Lire
la première partie
de la thèse
Annex A. GNSS signals

In this annex, a general view of the structure of a GNSS signal is presented and the architecture of a GNSS receiver is described. Nevertheless, before presenting the general structure of a GNSS signal, the minimum information data needed to provide positioning service is explained.

A.1. Minimum information data

The main fields needed to provide positioning service can be reduced to two different types of information. On one hand, the receiver-satellite distance; on the other hand, the satellite position with respect to Earth [MACABIAU and JULIEN, 2009] [SPILKER and ASHBYa, 1996].

A.1.1. Pseudo-range measurement

A receiver needs to know the distance between a GNSS satellite and itself. This distance is calculated from the signal propagation time and it is called pseudo-range [SPILKER and ASHBYb, 1996]. The pseudo-range is measured as the time offset between the signal received from the satellite at the antenna output and a local replica of this same signal generated by the receiver taking into account that the receiver ideally knows the time at which the satellite emitted the signal. The pseudo-range is the sum of the distance and the satellite-receiver clock offset. This means that the receiver has to take into account this lack of synchronization in order to correctly estimate the distance between the satellite and the receiver and thus in order to correctly estimate the user position. More specifically, the receiver and the satellite clocks should be both synchronized with the common GNSS general time but the reality is that neither of them is. Moreover, the lack of synchronization of each clock is processed differently due to the very different clock qualities.

In fact, the clock quality is much higher for a satellite clock, which is an atomic clock, than for a receiver clock which is not very accurate [SPILKER and ASHBYa, 1996]. Therefore, whereas the satellite clock bias is modeled as an error over the pseudo-range measurement, the receiver clock bias is so important and particular for each individual clock that the receiver is obliged to estimate the clock bias in order to remove its influence over the pseudo-range measurement.

The receiver cannot estimate its position with only one satellite pseudo-range, which means that the pseudo-range of several satellites should be used. Indeed, assuming a perfect synchronization between the satellites and the receiver clocks, with one satellite pseudo-range, the user position range is a sphere centered at the satellite position. With two satellite pseudo-ranges, the user position range is reduced to the common points (intersection) of the two satellite spheres, a circumference. And with three satellites pseudo-ranges, the user position range is reduced to only two points. Nevertheless, one of the two points is directly discarded because it is not placed on the Earth surface. Therefore, with three satellites a user should be able to estimate its position [SPILKER and ASHBYb, 1996]. See Figure A-1 for a graphical explanation.
A. GNSS signals

When assuming that the satellites and the receiver clocks are not synchronized, in order to obtain a 3-Dimensional user position, the receiver needs to measure 4 satellite pseudo-ranges; three satellites for the three user position components (x, y, z) and one for the user clock bias [SPILKER and ASHBYb, 1996].

Moreover, some additional information is required in order to correct pseudo-range measurement errors, such as the influence of the ionosphere, troposphere, etc and the previously commented satellite clock bias [MACABIAU and JULIEN, 2009].

A.1.2. Satellite ephemeris data

The receiver has to know the absolute position of the satellites, because, as it can be observed from Figure A-1, the estimated user position depends on the satellites position. Therefore, in order to obtain an absolute position instead of a relative one, the user has to know what the absolute positions of the satellites are.

A.1.3. Mathematical model of the estimation of the user position

Once the minimum information date needed to estimate a user position has been explained, the mathematical model which allows the user position estimation is given. This mathematical model relates the corrected pseudo-ranges measurements to the receiver position components and to the receiver clock bias [MACABIAU and JULIEN, 2009]:

\[
\begin{align*}
P^1(k) &= \sqrt{(x-x_1)^2 + (y-y_1)^2 + (z-z_1)^2} + c \cdot \Delta t_n(k) + b^1(k) \\
P^2(k) &= \sqrt{(x-x_2)^2 + (y-y_2)^2 + (z-z_2)^2} + c \cdot \Delta t_n(k) + b^2(k) \\
&\vdots \\
P^n(k) &= \sqrt{(x-x_n)^2 + (y-y_n)^2 + (z-z_n)^2} + c \cdot \Delta t_n(k) + b^n(k)
\end{align*}
\]  

(A-1)

Where:

- $P^i(k)$: pseudo-range measurement of satellite $i$ after applying the best corrections
- $(x, y, z)$: user coordinates
- $(x_i, y_i, z_i)$: satellite $i$ coordinates
• c: velocity of light in vacuum
• $\Delta t_u$: user clock bias
• $b'(k)$: noise plus multipath and additional residual errors after ionosphere, troposphere, satellite $i$ clock bias correction, etc.

Note that an increase of the number of measured pseudo-ranges leads to a more accurate estimation of the user position [MACABIAU and JULIEN, 2009]. Moreover, the accuracy of the estimation of the user position is higher for uniform distributions of the line-of-sights of the transmitting satellites [MACABIAU and JULIEN, 2009].

A.2. GNSS signal mathematical model

In this subsection, a general view of the GNSS signal structure is presented in order to better understand the signal processing discussed in the following subsections and along this dissertation. This general view consists in a general mathematical model and in the description of the GNSS signal components providing the necessary information data specified in annex A.1.

The presented mathematical model is a generic basis for any of the existing GNSS signals although, in order to exactly represent each particular GNSS signal, some specific variations have to be introduced to this mathematical model. Nevertheless, the exact mathematical expressions of each GNSS signal have been presented in 0.

A GNSS signal has three main components, the carrier frequency, the PRN code and the navigation data [MACABIAUa, 2009] [SPILKER and ASHBYc, 1996]. Moreover, the modern GNSS signals such as GPS L1C and GALILEO E1 have an additional component, a sub-carrier. The general mathematical model of an emitted GNSS signal is given next

$$r(t) = A \cdot d(t) \cdot c(t) \cdot s_c(t) \cos(2\pi f_0 t - \theta)$$

Where:

• A: signal amplitude
• $d(t)$: waveform representing the navigation data.
• $c(t)$: waveform representing the PRN code.
• $s_c(t)$: sub-carrier waveform
• $\theta$: signal carrier phase
• $f_0$: signal carrier frequency

First, the signal carrier frequency serves to locate the GNSS signal frequency content into its allocated frequency band. GNSS bands are situated in the L-band range frequencies.

Second, the PRN code or pseudo-random noise code is a spread spectrum code which has several functions. The first one is to spread the signal power density. The second one is to allow the transmission of several satellite signals into the same communication channel. This technique is called Code Division Multiple Access (CDMA) and is achieved by imposing that each satellite PRN code is orthogonal with any other satellite PRN code. The third function is to improve the signal resilience to the noise, to the multipath and to the interferences compared to a non-spread BPSK signal. The fourth and last function is to allow pseudo-range measurement by the receiver [SPILKER and ASHBYc, 1996], which is one of the two main
necessary information data presented in annex A.1. In order to allow the pseudo-range measurement, the PRN code autocorrelation function is designed to have a triangular form of a two-chip base width centered at $\tau=0$ and to be about 0 outside the triangle [MACABIAUa, 2009].

$$R_c(\tau) = \begin{cases} 
1 - \frac{|\tau|}{T_c} & |\tau| < T_c \\
0 & |\tau| > T_c
\end{cases} \quad (A-3)$$

Third, the navigation data carries the orbital parameters of the satellite broadcasting the signal with respect to the Earth and also carries the additional information necessary to calculate the pseudo-range measurement [SPILKER and ASHBYc, 1996]. Therefore, the navigation data provides the second information necessary to calculate the user position. Moreover, the navigation data also carries the satellites almanacs data, the information concerning the pseudo-range error correction, such as the ionosphere and satellite clock information, etc [SPILKER and ASHBYc, 1996].

Fourth and last, the subcarrier moves further the traditional BPSK power density spectrum around a central subcarrier frequency. Moreover, the introduction of the subcarrier is equivalent to a modification of the PRN code autocorrelation properties, where the central triangle width is reduced and additional triangles are created inside the initial two-chip base width.

Note that the mathematical model (A-3), when the subcarrier term is removed is a model of the autocorrelation of the GPS L1 C/A signal which was designed about 30 years ago [MACABIAUUb, 2009]. This signal, despite providing good positioning service performance, has shown to have some limitations in terms of demodulation, tracking and acquisition performance. Among other factors, these limitations are due to the data symbol duration since it bounds the maximum coherent integration time which can be used in the FLL, PLL and DLL [SPILKER and ASHBYyd, 1996]. The solution adopted by the modern GNSS signals is to transmit two signals instead of one. One signal contains the navigation data. The other signal is data free in order not to limit the FLL, PLL and DLL coherent integration time. The signal containing the data is called data channel and the signal not containing data is called dataless or pilot channel [ARINC, 2006].

The two signals are combined and transmitted together as one unique signal. Each GNSS signal is transmitted with a specific method. The signals can be transmitted in-phase [ARINC, 2006] [ESA, 2008], in phase-quadrature [ARINC, 2005] or they can be time multiplexed [ARINC, 2004]. The following equation presents the mathematical model of a signal with pilot and data channel transmitted in-phase.

$$r(t) = A \cdot [d(t) \cdot c_d(t) \cdot s_{cd}(t) + c_p(t) \cdot s_{cp}(t)] \cdot \cos(2\pi f_0 t - \theta) \quad (A-4)$$

Where:

- $c_d(t)$: data channel PRN code waveform
- $c_p(t)$: pilot channel PRN code waveform
- $s_{cd}(t)$: data channel sub-carrier waveform
- $s_{cp}(t)$: pilot channel sub-carrier waveform

Note that the pilot and the data channels have different PRN codes. The reason is that they need different orthogonal PRN codes in order to allow the receiver to separate them.
A.3. Architecture of a GNSS receiver

The general structure of GNSS receivers can differ from the general structure of traditional telecommunication receivers. Therefore for a better comprehension of the concepts and mathematical expressions given along this dissertation, the general structure of a GNSS receiver is presented in this subsection.

Moreover, since the receiver is only a part of the GNSS user segment, in this subsection, first the general structure of a GNSS user segment is presented and second the block diagram of the demodulation and signal carrier phase tracking processes are described.

A.3.1. General GNSS user segment block diagram

A GNSS user segment consists of five principal components: the antenna, the receiver, the navigator/receiver processor, the input/output (I/O) device such as a control display unit (CDU), and a power supply [KAPLAN and HEGARTY, 2006]. The block diagram is illustrated below:

![Figure A-2: Principal GNSS user segment components [KAPLAN and HEGARTY, 2006]](image)

The antenna is in charge of receiving the signal from the satellite. The receiver runs the three basic operations required to obtain the user position, the acquisition, the tracking and the demodulation. The navigator/receiver processor is in charge of controlling and commanding the receiver through its operational sequence, starting with the satellite signal acquisition and following by the satellite signal tracking and the satellite navigation message data demodulation. The I/O device is the interface between the GNSS and the user, and the power supply provides the power to the other user segment parts [KAPLAN and HEGARTY, 2006].
A more detailed block diagram of a generic GNSS SPS user segment is shown in Figure A-3.

In Figure A-3, it can be observed the separate processing performed for the different received satellite signals (marked as channels) and how the navigation/receiver processor command and control the processing on the different channels.

### A.3.2. Demodulation receiver block diagram

In this subsection, the block diagram of the demodulation elements is described. However, since each GNSS signal component has a different modulation, in this section, a standard signal without any sub-carrier and BPSK modulated is assumed. More specifically, the described demodulation receiver block diagram is for a GPS L1 C/A receiver, and thus, for any other GNSS signal, the block diagram should be modified.

In this example, the received signal is simplified as we only consider the data channel as in the GPS L1 C/A case. However, this simplification does not affect the demodulator scheme and its associated mathematical model since the pilot channel contribution on the demodulation process is negligible. This statement is justified because the data and the pilot PRN codes are orthogonal, and thus the pilot channel does not influence the data channel.
The block diagram of the demodulator is shown below.

![Block Diagram of the Demodulator](image)

Figure A-4: Block Diagram of the Demodulator [MACABIAUb, 2009]

The elements of the previous diagram block are commented below:

- \( r(t) \): received signal after the RF front-end processing
- \( \hat{\theta} \): estimated signal carrier phase
- \( \hat{\tau} \): estimated code delay
- \( T_D \): symbol duration
- \( r_i(t) \): correlator output signal
- \( r_i[k] \): numeric correlator output signal (after sampling)

The input signal after the front-end filter is assumed to be [MACABIAUb, 2009]:

\[
r(t) = A \cdot d(t-\tau) \cdot c_f(t-\tau) \cdot \cos(2\pi f_0 t - \theta) + b(t)
\]

(A-5)

Where:

- \( A \): signal amplitude
- \( d(t-\tau) \): navigation data
- \( c_f(t-\tau) \): PRN code after the RF front-end filtering
- \( \tau \): code delay due to the propagation time
- \( 0 \): signal carrier phase
- \( f_0 \): signal carrier frequency
- \( b(t) \): thermal noise after the RF front-end filtering

Therefore, the signal at correlator input is modeled as:

\[
r_i(t) = A \cdot d(t-\tau) \cdot c_f(t-\tau) \cdot c(t-\hat{\tau}) \cdot \cos(2\pi f_0 t - \theta) \cdot \cos(2\pi f_0 t - \hat{\theta}) + n(t)
\]

(A-6)

And the noise at the correlator input is likewise modeled as:

\[
n(t) = b(t) \cdot c(t-\hat{\tau}) \cdot \cos(2\pi f_0 t - \hat{\theta})
\]

(A-7)
Therefore, at the correlator output the signal and the noise are equal to:

\[
r_i[k] = \frac{A}{2} \cdot d[k] \cdot R_{cf}(\tau[k] - \hat{\tau}[k]) \cdot \cos(\hat{\theta}[k] - \hat{\theta}[k]) + n_i[k]
\]  \hspace{1cm} (A-8)

\[
n_i(t) = \frac{1}{T_D} \int_{t-kT_D}^{t} n(u)du
\]  \hspace{1cm} (A-9)

Where:

- \( R_{cf}(x) \): PRN code autocorrelation function after RF front-end filtering

And the noise power \( P_{ni} \) is modeled as:

\[
P_{ni} = \frac{N_0}{4 \cdot T_D}
\]  \hspace{1cm} (A-10)

Therefore, if a perfect estimation of the code delay and of the signal phase carrier is assumed, equation (A-8) is equivalent to:

\[
r_i[k] = \frac{A}{2} \cdot d[k] + n_i[k]
\]  \hspace{1cm} (A-11)

Finally, from \( r_i[k] \) the symbol estimation is immediately made if no channel code is implemented. But, if a channel code is implemented, \( r_i[k] \) is used to enter the detector block of Figure C-1.

**A.3.3. Signal carrier phase tracking receiver block diagram**

In this subsection, the block diagram of the tracking elements is described. More specifically, this tracking block diagram presents the traditional case where the received signal is tracked with a Phase Locked Loop (PLL) [KAPLAN and HEGARTYb, 2006].

The received signal model is simplified as done in the previous section. However, in this case, only the pilot channel is considered because the contribution of the data channel is negligible in comparison with the contribution of the pilot channel correlator output for the carrier phase tracking process. The justification is that the pilot and data PRN codes are orthogonal, and thus the data channel does not affect the pilot channel correlator output. Moreover, the signal sub-carrier is also removed for simplification purposes. The mathematical model of the signal after the front-end filter is presented below:

\[
r(t) = A \cdot c_j(t - \tau) \cos(2\pi f_c t - \theta) + b(t)
\]  \hspace{1cm} (A-12)

If the signal does not have a pilot channel, as for example the GPS L1 C/A signal, the signal used to track the carrier phase is the data channel.
A. GNSS signals

The PLL block diagram is presented below:

![PLL block diagram](image)

Where:
- \( r(k) \): sampled received signal after the RF front-end processing
- \( \hat{\theta} \): estimated signal carrier phase
- \( \hat{\tau} \): estimated code delay
- \( T_i \): coherent integration time
- \( I_p(n) \): Prompt I channel
- \( Q_p(n) \): Prompt Q channel
- \( V_e(n) \): Phase discriminator output
- \( V_c(n) \): DCO input

In Figure A-5, the output of the front-end block is directly a sampled signal. Therefore, all the blocks presented in Figure A-5 are digital blocks.

Another observation that can be made from the PLL block diagram is that since the signal phase is estimated, it cannot be assured that all the useful signal power is contained in the in-phase channel (I). Therefore, in order to use all the useful signal power during the signal carrier phase tracking process, the phase-quadrature channel (Q) is also employed. Nevertheless, note that if the signal carrier phase is well estimated, all the power should be in the I channel.
The mathematical model of the I and Q channels and their noises are [MACABIAUa, 2009]:

\[
I_p(n) = \frac{A}{2} \cdot R_{rf}(\tau - \hat{\tau}) \cdot \cos(\theta - \hat{\theta}) + n_I(n) \quad (A-13)
\]

\[
Q_p(n) = \frac{A}{2} \cdot R_{rf}(\tau - \hat{\tau}) \cdot \sin(\theta - \hat{\theta}) + n_Q(n) \quad (A-14)
\]

\[
n_I(n) = \frac{1}{T_i} \int_{\tau + kT_i}^{\tau} b(u) \cdot c(u - \hat{\tau}) \cdot \cos(2\pi f_0 u - \hat{\theta}) \, du \bigg|_{\tau = nT_i} \quad (A-15)
\]

\[
n_Q(n) = \frac{1}{T_i} \int_{\tau + kT_i}^{\tau} b(u) \cdot c(u - \hat{\tau}) \cdot \sin(2\pi f_0 u - \hat{\theta}) \, du \bigg|_{\tau = nT_i} \quad (A-16)
\]

The description of the blocks forming a PLL is presented next.

The discriminator block is the part of the PLL in charge of measuring the phase estimation error. Two main groups can be distinguished depending on their sensibility to the phase shifts introduced by the data bits [MACABIAUb, 2009].

