Developing a GNSS data analysis tool to characterize the faulty event called Code-carrier incoherence

Guillermo Centelles Hidalgo

Thesis to obtain the Master of Science Degree in

Aerospace Engineering

Supervisors: Prof. Agostinho Rui Alves da Fonseca
Dr. Pedro Tavares Moura e Silva

Examination Committee
Chairperson: Prof. Filipe Szolnoky Ramos Pinto Cunha
Supervisor: Prof. Agostinho Rui Alves da Fonseca
Member of the Committee: Prof. João Manuel Gonçalves de Sousa Oliveira

November 2017
Acknowledgments

First of all I want to thank the IST for let me introduce into the world of programming and navigation by satellite. I have been fascinated by the way they coordinate practical and theoretical teaching and from now on it is a university that I will always recommend. Thanks to my supervisor, Agostinho Fonseca for having supported me to make the Master Thesis that I liked the most unconditionally. Also, I want to thank Deimos Engenharia and Pedro Tavares, who have given me the opportunity to do the Master Thesis with them, I have enjoyed an experience of a real and current project. I want to thank all my friends who have supported me and encouraged me in the most difficult moments, when no results came out. To de Becerra Sanchez’s family, Isa and my father for making me feel that I always have their support.

Especially, and I left it for the end, there are two people who play an important role in my life to whom I would thank a thousand times. To Irene because the affection and strength she gives me can not be measured with words. She has made me never feel alone in the realization of this project. Thanks my life. And my brother because he gets me to never abandon what I propose to myself, I would never have become as I am if it were not for his dedication, his way of doing things and his desire to see me grow as a person. Thank you both for making me feel that this work is the result of a team that all three of us form.
Abstract

The accuracy, availability and integrity of the Galileo system are of significant concern because of the intended benefits of this system in many applications in a near future, when the system will be fully operative. Thus it is of critical importance that every satellite works and any potential error is minimized and bounded. This thesis analyzes the problem which a type of satellite failure, named code-carrier incoherence, generates and discusses how to overcome it with coherent multipath tracking, using the Galileo E1 frequency band as main frequency and the Galileo E5 frequency band as a secondary frequency. Then, variations with respect to the nominal Multipath receiver environment are detected and afterwards, data is combined to reduce the effect of measurements errors. For that, the antenna calibration has been performed, where the multipath mean computed as a function of elevation angle for a period of days. This work shows a way to effectively use Multipath and its standard deviation that avoids cycle slips and obtains well determined arcs without phase ambiguities as a result. A tool is developed in C++ to compute a code-carrier coherence (CCC) value and a Gaussian error propagation of CCC. Finally, a CCC threshold value is established and when the CCC value overpasses it, the code carrier incoherence is detected. Details about the current and future status of Galileo, its data use in receiver stations and in the GPS Tool Kit, the multipath estimation algorithm for Galileo measurements are also noted to provide context for future improvements of this developed tool.

Keywords: Code-Carrier Coherence (CCC), Multipath, Feared event
Resumo

A precisão, disponibilidade e integridade do sistema Galileo são motivo de preocupação, devido aos benefícios pretendidos deste sistema em muitas aplicações num futuro próximo, quando o sistema estiver totalmente operacional. Assim, é de importância crítica que cada satélite funcione e que qualquer erro potencial seja minimizado e delimitado. Esta tese analisa o problema que um tipo de falha de satélite, nomeadamente a incoerência do portador de código, gera, e discute como superá-lo com um localizador de múltiplas trajectórias consistente, usando a banda de frequência Galileo E1 como frequência principal e a banda de frequência Galileo E5 como frequência secundária. Em seguida, são detectadas variações em relação ao ambiente nominal do receptor de múltiplas trajectórias e, posteriormente, os dados são combinados para reduzir o efeito de erros de medição. Para isso, a calibração da antena foi realizada, onde a média de trajectórias é calculada como uma função do ângulo de elevação por um período de dias. Este trabalho mostra uma maneira de usar efetivamente as trajectórias e o seu desvio padrão que evita desvios de ciclo e obtém arcos bem determinados sem ambiguidades de fase como resultado. Uma ferramenta é desenvolvida em C++ para calcular um valor de coerência do portador de código (CCC) e uma propagação de erro gaussiana de CCC. Finalmente, um valor de limiar de CCC é estabelecido e, quando o valor CCC é ultrapassado, a incoerência do portador de código é detectada. Detalhes sobre o status atual e futuro do Galileo, o uso dos seus dados em estações receptoras e no conjunto de instrumentos GPS, o algoritmo de estimativa de trajectórias para medições Galileo são também assinalados para fornecer contexto para futuras melhorias desta ferramenta.

Palavras-Chave: Code-Carrier Coherence (CCC), Multipath, Evento temido
# Contents

List of Tables ................................. xvi

List of Figures ................................. xxi

1 Introduction ................................. 1
   1.1 Accuracy, Integrity, Continuity and Availability ....................... 1
   1.1.1 Accuracy ........................................ 1
   1.1.2 Availability ...................................... 2
   1.1.3 Continuity ....................................... 2
   1.1.4 Integrity ......................................... 2
   1.2 Feared events ..................................... 2
   1.2.1 Multipath ......................................... 3
   1.2.2 Code-Carrier incoherence ................................. 4
   1.3 Thesis layout .................................... 4

2 State of art ................................. 5
   2.1 GNSS and Galileo nowadays ........................................ 5
   2.1.1 Galileo status ..................................... 6
   2.1.2 Galileo, improvements .................................. 8
   2.2 Current Galileo performance .................................. 9
   2.3 Code-Carrier Coherence .................................. 9

3 Material and Methods ..................... 11
   3.1 Galileo ............................................. 11
   3.1.1 Frequency bands chosen .................................. 12
   3.2 GPSTK ............................................... 13
   3.2.1 C++ ............................................... 13
   3.2.2 Doxygen .......................................... 13
   3.3 International GNSS Service (IGS) ................................ 14
   3.3.1 Receiver Independent Exchange Format (RINEX) .................... 14
   3.3.2 Standard Product 3 .................................. 19
   3.4 Receiver stations .................................... 21
   3.5 Multipath ........................................... 24
   3.5.1 Linear combination of observables ................................ 24
   3.5.2 Assessing Multipath .................................. 25
   3.5.3 Multipath Template Technique ............................... 25
   3.5.4 Antenna Calibration .................................. 25
# Table of Contents

3.6 Code carrier coherence .................................................. 26  
3.6.1 Gaussian error propagation for CCC .............................. 27  
3.7 Programming code explanation ....................................... 28  
3.7.1 Requirements ............................................................ 28  
3.7.2 GPSTK limitations ....................................................... 28  
3.7.3 Cycle slip ................................................................. 28  
3.7.4 Input files ................................................................. 29  
3.7.5 Computing Antenna Calibration ................................. 31  
3.7.6 Computing CCC ......................................................... 32  
3.8 CCC threshold .............................................................. 33  

4 Data Results  
4.1 Suitable RINEX Input .................................................... 36  
4.2 Raw Multipath .............................................................. 37  
4.3 Antenna Calibration ....................................................... 37  
4.3.1 Discrete Multipath ..................................................... 38  
4.3.2 Derivative Multipath with Elevation ............................ 40  
4.3.3 Multipath of Antenna Calibration ............................... 42  
4.3.4 Standard Deviation ................................................... 44  
4.3.5 Different Bands ......................................................... 45  
4.4 Code-Carrier Coherence ................................................ 46  
4.4.1 Leveled Multipath ..................................................... 46  
4.4.2 Code-Carrier Coherence ............................................ 49  
4.4.3 Standard Deviation for Code-Carrier Coherence ............. 52  
4.5 Code-Carrier Coherence Threshold ................................ 55  

5 Discussion  
5.1 Galileo Problems .......................................................... 57  
5.1.1 Number of Satellites ................................................ 57  
5.2 Multipath as a mean of computation ................................ 57  
5.2.1 Standard Deviation ................................................... 58  
5.3 Day of the year ............................................................... 58  
5.4 Choosing a threshold for Code-Carrier Coherence .............. 59  

6 Conclusion ................................................................. 61  

7 Future Works ............................................................... 63  

Bibliography ................................................................. 64  

A R Programming language .................................................. 67  
A.1 RStudio ................................................................. 67  
A.2 Plotting Results .......................................................... 68  
A.3 R Programming code .................................................... 68  

B C++ Programming code ..................................................... 71  
B.1 Antenna Calibration part ............................................... 71  
B.2 Test Part ................................................................. 71  

viii
List of Acronyms

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARL:UT</td>
<td>Applied Research Laboratories at the University of Texas</td>
</tr>
<tr>
<td>BDS</td>
<td>BeiDou Navigation Satellite System</td>
</tr>
<tr>
<td>C/A</td>
<td>Coarse Acquisition</td>
</tr>
<tr>
<td>CCC</td>
<td>Code-Carrier Coherence</td>
</tr>
<tr>
<td>CLS</td>
<td>Collecte Localised Satellites</td>
</tr>
<tr>
<td>CNES</td>
<td>Centre National d’Etudes Spatiales</td>
</tr>
<tr>
<td>CORS</td>
<td>Continuously Operating Reference Station</td>
</tr>
<tr>
<td>CRC</td>
<td>Cycle Redundancy Check</td>
</tr>
<tr>
<td>CS</td>
<td>Commercial Service</td>
</tr>
<tr>
<td>ESA</td>
<td>European Space Agency</td>
</tr>
<tr>
<td>ESTEC</td>
<td>European Space Research and Technology Centre</td>
</tr>
<tr>
<td>EUREF</td>
<td>European Terrestrial Reference System</td>
</tr>
<tr>
<td>FTP</td>
<td>File Transfer Protocol</td>
</tr>
<tr>
<td>GGTO</td>
<td>GPS to Galileo time offset</td>
</tr>
<tr>
<td>GLONASS</td>
<td>Globalnaya navigatsionnaya Sputnikovaya Sistema</td>
</tr>
<tr>
<td>GNSS</td>
<td>Global Navigation Satellite System</td>
</tr>
<tr>
<td>GNSS</td>
<td>Global Navigation Satellite System</td>
</tr>
<tr>
<td>GPS</td>
<td>Global Positioning System</td>
</tr>
<tr>
<td>GPSTK</td>
<td>GPS ToolKit</td>
</tr>
<tr>
<td>IDE</td>
<td>Integrated Development Environment</td>
</tr>
<tr>
<td>IEC</td>
<td>Ionospheric Electron Content</td>
</tr>
<tr>
<td>IGS</td>
<td>International GNSS Service</td>
</tr>
<tr>
<td>IRNSS</td>
<td>Indian Regional Navigation Satellite System</td>
</tr>
<tr>
<td>IST</td>
<td>Instituto Superior Técnico</td>
</tr>
<tr>
<td>IST</td>
<td>Instituto Superior Técnico</td>
</tr>
<tr>
<td>LOS</td>
<td>Line of Sight</td>
</tr>
<tr>
<td>MP</td>
<td>Multipath</td>
</tr>
<tr>
<td>MPL</td>
<td>Minimum Performance Levels</td>
</tr>
<tr>
<td>NASA</td>
<td>National Aeronautics and Space Administration</td>
</tr>
<tr>
<td>NMA</td>
<td>Navigation Message Authentication</td>
</tr>
<tr>
<td>OS</td>
<td>Open Service</td>
</tr>
<tr>
<td>OS-SDD</td>
<td>Open Service-Service Definition Document</td>
</tr>
<tr>
<td>PR</td>
<td>Pseudorange</td>
</tr>
<tr>
<td>PRN</td>
<td>Pseudorandom noise</td>
</tr>
<tr>
<td>PRS</td>
<td>Public Regulated Service</td>
</tr>
<tr>
<td>QZSS</td>
<td>Quasi-Zenith Satellite System</td>
</tr>
<tr>
<td>Acronym</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>------------------------------------------------</td>
</tr>
<tr>
<td>RINEX</td>
<td>Receiver Independent Exchange Format</td>
</tr>
<tr>
<td>SAR</td>
<td>Search and Rescue</td>
</tr>
<tr>
<td>SBAS</td>
<td>Satellite Based Augmentation System</td>
</tr>
<tr>
<td>SoL</td>
<td>Safety of Life</td>
</tr>
<tr>
<td>SP3</td>
<td>Standard Product 3</td>
</tr>
<tr>
<td>UNAVCO</td>
<td>University-Governed Consortium</td>
</tr>
<tr>
<td>WAAS</td>
<td>Wide Area Augmentation System</td>
</tr>
</tbody>
</table>
List of Symbols

\( A \)  \hspace{1em} \text{Constant} = 40.3^3 s^{-2}
\( A_i \)  \hspace{1em} \text{Ambiguity term}
\( \text{Ant}(\phi) \)  \hspace{1em} \text{Antenna Calibration for elevation angle} \ \phi
\( c \)  \hspace{1em} \text{Light velocity}
\( CCC^i \)  \hspace{1em} \text{Code Carrier Coherence for satellite} \ i
\( dT \)  \hspace{1em} \text{Real-time-derived receiver clock offset}
\( e_{MP} \)  \hspace{1em} \text{Code Multipath and noise}
\( f_{\text{carrier}} \)  \hspace{1em} \text{Carrier phase frequency}
\( f_{\text{code}} \)  \hspace{1em} \text{Pseudorandom code frequency}
\( freq \)  \hspace{1em} \text{Frequency}
\( I_i \)  \hspace{1em} \text{Term due to ionospheric delay for frequency} \ i
\( K \)  \hspace{1em} \text{Integration constant}
\( MP \)  \hspace{1em} \text{Multipath}
\( MP_{cx} \)  \hspace{1em} \text{Multipath for a type of data called} \ cx
\( MP^i_j \)  \hspace{1em} \text{Biased Multipath for satellite} \ i \ \text{and receiver station} \ j
\( N \)  \hspace{1em} \text{Number of measurements}
\( N_{bins} \)  \hspace{1em} \text{Number of bins}
\( CCC \)  \hspace{1em} \text{Mean CCC}
\( MP \)  \hspace{1em} \text{Mean Multipath}
\( P_i \)  \hspace{1em} \text{Code measurement for frequency} \ i
\( PR \)  \hspace{1em} \text{Pseudorange}
\( s \)  \hspace{1em} \text{Term which includes geometric range and the difference of clock offsets}
\( \sigma \)  \hspace{1em} \text{Standard Deviation}
List of Greek Symbols

\( \alpha \)  Represents \( I_i / I_j \)
\( \delta \phi \)  Elevation angle increment
\( \epsilon_{MP} \)  Carrier phase Multipath and Noise
\( \mu \)  Mean
\( \mu_j^i \)  Leveled Multipath for satellite \( i \) and receiver station \( j \)
\( \phi \)  Elevation angle
\( \Phi_i \)  Carrier phase for frequency \( i \)
\( \sigma_{ccc} \)  CCC Standard Deviation
\( \sigma_{ccc}^i \)  CCC Standard Deviation for satellite \( i \)
\( \sigma_i^l \)  Standard Deviation of leveled Multipath from Antenna Calibration
\( \Theta_{ccc} \)  CCC Threshold
List of Tables

3.1 IGS formats [1] ................................................................. 14
3.2 Receiver station positions [2] ............................................. 22
4.1 Conversion from continuous Multipath to discrete Multipath function ........................................... 37
List of Figures

1.1 Concept of integrity [3] ................................................................. 2
1.3 Multipath propagation in sea ....................................................... 3

2.1 The four systems belonging to GNSS and an example of their satellites: GPS, GLONASS, Galileo and BDS. .................................................. 5
2.2 Example of GNSS app called GPS Test. As it shows, there are many satellites in view but none of them are Galileo (GPS are the circle shape ones, GLONASS are the triangle shape and Galileo are the diamond shape). ........................................... 7

3.1 Galileo Frequency Spectrum [5] .................................................. 11
3.2 GPSTK: The tool kit which helps working on GNSS [6] .................... 13
3.3 RINEX 2 observation data keys [7] .............................................. 16
3.4 RINEX 3 observation data keys [8] .............................................. 17
3.5 Example of RINEX file which has Galileo keys in the data type list in the header but that data does not appear in the body. ............................................. 18
3.6 Example of RINEX file which has Galileo keys in the data type list in the header and that data actually appears in the body. ................................. 19
3.7 Example of SP3 file which has Galileo satellite keys in the satellite list in the header and that data actually appears in the body. ................... 20
3.8 Map of the stations used in this project [2] ..................................... 22
3.9 Emplacement of the Receiver Stations used in this project (ISLK, OZST, P558, P588 and CAT3). [2] ...................................................... 23
3.10 RINEX files version 3 uploaded and ready for being converted ........ 30
3.11 RINEX file version 3 loaded. Here, one can choose different options. It is chosen to get as output a RINEX file version 2.11, with the name structure shown and only Galileo data ................................. 30
3.12 Example of how to bin elevation ............................................... 32

4.1 RINEX file 2.11 after conversion. Named OZST092A00.17O ............. 35
4.2 Continuous Multipath function of elevation .................................. 36
4.3 Discrete Multipath function of elevation ...................................... 38
4.4 Standard deviation of the continuous Multipath versus the discrete Multipath. It is function of Elevation. ........................................... 39
4.5 Derivative of discrete Multipath with elevation of all the satellites. It is a function of Elevation. .................................................. 40
4.6 Mean of the derivative of discrete Multipath with elevation of all the satellites. The result is the derivative of the Antenna Calibration with Elevation. .......... 41

1https://play.google.com/store/apps/
4.7 Standard deviation of the derivative Multipath with elevation of all the satellites from their mean.

