types. No multipath components are present and no noise. The cost functions are normalized.

It can be observed that for the signal compressed with CMC, the cost function has the same shape as the cost function for uncompressed signal, no mater how the whitening matrix is obtain. For the signal compressed with SMC, the cost function keeps the same shape as the cost function for uncompressed signal only when $\boldsymbol{R}$ is obtained from Cholesky decomposition. If $\boldsymbol{R}$ is the identity matrix, or a normalized version of it, the cost function is flattened.

Channel with noise

In order to test the performance of the compression methods against each other and to compare them with the performance of uncompressed signal, a high level of noise will be included, of 38 dB Hz. A usual level of noise is between 42 and 48 dB Hz. The same scenario as for the results in 5.1 is kept for the results in 5.2, with noise added.



(a) No compression and compression with SMC.

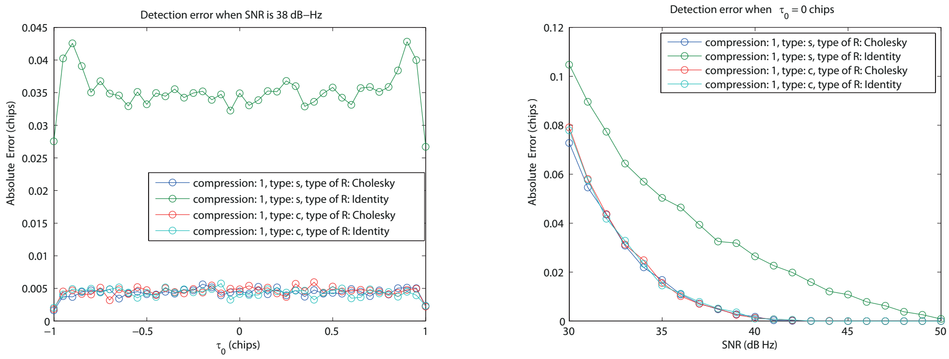
(b) No compression and compression with CMC.

Figure 5.2: Comparison between the cost function for compressed and uncompressed signals, and for different compression types. No multipath components are present and the SNR is 38dB Hz. The cost functions are normalized.

It can be noticed, that for signals compressed with SMC, with $\boldsymbol{R}$ equal with the identity matrix, or with a normalized version of it, the minimum of the cost function does not track the true value of the delay, even if the shape of the cost function is not apparently affected by noise, in contrast with all the other situations presented, in which the cost function shapes are affected by noise, but they track the true delay. A preliminary conclusion is that for compression based on SMC, the matrix $\boldsymbol{R}$ should be obtained from a Cholesky decomposition of $\overline{\boldsymbol{C}}_{\boldsymbol{cor}}$. For the compression based on CMC is seems that the matrix $\boldsymbol{R}$ can be replaced by the identity matrix and yet keeping the performance of the estimation.

In order to probe those conclusions, simulations will be conducted. First, for an SNR of 38 dB Hz and varying the delay of LOS in the interval $[-T_c, T_c]$. Next, for the delay of LOS equal with 0 chips, and varying the SNR in the interval from 30 dB Hz to 50 dB Hz.

The case of $\boldsymbol{R}$ equal with a normalized version of $\boldsymbol{I}$ is not considered further, because from figure 5.2, it shows that there is no difference between it and the case of $\boldsymbol{R}$ equal with $\boldsymbol{I}$.



(a) The SNR is fixed and the delay is varied. **(b)** The delay is fixed and the SNR is varied.

Figure 5.3: Comparison between the detection error with and without compression applied, and for different compression types. No Multipath component is considered. The error is the mean of absolute errors for 1000 realizations.

In figure 5.3a, the absolute error of the ML estimates as a function of the value of the LOS delay in a noisy environment is depicted. The estimation error is hardly affected by the value of the LOS delay. In this section, we will continue with $\tau_0 = 0$ chips.

Both figures 5.3a and 5.3b give the same conclusion, that for the SMC, it is important to include the matrix $\boldsymbol{R}$ with Cholesky decomposition, and that for the CMC it can be left out. The performance of the SMC with Cholesky is similar with the performance of CMC with Identity, at least in the ideal case of no multipath. For the rest of the simulations of this section, we will consider only the SMC with a whitening matrix obtained from Cholesky decomposition.

Another observation is that when the SNR is above 42 dB-Hz, the ML estimation has very good results, with the exception of SMC with $\boldsymbol{R}$ equal with $\boldsymbol{I}$, which we concluded is not a good option.

5.2.2 One multipath component

So far in this work, we have always made the assumption that the ML estimator knows the number of multipath components that it should search for. To evaluate the robustness of the estimator against mismatches in the assumed signal model, and three situations will be investigated, as presented below

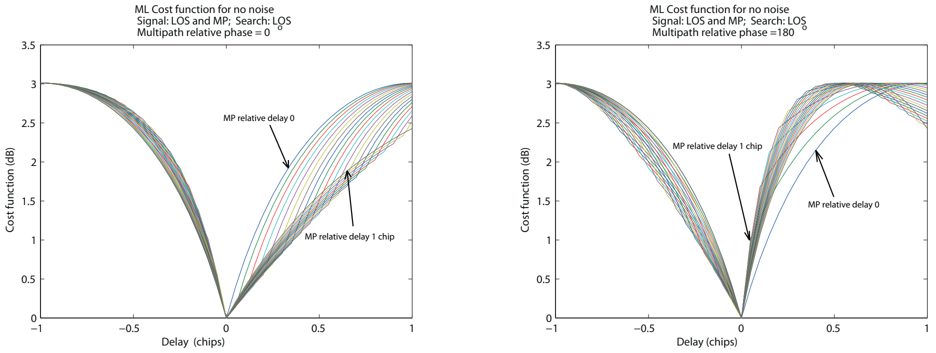
- Correct assumption on the number of multipaths
- Underestimation of the number of multipaths - One multipath is present in the signal, but the algorithm does not search for it
- Overestimation of the number of multipaths - One multipath is assumed, but is not present in the signal

The tests will be conducted without noise and with noise.

Noiseless channel

In a noise free environment, for correct assumption on the number of multipath components, the estimation is perfect for the uncompressed signal. It was tested that the same results are obtained for the compressed signal, with both compression methods.

For underestimation, when the multipath is part of the signal but not searched for, the ML estimation is still perfect, with 0 detection error, for noiseless channel. To show the effect of the multipath component which is not searched for, on the ML cost function, the figure 5.4 is presented for the uncompressed signal. The cost function has been evaluated for a set of different values of the relative multipath component delay, ranging from 0 to 1 chips. Tests have been conducted and the results look the same for the two compression methods presented.



(a) The MP is in phase with the LOS. **(b)** The MP is out of phase with the LOS.

Figure 5.4: The cost functions, when just the LOS is searched, but the signal has also multipath, with magnitude $A_1 \approx 0.5A_0$. The overlapping cost functions correspond to different values of the multipath delay. No noise is present.

For overestimation, the amplitude constraints are needed to avoid an error which can go up to two chips for the line of sight component. With constraints on the amplitudes, the error is 0. See figure E.1 in the appendix, for the simulation results without the amplitude constraints. The only component from the signal can be seen as a summation of two components having the same delay. But the ML estimation can not search for equal delays, and it will assign the true value to one of the components, giving the other component a value in the space allowed by the delay constraints. For more details, see section 3.3.3. In section 3.3.3 it was explained the case of one multipath delay which is so close to the line of sight delay that can be considered equal.

Channel with noise

Next, the accuracy of the ML estimation, under noisy conditions, will be tested.

Different values of the phase of the multipath, relative to the phase of the line of sight, will be investigated. The reason for it is that the relative phase of the multipath component with respect to the LOS component may have a strong effect on the SNR of the signal, depending on whether the components add up constructively or destructively.

Correct assumption on the number of multipaths The signal is composed from the line of sight and one multipath and the algorithm is searching for them.

The level of 42 dB-Hz is chosen, as it is a medium-high level of noise. The delay τ_0 is fixed at 0 chips, and the delay of the multipath is varied in the interval given by $[\tau_0, T_c]$.

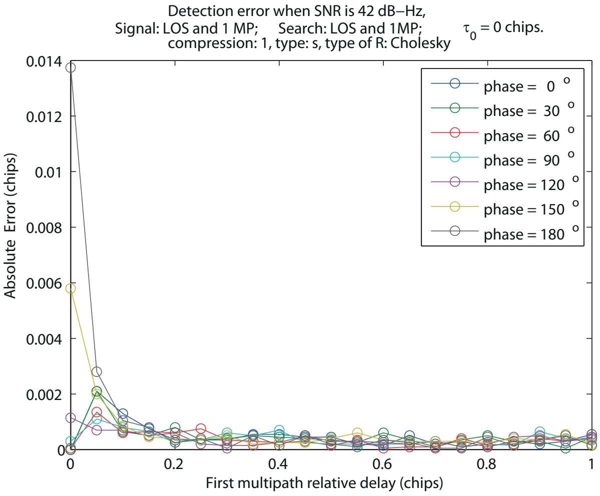


Figure 5.5: Error between the true delay and the detected delay. The delay of the direct component is 0 chips. One Multipath component is present and the algorithm searches for it. The error is the mean of absolute errors for 1000 realizations.

By analysing the results in figure 5.5, it is observed that the detection error is in general very low, excepting the case when the multipath component is very close to the LOS. For the out of phase components, the error, when the multipath delay is equal with the LOS delay, is growing, reaching the level of 0.014 chips. This behaviour can be explained by the fact that the out phase multipath is decreasing the power of the total signal component, as shown below,

$$\begin{aligned}P_{s.total} &= \frac{1}{N}\sum_{n=0}^{N}(a_0c_n(\tau_0)+a_1c_n(\tau_1))(a_0c_n(\tau_0)+a_1c_n(\tau_1))^* \\ &= \frac{1}{N}\sum_{n=0}^{N}(A_0e^{j0}c_n(\tau_0)+A_1e^{j\pi}c_n(\tau_0))(A_0e^{j0}c_n(\tau_0)+A_1e^{j\pi}c_n(\tau_0))^* \\ &= \frac{1}{N}\sum_{n=0}^{N}(A_0-A_1)^2 = (A_0-A_1)^2.\end{aligned} \tag{5.1}$$

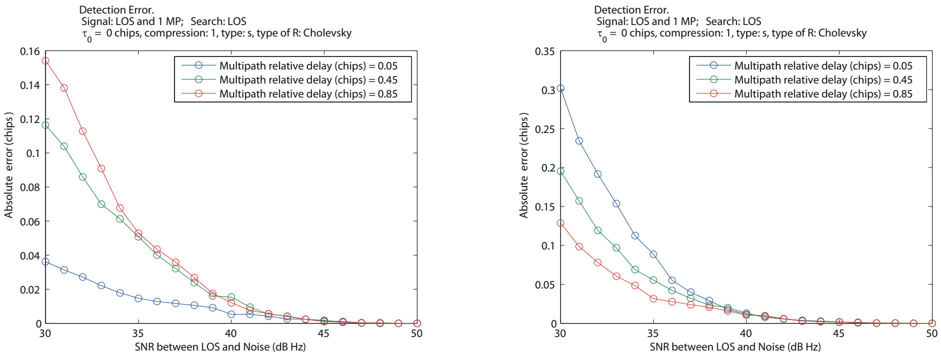
In the given simulations, $A_1 \approx 0.5A_0$, which gives $P_{s.total} = (0.5A_0)^2 = 0.5^2P_0$. The SNR for the total signal component is

$$\begin{aligned}\mathrm{SNR}_{dBHz} &= 10\log_{10}(\frac{P_{s.total}}{N_0}) = 10\log_{10}(\frac{0.5^2P_0}{N_0}) \\ &= 10\log_{10}(\frac{P_0}{N_0}) + 10\log_{10}(0.5^2) = (42 - 6.0206) \text{ dB Hz} \\ &= 35.9794 \text{ dB Hz},\end{aligned} \tag{5.2}$$

and so, this situation can be compared with the one in which the signal is composed just by the LOS, and with an SNR of 35.9794 dB Hz. This can be checked in figure 5.3b, for approximately 35.9794 dB Hz.

Underestimation of the number of multipaths For the case when the signal is composed by the LOS and one multipath, but the algorithm is searching just the LOS, it was shown that for delays on the sample grid, without noise, there is no detection error, even if the cost function shape is distorted by the multipath component. In the presence of noise, the distortions in the cost function will produce detection errors and the characteristics of those errors will be investigated.

First, simulations are conducted in order to evaluate the absolute detection error, for different SNR's and for few values of the multipath delay, as shown in figure 5.6.

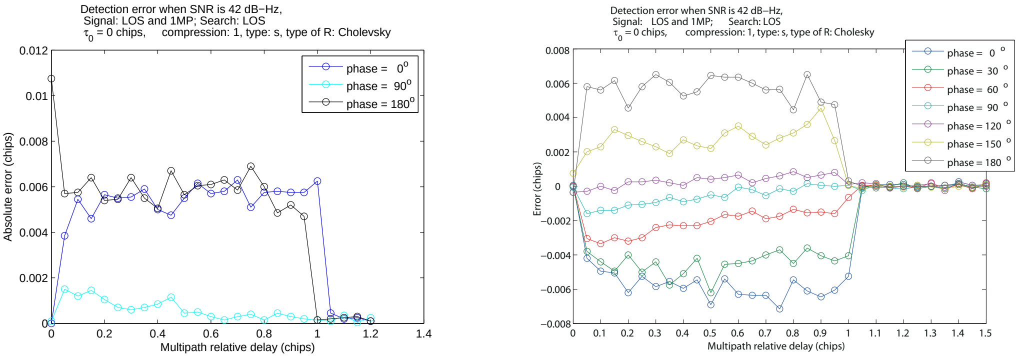


(a) Multipath in phase with the LOS. **(b)** Multipath out of phase with the LOS.

Figure 5.6: Error between the true delay and the detected delay. One multipath component is present but the algorithm doesn't search for it. The line of sight and the multipath are in phase. The error is the mean of absolute errors for 1000 realizations.

It can be observed in figure (5.6) that the out of phase components produces a higher detection error than the in-phase components and this is in line with the findings from equation 5.2, where it was shown that for out of phase, the total SNR of the signal is decreased and then a higher detection error is expected.