The Dot Product, or Costas, and the ArcTangent discriminators are not sensitive to the phase changes introduced by the data bits [MACABIAUb, 2009]. These discriminators are used with the GPS L1 C/A signal because this signal does not have a dataless channel and thus, the carrier phase has to be estimated with the signal containing the data. Consequently, the PLL has to remove the phase of the information data (±π) before measuring the error between the received signal carrier phase and the local signal carrier phase.

The Q (or Coherent) and ArcTangent2 discriminators are sensitive to the phase changes introduced by the data bits [MACABIAUb, 2009]. These discriminators are affected by the sign of the data and thus force the PLL to make a change of π on the signal estimated phase each time that the data sign varies. Therefore, these discriminators can only be used on pilot (dataless) channels. This means that they can only be applied on the GPS L2C, GPS L5, GPS L1C and GALILEO E1 signals.

Moreover, the discriminators of the second group are normally used when a pilot channel exist because they have better tracking performance than the discriminators of the first group [JULIEN, 2005]. For example, the loss of lock threshold of the Q and ArcTangent2 discriminators is quite lower in terms of dB-Hz than the loss of lock threshold of the Dot product or the ArcTangent discriminators.

The PLL filter, F(z), has two main functions. First, the filter has to reduce the noise affecting the discriminator output. Second, the filter has to allow the PLL to track the signal phase dynamics [KAPLAN and HEGARTYb, 2006]. Therefore, in order to accomplish these functions, three main parameters are defined: the filter bandwidth (B_L), the filter order (k) and the integration time (T_i). First, in order to eliminate the maximum possible noise, the filter has to have a small bandwidth. However, at the same time, the bandwidth has to be large enough so that the filter does not distort the phase signal measurement. Second, the order of the filter has to be high enough to follow the dynamics of the phase signal but without imposing an excessively complex PLL. Third and last, the filter has to be adapted to the integration time (T_i). From these 3 parameters, the filter coefficients are defined [STEPHENS and THOMAS, 1995].
The DCO or Digitally controlled oscillator is equivalent to digital VCO, which is an electronic oscillator designed to be controlled in oscillation frequency by a voltage input, in this case a sampled input.

Finally two of the sources of error affecting the PLL performance are presented, the thermal noise and the dynamic stress error. Moreover, the definition of PLL loss of lock is given and the choice of the PLL discriminator used for the simulations conducted in this dissertation is justified.

A.3.3.1. PLL Thermal Noise

The error source called thermal noise results from the impossibility of removing all the narrow-band noise existent at the RF/IF output block. Its influence on the carrier phase estimation is explained next.

First, the I and Q channels are used as inputs to the PLL discriminator in order to obtain a measurement of the carrier signal phase estimation error. Second, the PLL discriminator output is filtered and, third and last, the filtered result is used as input to the DCO in order to generate the local signal carrier phase. Nevertheless, since the I and Q channels are corrupted by the RF/IF filtered AWG noise introduced by the transmission channel and the filter cannot remove all the existent noise on the PLL discriminator output, the DCO cannot generate the exact signal phase.

Once the theoretical explanation of the thermal noise source of error has been given, a numerical quantification is presented. It is demonstrated [MACABIAUa, 2009] that the signal carrier phase estimation when only the thermal noise presence is considered can be calculated as:

\[
\hat{\theta}(z) = H[z] \cdot \theta[z] + H[z] \cdot N_e[z]
\]  

(A-17)

Where:

- \( \hat{\theta} \): signal phase estimation
- \( \theta \): signal phase
- \( H[z] \): closed loop PLL transfer function
- \( N_e[z] \): Equivalent noise at the input of the closed loop

\[
N_e[z] = n \cdot N_{ne}[z]
\]  

(A-18)

Where:

- \( N_{ne}[z] \): Noise at the discriminator output
- \( n \): Real number which depends on the applied discriminator

From equation (A-17), it can be observed that the contribution of the thermal noise on the phase estimation error is modelled as a noise filtered by the PLL closed loop transfer function. Therefore, the general expression of the power of the carrier phase estimation error depends on the general structure of the PLL and on the thermal noise power. In fact, depending on the type of discriminator used the mathematical model of the carrier phase estimation error changes.
The mathematical expression of the variance of the carrier phase estimation error is particularized for each different discriminator. The error variance for the product discriminator also known as (generic) Costas or product discriminator is:

\[
\sigma^2_{\varepsilon} = \frac{B_L}{C/N_0} \left(1 + \frac{1}{2 \cdot T_I \cdot (C/N_0)}\right)
\]  

(A-19)

Where:
- \(\sigma^2_{\varepsilon}\): Variance of the carrier phase estimation error due to thermal noise
- \(B_L\): PLL filter Bandwidth
- \(T_I\): Coherent integration Time

The expression for the Q discriminator is:

\[
\sigma^2_{\varepsilon} = \frac{B_L}{C/N_0}
\]  

(A-20)

The expressions for the ArcTangent and ArcTangent2 discriminators are very difficult to obtain mathematically; however it has been shown through Monte Carlo simulations that their expression can be approximated by the same expression as the product one [SPILKER and ASHBYd, 1996].

To sum up, the phase estimation error due to the thermal noise at the PLL output is modelled as a noise with a power defined in equation (A-19) or (A-20), with flat power density spectrum bounded by the bandwidth of the closed PLL transfer function.

A.3.3.2. Dynamic stress error

The dynamic stress error is phase jitter caused by the permanent motion of the satellites and the possible receiver motion [IRSIGLER and EISSFELLER, 2002], or in other words, it is phase jitter caused by the signal dynamics. The incoming carrier phase or signal dynamics is modelled as [LEGRAND, 2002]:

\[
\theta(t) = \theta_0 + a_1(t) \cdot t + a_2(t) \cdot t^2 + a_3(t) \cdot t^3 + \ldots
\]  

(A-21)

Where:
- \(\theta(t)\): Incoming signal phase
- \(\theta_0\): Initial signal phase
- \(a_1(t)\): Radial satellite-receiver velocity.
- \(a_2(t)\): Radial satellite-receiver acceleration.
- \(a_3(t)\): Radial satellite-receiver jerk.

Along this dissertation, the phase jitter is reduced to a simple bias on the estimated signal carrier phase since the majority of signal dynamics can be perfectly tracked by the PLL filter. Indeed, the highest signal dynamics order that can be tracked by the PLL depends on the PLL filter order. This means that a PLL with a 3\(^{rd}\) order filter can track up to 2\(^{nd}\) order phase variations. We assume in this dissertation that the PLL has a 3\(^{rd}\) order filter, so the only signal
dynamics which can affect the signal carrier phase estimation are the satellite and/or receiver acceleration variations, and higher order dynamics. The acceleration variations, called jerk, are the highest order signal dynamics considered in this dissertation.

The mathematical model of the dynamic stress error of a 3rd order PLL is given below [STEPHENS and THOMAS, 1995]. This error represents in meters the bias of phase introduced by the signal dynamics in the generated local carrier phase.

\[ \theta_e = \frac{T_i^3}{K_3} \cdot \frac{dR}{dt^3} (m) \]  \hspace{1cm} (A-22)

Where:
- \( \theta_e \): Dynamic stress error
- \( R \): signal phase delay between the satellite and the receiver (in metres)
- \( T_i \): Integration Time
- \( K_3 \): coefficient given by [STEPHENS and THOMAS, 1995], in their description of discrete-update PLL.

This expression is reduced to a constant bias if the signal dynamics generating the dynamic stress error is a constant jerk. We select the worst possible and realistic jerk introduced by the signal dynamics. Therefore, the results obtained by this Ph.D. when this worst jerk value is assumed represent a lower-bound of the tracking and demodulation performance.

For a constant jerk, that is \( a_3(t) \), as the only signal dynamics causing the dynamic stress error, the previous expression is equal to:

\[ \frac{dR}{dt^3} = m \text{ jerks} \quad m \in \Re \]  \hspace{1cm} (A-23)

\[ 1 \text{ jerk} = 1 \text{ g/s} = 9.8 \text{ m/s}^3 \]  \hspace{1cm} (A-24)

\[ \theta_e = 2\pi \frac{T_i^3}{K_3} \frac{m \cdot g}{\lambda} (rad) \]  \hspace{1cm} (A-25)

Where:
- \( \theta_e \): Dynamic stress error
- \( m \): Constant number of jerks
- \( g \): gravity acceleration
- \( \lambda \): Wavelength of the carrier signal frequency
- \( T_i \): Integration Time
- \( K_3 \): coefficient given by [STEPHENS and THOMAS, 1995], in their description of discrete-update PLL.

Note that the final expression (A-25) is expressed in radians; therefore, in order to obtain equation (A-25) from equation (A-22), equation (A-22) is multiplied by \((2\pi \text{ radians}/1 \text{ } \lambda)\) resulting into the substitution of the meters by the radians.
A.3.3.3. Definition of the PLL loss of lock threshold

Each of one of the discriminator defined in annex A.3.3 measures the difference between the received signal carrier phase and the generated local carrier phase in a different way. However, this measurement is not ideal due to the different sources of error commented in section 3.1.3.2 and due to the discriminator itself. Indeed, the output of a discriminator is a perfect phase difference, or in its default, a value proportional to the phase difference. In doing so, the DCO will be allowed to generate a new local carrier phase which will compensate the previous measured difference [KAPLAN and HEGARTYb, 2006]. The proportional term of the measurement is called discriminator normalization factor (KD), and it has to be removed in order to allow the correct functioning of the DCO. The action to remove this term is called normalize the discriminator output.

However, the proportional measurement of the difference of phases is only possible when the discriminator works in its linear zone; in this zone, its output is proportional to its input. If this is not the case, the discriminator works outside its linear zone and the measurement is distorted. And, although using the discriminator at the edges of this linear zone is allowed but infrequent, an allowed maximum measurement distortion has been set in order to guarantee a minimum degree of accuracy. The linear zone range and the possibility to work outside this zone depend on the signal C/N0. Therefore, for a determined discriminator, there exists a C/N0 value under which the discriminator does not work in the linear zone, the maximum allowed measurement distortion is reached and thus the discriminator cannot properly measure the phase difference. This value is called the PLL loss of lock threshold since it is considered that under that value the local carrier phase generated by the PLL is no longer accurate enough.

A graphical example of the variation of the discriminator linear zone is presented in Figure A-6 [JULIEN, et al. 2005]. This figure shows the slopes of two discriminators, Arctangent and Q, assuming that the discriminator output has been normalized. It can be observed that whereas the arctangent linear zone varies as a function of the pilot channel C/N0, the Q linear zone remains constant. These curves have been calculated assuming the thermal noise presence only.

![Graph showing the variation of discriminator linear zone](image)

**Figure A-6:** Mean discriminator output (rad) as a function of the input phase error (rad). Left - Arctangent, Right – Q Discriminator. Coherent integration time equal = 4ms. Each of the curves represents a different value of pilot C/N0 [JULIEN, et al. 2005].

Therefore, from the determination of the linear zone (depends on the accepted degradation of linearity), from the determination of a maximum accepted percentage of phase errors that can
fall outside the linear zone and from the knowledge that the phase error variance of the thermal noise depends on the pilot channel $C/N_0$ value, it is possible to calculate the pilot channel $C/N_0$ value at which the PLL loses its lock.

More specifically, this maximum accepted percentage of phase errors that can fall outside the linear zone has been defined through the phase standard deviation error ($\sigma$). Therefore, it has been defined that the PLL loses its lock when $3\sigma$ is larger than half the linear zone range [KAPLAN and HEGARTYb, 2006]. In other words, assuming that the tracking error is Gaussian, it is theoretically defined that the PLL conserves its lock for a given $C/N_0$ (PLL threshold lock) 99% of the time (99% of the time the phase errors fall into the linear zone). Consequently, the pilot channel $C/N_0$ threshold is the pilot channel $C/N_0$ value at which half the discriminator linear zone is equal to $3\sigma$.

Therefore, the mathematical expression allowing the calculation of the PLL loss of lock threshold when the 4 sources of noise are taken into account is presented below. The mathematical expression is derived from equations (3-5) and (3-6) [KAPLAN and HEGARTYb, 2006].

$$3\sigma_\varphi + \theta_e \leq \frac{L_\varphi}{2}$$

(A-26)

Where:

- $\sigma_\varphi = \sqrt{\sigma_{th}^2 + \sigma_{vib}^2 + \sigma_{osc}^2}$
  - $\sigma_{th}$: Thermal noise standard deviation error
  - $\sigma_{vib}$: Oscillator vibrations standard deviation error
  - $\sigma_{osc}$: Allan deviation noise standard deviation error
- $\theta_e$: Dynamic stress error
- $L_\varphi$: Two-sided discriminator linear tracking region

From equation (A-26), the PLL loss of lock threshold can be directly calculated.

Table A-1 and Table A-2 [JULIEN, et al. 2005] summarize the threshold values for the different discriminators. These tables are calculated for a TCXO oscillator, for a level of signal dynamics jerk equal to 0 and 1 g/s, and for an effective $B_L$ the nearest to 10Hz as possible. The integration time is 20ms because all the signals have a pilot channel.

Note that the values found in the reference [JULIEN, et al. 2005] are expressed in pilot $C/N_0$ values. Therefore, these values have to be converted to total signal $C/N_0$ values at the receiver antenna output. The PLL loss of lock threshold expressed in total $C/N_0$ values is given below.

<table>
<thead>
<tr>
<th>Signal</th>
<th>GPS L2C</th>
<th>GPS L5</th>
<th>GPS L1C</th>
<th>GALILEO</th>
<th>$B_L$ (Hz)</th>
</tr>
</thead>
<tbody>
<tr>
<td>DOT Product</td>
<td>29</td>
<td>29</td>
<td>27.25</td>
<td>29</td>
<td>7-10</td>
</tr>
<tr>
<td>Q</td>
<td>23</td>
<td>23</td>
<td>21.25</td>
<td>23</td>
<td>5-6</td>
</tr>
<tr>
<td>Atan</td>
<td>27.5</td>
<td>27.5</td>
<td>25.75</td>
<td>27.5</td>
<td>4-30</td>
</tr>
<tr>
<td>Atan2</td>
<td>26</td>
<td>26</td>
<td>24.25</td>
<td>26</td>
<td>2-30</td>
</tr>
</tbody>
</table>

Table A-1: PLL tracking loss thresholds with a TCXO oscillator and a jerk = 0 g/s
The first observation to make is that the thresholds values of the Arctangent and Arctangent2 discriminators do not change from Table A-1 and Table A-2. However, this is not entirely true. In fact, each presented threshold is calculated from the $B_L$ value inside the specified range of values which provides the lower PLL loss of lock threshold. Therefore, the $B_L$ used for a determined discriminator is different for each analyzed jerk value. And these $B_L$ values can also be different for each different discriminator. Therefore, in reality, for a given $B_L$, the PLL loss of lock threshold values of the Arctangent and Arctangent2 discriminators increase when a jerk of 1 g/s is applied.

Another observation is that the product and Arctangent discriminators are only applied over the pilot channel although they could also be used on the data channel. This means that the additional data channel power could also be used to track the signal and thus the threshold should be lower. However, the coherent integration time should be adapted to each GNSS signal since its value should become equal to the signal symbol duration. Therefore, it is not possible to directly add the power of the data channel which has not been used to the pilot channel power in order to calculate the new PLL loss of lock threshold.

The last observation to make is that the thresholds presented in the Table A-2 are calculated assuming that the PLL is always working in the linear region. In fact the bias introduced by the dynamic stress error, expressed in equation (A-25), is only valid when the discriminator is in the linear region. However, there is the possibility that the change in the propagation time between the satellite and the receiver induces a phase change greater than the discriminator linear tracking range during one coherent integration period [JULIEN, et al. 2005]. In other words, the dynamic stress error can provoke by itself a loss of lock. Nevertheless, in this dissertation, it is assumed that the PLL receiver is well dimensioned and that the PLL never loses its lock due to a dynamic stress error bigger than expected. In fact, the assumed value of 1 g/s is very large and hardly a user will receive a signal with a larger jerk; therefore, when the jerk is 1 g/s, the only element which could cause a PLL loss of lock is the thermal noise.

Table A-1 and Table A-2 constitute a good reference of PLL tracking threshold values. Nevertheless, these values are calculated from the theoretical noise standard deviation expressions resulting from models where some assumptions were made that are not always fulfilled during the signal tracking process. For example, the $B_L$ used in their calculation is not the 10Hz adopted for this research work.
A. GNSS signals

A.3.3.4. PLL discriminator selection for the simulations

In this subsection, the PLL discriminator selected to conduct all the simulations of the dissertation is presented. This selection was made from the discriminator performance since one of the objectives of this dissertation is to find the performance bounds of the demodulation of the different GNSS signals. The chosen discriminator is the Q or coherent discriminator and the reasons for its selection are given next.

There are two main reasons which justify the Q discriminator selection. The first reason is that since the majority of the GNSS signals analyzed all along this dissertation have a pilot channel (a dataless channel), it is allowed the utilization of a discriminator which is sensitive to the phase jumps of \( \pi \) introduced by the data because there is no data in the pilot channel. Therefore, since these types of discriminators outperform the discriminators which are insensitive to the phase jumps of \( \pi \), either the Q or the Arctangent 2 discriminators are chosen. The second reason is that the Q discriminator has a better performance than the Arctangent2 discriminator. This justification is given next.

The main differences between the Costas and Q discriminator with respect to the Arctangent and Arctangent2 discriminators are the following. First, the discriminator normalization factor (\( K_D \)) of the product and Q discriminators is estimated from the I and Q channels, whereas for the Arctangent and Arctangent2 discriminators, \( K_D \) is always 1. Note that this estimation gets worse with low pilot channel \( C/N_0 \) levels. Second, whereas the two-sided discriminator linear region varies as a function of the pilot channel \( C/N_0 \) for the Arctangent and Arctangent2 discriminators, its value remains constant for the product and Q discriminators.

Therefore, from the previous differences, it is assumed that for low \( C/N_0 \) levels, the Costas and Q discriminators have a better performance than the Arctangent and Arctangent2 discriminators. The justification is given next.