It is a function of Elevation. ................................................................. 42

4.8 Multipath computed by Antenna Calibration from one to seven days, function of elevation. . 43

4.9 Multipath computed by Antenna Calibration for every station for seven days and for the band E1E5, function of elevation. ................................................................. 44

4.10 Standard Deviation computed by Antenna Calibration for every station for seven days and for the band E1E5, function of elevation. ......................................................... 45

4.11 Multipath computed by Antenna Calibration for a different combinations of bands, function of elevation. ................................................................. 46

4.12 Leveled Multipath computed function of time for the day 92 of 2017. ................................. 47

4.13 Leveled Multipath computed function of time for the day 92 of 2017. ................................. 47

4.14 Leveled Multipath computed function of time for the day 106 of 2017. ............................... 48

4.15 Leveled Multipath computed function of time for the day 151 of 2017. ............................... 48

4.16 Code carrier coherence for the day 92 for the satellite 30. ................................................. 49

4.17 Code carrier coherence for the day 92 for every Galileo satellite. ....................................... 50

4.18 Code carrier coherence for the day 106. .............................................................................. 51

4.19 Code carrier coherence for the day 151. .............................................................................. 51

4.20 Standard deviation of CCC for day 92, for the satellite 30. ................................................ 52

4.21 Standard deviation of CCC for day 92. .............................................................................. 53

4.22 Standard deviation of CCC for day 106. .............................................................................. 54

4.23 Standard deviation of CCC for day 151. .............................................................................. 54

5.1 Days used to compute the CCC test ....................................................................................... 58

A.1 Logotype of RStudio ............................................................................................................ 67
Nowadays, most of the people over the world have the possibility to know their position in real time on Earth [9]. That is possible by the Global Navigation Satellites Systems (GNSS), a satellite navigation system with global coverage. Global Navigation is one of the most promising applications that satellites have been used for, and presently, several navigation systems are available:

- Global Positioning System (GPS) provided by the United States of America;
- Galileo by the European Union;
- GLONASS by the Russian Federation;
- BeiDou Navigation Satellite System (BDS) by China.

They compose the list of the uniques global navigation systems. One can check the status of GPS\(^1\) with 32 satellites, the status of Galileo\(^2\), with 18 satellites, the status of GLONASS\(^3\), with 24 satellites, and the status of BDS\(^4\), with 16 satellites, and getting an overview of how many of them are actually operating or what kind of failure or restriction they have. There are some local navigation systems as QZSS by Japan and IRNSS by India.

### 1.1 Accuracy, Integrity, Continuity and Availability

All of those satellite systems mentioned above are used to provide a good navigation service to users. The combination of every constellation creates a complex system which can provide a better positioning accuracy, integrity, continuity and availability. It is important to have these terms clarified to have a good understanding of the researching efforts in navigation through different approaches.

#### 1.1.1 Accuracy

The accuracy of an estimated or measured position at a given time of a user is the degree of conformance of that position with the true position, speed and time of the user. Accuracy is a statistical measure of the error of the position. The navigation system accuracy has no meaning unless it includes a statement of the uncertainty in the position that applies.

\(^1\)https://www.navcen.uscg.gov/?do=constellationStatus
\(^2\)https://www.gsa.europa.eu/european-gnss/galileo/system-status
\(^3\)https://www.glonass-iac.ru/en/GLONASS/
\(^4\)https://igscb.jpl.nasa.gov/projects/igscb/Status_BDS.htm
In terms of orthogonal axes, the 95 percent confidence level is used. Vertical accuracy and horizontal accuracy are specified in one-dimensional terms (2 sigma), 95 percent confidence level. [10]

1.1.2 Availability

The availability in satellite navigation systems is the time percentage which the services of the system can be accessible by the user. Availability is a measurement of the capacity of the system to provide a service in a specific position [3]. Signal availability is the time percentage that transmitted navigation signals from external sources are feasible for the user to use them. It is a function of both the environment features and also the technical capabilities of the transmitter characteristics. [10]

1.1.3 Continuity

The system continuity is the ability of the total system to perform its function without interruption during a specific operation. Deeper, continuity is the probability that the system performance is maintained for the duration of a phase of operation, presuming that the system was available at the beginning of that step of the total operation. [10]

1.1.4 Integrity

Integrity is the measure of the trust that can be placed in the correctness of the information given by a navigation system. It includes the system ability to provide timely warnings when the system should not be used for navigation to users. [10]

![Figure 1.1: Concept of integrity](image)

Integrity exists if it is detected when the measured position leaves the bounds (worst case situation). See Figure 1.1. This timeliness also includes protecting users against satellite faults whenever they occur. Such faults are called feared events, and to adequately protect the user they must be well characterized.

1.2 Feared events

In the navigation field there are many different sources of errors as satellite geometry, satellite orbits, multipath effect, atmospheric effects, clock inaccuracies and rounding errors. Some of them can alter the
accuracy, availability, continuity or even the integrity. These feared events have been studied in static receivers to create models to avoid them in future events. They are cycle slips, multipath and code-carrier incoherence. Although cycle slips is an interesting feared events to study because they can appear in almost every current GNSS receiver, they are not studied deeper in this project. In the case of the cycle slip, one should know that it is a discontinuity in a receiver's continuous phase lock on a satellite's signal. From now on, this project will focus in multipath and mainly in code-carrier incoherence. [11]

1.2.1 Multipath

Multipath interference occurs when the user device receives reflected signals in addition to the direct line of sight (LOS) signal. These interference signals are generally reflected from the ground, buildings like in Figure 1.2 or trees in terrestrial navigation, while signal reflections from the host-vehicle body are more common in airborne and marine applications. [12]

![Figure 1.2: Principle of multipath signal in cities](image)

![Figure 1.3: Multipath propagation in sea](image)

Two kinds of multipath exist: specular multipath arising from discrete, coherent reflections from smooth surfaces such as standing water, and diffuse multipath arising from diffuse scatterers and sources of diffraction. (The visible glint of sunlight off a choppy sea is an example of diffuse multipath like could happen in Figure 1.3 from 5)

Multipath signals are generally considered undesirable in the GNSS field because they destroy the correlation function shape used for time delay estimation, but can be useful in some cases e.g. for acquisition. Although some wireless communications techniques exploit multipath to provide signal diversity, the key point in GNSS is to efficiently mitigate the multipath effect because only the LOS signal is used for getting the satellite-receiver transit time offset for positioning. [12]

---

1.2.2 Code-Carrier incoherence

For a better understanding of what this faulty event is, it is preferable to explain first, what the code-carrier coherence is. The code-carrier coherence shows the coherence between the code phase and the carrier phase and has a significant role in some applications as carrier phase ranging, carrier phase smoothed pseudorange, etc. For satellite navigation systems, guaranteed code-carrier phase coherence of the satellite transmitted signals is not only important for its ranging function, but also one of the main factors which affects the positioning accuracy of satellite navigation signals. Signal coherence is the relationship between code phase and carrier phase, and phase coherence is the relationship between the phases, i.e., constant phases or different phases. One example of code-carrier coherence boundary is the offset between short-term (less than 10s) code frequency and carrier frequency that has been defined in the U.S. for the WAAS system signal code-carrier phase coherence. \[ \begin{align*}
\frac{f_{\text{code}}}{1.023\text{MHz}} - \frac{f_{\text{carrier}}}{1575.42\text{MHz}} < 5 \times 10^{-11}
\end{align*} \] (1.1)

Where the \( f_{\text{code}} \) is the pseudorandom code frequency and the \( f_{\text{carrier}} \) is the phase carrier frequency. Code carrier incoherence introduces an additional time varying bias between the code and carrier observables, which will degrade the positioning accuracy when using code and phase data together.

The main factors causing the failure in the code-carrier coherence are:

1. Failure in the satellite. Incoherence between the code emitted by the satellite and the carrier;
2. Ionosphere or propagation path error;
3. Irregularities of the code loop or carrier loop inside the receiver and other factors.

Accordingly, several receivers are used to observe at the same time, therefore as to eliminate the measurement errors caused by the irregularities of the internal code loop or the carrier loop of the receivers through mutual calibration. Finally, the code-carrier incoherence is mainly caused by the delay jitter of satellite signals. [13]

1.3 Thesis layout

The aim of this project is to detect where the feared event of the code-carrier incoherence appears in Galileo satellites. Up to arrive to that point, some steps have been done first, as avoiding cycle slips, computing multipath to track coherence and also variations with respect to the nominal MP receiver environment as a function of the elevation angle. Then the data is combined to reduce the error effects using different receiver stations.

This dissertation will present the current situation of Galileo and its performance in the State of art section, 2, then the methods and materials used are explained in Section 3, the results obtained for intermediate procedures and final plots are shown in Section 4, the discussion where the results are compared with papers and other works in Section 5, the conclusion of achievements and a summary of final results in Section 6 and at the end the future works in Section 7.
Chapter 2

State of art

In this section, the current situation of Galileo and its position in the GNSS world will be explained as its performance. There is also a brief introduction of what is currently known of the code-carrier coherence and its use in Galileo system.

2.1 GNSS and Galileo nowadays

At the moment, GPS and GLONASS are the most extensive and accessible positioning system. They are the only two fully operative systems in 2017. GPS, which is global operative since 1978, and GLONASS achieved Russia Global coverage up to 2007 when it had enough available satellites [14]. Until now, GNSS users have had to depend on non-civilian American GPS or Russian GLONASS signals. Galileo, the Europe’s Global Satellite Navigation System, is the reliable alternative that, unlike GLONASS and GPS, remains under civilian control.

Figure 2.1: The four systems belonging to GNSS and an example of their satellites: GPS, GLONASS, Galileo and BDS.
In the Figure 2.1 one can find the symbol and an example of satellite of each global navigation satellite. While European independence is a principal objective of the programme, Galileo also gives Europe a place in the GNSS world. The programme is designed to be compatible with all existing and planned GNSS and interoperable with GPS and GLONASS. In this sense, Galileo is positioned to enhance the coverage currently available, providing a more seamless and accurate experience for multi-constellation users around the world. [15]

2.1.1 Galileo status

Satellites GIOVE-A and GIOVE-B for experimental steps were launched in 2005 and 2008 respectively, testing critical Galileo technologies, and also the securing of the Galileo frequencies within the International Telecommunications Union. Operational Galileo satellites were launched in 2011.¹

Galileo has made important progresses in recent years. Eighteen Galileo satellites are currently orbiting the Earth, and the supporting ground station infrastructure is working properly. As a result, Galileo is now quiet ready to be used.

With the introduction of Galileo Initial Services, Galileo officially moves from a testing phase to a provision of live services. This is the first time that users around the world can be guided using the navigation, positioning and timing information given by Galileo’s global satellite constellation. Galileo Initial Services are the first step on the way to its Full Operational Capability. The Galileo performance will gradually improve as the constellation will increase with additional satellites. Once completed in 2020, users will benefit from its reliability and coverage and its full first-class performance. [15]

Galileo-enabled devices

European satellites are guiding users providing them with global positioning, navigation and timing information. In the advance to Galileo Initial Services, many ahead companies have created Galileo-enabled receivers, modules and chipsets — currently, many of which are available on the market. Today, 17 companies, which represent more than 95% of the global satellite navigation supply market, produce Galileo chips ready². To have a look of the range of the different devices which are Galileo-enable, there is a widget in the European Global Navigation Systems Agency webpage³.

Mobile devices

In mid-2016, the Spanish technology company BQ launched the Aquaris X5 Plus, the first European Galileo-ready smartphone. Integrating Qualcomm’s Snapdragon 652 processor, the Aquaris X5 is a multi-constellation smartphone that is capable of receiving also GPS, GLONASS and BDS signals. [16]

Since December 2016 Galileo satellite positioning system has been operating as an 'initial service', with 18 of the 30 satellites that will form the system in 2020. Its signal offers higher resolution than GPS and Glonass competitors, and the ability to use Galileo could be an argument to boost new mobile devices with more accurate location. Following mobile specifications, some chips as Kirin 960 eight-core chip are available to receive signals from the constellations GPS (US), Glonass (Russia), BDS (China) and Galileo (Europe).

¹http://www.esa.int/Our_Activities/Navigation/Galileo/What_is_Galileo
²https://www.gsa.europa.eu/galileo/services/initial-services/galileo-enabled-devices
³http://www.usegalileo.eu/EN/index.html
The test results has been disappointing. In theory, those new devices must capture all the available satellites, getting the best signals they receive to position themselves more effectively, transparently for the user. The system, so transparent to Galileo, that it can not be seen. Every latest chips can receive signals from all four systems. The question is why the available software does not recognizes them. Which is not apparently the situation.4

Figure 2.2: Example of GNSS app called GPS Test. As it shows, there are many satellites in view but none of them are Galileo (GPS are the circle shape ones, GLONASS are the triangle shape and Galileo are the diamond shape6)

As the app shown in Figure 2.2 there are many other apps to test the GNSS status. Unfortunately none of them shows the Galileo System for the moment. The requirements for that are that the mobile device (GNSS receiver) are made to use Galileo to compute its position and also the app can have the choice to include Galileo data. Both hardware and software must have included Galileo as one of the satellite navigation systems to test it.

Autonomous Vehicles

The Galileo project will provide satellite navigation to the autonomous car. This is confirmed by Javier Benedicto, head of the Galileo project at the European Space Agency (ESA), at a conference at the annual Ametic conference.7

The agency, in collaboration with Renault, Volkswagen and soon with Seat, research the development of this type of vehicles. The signal from Galileo is capable of reaching places where there is no direct line of sight to the satellite, may be the key to guarantee the safety of the autonomous car, together with the GPS and the 5G network. Galileo is protected from cyber attacks because it is not connected to the Internet and allows to deploy navigation in places where it does not reach 5G. Currently, the 18 satellites in orbit of Galileo provide

---

4http://www.elmundo.es/economia/2017/07/20/59709da3468aeb893a8b4670.html
an accuracy of less than one meter and the aim of the ESA is to fit “a few centimeters”, which will adjust their options for the autonomous vehicle.

More improvements in GNSS constellations are further contributing to the path in the way to the fully autonomous vehicle, with Galileo features such as Open Service NMA and dual frequency are a key role in providing an efficient, reliable, robust and low-cost defence against spoofing attacks or jamming. [16]

The Galileo Open Service (OS) will provide a Navigation Message Authentication feature, called the Open Service Navigation Message Authentication (OS-NMA). Through this feature, users can verify that a navigation message comes from a Galileo satellite and not from a potentially malicious source. [8]

2.1.2 Galileo, improvements

Nowadays, Galileo is providing improved positioning and timing data with significant positive implications for many European services and users. With its satellites working together with GPS and GLONASS, there are more satellites available, meaning more accurate and reliable positioning for end users. In particular, navigation in cities, where buildings can often block satellite signals. The increase in the number of satellites is fundamental for having at least the minimum of satellites required, taking into account that the elevation mask in cities could raise until very high elevation angles [17]. Indeed, Galileo is bringing many other improvement to the GNSS world as:

- The power received from the Galileo signals is more than the double of the power of the C/A code from the current GPS, which decrease tracking noise for both phase and code ranges; [18]

- Each Galileo signal also has a pilot (dataless) component, which can be easily acquired independently, with no data bits decoding. This has several benefits to a data-bearing signal like the GPS CA code, in addition to reduced noise and better tracking integrity at low signal power; [18]

- A better 3-step coding scheme for navigation bits is used (Viterbi convolutional encoding, bit interleaving and cyclic redundancy check (CRC)). This significantly increases the reliability of navigation message decoding when there is interference or with low signal power. [18]

- Galileo’s excellent 30 nanosecond timing accuracy helps to enabling more resilient synchronisation of banking and financial transactions, telecommunication and energy distribution networks which contributes to operate more efficiently; [15]

- It boosts European innovation, contributing to the creation of many products and services, creating jobs and allowing Europe to participate in the EUR 175 billion global GNSS market. [16]. Satellite navigation has helped the world economic growth of the high-tech industries. It is predicted that the global satellite navigation market will grow by more than 18% up until 2019. The Galileo resiliency is expected to enable many new applications and services that will benefit for increased positioning reliability, in addition to driving economic growth in Europe and beyond. [15]

- Indeed, Galileo provides European citizens with independence and sovereignty, a range of environmental benefits and several new specific services to the Galileo programme (Open Service, Search and Rescue, Commercial Service). Galileo’s Search and Rescue service provides a reduction of the time it takes to detect emergency pain beacon signals from up to three hours to just ten minutes. Since the locations of the beacons are also determined more accurately, people in distress, whether at sea or in the mountains, can be rescued more quickly. [15]

---

Mitigating risks

Satellite positioning has become a fundamental service that it is often taken for assumed. Just think what would happen if GNSS signals were suddenly disappeared. Ship and aircraft crews, truck and taxi drivers and thousand of millions of people around the world would suddenly be lost. Indeed, public utilities, financial and communication activities, humanitarian and security operations and emergency services would all come to a stop. e.i as the use of satellite-based navigation systems continues to expand, the consequences of a signal failure become even greater.