The ML estimator is unbiased, so it is expected that for a large number of realizations, the errors to cancel each other if are not considered in absolute values. But when the signal model used is mismatched, the ML estimator cannot be expected to be unbiased any more. This case is illustrate in figure 5.7b, for the case of underestimation.



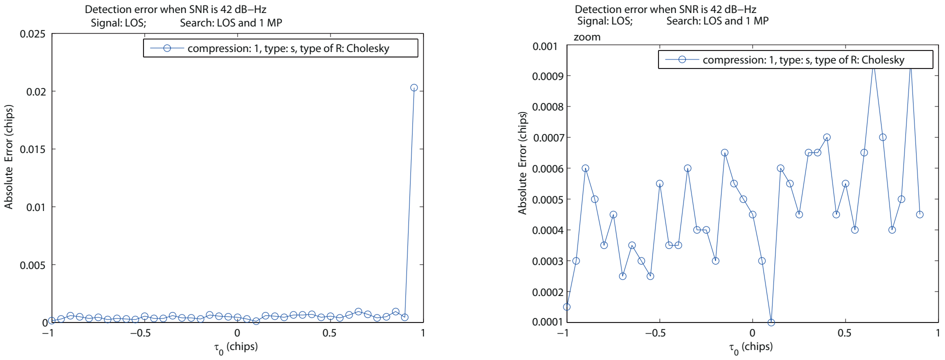
(a) The error is the mean of the absolute errors for 1000 realizations.

(b) The error is the mean of the errors for 1000 realizations.

Figure 5.7: Error between the true delay and the detected delay. The delay of the direct component is 0 chips. One Multipath component is present but the algorithm doesn't search for it.

In figure 5.7a, the same conclusion, as the one from figure 5.6, can be made. That the out of phase multipaths have a much sever effect on the estimation accuracy.

Overestimation of the number of multipaths When one multipath is assumed by the algorithm, but is not present in the signal, it was discussed for the case without noise, that without the amplitude constraints the errors would be large and with the constraints it goes to 0. In the presence of noise, an error is to be expected and it can be seen in figure 5.8 that this error is very low. The only exception is at the last delay on the search grid, when $\tau_0 = 0.95$ chips. It does not matter too much, because in the receiver, in the tracking stage, the value $\tau_0 = 0.95$ chips is not a realistic one, the delays being usually smaller.



(a) τ_0 in interval $[-T_c, T_c - T_s]$.

(b) zoom for τ_0 in interval $[-T_c, T_c - 2T_s]$.

Figure 5.8: Error between the true delay and the detected delay. One multipath component is assumed by the algorithm, but the signal is composed just from the direct component. The error is the mean of the absolute errors for 1000 realizations.

5.2.3 Two multipath components

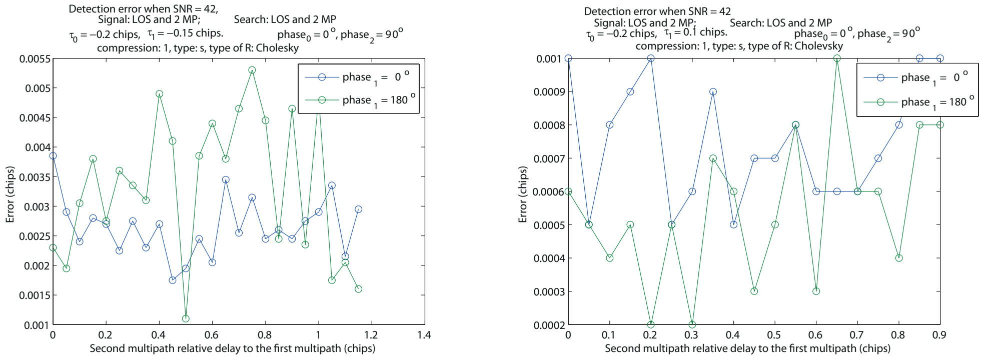
For a signal composed from the direct component and two multipaths, with delays that can be only on the sample grid, the estimation error is 0 in a noise free environment.

For an environment affected by noise, when the signal model is has the correct number of multipaths, the estimation error is strongly dependent on the difference between the direct component and the stronger multipath. In the current set-up, we consider the line of sight at 42 dB Hz, the first multipath at 36 dB Hz, and the second multipath at 34 dB Hz. So it is expected that the estimation error will be dependent on the difference between the LOS and the first multipath and not that much on the third component. To illustrate this, and also to investigate the level of error, two simulation scenarios will be considered, one with the first multipath very close to LOS and the second with the first multipath at a higher distance to LOS.

More details about the set-up are presented next.

- $\tau_0 = -0.2$ chips and $\text{phase}_0 = 0^o$. We chose τ_0 different than 0 chips just to vary a bit the simulations. And a negative value is possible because the PRN code is coarsely aligned with $+/-1$ chips at the beginning of the tracking loop.
- $\text{phase}_1 = \{0^o, 180^o\}$.
- $\tau_2 = [\tau_1, T_c]$, with a step of T_s and $\text{phase}_2 = 90^o$.
- First scenario: $\tau_1 - \tau_0 = 1T_s$, so $\tau_1 = -0.15$ chips
- Second scenario: $\tau_1 - \tau_0 = 6T_s$, so $\tau_1 = 0.1$ chips

The results are shown in figure 5.9, and are as expected. The delay of the strongest multipath component, relative to the LOS, is the one that determines mostly the error to get.



(a) τ_1 is spaced with 1 sample from τ_0. **(b)** τ_1 is spaced with 6 samples from τ_0.

Figure 5.9: Error between the true delay and the detected delay. The signal is composed by the line of sight and two multipaths and the algorithm is searching for them. The error is the mean of the absolute errors for 1000 realizations.

5.3 Continuous delays

In the previous section, the delays were on the grid defined by the sample period T_s. In practice, the delays can take any value between the grid and this is the situation that will be investigated in this section.

It was noticed in the previous section that, for both compression methods discussed, the compression does not affect the performance of the estimator. This investigation will be conducted in current section for the case of continuous delays.

The performance of the ML estimation algorithm will be compared against the performance of the classical DLL, with 0.1 and 0.3 early-late spacing, for channels with and without noise.

The compression is considered with full correlators, spaced with T_s. There will be 41 correlators for the signal matched correlators and 61 correlators in total for the code matched correlators.

For all simulations in this section, the constraint on the amplitudes is considered.

5.3.1 Search algorithm

The delay search in the previous section was made only in one step, on the coarse grid with T_s spacing, and the chip period T_c was an integer of the sample period. The ration between the chip rate and the sample frequency was 20.

In order to be able to have changes in the PRN code for delays smaller than one sample period, there are more implementation approaches. In this project, the approach according to which the sample frequency is not an integer of the chip rate is chosen, due to its simple implementation. The performance of the method is sensitive to the values of the ratio between the chip rate and the sample frequency and to the length of the observation interval. Those are calibration parameters and if they are not chosen properly, some delays will produce the same change in the PRN code, and the ML estimation will not be able to distinguish between them. For delays in interval $[-T_c, T_c]$, with a step of $0.01T_s$, it was noticed that, for an observation interval of 0.01 seconds, a good ration between the chip rate and the sample frequency is equal with 20.1111.

The delay search could be made directly on a fine grid, but that will have a high computational burden. To be more efficient, the search will be made in two steps. First on the coarse grid with T_s spacing, and the second search around the initial solution, with a spacing of $0.1T_s$. For a good selection of observation interval and ratio between the sample frequency and chip rate, the resolution of the search is increased 10 times comparing to one step search on the coarse grid. The error that is expected in the estimation is half of the resolution of the grid, so half of $0.1T_s$. If ϵ is the estimation error in chips, then $\epsilon_{maxim} = 2.49 \cdot 10^{-3}$ chips, which is the equivalent of 0.7286 meters and this is an acceptable error in GNSS.

5.3.2 No Multipath Component

As already mentioned, the delays are inside the interval $[-T_c, T_c]$, with a step of $0.01T_s$.

For the case of no noise, the resolution error, with $\epsilon_{maxim} = 2.49 \cdot 10^{-3}$ chips. This is for both not compressed and for compressed signals, with the compression methods discussed. For illustration, in figure 5.10 we consider the delay from $[-0.05T_c, 0.05T_c]$. It has been tested, that this uniform behaviour is for the full interval $[-T_c, T_c]$.

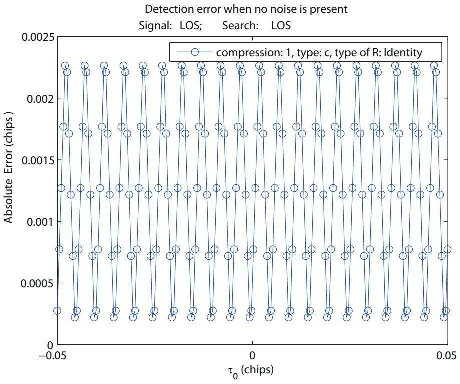


Figure 5.10: Error between the true delay and the detected delay. The signal has just the line of sight, and the algorithm searches just for it. The channel is not affected by noise.

For a channel affected by noise, is interesting to test if both the compressed and uncompressed signals have the same tracking performance. Based on results from previous section, from figure 5.3a, it can be concluded that the error has a uniform behaviour over all delays in interval $[-T_c, T_c]$, with a small deviation at the extremes. It could be argue that the uniform behaviour might change because the delays are continuous and the correlators of the compression are spaced with one sample period. That is why we conduct the simulations again in the interval $[-0.2T_c, 0.2T_c]$, and the interval should be enough to show the behaviour.Due to time limitations, the simulations results are not ready at the time of submiting this thesis.

5.3.3 One multipath component

In this subsection, the tracking performance of the ML estimation algorithm will be compared with the performance of DLL with 0.1 and 0.3 early-late spacing. For no multipath, the DLL performs very good in general and is not of interest to compare it with other methods. It makes sense for the ML estimator to be replace the classical DLL only if multipath is expected and if the results are proven to be better.

Noiseless channel

It was shown in section 5.2.1 that in the case of no noise, for one multipath component, when the constraint on the amplitudes is included, the error is 0, even if the number of multipaths is assumed correctly or not. It will be investigated if this is the case for continuous delays. Some numerical issues are to be expected.

Correct assumption on the number of multipaths When correct assumptions on the signal model are made, for one multipath component, we investigate the performance of the ML estimator comparing with the DLL, in the case of no noise.

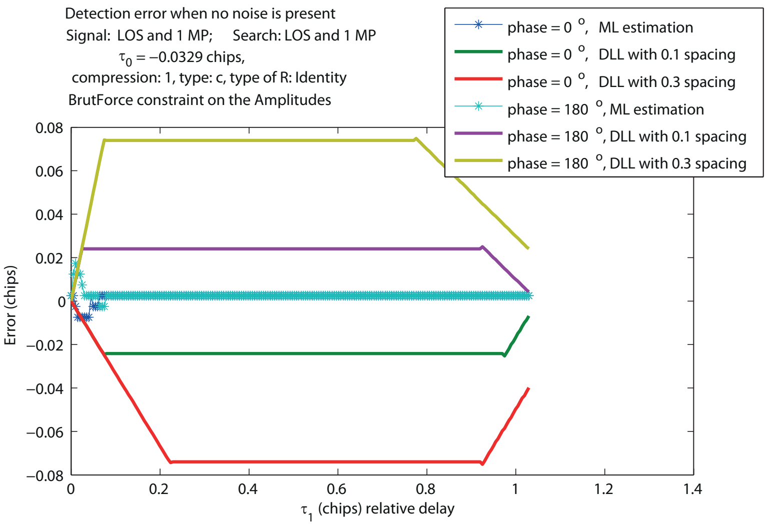
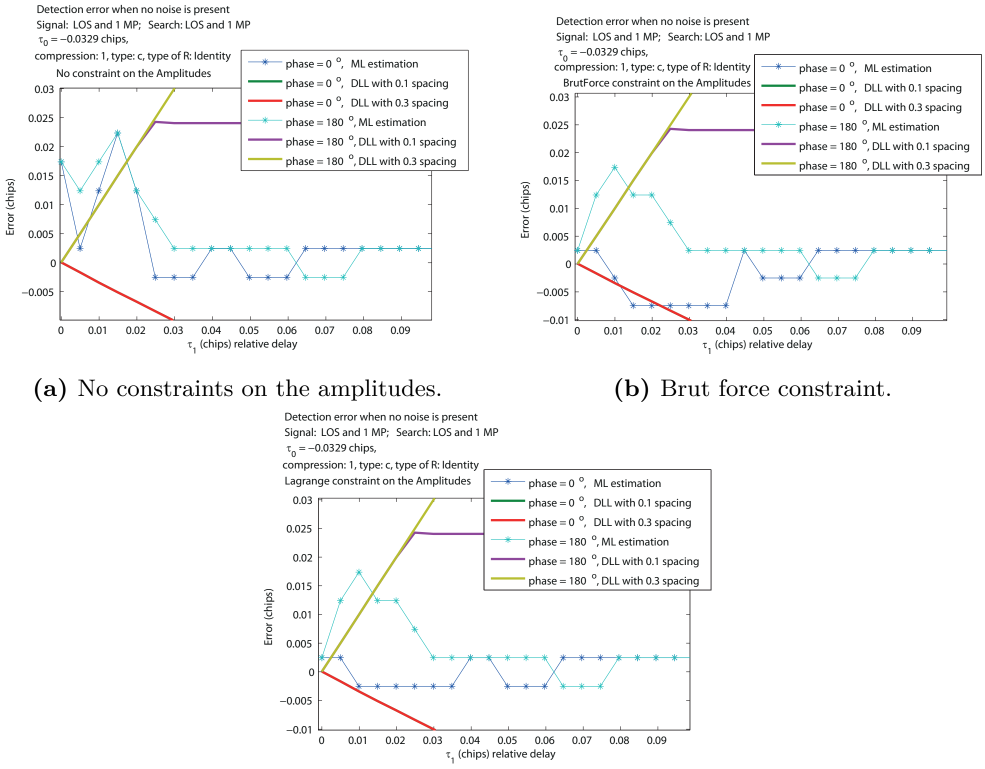


Figure 5.11: Error between the true delay and the detected delay. The signal is composed by the line of sight and one multipath and the algorithm is searching for them. The channel is not affected by noise.