For low \( C/N_0 \) values, the \( K_D \) estimation of the product and Q discriminators is quite affected by the thermal noise. Nevertheless, this estimation can be significantly improved when the \( K_D \) estimation is made using long periods of time. Obviously, this assumption is only valid when the signal \( C/N_0 \) does not vary significantly during the \( K_D \) estimation time. This assumption is valid for an AWGN channel.

For low \( C/N_0 \) values, the linear region of the Arctangent and Arctangent2 discriminators is quite reduced. This region length can be enlarged applying longer coherent integration times. However, the coherent integration time cannot be enlarged indefinitely due to the signal phase variations. In fact, the larger the coherent integration time is, the more the received signal carrier phase can vary between the coherent integration time beginning and the coherent integration time end. Indeed, since the PLL can only provide a linearly varying phase during a coherent integration time period, the integration time size is bounded. And this means that the improvement of the tracking performance of the Arctangent and Arctangent2 discriminators is limited by the coherent integration time.

Therefore, since the tracking performance of the ArcTangent2 discriminator is limited by the coherent integration time, and this time cannot be so easily enlarged as the time used to estimate the \( K_D \) term, which is the term limiting the tracking performance of the Q discriminator, the Q discriminator outperforms the ArcTangent2 discriminator from the tracking performance point of view in low \( C/N_0 \) transmission channels. This statement can be seen in Table A-1 in spite of the Table A-2 values. Note that the threshold values of Table A-1 and Table A-2 for the different discriminators are calculated for different \( B_L \) values.
Therefore, due to the limited improvement of the tracking performance of the Arctangent and Arctangent2 discriminators for low C/N₀ levels and the possibility to use long time periods for the estimation of the Kᵦ normalization factor of the products and Q discriminators, we assume that the product and Q discriminators have better tracking performance for low C/N₀ levels than the Arctangent and Arctangent2 discriminators. And thus, we choose to implement in the simulations of this dissertation the Q discriminator.
Annex B. Figures of merit

In this subsection, three figures of merit are defined. They are the SNR, the $E_b/N_0$ and the $C/N_0$ and they are used along the entire dissertation. Therefore a good understanding of what these figures represent, what are their units and what are their inter-relationships is necessary to correctly follow the analyses conducted along the dissertation.

B.1. Signal-to-noise ratio (SNR)

The signal-to-noise ratio expresses the relationship between the signal power and the noise power [ATIS, 2000].

\[
SNR = \frac{P_{signal}}{P_{noise}} \quad (B-1)
\]

More specifically, this figure of merit compares the level of a desired signal to the level of background noise. In other words, the SNR represents how much the noise distorts the useful signal. The higher the ratio, the less disturbing the background noise is. The SNR is normally expressed in dB [ATIS, 2000].

\[
SNR \ (dB) = 10 \cdot \log_{10} \left( \frac{P_{signal}}{P_{noise}} \right) \quad (B-2)
\]

Finally, this figure of merit is widely used in the telecommunications field.

B.2. Energy per bit to noise density ratio ($E_b/N_0$)

The $E_b/N_0$ is an important parameter in digital communication or data transmission. It is a normalized signal-to-noise ratio (SNR) figure, also known as the “SNR per bit” [ATIS, 2000].

In fact, the relationship between $E_b/N_0$ and SNR is easily calculated. Nevertheless, in order to establish this relationship, some other figures of merit have to be defined. First, the energy per symbol to noise density ratio ($E_s/N_0$) can be approximated as an equivalence to the signal-to-noise ratio (SNR), where a symbol is the physical representation of a bit or of a group of bits. Indeed, the bits are never directly transmitted through a channel since they belong to the digital domain whereas the channel belongs to the continuous one. Therefore, each bit or group of bits is assigned to a physical waveform in order to adapt the digital signal to its transmission through a channel. This physical waveform is called symbol [PROAKISb, 2001].

The following equation shows the equivalence between the $E_s/N_0$ and SNR, an equivalence which is only true when a matched filter is used to process the received signal [PROAKISc, 2001]. However, the use of the matched filter is the normal strategy followed by any receiver:

\[
\frac{E_s}{N_0} \approx \frac{E_s \cdot R_s}{N_0 \cdot B} = \frac{P_{signal}}{P_{noise}} = SNR \quad (B-3)
\]

Where:

- \( R_s \): Symbol transmission rate
- \( B \): Channel bandwidth
Note that the use of the matched filter sets the channel bandwidth equal to the symbol transmission rate [PROAKISc, 2001]. In fact, the channel bandwidth is the bandwidth of the noise where this noise is the signal noise after the signal processing (filters, etc).

Therefore, since it has been defined that a symbol represents a bit or a group of bits, we can directly relate the $E_s/N_0$ with the $E_b/N_0$:

$$\frac{E_b}{N_0} = \frac{E_s}{N_0} \cdot \frac{1}{b}$$ \hspace{1cm} (B-4)

Where:

- $b$: Number of bits represented by each symbol

Nevertheless, the bits transmitted through the channel are not the signal information bits. In fact, the information bits are usually encoded by a channel code. The definition of encoding and decoding is given in annex C.3 and C.4. Therefore, $E_s/N_0$ is normally related to the energy per coded bit to noise density ratio ($E_c/N_0$) when a channel code is applied on the message. Consequently, equation (B-4) may be expressed as:

$$\frac{E_c}{N_0} = \frac{E_s}{N_0} \cdot \frac{1}{b}$$ \hspace{1cm} (B-5)

Finally, the $E_c/N_0$ can be related to the $E_b/N_0$ using the code rate, $r$.

$$\frac{E_b}{N_0} = \frac{E_c}{N_0} \cdot \frac{1}{r}$$ \hspace{1cm} (B-6)

This code rate, $r$, determines the quantity $Y$ of coded bits needed to represent the quantity $X$ of information bits [PROAKISe, 2001].

$$r = \frac{X}{Y}$$ \hspace{1cm} (B-7)

To summarize, the final relationship between the SNR and the $E_b/N_0$ is expressed in equation (B-8) where the $E_b/N_0$ unit is dB [ATIS, 2000]:

$$\frac{E_b}{N_0} (dB) = \text{SNR}(dB) - 10 \cdot \log_{10}(b) - 10 \cdot \log_{10}(r)$$ \hspace{1cm} (B-8)

Note that, in case that the signal message does not have any channel code, the code rate is equal to 1 and thus the influence of the channel code is cancelled.

This figure of merit is also widely used in the telecommunication field.

**B.3. Carrier-to-noise density ratio ($C/N_0$)**

The $C/N_0$ is the ratio between the received carrier power and the receiver noise density [ATIS, 2000]. This figure of merit compares the level of a desired signal to the level of the background noise power density. The advantage of the $C/N_0$ in comparison to the SNR is that this former figure of merit does not take into account the channel bandwidth ($B$), and thus it is valid for any signal processing. Note that the signal characteristics and the signal processing inside the receiver determine the channel bandwidth. The relationship between the $C/N_0$ and the SNR is shown in equation (B-9).
Therefore, the $C/N_0$ is expressed in dB-Hz [ATIS, 2000]. Moreover, the $E_b/N_0$ and $C/N_0$ can be related as shown below:

$$\frac{C}{N_0} = SNR \cdot B$$  \hspace{1cm} (B-9)

Finally, this figure of merit is widely used in the satellite navigation field.
B. Figures of merit
Annex C. GNSS fundamental processes

In this annex, the definitions of the three main processes conducted by a GNSS receiver are described. These processes are the demodulation of the navigation message, the tracking of the signal and the acquisition of the signal. Additionally, since the demodulation of the navigation message normally implies the decoding of the channel code implemented over the message, the encoding and decoding process of a message are also described.

The descriptions given in this annex are generalized for any type of signal, not only the GNSS signals.

C.1. Signal Demodulation

The demodulation process of a digital communication of a signal consists in estimating the transmitted signal bits values from the antenna output received signal. In other words, the receiver has to decide from the received signal values which is the more probable bit being transmitted [PROAKISc, 2001].

The communication of a digital signal is achieved by transmitting either a waveform or a specific combination of waveforms, where each waveform or specific combination of waveforms represents either a bit or group of bits. This association between bits and waveforms is necessary in order to adapt the digital nature of the bits to the continuous nature of the channel [PROAKISc, 2001].

Therefore, due to the adaptation of the bits to the channel, the receiver must measure the received signal waveforms and/or the received signal waveform amplitudes in order to estimate which waveforms or waveform amplitudes have been transmitted. And from the estimation of the received signal waveforms and the received signal waveform amplitudes, the transmitted bits can be estimated since the assignment between waveforms and bits at the emission is known [PROAKISc, 2001]. This dissertation calls symbol the set of waveforms and/or the waveform amplitudes assigned to a bit or a group of bits. And the action of assigning a bit or group of bits to a determined signal waveform is called to modulate [PROAKISb, 2001].

Therefore, for an ideal channel, a channel that transmits the signal between the transmitter and the receiver without any kind of distortion or any kind of noise, the estimated symbols at the receiver are identical to the transmitted ones. This means that the transmitted bits are perfectly recovered at reception.

However, in reality, even the simplest channel, the AWGN channel, distorts the received signal after the communication. Therefore, the received signal waveforms and/or the received signal waveform amplitudes no longer correspond to a signal waveform and/or a signal waveform amplitude assigned to a bit or group of bits. Consequently, the receiver has to decide from the received signal waveforms or received signal waveform amplitudes which were the original transmitted symbols. The received signal waveforms or received signal waveform amplitudes are called the channel observations and play an important role into the demodulation of the signal [PROAKISc, 2001].

There are several criterions that can be used to decide which symbol has been transmitted and the most common is the Maximum a Priori (MAP) criterion [PROAKISc, 2001]. This criterion searches the probability of transmitting the symbol $s(t)$, associated to a bit or group
of bits, when the signal \( r(t) \) has been received. Equation (C-1) represents its mathematical expression:

\[
P(s_i(t)|r(t))
\]  
(C-1)

Where:
- \( P(x|y) \): Probability of \( x \) when \( y \) occurs

Therefore, the receiver decides that the symbol \( i \) has been transmitted if the symbol \( i \) has the largest MAP probability among all the possible symbols [PROAKISc, 2001]:

\[
s_i(t) \text{ transmitted} \iff P(s_i(t)|r(t)) > P(s_j(t)|r(t)) \quad j = 0..L-1, j \neq i
\]  
(C-2)

Where:
- \( L \): Number of symbols

However, the probability in (C-1) cannot be directly obtained and thus, this criterion is related to the Maximum Likelihood (ML) criterion by the Bayes theorem [PROAKISc, 2001]. The ML criterion searches the probability of receiving the signal \( r(t) \) when the symbol \( s_i(t) \) has been transmitted. This theorem has an equivalent expression for the probability density function.

\[
f(r(t)|s_i(t))
\]  
(C-3)

Where:
- \( f(x) \): probability density function of \( x \)

Therefore, the final MAP expression is [PROAKISc, 2001]:

\[
s_i(t) \text{ transmitted} \downarrow f(r(t)|s_i(t)) \cdot P(s_i(t)) > f(r(t)|s_j(t)) \cdot P(s_j(t)) \quad j = 0..L-1, j \neq i
\]  
(C-4)

In equation (C-4), it can be observed that apart from the observation of the channel, \( r(t) \), another factor influences the final symbol decision, the term \( P(s_i(t)) \). This factor is the a priori information of the signal communication, and altogether with the channel observation, they are the major actors of the symbol estimation process [PROAKISc, 2001].

Finally, one last intuitive concept can be extracted from equations (C-3) and (C-4). These equations show that the detected symbol will be the symbol which is the most similar to the received signal waveforms but weighted by the a priori information.

C.2. Demodulation Performance

The demodulation performance expresses how well a transmitted signal is demodulated: it indicates the quantity of errors made when demodulating the received signal [PROAKISc, 2001]. Obviously, the demodulation performance depends on the transmission channel, the employed demodulation technique, the signal characteristics, etc. All these factors define a scenario with fixed characteristics except for the signal power and the noise power. Therefore, the demodulation performance of a given scenario is given by the quantity of decision errors as a function of the signal power and the noise power [PROAKISc, 2001].
C. GNSS fundamental processes

The relationship between the signal power and the noise power is provided by the three different figures of merit already defined in 0: the SNR, the $E_b/N_0$ and the $C/N_0$. The $E_b/N_0$ is the most used figure in the telecommunications field. However, the $C/N_0$ is also used along this dissertation since it is the reference figure of merit in the satellite navigation field.

The quantity of errors made when demodulating the received signal can be expressed with three different parameters. Each parameter defines the percentage of errors of a different information unit. The parameters are the Bit Error Rate (BER), the Word Error Rate (WER) and the Ephemeris Error Rate (EER).

To sum up, the demodulation performance of the different GNSS signals analyzed in this study is expressed as the BER, the WER and the EER as a function of either the $E_b/N_0$ or the $C/N_0$.

The BER, the WER and the EER are detailed and their influence is commented below.

C.2.1. Bit Error Rate (BER)

The BER is equal to the number of wrong demodulated bits divided by the number of total transmitted bits [ATIS, 2000]. Therefore, the BER expresses the percentage of bits that would be wrong demodulated during the communication of a specific signal through a determined channel.

$$BER = \frac{Number\ of\ Erroneous\ Bits}{Total\ Number\ of\ Bits} \quad (C-5)$$

Besides, the BER can be calculated either over the information bits or over the coded bits. The most significant figure of merit is the BER of the information bits since the final user is only concerned by the information, and not on how this information is transmitted [PROAKIS, 2001]. Moreover, the BER of the information bits provides more information than the BER of the coded bits as detailed hereafter.

The BER of the coded bits shows the influence of the transmission channel and the influence of the signal adaptation to the channel (modulation, etc). Moreover, this BER indicates the influence of the energy per bit and, depending on the figure of merit associated to the BER ($C/N_0$ for example), the symbol transmission rate influence. But, in addition to all these influences, the BER of the information bits also shows the additional protection provided by the channel code implemented over the message.

To sum up, the BER of the information symbols provides the percentage of error on transmitted information bits, which is the only interesting value to the final user, and shows the improvement of the signal demodulation due to the introduction of a channel code. Remember that each GNSS signal has a distinct channel code.

Finally, the BER is a great indicator of the signal design quality in terms of demodulation performance. However, the BER does not specify the level of $E_b/N_0$ or $C/N_0$ necessary to demodulate a determined information message with a given percentage of success.

C.2.2. Word Error Rate (WER)

The information transmitted during a communication is not usually a random stream of bits, but rather a structured unit. In fact, all the bits representing an information field, such as the satellite inclination, are usually transmitted together. And, normally, the information fields are
grouped depending on their importance, on their type, etc, forming different units of information. These information units have been called packets or words along this dissertation and a different channel code can be implemented on each different word. Moreover, these words are grouped again forming new structures such as subframes or frames. Then, the process of structuring the information in words or subframes is applied over and over again until a periodic repetition of the structure information is achieved.

Additionally, each word usually carries some information which enables the receiver to detect whether any of the bits forming the word are wrong. Therefore, the receiver is usually able to know if the demodulated word information contains any error. This means that it is possible to measure the percentage of times that a word is correctly demodulated. And, even if a word did not carry any information allowing the verification of its integrity, it is always interesting to know the percentage of times that a word can be demodulated without error.

Therefore, the WER is a very interesting figure of merit in terms of percentage of times that an information field can be demodulated without errors. Moreover, the WER shows the influence of the word size because when a word is larger, a better channel code can be implemented. This statement was demonstrated by Shannon [PROAKISd, 2001].

More specifically, the WER is equal to the number of wrong demodulated words divided by the number of total transmitted words [ATIS, 2000].

\[
WER = \frac{Number \ of \ Erroneous \ Words}{Total \ Number \ of \ Words}\tag{C-6}
\]

Therefore, the WER expresses the percentage of words that would be wrong demodulated during the communication of a specific signal through a determined channel.

Note that the WER can be calculated for each different word transmitted by the signal. However, if the only difference among words is their position inside the signal structure, their WER is the same.

### C.2.3. Ephemeris Error Rate (EER)

During the description of the WER, it was said that the information fields are usually grouped into words. Moreover, it was commented that this regrouping is done among information fields providing the same type of information. This is the case of the different GNSS signals, where the information fields containing the satellite broadcasting ephemeris set or clock error corrections are transmitted into the same words. The entire ephemeris data set can be divided into several words as it is done for GPS L1 C/A, GPS L2C [ARINC, 2004], GPS L5 [ARINC, 2005] and GALILEO E1 [ESA, 2008] signals, or all the information can be grouped into only one word as it is done for GPS L1C [ARINC, 2006].

In any case, since the ephemeris and clock data are the only information necessary to obtain the satellite position and thus the final user position, the only important figure of merit from the point of view of the final user is the EER.

Consequently, in this dissertation, the EER is defined as the percentage of times that a receiver can correctly demodulate the signal ephemeris data set. And the EER is calculated by dividing the number of wrong demodulated ephemeris data sets by the number of total transmitted ephemeris data sets.
C. GNSS fundamental processes

C.1. Data Ephemeris

Sets DataEphemeris of Number Total
Sets DataEphemeris Erroneous of Number WER

\[
WER = \frac{\text{Number of Erroneous Ephemeris Data Sets}}{\text{Total Number of Ephemeris Data Sets}} \quad (C-7)
\]

Note that the Ephemeris Error Rate and the Word Error Rate are equal for the GNSS signals transmitting the ephemeris set and clock data into only one word. Therefore, for the signals transmitting the satellite ephemeris data set into more than one word the only interest of the WER is the following: to compare the value of either \(E_b/N_0\) or \(C/N_0\) necessary to obtain a given WER value with the value of either \(E_b/N_0\) or \(C/N_0\) necessary to obtain a EER value equal to the WER value. From this comparison, it can be concluded whether the ephemeris data set division into several words has improved the signal demodulation performance.