With the addition of Galileo to the global GNSS constellation, these risks are minimised and also ensure better performance and accuracy for the all the users. [15]

2.2 Current Galileo performance

After the Declaration of Galileo Initial Services, in the first quarterly reporting period the measured Galileo Initial Open Service performance figures generally exceeded the Minimum Performance Levels (MPL) targets specified in the OS-SDD by important margins. It has summarized the following data on Galileo:

- Availability of a Healthy Signal, with average monthly values better than 97.33%, is significantly above expectations, where the MPL is of 87%. The “Availability of HEALTHY Signal in Space” is defined, for each Galileo constellation operational satellite, as the percentage of time that the specific satellite broadcasts healthy Galileo Open Service Signals in Space. [19]

- The Space Ranging Accuracy signal shows a 95% monthly accuracy better than 1.07 [m] for individual space vehicles. In concern of the [OS-SDD] MPL, in this case it is also achieved, the threshold is set to 7 [m]; [19]

- Galileo Availability UTC time determination service was achieved compared to the [OS-SDD] MPL target of 87 percent, with a monthly value of 100 percent;

- “Availability of GPS to Galileo time offset (GGTO) determination was 100 percent in January and March”. February has showed a slightly lower figure of 96.44 percent, although it is still better than the [OS-SDD] MPL aim of 80 percent.

The purpose of this initial OS SDD is to describe the characteristics and performance of the Galileo open system during its initial phase. The initial OS SDD presents the MPLs targeted for such service and defines the conditions under which such MPLs can be reached. [20]

Galileo satellites are broadcasting the time offset between the parallel navigation systems timings, accurate to a few billionths of a second. Formally known as the GGTO the accuracy of the offset is being benchmarked at five nanoseconds or less. [10]

2.3 Code-Carrier Coherence

The quality of GNSS signals directly influences the navigation performance and great importance is attached to the test and evaluation of navigation signal quality at home and abroad. For instance, Stanford

\[ \text{http://gpsworld.com/first-galileo-open-service-performance-report-published/} \]

\[ \text{http://www.esa.int/Our_Activities/Navigation/Galileo_and_GPS_synchronise_watches_new_time_offset_helps_working_together} \]
University has set up a GPS navigation laboratory (GPS Lab) which is committed to monitoring and assessing the quality of navigation signals, and establishing a satellite navigation signal monitoring system. The European Space Research and Technology Centre (ESTEC) conducted the monitoring and assessment of space signals in its navigation labs using the Galileo system and built up several ground-based observatories in different places. [13] Although modern GPS receivers achieve high pseudorange accuracy in line-of-sight (LOS) conditions, multipath remains a dominant source of ranging error in GNSS. On the other hand, for GPS, the carrier code incoherence is well characterised, but for the new constellation it is not. Concerning on the validation of the monitoring receiver based test method, test results show that the accuracy of the code-carrier coherence test is less than 0.5 degree [13]. As it is explained in the Section 1.2.2. Examples of WAAS monitors include a code-carrier coherence monitor, which checks for divergence of the timing references for code and carrier measurements. [21]. Thus, the developed tool for Galileo in this project gains relevance.
Chapter 3

Material and Methods

This chapter explains the procedure and material used to create the tool and which steps to follow to achieve to the same results as shown in 4.

3.1 Galileo

The tool developed in this project is oriented to the Galileo satellite navigation system, thus, it is important having a deep knowledge of what Galileo offers to the world and taking a look of how its internal structure is formed.

The Galileo satellite navigation system will provide five different services: Open Service (OS), Commercial Service (CS), Safety of Life (SoL), Public Regulated Service (PRS), and Search and rescue (SAR). Each service has its application and focuses to a different public. [15]

![Galileo Frequency Spectrum](image)

**Figure 3.1: Galileo Frequency Spectrum** [5]

OS is the most popular GNSS service, it is free of charge and it is not encrypted. It will have world-wide coverage and it is optimized for mass market applications. The OS is provided on three different frequency bands (E1, E5a and E5b). CS is mostly the same as OS but provides an added value by incorporating additional information in the data-stream such as correction data and integrity information. CS is restricted via encryption for commercial exploitation. This service also includes E6. The SoL is adapted for requirements from commercial flight regulations. This has resulted in stringent requirements on integrity monitoring and time
to alert limits. SoL will be provided free of direct user charges but access will probably be restricted to specific user groups by encryption. [5] [17]

The Galileo PRS is encrypted and is "specifically designed to resist interferences, as spoofing" (an attack in which a person or organization supplants an identity by falsifying data, thereby gaining an illegitimate advantage). With its improved security features PRS allows government authorities to enjoy a continuity of service across a wide range of military and civilian applications with important benefits for public safety and security. PRS will be only available to users authorized by the Member States of the European Union and ESA. 1

SAR is a support of COSPAR/SARSAT system. "The added value of the SAR service is the communications link between the person needing rescue and the search and rescue operation centre". The communication link is double way and allows the communication of short text messages. The Galileo system will provide a global coverage of this service. [5]

The different services are offered on multiple signals and frequency bands. In total, a Galileo satellite has five different RF-signals on three different frequency bands E1, E5 and E6. As it is shown in Figure 3.1, some of the signals have a data channel (navigation data and ranging code) and a pilot channel (no navigation data, only ranging code) with both the I (data) and Q (pilot) components. In total, there are 10 different signals which are used to define the Galileo services.

3.1.1 Frequency bands chosen

After having an overall idea of the main features of each frequency band, it is possible to explain the choice of the frequency bands used in this project. In the Figure 3.1 it is shown the all the frequencies and also which type of service is provided within each frequency band. Indeed, it is shown which frequency band provide also pilot data. The most popular service, as it is said above is the Open Service, thus, only the frequency bands available for this service are chosen. These are E1, E5a and E5b frequency. Then, as it is explained in Section 3.5, 2 frequencies are required. The main one is from which the multipath is obtained and the secondary one contributes with its carrier phase. The reason for this choice is that the effect of removing the ionosphere stronger when the carrier frequencies are further away. Moreover, because the E1 frequency also known as L1 frequency is the most widely used around the world. For those reasons the main frequency will be E1 (1575.4 MHz) and the secondary frequency will be E5a (1176.45 MHz).

3.2 GPSTK

The GPS Toolkit (GPSTK) was developed to support GPS applications, it is an open source library and a suite of applications created for the satellite navigation community, to researchers research purposes and not at lower level coding.\(^2\)

The GPSTK suite consists of a set of core library and auxiliary libraries, and a set of applications. The GPSTK provides a big quantity of functions to solve processing problems related to GNSS such as handle standard formats such as Receiver Independent Exchange Format (RINEX) and Standard Product 3 (SP3). The libraries are the basis for deeper advanced applications distributed as a part of the GPSTK suite. [6].

![GPSTK: The tool kit which helps working on GNSS](http://www.gpstk.org/bin/view/Documentation/WebHome)

Since its first release the GNSS world is evolving into a multi-constellation system: GLONASS has been revived, Galileo is a reality, the Chinese have launched Beidou, the Japanese QZSS, and GPS has slowly been adding the more types of signals to the existing constellation. None of these new systems or signals were operational at the time of the GPSTK's first release, when it was aimed at handling only GPS L1 and L2 signals.

Many changes required to support this multi-GNSS processing have been incorporated into the GPSTK. Examples of this changes in the last years are due to RINEX from version 2 to 3, incorporation of Galileo or Beidou, addition of frequency L5 in GPS, etc. [22]. The Space and Geophysics Laboratory sponsors GPSTK, with the collaboration of the Applied Research Laboratories at the University of Texas at Austin (ARL:UT). GPSTK is a product of GPS research at ARL:UT since before the first satellite was launched in 1978 as a combination of the effort of many software engineers and scientists. Opening source much of their basic GNSS processing software as the GPSTK was decided by the research staff at ARL:UT in 2003. [6]

3.2.1 C++

The GPSTK has an object-oriented design and a C++ implementation, and the original design, which, while sufficient to handle the world of dual-frequency GPS measurements, needed to be extended and expanded to address the multi-GNSS complexities that now exist. This changes have been made in the classes of the library, adding functions and new choices for the objects as TypeID, Satsystem, etc, which may be the inputs for many functions, which just work with the more basic mode, GPS L1 and L2 frequencies. The quantity of current and planned efforts are aimed at improving the reliability, maintainability, and usability of the GPSTK [22].

3.2.2 Doxygen

Doxygen has the GPS ToolKit Software Library Documentation. It describes the architecture and design of the GPS ToolKit (GPSTK), produced by the Space and Geophysics Laboratory (SGL) of ARL:UT. This\(^2\)
documentation is nightly generated from the GPSTk Subversion repository. In this web page, the classes, structures, unions and interfaces with brief descriptions are publicly available. Clicking in each one, its members (public or private) are shown and some information about its hierarchy. It helps the user to know how implement their code, and which class, struct, union or interface fits better. On the other hand, the file list it is also available, where it is openly visible the code of each class. It is useful to know how they work and for doing debugging in the user program.

Due to the fact that there is not a specific manual for using GPSTK, Doxygen is the best way to learn about how to use GPSTK and what it is possible to do and its limitations. There is also some examples which can be imitated for the first steps of a new program.

### 3.3 International GNSS Service (IGS)

The International GNSS Service (IGS), formerly the International GPS Service, is a voluntary collaboration of more than 200 contributing organizations in more than 80 countries. The IGS global tracking network of more than 300 permanent, continuously-operating GNSS stations provides a rich data set to the IGS Analysis Centers, which formulate precise products such as satellite ephemerides and clock solutions. IGS Data Centers freely provide all IGS data and products for the benefit of any investigator.

Perhaps the most important prerequisite for a successful service and the ease of utilization of IGS products is the standardization of data and product formats. IGS has adopted and developed a number of standard formats. The formats used in this project: RINEX and SP3 undergo regular revisions to accommodate receiver/satellite upgrades, or multi-technique solutions, respectively.

<table>
<thead>
<tr>
<th>Format name</th>
<th>IGS Product</th>
<th>Sampling</th>
</tr>
</thead>
<tbody>
<tr>
<td>RINEX</td>
<td>GNSS data</td>
<td>30 sec</td>
</tr>
<tr>
<td>SP3</td>
<td>Orbit/Clocks</td>
<td>15 min (900 sec)</td>
</tr>
</tbody>
</table>

Table 3.1: IGS formats

For RINEX and SP3 files, their products and sampling is shown in the Table 3.1.

### 3.3.1 Receiver Independent Exchange Format (RINEX)

The first proposal for the Receiver Independent Exchange Format (RINEX) was developed by the Astronomical Institute of the University of Berne for the easy exchange of the Global Positioning System (GPS) data to be collected during the first large European GPS campaign EUREF 89, which involved more than 60 GPS receivers of 4 different manufacturers. In August 2015, the CORS network had almost 2,000 stations, belonged by over 200 different organizations, and the network continues to expand.

And more than 40 percent of GNSS receivers are Galileo-ready. Most GPS data software use a well-defined set of observables:

- The carrier-phase measurement at one or both carriers (actually being a measurement on the beat frequency between a generated reference frequency of the receiver and the received carrier of the satellite signal);
- The pseudorange (code) measurement, equivalent to the difference of the time of transmission (shown in the time frame of the satellite) and the time of reception (shown in the time frame of the receiver) of a distinct satellite signal;

• The observation time of the receiver clock reading at the instant of genuineness of the carrier-phase and/or the code measurements; Normally the software assumes that the observation time worths for both the phase and the code measurements, and for every satellites observed.

Consequently most of these programmes do not need the major part of the information that is usually saved by the receivers: They need code, phase and time in the before mentioned definitions, and some station related information like antenna height, station name, etc. [8]

**Observables files**

Each RINEX file consists on a header section and a data section. The header section is placed at the beginning of the file and contains global information for the entire file. The header section contains the same number of header labels in the right side as the number of lines contained in the header section. These labels are a must and have to follow a protocol.

The format has been optimized to extend the minimum space possible and achieving the requirements independently of the number of different observation types of a specific receiver by indicating in the header the observation types to be stored. Allowing variable record lengths the observation records may then be kept as short as possible in computer systems.

Each Observation file basically contains the data from one site and one period. RINEX observables data files include three fundamental quantities that must be defined: Time, Phase, and Range.

**Time**

The measurement time is the receiver time of the received signals. It is identical for the phase and range measurements and is identical for all satellites observed at each epoch. It is expressed in GPS time (not Universal Time).

**Pseudorange**

The pseudorange (PR) is the distance from the receiver antenna to the satellite antenna having into account the receiver and satellite clock offsets:

\[
PR = \text{distance} + (\text{receiver clock offset} - \text{satellite clock offset}) \cdot c \tag{3.1}
\]

where \( c \) is the light velocity.

The Equation 3.1 shows that the pseudorange reflects the real behavior of the receiver and satellite clocks. The pseudo-range is stored in units of meters.

**Phase**

The phase is the whole cycles in the carrier-phase measurement. The measured half-cycles by receivers need to be converted to whole cycles and flagged in the header section by the wavelength factor. The phase, because of the negative Doppler, changes in the same sense as the range. The phase observations between epochs must be connected by including the integer number of cycles. The phase observations do not contain any systematic drifts from the reference oscillators intentional offsets. The observables are not corrected for external effects as satellite clock offsets, atmospheric refraction, etc.
If the converter software or the receiver adjusts the measurements using the real-time-derived receiver clock offsets \( dT(r) \), the coherence of the 3 quantities phase / pseudo-range / epoch must be maintained, i.e. the receiver clock correction must be applied to all 3 observables:

\[
\begin{align*}
  T_{\text{corr}}(t) &= T_r(t) - dT(r) \\
  PR_{\text{corr}}(t) &= PR_r(t) - dT(r) \times c \\
  \text{phase}_{\text{corr}}(t) &= \text{phase}(t) - dT(r) \times \text{freq}
\end{align*}
\]

### Versions

At the present time three major format versions have been developed:
The original RINEX Version 1 presented at and accepted by the 5th International Geodetic Symposium on Satellite Positioning in Las Cruces, 1989.; RINEX Version 2 presented at and accepted by the Second International Symposium of Precise Positioning with the Global Positioning system in Ottawa, 1990, mainly adding the possibility to include tracking data from different satellite systems (GLONASS, SBAS); RINEX Version 3, currently under revision. [8] [24]

<table>
<thead>
<tr>
<th>System</th>
<th>Freq./Band</th>
<th>Frequency</th>
<th>RINEX 2-character Code</th>
<th>Ps.Range</th>
<th>Carr.Phase</th>
<th>Doppler</th>
<th>Sign.Strength</th>
</tr>
</thead>
<tbody>
<tr>
<td>GPS</td>
<td>L1</td>
<td>1575.42</td>
<td>C1,P1</td>
<td>L1</td>
<td>D1</td>
<td>S1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>L2</td>
<td>1227.60</td>
<td>C2,P2</td>
<td>L2</td>
<td>D2</td>
<td>S2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>L5</td>
<td>1176.45</td>
<td>C5</td>
<td>L5</td>
<td>D5</td>
<td>S5</td>
<td></td>
</tr>
<tr>
<td>Glonass</td>
<td>G1</td>
<td>1602+9/16</td>
<td>C1,P1</td>
<td>L1</td>
<td>D1</td>
<td>S1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>G2</td>
<td>1216+7/16</td>
<td>C2,P2</td>
<td>L2</td>
<td>D2</td>
<td>S2</td>
<td></td>
</tr>
<tr>
<td>Galileo</td>
<td>E5</td>
<td>1176.45</td>
<td>C5</td>
<td>L5</td>
<td>D5</td>
<td>S5</td>
<td></td>
</tr>
<tr>
<td></td>
<td>E5a</td>
<td>1207.140</td>
<td>C7</td>
<td>L7</td>
<td>D7</td>
<td>S7</td>
<td></td>
</tr>
<tr>
<td></td>
<td>E5a+b</td>
<td>1191.795</td>
<td>C8</td>
<td>L8</td>
<td>D8</td>
<td>S8</td>
<td></td>
</tr>
<tr>
<td></td>
<td>E6</td>
<td>1278.75</td>
<td>C6</td>
<td>L6</td>
<td>D6</td>
<td>S6</td>
<td></td>
</tr>
<tr>
<td>SBAS</td>
<td>L1</td>
<td>1575.42</td>
<td>C1</td>
<td>L1</td>
<td>D1</td>
<td>S1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>L5</td>
<td>1176.45</td>
<td>C5</td>
<td>L5</td>
<td>D5</td>
<td>S5</td>
<td></td>
</tr>
</tbody>
</table>

Figure 3.3: RINEX 2 observation data keys  [7]

For each type of data, there is a key to call them. They are different in RINEX 2 and 3 and they include the system, the frequency and the type of data. E.g. In RINEX 2 if the data is the GPS carrier phase for the frequency L2, the key is called L2. As one can check in the Figure 3.3. For RINEX 3, the keys have three digits. The information also includes the channel or code. E.g. if the data is the pseudorange for the galileo frequency E5 with no data (pilot) it is called C5Q. As one can check in the Figure 3.4. Fortunately there is an easy conversion between RINEX 2 keys and RINEX 3 keys, but unfortunately not every RINEX 3 key has a match in RINEX 2. Thus, if is precise to work with both RINEX files, it is mandatory to use type of data which can be managed in both RINEX types.
Galileo data in a RINEX file

For the development of this project, has been mandatory, check several times if different type of RINEX files have or not the appropriate data. In this case, the required data is from the Galileo system. As it is said above, the Galileo system data has the first letter as an ‘E’ and in the case of the satellites, in the Galileo system, the format is formed by the E followed by the satellite number.