It can be noticed from figure 5.11 for relative delays smaller than $0.1T_c$, some errors can be noticed in the ML estimation. For the uncompressed signals they are usually smaller, as it can be seen in appendix E.2, where this case was tested for the uncompressed signal and for the one compressed with SMC. Also they differ if the constraint on the amplitudes is with Lagrange or with BrutForce, as it can be seen in the next figure.



(a) No constraints on the amplitudes.

(b) Brut force constraint.

(c) Lagrange constraint.

Figure 5.12: Error between the true delay and the detected delay. The signal is composed by the line of sight and one multipath and the algorithm is searching for them. The channel is not affected by noise. The compression is based on Code Matched Correlators.

The case presented in figure 5.12 is a very bad one, in which the amplitude constraints can not fix much of the error. But in general they remove an important part of it. And usually Lagrange performs better in those situations.

Underestimation of the number of multipaths For the case of one multipath present in the signal, but mismatched signal model with just the line of sight, it is tested how the extra component, which is not searched for, affects the estimation accuracy and it is compared with the DLL accuracy, as shown in figure 5.13.

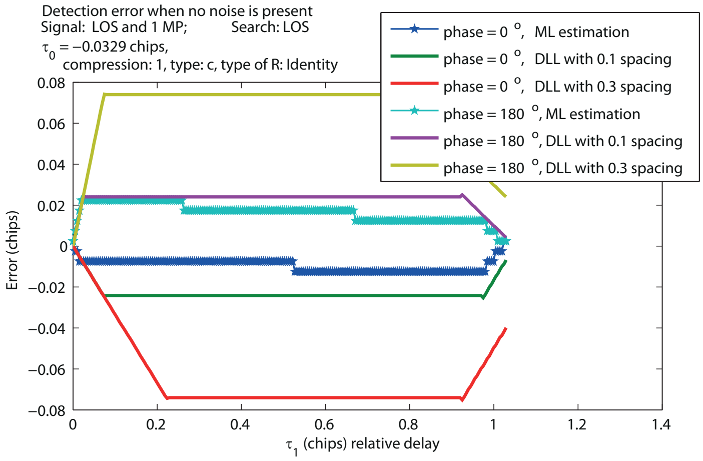
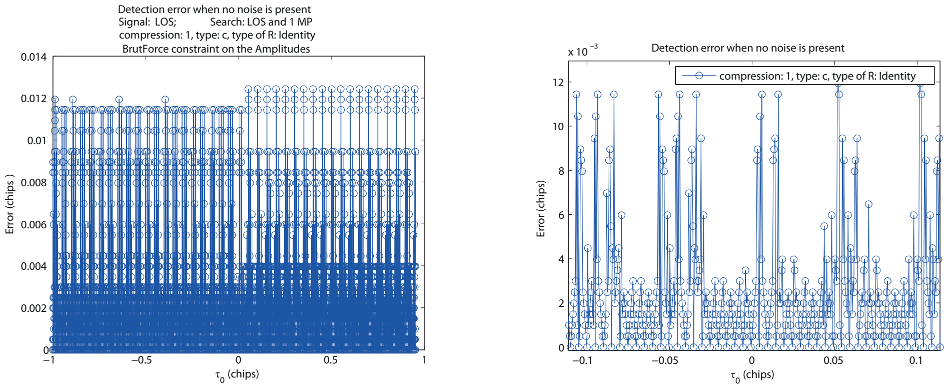


Figure 5.13: Error between the true delay and the detected delay. The signal has LOS and 1MP, but the algorithm searches just for LOS. The channel is not affected by noise.

If when the delays were on the grid, we couldn't see any error for the case of no noise, for the continuous delays case, the results look much different. When the multipath is out of phase relative to the direct component, the error of the ML is almost equal with that of the DLL with 0.1 spacing. Simulations have been conducted also for signals compressed with SMC and for no compression, with the results shown in appendix in E.3. The results are much better for the uncompressed signal. The explanation is subject to further research.

Overestimation of the number of multipaths For the case when no multipath component is present, and mismatched signal model that assumes one multipath, the results are shown in figure 5.14. No comparison with the DLL was made, because in this case, the performance of the DLL is very good, and we gain no information by comparing them.



(a) τ_0 in the interval $[-T_c, T_c]$.

(b) zoom - τ_0 in the interval $[-0.1T_c, 0.1T_c]$.

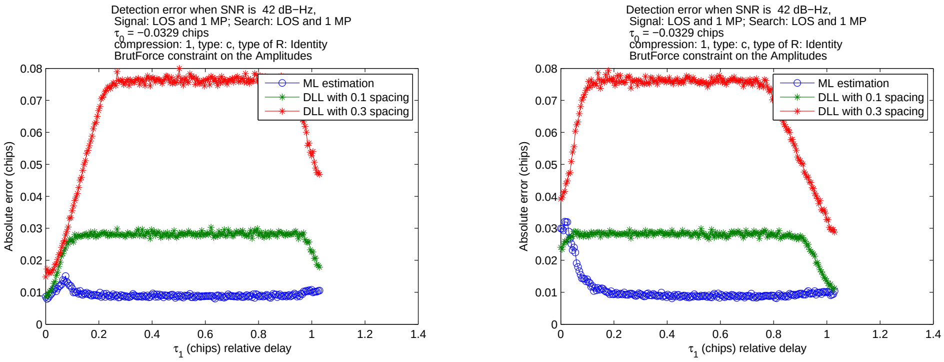
Figure 5.14: Error between the true delay and the detected delay. The signal has just the line of sight, but the algorithm searches for LOS and one MP. The channel is not affected by noise.

Tests have been conducted for the same set-up as in figure 5.14, but for signals compressed with SMC and for not compressed signals, and the results are much better when the signal is not compressed, the error being equal with the resolution of the grid. Again, as in the underestimation case, this is subject to further research.

Channel with noise

The simulations in this chapter are meant to illustrate how the DLL and the ML estimator performance are affected by noise in situations which are close to real-life scenarios.

Correct assumption on the number of multipaths The results presented next, for the case of line of sight and one multipath and correct signal model, in a noisy environment, are considered very important for the comparison between the DLL and the ML estimator. Those will be the ones used to make a initial conclusion for the comparison. But it has to be noted that depending on the set-up of the system, like the length of the observation interval, the SNR and some other parameters, the point where the lines intersect will be at different positions. So the results should be used just for an initial feeling of the comparison between the DLL and ML. For some clear conclusions, more thorough tests are needed.



(a) The multipath component is in phase with the LOS.

(b) The multipath component is out of phase with the LOS.

Figure 5.15: Error between the true delay and the detected delay. The signal is composed by the line of sight and one multipath and the algorithm is searching for them. The channel is affected by noise.

It can be noticed in 5.15, that for the out of phase multipaths, for very small relative delays, there is a region where the DLL with 0.1 spacing performs better than the DLL. in real life situations, the phase of the multipath components has a random behaviour, so a mean between the 2 figures in 5.15 is usually expected.

Underestimation of the number of multipaths One multipath is present in the signal, but the algorithm assumes just the line of sight.

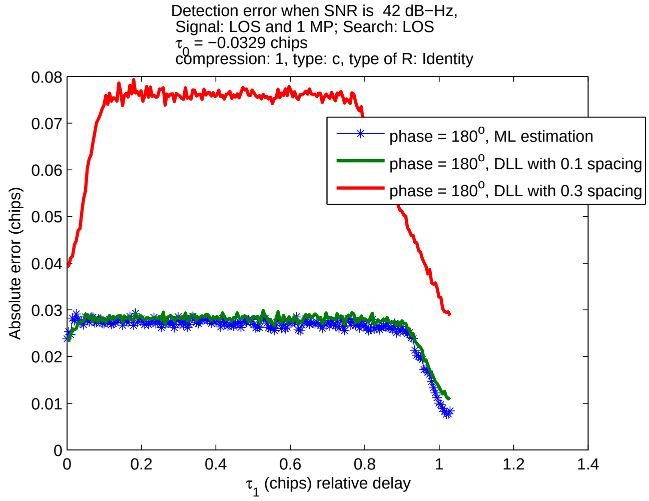


Figure 5.16: Error between the true delay and the detected delay. The signal has LOS and 1MP, but the algorithm searches just for LOS. The multipath is out of phase with the LOS. The channel is affected by noise.

For the case of underestimation presented in figure 5.16, the performance of the ML estimator is almost the same with the performance of the DLL with 0.1 spacing. Base on the results from no noise channel for underestimation, is to be supposed that for no compression, the performance of the ML would be much better than the DLL.

Overestimation of the number of multipaths One multipath is assumed by the algorithm, but is not present in the signal

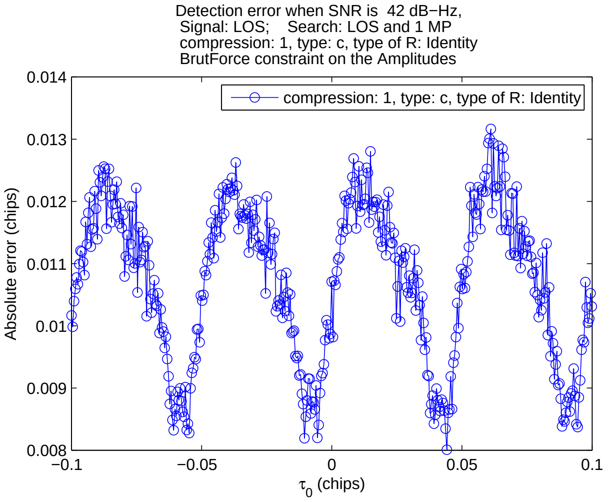


Figure 5.17: Error between the true delay and the detected delay. The signal has just the line of sight, but the algorithm searches for LOS and one MP. The channel is affected by noise.

For overestimation, comparing with the case without noise, the maximum error is just slightly larger, but the distribution of the intermediate errors is different.

5.3.4 Two multipath components - Noiseless channel

Due to time limitations, for the case of two multipath components, only the simulations for a channel not affected by noise are presented.

The ML estimation error is strongly dependent on how close the first multipath is to the line of sight and what phase difference they have. Is to be expected that when the first multipath is very close to the line of sight, the ML estimator will give worst results than the DLL. And when the fist multipath is farther away from the line of sight, the ML estimator to give better results than the DLL. This will be shown in the next simulations.

The set-up of the system is:

- $\tau_0 = -0.0329$ chips and phase$_0 = 0^o$. We chose τ_0 different than 0 chips just to vary a bit the simulations. And a negative value is possible because the PRN code is coarsely aligned with $+/-1$ chips at the beginning of the tracking loop.
- phase$_1 = \{0^o, 180^o\}$.
- $\tau_2 = [\tau_1, T_c]$, with a step of $0.1T_s$ and phase$_2 = 90^o$.
- First scenario: $\tau_1 - \tau_0 = 0.3T_s$, so $\tau_1 = -0.017983$ chips
- Second scenario: $\tau_1 - \tau_0 = 6T_s$, so $\tau_1 = 0.26544$ chips

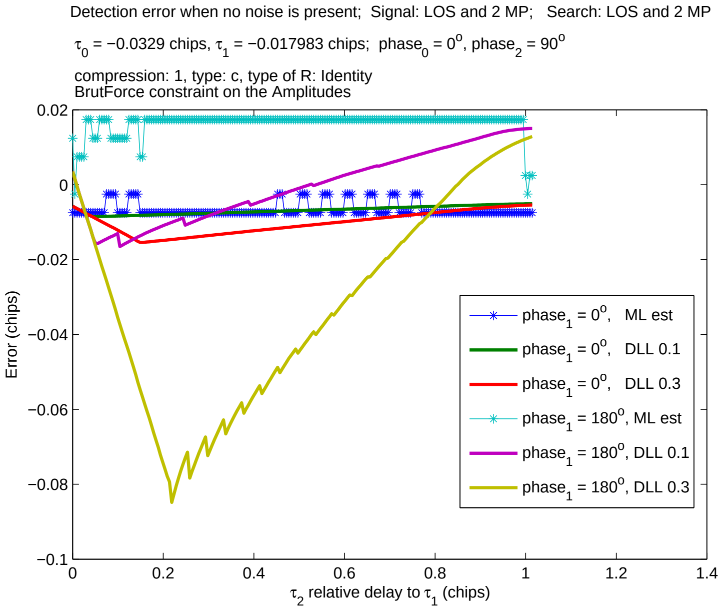


Figure 5.18: Error between the true delay and the detected delay. The first multipath is very close to the line of sight, at a distance of $0.3T_s$.

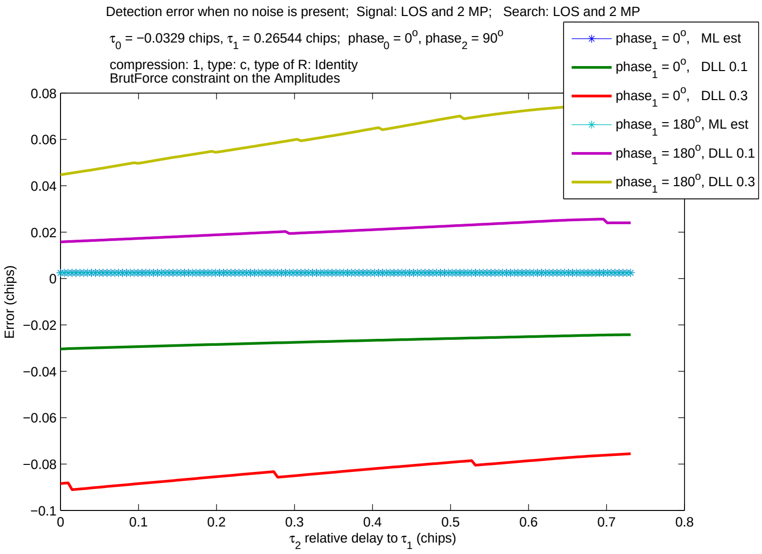


Figure 5.19: Error between the true delay and the detected delay. The first multipath is far from to the line of sight, at a distance of $6T_s$.

Based on the results in figures 5.18 and 5.19, we can conclude that when the strong multipath components is very close to the line of sight, the DLL performas better. For the rest of the situations, the proposed and implemented ML estimation algorithm outperforms the DLL, when the signal model is accurate.