C.3. Signal Encoding

The encoding of the signal message is the application of a channel code over the signal message. This process consists in introducing, in a controlled manner, some redundancy in the binary information sequence of the signal. This redundancy can be used at the receiver level to overcome the effects of noise and interference encountered in the transmission of the signal through the channel [PROAKISd, 2001]. In other words, from the information bits point of view, the encoding process generates some new bits which represent the same content as the information bits. Therefore, these new bits are redundant since the original information content is already transmitted by the information bits. And the inclusion of these new binary bits into the transmitted word means that if some of the original information bits are corrupted or lost at the reception, the receiver will still have the possibility to recover the message since the lost information could be reconstructed from these new generated bits.

Moreover, in addition to generating the new bits and introducing them to the transmitted word, the information bits can be modified during the encoding process. Therefore, the encoding process can be seen as a generation of a new binary word from the original binary information word, where the original word can be preserved or not. These bits of the new binary word are called coded bits and the new binary word is called coded word [PROAKISd, 2001]. Besides, note that the coded binary word is the binary word used to modulate the signal. Therefore, when a channel code is applied, equation (B-5) is used instead of equation (B-4).

Additionally, the ratio between the size of the original information word and the size of the coded word is called code rate (r). Its expression has already been given in equation (B-6).

C.4. Signal Decoding

The decoding of the received signal message is the action of exploiting the available redundancy contained into the received message in order to correct the perturbations introduced by the channel [PROAKISd, 2001]. Therefore, due to this correction, the receiver can try to recover the original binary information word.

Nevertheless, there are two types of decoding process depending on the decoder block input. In fact, the received signal passes through a demodulator and a detector before entering the decoder [PROAKISd, 2001], as depicted below.
The demodulator block processes the received channel-corrupted waveform and reduces each waveform to a scalar or to a vector that represents an estimate of the transmitted data symbol. The detector block quantizes the output of the demodulator block and from this quantization the two possible decoding types are defined [PROAKISd, 2001].

If the number of quantization steps is equal to the number of possible symbols, the detection process has already chosen the symbol and thus has set the binary value of the coded bits. Therefore, since these coded bits are directly fed to the decoder block, this type of decoding is called hard decision. If the level of quantization is infinite, or much larger than the number of symbols, the value of the decoded bits is not binary but real. Therefore, this type of decoding is called soft decision [PROAKISd, 2001].

C.4.1. Strategies of channel codes use

There are two different strategies of channel code use. The first one is called Forward Error Correction (FEC) and consists in using the redundancy transmitted into the signal message in order to correct any possible errors introduced by the channel [PROAKISe, 2001].

The other use is called Automatic Repeat-Request (ARQ). It consists in detecting if any error has been introduced by the transmission channel into the received signal and in asking the transmitter to resend the information if some error was detected [PETERSON and DAVIE, 2003]. Along this Ph.D. manuscript, the action of using the word channel code to detect if the word contains any erroneous bit has been called verification, and the verification process is called verify the channel code. Therefore, if a word fails the channel code verification, it means that the word contain erroneous bits. And, if a word passes the channel code verification, it means that the word is error free.

In satellite navigation systems, the ARQ is not possible since the transmitters are only a few satellites whereas the number of receivers is much bigger. Therefore, a single receiver cannot request the retransmission of the navigation message. Nevertheless, the ephemeris data set is continuously repeated during a period of time, and when another set replaces the previous one, the user no longer needs the first ephemeris data set. Therefore, even if the user does not correctly decode an ephemeris data set, a retransmission request is unnecessary since the needed information is continuously broadcasted by the satellite [SPILKER and ASHBYb, 1996]. However, even if the ARQ strategy is not implemented, each GNSS signal has a channel code allowing the verification of the received words as seen in 0. And normally, if the word verification fails, the word is discarded.

Finally, in addition to the channel code used to detect whether the word is error free or not, the majority of GNSS signals have another channel code used to implement the FEC as it has also been seen in 0.
C. GNSS fundamental processes

C.5. Decoding Performance

Two different decoding performance can be analyzed, one for the FEC strategy and the other for the ARQ strategy.

The decoding performance of the ARQ mode consists in measuring the number of erroneous words which verify the implemented channel code in relation to the total number of received words [PETERSON and DAVIE, 2003].

\[ ARQ\ Perfo = \frac{Wrong\ words\ passing\ the\ code\ verification}{Total\ Words} \]  

This decoding performance is ignored in this dissertation mainly since the performance of the channel code implemented on the GNSS signals has already been analyzed in the literature.

The decoding performance of the FEC mode indicates how well the received signal is decoded [PROAKISE, 2001]. Therefore, this performance is represented with the BER, the WER and/or the EER of the received signal as a function of the \( E_b/N_0 \) [PROAKISE, 2001].

In order to completely analyze the decoding performance, the BER of the signal when a channel code is implemented has to be compared with the BER of the signal when no channel code is implemented [PROAKISE, 2001]. More specifically, from the plot of the two BER curves, the gain of \( E_b/N_0 \) for the signal implementing a channel code to reach the same BER value as the signal not implementing a channel code can be quantified. Note that this comparison only analyzes the improvement introduced by the code and does not reflect any other signal characteristic.

Finally, the decoding performance of the FEC mode can also be expressed as a function of the \( C/N_0 \). However, in this case, the performance not only shows the influence of the channel code, but also the symbol transmission rate, etc. Therefore, the decoding performance of the FEC mode when expressed as a function of the \( C/N_0 \) is considered equivalent to the demodulation performance all along this dissertation.

C.6. Signal Carrier Tracking

The carrier tracking process of a digital signal consists in estimating the instantaneous carrier frequency and the carrier phase of the received signal [KAPLAN and HEGARTYb, 2006]. There are two main methods employed to achieve this estimation, and both methods have been used along this dissertation. The first one consists in estimating the carrier frequency and carrier phase signal using a FLL and a PLL [PROAKISf, 2001], and the second one consists in estimating the transmission channel [PROAKISg, 2001]. Nevertheless, the satellite navigation systems traditionally use FLLs and PLLs [KAPLAN and HEGARTYb, 2006]. Therefore this method is more widely presented and studied along this dissertation than the channel estimation technique. A brief explanation of both methods is given below.

The FLL and PLL methods consist in the following. The FLL (Frequency Locked Loop) has to find the signal carrier frequency in order to down-convert the signal to baseband, and the PLL (Phase Locked Loop) has to find and subtract the instantaneous carrier phase of the signal in order to recover the maximum useful signal power [PROAKISf, 2001].

Nevertheless, in this thesis, the influence of the carrier tracking process is reduced to the study of the PLL. The reason is that the FLL is able to recover the signal carrier frequency with a sufficient accuracy to neglect its impact over the demodulation performance for the analysed levels of \( C/N_0 \) of this dissertation [MACABIAU et al, 2003]. However, the phase tracking errors must be taken into account.
The channel estimation consists first in estimating the phase and the amplitude introduced by the channel on the received signal. Second, once the estimation is made, the method consists in equalizing these introduced phase and amplitude in order to maximize the received signal SNR [PROAKISg, 2001]. Different techniques with different performance are used to estimate the signal phase and amplitude.

The different performance of phase tracking methods is presented in the following subsections.

C.6.1. PLL tracking performance

The performance of the PLL in the satellite navigation domain is measured by three different parameters: the PLL tracking loss threshold, the tracking accuracy and the cycle slip rate [KAPLAN and HEGARTYb, 2006].

The first one is the PLL tracking loss of lock threshold and it is expressed in terms of $C/N_0$. This threshold indicates the level of $C/N_0$ at which the PLL is no longer able to estimate the signal phase. This parameter is detailed in subsection annex A.3.3.3.

The second parameter defining the PLL tracking performance is the tracking precision: the bias and the variance of the carrier phase estimation error. This error is introduced by several noise sources which are presented in subsection 3.1.3.2.

The third parameter is the cycle slip rate. The cycle slip rate is the percentages of times that a phase cycle slip occurs during a second. However, this part of the tracking performance, although it affects the demodulation performance, does not have any relationship with the objective of this research work and the conducted studies. Therefore, this parameter is not analyzed and is not commented along this dissertation.

Finally, the influence of the PLL tracking performance is analyzed along this dissertation through its impact on the demodulation performance since the demodulation performance study is the main objective of this research work.

C.6.2. Channel estimation performance

The channel estimation performance is measured with one of the PLL tracking performance parameters. This parameter is the phase accuracy, which is represented by the bias and variance of the signal carrier phase estimation error and of the signal amplitude estimation error. The bias and the variance are presented as a function of the $C/N_0$. Nevertheless, in this dissertation, the channel estimation performance is also shown through its impact on the demodulation performance since the demodulation performance study is the main objective of this research work.

C.7. Signal Acquisition

The acquisition process of a GNSS signal is a detection process carried out as a search. This search process consists in finding at which frequency and at which code delay the GNSS signal is being received [KAPLAN and HEGARTYb, 2006].

In order to do so, the frequency domain is divided into bins and the same is done for the code delay domain: the combination of one code bin and of one frequency or Doppler bin is called a cell. Therefore, the acquisition process consists in detecting which cell contains the received signal carrier frequency and code delay. And thus, for each cell a signal replica with the
determined carrier frequency and code delay is generated and used to calculate the correlation with the received signal. Finally, the decision of which cell represents better the signal carrier frequency and code phase depends on the cell correlation results.

Several strategies can be used to select the cell. The receiver can explore the entire two-dimensional domain and choose the cell having the biggest correlation output [O’DRISCOILL, 2007]. The receiver can also explore the two-dimensional domain beginning from an arbitrary cell and ending with the first cell having a correlation output larger than a given threshold [KAPLAN and HEGARTYb, 2006]. This cell is then tested for tracking. Figure C-2 illustrates a graphical scheme of the GPS L1 C/A acquisition process:

![Figure C-2: GPS L1 C/A acquisition process](image)

Once an overview of the acquisition process is presented, the frequency and time domain bin division is described next. The carrier frequency search domain is required because the received signal carrier frequency can be different from the expected nominal signal carrier frequency due to the Doppler Effect. Therefore, the frequency search bounds are determined by the maximum Doppler shift that can affect the received signal [KAPLAN and HEGARTYb, 2006]. The code delay search domain or time search domain is required because the receiver does not know a priori which is the received signal delay, and thus cannot know the received code delay. Consequently, the code delay domain bounds are all the possible code delays, e.g. 1023 code delays for GPS L1 C/A. Nevertheless, the number of code delay bins is not equal to the number of code chips and the reason is presented next.

Since the receiver can only obtain a correlation value for a determined delay and not for a determined period, each bin represents a point of time rather than an interval of time. In other words, the code delay division is in reality a sampling process of the time delay domain. Therefore, it can be that none of the sampled code delay points is exactly equal to the received signal code delay but rather that the received signal code delay is between two sampled code delay points [KAPLAN and HEGARTYb, 2006]. This means that for this latter case, the bins where the received signal has its largest power are the bins having the code delay nearest to the received signal code delay. Nevertheless, even for these best bins, the useful signal power is lower than the useful signal power when the code delay is perfectly estimated since the signal PRN code autocorrelation is not evaluated at its peak. Therefore, the sampling criterion of the code delay domain is defined so that, in the worst case, the useful signal power at the correlator output loses a given proportion of correlation power [BASTIDÉ, 2004], normally 2.5 dB. Note that since the code delay sampling depends on the PRN code autocorrelation
function and on the signal modulation, the GPS L1 C/A signal has a different code delay sampling than the GPS L1C signal.

Moreover, the same principle can be applied to the Doppler search, where the frequency sampling is set in order to guarantee a fixed loss of signal power at most, normally 0.9 dB [BASTIDE, 2004].

The code and frequency bounds defined in the previous paragraphs correspond to the case where no previous information is known. However, if for some reason the receiver already has some signal information, the search bounds are reduced to $3\sigma$ of the previous known frequency and code delay tracked values uncertainties. This can be the case where the receiver is trying to acquire the signal only moments ago after losing the PLL lock [KAPLAN and HEGARTYb, 2006].

Finally, one last important element of the acquisition process is the false alarm rate. A false alarm occurs whenever a cell not representing the received signal carrier frequency and code delay is selected as the cell containing the useful signal [KAPLAN and HEGARTYb, 2006]. Until now, the only considered source of error was the sampling limitation since the receiver cannot inspect an infinite number of different code delays for example. However, there are other error sources which can influence the cell selection. These sources are, for example, the thermal noise, the interferences and the signal dynamics. These sources of error can change the cell correlation level. This means that cells not containing the useful signal can have their amplitude level increased and cells containing the useful signal can have their amplitude level decreased. In both cases the final cell choice can be corrupted and thus a false alarm can appear. Therefore, due to these sources of errors, the acquisition process has to set a decision threshold on the correlation values to guarantee a given maximum level of false alarms. Moreover, note that due to this introduction of the decision, a cell containing the useful signal can be missed.

C.7.1. Acquisition performance

The acquisition performance is measured by the statistics of the time required to acquire the signal as a function of the $C/N_0$ at the antenna output. This time is calculated for a determined false alarm probability in order to compare different acquisition parameters which are the cell selection criteria and the acquisition techniques.

Note that the difference among the acquisition parameters is really translated into different detection probability values. This detection probability refers to the probability of choosing the cell which contains the effective signal carrier frequency and the effective code properly delay.

C.7.2. Acquisition of the GALILEO E1 signal

In this section, the GALILEO E1 acquisition parameters are presented. These parameters are defined in order to calculate the GALILEO E1 signal acquisition performance case where the receiver is in cold start and no a-priori information is available. Therefore, the entire PRN code and the full range of possible Doppler frequencies have to be explored. Moreover, in this section, the particular case where a part of the data channel is modulated with the CSK signaling technique is also presented. Additionally, the simulator used to calculate the acquisition performance is described.
C.7.2.1. Acquisition parameters of the GALILEO E1 signal

Four different parameters can define the acquisition search of the GALILEO E1 OS signal. These parameters are the total number of bins required to sweep an entire PRN code period and entire range of Doppler frequencies, the maximal allowed false alarm probability (Pfa), the acquisition threshold, and the number of bins inspected each non-coherent integration time.

C.7.2.1.1. Total number of bins

The total number of bins is the number of cells into which the two dimensional search of the signal (time/frequency) is divided in order to cover an entire PRN code chip period and the entire Doppler frequency spectrum. The total number of bins is selected in order to maximize the chances that the signal appears in at least one of the inspected bins.

The received signal is attenuated at the correlator output because the generated code local replica and the generated local carrier have a delay and a frequency fixed by each bin, which is different from the true values. Therefore, the division of the two dimensional search space must be made with a step in frequency and a step in time small enough to ensure that, in the correct bin, the difference between the code local replica and the received PRN code, and the difference between the local carrier and the signal carrier is small enough so that the signal loss of power does not exceed a given maximum attenuation. Note that, the more the signal is attenuated, the more the signal is masked by the noise, and thus, the more it is difficult to identify it.

The model of the received signal at the correlator output is [BASTIDE, 2004].

\[
I[k] = \frac{A}{2} \cdot \sin(\pi \Delta f T_p) \cdot \cos(\varepsilon_{\theta}[k]) \cdot R_{code,f}(\varepsilon_{\tau}[k]) + n_I[k] \tag{C-9}
\]

\[
Q[k] = \frac{A}{2} \cdot \sin(\pi \Delta f T_p) \cdot \sin(\varepsilon_{\theta}[k]) \cdot R_{code,f}(\varepsilon_{\tau}[k]) + n_Q[k] \tag{C-10}
\]

Where:

- \(I[k], Q[k]\): Channel I and channel Q signal respectively.
- \(A\): Useful signal amplitude at the ADC output
- \(\Delta f\): Frequency offset between the received and the local carriers
- \(\varepsilon_{\theta}\): Phase offset between the received and the local carriers
- \(\varepsilon_{\tau}\): Phase offset between the received and the local PRN codes
- \(R_{code,f}\): Cross-correlation between the filtered received code and the unfiltered local code
- \(T_p\): Coherent integration time
- \(n_I[k]\): I channel noise at epoch k
- \(n_Q[n]\): Q channel noise at epoch k

Note that this expression is valid for the pilot channel since a data term should be multiplied to the first term of equations (C-9) and (C-10) in order to obtain the data channel expression.
C. GNSS fundamental processes

From expressions (C-9) and (C-10), it can be seen that the signal evaluated on a bin has 3 different kinds of attenuations. The first attenuation is the sinc term and is related to the difference of frequency between the received signal carrier and the generated local carrier. The worst attenuation introduced by this term over the signal is found when the received signal falls exactly in the middle of two frequency bins. In this case, the attenuation is calculated by evaluating the term at half the length of the frequency domain step. The expression is given below:

\[
A_{freq\_max} = \left| \frac{\sin(\pi \cdot L_f \cdot Tp / 2)}{\pi \cdot L_f \cdot Tp / 2} \right| \]

(C-11)

Where:
- \( L_f \): Frequency domain step
- \( A_{freq\_max} \): Maximum attenuation caused by the frequency domain step

The second attenuation is the cosine and sine terms which depend on the phase offset, \( \epsilon_\theta \), between the received and the local carrier. Nevertheless, this factor disappears since the I and Q channels are powered by two and are added to form the acquisition test criterion. Therefore, the attenuation introduced by the phase is neglected.

The third attenuation is introduced by the cross-correlation between the filtered received code and the unfiltered local one. In this case, the worst attenuation is found following the same principle as the frequency attenuation one. This means that the worst attenuation is calculated by evaluating this term at half the length of the time domain step.

\[
A_{time\_max} = \left| R_{code,f} (L_t / 2) \right| \]

(C-12)

Where:
- \( L_t \): Time domain step
- \( A_{time\_max} \): Maximum attenuation caused by the time domain step

Once the attenuation expressions have been presented, the total number of bins can be calculated.

The total number of bins is found by multiplying 3 factors: the number of chips forming a PRN code period, the number of different frequency bins which have to be inspected in order to cover the maximum Doppler frequency, and the number of parts into which a chip is divided.

\[
Num\_total\_bins = Num\_chip \times Num\_freqs \times Num\_parts\_chip \]

(C-13)

The number of chips of each PRN code is defined by the signal itself [ESA, 2008]. In this case, each channel PRN code is formed by 4092 chips.