Two steps are important to follow to check if a RINEX file has or not Galileo data:

- First, the RINEX observable file has to have the type system as mixed or Galileo. It appears in the RINEX header as the letter M for Mixed and E for Galileo. Not only that, then one has to check if the data type need is in the list of keys;
- Second, this can seem not necessary but nowadays, most of the files are provide with capability to have Galileo data, even they say in the headers that they have that data, but sadly, at the end there is no data with that information in the body. For that, it is important to check if really there is any satellite in the body providing the proper data.

As an example of what is said above, the Figure 3.5 is an example of a RINEX file, in this case, this a RINEX file version 3.02, which can manage Galileo data. It is shown how the Galileo data is announced in the header, where it is typed that there are many bands collected from Galileo system in the file. Unfortunately, that is not true, as it can be checked in the body. Due to the extension of the file, only the beginning of the body is printed. It can be checked for anybody who looks up the file called "TWTF00TWN_R_20170850000_01D_30S_MO.rnx"
in the FTP directory from NASA\textsuperscript{6}.

Figure 3.5: Example of RINEX file which has Galileo keys in the data type list in the header but that data does not appear in the body.

The reason why it is said that there is no Galileo data in the body is because the Galileo data is provided by the Galileo satellites, these are named as the letter 'E' plus the satellite number and in the Figure 3.5, every new data line begin with a 'G' which mean GPS satellite. There is no line beginning with the letter 'E'. As an example of a good RINEX file, fitable for this project is the one shown in the Figure 3.6. In this, it is shown the keys in the data type list, equal to the RINEX file in the Figure 3.5, but in this one, one can find the Galileo satellites providing the proper data in the body (E02, E11, E12 and E08 in the first epoch of the day 85 of the year 2017). This RINEX file is named as OZST00USA_R201709200001D_15S_MO.rnx

\textsuperscript{6}https://cddis.nasa.gov/Data_and_Derived_Products/GNSS/RINEX_Version_3.html
3.3.2 Standard Product 3

The first Standard Product 3 format (SP3-a) was proposed in 1989, with the main purpose of exchanging satellite related data (orbit and clock information). The basic format of an SP3 file is a heading, followed by a series of records containing the position and clock records for each satellite listed in the header. A second, optional, record contains the satellite velocity and clock correction rate-of-change.

Three major SP3 versions are defined:

- Original SP3-a proposed in 1989;
- SP3 version b, proposed in 1998, defined to allow the combination of GPS orbits and GLONASS orbits;
- And the current SP3 version c, proposed in 2000. [24]

One can find deeper information of the format and how it is composed in the NASA web page 7. As it is said about the problem of finding Galileo data in the RINEX observable files, there is a similar problem in the SP3 files. Some files have Galileo keys in the satellite list in the header, but then there is no data about Galileo

---

1 https://igscb.jpl.nasa.gov/igscb/data/format/sp3_docu.txt
Figure 3.7: Example of SP3 file which has Galileo satellite keys in the satellite list in the header and that data actually appears in the body.
The Figure 3.7 is an example of a SP3 file which has the Galileo satellite keys in the satellite list in the header and also in the body. One can check that there are several Galileo satellites providing data in the body, concretely the satellites E01, E02, E08, E09, E11, E12, E14, E18, E19, E22, E24, E26 and E30 provide their position and clock data for the first epoch of the week number 1942 and the third day of the week. For that this name of the SP3 is grm19423.sp3.

3.4 Receiver stations

For developing this project, it is required to find receiver stations with some special features:

- Every station chosen has to be located close to the others. There is no a specific numerical limit but they should have the same satellites in view at the same time;
- Every station has to be configured to manage Galileo data and to add it to their RINEX observable files;
- That data, stored in RINEX version 3 files, have to be uploaded online, if not it is not available and testable for everybody.

With the features mentioned above, It is found seven receiver stations in the GPS/GNSS FTP Server Layout provided by UNAVCO from its main webpage. Then one can search mostly every receiver position in the National Oceanic and AtmosphericAdministration main webpage [2]. It is shown the coordinates for these 7 receiver stations in table 3.2. Randomly, 5 of them are selected to developing the tool. It is considered that 5 receiver stations are more than enough to have accurate results. Using 7 stations just would increase the computational cost. For testing, the tool can accept the other station data files as input arguments. In 3.8 one can see the position of this stations in a fragment of the world map.
<table>
<thead>
<tr>
<th>Station ID</th>
<th>X(m)</th>
<th>Y(m)</th>
<th>Z(m)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ISLK</td>
<td>-2,473,741.810</td>
<td>-4,560,945.587</td>
<td>3,698,290.699</td>
</tr>
<tr>
<td>OZST</td>
<td>-2,574,193.153</td>
<td>-4,577,158.410</td>
<td>3,609,640.577</td>
</tr>
<tr>
<td>P513</td>
<td>-2,669,568.304</td>
<td>-4,504,987.875</td>
<td>3,629,597.378</td>
</tr>
<tr>
<td>P558</td>
<td>-2,501,017.874</td>
<td>-4,584,974.606</td>
<td>3,651,264.864</td>
</tr>
<tr>
<td>P588</td>
<td>-2,402,808.890</td>
<td>-4,661,826.080</td>
<td>3,618,747.882</td>
</tr>
<tr>
<td>ACSX</td>
<td>-2,460,183.687</td>
<td>-4,737,087.405</td>
<td>3,479,422.780</td>
</tr>
<tr>
<td>CAT3</td>
<td>-2,540,623.180</td>
<td>-4,682,556.597</td>
<td>3,495,319.263</td>
</tr>
</tbody>
</table>

Table 3.2: Receiver station positions [2]

Also, in Figure [2] there is deeper information about every available receiver station. Clicking in each one, it is possible to access to photographs, site log, even data files as RINEX (The RINEX observables files are not collected from this FTP server because they only have GPS data and neither Galileo nor Glonass data).
Figure 3.9: Emplacement of the Receiver Stations used in this project (ISLK, OZST, P558, P588 and CAT3). [2]
3.5 Multipath

3.5.1 Linear combination of observables

Combining the equations of the observables of carrier phase and code measurements, allows to avoid terms as the ionospheric and geometric terms [25]. Then it is possible to focus on the main terms of this project. The pseudorange measurements can be modeled as:

\[ P_i = s + I_i + e_{MPi} \]  
\[ P_j = s + I_j + e_{MPj} \]  

Where the \( P \) represents any GNSS code measurement, \( s \) is the term which includes the geometric ranges and the difference of clock offsets, \( I \) is the term due to the ionospheric delay and \( e_{MP} \) represents the effect of multipath and noise in the code measurement. And the carrier phase measurements are represented as:

\[ \Phi_i = s - I_i + A_i + e_{MPi} \]  
\[ \Phi_j = s - I_j + A_j + e_{MPj} \]

where \( \Phi \) is the measured carrier phase, \( A \) is the ambiguity term and \( e_{MP} \) represents the carrier phase multipath and noise. Subtracting \( \Phi_i \) from \( P_i \) to find \( MP_i \),

\[ P_i - \Phi_i = 2I_i - A_i + MP_i \]  
\[ MP_i - A_i = P_i - \Phi_i - 2I_i \]

The following steps are required to avoid the ionospheric term Subtracting \( \Phi_j \) from \( P_i \) to solve for \( I_i \):

\[ \Phi_i - \Phi_j = I_j - I_i + A_i - A_j \]

The Equations 3.9 and 3.11 are geometric-free combinations because the \( s \) term is removed with the geometric range inside. The ionospheric delay is proportional to the Ionospheric Electron Content (IEC) and depends on the signal wavelength.

\[ I_i = \frac{A}{f_i^2} IEC, I_j = \frac{A}{f_j^2} IEC \]

\[ \frac{I_j}{I_i} = \frac{f_i^2}{f_j^2} = \alpha \]

Where \( f \) is frequency. With the constant \( A = 40.3 m^3 s^{-2} \).

Substituting \( I_j = \alpha I_i \) into Equation 3.11:

\[ \Phi_i - \Phi_j = I_i(\alpha - 1) + A_i - A_j \]

\[ I_i = \frac{\Phi_i - \Phi_j + A_j - A_i}{\alpha - 1} \]

Where \( I_i \) is a function of the carrier phases, the ambiguities and the frequencies.
3.5.2 Assessing Multipath

As one can see above, the multipath can be obtained as a function of the other terms of the Equation 3.10, and as the ionospheric effect is not interesting in this project, this can be removed. Substituting 3.15 into 3.10:

\[
MP_i = P_i - \frac{2}{\alpha - 1} \left( A_j - A_i \right) + A_i \tag{3.16}
\]

After processing \( MP_i \) for having a zero mean over elevation, which can be checked in Section 3.5.4, the constant term in each arc \( \frac{2}{\alpha - 1} (A_j - A_i) + A_i \) can be removed by finding the average over a given orbital arc and then subtracted from the calculated \( MP_i \) for each epoch.

\[
MP_i = P_i - \Phi_i - \frac{2}{\alpha - 1} (\Phi_i - \Phi_j) \tag{3.17}
\]

Where \( MP_i \) is the estimation of the code multipath error on a pseudorange \( P_i \), while \( \Phi_i \) and \( \Phi_j \) are the carrier phase observables (in units of length) for a \( \alpha \) which depends on the frequencies \( f_i \) and \( f_j \). \( j \) represents any band which is different than \( i \). With multi-frequency Galileo signals, several values of \( j \) are possible, but the particular selection of \( j \) does not significantly affect the results. [26] [25]

3.5.3 Multipath Template Technique

The multipath effect on GNSS measurements depends on the physical environment and the receiver-satellite geometry. As the Galileo satellites are in nearly circular orbits at an approximate altitude of 23,222 km, they will again be on the same position on the earth’s surface at the end of 10 sidereal day (each is approximately 23 hrs 56 mins in length). Thus the viewing geometry is the same each 10 days with respect to solar day, but with a shift of about four minutes per day. When the physical environment remains unchanged from day to day, then the multipath disturbance will be almost constant every 10 days. This point is interesting from the point of view of forecasting and seeking anomalies. From the model based on several days of multipath measurements, the multipath values and curve shape should be similar, as it shown in [27].

3.5.4 Antenna Calibration

The linear combination of observable equations shown above eliminates the range, clocks, troposphere and the ionosphere delays from the observations. What remains (Equation 3.17) is a combination of code and carrier phase multipath, carrier phase ambiguities, noise, and finally code and carrier antenna effects. The carrier multipath, noise, and antenna effects are much smaller than the corresponding effects for code, thus, carrier multipath and noise can be disregarded. The presence of the carrier phase ambiguities means that it is not possible to perform an absolute calibration of the antenna effects. To eliminate this ambiguity it is used the variation of the multipath observable with elevation to retrieve the multipath measured for a specific angle. Thus, for observables in the same arc, it is computed:

\[
dMP_{cx} = MP_{cx}(\varphi - \Delta\varphi) - MP_{cx} \tag{3.18}
\]

Where \( \varphi \) is the elevation angle and \( MP_{cx} \) is the multipath for a type of data called "cx" (for example, c1 for L1 frequency or c5 for E5 frequency) This removes the constant terms from the multipath combination, namely the ambiguities. Taking this into account, it is possible to define an approximate derivative with relation
to the elevation angle as:

$$\frac{dMP_{cx}}{d\varphi} = \frac{MP_{cx}(\varphi + \Delta\varphi) - MP_{cx}(\varphi)}{\Delta\varphi}$$  \hspace{1cm} (3.19)

The derivative multipath data vs the elevation angle can be binned into elevation bins. Every bin has the same width $\delta\varphi$, and it is centered on the a discrete set of angles, referred as $\varphi_i$. Moreover, the derivative will be modeled as a constant value on each bin. Averaging all the observations in the same nadir angle bin over time will eliminate the effect of noise, and will provide a better estimate of the derivative of the Antenna Calibration for each nadir angle:

$$\frac{d\text{Ant}}{d\varphi}(\varphi_i) = \sum_{\varphi \in \varphi_i \pm \delta\varphi_{bin}} \frac{dMP_{cx}(\varphi)}{d\varphi}$$  \hspace{1cm} (3.20)

Where $N$ is the number of observations with nadir elevation angle. After this step, it is left with a quantity that should have very little noise, but it is merely the rate of change with respect to the elevation angle. Integrating this it is found the antenna effect for a given angle, affected by a constant value:

$$\text{Ant}(\varphi) = \int_{\varphi_{min}}^{\varphi} \frac{d\text{Ant}}{d\varphi} (\delta) d\delta + K$$  \hspace{1cm} (3.21)

The constant $K$ is determined by an adequate boundary condition. It it considered a zero mean boundary condition, defined as:

$$\langle \text{Ant}(\varphi) \rangle = \frac{1}{\varphi - \varphi_{min}} \int_{\varphi_{min}}^{\varphi} \frac{d\text{Ant}}{d\varphi} (\delta) d\delta = 0$$  \hspace{1cm} (3.22)

This means that the integration constant is defined as:

$$K = -\frac{1}{\varphi - \varphi_{min}} \int_{\varphi_{min}}^{\varphi} \frac{d\text{Ant}}{d\varphi} (\delta) d\delta$$  \hspace{1cm} (3.23)

Finally, because it is dealing with bins, the above expressions are replaced by discrete sums. The Antenna bias at the bin $j$ is given by:

$$\text{Ant}(j) = \sum_{i=1}^{i=j} \left( \left. \frac{d\text{Ant}}{d\varphi} \right|_{i} \Delta\varphi \right) + \left. \frac{d\text{Ant}}{d\varphi} \right|_{j} \frac{\Delta\varphi}{2} + K$$  \hspace{1cm} (3.24)

$$K = \frac{1}{N_{bins}} \sum_{j=1}^{j=N_{bins}} \left[ \sum_{i=1}^{i=j} \left( \left. \frac{d\text{Ant}}{d\varphi} \right|_{i} \Delta\varphi \right) + \left. \frac{d\text{Ant}}{d\varphi} \right|_{j} \frac{\Delta\varphi}{2} \right]$$  \hspace{1cm} (3.25)

### 3.6 Code carrier coherence

Until this point, Multipath has been computing to track coherence, variations with respect to the nominal Multipath receiver environment has been detected, and at the end data is combined from various receivers to reduce the effect of errors. Now, all of mentioned above is joined to compute if there is any distortion in the data results.

Detection of this distortion is made difficult due to the fact that the ranging error caused by such distortion is dependent on the spread spectrum receiver discriminator type, correlator spacing, and bandwidth. The importance of avoid it is because it is a threat to the integrity of the whole system.

$$ccc^i = \frac{\sum_j \left[ \mu_j^2/(\sigma_j^0)^2 \right]}{\sum_j \left[ 1/(\sigma_j^0)^2 \right]}$$  \hspace{1cm} (3.26)
where \( \mu_{ij} \) is the multipath deviation for reference receiver \( j \) and satellite \( i \), and \( \sigma_{ij}^2 \) is the multipath error Standard Deviation for reference receiver \( j \) and satellite \( i \). A satellite failure is declared when the code-carrier coherence test statistic, \( ccc^i \), exceeds a threshold. [28]

### 3.6.1 Gaussian error propagation for CCC

If some quantities \( x_1, x_2, ..., x_n \) are measured with uncertainties \( \delta x_1, \delta x_2, ..., \delta x_n \) and then is it required to calculate some other quantity \( F \) which depends on \( x_1 \) and \( x_2 \) and so forth, what is the uncertainty in a function \( F \)? The answer will show that the uncertainties \( \delta x_1, \delta x_2, ..., \delta x_n \) “propagate” to the uncertainty of a function “\( Q \)”. The error propagation assumes that the quantities \( x_1, x_2, ..., x_n \) have errors which are uncorrelated and random. (These rules can all be derived from the Gaussian equation for normally-distributed errors and, merely to be able to use them.) [29]

To compute the error propagation for CCC, the variables of the CCC function have to have their own and independent Standard Deviation and they have to be the mean of Gaussian functions.

With that condition:
Let \( x_1, x_2, ..., x_n \) be random variables with Gaussian Mean \( \mu_1, \mu_2, ..., \mu_n \) and Standard Deviations \( \sigma_1, \sigma_2, ..., \sigma_n \). Let \( F \) be a linear function from \( \mathbb{R}^n \) to \( \mathbb{R} \):

\[
F(x_1, x_2, ..., x_n) = \sum_i a_i x_i + b \tag{3.27}
\]

Then \( F \) is a random variable with mean and Standard Deviations given by

\[
\mu_F = \sum_i a_i \mu_i + b \tag{3.28}
\]

\[
\sigma_F^2 = \sum_i a_i^2 \sigma_i^2 \tag{3.29}
\]

Applying this procedure for CCC it is relatively simple, because in the Equation 3.26 CCC depends only on \( \mu_{ij} \) and its Standard Deviation.

Since the individual leveled Multipath mean is zero, the mean CCC will be zero. As for the error, the variance is:

\[
(\sigma_{ccc}^i)^2 = \sum_j \left( \frac{1}{\sum_k (\sigma_{ij})^2} \right)^2 (\sigma_{ij})^2 \tag{3.30}
\]

The variance deviation for the CCC becomes simply:

\[
(\sigma_{ccc}^i)^2 = \frac{1}{\sum_j (\sigma_{ij})^2} \tag{3.31}
\]

If every receiver station have the same noise, the global variance would be the Standard Deviation divided by the number of satellites. In this project 5 satellites are used to compute ccc.

\[
(\sigma_{ccc}^i)^2 \approx \frac{(\sigma^i)^2}{N} \tag{3.32}
\]

---

[28] https://www.physics.ohio-state.edu/~gan/teaching/spring04/Chapter4.pdf
3.7 Programming code explanation

In Appendix B is shown the developed code with comments. In this section the procedure of what is programmed is explained.