Chapter 6

Conclusion

The aim of this thesis has been to research the joint estimation of the delays of the direct and the reflected signals in GNSS systems and to propose and implement an efficient solution, which can outperform the classical DLL.

In order to achieve this goal, two main directions of research of the maximum likelihood estimation in GNSS multipath mitigation, given by Weill in [15] and Selva in [11], have been reviewed in detail. Based on the findings, a maximum likelihood estimation algorithm was proposed.

Two compression methods, with code matched correlators and signal matched correlators, have been studied and successfully implemented together with the joint maximum likelihood estimator. For increasing the efficiency of the estimation, constraints on the amplitudes of the components have been imposed. Two type of constraints were developed, one proposed by Weill, with Lagrange multipliers, and another method, developed in this thesis, following a brut force.

A simulation environment was successfully developed in Matlab and a DLL performance assessment capability was implemented, which simulates the solution given by the DLL in a real receiver. Using this tool, a detailed performance assessment of the maximum likelihood estimator was performed and compared with the DLL performance.

The results obtained indicate that the maximum likelihood estimator outperforms the DLL in general, but exceptions can occur if the stronger multipath components is very close and out of phase with respect to the line of sight. The performance of the maximum likelihood estimator depends on the correct assumptions about the number of multipath components.

The compression level can be increased, yet keeping a good accuracy of the estimation. But this was not detailed in the thesis and a higher level of compression is subject of further research.

The band limiting effect in the receiver has not been a topic for this thesis and how it impacts the accuracy of the estimation is subject to further investigation. Also, for a complete picture, simulations should be done on signals generated outside the simulator and for dynamic channels during the observation interval.

Although there are still aspects which need to be investigated, the proposed solution is a promising and feasible approach to mitigate the positioning errors induced by multipath.

As directions of further development, the next step should be the implementation of the proposed maximum likelihood estimator in the receiver. Also, other estimators, which perform better in a dynamic channel, and which would be considered feasible for the implementation, can be researched and developed, i.e. the particle filter.

Bibliography

[1] Sources of errors in gps, 04 2009. URL `http://www.kowoma.de/en/gps/errors.htm`.

[2] Pratibha B. Anantharamu, Daniele Borio, and Gérard Lachapelle. Sub-carrier shaping for boc modulated gnss signals. *EURASIP J. Adv. Sig. Proc.*, 2011:133, 2011. URL `http://dblp.uni-trier.de/db/journals/ejasp/ejasp2011.html#AnantharamuBL11`.

[3] Wolfgang Damm. Signal-to-noise, carrier-to-noise, ebno, 11 2010. URL `http://www.noisecom.com/resource-library/webinars/sn-cn-ebno-webinar`.

[4] Yinyu Ye David G. Luenberger. *Linear and Nonlinear Programming*. Springer Science+Business Media, LLC, 2008. ISBN 978-0-387-74502-2.

[5] P.D. Groves. *Principles of GNSS, inertial, and multi-sensor integrated navigation systems*. GNSS technology and applications series. Artech House, 2008. ISBN 9781580532556. URL `http://books.google.de/books?id=WssZAQAAIAAJ`.

[6] E.D. Kaplan and C.J. Hegarty. *Understanding GPS: Principles and Applications*. Artech House mobile communications series. Artech House, 2006. ISBN 9781580538947. URL `http://books.google.de/books?id=V_5OAAAAMAAJ`.

[7] Steven M. Kay. *Fundamentals of statistical signal processing: estimation theory*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1993. ISBN 0-13-345711-7.

[8] Qingchong Liu. Doppler measurement and compensation in mobile satellite communications systems. *Military Communications Conference Proceedings, 1999. MILCOM 1999. IEEE*, pages 316 – 320, vol1, 1999.

[9] Lepinsy Chanthalansy Mark Petovello, Brad Badke and Aboelmagd Noureldin. Gnss solutions: Carrier-to-noise density and ai for ins/gps integration. *InsideGNSS*, pages 20–29, September/ Octomber2009.

[10] Jason Jones Patrick C. Fenton. The theory and performance of novatel inc.'s vision correlator, journal = ION GNSS 18th International Technical Meeting of the Satellite Division, Long Beach, CA, year = 13-16 September 2005, pages = 2178 - 2186, owner = Ioana, timestamp = 2014.06.09.

[11] J. Selva Vera. *Efficient Multipath Mitigation in Navigation Systems*. PhD thesis, Dpt. of Signal Theory and Communications, UPC, 2004 February. URL `https://spcom.upc.edu/documents/Thesis_Selva.pdf`.

[12] L. R. Weill. Achieving theoretical accuracy limits for pseudoranging in the presence of multipath. *Proceedings of the ION GPS-95*, pages 1521–1530.

[13] L. R. Weill. C/a code pseudoranging: How good can it get? *Proceedings of the ION GPS-94 7th International Technical Meeting, Salt Lake City, Utah*, September, 1994.

[14] L. R. Weill. Multipath mitigation using modernized gps signals: How good can it get? *ION GPS 2002, Portland, Oregon*, pages 24–27, September, 2002.

[15] L.R. Weill. Achieving theoretical bounds for receiver-based multipath mitigation using galileo os signals. *Proceedings of the 19th International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS 2006), Fort Worth, TX*, pages 1035 – 1047, September 2006.

Appendix A

SNR in DLL

After correlation in the DLL, the SNR will be the ratio between the power of the signal part from the correlation and the power of the noise part from the correlation. We consider the correlation between the received signal $a_0 c(t-\tau_0)+w(t)$, and one of the replica PRN code. If the prompt PRN replica is considered, then we are interested in the value of the correlation function at the prompt delay τ_p, which is given by

$$\begin{aligned} r_{sc}(\tau_p) &= \sum_{n=1}^{N} c_{rep}(t_n - \tau_p) s_{DLL}(t_n) \\ &= \sum_{n=1}^{N} c_{rep}(t_n - \tau_p) \left(a_0 c(t_n - \tau_0) + w(t_n) \right) \\ &= a_0 \sum_{n=1}^{N} c_{rep}(t_n - \tau_p) c(t_n - \tau_0) + \sum_{n=1}^{N} c_{rep}(t_n - \tau_p) w(t_n) \end{aligned} \tag{A.1}$$

Depending on the difference between τ_p and τ_0, the summation $\sum_{n=1}^{N} c_{rep}(t_n - \tau_p) c(t_n - \tau_0)$ has a fixed value. For $\tau_p = \tau_0$:

$$\sum_{n=1}^{N} c_{rep}(t_n - \tau_p) c(t_n - \tau_0) = N \tag{A.2}$$

Considering this case further,

$$r_{sc}(\tau_p) = a_0 N + \sum_{n=1}^{N} c_{rep}(t_n - \tau_p) w(t_n) \tag{A.3}$$

In the expression (A.3), it can be noticed that the first term is the signal term and the second is the noise term. The SNR will be given by

$$\mathrm{SNR}_r = \frac{P(N a_0)}{P(\sum_{n=1}^{N} c_{rep}(t_n - \tau_p) w(t_n))} = \frac{N^2 |a_0|^2}{P(\sum_{n=1}^{N} c_{rep}(t_n - \tau_p) w(t_n))}. \tag{A.4}$$

The power of the noise term is

$$\begin{aligned} P(\sum_{n=1}^{N} c_{rep}(t_n - \tau_p)w(t_n)) &= E[(\sum_{n=1}^{N} c_{rep}(t_n - \tau_p)w(t_n))(\sum_{n=1}^{N} c_{rep}(t_n - \tau_p)w(t_n))^*] \\ &= \sum_{n=1}^{N} E[c_{rep}(t_n - \tau_p)w(t_n)c_{rep}(t_n - \tau_p)w^*(t_n)] \\ &= \sum_{n=1}^{N} E[1w(t_n)w^*(t_n)] = N2\sigma^2 \end{aligned} \tag{A.5}$$

and so the SNR of $r_{sc}(\tau_p)$, when $\tau_p = \tau_0$, is equal with

$$\text{SNR}_r = \frac{N^2|a_0|^2}{N2\sigma^2} = N\frac{|a_0|^2}{2\sigma^2} = N \cdot \text{SNR}. \tag{A.6}$$

Appendix B

Amplitudes derivation without constraints

Taking into account that the superscript $(\cdot)^H$ denotes transpose conjugate, it is equivalent with $((\cdot)^*)^T$, where $(\cdot)^*$ denotes conjugate and $(\cdot)^T$ denotes transpose, and the cost function from (4.7) can be expressed as

$$J(\boldsymbol{\theta}) = (\boldsymbol{s}^*)^T\boldsymbol{s} - (\boldsymbol{s}^*)^T\boldsymbol{C}(\boldsymbol{\tau})\boldsymbol{a} - (\boldsymbol{a}^*)^T(\boldsymbol{C}(\boldsymbol{\tau})^*)^T\boldsymbol{s} + (\boldsymbol{a}^*)^T(\boldsymbol{C}(\boldsymbol{\tau})^*)^T\boldsymbol{C}(\boldsymbol{\tau})\boldsymbol{a}. \tag{B.1}$$

From $\nabla_{\boldsymbol{a}} J(\boldsymbol{a}_{ML}, \boldsymbol{\tau}) = \left[\frac{\partial J}{\partial \boldsymbol{a}}, \frac{\partial J}{\partial \boldsymbol{a}^*}\right] = \boldsymbol{0}$, the following is obtained

$$\left[-(\boldsymbol{s}^*)^T\boldsymbol{C}(\boldsymbol{\tau}) + (\boldsymbol{a}^*_{ML})^T(\boldsymbol{C}(\boldsymbol{\tau})^*)^T\boldsymbol{C}(\boldsymbol{\tau}),\ -\big((\boldsymbol{C}(\boldsymbol{\tau})^*)^T\boldsymbol{s}\big)^T + \big((\boldsymbol{C}(\boldsymbol{\tau})^*)^T\boldsymbol{C}(\boldsymbol{\tau})\boldsymbol{a}_{ML}\big)^T\right] = \boldsymbol{0}. \tag{B.2}$$

By going back the to $(\cdot)^H$ superscript, the expression in (B.2) becomes

$$\left[-\boldsymbol{s}^H\boldsymbol{C}(\boldsymbol{\tau}) + \boldsymbol{a}^H_{ML}\boldsymbol{C}(\boldsymbol{\tau})^H\boldsymbol{C}(\boldsymbol{\tau}),\ -\big(\boldsymbol{C}(\boldsymbol{\tau})^H\boldsymbol{s}\big)^T + \big(\boldsymbol{C}(\boldsymbol{\tau})^H\boldsymbol{C}(\boldsymbol{\tau})\boldsymbol{a}_{ML}\big)^T\right] = \boldsymbol{0}. \tag{B.3}$$

The equation is solved for each term in the vector. For the first term,

$$-\boldsymbol{s}^H\boldsymbol{C}(\boldsymbol{\tau}) + \boldsymbol{a}^H_{ML}\boldsymbol{C}(\boldsymbol{\tau})^H\boldsymbol{C}(\boldsymbol{\tau}) = 0, \tag{B.4a}$$

$$\boldsymbol{C}(\boldsymbol{\tau})^H\boldsymbol{C}(\boldsymbol{\tau})\boldsymbol{a}_{ML} = \boldsymbol{C}(\boldsymbol{\tau})^H\boldsymbol{s}. \tag{B.4b}$$

Next, for the second term

$$-\big(\boldsymbol{C}(\boldsymbol{\tau})^H\boldsymbol{s}\big)^T + \big(\boldsymbol{C}(\boldsymbol{\tau})^H\boldsymbol{C}(\boldsymbol{\tau})\boldsymbol{a}_{ML}\big)^T = 0, \tag{B.5a}$$

$$\boldsymbol{C}(\boldsymbol{\tau})^H\boldsymbol{C}(\boldsymbol{\tau})\boldsymbol{a}_{ML} = \boldsymbol{C}(\boldsymbol{\tau})^H\boldsymbol{s}. \tag{B.5b}$$

It can be notices that the expressions in (B.4b) and (B.5b) are the same. If $\boldsymbol{C}(\boldsymbol{\tau})^H\boldsymbol{C}(\boldsymbol{\tau})$ is invertible, the complex amplitude vector $\boldsymbol{a}_{ML}$ has the following solution

$$\boldsymbol{a}_{ML} = (\boldsymbol{C}(\boldsymbol{\tau})^H\boldsymbol{C}(\boldsymbol{\tau}))^{-1}\boldsymbol{C}(\boldsymbol{\tau})^H\boldsymbol{s} \tag{B.6}$$

Appendix C

Amplitudes derivation with Lagrange constraint

The Lagrange function is

$$L(\boldsymbol{\theta}, \lambda) = J_1(\boldsymbol{\theta}) - \lambda h(\boldsymbol{a}) \tag{C.1}$$

Expression $\nabla_{\boldsymbol{a}} L(\boldsymbol{a}_{ML,L}, \boldsymbol{\tau}, \lambda_L) = \mathbf{0}$ is equivalent with

$$\nabla_{\boldsymbol{a}} J_1(\boldsymbol{a}_{ML,L}, \boldsymbol{\tau}) - \lambda_L \nabla_{\boldsymbol{a}} h(\boldsymbol{a}_{ML,L}) = \mathbf{0}, \tag{C.2}$$

where

$$\nabla_{\boldsymbol{a}} h(\boldsymbol{a}_{ML,L}) = \left[\frac{\partial h}{\partial \boldsymbol{a}}, \ \frac{\partial h}{\partial \boldsymbol{a}^*}\right] = \left[\boldsymbol{a}_{ML,L}^H \boldsymbol{\Omega}, \ \left(\boldsymbol{\Omega} \boldsymbol{a}_{ML,L}\right)^T\right], \tag{C.3}$$

and, as shown in (B.3),

$$\nabla_{\boldsymbol{a}} J_1(\boldsymbol{a}_{ML,L}) = \left[-\boldsymbol{s}^H \boldsymbol{C}(\boldsymbol{\tau}) + \boldsymbol{a}_{ML,L}^H \boldsymbol{C}(\boldsymbol{\tau})^H \boldsymbol{C}(\boldsymbol{\tau}), \ -\big(\boldsymbol{C}(\boldsymbol{\tau})^H \boldsymbol{s}\big)^T + \big(\boldsymbol{C}(\boldsymbol{\tau})^H \boldsymbol{C}(\boldsymbol{\tau}) \boldsymbol{a}_{ML,L}\big)^T\right]. \tag{C.4}$$