The number of different inspected frequencies can be calculated by dividing the total range of inspected frequencies by the length of the frequency domain step. The frequency range has to cover the entire Doppler frequency spectrum at which the signal can be received. Therefore, the acquisition has to sweep from the nominal frequency minus the maximal Doppler frequency to the nominal frequency plus the maximal Doppler frequency. In this case and for a cold start, the Doppler frequency range is set to +/- 1.5 kHz. The length of the frequency domain step is calculated so that the maximum allowed frequency attenuation is equal to 0.9 dB (\( A_{freq\_max} = 0.8128 \)). In this case, the frequency step is equal to 125Hz.
Finally, the number of parts into which a chip is divided is found by imposing that the maximum allowed time attenuation is equal to 0.75 ($A_{\text{time, max}} = 0.75$). Therefore, in order to find the time step length, the autocorrelation of both channels have to be presented. The expressions of GALILEO E1 OS pilot and data channels autocorrelations before taking into account the front-end filter is shown below [JULIEN et al., 2006].

\[
R_{\text{pilot}}(\tau) = \frac{10}{11} \cdot R_X(\tau) + \frac{1}{11} \cdot R_X(\tau) - 2 \cdot \sqrt{\frac{10}{11}} R_{X/Y}(\tau) \quad (C-14)
\]

\[
R_{\text{pilot}}(\tau) = \frac{10}{11} \cdot R_X(\tau) + \frac{1}{11} \cdot R_X(\tau) + 2 \cdot \sqrt{\frac{10}{11}} R_{X/Y}(\tau) \quad (C-15)
\]

Where:
- $R_X(\tau)$: BOC(1, 1) autocorrelation function.
- $R_Y(\tau)$: BOC(6, 1) autocorrelation function.
- $R_{X/Y}(\tau)$: cross-correlation function between the BOC(6, 1) and BOC(1, 1) waveforms.

From expressions (C-14) and (C-15), and taking a front-end filter of 14 MHz of bandwidth double-sided, it is calculated that the length of the time domain step is about $(1/11) \cdot T_{\text{chip}}$. And this means that each chip is evaluated on 11 points.

Finally, multiplying the previous 3 factors, the number of total bins is equal to 1080288 bins.

C.7.2.1.2. False alarm probability

The false alarm probability is calculated in order to guarantee a maximum rate of 1 false alarm per code search space sweep. Therefore, the false alarm value is the inverse of the total number of bins.

\[
P_{fa} = 1/\text{Num_total_bins} \quad (C-16)
\]

The false alarm value is equal to $9.2568e-007$ for the previous number of total bins value. Finally, the time required to verify that the signal has been falsely acquired (TFAF) is set to 1 second.

C.7.2.1.3. Acquisition threshold

The acquisition threshold is computed assuming that the useful signal is not present and that only thermal noise contributes to the total signal power. This means that the acquisition threshold value has to be calculated in order to guarantee that the presence of thermal noise only is interpreted as the presence of useful signal with a probability equal to the false alarm probability.

Therefore, in order to calculate the acquisition threshold, the noise probability density has to be determined. This expression has already been calculated in [BASTIDE, 2004]; nevertheless, in this dissertation the noises appearing in each inspected bin are considered independent when they are not. However, for the analysis conducted in this dissertation, the [BASTIDE, 2004] expression is considered accurate enough for a preliminary study.

The noise to be compared to the acquisition threshold is the sum of squared Gaussian noises:
C. GNSS fundamental processes

\[ T_0 = \sum_{i=1}^{M} \left[ n_{I,\text{data}}^2 + n_{Q,\text{data}}^2 + n_{I,\text{pilot}}^2 + n_{Q,\text{pilot}}^2 \right] \]  \hspace{1cm} (C-17)

Where:
- \( T_0 \): Acquisition criteria without the useful signal presence
- \( M \): Number of non-coherent accumulations
- \( n_{I,\text{data}}, n_{Q,\text{data}}, n_{I,\text{pilot}}, n_{Q,\text{pilot}} \): Gaussian noises from the I and Q data channels and from the I and Q pilot channels respectively with \( N(0, \sigma_n^2) \).

Therefore, if the noise samples are considered independent, \( T_0/\sigma_n^2 \) has a chi-square distribution with \( 4 \cdot M \) degrees of freedom when the data and pilot channels are used to make the acquisition, and \( T_0/\sigma_n^2 \) has a chi-square distribution with \( 2 \cdot M \) degrees of freedom when only the pilot channels is used to make the acquisition.

Finally, the acquisition threshold \( (T_h) \) is calculated by imposing:

\[ P_{fa} = P(T_h > T_0) \text{ in presence of noise only} \]  \hspace{1cm} (C-18)

Where:
- \( P(X) \): Chi-square distribution with either \( 2 \cdot M \) or \( 4 \cdot M \) degrees of freedom.

C.7.2.1.4. Number of inspected bins each non-coherent integration time

The number of bins which are inspected each non-coherent integration time seconds is the main parameter which determines the total amount of time required to acquire the GALILEO E1 signal. More specifically, the receiver needs non-coherent integration time seconds to analyze a bin and to decide whether the useful signal is contained into the bin. Therefore, the acquisition search is limited by the non-coherent integration time and the number of bins which can be inspected at the same time.

On one hand, a long non-coherent integration time is necessary in order to reduce the noise influence over very noised signals, which means that this value is normally fixed. In this simulation, the coherent integration time is fixed to 4ms and the non-coherent integration time is fixed to 100ms, which implies a value of \( M \) equal to 25.

On the other hand, the number of bins inspected each non-coherent integration time depends on the number of available correlators and on the number of channels used to inspect the bin. More specifically, 4 correlators are necessary to inspected a bin when the data and pilot channels are used, 1 for the I data channel, 1 for the Q data channel, 1 for the I pilot channel and 1 for the Q pilot channel. And thus, only 2 correlators are used when the pilot channel is employed alone, 1 for the I pilot channel and 1 for the Q pilot channel.

To sum up, in this analysis the number of bins inspected each non-coherent integration time depends on the initial available number of correlators and on the selected acquisition option; if the signal is acquired using the pilot and data channels or if the signal is acquired using only the pilot channel.
C.7.2.2. Acquisition differences with a signal implementing CSK

The introduction of the CSK technique implies that two consecutives symbols carry data channel PRN codes with a different delay. In other words, each symbol carries a different shift of the data channel PRN code compared to the symbols transmitted before and after it. This means that a blind non-coherent integration technique cannot be used over the data channel since all the added coherent integrations can have different code shifted versions. Therefore, if a navigation signal implements the CSK technique over the entire data channel, this channel cannot be used to acquire the signal, since the non-coherent integration is necessary in order to acquire the signal without too high $C/N_0$ values.

However, the use of the data channel can be considered when the CSK technique is implemented over only a part of the channel. Indeed, it has been shown that the use of pilot and data channels altogether enhance the acquisition performance [BASTIDE, 2004]. However, there are three main problems in using both channels to acquire the signal when the data channel has implemented the CSK technique.

First, the acquisition threshold calculated when using both channels is higher than the acquisition threshold calculated when using only one channel since the number of noise samples for the former case is twice larger than in the latter case. However, the increase of this threshold is compensated by the useful signals of the pilot and data channel. Nevertheless, this cannot be true when the CSK is implemented on the data channel. In this case, when the CSK data channel is used to acquire the signal, only the pilot channel is providing useful signal although the receiver is using both channels. Therefore, since the threshold has been computed assuming the two channels case, the probability of acquiring the signal is low when the analyzed part of the data channel carries the CSK.

Second, since the signal is in process to be acquired, the receiver cannot know if the analyzed part of the data channel carries or does not carry the CSK signaling. In fact, if the receiver had this information, it could choose to change the acquisition threshold in order to adapt itself to the situation. However, this is not normally the case and thus the receiver has to use the threshold computed for the two channels. This is normally the bad option.

Third and last, the probability of false alarm is increased due to the CSK technique. Normally, for a navigation signal without implementing the CSK technique, only the thermal noise term appears into the inspected bins which do not contain the pilot PRN code. However, if the data channel signal implements the CSK signaling technique, in a inspected bin which does not contain the pilot channel PRN code, the correlation between the received circular shifted code version of the data channel PRN code and the generated replica can be evaluated at its peak and thus an additional term is added to the acquisition criteria $T_0$. Therefore, since the acquisition threshold is calculated assuming the presence of only thermal noise in the acquisition criteria $T_0$, the inspected bins where, in addition to the thermal noise terms, the correlation between the generated code replica and the received shifted version of the data channel PRN code contributes to the acquisition criteria $T_0$, have more probabilities of surpassing the threshold and thus have more probabilities to cause a false alarm.

A graphical example is shown in Figure C-3. In this figure, the green and blue areas represent the part of the correlation function between the generated local data channel PRN code replica and the received data channel PRN code which is not zero and thus produces undesired useful signal.
C. GNSS fundamental processes

Figure C-3: CSK acquisition undesired correlation between the data channel PRN and the generated local data channel PRN code replica

This study neglected what we call the border effects. In fact, as seen in Figure C-4, the different circular code shifts between two consecutive symbols create a new PRN code. In this case, the original autocorrelations properties of the PRN code are lost since the original data channel PRN code is correlated with a new created code. Therefore, it can no longer be assured that the correlation is about 0 when the codes are delayed by more than one chip.

Figure C-4: CSK acquisition border effect

To summarize, due to these different problems, the acquisition process for the GALILEO E1 OS signal proposed by TAS-France must be analyzed in order to examine how the acquisition performance is perturbed when using the pilot and data channel to acquire the signal.

C.7.2.3. Optimal CSK symbol shift source mapping

In this section, the optimal CSK symbol shift mapping source is presented. The optimal CSK symbol shift mapping source is the optimal association between code circular shifts and CSK symbols.

In order to make that choice, this analysis searches the circular codes shifts which provide the lowest level of false alarm probability; or, in other words, this analysis searches the lowest contribution of the correlation between a shifted data channel PRN code and the generated data channel PRN code replica to the acquisition criteria $T_0$. The assumptions taken into account are that the border effects are negligible and that the partial data channel PRN code correlations are about 0. Therefore, inspecting Figure C-3, two cases can be observed.

The first case is that the same two CSK symbols are transmitted consecutively, which means that the coherent integration of one PRN code period of the local replica achieves the maximal autocorrelation value. In fact, in this case, the transmission of two consecutive identical CSK symbols means that the original unshifted PRN code is found along the transmission of these two symbols. And note that, regardless of the selected circular shift, the correlation always reaches its maximum value which means that this case does not condition the CSK symbol shift mapping.
The second case is found when two consecutive transmitted CSK symbols are not equal. In this situation, it can be observed from Figure C-3 that two consecutive coherent integrations appear at the same time, where each one correlates a part of the useful signal depending on the transmitted code circular shift. Moreover, note that since no identical CSK symbols are transmitted consecutively, each individual coherent integration is only affected by one received CSK symbol. Therefore, the circular shift selection is done in order to minimize the useful signal contribution made by two consecutive coherent integrations. The function defining the contribution is shown below:

\[
f(n) = \left( \frac{N - n + 1}{N} \right)^2 + \left( \frac{n}{N} \right)^2
\]

Where:
- \( N \): Total number of chips forming the PRN code
- \( n \): Circular code shift

Searching the minimum of the previous function, the optimal circular code shift can be found. And this shift is equal to \((N+1)/2\), or equivalently, the code is shifted circularly half its length.

Finally, since the number of circular shifts has to be equal to the number of CSK symbols, the optimal shift mapping is obtained by associating each CSK symbol to the nearest available shift to \((N+1)/2\).

### C.7.2.4. Description of the simulator used to calculate the GALILEO E1 signal acquisition performance

The simulator implemented in order to analyze the GALILEO E1 OS signal performance is described next. The main principle of the simulator is explained first.

Each \( MT_p \) seconds, where \( T_p \) is the coherent integration time and \( M \) is the number of non-coherent accumulations, the simulator analyzes a number of bins determined by the number of correlators and by the acquisition configuration. Each bin is analyzed using the part of the signal received during its inspection. This means that the received signal used to inspect two bins which are analyzed in different \( MT_p \) periods of time is not the same. In fact, during \( MT_p \) seconds, the receiver is blocked analyzing a set of bins and cannot make any other action. But, once the inspection of the set of bins is finished, the receiver begins to inspect a new set of bins using the signal being received at the new epoch. This process is repeated until one inspected bin obtains a power value which is higher than the acquisition threshold. And, this means that the acquisition process can sweep several times the entire bin time/frequency domain before acquiring the signal or may not sweep the entire domain even once.

Once the fundamental principle is explained, a more detailed description of the simulator is given. First, the acquisition threshold is calculated from the assumed \( N_0 \) at the receiver antenna output and the number of channels used to acquire the signal, either the pilot and the data channel or only the pilot. This acquisition threshold is calculated by modeling the noise with a central chi-square function. Therefore, the value obtained has to be multiplied by the noise variance at the correlator output, \( \sigma_n^2 \), in order to obtain the real threshold value. For this simulation, the noise variance at the correlator output is calculated without taking into account the front-end filter.
Second, once the acquisition threshold is determined, the generation of the acquisition criteria is simulated. Normally, this process is simulated as a non-central chi-square function. Nevertheless, since in this simulation the noise correlation between consecutives bins is taken into account, another solution has to be used. The solution chosen is first to generate all the correlated noise samples of all the bins inspected during a non-coherent integration time. Then, we generate the useful signal correlator outputs associated to each bin. Note that since the received signal used to inspect the bins analyzed during different MT\textsubscript{P} periods of time is not the same, the noise of a bin can only be correlated with noises of bins analyzed during the same MT\textsubscript{P} period.

Third and last, the acquisition criterion is compared to the acquisition threshold in order to determine if the signal is detected. Therefore, depending on the acquisition criterion, on the acquisition threshold and on the true portion of the peak in the search space, the false alarm and detection counters are updated. Moreover, each time that a group of bins has been inspected, the acquisition time is increased by MT\textsubscript{P}. This time is also increased by TFAF seconds each time that the signal is acquired when it should not. Finally, if the acquisition is made, the simulation is ended.
Annex D. Advanced description of the GNSS signals

In this annex, a more detailed description of the GNSS signals analyzed in this research work is presented. Besides, a navigation message structure for the GALILEO E1 signal proposed by THALES ALENIA SPACE France is described and its demodulation performance is commented.

D.1. GPS L1 C/A signal structure

The GPS L1 C/A signal is contained within one band of 20.46 MHz centered on L1. The nominal carrier frequency \(f_0\) for L1 is 1575.42 MHz.

The spreading code of the GPS L1 C/A signal is a pseudo-random noise sequence used to spread the signal spectrum. The PRN code is different for each satellite and it allows the GPS L1 C/A receiver to differentiate among the different satellites transmitting at the same frequency carrier L1. Each L1 C/A PRN code has a duration of 1 ms at a chipping rate of 1023 kbps, meaning that each code has a length of 1023 chips.

Moreover, each PRN code waveform is labeled as \(G_i(t)\) and it is the physical materialization of a 1023 bit Gold Code. Therefore, each PRN code is generated from the modulo-2 sum of two 1023 bit m-sequences, \(G_1\) and \(G_2\). \(G_2\) is constructed by delaying the nominal sequence \(G_2\) by an integer number of chips. Finally, \(G_1\) and \(G_2\) m-sequences are generated by 10-stage shift registers.

D.2. GPS L2C signal structure

The GPS L2C signal is contained within one band of 20.46 MHz centered on L2. The nominal carrier frequency \(f_0\) for L2 is 1227.6 MHz.

Following the same logic as the GPS L1 C/A signal conception, the PRN codes of the GPS L2C signal, CL and CM, are different for each transmitting satellite and they are different from each other. This property allows the differentiation among satellites in addition to allow the separation between the part of the signal carrying the L2 CL code and the part of the signal carrying the L2 CM code. Therefore, due to the differentiation among satellites and the chip-by-chip time multiplex, the GPS L2C signal can be divided into two channels: the pilot channel, the part of the signal containing the L2 CL code, and the data channel, the part of the signal containing the L2 CM code plus the navigation data message.

This separation into two channels allows a better performance in terms of acquisition and tracking process since the accumulation time on the pilot channel is no longer restricted by the duration of a data symbol [JULIEN, 2005]. The power distribution between the two channels is 50% for the pilot channel and 50% for the data channel of the total amount of power.

The L2 CL code has a duration of 1.5 seconds at a chipping rate of 511.5 Kbps, meaning a length of 767250 chips. This code is generated with a modular-type shift register generator with a determined code generator polynomial [ARINC, 2004] and it is reinitialized after 767250 chips. Each different code \(CL_i(t)\) is created from a different configuration of the register’s initial state while having the same code generator polynomial.
The L2 CM code has a duration of 20 milliseconds at a chipping rate of 511.5 Kbps, meaning a length of 10230 chips. This code is generated in the same way as the CL code, using the same modular-type shift register generator and the same code generator polynomial, but it is reinitialized only after the generation of 10230 chips. Each different code CM\(_i\)(t) is created from a different configuration of the register’s initial state, and this configuration is different from any CL\(_i\)(t) code.

D.3. GPS L5 signal

The GPS L5 signal is contained within a band of 24 MHz centered on L5. The nominal carrier frequency (f\(_0\)) for L5 is 1176.45 MHz.

Following the same logic as the previous GPS signal conception, the PRN codes of the GPS L5 signal, I5-code and Q5-code, are different for each transmitting satellite and they are different from each other. This property allows the differentiation among satellites in addition to the separation between the part of the signal carrying the I5-code and the part of the signal carrying the Q5-code. Therefore, due to the differentiation among satellites and the in phase-quadrature modulation, the GPS L5 signal can be divided into two channels: the pilot channel, part of the signal containing the Q5-code, and the data channel, part of the signal containing the I5-code plus the navigation data message.

This separation into two channels allows a better performance in terms of acquisition and tracking process since the accumulation time on the pilot channel is no longer restricted by the duration of a data symbol [JULIEN, 2005]. The power distribution between the two channels is 50% for the pilot channel and 50% for the data channel of the total amount of power.