3.7.1 Requirements

First of all, it is important to make clear what is necessary for developing the tool.

- GPSTK is a dynamic library used as a support for managing GNSS data. It has many advantages, because it has several classes to manage RINEX, use GNSS structures, etc. but it also has some limitations. See Section 3.7.2;
- The main objective, as it is said before, is to compute the ccc. In that formula (Equation 3.26) first, the multipath and the Standard Deviation must be calculated;
- The multipath is calculated as it is explain in Section 3.5. For that it is required the code and phase from Galileo system. The code is required in 1 band and the phase is required in 2 bands. In this project the Multipath is computed as function of elevation angle;
- The code and phase are collected from RINEX observable files. How to collect this type of data is explained in Section 3.7.5. These RINEX observable files must have the convenient data. Not all the RINEX file versions can take Galileo data. See Section 3.3.1;
- For getting the RINEX observable files, one can find them in some FTP servers. See Section 3.7.4;
- The elevation angle is obtained from the data of the antenna position and the satellite position. The antenna position is obtained from the RINEX header and the satellite position from the SP3 files;
- The SP3 are obtain from FTP servers, these files must have also the appropriate data. They must have the Galileo ephemeris;
- The Standard Deviation is computed from the Antenna Calibration and the multipath data. The Antenna Calibration is estimate as it is shown in Section 3.5.4;
- Avoid cycle slips. They can provide wrong measurements of data (phase measurements) and the following error in the multipath estimation. See Section 3.7.3.

3.7.2 GPSTK limitations

Most of the limitations found for using GPSTK are due to the fact that the tool is oriented to the Galileo System. This System is no completely operative, and many of the classes and functions of GPSTK are created to work with GPS (L1 and L2 bands). Thus, working with Galileo system and its bands may have limitations as it is shown further on.

3.7.3 Cycle slip

First, it is required to avoid cycle slip for computing multipath and the Antenna Calibration. In the Antenna Calibration procedure, the input is the derivative of multipath along elevation (Those curves can have similar shape than multipath along time). Avoiding cycle slip is important here because that derivative in those points are extremely high due to the cycle slip, it is shown as a jump in the Multipath curve of the Figure 4.3. To
calculate that, there are some performed classes in gpstk to detect situations where cycle slips could be present. For GPS, and with frequencies L1 and L2 there are a combination of two methods which uses the Melbourne-Wubbena combination and a 2nd order fitting curve using a least mean squares adjustment method. Probably, this combination is the most effective way to detect cycle slip but using them with Galileo is not possible. Instead, other algorithm in GPSTK is used to detect cycle slips. This algorithm will compute the bias between code and phase, and will compare it with a mean bias that is computed on the fly. If the current bias exceeds a given threshold, then a cycle slip is declared. Then, if a cycle slip is declared that epoch is considered invalid, and there is a gap in the elevation data. When that gap appears, the arc is closed and the new data is added for the following arc.

### 3.7.4 Input files

The input files can appear inside the code or as arguments when the program is run. Two types of files are used: RINEX Observable files and SP3 files.

#### RINEX Observable files

For using Galileo, as it is said in 3.3.1, although RINEX 2.11 can have Galileo data, it is difficult to find it in RINEX version 2, it has been searched in many stations without success. In GPSTK, the classes which are performed to avoid cycle slips only can have as input RINEX observable files version 2, but unfortunately, they can not accept RINEX version 3. There are two options, create an algorithm to detect cycle slips, which could be hard and time expensive, or obtain Galileo data 2.11 with the required data. It is more feasible to find Galileo data in RINEX 3 (RINEX 2 files do not have Galileo data) and convert them to RINEX version 2.

For using data structures in GPSTK, in concrete in gnssRINEX, it is required to use RINEX observable files version 2, for the moment version 3 is not allowed as the input of this data structure. On the other hand, although RINEX version 2.11 can have Galileo data, the unique version uploaded which have this kind of data is version 3. There are many FTP directories with RINEX data for downloading, some of them, it is supposed they can have Galileo data (The data is Mixed if the RINEX file has more than a unique type of data, for example GPS, Glonass or Galileo), a few of them have Galileo data and neither of them which have Galileo data are RINEX version 2.

The UNAVCO main webpage has a FTP server[^9] which has both RINEX 2 and RINEX 3 files for downloading. What is performed is downloading RINEX files version 3 with Galileo data and convert them into RINEX files version 2.11 which can contain this data. There is a free online tool to do this[^10]. For the output files, their names are composed for: first, the name of the receiver station, then, the day of the year and dot plus the year. For the example, it is required the RINEX observable file for the receiver station with id P558 for the day 91 of the year 2017. The result is “P558091A00.17O” where the “O” at the end means observable file.

#### SP3 files

On the other hand, it is mandatory to have proper SP3 files. They must have Galileo ephemeris data. Although Galileo is not operative yet, there are some available ones (Not many). For example CNES (Centre National d’Etudes Spatiales) and CLS (Collecte Localisation Satellites). Since 2007, CNES/GS and CLS teams have been processing regularly GPS and GNSS data from a worldwide network of IGS permanent stations[^11].

Figure 3.10: RINEX files version 3 uploaded and ready for being converted

Figure 3.11: RINEX file version 3 loaded. Here, one can choose different options. It is chosen to get as output a RINEX file version 2.11, with the name structure shown and only Galileo data
They compute precise GNSS orbits together with Earth rotation parameters and stations coordinates at the sub centimeter level. There is a SP3 file from the list of files called grm, which has Daily GPS, GLONASS and GALILEO ephemeris/clock at 15-min intervals.

### 3.7.5 Computing Antenna Calibration

The Antenna Calibration is what is called the mean multipath for a specific receiver station along a concrete number of days. From that definition is possible to define the Standard Deviation forward in Section 3.7.5.

#### Collecting data

To compute multipath in the section 3.5, it was said that two frequencies are required. In section 3.1.1 it is explained why they are E1 and E5. For the Multipath equation 3.17, the E1 Multipath is computed, for achieving that, the E1 and E5 codes must be obtained but also E1 carrier phase. The GPSTK library let one work reading the observable file epoch by epoch and adding the relevant data and avoiding every type of damaged or incorrect information to a data structure called gnssRINEX. What is required to get this is specify the typeid (kind of data required), then a loop is developed for every present satellite in the available Galileo satellite list. For each satellite, the tool for computing the linear combination of equations and cycle slips is declared and defined (see section 3.7.3). After that, the codes and phases are obtained and the Multipath computed. At the same time the elevation angle is calculated as the elevation angle from the antenna position, which is obtained from the RINEX observable file header, and from the satellite position. The satellite position is obtained from the SP3 files.

#### Binning Elevation

For representing plots for multipaths depending on elevation, the elevation was binned into integer numbers. As a result, the elevation angle is a discrete variable and thus, multipath values are represented against a range of numbers from 0 to 90 where there are only integer values. To have a value of Multipath for each elevation angle for every arc, while the satellite is in the range of the same whole elevation angle number, the mean of multipath is computed.
Figure 3.12: Example of how to bin elevation

In the Figure 3.12 is shown an example of a binning elevation. The grey dots represent the multipath computed from the data in the RINEX observable files. They are not fixed in specific elevation values, but they are distanced between 0.2-0.4 degrees one from the previous. In the Section 4.3.1 it is shown the results for the multipath after binning elevation.

Computing Standard Deviation

After computing the Antenna Calibration, it is possible to compute the Standard Deviation as the deviation of every Multipath values from the Antenna Calibration Multipath. The Standard Deviation is computed as:

$$\sigma^i = \sqrt{\frac{\sum_{k}^N (\mu_k - \text{Ant})^2}{N - 1}}$$

(3.33)

Where $\sigma^i$ is the Standard Deviation for the receiver station $i$ for a specific elevation angle, $\mu_k$ is one of the multipath measurements for that specific elevation angle and $N$ is the number of measurements obtained along a specific period of time for an specific elevation angle. The time chosen for getting all the data for computing both the Antenna Calibration and the following Standard Deviation is one week. This computation is done for every satellite together instead of using one by one.

3.7.6 Computing CCC

To compute the CCC value, it is used the Equation 3.26. For accomplish that, the measured leveled multipath is used in place of $\mu^j_i$, and the Standard Deviation of Multipath as a function of elevation in place of $\sigma_i$.

The mean Multipath that is computed identifies any systematic effects in the data. That the Multipath is close to zero indicates that the systematic effect is mostly zero, which is to be expected given the requirements on IGS stations to have good quality.

The Standard deviation of Multipath that was computed in the Section 3.7.5 measures the Multipath noise level as a function of elevation per site. That was done for the multipath values used to compute the Antenna Calibration, here the Standard Deviation is done with the leveled multipath explained in Section 3.7.6. This
new technique consists on:

- Compute Biased Multipath: for each epoch in the pass, compute the $MP$;
- Compute mean $MP$ per pass: For each pass compute the mean $MP$;
- Compute leveled $MP$: For each epoch, subtract mean Multipath from the $MP$ at that epoch;
- Compute leveled $MP$ as a function of elevation: Bin the leveled $MP$ as a function of elevation bins;
- Compute the mean and Standard Deviation of the leveled Multipath for each elevation bin.

**Computing leveled Multipath**

To compute the leveled Multipath is mandatory to compute first, the Biased Multipath and then, the Mean Multipath.

To compute Biased Multipath: for each epoch in the pass, it is computed the Multipath linear combination for each frequency, using the procedure used in Section 3.5.

To compute Mean Multipath per pass for each frequency: For each pass and frequency the mean Multipath over the full pass is computed as:

$$\bar{MP} = \frac{\sum_i M P^i_{j,f}(t)}{N_{j,f}^i}$$

(3.34)

Here $N$ is the number of observations for the pass where $f$ means that belongs to the same pass. Finally, to compute leveled Multipath: For each epoch, subtract $\bar{MP}$ from the $MP$ at that epoch:

$$\mu^i_j(t) = M P^i_j(t) - \bar{MP}$$

(3.35)

For each epoch it is important to compute also the elevation, so from the leveled Multipath and elevation it is possible to organize the data in terms of leveled Multipath as a function of elevation.

This procedure provides the Multipath observable per pass.

### 3.8 CCC threshold

After computing the leveled Multipath and its deviation from the Antenna Calibration Multipath, every variable from the right part of the CCC equation (Equation 3.26) is known. Then, it is possible to compute CCC.

The CCC should be a function of time with zero mean and with noisy shape. Its Standard Deviation is computed through the procedure called Gaussian error propagation explained in Section 3.6.1.

The threshold value depends on how safe the system is required to be. The safer the system the more false alarms there will be. If a probability of 95% of having the data in the correct environment is required, there will be a 5% where the values would overpass this threshold and the flag would be on, in that case, there will be 5% of the values causing false alarms, because the code-carrier incoherence does not really exist.

If the threshold value is considered higher, the system would last longer to detect this feared event but there would be fewer false alarms. If the threshold is establish lower, it would happen the opposite.

To establish a criteria of what threshold is appropriate, a probability of having 95% of good data is chosen but using the worst case standard deviation, which means there is a much smaller probability than 5% of
having a CCC value which overpass the threshold because most of the data will not have the highest Standard Deviation, they will have a lower one. If the Standard Deviation is lower, as it said above the probability of overpassing the threshold is lower than 5%.

\[ PR(\mu - 2\sigma \leq X \leq \mu + 2\sigma) \approx 0.9545 \] (3.36)

Where \( \mu \) is the mean, \( \sigma \) is the Standard Deviation and \( X \) is an observation from a normally distributed random variable. Applying to this work, \( X \) is the ccc value, \( \mu \) is the mean of CCC, which should be zero, and \( \sigma \) is the most extreme Standard Deviation of CCC obtained along the time.
The Equation 3.36 shows that the CCC threshold is \( \mu \pm 2\sigma \).
In this chapter the results obtained are plotted and shown. It is shown the evolution of the procedure until getting the CCC aimed threshold. In Appendix A it is explained what R is and how it is used to plot the data results.

(a) Header and beginning of body

(b) Continuation of the body (not the end of the body)
4.1 Suitable RINEX Input

After downloading the RINEX observable file version 3 and converting it into version 2 as explained in section 3.7.4, it is shown in the Figure 3.10 and in the Figure 3.11 how to do it, here in the Figure 4.1 the output is shown. As one can see, there is information of what Galileo data is available in the observable type list. Because it has been created a RINEX Observable File with only Galileo data, there are not GPS nor GLONASS information. This file fits perfectly as input in the program created of C++. It is appropriate to be manipulated by the GPSTK classes, in concrete, to be the input of data structures. This part became essential because other RINEX version are not allowed as inputs. The Figure 4.1 shows the RINEX Observable file of the station called OZST for the day 92 of the year 2017. It corresponds with the 2nd of April of 2017, this is important to know because the SP3 file date is typed in week number and day of the week (WN: 1943 and DoW: 1). For the following sections, this station (OZST) and that date will be used to show an example of the intermediate results.

Figure 4.2: Continuous Multipath function of elevation
4.2 Raw Multipath

After computing Multipath for every epoch during a day, the result is a big amount of data with a jump between two consecutive points in elevation of 0.03 degrees approx. and several points to choose to identify a precise Multipath to each elevation angle. In the Figure 4.2 it is shown an example for the station OZST for the day 85 of 2017.

As one can appreciate from the Figure 4.2 the Multipath values normally are noise and more or less constant with the elevation, also there is a jump in the satellite of the color purple, and the most remarkable feature is that the data lines are separate one from the other.

The cycle slip is taken into account for doing the program and it is avoidable using the appropriate algorithm, it is used and works properly for further results. The separation between the curves is due to the term called ambiguity. This ambiguity term is a constant number for each arc but is different for different arcs. Because of that reason, it is necessary to compute all the results shown later.

4.3 Antenna Calibration

To show the results of the Antenna Calibration process, firstly, the discrete Multipath will be shown, then the derivative of Multipath with elevation and at the end the result of the computation of the Antenna Calibration for different quantity of days used and also for different selection of frequency bands.

<table>
<thead>
<tr>
<th>Continuous function</th>
<th>Discrete function</th>
</tr>
</thead>
<tbody>
<tr>
<td>{Elevation, Multipath}</td>
<td>{Elevation, Multipath}</td>
</tr>
<tr>
<td>{15.5858, -11.7399}</td>
<td>{16, -11.7239}</td>
</tr>
<tr>
<td>{15.6744, -11.5636}</td>
<td>{16, -11.8801}</td>
</tr>
<tr>
<td>{15.7630, -11.9429}</td>
<td>{16, -11.7652}</td>
</tr>
<tr>
<td>{15.8516, -11.8559}</td>
<td>{16, -11.8710}</td>
</tr>
<tr>
<td>{15.9402, -11.6807}</td>
<td>{16, -11.7217}</td>
</tr>
<tr>
<td>{16.0289, -11.5541}</td>
<td>{16, -11.7217}</td>
</tr>
<tr>
<td>{16.1176, -11.3875}</td>
<td>{16, -11.7217}</td>
</tr>
<tr>
<td>{16.2063, -11.8801}</td>
<td>{16, -11.7217}</td>
</tr>
<tr>
<td>{16.2951, -11.7652}</td>
<td>{16, -11.7217}</td>
</tr>
</tbody>
</table>

Table 4.1: Conversion from continuous Multipath to discrete Multipath function
4.3.1 Discrete Multipath

To compute the discrete Multipath, it was chosen an elevation bin of 1 degree of elevation angle. Then, from the Figure 4.2, for every Multipath consecutive point which share the elevation angles, are collected to compute a discrete Multipath point. e.g. Every consecutive Multipath from the continuous function which has an elevation value from 15.5 to 16.49 are collected to compute the mean and obtaining the discrete Multipath point for the elevation angle of 16.

![Figure 4.3: Discrete Multipath function of elevation](image)

As a result, the curves shown in Figure 4.3 are shown cleaner, and one can work with them to compute derivatives and treat them as discrete functions. The Table 4.3 shows an example of how 11 continuous points are converted into just one for the discrete function. This is also explained in the Figure 3.12. This process is explained in Section 3.5.4 where the discretization is a previous step of the process explained there.

**Standard Deviation**

The Figure 4.4 shows the standard deviation between each set of points in the continuous Multipath...
function (Figure 4.2) and the discrete point computed as the mean. In the Figure 4.4 one can appreciate higher values of standard deviation at lower elevations. The different colors means different satellites, for thus there is not a single value for each elevation value, instead of that, there as many values for each elevation angle value as number of satellites and passes were available at that day.

Figure 4.4: Standard deviation of the continuous Multipath versus the discrete Multipath. It is function of Elevation.
4.3.2 Derivative Multipath with Elevation

After computing the discrete Multipath, the derivative Multipath is calculated of each arc between consecutive points. The result is shown in the Figure 4.5. Here, it is shown the derivative of Multipath with elevation for every satellite. It is possible to have more than one value for a specific satellite sharing the elevation angle value.

As it is shown, most of the values are negative, due to the fact that the function has higher Multipath values at lower elevations and lower Multipath values at higher elevation angle values. On the other hand, there are more positive values at lower elevations due to the fact that at lower elevation angles there are more dispersion and the Multipath values are more unpredictable.