By solving (C.2), two sets of equations are obtained, as follows

$$\begin{aligned} -\boldsymbol{s}^H \boldsymbol{C}(\boldsymbol{\tau}) + \boldsymbol{a}_{ML,L}^H \boldsymbol{C}(\boldsymbol{\tau})^H \boldsymbol{C}(\boldsymbol{\tau}) - \lambda_L \boldsymbol{a}_{ML,L}^H \boldsymbol{\Omega} &= 0, && \text{(C.5a)}\\ \boldsymbol{a}_{ML,L}^H \big(\boldsymbol{C}(\boldsymbol{\tau})^H \boldsymbol{C}(\boldsymbol{\tau}) - \lambda_L \boldsymbol{\Omega}\big) &= \boldsymbol{s}^H \boldsymbol{C}(\boldsymbol{\tau}) && \text{(C.5b)}\\ \big(\boldsymbol{C}(\boldsymbol{\tau})^H \boldsymbol{C}(\boldsymbol{\tau}) - \lambda_L \boldsymbol{\Omega}\big) \boldsymbol{a}_{ML,L} &= \boldsymbol{C}(\boldsymbol{\tau})^H \boldsymbol{s}, && \text{(C.5c)} \end{aligned}$$

and

$$\begin{aligned} -\big(\boldsymbol{C}(\boldsymbol{\tau})^H \boldsymbol{s}\big)^T + \big(\boldsymbol{C}(\boldsymbol{\tau})^H \boldsymbol{C}(\boldsymbol{\tau}) \boldsymbol{a}_{ML,L}\big)^T - \lambda_L (\boldsymbol{\Omega} \boldsymbol{a}_{ML,L})^T &= 0, && \text{(C.6a)}\\ \boldsymbol{C}(\boldsymbol{\tau})^H \boldsymbol{C}(\boldsymbol{\tau}) \boldsymbol{a}_{ML,L} - \lambda_L \boldsymbol{\Omega} \boldsymbol{a}_{ML,L} &= \boldsymbol{C}(\boldsymbol{\tau})^H \boldsymbol{s}, && \text{(C.6b)}\\ \big(\boldsymbol{C}(\boldsymbol{\tau})^H \boldsymbol{C}(\boldsymbol{\tau}) - \lambda_L \boldsymbol{\Omega}\big) \boldsymbol{a}_{ML,L} &= \boldsymbol{C}(\boldsymbol{\tau})^H \boldsymbol{s}, && \text{(C.6c)} \end{aligned}$$

It can be noticed that the expressions in (C.5c) and (C.6c) are the same. If $\big(\boldsymbol{C}(\boldsymbol{\tau})^H \boldsymbol{C}(\boldsymbol{\tau}) - \lambda_L \boldsymbol{\Omega}\big)$ is invertible, then

$$\boldsymbol{a}_{ML,L} = \left(\boldsymbol{C}(\boldsymbol{\tau})^H\boldsymbol{C}(\boldsymbol{\tau}) - \lambda_L\boldsymbol{\Omega}\right)^{(-1)}\boldsymbol{C}(\boldsymbol{\tau})^H\boldsymbol{s}. \tag{C.7}$$

In order to find λ_L, the new expression for $\boldsymbol{a}_{ML,L}$ is used in

$$h(\boldsymbol{a}_{ML,L}) = \boldsymbol{a}_{ML,L}^H\boldsymbol{\Omega}\boldsymbol{a}_{ML,L} = 0, \tag{C.8}$$

resulting in

$$\left(\left(\boldsymbol{C}(\boldsymbol{\tau})^H\boldsymbol{C}(\boldsymbol{\tau}) - \lambda_L\boldsymbol{\Omega}\right)^{(-1)}\boldsymbol{C}(\boldsymbol{\tau})^H\boldsymbol{s}\right)^H\boldsymbol{\Omega}\left(\left(\boldsymbol{C}(\boldsymbol{\tau})^H\boldsymbol{C}(\boldsymbol{\tau}) - \lambda_L\boldsymbol{\Omega}\right)^{(-1)}\boldsymbol{C}(\boldsymbol{\tau})^H\boldsymbol{s}\right) = 0, \tag{C.9a}$$

$$\left(\boldsymbol{C}(\boldsymbol{\tau})^H\boldsymbol{s}\right)^H\left(\boldsymbol{C}(\boldsymbol{\tau})^H\boldsymbol{C}(\boldsymbol{\tau}) - \lambda_L\boldsymbol{\Omega}\right)^{(-1)}\boldsymbol{\Omega}\left(\boldsymbol{C}(\boldsymbol{\tau})^H\boldsymbol{C}(\boldsymbol{\tau}) - \lambda_L\boldsymbol{\Omega}\right)^{(-1)}\boldsymbol{C}(\boldsymbol{\tau})^H\boldsymbol{s} = 0. \tag{C.9b}$$

With the notation in (3.52) and (3.53), the expression becomes

$$\boldsymbol{r}_{sc}(\boldsymbol{\tau})^H\left(\boldsymbol{R}_{cc}(\boldsymbol{\tau}) - \lambda_L\boldsymbol{\Omega}\right)^{(-1)}\boldsymbol{\Omega}\left(\boldsymbol{R}_{cc}(\boldsymbol{\tau}) - \lambda_L\boldsymbol{\Omega}\right)^{(-1)}\boldsymbol{r}_{sc}(\boldsymbol{\tau}) = 0. \tag{C.10}$$

For the case of $M = 1$,

$$\boldsymbol{\Omega} = \begin{bmatrix} \alpha^2 & 0 \\ 0 & -1 \end{bmatrix}, \tag{C.11}$$

and

$$\boldsymbol{R}_{cc}(\boldsymbol{\tau}) - \lambda_L\boldsymbol{\Omega} = \begin{bmatrix} R_{cc}(\boldsymbol{\tau})_{1,1} - \alpha^2\lambda_L & R_{cc}(\boldsymbol{\tau})_{1,2} \\ R_{cc}(\boldsymbol{\tau})_{2,1} & R_{cc}(\boldsymbol{\tau})_{2,2} + \lambda_L \end{bmatrix}. \tag{C.12}$$

The inverse is given by

$$\left(\boldsymbol{R}_{cc}(\boldsymbol{\tau}) - \lambda_L\boldsymbol{\Omega}\right)^{(-1)} = \frac{1}{\det(\boldsymbol{R}_{cc}(\boldsymbol{\tau}) - \lambda_L\boldsymbol{\Omega})}\begin{bmatrix} R_{cc}(\boldsymbol{\tau})_{2,2} + \lambda_L & -R_{cc}(\boldsymbol{\tau})_{1,2} \\ -R_{cc}(\boldsymbol{\tau})_{2,1} & R_{cc}(\boldsymbol{\tau})_{1,1} - \alpha^2\lambda_L \end{bmatrix}, \tag{C.13}$$

where

$$\begin{aligned}\det(\boldsymbol{R}_{cc}(\boldsymbol{\tau}) - \lambda_L\boldsymbol{\Omega}) &= \left(R_{cc}(\boldsymbol{\tau})_{1,1} - \alpha^2\lambda_L\right)\left(R_{cc}(\boldsymbol{\tau})_{2,2} + \lambda_L\right) - R_{cc}(\boldsymbol{\tau})_{1,2}R_{cc}(\boldsymbol{\tau})_{2,1} \\ &= R_{cc}(\boldsymbol{\tau})_{1,1}R_{cc}(\boldsymbol{\tau})_{2,2} - R_{cc}(\boldsymbol{\tau})_{1,2}R_{cc}(\boldsymbol{\tau})_{2,1} - \alpha^2\lambda_L R_{cc}(\boldsymbol{\tau})_{2,2} + \lambda_L R_{cc}(\boldsymbol{\tau})_{1,1} \\ &= \det(R_{cc}) - \lambda_L\left(\alpha^2 R_{cc}(\boldsymbol{\tau})_{2,2} - R_{cc}(\boldsymbol{\tau})_{1,1}\right),\end{aligned} \tag{C.14}$$

with $\det(R_{cc})$ being the determinant of $\boldsymbol{R}_{cc}(\boldsymbol{\tau})$ given by

$$\det(R_{cc}) = R_{cc}(\boldsymbol{\tau})_{1,1}R_{cc}(\boldsymbol{\tau})_{2,2} - R_{cc}(\boldsymbol{\tau})_{1,2}R_{cc}(\boldsymbol{\tau})_{2,1}. \tag{C.15}$$

It can be observed that the expression in (C.13) can be restated as

$$\left(\boldsymbol{R}_{cc}(\boldsymbol{\tau}) - \lambda_L\boldsymbol{\Omega}\right)^{(-1)} = \frac{1}{\det(\boldsymbol{R}_{cc}(\boldsymbol{\tau}) - \lambda_L\boldsymbol{\Omega})}(\boldsymbol{A} - \boldsymbol{B}), \tag{C.16a}$$

where

$$\boldsymbol{A} = \left[\begin{array}{cc} R_{cc}(\boldsymbol{\tau})_{2,2} & -R_{cc}(\boldsymbol{\tau})_{1,2} \\ -R_{cc}(\boldsymbol{\tau})_{2,1} & R_{cc}(\boldsymbol{\tau})_{1,1} \end{array}\right], \tag{C.16b}$$

$$\boldsymbol{B} = \left[\begin{array}{cc} -\lambda_L & 0 \\ 0 & \alpha^2\lambda_L \end{array}\right]. \tag{C.16c}$$

Matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ are

$$\boldsymbol{A} = \det(R_{cc})\boldsymbol{R}_{cc}(\boldsymbol{\tau})^{(-1)}. \tag{C.17}$$

$$\boldsymbol{B} = \lambda_L \left[\begin{array}{cc} -1 & 0 \\ 0 & \alpha^2 \end{array}\right] = \lambda_L(-\alpha^2)\boldsymbol{\Omega}^{(-1)}. \tag{C.18}$$

Using (C.16a) in (C.10), the equation becomes

$$\boldsymbol{r}_{sc}(\boldsymbol{\tau})^H \frac{1}{\det(\boldsymbol{R}_{cc}(\boldsymbol{\tau}) - \lambda_L\boldsymbol{\Omega})}(\boldsymbol{A} - \boldsymbol{B})\boldsymbol{\Omega}\frac{1}{\det(\boldsymbol{R}_{cc}(\boldsymbol{\tau}) - \lambda_L\boldsymbol{\Omega})}(\boldsymbol{A} - \boldsymbol{B})\boldsymbol{r}_{sc}(\boldsymbol{\tau}) = 0, \tag{C.19}$$

and by extending it

$$\begin{aligned} &\frac{1}{\det^2(\boldsymbol{R}_{cc}(\boldsymbol{\tau}) - \lambda_L\boldsymbol{\Omega})}\boldsymbol{r}_{sc}(\boldsymbol{\tau})^H(\boldsymbol{A} - \boldsymbol{B})\boldsymbol{\Omega}(\boldsymbol{A} - \boldsymbol{B})\boldsymbol{r}_{sc}(\boldsymbol{\tau}) = \\ &= \frac{1}{\det^2(\boldsymbol{R}_{cc}(\boldsymbol{\tau}) - \lambda_L\boldsymbol{\Omega})}\boldsymbol{r}_{sc}(\boldsymbol{\tau})^H\left(\boldsymbol{A\Omega A} - \boldsymbol{A\Omega B} - \boldsymbol{B\Omega A} + \boldsymbol{B\Omega B}\right)\boldsymbol{r}_{sc}(\boldsymbol{\tau}) \\ &= 0; \end{aligned} \tag{C.20}$$

$$\begin{aligned} \boldsymbol{r}_{sc}(\boldsymbol{\tau})^H\boldsymbol{B\Omega B}\boldsymbol{r}_{sc}(\boldsymbol{\tau}) &= \boldsymbol{r}_{sc}(\boldsymbol{\tau})^H\lambda_L(-\alpha^2)\boldsymbol{\Omega}^{(-1)}\boldsymbol{\Omega}\lambda_L(-\alpha^2)\boldsymbol{\Omega}^{(-1)}\boldsymbol{r}_{sc}(\boldsymbol{\tau}) \\ &= \lambda_L^2\alpha^4\boldsymbol{r}_{sc}(\boldsymbol{\tau})^H\boldsymbol{\Omega}^{(-1)}\boldsymbol{r}_{sc}(\boldsymbol{\tau}). \end{aligned} \tag{C.21}$$

$$\begin{aligned} \boldsymbol{r}_{sc}(\boldsymbol{\tau})^H\boldsymbol{A\Omega A}\boldsymbol{r}_{sc}(\boldsymbol{\tau}) &= \boldsymbol{r}_{sc}(\boldsymbol{\tau})^H\det(R_{cc})\boldsymbol{R}_{cc}(\boldsymbol{\tau})^{(-1)}\boldsymbol{\Omega}\det(R_{cc})\boldsymbol{R}_{cc}(\boldsymbol{\tau})^{(-1)}\boldsymbol{r}_{sc}(\boldsymbol{\tau}) \\ &= \det(R_{cc})^2\boldsymbol{r}_{sc}(\boldsymbol{\tau})^H\boldsymbol{R}_{cc}(\boldsymbol{\tau})^{(-1)}\boldsymbol{\Omega}\boldsymbol{R}_{cc}(\boldsymbol{\tau})^{(-1)}\boldsymbol{r}_{sc}(\boldsymbol{\tau}). \end{aligned} \tag{C.22}$$

$$\begin{aligned} \boldsymbol{r}_{sc}(\boldsymbol{\tau})^H\boldsymbol{A\Omega B}\boldsymbol{r}_{sc}(\boldsymbol{\tau}) &= \boldsymbol{r}_{sc}(\boldsymbol{\tau})^H\det(R_{cc})\boldsymbol{R}_{cc}(\boldsymbol{\tau})^{(-1)}\boldsymbol{\Omega}\lambda_L(-\alpha^2)\boldsymbol{\Omega}^{(-1)}\boldsymbol{r}_{sc}(\boldsymbol{\tau}) \\ &= \lambda_L(-\alpha^2)\det(R_{cc})\boldsymbol{r}_{sc}(\boldsymbol{\tau})^H\boldsymbol{R}_{cc}(\boldsymbol{\tau})^{(-1)}\boldsymbol{r}_{sc}(\boldsymbol{\tau}) \\ &= \boldsymbol{r}_{sc}(\boldsymbol{\tau})^H\boldsymbol{B\Omega A}\boldsymbol{r}_{sc}(\boldsymbol{\tau}). \end{aligned} \tag{C.23}$$