The I5-code has a duration of 1 millisecond at a chipping rate of 10.23 Mbps, meaning a length of 10230 chips. This code is generated by a 2-modulo sum of two extended patterns, XA and XBI\(_i\), also clocked at 10.23 Mbps. XA is a 8190 chip length code which is 1-bit short cycled before its natural period end, and XBI\(_i\) is a 8191 chip length code which is not short cycled. Each XBI\(_i\) code has a different initial condition but all of them are generated by the same generator polynomial.

The Q5-code has a duration of 1 millisecond at a chipping rate of 10.23 Mbps, meaning a length of 10230 chips. This code is generated by a 2-modulo sum of two extended patterns, XA and XBQ\(_i\), also clocked at 10.23 Mbps. XA is a 8190 chip length code which is 1-bit short cycled before its natural period end, and XBQ\(_i\) is an 8191 chip length code which is not short cycled. Each XBQ\(_i\) code has a different initial condition but all of them are generated by the same generator polynomial, which is also the XBI\(_i\) generator polynomial. The initial conditions of XBI\(_i\) and XBQ\(_i\) are different.

The synchronization sequence of the data channel is a 10-bit Neuman-Hofman code clocked at rate of 1 kHz, and thus has a duration equal to 10 ms, equal to 1 navigation data symbol. This code is synchronized with the data information symbols.

The synchronization sequence of the pilot channel is a 20-bit Neuman-Hofman code clocked at rate of 1 kHz, and thus has a duration equal to 20 ms, equal to 2 navigation data symbols. This code is synchronized with one out of two data information symbols.
D. Advanced description of the GNSS signals

D.4. GPS L1C signal

The GPS L1C signal is contained within a band of 30.69 MHz centered on L1. The nominal carrier frequency \( f_0 \) for L1 is 1575.42 MHz.

Following the same logic of the previous GPS signals conception, the PRN codes of the GPS L1 signal, L1C_P and L1C_D, are different for each transmitting satellite and they are different from each other. This property allows the differentiation among satellites in addition to the separation between the part of the signal carrying the L1 C_P code and the part of the signal carrying the L1 C_D code. Therefore, due to the differentiation among satellites and the differentiation among L1 codes, the GPS L1C signal can be divided into two channels: the pilot channel, part of the signal containing the C_P code, and the data channel, part of the signal containing the C_D code plus the navigation data message.

This separation into two channels allows a better performance in terms of acquisition and tracking process since the accumulation time on the pilot channel is no longer restricted by the duration of a data symbol [JULIEN, 2005]. The power distribution between the two channels is 75% for the pilot channel and 25% for the data channel of the total amount of power.

The L1C_P(t) and the L1C_D(t) codes have a duration of 10ms at a chipping rate of 1.023 Mbps, meaning a length of 10230 chips. Both codes are the result of inserting into a unique 10233-length sequence a fixed 7-bit length sequence. The insertion point of the 7-bit length sequence varies for each satellite and is not the same for the L1C_P(t) codes than for the L1C_D(t) codes. This unique 10233-length sequence is a Weil code generated from the 2-modulo sum of two 10233 Legendre sequences. These two Legendre sequences are the same but one is shifted in relation to the other. This shift is different between the L1C_P(t) codes and the L1C_D(t) codes, and it also varies among satellites.

The overlay codes L1C_O(t) are constructed using the Linear Feedback Shift Register (LFSR) method. These codes are 2047 bits long sequences truncated to 1800-bits long sequences at a rate of 100 bit/sec, and thus their duration is 18s. These codes are constructed using 11-stage LFSR generators. More precisely, each code is generated by 2-modulo adding two sequences S1 and S2 with different polynomial generators for each satellite. Nevertheless, the first 64 sequences are generated by only using S1.

Finally, the interleaver applied to the subframe 2 and subframe 3 of the GPS L1C navigation message is presented.

D.4.1. Interleaver and Desinterleaver

In this section the interleaving block and the way it works are described. The function of the desinterleaver which is supposed to be the inverse process of the interleaver is not detailed here.

The block interleaver is conceptually described using a two-dimensional array of 38 rows and 46 columns (the multiplication result is 1748: the exact number of bits of subframe 2 plus subframe 3), as depicted in Figure D-1 [ARINC, 2006]. The LDPC encoded subframe 2 bits are written first (MSB first) into the interleaver from left to right starting at row 1. After row 1 is filled, row 2 is filled from left to right and this process continues until the 1748th bit (LSB of LDPC encoded subframe 3) is written into the rightmost cell of the last (38th) row. Once all 1748 bits are written into the array, the symbols are sequentially read out of the array, for broadcast to user, from top to bottom starting at column 1. After reading out the last (38th)
symbol in column 1, column 2 bits are read out from top to bottom and this process continues until the last symbol (38\textsuperscript{th}) of the last column (46\textsuperscript{th}) is read out.

Therefore, the desinterleaver process is to first fill the array from column 1 to column 46 (last one) beginning at the top and finishing at the bottom of the columns, and, afterwards, to read the array from row 1 to row 38 from left to right of the rows. Figure D-2 shows a scheme of the desinterleaver process [ARINC, 2006].
D.5. GALILEO E1 signal

A possible GALILEO E1 OS signal modulator scheme is shown below.

Where:
- $e_{E1-B}$ is the I/NAV navigation data waveform $D_{E1-B}$ multiplied by the ranging code waveform $C_{E1-B}$, then modulating the sub-carriers $sc_{E1-B,a}$ and $sc_{E1-B,b}$
- $e_{E1-C}$ (pilot component) is the ranging code waveform $C_{E1-C}$ including its secondary code, then modulating the sub-carriers $sc_{E1-C,a}$ and $sc_{E1-C,b}$

The mathematical models of $c_{E1-B}(t)$, $c_{E1-C}(t)$ and $d_{E1-B}(t)$ are:

$$c_{E1-B}(t) = \sum_{k=-\infty}^{\infty} c_{E1-B}[k] \cdot rect_{T_c}[t-k \cdot T_c] \tag{D-1}$$

$$c_{E1-C}(t) = \sum_{k=-\infty}^{\infty} c_{E1-C}[k] \cdot rect_{T_c}[t-k \cdot T_c] \tag{D-2}$$

$$d_{E1-B}(t) = \sum_{k=-\infty}^{\infty} d_{E1-B}[k] \cdot rect_{T_d}[t-k \cdot T_d] \tag{D-3}$$

Where:
- $T_c$: Duration of a PRN code chip
- $T_c$: Duration of a data symbol
- $rect_{T}(t)$: Rectangular impulse of $T$ width centered at $t = 0$ The sub-carrier rates are:

<table>
<thead>
<tr>
<th>Component (Parameter Y)</th>
<th>Sub-carrier Type</th>
<th>Sub-carrier Rate (MHz)</th>
<th>Ranging Code Chip-Rate $R_{C,E1-Y}$ (Mcps)</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>CBOC, in-phase</td>
<td>1.023</td>
<td>6.138</td>
</tr>
<tr>
<td>C</td>
<td>CBOC, anti-phase</td>
<td>1.023</td>
<td>6.138</td>
</tr>
</tbody>
</table>

Table D-1: GALILEO E1-B CBOC Chip and Sub-carriers Rate
The final mathematical model of $s_{E1}(t)$ is:

$$ s_{E1}(t) = \frac{1}{\sqrt{2}} \cdot m(t) \cdot \cos(2\pi f_s t) \quad (D-4) $$

Where:

$$ m(t) = e_{E1-B}(t) \cdot (\alpha \cdot sc_{E1-B,C}(t) + \beta \cdot sc_{E1-B,A}(t)) - e_{E1-C}(t) \cdot (\alpha \cdot sc_{E1-C,A}(t) - \beta \cdot sc_{E1-C,B}(t)) \quad (D-5) $$

$$ sc_\times(t) = \text{sgn}[\sin(2\pi R_{3,\times}(t))] \quad (D-6) $$

$$ \alpha = \sqrt\frac{10}{11} \quad \beta = \sqrt\frac{1}{11} \quad (D-7) $$

The GALILEO E1 signal is contained within a band of 24.552 MHz centered on L1. The nominal carrier frequency ($f_0$) for L1 is 1575.42 MHz.

The PRN codes of the GALILEO E1 signal, $C_{E1-C}$ and $C_{E1-B}$, are different for each transmitting satellite as well as they are different from each other. This property allows the differentiation among satellites in addition to the separation between the part of the signal carrying the $C_{E1-C}$ code and the part of the signal carrying the $C_{E1-B}$ code. Therefore, due to the differentiation among satellites and the differentiation among GALILEO E1 codes, the GALILEO E1 signal can be divided into two channels: the pilot channel, part of the signal containing the $C_{E1-C}$ code, and the data channel, part of the signal containing the $C_{E1-B}$ code plus the navigation data message.

This separation into two channels allows a better performance in terms of acquisition and tracking process since the accumulation time on the pilot channel is no longer restricted by the duration of a data symbol [JULIEN, 2005]. The power distribution between the two channels is 50% for the pilot channel and 50% for the data channel of the total amount of power.

The PRN codes $C_{E1-C}$ and $C_{E1-B}$ are memory codes, or in other words, optimized pseudo-noise sequences which need to be stored in memory since they cannot be reproduced from a LFSR for example. The $C_{E1-B}$ is transmitted at a rate of 1.023 Mchip/s and has a length of 4092 chips meaning a duration of 4ms. The $C_{E1-C}$ is a primary memory code further modified by a secondary code. This secondary code is also a memory code, it is called CS251 and it has a length of 25 chips. This means that the primary code is repeated 25 times and its sign is controlled by the chips of the secondary code, see Figure D-4. Therefore, since the length of the primary code is 4092 chips transmitted at a rate of 1.023 Mchips/sec, the primary code has a duration of 4ms whereas the secondary one lasts 100ms. Figure D-4 illustrates the $C_{E1-C}$ code generation [ESA, 2008].
D. Advanced description of the GNSS signals

Figure D-4: Tiered Codes Generation [ESA, 2008]

D.5.1. Navigation message

In this annex, the different information units of the GALILEO E1 OS signal are described. The information units are the part of a page, a page, subframe and frame. Moreover, the interleaver used to interleave a part of a page is also described.

D.5.1.1. Part of a Page Structure

Each type of page is divided into two parts; the first part is denoted ‘even’ and the second one is denoted ‘odd’. A part of a page, even or odd, carries 120 information bits which are encoded with a convolutional code (171, 133, r = ½) resulting into 240 coded bits. Moreover, 10 additional synchronization bits are inserted before the coded bits in order to allow the synchro-part-of-a-page process at reception. Figure D-5 shows the construction of a part of a page [ESA, 2008].

Figure D-5: Even or Odd Part of a Page Structure Galileo E1 OS [ESA, 2008]

The synchronization symbols are equal to 0101100000.

Since the total size of an even or odd part of a page is 250 symbols after applying the CBOC modulation and the GALILEO E1 signal symbol transmission rate is equal to 250 symb/s, the part of a page duration is 1 second.
D.5.1.2. Interleaver and desinterleaver

Each part of a page, even or odd, is processed with an interleaver block before being transmitted to the channel. The interleaver block is simply a matrix of 30 columns and 8 rows, which is equivalent to 240 slots. That is exactly the same number as the total amount of bits in a part of a page before inserting the synchronization symbols.

The interleaver first fills the array with the 240 coded bits starting at column 1 from its top to its bottom and continuing with column 2 until column 30. Afterwards, the bits are read from the array and transmitted to the channel starting at row 1 from its left to its right, and continuing with row 2 until row 8.

The process for the desinterleaver is the opposite, first the rows are filled and after the information is read from the columns.

D.5.1.3. Page Structure

Each page is composed of two parts of a page, the even part and the odd one. Moreover, each page is fully transmitted on the GALILEO E1 signal although their even and odd parts can be, at the same time, transmitted on the GALILEO E5b-I signal. Therefore, since the duration of a part of a page is 1 second, the duration of a page is always 2 seconds.

There are two different types of pages. These pages are called Nominal Pages and Alert Pages. The nominal pages are transmitted sequentially in time on the GALILEO E1 signal; their even and odd parts are transmitted one after the other, and they are not repeated at the same time on the GALILEO E5b-I signal. The alert pages are also transmitted sequentially in time on the GALILEO E1 signal, but during the epoch that this signal transmits the even part, the GALILEO E5b-I signal transmits the odd part; the opposite is also true.

The alert page structure is not available in the GALILEO ICD draft 1 [ESA, 2008]; therefore, the only page structure studied and displayed along this dissertation is the nominal one. This page carries the fundamental information necessary to obtain the user final position. Nevertheless, not all the information of a nominal page serves to carry the Open Service. In fact, only 128 out of the total 240 information bits carry the OS content. The other bits have different functions which are listed next.

- Part of the other services, SoL or CS (noted as reserved)
- Page and word identifiers
- SAR data: System information generated by the satellites destined to the ground segment.
- A parity check code: CRC-24Q which validates all the page information transmitted before it.
- Tail bits: Fixes the final state of the convolutional code applied on the page. A more detailed explanation is provided in annex E.3.2.3.
D. Advanced description of the GNSS signals

The structure of a nominal page is described below [ESA, 2008].

![Figure D-6: Page Structure Galileo E1 OS [ESA, 2008]](image)

Note that Figure D-6 shows the page structure of the information bits of a part of a page. Therefore, each part of a page has first to be encoded with the convolutional code, second the coded bits have to be interleaved and third the synchronisation symbols have to be added. Only then, the parts of a page are transmitted to the channel.

### D.5.1.4. Subframe Structure

A subframe is composed of 15 pages. The information content type of a page depends on its position inside the subframe. This means that pages situated at the same position inside the subframe but transmitted inside different subframes have the same type of information content. Figure D-7 shows the GALILEO E1 OS signal subframe scheme [ESA, 2008].

![Figure D-7: Subframe Structure Galileo E1 OS [ESA, 2008]](image)

Pages 1 to 4 contain the ephemeris and clock correction data. Page 5 contains the BGD, the signal health and data validity status, the ionospheric correction and the GST. The content of Page 6 is the GST-UTC conversion parameters, the signal health and the data validity status. Finally Pages 7 to 10 contain the almanacs.

The duration of a subframe is 30 seconds since it is carries 15 pages or 30 parts of a page.

### D.5.1.5. Frame Structure

Pages 1 to 4 and, 5 and 6, have a content which is constant from one subframe to the next one until one new set of ephemeris and clock correction data is broadcasted. The time between
D. Advanced description of the GNSS signals

consecutive broadcasted ephemeris set is not specified, which means that this time can have any value.

Pages 7 to 10 carry the almanac information of all the satellites. One subframe is able to transport all the almanac data of a given satellite, and the satellite of which the almanac data is broadcasted by subframe \( n \) is different from the satellite of which the almanac data is broadcasted by subframe \( n + 1 \). Hereby, a frame is defined by the group of subframes which broadcast all the available satellites almanac data. In GALILEO E1 OS navigation message a frame is composed of 24 subframes. A GALILEO E1 OS frame structure scheme is shown below [ESA, 2008].

In this scheme, it can be seen that only one couple of pages either, 7 and 8, or, 9 and 10, are transmitted for each subframe. This is due to the subframe structure which only reserves 2 pages for the almanac transmission.

The duration of a frame is 720 second that is to say 12 minutes.

### D.6. New proposed GALILEO E1 navigation message structure

The new navigation message structure proposed by THALES ALENIA SPACE is presented in this section. The objective behind this navigation message proposal for a future GALILEO E1 signal is to improve the demodulation performance obtained by the current GALILEO E1 OS signal since this demodulation performance is proven to not be satisfying enough in 0. Moreover, the new proposed navigation message structure also seeks to increase the data symbol transmission rate in order to allow the providing of new services.
The new navigation message structure proposed by TAS-France recovers elements from the GPS L1C navigation message structure and also implements the CSK technique. This combination results into a new navigation message with a higher information transmission rate and a better ephemeris and almanac demodulation performance than the information transmission rate and demodulation performance obtained for the current defined GALILEO E1 navigation message.

More specifically, the GALILEO E1 original words and information structure are modified. The Viterbi pages of 250 symbols and their convolutional code no longer exist. They have been substituted by the GPS L1C subframe structures and its implemented channel codes. Nevertheless, in order to conserve the frame length of 30 seconds, the GALILEO E1 signal ephemerides and almanacs fields are located differently inside the frame from the location used for the GPS L1C fields.

Moreover, in addition to modifying the navigation message structure, this proposal also suggests changing the data channel PRN code structure in order to implement the CSK technique. Remember that the CSK technique is responsible of increasing the signal information transmission rate. The CSK is fully described in subsection 7.1. However, although this proposal changes the data channel PRN code structure, the power distribution between the data and pilot channels, the symbol transmission rate of the non-CSK symbols and the CBOC modulation of the pilot and data channels are not modified. More specifically, as it can be seen from Figure D-9, the total amount of 250 symbols is transmitted each second, which means that the original symbol transmission rate is conserved as it has been said previously.

A scheme of the new proposed navigation message structure called ERIS is shown below. More specifically, the frame of 30 seconds is presented.