Figure 4.5: Derivative of discrete Multipath with elevation of all the satellites. It is a function of Elevation.
Then, the mean of those values are computed, having as a result the Figure 4.6, where there is a single value for each elevation angle. For computing this is used the Equation 3.20. Here, except for lower elevations, all the values are in the negative part, doing the function of mean Multipath decreasing with elevation.

![Figure 4.6](image)

Figure 4.6: Mean of the derivative of discreteMultipath with elevation of all the satellites. The result is the derivative of the Antenna Calibration with Elevation.

As explained above, the values of the standard deviation here are predictable. At lower elevations, the derivative Multipath with elevation are spreaded, then the standard deviation is higher.
Figure 4.7: Standard deviation of the derivative Multipath with elevation of all the satellites from their mean. It is a function of Elevation.

4.3.3 Multipath of Antenna Calibration

As final results for the Antenna Calibrations, some analysis are done. The previous results are done for a single day. That analysis can be done for as many days as is required. First, the Figure 4.8 shows the quantity of days used to compute the Antenna Calibration. One can check the fact that increasing the number of days, the functions tend to converge, for that it is used 7 days as an appropriate number of days, because it is enough to have good results. More days would increase the computational cost and would not provide more accurate results.

For one and two days, the values depends highly one what features of those days could change a little bit the Multipath values. More than 3 days provide repetitive results wich can be extrapolated to other dates.
This results for both Figure 4.8 and Figure 4.9 are got using the frequency band E1 for the main frequency (Then, the Multipath is called Multipath for the frequency E1) and E5 as the second frequency band.

Figure 4.8: Multipath computed by Antenna Calibration from one to seven days, function of elevation.

The Figure 4.9 shows the results of the Antenna Calibration for different receiver stations. As it is shown, the results are quiet similar for every receiver station. The main differences are at lower elevation, probably are due to the mountain, vegetation, etc. which appear near each receiver station. This values are got for future computation on the test part.

As one can see, the Antenna Calibration function is a zero mean function, which has the property of having the sum of all Multipath values the result of zero.
4.3.4 Standard Deviation

To compute the standard deviation, the procedure explained in Section 3.7.5 is used. This Standard Deviation is the deviation of the Multipath values during a week from the Antenna Calibration shown in the Section 4.3.3. Where the elevation is binned in whole numbers of elevation angles. Figure 4.10 shows the standard deviation computed for the Antenna Calibration obtained in Figure 4.9. Despite having very close curves for the Antenna Calibration, there are bigger differences in the standard deviation curves. The shapes of them are very similar but for lower elevation angles they are quiet different. At lower elevation, the terrain could be different and as the azimuth angle is considered, there could be a mountain in one side of the receiver station or other geographic feature on a single point.
Figure 4.10: Standard Deviation computed by Antenna Calibration for every station for seven days and for the band E1E5, function of elevation.

4.3.5 Different Bands

Figure 4.11 shows the results of the Antenna Calibration for the receiver station called ISLK. Here, a comparison of the Antenna Calibration is shown using different combination of frequency bands. As one can see, the Multipath is lower for E1- E5 and E1- E6. In this project E1- E5 combination is chosen to compute the code-carrier coherence. That is using Galileo E1 frequency band as the main, and Galileo frequency band E5 as the secondary frequency band. The reason for this choice is that the effect of removing the ionosphere is stronger when the carrier frequencies are further away. Moreover, it is the linear combinations that is more widely used as it is explained in Section 3.1.1.
4.4 Code-Carrier Coherence

To compute the test part of the project, it is followed the procedure explained in Section 3.7.6 where it is used the leveled Multipath and the standard deviation. The standard deviation results are provided in Section 4.3.4 and leveled Multipath results will provided then in Section 4.4.1.

4.4.1 Leveled Multipath

The leveled Multipath is explained in Section 3.7.6 and is obtained from the Equation 3.35. For the following plots the E1-E5 frequency band combination is used and the results of using the five receiver station mentioned in Section 3.2 are shown. Then, from the the RINEX observable files of the day 92 of the year 2017 and from the SP3 files from the day 0 and 1 of the week 1943, the Figure 4.12 shows the leveled Multipath for the satellite 30. The Figure 4.13 shows the leveled Multipath for every satellite for the day 92 of 2017.
Figure 4.12: Leveled Multipath computed function of time for the day 92 of 2017.

Figure 4.13: Leveled Multipath computed function of time for the day 92 of 2017.
Figure 4.14: Leveled Multipath computed function of time for the day 106 of 2017.

Figure 4.15: Leveled Multipath computed function of time for the day 151 of 2017.
The Figure 4.14 and the Figure 4.15 shows the leveled Multipath for the days 106 and 151. The Antenna Calibration is done for the days from 85 to 91. These results have been done to compare the performance of using the Antenna Calibration for a test done 1 day later (day 92), 1 week later (day 106) and some months later (day 151).

### 4.4.2 Code-Carrier Coherence

This section provides the representation of the Equation 3.26.

![Figure 4.16: Code carrier coherence for the day 92 for the satellite 30.](image)

The Figure 4.16 shows the code carrier coherence of the satellite 30, using the Leveled Multipath represented in the Figure 4.12 and the standard deviation between this Multipath and the Antenna Calibration shown for every station in the Figure 4.9.
The Figure 4.17 shows the code carrier coherence of every available station in the day 92 of 2017, using the Leveled Multipath represented in the Figure 4.13 and the standard deviation between this Multipath and the Antenna Calibration shown for every station in the Figure 4.9.

![Figure 4.17: Code carrier coherence for the day 92 for every Galileo satellite.](image)

The Figure 4.17 shows a maximum of 0.53 and a minimum of -0.57. Then, the Figure 4.18 and the Figure 4.19 provide the code carrier coherence for the days 106 and 151. They use the Leveled Multipath shown in Figure 4.14 and in Figure 4.15. As one can appreciate, the values at the beginning and at the end of the pass are more scattered than at higher elevations. The satellite at the beginning of the pass raises from the horizon (because it is fixed an elevation mask of 5, the lowest elevation is supposed to be 5) and increases its elevation up to a maximum and then go down until it disappears again into the horizon.
Figure 4.18: Code carrier coherence for the day 106.

Figure 4.19: Code carrier coherence for the day 151.
The maximum of the values shown in Figure 4.18 and in Figure 4.19 is 0.53 and the minimum is -0.55.

### 4.4.3 Standard Deviation for Code-Carrier Coherence

To know how the code carrier coherence is evolving, it has been developed a Gaussian error propagation method explained in Section 3.6.1. Now, this Gaussian error propagation is shown as the standard deviation for CCC. Again, the results are obtained for the days 92, 106 and 151, for frequency band combination of E1-E5 and with the use of 5 receiver station.

To establish a threshold of a flag for knowing when a carrier code incoherence is happening, this plots can help to appreciate how big the values can be and if the Antenna Calibration of the week from the day 85 to the day 92 is valid for the day just after that week, the day of the week later and some day after some months. The Figure 4.20 shows the standard deviation of CCC shown in the Figure 4.16 for the satellite 30 for the day 92.

![Figure 4.20: Standard deviation of CCC for day 92, for the satellite 30.](image)

As one can see in the Figure 4.20 the values do not overpass the value of 0.4 and just one value overpass the value of 0.3.

The Figure 4.21 shows the standard deviation of CCC of the Figure 4.17.
There are only three values shown in the Figure 4.21 overpass the value of 0.3. One of those is from the satellite 30 also used for compute the Figure 4.20.

![Figure 4.21: Standard deviation of CCC for day 92.](image)

The following figures, Figure 4.22 and Figure 4.23 show the Standard deviation for every satellite for the days 106 and 151.
Figure 4.22: Standard deviation of CCC for day 106.

Figure 4.23: Standard deviation of CCC for day 151.
As one can see from the Figure 4.22 and Figure 4.23 is that just few points overpass the value of 0.3 and again, none of them overpass the value of 0.4.

4.5 Code-Carrier Coherence Threshold

After collecting the data shown in the Section 4.4.1, Section 4.4.2 and Section 4.4.3, it is possible to sum some results:

- There are higher values of standard deviation of Multipath at lower elevation, as it is shown in Section 4.4.1 where the dispersion at the beginning and at the end of the pass are higher;
- Leveled Multipath plots show most of the values near zero
- CCC is close to zero and with extreme values of around 0.53 as maximum and -0.57 as minimum;
- Standard Deviation of Multipath is close to zero;
- Only a few points of the Standard Deviation of Multipath are higher than 0.3 and have never been higher than 0.4.

Because the maximum Standard Deviation of CCC is around 0.3, this is considered the most extreme Standard Deviation for the procedure of computing the threshold. As it is explained in Section 3.8 the threshold for CCC is:

\[ \Theta_{ccc} = \overline{CCC} \pm 2\sigma_{ccc} \]  (4.1)

Where \( \Theta_{ccc} \) is the CCC threshold, \( \overline{CCC} \) is the CCC mean and \( \sigma_{ccc} \) is the Standard Deviation for CCC. The mean can be considered zero as it shown in Figures 4.17, 4.18 and 4.19. Then, the threshold depends only in the extreme value of Standard Deviation which is considered 0.3

\[ \Theta_{ccc} = \pm 2\sigma_{ccc} = \pm 0.6 \]  (4.2)
Chapter 5

Discussion

As a result of this project, the seeking of detecting the failure of a satellite because of code carrier incoherence has been concluded successfully.

5.1 Galileo Problems

During the project, Galileo data has been deeply searched, analyzed, rejected and finally used carefully. The main problems of using Galileo RINEX and SP3 files are based on the lack of completion of the Galileo system and the inviability of the systems in general and the receiver stations in particular to hold this "recent" data. On the other hand, although the system requirements are covered to hold Galileo data, the benefits of using a GNSS underdeveloped may be for them not interesting at all. It depends on a balance of advantages and disadvantages. One of the main disadvantages is adjust a global format as RINEX or SP3 from a specific receiver station to hold Galileo data for just obtaining the data from a few satellites.

5.1.1 Number of Satellites

As the Galileo system is underdeveloped, new satellites become available continuously. Although they can be operative, their data broadcast could be not good enough for the filters imposed to compute the CCC value. From the day 92 to the day 151 the number of satellites has increased in three, having 14 for the days 92 and 106 and 17 for the day 151. Fortunately the number of satellites will increase until having the Galileo constellation fully operative. Nicole, Zofia and Alexandre will be launch in the year 2017. ¹

5.2 Multipath as a mean of computation

As one of the main features of the project, Multipath has been computed along the project many times. That is because it appears in the main equation, Equation 3.26, as the mean of Multipath values, as standard deviation, and even as a result of the Antenna Calibration. For that, before knowing if the code-carrier coherence results have been properly computed it is interesting, firstly, test multipath results. There is a big difference between the leveled Multipath, the raw Multipath and the discrete Multipath. While the raw Multipath shows the Multipath for a not binned elevation angles and the values are got with ambiguity,

¹http://space.skyrocket.de/doc_sdat/galileo-foc.htm
leveled Multipath has no ambiguity because the mean has been removed and although discrete Multipath has included the ambiguity term, it is binned. The results obtained from Antenna Calibration Multipath values are lower than 0.2 meters from the mean, which is in coherence with [30], which in “Figure 2.14: Illustration of multipath bias per elevation bin” shows values lower than 1 meter ten years ago (2007).

5.2.1 Standard Deviation

On the other hand, the Multipath data used to compute the Antenna Calibration shows a Standard Deviation lower than 2 meters as it is shown in the Figure 4.10 can be compared with [31] where curve shape and maximum values “Figure 2. L1 Pseudorange Errors for WAAS Network” of their are completely similar as the ones shown in this project. Their Standard Deviation for Multipath L1 is lower than 1.6 meters. In both cases, there are higher values of standard deviation of multipath at lower elevation, as it is shown in Section 4.4.1 where the dispersion at the beginning and at the end of the pass are higher. In this project E1 from Galileo has been used, which is the same band as the L1 from GPS.

5.3 Day of the year

As it is commented several times during the project, the test to compute the CCC value is done for three different days. The first one is the day after the week chosen to compute the Antenna Calibration, the second is the day just after a week after the Antenna Calibration and the third is some day after several months. All of them show results quite similar, for that if there are no any appreciable physical change in the terrain nor in the own receiver station, the same Antenna Calibration can be used for longer time than some days or a week as long as it is safe to trust that there are no changes in the above mentioned.

![Figure 5.1: Days used to compute the CCC test](image)

In Figure 5.1 it is shown the days used as input of the Antenna Calibration in yellow (from the day 85 to the day 91) and the days 92, 106 and 151 used for the text in color blue.
5.4 Choosing a threshold for Code-Carrier Coherence

As the CCC is a value with a very low mean and great dispersion at low elevation angles, an error propagation is used to measure how to detect whenever the values go away from the normality. In that case, it is said that a code-carrier incoherence exists. To know that, it is a good idea to establish a threshold to know if a value, or a group of them overpass it, and turn on a flag to alert the dependent authority. As it is shown in [28] the aim is to establish a threshold which depends on the user differential range error values that are broadcast by each satellite.

This threshold value have to be computed for every new work which uses different receiver stations and satellites and times. Indeed the threshold value depends on the number of receiver stations. As it is shown in Section 3.6.1 the CCC Standard Deviation is inversely proportional of the square root of the number of satellites. That means that the threshold would decrease with the number of receiver stations.

The threshold obtained in Section 4.5 is $\pm 0.6$. This value means that if a CCC value along the time goes higher than 0.6 or lower than -0.6 a flag would become on. This threshold values gains coherence because there is no CCC value which exceed those bounds in the plots shown in Section 4.4.2. As it is said there, the maximum is 0.53 and the minimum -0.57.
Chapter 6

Conclusion

This project has developed a method to find code-carrier incoherences in satellites. Specifically, this method has been applied in the current underdeveloped Galileo system. The lack of research around Galileo compared to the whole GPS ecosystem remarks the importance and innovation of the method described. Throughout the project, ambiguities in the carrier phase observable equation have been cleared and arches have been detected, to accomplish that, data with cycle slips has been removed and Multipath has been computed, analyzed and used to track coherence. The E1 frequency band is used as the main frequency and the E5 frequency band as a secondary frequency.

The first part of the tool consists in obtaining the Antenna Calibration of different receiver stations. This Antenna Calibration gets its value by measuring a mean multipath in a seven-day time frame, which allows to reduce out of range value effects and to know how the multipath performance is along elevation angle. This Antenna Calibration is the base to compute the variations in a posteriori computed leveled Multipath. This procedure is done for five different receiver stations. Then, the second part of the tool computes the leveled Multipath for a specific day and its Standard Deviation from the Antenna Calibration. The code-carrier incoherence tracking can be done thanks to the plots of CCC where the values can be considered correct if never exceed from the normal environment. But a better method is to compute the error propagation of CCC and set a threshold. To sum up, the method consists on establish from the error propagation of CCC, a flag which would become "on" whenever this feared event appears.

It is studied for how long is permissible to use the Antenna Calibration for a specific week. It has been chosen a day, a week and some months later and the results can be considered good for this three test days. In conclusion, if there is no significant change in the receiver station nor in the satellite, there would not be problem of using the same week to compute the Antenna Calibration for a test day method some months later.

The fact of using five receiver stations located in a small area results in a dependency on the elevation. For every single time, every satellite is at similar elevation angle of view for every receiver station chosen in this work. That makes the computation of CCC easier due to the fact that every receiver station is viewing the same satellites. On the other hand, the number of receiver stations is important to establish the threshold, as it is explained in this project, the Standard Deviation of CCC is inversely proportional of the square root of the number of satellites. In conclusion, increasing the number of receiver stations would decrease the threshold. What makes sense because the CCC values become noisy reduced.

The threshold chosen is $\pm 0.6$, what means that any time a CCC value overpasses it, a possible code-
carrier incoherence can be occurring or there is a value which is out of range. On the other hand, if actually this feared event is occurring, there would be many values overpassing the selected threshold.

In my personal opinion, this value should evolve with experience, depending on how many out of range values are during a period of time, and how better it is to receive many false alarms versus catching this feared event on time (if the threshold is lower than the optimal value, there will be many false alarms, but if the threshold is higher, it will be difficult to catch the code-carrier incoherence on time).
Chapter 7

Future Works

This chapter has been done to include every task which has not been done during the dissertation period. There are some tasks which can improve or expand the content of this project, but they are hard enough tasks to carry out their own dissertation or because the required tools to perform them are not currently available.

- The tool performed in this project can support every different combination of frequency bands. To analyze the results of the code-carrier coherence method it is used only the combination between E1 and E5 for Galileo. It would be interesting to research the results for other combination of bands to other services apart of the open-service of Galileo\(^1\);

- Unfortunately, there are only few satellites available from the Galileo system. This project could use every satellite in a near future whenever they will be fully operative\(^2\). This is a very interesting point to test the whole system in case there is any damaged satellite;

- The tool has two steps: first, it computes the antenna calibration from a period of time and then it does a test to compute if there are any code-carrier incoherence in a satellite. An enhance tool would be one which is capable to do it in real time. There is an advantage from this project having the ability to know in real time if there are any damaged satellite because of the code-carrier incoherence;

- Rinex version 2 is used as the input of the tool, as it is explained that is because of the limitation of the GPSTK\(^3\). This open source library is being improved continuously, thus the tool could be improved when those limitations will disappear.