And equation (C.20) becomes

$$\begin{aligned} \frac{1}{\det^2(\boldsymbol{R}_{cc}(\boldsymbol{\tau}) - \lambda_L\boldsymbol{\Omega})}&\big(\lambda_L^2\alpha^4\boldsymbol{R}_{sc}(\boldsymbol{\tau})^H\boldsymbol{\Omega}^{(-1)}\boldsymbol{r}_{sc}(\boldsymbol{\tau}) + \lambda_L 2\alpha^2\det(R_{cc})\boldsymbol{r}_{sc}(\boldsymbol{\tau})^H\boldsymbol{R}_{cc}(\boldsymbol{\tau})^{(-1)}\boldsymbol{r}_{sc}(\boldsymbol{\tau}) \\ &+ \det(R_{cc})^2\boldsymbol{r}_{sc}(\boldsymbol{\tau})^H\boldsymbol{R}_{cc}(\boldsymbol{\tau})^{(-1)}\boldsymbol{\Omega}\boldsymbol{R}_{cc}(\boldsymbol{\tau})^{(-1)}\boldsymbol{r}_{sc}(\boldsymbol{\tau})\big) = 0; \end{aligned} \tag{C.24}$$

The following notations are used, for easier visualization of the equation

$$\begin{aligned}
\alpha^4 \boldsymbol{r}_{sc}(\boldsymbol{\tau})^H \boldsymbol{\Omega}^{(-1)} \boldsymbol{r}_{sc}(\boldsymbol{\tau}) &= a, \\
2\alpha^2 \det(R_{cc}) \boldsymbol{r}_{sc}(\boldsymbol{\tau})^H \boldsymbol{R}_{cc}(\boldsymbol{\tau})^{(-1)} \boldsymbol{r}_{sc}(\boldsymbol{\tau}) &= b, \\
\det(R_{cc})^2 \boldsymbol{r}_{sc}(\boldsymbol{\tau})^H \boldsymbol{R}_{cc}(\boldsymbol{\tau})^{(-1)} \boldsymbol{\Omega} \boldsymbol{R}_{cc}(\boldsymbol{\tau})^{(-1)} \boldsymbol{r}_{sc}(\boldsymbol{\tau}) &= c, \\
\det(R_{cc}) &= d, \\
\alpha^2 R_{cc}(\boldsymbol{\tau})_{2,2} - R_{cc}(\boldsymbol{\tau})_{1,1} &= e.
\end{aligned} \tag{C.25}$$

With the notation in (C.25), equation (C.24) is

$$\frac{(a\lambda_L^2 + b\lambda_L + c)}{(d - \lambda_L e)^2} = 0, \tag{C.26}$$

which gives

$$\lambda_L = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}, \text{subject to } \lambda_L \neq \frac{d}{e}. \tag{C.27}$$

Appendix D

Cost Function minimization following Weill [15]

In [15], only one multipath component is considered. The cost function in (3.17) is extended and expressed as a function of the autocorrelation of the PRN signal, and the correlation between the PRN signal and the observed signal. This is useful because it can be related with the DLL structure in the receiver. When minimizing the cost function, Weill is imposing the constraint that the amplitude of the direct signal should be larger than the amplitude of the multipath, with a certain factor. But in this thesis, this constraint will be ignored and it will be shown that by ignoring it, the result of the minimization coincides with the results reported in [11].

For one multipath component, the signal model in continuous time is

$$s(t) = a_0 c(t - \tau_0) + a_1 c(t - \tau_1) + w(t), \tag{D.1}$$

and in vector form

$$\boldsymbol{s} = a_0 \boldsymbol{c}(\tau_0) + a_1 \boldsymbol{c}(\tau_1) + \boldsymbol{w}, \tag{D.2}$$

with the same notations as before.

for the simplified model in (D.2) can be written as

$$\begin{aligned} J &= (\boldsymbol{s}^H - \boldsymbol{c}(\tau_0)^H a_0^* - \boldsymbol{c}(\tau_1)^H a_1^*)(\boldsymbol{s} - a_0 \boldsymbol{c}(\tau_0) - a_1 \boldsymbol{c}(\tau_1)) \\ &= \boldsymbol{s}^H \boldsymbol{s} + a_0^* a_0 \boldsymbol{c}(\tau_0)^H \boldsymbol{c}(\tau_0) + a_1^* a_1 \boldsymbol{c}(\tau_1)^H \boldsymbol{c}(\tau_1) \\ &\quad + a_1^* a_0 \boldsymbol{c}(\tau_1)^H \boldsymbol{c}(\tau_0) + a_0^* a_1 \boldsymbol{c}(\tau_0)^H \boldsymbol{c}(\tau_1) \\ &\quad - a_0^* \boldsymbol{c}(\tau_0)^H \boldsymbol{s} - a_1^* \boldsymbol{c}(\tau_1)^H \boldsymbol{s} - a_0 \boldsymbol{s}^H \boldsymbol{c}(\tau_0) - a_1 \boldsymbol{s}^H \boldsymbol{c}(\tau_1). \end{aligned} \tag{D.3}$$

When N is large enough, as it is common in the GNSS signal processing, the products of $\boldsymbol{s}$ and $\boldsymbol{c}(\tau_m)$ represent different correlations. As a prof for this see equation (2.9). The auto-correlation of the PRN vector, $\boldsymbol{c}(0)$, is noted with $R_{cc}(0)$ and is given by

$$R_{cc}(0) = \boldsymbol{c}(0)^H \boldsymbol{c}(0) = \boldsymbol{c}(\tau_0)^H \boldsymbol{c}(\tau_0) = \boldsymbol{c}(\tau_1)^H \boldsymbol{c}(\tau_1). \tag{D.4}$$

The correlation between the delayed PRN vector, $\boldsymbol{c}(\tau_m - \tau_q)$, and one not delayed, $\boldsymbol{c}(0)$, is noted with $R_{cc}(\tau_m - \tau_q)$ and is given by

$$\begin{aligned} R_{cc}(\tau_1-\tau_0) &= \boldsymbol{c}(\tau_1)^H\boldsymbol{c}(\tau_0) \\ R_{cc}(\tau_0-\tau_1) &= \boldsymbol{c}(\tau_0)^H\boldsymbol{c}(\tau_1) = R_{cc}(\tau_1-\tau_0)^*. \end{aligned} \tag{D.5}$$

The cross-correlation between the delayed PRN vector, $\boldsymbol{c}(\tau_m)$, and the observed vector, $\boldsymbol{s}$, is noted with $R_{sc}(\tau_m)$ and is given by

$$\begin{aligned} R_{sc}(\tau_0) &= \boldsymbol{c}(\tau_0)^H\boldsymbol{s} \\ R_{sc}(\tau_1) &= \boldsymbol{c}(\tau_1)^H\boldsymbol{s}. \end{aligned} \tag{D.6}$$

By inserting (D.4), (D.5) and (D.6) into (D.3) and by dropping $\boldsymbol{s}^H\boldsymbol{s}$ due to its independence of $\boldsymbol{\theta}$, the new expression for J is

$$\begin{aligned} J(\boldsymbol{\theta}) \propto^e\ & (a_0^*a_0 + a_1^*a_1)R_{cc}(0) + a_1^*a_0R_{cc}(\tau_1-\tau_0) + a_0^*a_1R_{cc}(\tau_1-\tau_0)^* \\ & - a_0^*R_{sc}(\tau_0) - a_1^*R_{sc}(\tau_1) - a_0R_{sc}(\tau_0)^* - a_1R_{sc}(\tau_1)^*, \end{aligned} \tag{D.7}$$

where $\propto^e$ indicates proportionality.

The complex amplitudes, a_0 and a_1, can be expressed as

$$\begin{aligned} a_0 &= a_0^R + ja_0^I \\ a_1 &= a_1^R + ja_1^I, \end{aligned} \tag{D.8}$$

where the superscripts R and I are indicating the real and imaginary parts.

To reduce the size of the minimization of J, following Weill approach, we can equate to zero the partial derivatives of J with respect to the real and imaginary parts of a_0 and a_1.

$$\frac{\partial J}{\partial a_0^R} = 0;\ \frac{\partial J}{\partial a_0^I} = 0;\ \frac{\partial J}{\partial a_1^R} = 0;\ \frac{\partial J}{\partial a_1^I} = 0. \tag{D.9}$$

By solving the system of equations in (D.9), the following expressions are obtained for the real an imaginary part of the amplitudes, as it is shown in D.1,

$$a_0^R = \frac{R_{cc}(0)(R_{sc}(\tau_0)^* + R_{sc}(\tau_0)) - R_{sc}(\tau_1)^*R_{cc}(\tau_1-\tau_0) - R_{sc}(\tau_1)R_{cc}(\tau_1-\tau_0)^*}{2(R_{cc}(0)^2 - R_{cc}(\tau_1-\tau_0)R_{cc}(\tau_1-\tau_0)^*)} \tag{D.10a}$$

$$a_0^I = j\frac{R_{cc}(0)(R_{sc}(\tau_0)^* - R_{sc}(\tau_1)) - R_{sc}(\tau_1)^*R_{cc}(\tau_1-\tau_0) + R_{sc}(\tau_1)R_{cc}(\tau_1-\tau_0)^*}{2(R_{cc}(0)^2 - R_{cc}(\tau_1-\tau_0)R_{cc}(\tau_1-\tau_0)^*)} \tag{D.10b}$$

$$a_1^R = \frac{R_{cc}(0)(R_{sc}(\tau_1)^* + R_{sc}(\tau_1)) - R_{sc}(\tau_1)^*R_{cc}(\tau_1-\tau_0)^* - R_{sc}(\tau_0)R_{cc}(\tau_1-\tau_0)}{2(R_{cc}(0)^2 - R_{cc}(\tau_1-\tau_0)R_{cc}(\tau_1-\tau_0)^*)} \tag{D.10c}$$

$$a_1^I = j\frac{R_{cc}(0)(R_{sc}(\tau_1)^* - R_{sc}(\tau_1)) + R_{sc}(\tau_0)R_{cc}(\tau_1-\tau_0) - R_{sc}(\tau_0)^*R_{cc}(\tau_1-\tau_0)^*}{2(R_{cc}(0)^2 - R_{cc}(\tau_1-\tau_0)R_{cc}(\tau_1-\tau_0)^*)}. \tag{D.10d}$$

The values of a_0^R, a_0^I, a_1^R and a_1^I can be computed for any fixed value of τ_0 and τ_1. The minimization of J is now a two dimensional minimization problem dependent on τ_0 and τ_1.

D.1 Amplitudes derivations

The ML cost function, depending on the real and imaginary parts of the amplitudes, for just one multipath component, can be obtained by combining equations (D.8) and (D.7), as follows

$$\begin{aligned} J =&((A_0^R - jA_0^I)(A_0^R + jA_0^I) + (A_1^R - jA_1^I)(A_1^R + jA_1^I))R_{cc}(0) \\ &+ (A_1^R - jA_1^I)(A_0^R + jA_0^I)R_{cc}(\tau_1 - \tau_0) + (A_0^R - jA_0^I)(A_1^R + jA_1^I)R_{cc}(\tau_1 - \tau_0)^* \\ &- (A_0^R - jA_0^I)R_{sc}(\tau_0) - (A_0^R + jA_0^I)R_{sc}(\tau_0)^* \\ &- (A_1^R - jA_1^I)R_{sc}(\tau_1) - (A_1^R + jA_1^I)R_{sc}(\tau_1)^*, \end{aligned} \tag{D.11a}$$

and by extending it, the cost function becomes

$$\begin{aligned} J =&((A_0^R)^2 + (A_0^I)^2 + (A_1^R)^2 + (A_1^I)^2)R_{cc}(0) \\ &+ (A_0^R A_1^R - jA_0^R A_1^I + jA_0^I A_1^R + A_0^I A_1^I)R_{cc}(\tau_1 - \tau_0) \\ &+ (A_0^R A_1^R + jA_0^R A_1^I - jA_0^I A_1^R + A_0^I A_1^I)R_{cc}(\tau_1 - \tau_0)^* \\ &- (A_0^R - jA_0^I)R_{sc}(\tau_0) - (A_0^R + jA_0^I)R_{sc}(\tau_0)^* \\ &- (A_1^R - jA_1^I)R_{sc}(\tau_1) - (A_1^R + jA_1^I)R_{sc}(\tau_1)^*. \end{aligned} \tag{D.11b}$$

Next, the partial derivatives of J with respect to A_0^R, A_0^I, A_1^R and A_1^I are computed and equalled to 0, as it was indicated in equation (D.9), for getting the expressions for A_0^R, A_0^I, A_1^R and A_1^I.