<table>
<thead>
<tr>
<th>Time (s)</th>
<th>Subframe (number of bits)</th>
<th>Total Number of bits</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>1</td>
<td>42</td>
<td>8</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>60</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>62</td>
</tr>
<tr>
<td>4</td>
<td>40</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>40</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>40</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>40</td>
<td>0</td>
</tr>
<tr>
<td>8</td>
<td>40</td>
<td>0</td>
</tr>
<tr>
<td>9</td>
<td>40</td>
<td>0</td>
</tr>
<tr>
<td>10</td>
<td>40</td>
<td>0</td>
</tr>
<tr>
<td>11</td>
<td>40</td>
<td>0</td>
</tr>
<tr>
<td>12</td>
<td>40</td>
<td>0</td>
</tr>
<tr>
<td>13</td>
<td>40</td>
<td>0</td>
</tr>
<tr>
<td>14</td>
<td>40</td>
<td>0</td>
</tr>
<tr>
<td>15</td>
<td>40</td>
<td>0</td>
</tr>
<tr>
<td>16</td>
<td>40</td>
<td>0</td>
</tr>
<tr>
<td>17</td>
<td>40</td>
<td>0</td>
</tr>
<tr>
<td>18</td>
<td>40</td>
<td>0</td>
</tr>
<tr>
<td>19</td>
<td>40</td>
<td>0</td>
</tr>
<tr>
<td>20</td>
<td>40</td>
<td>0</td>
</tr>
<tr>
<td>21</td>
<td>40</td>
<td>0</td>
</tr>
<tr>
<td>22</td>
<td>40</td>
<td>0</td>
</tr>
<tr>
<td>23</td>
<td>40</td>
<td>0</td>
</tr>
<tr>
<td>24</td>
<td>40</td>
<td>0</td>
</tr>
<tr>
<td>25</td>
<td>40</td>
<td>0</td>
</tr>
<tr>
<td>26</td>
<td>40</td>
<td>0</td>
</tr>
<tr>
<td>27</td>
<td>40</td>
<td>0</td>
</tr>
<tr>
<td>28</td>
<td>40</td>
<td>0</td>
</tr>
<tr>
<td>29</td>
<td>40</td>
<td>0</td>
</tr>
<tr>
<td>30</td>
<td>40</td>
<td>0</td>
</tr>
</tbody>
</table>

Figure D-9: THALES ALENIA SPACE proposed ERIS frame of 30 seconds
The different subframes content is described next.

- **Sub-frame 1** is transmitted during the 1st second. Carries the Time of Interval (TOI), a counter of 9 information bits which after BCH encoding results into 52 coded bits. Each coded bit is mapped with a BPSK symbol. This subframe is represented with yellow color.

- **Sub-frame 2** is transmitted during the 30 seconds. Carries the Ephemeris and clock corrections error data parameters. This information is represented by 600 information bits which after LDPC encoding results into 1200 coded bits. Each coded bit is mapped with a BPSK symbol. This subframe is represented with green color.

- **Sub-frame 3** is transmitted during the last 28 seconds. Carries different navigation parameters such as the almanac parameters, UTC parameters, ionospheric parameters, etc. This information is represented by 274 information bits which after LDPC encoding results into 548 coded bits. Each coded bit is mapped with a BPSK symbol. This subframe is represented with yellow-orange color.

- **Sub-frame 4** is transmitted during 30 seconds. The information carried by this subframe is still to be determined. Nevertheless, it has to carry the data necessary to provide additional services such as Safety of Life, ERIS, SAR Return, ALIVE, etc. Moreover, the CSK signaling technique is being considered to be implemented in this subframe. In this case, the data channel PRN code of the subframe 4 has to be modified. This subframe is represented with violet color.

- **Sub-frame 5** is transmitted during 30 seconds. This subframe carries the information necessary to provide each second the Integrity Status. This subframe is represented with blue color.

The navigation message structure proposed by THALES ALENIA SPACE suggests the introduction of the CSK technique inside the subframe 4. Subframe 4 is completely used to implement the CSK signaling technique except for some bits which determine the configuration of the content of subframe 5. Therefore, the number of available bits in subframe 4 which can be used to implement the CSK is 138. Some other parameters associated to the CSK signaling technique are not defined by this proposal, only some suggestions are made.
Annex E. Decoding methods

In this annex, the different decoding methods used or developed in this research work are presented.

E.1. Extended Hamming code (32, 26)

The extended Hamming code (32, 26) is the channel code implemented for the GPS L1 C/A signal. The encoding and decoding process of this channel code as well as the strategy of application of this code are explained next.

E.1.1. Encoding process

A Hamming code [PROAKISe, 2001] is a linear block code which can be systematic. A code is systematic if the information word is exactly found as a part of the code word, where the information word is the ensemble of information bits and the code word is the ensemble of coded bits. Besides, a block channel code normally transforms the information word to the code word by multiplying a vector defined from the information word by the generation matrix (G) of the channel code [PROAKISe, 2001].

The introduction of a parity bit is the process of adding to the n-bits code word an extra bit whose value is equal to the modulo-2 sum of the n bits of the code word. In other words, if the modulo-2 sum is equal to 0 the added bit is also equal to 0, but if the modulo-2 sum is 1 the bit is equal to 1. Another way of introducing the parity bit into the code word is to define a new generation matrix (G') for the extended channel code which takes into account the parity bit generation. In this latter case, the new generation matrix is the generation matrix of the extended code. In the GPS L1 C/A case, the original channel code is the (31, 26) Hamming code and the final extended code is the (32, 26) extended Hamming code.

Therefore, in order to obtain the GPS L1 C/A code word, the encoder should define an input vector of 26 information bits and multiply it by the generation matrix (G') in order to obtain the 32 bits code word. However, in the GPS L1 C/A navigation message, the size of the code word is equal to 30 bits where 24 are information bits and 6 are parity ones. This means that two information bits are discarded or not transmitted with the code word although they are necessary to execute the coding/decoding process. In fact, these 2 bits are the last 2 bits of the previous code word. In other words, the bit 29 and 30 (parity bits) of the code word \(n\) are used as the first 2 information bits of the information word \((n+1)\). Therefore, these 2 bits are not transmitted again with the code word \((n+1)\) because their value is already known when the word \(n\) is received. The entire process is displayed on Figure E-1.
Finally, one important characteristic of the GPS L1 C/A code is that it is not systematic, since the bit 30 of the word \( n \) determines if the information bits values of the word \( (n+1) \) are inverted. Therefore, the information word \( (n+1) \) may not be found in the code word \( (n+1) \) but instead its complementary can be.

Another method to generate a GPS L1 C/A coded word is to use the algorithm presented in the document [ARINC, 2004].

### E.1.2. Decoding process

The decoding process of a channel code consists in correcting the erroneous bits introduced by the transmission channel (annex C.4). Nevertheless, the extended Hamming code \((32,26)\) was not originally designed to correct errors but rather to detect the erroneous words. Therefore, the GPS L1 C/A channel code strategy was to discard all the erroneous words and to only use the information of the a priori correct words.

The traditional linear block channel codes technique employed to detect if the received code word is free from errors consists in calculating a vector called syndrome. This vector is calculated from the multiplication of the received word by the parity matrix \((H)\), where the parity matrix can be generated from the generation matrix [PROAKISE, 2001].

Once the syndrome has been obtained, the inspection of its value determines which pattern of errors is found on the word. The null syndrome, i.e. the all-zeros vector, indicates that the word does not have any error. Therefore, if the syndrome is the all-zeros vector, the receiver declares the word as error free; otherwise the word is discarded. Additionally, it is possible to correct some of the errors introduced by the transmission channel by using the syndrome vector. This method is fully described in the next reference [PROAKISE, 2001].

Nevertheless, the previously explained detection technique needs as input the complete received word, the word \( n \) plus the last two bits of the word \((n-1)\); which means that before calculating the syndrome, the original coded word has to be reconstructed. Figure E-2 illustrates this reconstruction.
Finally, another technique applied to detect the errors is described in this document [ARINC, 2004].

### E.2. CRC-24 Qualcomm channel code

The CRC-24Q [ARINC, 2004] is the outer channel code implemented on the GPS L2C, GPS L5, GPS L1C and GALILEO E1 OS signal. The CRC-24Q is a cyclic channel code which generates 24 parity bits from any number of input information bits. The coding process is achieved through algebraic manipulation and the process is customized in order to make the code systematic.

The code word is equal to:

\[ c(X) = m(X) \cdot X^{24} + r(X) \]  \hspace{1cm} (E-1)

\[ r(X) = \frac{m(X) \cdot X^{24}}{g(X)} \]  \hspace{1cm} (E-2)

Where:

- \( c(X) \): Code word
- \( m(X) \): Information word
- \( g(X) \): Generator polynomial
  - \( g(X) = (1 + X) \cdot p(X) \)
  - \( p(X) = X^{23} + X^{17} + X^{13} + X^{12} + X^{11} + X^9 + X^8 + X^7 + X^5 + X^3 + 1 \)
- \( r(X) \): Remainder of the division between \( m(X) \cdot X^{24} \) and \( q(X) \)

This remainder represents the cyclic code parity bits and the information word \( m(X) \) is expressed as:

\[ m(X) = m_k + m_{k-1} \cdot X + m_{k-2} \cdot X^2 + \ldots + m_1 \cdot X^{k-1} \]  \hspace{1cm} (E-3)

Where

- \( k \): Number of input bits

Finally, the decoding process consists in dividing the received code word, a word provided by the inner channel code, by the generator polynomial \( g(X) \). In fact, this operation obtains a
E. Decoding methods

syndrome and thus, if the syndrome is equal to the zero polynomial, the word is claimed as error free. And, since the channel code is systematic the information bits correspond to the first K-24 bits of the received code word, where K is the total number of information bits.

As has been said previously, the objective of the outer channel code is to detect and to discard the words provided by the inner channel code which still contain some error. Some properties concerning the CRC-24Q detection function are given below [ARINC, 2004]:

1. It detects all single bit errors per code word.
2. It detects all double bit error combinations in a codeword because the generator polynomial \( g(X) \) has a factor of at least three terms.
3. It detects any odd number of errors because \( g(X) \) contains a factor \( 1+X \).
4. It detects any burst error for which the length of the burst is \( \leq 24 \) bits.
5. It detects most large error bursts with length greater than the parity length \( r = 24 \) bits. The fraction of error bursts of length \( b > 24 \) that are undetected is:
   a. \( 2^{-24} = 5.96 \times 10^{-8} \), if \( b > 25 \) bits.
   b. \( 2^{-23} = 1.19 \times 10^{-7} \), if \( b = 25 \) bits.

E.3. Convolutional code

The convolutional code presented in this annex is the inner convolutional code implemented for GPS L2C signal, for GPS L5 signal and for GALILEO E1 OS signal. This channel code has a code rate, \( r \), equal to \( \frac{1}{2} \) and two polynomial generators equal to \( G_1 = 171_{\text{octal}} \) and \( G_2 = 133_{\text{octal}} \).

The encoding and decoding process of this convolutional code are presented next.

E.3.1. Convolutional channel code coding process

The coding process can be perfectly summarized by the following block scheme. See Figure E-3 [ARINC, 2004].

![Figure E-3: Encoding block scheme of the convolutional code (171,133) [ARINC, 2004]](image)

For each bit entering the linear shift register at its left side, the register calculates two new output bits. Each one of these 2 new output bits is calculated from a 2-modulo addition between the new entering bit and the previous L-1 input bits, where all the bits are weighted by a polynomial generator. Afterwards, the bit calculated from \( G_1 \) is transmitted first to the
channel followed by the G2 bit. Finally, the oldest bit of the L-1 previous bits is eliminated and the entering bit is stored inside the encoder memory in order to prepare the system for the arrival of the next input bit.

In the previous paragraph, it has been said that the generation of the output bits depends on the polynomial generators (G1, G2), on the new input bit (bit $n$) and on the old ones (bits $n-1..n-L+1$). In fact, these old input bits are the memory of the code and they define the state of the encoder. Therefore, the output bits only depend on the input bit and on the state of the encoder. Moreover, the transition of one state to another is controlled by the input bit and thus some transitions are impossible. This is the one of the keys of the most common decoding method and has to be remembered.

### E.3.2. Convolutional channel code decoding process

The inner channel code is a convolutional code, therefore the two main decoding methods are the Viterbi algorithm [VITERBI, 1967], [VITERBI and OMURA, 1979], and the BCJR (Bahl, Cocke, Jelinek and Raviv) algorithm [BAHL et al., 1974].

The Viterbi technique searches for the most probable sequence of symbols/bits being transmitted; or, in other words, this method minimizes the probability of error of the decoded sequence. The BCJR method searches the most probable transmitted symbol/bit; or, in other words, this technique minimizes the probability of error of each received symbol/bit. The most used decoding method is the Viterbi algorithm because, although the performance of both methods is almost the same, the Viterbi algorithm has smaller computational and storing costs. Therefore, the decoding technique described below is the Viterbi algorithm. Moreover, the BCJR technique does not allow the implementation of the technique presented in 0.

#### E.3.2.1. Viterbi Definitions

First of all, some definitions and concepts are given in order to better understand the Viterbi algorithm and in order to allow an easier comprehension of the algorithm. Remember that the detailed Viterbi algorithm can be found in [VITERBI, 1967], [VITERBI and OMURA, 1979].

The state, the state of the encoder or the state of the decoder is the set of the last L-1 information bits which have been encoded. The state altogether with the polynomial generators and the input information bit to be encoded determine the value of the code bit. From the state notion, it can be said that each information bit sequence generates or travels a determined sequence of states. Therefore each sequence can be defined by the sequence of states that it travels. In this report, a sequence of states is called path, and thus each different sequence travels a different path.

Moreover, since the transition from one state to another one can only occur when a unique pattern of symbols is applied at the convolution encoder input, a unique path also determines a unique sequence of symbols. In this case, 1 symbol maps only 1 bit.

The trellis of a convolutional code is a time representation of all the possible paths that can be generated by the convolutional code. Obviously, each different path is generated by a different symbols sequence input. A simple trellis example is shown in Figure E-4. The trellis corresponds to a convolutional code $(7, 5, r = \frac{1}{2})$ and it is only plotted until $t = 4$ because from there on the trellis is repeated; there is at least one path in each state.
E. Decoding methods

In Figure E-4, it can be seen that more than one sequence, in this case 2, arrive at the same state. In this dissertation, the action of two or more sequences arriving at the same state at the same instant of time is called as that these sequences merge into that state. Moreover, the state where the two or more sequences merge is called merging state.

Another important definition is the transition distance between two states (state X and state Y). This distance is calculated from the coded symbol values estimated by the receiver and the ideal coded symbol values expected between the states. A code symbol value estimated at time $n$ by the receiver is the transmitted coded symbol value at time $n$ plus the distortion and/or noise introduced by the transmission channel. An ideal coded symbol value expected between two states (state X and state Y) at time $n$ is the coded symbol value generated by the encoder when its state at time $n-1$ is the state X and the input information symbol/s makes the encoder to pass to the state Y at time $n$.

Depending on the type of decoding, soft or hard, this distance is calculated differently. In this PhD manuscript, the decoding is always soft; therefore the transition distance is evaluated as the Euclidian distance between the receiver estimated code symbol values and the ideal expected code symbol values:

$$\text{Transition Distance of the } k^{th} \text{ info symbol} = \sum_{i=1}^{N} |r_{ki} - c_{XY ki}|^2$$

(E-4)

Where:

- $N$: Number of coded symbols encoding an information symbol
- $r_{ki}$: $i^{th}$ estimated coded symbol component by the receiver of the $k^{th}$ estimated coded symbol (representing the $k^{th}$ information symbol).
- $c_{XY ki}$: Ideal $i^{th}$ coded symbol from the $k^{th}$ information symbol within the inspected sequence. The coded symbol values are generated by passing from state X to state Y.

The last definition is the sequence accumulated distance. The accumulated distance of a sequence is the sum of all the path transition distances; or in other words, the addition of all the individual transition distances of the transitions between the states forming the path associated to the sequence. Moreover, the accumulated distance of a sequence determines the probability for that sequence to be the transmitted sequence. In fact, the larger the accumulated distance of a sequence is, the smaller the probability for that sequence to be the transmitted sequence is. The same can be said for the inverse case.
E.3.2.2. Viterbi Algorithm

The main principle of the Viterbi algorithm is to choose the sequence with the smallest accumulated distance as the transmitted sequence because, as it has been said in the previous section, the sequence with the smallest accumulated distance is the most probable transmitted sequence. Therefore, the Viterbi algorithm should calculate the accumulated distance of all the possible sequences that can be transmitted. However, this calculation is not possible to achieve because the number of possible sequences grows exponentially with the sequences length. Therefore, the main principle of the Viterbi algorithm is to calculate the accumulated distances of all the sequences with possibilities of being the transmitted sequence. This means that it is not necessary to wait until the end of the sequence reception to decide which the most probable sequence is. In fact, during the reception process, there are some sequences which can already be discarded since they do not have any possibility of being selected by the receiver. Moreover, note that the receiver cannot usually wait until the end of the sequence reception since the data input can be a constant stream of bits.

The reason for early sequence elimination is explained next. The accumulated distance of a sequence depends on the states travelled by the sequence. Therefore, when two or more sequences merge into a determined state, a first accumulated distance comparison is made. This comparison evaluates the path travelled until this moment for each individual sequence. The path of each sequence can be partially or completely different. Therefore, the accumulated distance of each sequence at this point shows which sequence is more probable of being transmitted until this point. Besides, this first comparison already determines which sequence among the two sequences will be the most probable transmitted sequence at any future instant. Or, in other words, the comparison determines that the sequence with the smallest accumulated distance at the merging state will always have the smallest accumulated distance among all the sequences merging at that instant at the merging state. In fact, from the merging state on, any possible path travelled by the largest accumulated distance sequence can also be followed by the smallest accumulated distance sequence. Therefore, since the additional accumulated distance calculated from the merging state depends only on the future path, a path which can be equal for both sequences, the sequence with the largest accumulated distance will always keep the largest accumulated distance in relation to the other sequence. Therefore, since the largest accumulated distance sequence will always be less probable than the smallest accumulated distance sequence, the former sequence can be eliminated without consequences.

To sum up, only one of the sequences merging in the state $S$ at each time $t$ survives whereas the others are eliminated. This path is called the survivor. An example is given below.

![Figure E-5: Merging State and surviving path.](image-url)
From Figure E-5, the $X_0$ path accumulated distance at the merging state was smaller than the $X_1$ path accumulated distance at the merging state. Therefore, the path $X_0$ is the survivor whereas the path $X_1$ is eliminated.