\(^1\)https://www.gsa.europa.eu/news/prs-huge-potential-europe


Appendix A

R Programming language

R is a free software environment for statistical computing and graphics. It compiles and runs on a wide variety of UNIX platforms, Windows and MacOS. One of R’s strengths is the ease with which well-designed publication-quality plots can be produced, including mathematical symbols and formulation where needed. Great care has been taken over the defaults for the minor design choices in graphics, but the user retains full control.¹ R is a programming language and environment commonly used in statistical computing, data analytics and scientific research. It is one of the most popular languages used by statisticians, data analysts, researchers and marketers to retrieve, clean, analyze, visualize and present data. Due to its expressive syntax and easy-to-use interface, it has grown in popularity in recent years.²

A.1 RStudio

RStudio is an active member of the R community. Its belief of freedom and open source data analysis software is a foundation for innovative and important work in science, education, and industry.

Figure A.1: Logotype of RStudio

The many customers who value its professional software capabilities help them contribute to this community.³ In a few words, RStudio is a new open-source IDE for R. It has interesting features for both new and experienced R developers including code completion, execute from source, searchable history, and support for authoring Sweave documents.

RStudio runs on all major desktop platforms (Windows, Mac OS X, Ubuntu, or Fedora) and can also run as a server which enables multiple users to access the IDE using a web browser.

¹https://www.r-project.org/about.html
²https://www.programiz.com/r-programming
³https://www.rstudio.com
A.2 Plotting Results

Due to the big amount of data which have been managed in this projects, it has been necessary to use a tool to extract the information from the output of the program created by C++ Programming Language. The graphs plotted in the results have been done by this program called RStudio.

A.3 R Programming code

```r
library(ggplot2)
library(timeSeries)
library(lmtest)

ContinuousMuVsEl <- read.table("/Users/Guillermo/Documents/master-thesis/calc/MultpathVSElevation.txt", header = TRUE)
ggplot(ContinuousMuVsEl, aes(x=Elevation, y=Multpath, color=PRN)) + geom_point()
+ scale_colour_gradientn(colours = rainbow(10, s=1, v=0.8, start=0, end=max(1,9/10), alpha=1))
+ theme(text = element_text(size=25), axis.text.x = element_text(angle=0, hjust=1))
+ labs(x= "Elevation (degrees)", y= "Multpath (m)"

DiscreteMuVsEl <- read.table("/Users/Guillermo/Documents/master-thesis/calc/MultpathVSElevationWithStandardDeviation.txt", header = TRUE)
ggplot(DiscreteMuVsEl, aes(x=Elevation, y=Multpath, color=PRN)) + geom_point()
+ scale_colour_gradientn(colours = rainbow(10, s=1, v=0.8, start=0, end=max(1,9/10), alpha=1))
+ theme(text = element_text(size=25), axis.text.x = element_text(angle=0, hjust=1))
+ labs(x= "Elevation (degrees)", y= "Multpath (m)"

AntennaCalibrationDays <- read.table("/Users/Guillermo/Documents/master-thesis/data/AntennaCalibrationDays.txt", header = TRUE)
ggplot(AntennaCalibrationDays, aes(x=Elevation)) + geom_line(aes(y=StDev, colour="StDev")) + geom_line(aes(y=StDev, colour="StDev")) + geom_line(aes(y=StDev, colour="StDev")) + geom_line(aes(y=StDev, colour="StDev")) + geom_line(aes(y=StDev, colour="StDev")) + geom_line(aes(y=StDev, colour="StDev")) + geom_line(aes(y=StDev, colour="StDev")) + geom_line(aes(y=StDev, colour="StDev"))
+ theme(text = element_text(size=25), axis.text.x = element_text(angle=0, hjust=1))
+ labs(x= "Elevation (degrees)", y= "StDev (m)

AntennaCalibrationStations <- read.table("/Users/Guillermo/Documents/master-thesis/data/AntennaCalibrationDifferentStations.txt", header = TRUE)
ggplot(AntennaCalibrationStations, aes(x=Elevation)) + geom_line(aes(y=StDev, colour="StDev")) + geom_line(aes(y=StDev, colour="StDev")) + geom_line(aes(y=StDev, colour="StDev")) + geom_line(aes(y=StDev, colour="StDev")) + geom_line(aes(y=StDev, colour="StDev")) + geom_line(aes(y=StDev, colour="StDev")) + geom_line(aes(y=StDev, colour="StDev")) + geom_line(aes(y=StDev, colour="StDev"))
+ theme(text = element_text(size=25), axis.text.x = element_text(angle=0, hjust=1))
+ labs(x= "Elevation (degrees)", y= "StDev (m)

AntennaCalibrationBands <- read.table("/Users/Guillermo/Documents/master-thesis/data/AntennaComparison.txt", header = TRUE)
ggplot(AntennaCalibrationBands, aes(x=Elevation)) + geom_line(aes(y=StDev, colour="StDev")) + geom_line(aes(y=StDev, colour="StDev")) + geom_line(aes(y=StDev, colour="StDev")) + geom_line(aes(y=StDev, colour="StDev")) + geom_line(aes(y=StDev, colour="StDev")) + geom_line(aes(y=StDev, colour="StDev")) + geom_line(aes(y=StDev, colour="StDev")) + geom_line(aes(y=StDev, colour="StDev"))
+ theme(text = element_text(size=25), axis.text.x = element_text(angle=0, hjust=1))
+ labs(x= "Elevation (degrees)", y= "Multpath (m)"

StDev <- read.table("/Users/Guillermo/Documents/master-thesis/data/StandardDeviation.txt", header = TRUE)
ggplot(StDev, aes(x=Elevation)) + geom_line(aes(y=StDev, colour="StDev")) + geom_line(aes(y=StDev, colour="StDev")) + geom_line(aes(y=StDev, colour="StDev")) + geom_line(aes(y=StDev, colour="StDev")) + geom_line(aes(y=StDev, colour="StDev")) + geom_line(aes(y=StDev, colour="StDev")) + geom_line(aes(y=StDev, colour="StDev")) + geom_line(aes(y=StDev, colour="StDev"))
+ theme(text = element_text(size=25), axis.text.x = element_text(angle=0, hjust=1))
+ labs(x= "Elevation (degrees)", y= "Standard Deviation (m)"

dAntEl <- read.table("/Users/Guillermo/Documents/master-thesis/calc/dAntEl.txt", header = TRUE)
ggplot(dAntEl, aes(x=Elevation, y=dAntEl)) + geom_point()
+ theme(text = element_text(size=25), axis.text.x = element_text(angle=0, hjust=1))
+ labs(x= "Elevation (degrees)", y= "dAntEl (m)"

OCC <- read.table("/Users/Guillermo/Documents/master-thesis/data/OCC.txt", header = TRUE)
ggplot(OCC, aes(x=Time, y=OCC, color=PRN)) + geom_point() + scale_colour_gradientn(colours = rainbow(10, s=1, v=0.8, start=0, end=max(1,9/10), alpha=1))
+ theme(text = element_text(size=25), axis.text.x = element_text(angle=0, hjust=1))
+ labs(x= "Time (s)", y= "OCC (m)"
+ coord_cartesian(ylim=c(-1, 1))
ggplot(OCC, aes(x=Time, y=StDev, color=PRN)) + geom_point() + scale_colour_gradientn(colours = "red")
+ theme(text = element_text(size=25), axis.text.x = element_text(angle=0, hjust=1))
+ labs(x= "Time (s)", y= "Standard Deviation of OCC (m)"
+ coord_cartesian(ylim=c(-1, 1))

MuMinusMean <- read.table("/Users/Guillermo/Documents/master-thesis/data/MuMinusMean.txt", header = TRUE)
ggplot(MuMinusMean, aes(x=Time, y=Mu, color=PRN)) + geom_point() + scale_colour_gradientn(colours = rainbow(10, s=1, v=0.8, start=0, end=max(1,9/10), alpha=1))
+ theme(text = element_text(size=25), axis.text.x = element_text(angle=0, hjust=1))
+ labs(x= "Time (s)", y= "Standard Deviation of Mu (m)"
+ coord_cartesian(ylim=c(-1, 1))
```

68
```r
StDevLev <- read.table("/Users/Guillermo/Documents/master-thesis/data/standardDeviationLeveledMu0.txt", header = TRUE)
ggplot(StDevLev, aes(x=Time, y=ccc, color=PRN)) + geom_point() + scale_colour_gradientn(colours = rainbow(10, s=1, v=0.8, start=0, end=max(1,9/10), alpha=1)) + theme(text = element_text(size=25), axis.text.x = element_text(angle=0, hjust=1)) + labs(x="Time (s)", y="ccc (m)") + coord_cartesian(ylim=c(-1, 1))
```
Appendix B

C++ Programming code

Here it is attached the C++ programming code use to develop the tool. It is used many guides and online documentation. It is impossible to cite all of them but to rebuild this results, there are some forums\(^1\) where to ask for help for understanding some procedures.\(^2\)

B.1 Antenna Calibration part

Copyright. All rights reserved

B.2 Test Part

```cpp
#include <iostream>
#include <iomanip>
#include <fstream>

#include "Rinex3ObsBase.hpp"
#include "Rinex3ObsData.hpp"
#include "Rinex3ObsStream.hpp"

#include "SP3EphemerisStore.hpp"
#include "Position.hpp"
#include <numeric>

#include "Requirement.hpp"
#include "ComputeLinear.hpp"

#include "Decimate.hpp"
#include "Dumper.hpp"
#include "OneFreqCSDetector.hpp"

using boost::tuple;
using namespace std;
using namespace gpstk;
```

\(^1\)http://www.gpstk.org/bin/view/Forum/WebHome
\(^2\)http://www.gpstk.org/bin/view/Documentation/WebHome
typedef tuple<CivilTime, int, double> Tuple3;
typedef vector<Tuple3> vectorOfTuple3;
typedef tuple<CivilTime, int, double> Tuple4;
typedef vector<Tuple4> vectorOfTuple4;

// Global variables
vectorOfTuple4 Multipath4;
vectorOfTuple4 Multipath3;
vectorOfTuple4 Multipath2;
vectorOfTuple4 Multipath1;
vectorOfTuple4 Multipath0;
vector ipairs, int loopCounter;

void saveStandardDeviation(int PRN);
void pushBackMultipathAsTupleOf4(vectorOfTuple3 MuMinusMeanMu, int loopCounter);

void computeMultipath(string obsFile, string SP3File1, string SP3File2, int SV, int iValue, int jValue, int i);

typedef tuple<string, 2>

int main()
{
    vector<tuple<string, 2> > dMultipath;
    vector<tuple<string, 2> > dAntEl;
    vector<tuple<string, 2> > dAntEl);
    vector<tuple<string, 2> > dAntElCA3;

    for (int i = 0; i < numberObsFiles; i++)
    {
        cout << "obsFile = " << obsFile << endl;
        cout << "AntennaCalibration = " << AntennaCalibrationISLK.txt << endl;
        cout << "StandardDeviationISLK = " << StandardDeviationISLK.txt << endl;
        cout << "AntennaCalibrationOZST = " << AntennaCalibrationOZST.txt << endl;
        cout << "StandardDeviationOZST = " << StandardDeviationOZST.txt << endl;
        cout << "AntennaCalibrationCAT3 = " << AntennaCalibrationCAT3.txt << endl;
        cout << "StandardDeviationCAT3 = " << StandardDeviationCAT3.txt << endl;
        cout << "dAntElISLK = " << dAntElISLK.txt << endl;
        cout << "dAntElOZST = " << dAntElOZST.txt << endl;
        cout << "dAntElCAT3 = " << dAntElCAT3.txt << endl;
    }

    // This is what you really can modify easily. Here you can set up the day when you want to test the tool.
    // Change the values for 92, 93, 94 etc. Make sure that you have the files in the current folder before.
    // The file array is the set of the antenna calibration files already computed. For default they are 5.
    array<string, numberObsFiles> dObsFiles = {
        "ISLK092A00.17O", "OZST092A00.17O", "P558092A00.17O", "P588092A00.17O", "CAT3092A00.17O"};
    // This array is linked with the array above. You must select the same day for the SP3 file and the observable index file.
    // Also, you must add the following day of the SP3 file because it is required by the GPS TK functions. At the end of each day,
    // they need the information of the satellites of the following day.
    array<string, 2> SP3Files = {"gm19450.sp3", "gm19451.sp3"};
}

int const numberObsFiles = 5;
int const iValue = 1, jValue = 5;

for (int i = 0; i < numberObsFiles; i++)
{
    cout << "obsFile = " << obsFile << endl;
    cout << "AntennaCalibration = " << AntennaCalibrationISLK.txt << endl;
    cout << "StandardDeviationISLK = " << StandardDeviationISLK.txt << endl;
    cout << "AntennaCalibrationOZST = " << AntennaCalibrationOZST.txt << endl;
    cout << "StandardDeviationOZST = " << StandardDeviationOZST.txt << endl;
    cout << "AntennaCalibrationCAT3 = " << AntennaCalibrationCAT3.txt << endl;
    cout << "StandardDeviationCAT3 = " << StandardDeviationCAT3.txt << endl;
    cout << "dAntElISLK = " << dAntElISLK.txt << endl;
    cout << "dAntElOZST = " << dAntElOZST.txt << endl;
    cout << "dAntElCAT3 = " << dAntElCAT3.txt << endl;
}

// This is what you really can modify easily. Here you can set up the day when you want to test the tool.
// Change the values for 92, 93, 94 etc. Make sure that you have the files in the current folder before.
// The file array is the set of the antenna calibration files already computed. For default they are 5.
Set flags to reject satellites with bad or absent positional values or clocks
EphList.rejectBadPositions(true);
EphList.rejectBadClocks(true);
EphList.loadFile(SP3File[0]);
EphList.loadFile(SP3File[1]);

vector<SatID> satList; // List with the available satellites of every constellation

TimeSystem SP3System = EphList.getTimeSystem();
cout << "SP3 File has a time System: " << SP3System.toString() << endl;

vector<int> PRNList; // List with the IDs of the available satellites in Galileo
for (int i = 0; i < satList.size(); i++)
{
    if (satList[i].system == SatID::systemGalileo)
    {
        PRNList.push_back(satList[i].id);
    }
}

// For getting the standard deviation from files and save them in global variables
for (int nObsFile = 0; nObsFile < numberObsFiles; nObsFile++)
{
    AntennaCalibration = getFromFile(AntCalFile[nObsFile]);
    switch (nObsFile) {
    case 0:
        for (int i = 0; i < AntennaCalibration.size(); i++)
        {
            AntennaCalibration0.push_back(AntennaCalibration[i]);
        }
        break;
    case 1:
        for (int i = 0; i < AntennaCalibration.size(); i++)
        {
            AntennaCalibration1.push_back(AntennaCalibration[i]);
        }
        break;
    case 2:
        for (int i = 0; i < AntennaCalibration.size(); i++)
        {
            AntennaCalibration2.push_back(AntennaCalibration[i]);
        }
        break;
    case 3:
        for (int i = 0; i < AntennaCalibration.size(); i++)
        {
            AntennaCalibration3.push_back(AntennaCalibration[i]);
        }
        break;
    case 4:
        for (int i = 0; i < AntennaCalibration.size(); i++)
        {
            AntennaCalibration4.push_back(AntennaCalibration[i]);
        }
        break;
    default:
        cout << "Error because the number of observable files is inappropriate" << endl;
        break;
    }
    cout << "Number of satellites: " << PRNList.size() << endl; // It should be 14
    int m = 0; // Start from 0
    while (m < PRNList.size())
    {
        int PRN = PRNList[m];
        m++;
        Multipath0.clear();
        Multipath1.clear();
        Multipath2.clear();
        Multipath3.clear();
        Multipath4.clear();
    }
    for (int nObsFile = 0; nObsFile < numberObsFiles; nObsFile++)
    {
        computeMultipath(obsFile[nObsFile], SP3File[0], SP3File[1], PRN, iValue, jValue, nObsFile);
        cout << "IMPORTANT TEST" << endl;
        cout << Multipath0.size() << " " << Multipath1.size() << " " << Multipath2.size() << " " << Multipath3.size() << " " << Multipath4.size() << endl;
    }
} // end of for (int j = 0; j < numberObsFiles; j++)

73
if ((multipath0.size() == 0) || (multipath1.size() == 0) || (multipath2.size() == 0) || (multipath3.size() == 0) || (multipath4.size() == 0))
    cout << "There is no available information for the PRN: " << PRN << endl;
    // myFile.open("ccc.txt",ios::app);
    // myFile << "Satellite: " << PRN << " has no available data" << endl;
    // myFile.close();
    continue;
}

saveStandardDeviation(PRN);

cout << "CHECKPOINT" << endl;
vector<tuple<CommonTime, double, double>> CCC = computeCCC();

myFile.open("ccc.txt", ios::app);
for (int i = 0; i < CCC.size(); i++) {
    cout << "Satellite: " << PRN << " has available data " << get<0>(CCC[i]) << endl;
    cout << "Common Time: " << get<1>(CCC[i]) << endl;
    cout << "Lt1: " << get<2>(CCC[i]) << endl;
    myFile.close();
}