The derivative of J with respect to A_0^R is

$$\begin{aligned} \frac{\partial J}{\partial A_0^R} =&2A_0^R R_{cc}(0) + (A_1^R - jA_1^I)R_{cc}(\tau_1 - \tau_0) + (A_1^R + jA_1^I)R_{cc}(\tau_1 - \tau_0)^* \\ &- R_{sc}(\tau_0) - R_{sc}(\tau_0)^* = 0, \end{aligned} \tag{D.12}$$

giving the following expression for A_0^R

$$\begin{aligned} A_0^R =&\frac{1}{2R_{cc}(0)}(R_{sc}(\tau_0) + R_{sc}(\tau_0)^* - A_1^R(R_{cc}(\tau_1 - \tau_0) + R_{cc}(\tau_1 - \tau_0)^*) \\ &+ jA_1^I(R_{cc}(\tau_1 - \tau_0) - R_{cc}(\tau_1 - \tau_0)^*)). \end{aligned} \tag{D.13}$$

The derivative of J with respect to A_0^I is

$$\begin{aligned} \frac{\partial J}{\partial A_0^I} =&2A_0^I R_{cc}(0) + (jA_1^R + A_1^I)R_{cc}(\tau_1 - \tau_0) + (-jA_1^R + A_1^I)R_{cc}(\tau_1 - \tau_0)^* \\ &+ jR_{sc}(\tau_0) - jR_{sc}(\tau_0)^* = 0, \end{aligned} \tag{D.14}$$

giving the following expression for A_0^I

$$\begin{aligned} A_0^I =&\frac{j}{2R_{cc}(0)}(-R_{sc}(\tau_0) + R_{sc}(\tau_0)^* - A_1^R(R_{cc}(\tau_1 - \tau_0) - R_{cc}(\tau_1 - \tau_0)^*) \\ &+ jA_1^I(R_{cc}(\tau_1 - \tau_0) + R_{cc}(\tau_1 - \tau_0)^*)). \end{aligned} \tag{D.15}$$

The derivative of J with respect to A_1^R is

$$\begin{aligned}\frac{\partial J}{\partial A_1^R} =& 2A_1^R R_{cc}(0) + (A_0^R + jA_0^I)R_{cc}(\tau_1 - \tau_0) + (A_0^R - jA_0^I)R_{cc}(\tau_1 - \tau_0)^* \\ & - R_{sc}(\tau_1) - R_{sc}(\tau_1)^* = 0,\end{aligned} \tag{D.16}$$

giving the following expression for A_1^R

$$\begin{aligned}A_1^R =& \frac{1}{2R_{cc}(0)}(R_{sc}(\tau_1) + R_{sc}(\tau_1)^* - A_0^R(R_{cc}(\tau_1 - \tau_0) + R_{cc}(\tau_1 - \tau_0)^*) \\ & - jA_0^I(R_{cc}(\tau_1 - \tau_0) - R_{cc}(\tau_1 - \tau_0)^*)).\end{aligned} \tag{D.17}$$

The derivative of J with respect to A_1^I is

$$\begin{aligned}\frac{\partial J}{\partial A_1^I} =& 2A_1^I R_{cc}(0) + (-jA_0^R + A_0^I)R_{cc}(\tau_1 - \tau_0) + (jA_0^R + A_0^I)R_{cc}(\tau_1 - \tau_0)^* \\ & + jR_{sc}(\tau_1) - jR_{sc}(\tau_1)^* = 0,\end{aligned} \tag{D.18}$$

giving the following expression for A_1^I

$$\begin{aligned}A_1^I =& \frac{j}{2R_{cc}(0)}(-R_{sc}(\tau_1) + R_{sc}(\tau_1)^* + A_0^R(R_{cc}(\tau_1 - \tau_0) - R_{cc}(\tau_1 - \tau_0)^*) \\ & + jA_0^I(R_{cc}(\tau_1 - \tau_0) + R_{cc}(\tau_1 - \tau_0)^*)).\end{aligned} \tag{D.19}$$

The expressions (D.13), (D.15), (D.17) and (D.19) form a four system of equations with four unknowns, which can be solved.

First, equations (D.13) and (D.15) are used in equation (D.17), giving

$$\begin{aligned}A_1^R =& \frac{1}{2R_{cc}(0)}\Bigg(R_{sc}(\tau_1) + R_{sc}(\tau_1)^* \\ & - \frac{1}{2R_{cc}(0)}(R_{sc}(\tau_0) + R_{sc}(\tau_0)^* - A_1^R(R_{cc}(\tau_1 - \tau_0) + R_{cc}(\tau_1 - \tau_0)^*) \\ & + jA_1^I(R_{cc}(\tau_1 - \tau_0) - R_{cc}(\tau_1 - \tau_0)^*))(R_{cc}(\tau_1 - \tau_0) + R_{cc}(\tau_1 - \tau_0)^*) \\ & - j\frac{j}{2R_{cc}(0)}(-R_{sc}(\tau_0) + R_{sc}(\tau_0)^* - A_1^R(R_{cc}(\tau_1 - \tau_0) - R_{cc}(\tau_1 - \tau_0)^*) \\ & + jA_1^I(R_{cc}(\tau_1 - \tau_0) + R_{cc}(\tau_1 - \tau_0)^*))(R_{cc}(\tau_1 - \tau_0) - R_{cc}(\tau_1 - \tau_0)^*)\Bigg),\end{aligned} \tag{D.20a}$$

and by extending it, A_1^R becomes

$$\begin{aligned}
A_1^R = & \frac{1}{4R_{cc}(0)^2}\Bigg(2R_{cc}(0)(R_{sc}(\tau_1) + R_{sc}(\tau_1)^*) \\
& - (R_{sc}(\tau_0) + R_{sc}(\tau_0)^* - A_1^R(R_{cc}(\tau_1 - \tau_0) + R_{cc}(\tau_1 - \tau_0)^*) \\
& + jA_1^I(R_{cc}(\tau_1 - \tau_0) - R_{cc}(\tau_1 - \tau_0)^*))(R_{cc}(\tau_1 - \tau_0) + R_{cc}(\tau_1 - \tau_0)^*) \\
& + (-R_{sc}(\tau_0) + R_{sc}(\tau_0)^* - A_1^R(R_{cc}(\tau_1 - \tau_0) - R_{cc}(\tau_1 - \tau_0)^*) \\
& + jA_1^I(R_{cc}(\tau_1 - \tau_0) + R_{cc}(\tau_1 - \tau_0)^*))(R_{cc}(\tau_1 - \tau_0) - R_{cc}(\tau_1 - \tau_0)^*)\Bigg),
\end{aligned} \tag{D.20b}$$

$$\begin{aligned}
= & \frac{1}{4R_{cc}(0)^2}\Bigg(2R_{cc}(0)(R_{sc}(\tau_1) + R_{sc}(\tau_1)^*) \\
& - R_{sc}(\tau_0)(R_{cc}(\tau_1 - \tau_0) + R_{cc}(\tau_1 - \tau_0)^*) - R_{sc}(\tau_0)^*(R_{cc}(\tau_1 - \tau_0) + R_{cc}(\tau_1 - \tau_0)^*) \\
& + A_1^R(R_{cc}(\tau_1 - \tau_0) + R_{cc}(\tau_1 - \tau_0)^*)(R_{cc}(\tau_1 - \tau_0) + R_{cc}(\tau_1 - \tau_0)^*) \\
& - jA_1^I(R_{cc}(\tau_1 - \tau_0) - R_{cc}(\tau_1 - \tau_0)^*)(R_{cc}(\tau_1 - \tau_0) + R_{cc}(\tau_1 - \tau_0)^*) \\
& - R_{sc}(\tau_0)(R_{cc}(\tau_1 - \tau_0) - R_{cc}(\tau_1 - \tau_0)^*) + R_{sc}(\tau_0)^*(R_{cc}(\tau_1 - \tau_0) - R_{cc}(\tau_1 - \tau_0)^*) \\
& - A_1^R(R_{cc}(\tau_1 - \tau_0) - R_{cc}(\tau_1 - \tau_0)^*)(R_{cc}(\tau_1 - \tau_0) - R_{cc}(\tau_1 - \tau_0)^*) \\
& + jA_1^I(R_{cc}(\tau_1 - \tau_0) + R_{cc}(\tau_1 - \tau_0)^*)(R_{cc}(\tau_1 - \tau_0) - R_{cc}(\tau_1 - \tau_0)^*)\Bigg),
\end{aligned} \tag{D.20c}$$

$$\begin{aligned}
= & \frac{1}{4R_{cc}(0)^2}\Bigg(2R_{cc}(0)(R_{sc}(\tau_1) + R_{sc}(\tau_1)^*) - 2R_{cc}(\tau_1 - \tau_0)R_{sc}(\tau_0) \\
& - 2R_{cc}(\tau_1 - \tau_0)^*R_{sc}(\tau_0)^* + A_1^R(4R_{cc}(\tau_1 - \tau_0)R_{cc}(\tau_1 - \tau_0)^*)\Bigg),
\end{aligned} \tag{D.20d}$$

From the last equation is resulting that

$$A_1^R = \frac{R_{cc}(0)(R_{sc}(\tau_1) + R_{sc}(\tau_1)^*) - R_{cc}(\tau_1 - \tau_0)R_{sc}(\tau_0) - R_{cc}(\tau_1 - \tau_0)^*R_{sc}(\tau_0)^*}{2R_{cc}(0)^2 - 2R_{cc}(\tau_1 - \tau_0)R_{cc}(\tau_1 - \tau_0)^*}. \tag{D.21}$$

Next, equations (D.13) and (D.15) are used in equation (D.19), giving

$$\begin{aligned}
A_1^I = & \frac{j}{2R_{cc}(0)}\Bigg(- R_{sc}(\tau_1) + R_{sc}(\tau_1)^* \\
& + \frac{1}{2R_{cc}(0)}(R_{sc}(\tau_0) + R_{sc}(\tau_0)^* - A_1^R(R_{cc}(\tau_1 - \tau_0) + R_{cc}(\tau_1 - \tau_0)^*) \\
& + jA_1^I(R_{cc}(\tau_1 - \tau_0) - R_{cc}(\tau_1 - \tau_0)^*))(R_{cc}(\tau_1 - \tau_0) - R_{cc}(\tau_1 - \tau_0)^*) \\
& + j\frac{j}{2R_{cc}(0)}(-R_{sc}(\tau_0) + R_{sc}(\tau_0)^* - A_1^R(R_{cc}(\tau_1 - \tau_0) - R_{cc}(\tau_1 - \tau_0)^*) \\
& + jA_1^I(R_{cc}(\tau_1 - \tau_0) + R_{cc}(\tau_1 - \tau_0)^*))(R_{cc}(\tau_1 - \tau_0) + R_{cc}(\tau_1 - \tau_0)^*)\Bigg),
\end{aligned} \tag{D.22a}$$

and by extending it, A_1^I becomes

$$\begin{aligned}
A_1^I = & \frac{j}{4R_{cc}(0)^2}\Bigg(2R_{cc}(0)(-R_{sc}(\tau_1) + R_{sc}(\tau_1)^*) \\
& + (R_{sc}(\tau_0) + R_{sc}(\tau_0)^* - A_1^R(R_{cc}(\tau_1-\tau_0) + R_{cc}(\tau_1-\tau_0)^*) \\
& + jA_1^I(R_{cc}(\tau_1-\tau_0) - R_{cc}(\tau_1-\tau_0)^*))(R_{cc}(\tau_1-\tau_0) - R_{cc}(\tau_1-\tau_0)^*) \\
& - (-R_{sc}(\tau_0) + R_{sc}(\tau_0)^* - A_1^R(R_{cc}(\tau_1-\tau_0) - R_{cc}(\tau_1-\tau_0)^*) \\
& + jA_1^I(R_{cc}(\tau_1-\tau_0) + R_{cc}(\tau_1-\tau_0)^*))(R_{cc}(\tau_1-\tau_0) + R_{cc}(\tau_1-\tau_0)^*)\Bigg), \qquad \text{(D.22b)}
\end{aligned}$$

$$\begin{aligned}
= & \frac{j}{4R_{cc}(0)^2}\Bigg(2R_{cc}(0)(-R_{sc}(\tau_1) + R_{sc}(\tau_1)^*) \\
& + R_{sc}(\tau_0)(R_{cc}(\tau_1-\tau_0) - R_{cc}(\tau_1-\tau_0)^*) + R_{sc}(\tau_0)^*(R_{cc}(\tau_1-\tau_0) - R_{cc}(\tau_1-\tau_0)^*) \\
& - A_1^R(R_{cc}(\tau_1-\tau_0) + R_{cc}(\tau_1-\tau_0)^*)(R_{cc}(\tau_1-\tau_0) - R_{cc}(\tau_1-\tau_0)^*) \\
& + jA_1^I(R_{cc}(\tau_1-\tau_0) - R_{cc}(\tau_1-\tau_0)^*)(R_{cc}(\tau_1-\tau_0) - R_{cc}(\tau_1-\tau_0)^*) \\
& + R_{sc}(\tau_0)(R_{cc}(\tau_1-\tau_0) + R_{cc}(\tau_1-\tau_0)^*) - R_{sc}(\tau_0)^*(R_{cc}(\tau_1-\tau_0) + R_{cc}(\tau_1-\tau_0)^*) \\
& + A_1^R(R_{cc}(\tau_1-\tau_0) - R_{cc}(\tau_1-\tau_0)^*)(R_{cc}(\tau_1-\tau_0) + R_{cc}(\tau_1-\tau_0)^*) \\
& - jA_1^I(R_{cc}(\tau_1-\tau_0) + R_{cc}(\tau_1-\tau_0)^*)(R_{cc}(\tau_1-\tau_0) + R_{cc}(\tau_1-\tau_0)^*)\Bigg), \qquad \text{(D.22c)}
\end{aligned}$$

$$\begin{aligned}
= & \frac{j}{4R_{cc}(0)^2}\Bigg(2R_{cc}(0)(-R_{sc}(\tau_1) + R_{sc}(\tau_1)^*) + 2R_{cc}(\tau_1-\tau_0)R_{sc}(\tau_0) \\
& - 2R_{cc}(\tau_1-\tau_0)^*R_{sc}(\tau_0)^* + jA_1^I(-4R_{cc}(\tau_1-\tau_0)R_{cc}(\tau_1-\tau_0)^*)\Bigg). \qquad \text{(D.22d)}
\end{aligned}$$

From the last equation is resulting that

$$A_1^I = j\frac{R_{cc}(0)(-R_{sc}(\tau_1) + R_{sc}(\tau_1)^*) + R_{cc}(\tau_1-\tau_0)R_{sc}(\tau_0) - R_{cc}(\tau_1-\tau_0)^*R_{sc}(\tau_0)^*}{2R_{cc}(0)^2 - 2R_{cc}(\tau_1-\tau_0)R_{cc}(\tau_1-\tau_0)^*)}. \tag{D.23}$$