// End of while (m < PRNlist.size())
return 0;
// End of main

vector<pair<int, double>>FromFile(string file) {
    char cFirst[100];
    char cSecond[100];
    int first;
    double second;
    vector<pair<int, double>> vectorOfPairs;
    ifstream myReadFile;
    myReadFile.open(file);
    if (myReadFile.is_open()) {
        while (!myReadFile.eof()) {
            myReadFile >> cFirst >> cSecond;
            first = stoi(cFirst);
            second = stod(cSecond);
            vectorOfPairs.push_back(make_pair(first, second));
        }
    }
    myReadFile.close();
    return vectorOfPairs;
};

void computeMultipath(string obsFile, string SP3File1, string SP3File2, int SV, int iValue, int jValue, int loopCounter) {
    SP3EphemerisStore EphList;
    // Set flags to reject satellites with bad or absent positional
    // values or clocks
    EphList.rejectBadPositions(true);
    EphList.rejectBadClocks(true);
    EphList.loadFile(SP3File1);
    EphList.loadFile(SP3File2);
    vector<SatID> satList;
    satList = EphList.getSatList(); // List with the available satellites of every constellation
    TimeSystem SP3System = EphList.getTimeSystem();
    cout << "SP3File has as time System: " << SP3System.asString() << endl;
    vector<int> PRNList; // List with the IDs of the available satellites in Galileo
    for (int i = 0; i < satList.size(); i++) {
        cout << "satList[" << i << "].system = SatID::systemGalileo{" << endl;
        if (satList[i].system == SatID::systemGalileo){
            if (satList[i].id == SV) {
                PRNList.push_back(satList[i].id);
            }
        }
    }
    double Lambda;
    TypeID pse;
    TypeID phase;
    TypeID LII;
    TypeID CSL;
    double Lambda2;
    TypeID pse2;
    TypeID phase2;
    TypeID LII2;
    TypeID CSL2;
    switch (iValue) {
        case 1: Lambda = C1PS / L1_FREQ_GPS;
            pse = TypeID::C1;
            phase = TypeID::L1;
            LII = TypeID::L1;
            break;
        case 2: Lambda = C2PS / L2_FREQ_GPS;
        // other cases...
    }
```cpp
switch (jValue) {
    case 1: Lambda = C\text{MPS} / L1,FREQ,\text{GPS};
            psei = TypeID::C1;
            phasei = TypeID::L1;
            LLIi = TypeID::LLI1;
            CSLi = TypeID::CSL1;
            break;
    case 2: Lambda = C\text{MPS} / L2,FREQ,\text{GPS};
            psei = TypeID::C2;
            phasei = TypeID::L2;
            LLIi = TypeID::LLI2;
            CSLi = TypeID::CSL2;
            break;
    case 5: Lambda = C\text{MPS} / L5,FREQ,\text{GAL};
            psei = TypeID::C5;
            phasei = TypeID::L5;
            LLIi = TypeID::LLI5;
            CSLi = TypeID::CSL5;
            break;
    case 6: Lambda = C\text{MPS} / L6,FREQ,\text{GAL};
            psei = TypeID::C6;
            phasei = TypeID::L6;
            LLIi = TypeID::LLI6;
            CSLi = TypeID::CSL6;
            break;
    case 7: Lambda = C\text{MPS} / L7,FREQ,\text{GAL};
            psei = TypeID::C7; //pseudorange
            phasei = TypeID::L7; //carrier phase
            LLIi = TypeID::LLI7; //Loss of Lock Indicator
            CSLi = TypeID::CSL7; //Cycle Slip Indicator
            break;
    case 8: Lambda = C\text{MPS} / L8,FREQ,\text{GAL};
            psei = TypeID::C8;
            phasei = TypeID::L8;
            LLIi = TypeID::LLI8;
            CSLi = TypeID::CSL8;
            break;
    default : cout << "Choose a correct frequency band" << endl;
              break;
}

//This is the GNSS data structure that will hold all the
//GNSS-related information
gnssRinex gRin;
// Declare an object to check that all required observables are present
RequireObservables requireObs (psei);
requireObs.addRequiredType(PhaseType);
requireObs.addRequiredType(PhaseType);
requireObs.addRequiredType(PhaseType);

// Objects to mark cycle slips
OneFreqCSDetector markCSi;
markCSi.setCodeType(psei);
markCSi.setPhaseType(phasei);
markCSi.setLLIType(LLIi); //Loss of Lock Indicator
markCSi.setResultType(CSLi);
```

75
OneFreqCSDetector markCSj;
markCSj.setCodeType(psei);
markCSj.setPhaseType(phasej);
markCSj.setLLIType(LLIj); // Loss of Lock Indicator
markCSj.setResultType(CSLj);

// This object defines several handy linear combinations
LinearCombinations comb;

// Object to compute linear combinations for cycle slip detection
ComputeLinear linear1(comb.pdeltaCombination);
linear1.addLinear(comb.ideltaCombination);
linear1.addLinear(comb.mwubbenaCombination);
linear1.addLinear(comb.lCombination);
vectorOfTuple3 vecMu;
vectorOfTuple3 MuMinusMeanMu;

try {
  int usefulDataCounter = 0;
  int m = 0;
  while (m < PRNList.size()) {
    int myprn = PRNList[m];

    Rinex3ObsStream roffs(obsFile);
    // Loop through epochs and process data for each.
    // --------------------------------------------------
    Dumper dumpObj;

    // Satellite identification: PRN and satellite system
    SatID pn(myprn, SatID::systemGalileo);
    double ele0 = 0;
    bool flag = false;
    bool flag2 = false;
    int Dele;
    int Dele0;
    double mu0;

    while (roffs >> gRin) {
      // Position recxyz = gRin.header.antennaPosition;
      TimeSystem timeSys = TimeSystem::GPS;
      CommonTime time = gRin.header.epoch;
      time.setTimeSystem(SP3System);
      // Let's use the CivilTime class to print time
      // cout << SP3System.asString() << " time " << time << " PRN: " << myprn << endl;
      CivilTime civtime(time);

      try {
        // The following lines are indeed just one line
        // Satellites without the specific required observations will be deleted
        gRin >> requireObs // Check if required observations are present
        >> linear1 // Compute linear combinations to detect CSL
        >> markCSi // Mark cycle slips : L1 algorithm
        >> markCSj; // Mark cycle slips : Melbourne-Wubbena
      } catch (Exceptions& e) {
        cerr << "Exception at epoch: " << time << ": " << e << endl;
        continue;
      } catch (...) {
        cerr << "Unknown exception at epoch: " << time << endl;
        continue;
      }
      // Get P1, P2 and L1 observations
      // Let's use the CivilTime class to print time
      try {
        double Li = gRin.body.getValue(pn, phasei);
        double Lj = gRin.body.getValue(pn, phasej);
        double Pi = gRin.body.getValue(pn, psei);
        if (Li == 0) || (Lj == 0) || (Pi == 0) {
          continue;
        }
        // Compute multipath
        // ---------------------
      }
    }
  }
}

76
double m_u = Pi \(-\) Li \(2\times\) \(\{\) Li \(-\) Lj \} / \(\{\) pow(Lambdai, 2) \(-\) pow(Lambdaj, 2)\} ;

usefulDataCounter++; // Adding 1 to the useful data counter

Position satPos = EphList.getPosition(prn, civtime); // If timeSys is Gal, one error, if it is GPS segmentation fault
double ele = recxyz.elevation(satPos);

/////////////////////////////////////////////////////////// OUTPUT TO TERMINAL /////////////
cout << civtime << " " <<
<< fixed << setw(7) << setprecision(3) << PRN << biasd_multiPath " << m_u <<
<< " Pi:" << Pi << " Li:" << Li << " Lj:" << Lj << " ele:" <<
ele << endl;

if (ele < 5) continue;
vecMu.push_back( boost::make_tuple( civtime , ele , m_u));

if ((abs(mu - mu0) > 10) {
    flag = false;
}
continue;

ofstream ff;
ff.open("MuMinusMean.txt",ios::app);
for(int i=0;i<MuMinusMeanMu.size();i++){
    cout << myprn << " " << get<0>(MuMinusMeanMu[i]) . convertToCommonTime() << " " << get<1>(MuMinusMeanMu[i]) << " " << get<2>(MuMinusMeanMu[i]) << " " << get<3>(MuMinusMeanMu[i]) << endl;
}
ff .close();
pushBackMultipathAsTupleOf4 (MuMinusMeanMu, loopCounter);
vecMu.clear();
}
//end of if Dele...

std::cout << "SatIDNotFound" << gRin;

MuMinusMeanMu = computeMuMinusMeanMu(vecMu);
ofstream ff;
ff.open("MuMinusMean.txt",ios::app);
for(int i=0;i<MuMinusMeanMu.size();i++){
    cout << myprn << " " << get<0>(MuMinusMeanMu[i]) . convertToCommonTime() << " " << get<1>(MuMinusMeanMu[i]) << " " << get<2>(MuMinusMeanMu[i]) << " " << get<3>(MuMinusMeanMu[i]) << endl;
}
ff .close();
pushBackMultipathAsTupleOf4 (MuMinusMeanMu, loopCounter);
vecMu.clear();

77
void Mu.clear();

cout << "Read " << roffs.recordNumber << " epochs of PRN: " << myprn << " with " << usefulDataCounter
cout epochs with useful data in the observable file: " << obsFile << endl;

M += // counter along prnList which exist in the SP3File
} // End of while
return;
} catch (Exception& e)
{
    cout << e;
    exit(-1);
}
} catch (...)
{
    cout << "Caught an unknown exception" << endl;
    exit(-1);
}
} // End of computeMultipath

// Function that computes the Mean of the Mu and the compute the Mu minus the mean for each elevation angle
vectorOfTuple3 computeMeanMu(vectorOfTuple3 MuEl)
double meanMu = 0;
for (int i = 0; i < MuEl.size(); i++)
    meanMu += get(MuEl[i].second);
meanMu /= MuEl.size();
vectorOfTuple3 MuMinusMean;

// cout << "Mean Multipath
for (int i = 0; i < MuMinusMean.size(); i++)

    // cout << get(MuMinusMean[i]) << " Mean Multipath
switch (loopCounter)
{
    case 0 :
    
        // cout << "nMultipath 0:n" << endl;
        for (int j = 0; j < AntennaCalibration0.size(); j++)
            if (AntennaCalibration0[j].first.first == get(MuMinusMean[j].second.first))
                Multipath0.push_back(boost::make_tuple(get(MuEl[j].first.first), get(MuMinusMean[j].second.first) - meanMu));
        // cout << get(MuMinusMean[j].second.second) << "n"
        return MuMinusMean;

        break;
    }
    
    case 1 :
    
        // cout << "nMultipath 1:n" << endl;
        for (int j = 0; j < AntennaCalibration1.size(); j++)
            if (AntennaCalibration1[j].first.first == get(MuMinusMean[j].second.first))
                Multipath1.push_back(boost::make_tuple(get(MuEl[j].first.first), get(MuMinusMean[j].second.first) - meanMu));
        // cout << get(MuMinusMean[j].second.second) << "n"
        return MuMinusMean;

        break;
    }
    
    case 2 :
    
        // cout << "nMultipath 2:n" << endl;
        for (int j = 0; j < AntennaCalibration2.size(); j++)
            if (AntennaCalibration2[j].first.first == get(MuMinusMean[j].second.first))
                Multipath2.push_back(boost::make_tuple(get(MuEl[j].first.first), get(MuMinusMean[j].second.first) - meanMu));
        // cout << get(MuMinusMean[j].second.second) << "n"
        return MuMinusMean;

        break;
    }
}
vector<tuple<CommonTime, double, double>> computeCCC()
{
    int c0=0, c1=0, c2=0, c3=0, c4=0;  // counters
    long double num;
    double occ;
    double stdDevCCC;
    double sumStdDev;
    vector<tuple<CommonTime, double, double>> AntennaCalibration0;
    vector<tuple<CommonTime, double, double>> AntennaCalibration1;
    vector<tuple<CommonTime, double, double>> AntennaCalibration2;
    vector<tuple<CommonTime, double, double>> AntennaCalibration3;
    vector<tuple<CommonTime, double, double>> AntennaCalibration4;
    int n;

    cout << "Multipath sizes:" << endl;
    cout << Multipath0.size() << " " << Multipath1.size() << " " << Multipath2.size() << " " << Multipath3.size() << " " << Multipath4.size() << endl;
    cout << "AntennaCalibration0.size()" << endl;
    cout << "AntennaCalibration1.size()" << endl;
    cout << "AntennaCalibration2.size()" << endl;
    cout << "AntennaCalibration3.size()" << endl;
    cout << "AntennaCalibration4.size()" << endl;
    if ( (AntennaCalibration0.size() == 0) || (AntennaCalibration1.size() == 0) || (AntennaCalibration2.size() == 0) || (AntennaCalibration3.size() == 0) || (AntennaCalibration4.size() == 0) )
    return OCC;
}

// Firstly, choose the first and the last time of the computation
CivilTime runningTime = max(std::max(get<0>(Multipath0[0]), get<0>(Multipath1[0]), get<0>(Multipath2[0]), get<0>(Multipath3[0]), get<0>(Multipath4[0])), get<0>(Multipath2[0]), get<0>(Multipath3[0]));
CivilTime lasttime = min(std::min(get<0>(Multipath0.end()-1), get<0>(Multipath1.end()-1), get<0>(Multipath2.end()-1), get<0>(Multipath3.end()-1), get<0>(Multipath4.end()-1));

while (runningTime < lasttime) {
    // This moves all the collected multipath data from every receiver to the same time
    while (get<0>(Multipath0[0]) < runningTime) c0++;
    while (get<0>(Multipath1[0]) < runningTime) c1++;
    while (get<0>(Multipath2[0]) < runningTime) c2++;
    while (get<0>(Multipath3[0]) < runningTime) c3++;
    while (get<0>(Multipath4[0]) < runningTime) c4++;
    //
    cout << get<0>(Multipath0[0]) << " " << get<0>(Multipath1[0]) << " " << get<0>(Multipath2[0]) << " " << get<0>(Multipath3[0]) << " " << get<0>(Multipath4[0]) << endl;
    if ( (get<0>(Multipath0[0]) == runningTime) && (get<0>(Multipath1[0]) == runningTime) && (get<0>(Multipath2[0]) == runningTime) && (get<0>(Multipath3[0]) == runningTime) && (get<0>(Multipath4[0]) == runningTime)) {
        array<Event, numberObsFiles>::stdDev = get<0>(Multipath0[0]) - get<0>(Multipath1[0]);
        array<Event, numberObsFiles>::stdDev = get<0>(Multipath0[0]) - get<0>(Multipath2[0]);
        array<Event, numberObsFiles>::stdDev = get<0>(Multipath0[0]) - get<0>(Multipath3[0]);
        array<Event, numberObsFiles>::stdDev = get<0>(Multipath0[0]) - get<0>(Multipath4[0]);
        cout << Multipath0[0] << " Multipath1[0]" << Multipath2[0] << Multipath3[0] << Multipath4[0] << endl;
        num = 0;
    }
invStdDev2.clear();
sumStDev=0;
for(int i=0; i<numberObsFiles; i++){
    invStdDev2.push_back(1./pow(stdDev[i],2));
    num += MultiPath[i]*invStdDev2[i];
    sumStDev += invStdDev2[i];
}

stDevCCC = 1./(stdDev.size())*sqrt(sumStDev); // stDevCCC = sqrt(pow(stDev[1],2)/stDev.size());
ccc = num*(accumulate(invStdDev2.begin(), invStdDev2.end(),0.0));
CCC.push_back(boost::make_tuple(runningTime.convertToCommonTime(), ccc, stDevCCC));
}

else{
    cout << get<0>(MultiPath0[i]) << " " << get<0>(MultiPath1[i]) << " " << get<0>(MultiPath2[i]) << " " << get<0>(MultiPath3[i]) << " " << get<0>(MultiPath4[i]) << endl;
}

CommonTime runningCommonTime = runningTime.convertToCommonTime();
runningCommonTime +=15;
runningTime.convertFromCommonTime(runningCommonTime);
}
}

return CCC;
}

void saveStandardDeviation(int PRN)
{
    ofstream myFile;
    myFile.open("standardDeviationLeveledMu0.txt",ios::app);
    for(int i=0;i<MultiPath0.size();i++){
        myFile<<PRN<<" "<<get<0>(MultiPath0[i])<<" "<<get<0>(MultiPath1[i])<<" "<<get<0>(MultiPath2[i])<<" "<<get<0>(MultiPath3[i])<<" "<<get<0>(MultiPath4[i])<<endl;
    }
    myFile.close();
    myFile.open("standardDeviationLeveledMu1.txt",ios::app);
    for(int i=0;i<MultiPath1.size();i++){
        myFile<<PRN<<" "<<get<0>(MultiPath1[i])<<" "<<get<0>(MultiPath2[i])<<" "<<get<0>(MultiPath3[i])<<" "<<get<0>(MultiPath4[i])<<endl;
    }
    myFile.close();
    myFile.open("standardDeviationLeveledMu2.txt",ios::app);
    for(int i=0;i<MultiPath2.size();i++){
        myFile<<PRN<<" "<<get<0>(MultiPath2[i])<<" "<<get<0>(MultiPath3[i])<<" "<<get<0>(MultiPath4[i])<<endl;
    }
    myFile.close();
    myFile.open("standardDeviationLeveledMu3.txt",ios::app);
    for(int i=0;i<MultiPath3.size();i++){
        myFile<<PRN<<" "<<get<0>(MultiPath3[i])<<" "<<get<0>(MultiPath4[i])<<endl;
    }
    myFile.close();
    myFile.open("standardDeviationLeveledMu4.txt",ios::app);
    for(int i=0;i<MultiPath4.size();i++){
        myFile<<PRN<<" "<<get<0>(MultiPath4[i])<<endl;
    }
    myFile.close();
}