The equations (D.21) and (D.23) are used in equation (D.13), in order to compute the expression for A_0^R, as below

$$\begin{aligned}A_0^R = \frac{1}{2R_{cc}(0)}\Bigg(&R_{sc}(\tau_0)+R_{sc}(\tau_0)^* \\ &-\frac{R_{cc}(0)(R_{sc}(\tau_1)+R_{sc}(\tau_1)^*)-R_{cc}(\tau_1-\tau_0)R_{sc}(\tau_0)-R_{cc}(\tau_1-\tau_0)^*R_{sc}(\tau_0)^*}{2R_{cc}(0)^2-2R_{cc}(\tau_1-\tau_0)R_{cc}(\tau_1-\tau_0)^*} \\ &(R_{cc}(\tau_1-\tau_0)+R_{cc}(\tau_1-\tau_0)^*) \\ &+jj\frac{R_{cc}(0)(-R_{sc}(\tau_1)+R_{sc}(\tau_1)^*)+R_{cc}(\tau_1-\tau_0)R_{sc}(\tau_0)-R_{cc}(\tau_1-\tau_0)^*R_{sc}(\tau_0)^*}{2R_{cc}(0)^2-2R_{cc}(\tau_1-\tau_0)R_{cc}(\tau_1-\tau_0)^*} \\ &(R_{cc}(\tau_1-\tau_0)-R_{cc}(\tau_1-\tau_0)^*)\Bigg),\end{aligned} \tag{D.24a}$$

and by extending it, A_0^R becomes

$$\begin{aligned}A_0^R =&\frac{1}{2R_{cc}(0)(2R_{cc}(0)^2-2R_{cc}(\tau_1-\tau_0)R_{cc}(\tau_1-\tau_0)^*)} \\ &\Bigg((R_{sc}(\tau_0)+R_{sc}(\tau_0)^*)(2R_{cc}(0)^2-2R_{cc}(\tau_1-\tau_0)R_{cc}(\tau_1-\tau_0)^*) \\ &-(R_{cc}(\tau_1-\tau_0)+R_{cc}(\tau_1-\tau_0)^*)R_{cc}(0)(R_{sc}(\tau_1)+R_{sc}(\tau_1)^*) \\ &+(R_{cc}(\tau_1-\tau_0)+R_{cc}(\tau_1-\tau_0)^*)(R_{cc}(\tau_1-\tau_0)R_{sc}(\tau_0)+R_{cc}(\tau_1-\tau_0)^*R_{sc}(\tau_0)^*) \\ &-(R_{cc}(\tau_1-\tau_0)-R_{cc}(\tau_1-\tau_0)^*)R_{cc}(0)(-R_{sc}(\tau_1)+R_{sc}(\tau_1)^*) \\ &-(R_{cc}(\tau_1-\tau_0)-R_{cc}(\tau_1-\tau_0)^*)(R_{cc}(\tau_1-\tau_0)R_{sc}(\tau_0)-R_{cc}(\tau_1-\tau_0)^*R_{sc}(\tau_0)^*)\Bigg),\end{aligned} \tag{D.24b}$$

$$\begin{aligned}=&\frac{1}{2R_{cc}(0)(2R_{cc}(0)^2-2R_{cc}(\tau_1-\tau_0)R_{cc}(\tau_1-\tau_0)^*)} \\ &\Bigg((R_{sc}(\tau_0)+R_{sc}(\tau_0)^*)2R_{cc}(0)^2 \\ &-2R_{cc}(0)(R_{sc}(\tau_1)^*R_{cc}(\tau_1-\tau_0)+R_{sc}(\tau_1)R_{cc}(\tau_1-\tau_0)^*)\Bigg),\end{aligned} \tag{D.24c}$$

and the final expression is

$$A_0^R =\frac{(R_{sc}(\tau_0)+R_{sc}(\tau_0)^*)R_{cc}(0)-(R_{sc}(\tau_1)^*R_{cc}(\tau_1-\tau_0)+R_{sc}(\tau_1)R_{cc}(\tau_1-\tau_0)^*)}{2R_{cc}(0)^2-2R_{cc}(\tau_1-\tau_0)R_{cc}(\tau_1-\tau_0)^*}. \tag{D.24d}$$

The final step is to use the equations (D.21) and (D.23) in equation (D.15), in order to compute the expression for A_0^I, as below

$$\begin{aligned}A_0^I =&\frac{j}{2R_{cc}(0)}\Bigg(- R_{sc}(\tau_0) + R_{sc}(\tau_0)^* \\ &- \frac{R_{cc}(0)(R_{sc}(\tau_1) + R_{sc}(\tau_1)^*) - R_{cc}(\tau_1-\tau_0)R_{sc}(\tau_0) - R_{cc}(\tau_1-\tau_0)^*R_{sc}(\tau_0)^*}{2R_{cc}(0)^2 - 2R_{cc}(\tau_1-\tau_0)R_{cc}(\tau_1-\tau_0)^*} \\ &(R_{cc}(\tau_1-\tau_0) - R_{cc}(\tau_1-\tau_0)^*) \\ &+ jj\frac{R_{cc}(0)(-R_{sc}(\tau_1) + R_{sc}(\tau_1)^*) + R_{cc}(\tau_1-\tau_0)R_{sc}(\tau_0) - R_{cc}(\tau_1-\tau_0)^*R_{sc}(\tau_0)^*}{2R_{cc}(0)^2 - 2R_{cc}(\tau_1-\tau_0)R_{cc}(\tau_1-\tau_0)^*)} \\ &(R_{cc}(\tau_1-\tau_0) + R_{cc}(\tau_1-\tau_0)^*)\Bigg),\end{aligned} \tag{D.25a}$$

and by extending it, A_0^I becomes

$$\begin{aligned}A_0^I =&\frac{j}{2R_{cc}(0)(2R_{cc}(0)^2 - 2R_{cc}(\tau_1-\tau_0)R_{cc}(\tau_1-\tau_0)^*)} \\ &\Bigg((-R_{sc}(\tau_0) + R_{sc}(\tau_0)^*)(2R_{cc}(0)^2 - 2R_{cc}(\tau_1-\tau_0)R_{cc}(\tau_1-\tau_0)^*) \\ &- R_{cc}(0)(R_{sc}(\tau_1) + R_{sc}(\tau_1)^*)(R_{cc}(\tau_1-\tau_0) - R_{cc}(\tau_1-\tau_0)^*) \\ &+ (R_{cc}(\tau_1-\tau_0)R_{sc}(\tau_0) + R_{cc}(\tau_1-\tau_0)^*R_{sc}(\tau_0)^*)(R_{cc}(\tau_1-\tau_0) - R_{cc}(\tau_1-\tau_0)^*) \\ &- R_{cc}(0)(-R_{sc}(\tau_1) + R_{sc}(\tau_1)^*)(R_{cc}(\tau_1-\tau_0) + R_{cc}(\tau_1-\tau_0)^*) \\ &- (R_{cc}(\tau_1-\tau_0)R_{sc}(\tau_0) - R_{cc}(\tau_1-\tau_0)^*R_{sc}(\tau_0)^*)(R_{cc}(\tau_1-\tau_0) + R_{cc}(\tau_1-\tau_0)^*)\Bigg),\end{aligned} \tag{D.25b}$$

$$\begin{aligned}&= \frac{j}{2R_{cc}(0)(2R_{cc}(0)^2 - 2R_{cc}(\tau_1-\tau_0)R_{cc}(\tau_1-\tau_0)^*)} \\ &\Bigg(2(-R_{sc}(\tau_0) + R_{sc}(\tau_0)^*)R_{cc}(0)^2 + 2R_{cc}(0)R_{sc}(\tau_1)R_{cc}(\tau_1-\tau_0)^* \\ &- 2R_{cc}(0)R_{sc}(\tau_1)^*R_{cc}(\tau_1-\tau_0)\Bigg),\end{aligned} \tag{D.25c}$$

and the final expression is

$$A_0^I =j\frac{(-R_{sc}(\tau_0) + R_{sc}(\tau_0)^*)R_{cc}(0) + R_{sc}(\tau_1)R_{cc}(\tau_1-\tau_0)^* - R_{sc}(\tau_1)^*R_{cc}(\tau_1-\tau_0)}{2R_{cc}(0)^2 - 2R_{cc}(\tau_1-\tau_0)R_{cc}(\tau_1-\tau_0)^*}. \tag{D.25d}$$

Appendix E

Additional simulation figures

E.1 Grid delays, LOS and 1 MP, No Noise

Overestimation of the number of multipaths No constraint on the amplitudes

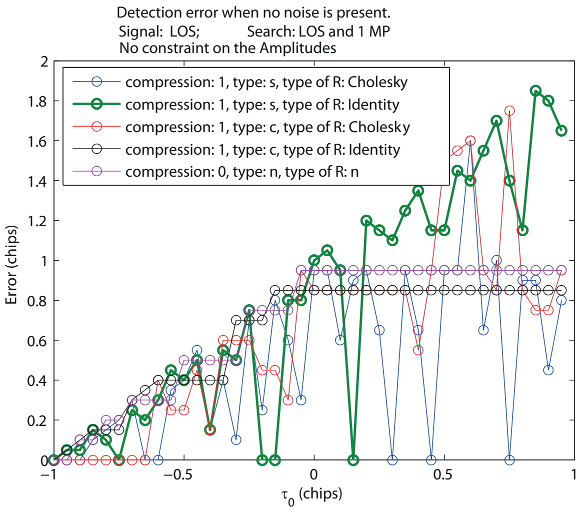
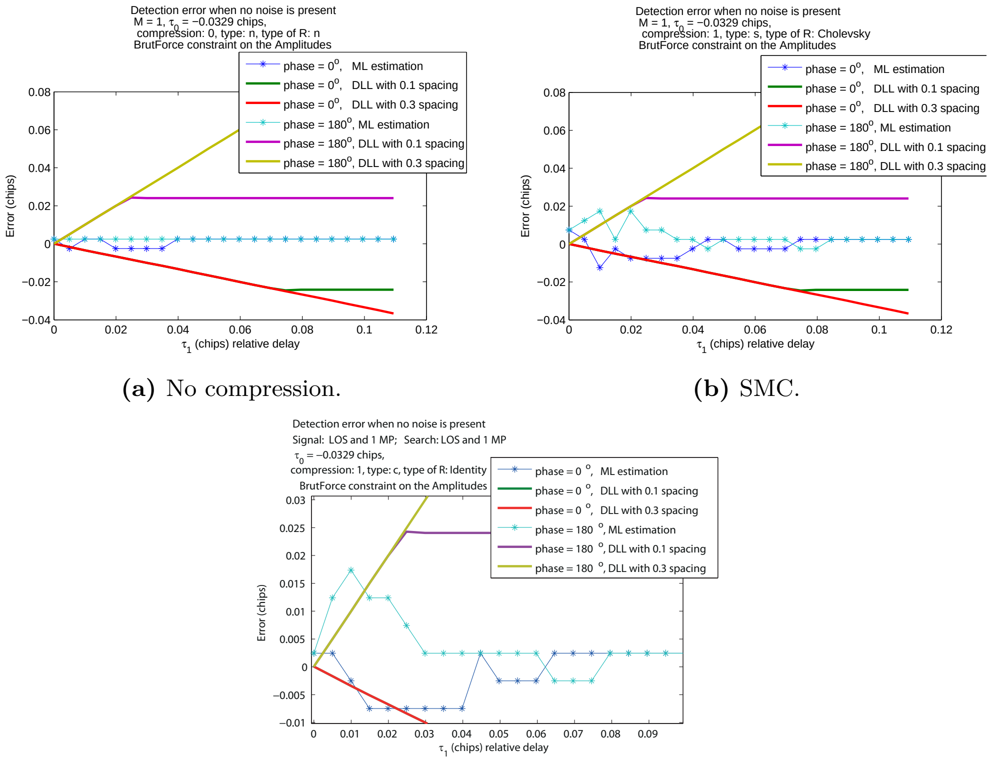


Figure E.1: Error between the true delay and the detected delay. The signal has just the line of sight, but the algorithm searches for line of sight plus one multipath. No amplitude constraints are imposed.

E.2 Continuous delays, LOS and 1 MP, No Noise

Accurate number of multipaths Uncompressed and compressed signal



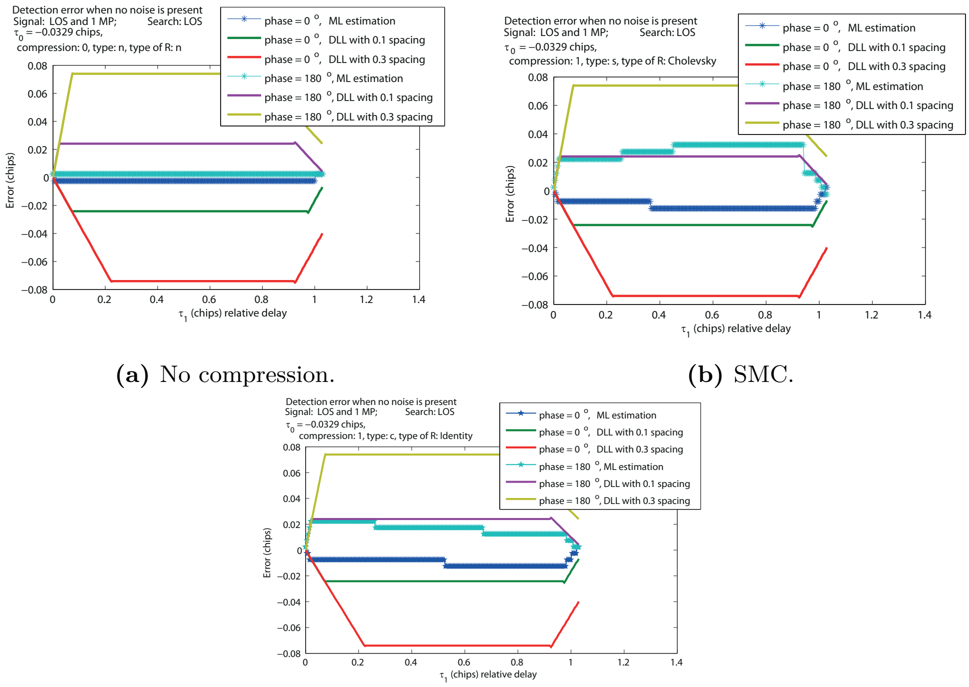
(a) No compression.

(b) SMC.

(c) CMC.

Figure E.2: Error between the true delay and the detected delay. The signal is composed by the line of sight and one multipath and the algorithm is searching for them. The channel is not affected by noise. Comparison between different compression methods.

Underestimation of the number of multipaths Uncompressed and compressed signal



(a) No compression.

(b) SMC.

(c) CMC.

Figure E.3: Error between the true delay and the detected delay. The signal is composed by the LOS and one MP and but the algorithm is searching just for LOS. The channel is not affected by noise. Comparison between different compression methods.