Once the justification of why not all the paths have to be conserved in order to find the more probable transmitted sequence has been explained, the Viterbi algorithm can be easily understood. First of all, a sequence of accumulated distance equal to 0 is stored at time $t=0$ for each possible state. Second, using the received estimated code symbols values and the ideal coded symbol values of each pair of states, the transition distances between $t=0$ and $t=1$ are calculated using equation (E-4). These transitions distances are added to the origin state survivors accumulated distance. In this case, the accumulated distance is 0. Once the sum has been done, each result is compared to the result of the other sequence/s merging in the same destination state. Finally, only the sequence with the smallest addition result is conserved whereas the others are eliminated. And the addition result is the new accumulated distance of the survivor. The process is repeated until the reception end when the sequence with the smallest accumulated distance is chosen as the transmitted one.

A scheme of the Viterbi algorithm at the iteration $n$ is shown below.

![Figure E-6: Viterbi Algorithm - State Transition](image)

Where:
- $\lambda_i(k) =$ Accumulated Distance for the survivor sequence at state $i$ at the instant $kT$
- $\lambda_{ij}(k) =$ Transition Distance : Euclidian distance $d_E(X_{ij}, Y_k)$ between:
  - $X_{ij}$: Transmitted word between the $i^{th}$ state and the $j^{th}$ state (ideal coded symbols)
  - $Y_k$: Received Word at the instant $kT$ (estimated code symbols)

This part of the algorithm is called ACS (Add Compare Select).

Finally, the choice of the most probable sequence is explained. Until now, the choice of the sequence with the smallest accumulated distance as the transmitted sequence has been avoided by assuming that at some instant the input of bits will stop. However, the input is usually a constant stream of bits and thus the decoder cannot wait until the transmission end, which does not exist, before providing the decoded bits. Therefore, the algorithm has to find another way of decoding the symbols while the transmission is still in process. In fact, a simple solution can be implemented since after several iterations, or state transitions, it is observed that each survivor has the beginning of its path equal to the other survivors.
beginning of their path. This means that all the survivors have converged to the same path in the past. Therefore, the receiver can provide the information symbols corresponding to the common path part since it can be guaranteed that the most probable transmitted sequence would include these symbols. However, the path convergence does not always occur at the same past instant for each survivor. Besides, the time of path convergence is variable. Nevertheless, since it is known that the path of all the survivors converge with a high probability at time \( nT \) when the decoder is receiving the coded symbols of time \( (n+\Delta)T \), the decoders can adopt the following strategy. At time \( nT \), the receiver picks the path with the lowest actual accumulated distance, travels back the path of the chosen sequence \( \Delta T \) seconds in time and selects as the transmitted information symbol the symbol marked by the selected path at the instant \( (n-\Delta)T \). This effect is called Decoding Delay and the quantity \( \Delta \) (= 3·L to 5·L, where L is the constraint length) is called Decoding Depth.

E.3.2.3. Insertion of tail bits in the information word

The insertion of tail bits in the information word has as effect to convert the convolutional code into a block code. This means that each block of bits is encoded independently with respect to another block of bits because the initial state and the final state of the encoder for each block of bits are known.

The GALILEO E1 OS navigation message implements the convolutional code explained in annex E.3 and introduces tail bits in order to convert the convolutional code into a block code. The tail bits are introduced at the end of the information bits of each part of a page which means that the input block of the code is the part of a page.

The number of tail bits is set to L-1, 6 in this case, in order to know the encoder register state after the tail bits encoding. As the tails bits are known from [ESA, 2008] (set to 0), the encoder always begins the encoding process of the next part of a page at the same state of the convolutional code. And this knowledge of the initial state transforms the convolutional code into a sort of block channel code. Moreover, since the initial and final states of the coded block are known, 0 state since the tail bits are equal to 0, the decoding performance is improved due to the decrease of the number of possible transmitted sequences. The only inconvenient of the transformation process of a convolutional code into a block code is the addition of L-1 additional bits; at least for the implementation used in GALILEO E1 OS.

An example of the effect of the insertion of the tail bits can be seen in Figure E-7. This figure represents the trellis of a convolutional code \((G_1, G_2, r = \frac{1}{2})\) with \( L = 3 \). Therefore, the last two transmitted bits are the tails bits which are set to 0 forcing the last encoder state to the 0 state.

![Figure E-7: Tail bits influence into the trellis of a convolutional code (G1, G2, r=1/2)](image-url)
E.3.2.4. Viterbi algorithm with a priori bit probabilities

In order to better comprehend this section, we have presented the convolutional code encoding process and the Viterbi algorithm in annex E.3.1 and E.3.2.2.

The Viterbi algorithm objective is to minimize the error probability of the received sequence. Or, in other words, the fundamental expression consists of selecting the sequence, \( s^{(m)} \), among all the possible existing sequences that have the highest probability of being the transmitted sequence when sequence \( r \) has been received.

**Fundamental equation:**

\[
\max_{s^{(m)}} f(s^{(m)}|r) 
\]

\( r_k = c_k + n_k \) \hspace{1cm} (E-5)

Where:

- \( m \): Number of symbols of the source X alphabet
- \( s^{(m)} \): Information sequence of K elements from source X alphabet
- \( s_k^{(m)} \): k\(^{th}\) information symbol of the \( s^{(m)} \) sequence
- \( r \): Estimated coded sequence by the receiver of K elements
- \( r_k \): k\(^{th}\) element or estimated coded symbol of the estimated coded sequence
- \( n_k \): Additive Gaussian White Noise.
- \( c_k \): k\(^{th}\) coded symbol of the coded sequence
- \( c_{ki} \): i\(^{th}\) component of the k\(^{th}\) coded symbol
- \( f(s^{(m)}|r) \): Probability of having transmitted the \( s^{(m)} \) sequence when the \( r \) sequence has been received.

In order to better understand equation (E-5), one important remark concerning the information sequence, the information symbols, the coded sequence and the coded symbols for a convolutional code has to be made. An information symbol is formed by the bits fed to the convolutional code encoder at the same time, and each of these input bits is also called an information symbol component of this information symbol. A coded symbol is formed by the bits at the output of the convolutional encoder at the same time, and each one of these output bits is also called a coded symbol component of this coded symbol. Finally, an information sequence is the ensemble of all the transmitted information symbols and a coded sequence is the ensemble of all the transmitted coded symbols. Note that if the input is an infinite stream of bits the sequences have an infinite length.

For example, in the GPS L2C and GPS L5 signal case, the channel code rate is \( r = \frac{1}{2} \), which means that the information symbol is formed by 1 information symbol component (1 bit) and that the coded symbol is formed by 2 coded symbol components (2 bits).

If we develop equation (E-5) by applying the Bayes theorem and since the denominator does not depend on \( s^{(m)} \), the fundamental expression is equivalent to:

**Bayes Theorem:**

\[
f(s^{(m)}|r) = \frac{f(r_1 \ldots r_K | s^{(m)}) \cdot P(s^{(m)})}{f(r_1 \ldots r_K)} \]

\[
\max_{s^{(m)}} f(s^{(m)}|r) = \max_{s^{(m)}} f(r_1 \ldots r_K | s^{(m)}) \cdot P(s^{(m)}) \] \hspace{1cm} (E-7)

\[
\max_{s^{(m)}} f(s^{(m)}|r) = \max_{s^{(m)}} f(r_1 \ldots r_K | s^{(m)}) \cdot P(s^{(m)}) \] \hspace{1cm} (E-8)
If we develop the right term of equation (E-8), we obtain:

\[
\begin{align*}
&f(r_1 \ldots r_K | s^{(m)}) P(s^{(m)}) = f(r_K | r_{K-1} \ldots r_1, s^{(m)}) \cdot f(r_{K-1} | r_{K-2} \ldots r_1, s^{(m)}) \ldots f(r_1 | s^{(m)}) \cdot P(s^{(m)})
\end{align*}
\]  

(E-9)

Each individual term of the right part of equation (E-9) is conditioned by several factors; however, the majority of them can be removed by a close inspection of each component \( r_k \).

First, \( r_k \) depends on the code symbol \( c_k \) and AWG noise \( n_k \). Second, \( c_k \) is determined by the \( k \)-th information symbol, \( s_k^{(m)} \), and the convolutional encoder state as has been explained in annex E.3.1. Third, the convolutional encoder state is determined by the \( L-1 \) previous information symbols, \( s_{k-1}^{(m)} \ldots s_{k-L+1}^{(m)} \). Therefore, the other information symbols do not condition the \( r_k \) element. Fifth, since the \( n_k \) noises samples are independent among them and the \( c_k \) value is fully determined by the information symbols, the terms from \( r_{k-1} \) to \( r_1 \) influence on \( r_k \) can be removed from expression (E-9). Finally, equation (E-9) can thus be expressed as:

\[
\begin{align*}
&f(r_1 \ldots r_k | s^{(m)}) P(s^{(m)}) = f(r_k | s_k^{(m)} \ldots s_{K-L+1}^{(m)}) \ldots f(r_1 | s_1^{(m)}) \cdot P(s_k^{(m)} \ldots s_1^{(m)})
\end{align*}
\]  

(E-10)

Since symbols \( s_k^{(m)} \) to \( s_{k-L+1}^{(m)} \) determine the value \( c_k \), the previous expression is equivalent to the following one:

\[
\begin{align*}
&f(r_1 \ldots r_k | s^{(m)}) P(s^{(m)}) = f(r_k | c_k) \cdot f(r_{k-1} | c_{k-1}) \ldots f(r_1 | c_1) \cdot P(s_k^{(m)} \ldots s_1^{(m)})
\end{align*}
\]  

(E-11)

Therefore, it only remains to express the different probabilities terms, where each conditioned probability term is the determined by the probability density function of an AWGN:

\[
\begin{align*}
&f(r_k | c_k) = \frac{1}{\sqrt{2\pi \cdot \sigma^2}} \cdot \exp \left( -\frac{1}{2 \cdot \sigma^2} \sum_i |r_{ki} - c_{ki}|^2 \right)
\end{align*}
\]  

(E-12)

Where:

- \( r_{ki} \): \( i \)-th component of the \( k \)-th estimated coded symbol
- \( c_{ki} \): \( i \)-th component of the \( k \)-th coded symbol
- \( \sigma^2 \): AWGN variance

Therefore, equation (E-8) can be rewritten as:

\[
\begin{align*}
&\max_{s^{(m)}} f(s^{(m)} | r) = \max_{s^{(m)}} \frac{1}{(2\pi \cdot \sigma^2)^{K/2}} \cdot \exp \left[ -\frac{1}{2 \cdot \sigma^2} \sum_{k=1}^K \sum_{i} |r_{ki} - c_{ki}|^2 \right] \cdot P(s_1^{(m)}) \ldots P(s_K^{(m)})
\end{align*}
\]  

(E-13)

Since \( f(s^{(m)} | r) \) is always positive by definition, it is the same to search for its maximum or for the maximum of its logarithm. Therefore, the equivalent expression is:

\[
\begin{align*}
&\max_{s^{(m)}} f(s^{(m)} | r) \Leftrightarrow \min_{s^{(m)}} \sum_{k=1}^K \left( \sum_{i} |r_{ki} - c_{ki}|^2 - 2\sigma^2 \log \left(P(s_k^{(m)})\right) \right)
\end{align*}
\]  

(E-14)

Finally, a closer inspection of equation (E-14) provides the modification to apply to the original Viterbi algorithm. From equation (E-14), we can see that index \( k \) represents the different coded symbols forming the sequence. Therefore, the addition really represents the transition distances addition and the result represents the sequence accumulated distance. This means that we can identify from the individual addition term the modification which has been added to the original transition distance. This term is:

\[
\begin{align*}
&\text{Additional transition distance term} = -2\sigma^2 \log \left(P(s_k^{(m)})\right)
\end{align*}
\]  

(E-15)
Note that this additional term depends on the noise power and on the a priori bit probability at instant $k$. Therefore, the new transition distance is:

$$Modified\ Transition\ Distance = \sum_i |r_{ki} - c_i|^2 - 2\sigma^2 \log(P_{ki}^{(m)})$$  \hspace{1cm} (E-16)

To sum up, the modified Viterbi algorithm is exactly the same as the original one but using equation (E-16) instead of equation (E-4).

### E.4. Decoding method based on the combination of the inner convolutional code and the outer channel code

In this section, the decoding method based on the inner and outer channel codes combination which is used in 0 (see section 5.2.2) is further detailed. The inner channel code is the convolutional code presented in annex E.3 and the outer channel code can be the code presented in either the annex E.1 or the annex E.2.

The main idea of this method is to determine as the transmitted sequence, the sequence provided by the Viterbi decoding of the inner channel code which has the minimum accumulated distance and which verifies the outer channel code.

The main steps of this decoding method as well as its complete algorithm are detailed in this annex.

#### E.4.1. Search of the $i^{th}$ candidate

The $i^{th}$ candidate search is the identification of the sequence having the smallest accumulated distance among all the remaining sequences which are not yet candidates when $(i-1)$ candidates have already been determined.

Before detailing the $i^{th}$ candidate search, some definitions are presented. The particularities of the search of a candidate when the initial and/or the final state are known are also described.

#### E.4.1.1. Definitions of the method based on the inner and outer channel codes combination

In this subsection, some definitions required to correctly follow the explanation of the process of the $i^{th}$ candidate search are presented. This subsection assumes that the reader is familiar with the candidate concept which is also an essential definition given in section 5.2.2.5 and with the Viterbi concepts given in annexes E.3.2.1 and E.3.2.2.

The minimum alternative path of the $k^{th}$ candidate is defined as the path merging with the $k^{th}$ candidate that has the minimal final accumulated distance among all the paths that, at any moment and at any state, merge once and only once with the $k^{th}$ candidate. And once the minimum alternative path has merged with the $k^{th}$ candidate, the minimum alternative path travels exactly the same path as the $k^{th}$ candidate. One scheme representing the paths merging with a candidate is represented below.
Another important definition is the state difference. The state difference of state X is the difference between the accumulated distances evaluated at state X of two sequences merging at state X.

The search of the minimum alternative path of the k\textsuperscript{th} candidate is equivalent to seek the state of the k\textsuperscript{th} candidate which has the minimal state difference among all the k\textsuperscript{th} candidate states. This state having the smallest state difference is called the minimal state. And once the minimal state has been found, we can directly determine the minimum alternative path of the k\textsuperscript{th} candidate. This path is defined in two parts. The first part is defined from $t = 0$ to the minimal state by the sequence which merges with the candidate at its minimal state. The second part is defined from the minimal state to the transmission end by the k\textsuperscript{th} candidate. Therefore, the minimum alternative path of the k\textsuperscript{th} candidate is completely defined in time by the two previous described parts.

The reason of this equivalence between the minimal state search and the minimum alternative path is presented next. From the definition of the minimum alternative path, we know that the minimum alternative path and the candidate follow the same path from their merging state. Therefore, the accumulated distance from the merging state until the transmission end is the same for both sequences. Consequently, the final accumulated distance of a sequence which merges with a candidate can be calculated by adding the final accumulated distance of the candidate to the accumulated distance of the sequence at the merging state with the candidate. In other words, the distance of the sequence is the final accumulated distance of the candidate plus the state difference of the merging state. Figure E-9 shows a graphical explanation.

From Figure E-9 and from its previous explanation, it can be seen that the minimal state of a candidate is the merging state between the candidate and the minimum alternative path of the candidate.

Finally, we call initial survivor to each sequence surviving all the Viterbi decoding process. Therefore, the number of initial survivors is equal to the number of states. Moreover, even if the transmission has no end, at each instant of time the Viterbi has survivors although these survivors can be eliminated in the future.
E. Decoding methods

E.4.1.2. Initial and final state knowledge effects

The main effect of the final and the initial state knowledge or their imposition is that all the possible decoded sequences begin at the same state A and finish at the same state B. Or, in other words, there is only one initial survivor. Therefore, it can be assured that the nearest sequence to another sequence in accumulated distance terms is a path that diverges from the original sequence at some instant \( t \) and merges again with it at some later instant \( t' \). Consequently, one sequence which diverges twice or more times from the original sequence and thus merges twice or more with the original sequence can never be the nearest sequence to the original sequence in accumulated distance terms. The justification is presented next.

First, we assume a path \( X_1 \) diverging twice from the original path \( X_0 \). Second, we assume another path \( X_2 \) being equal to sequence \( X_1 \) until sequence \( X_1 \) merges for the first time with \( X_0 \) at time \( t' \). Afterwards, path \( X_2 \) follows the original path \( X_0 \). Therefore, we have assumed two paths, \( X_1 \) and \( X_2 \), having the same accumulated distance until \( t' \). However, these paths have a different accumulated distance from \( t' \) until the transmission end. And it is path \( X_1 \) which has a larger accumulated distance in relation to the path \( X_0 \) accumulated distance. The reason is that path \( X_2 \) follows exactly the same states as the original path \( X_0 \). Therefore, path \( X_2 \) is nearer to path \( X_0 \) than path \( X_1 \). Consequently, since it is always possible to find a path \( X_2 \) for any path \( X_1 \), none path diverging twice or more times from the original sequence can be the nearest path to the original sequence. Figure E-10 shows a graphical example.

![Figure E-10: Viterbi paths \( X_0, X_1 \) and \( X_2 \)](image)

The previous statement is only correct when all the paths begin and end at the same states. Therefore, if all the paths do not begin and do not end at the same state, nothing avoids that a random path is closer to the original path in terms of accumulated distance than the nearest path described previously. Figure E-11 shows a graphical example.

![Figure E-11: Different possible Viterbi paths when the initial and the final state are not imposed](image)
E. Decoding methods

E.4.1.3. Search process of the \(i^{th}\) candidate

The search of the \(i^{th}\) candidate is accomplished by following the next 3 steps:

1. Searching among the initial survivors, which are yet not candidates, the one with the minimal accumulated distance.
2. Searching the minimum alternative path of each existent candidate.
3. Searching between the step 1 initial survivor and step 2 minimum alternative paths the sequence with the minimum accumulated distance. This sequence is the \(i^{th}\) candidate.

The justification is presented next. Nevertheless, in order to simplify the comprehension, we begin by explaining the case where the initial and final states are imposed, which means that there is only initial survivor of the Viterbi decoding.

The 1\(^{st}\) candidate is the only initial survivor of the Viterbi decoding since this sequence has the smallest accumulated distance. The 2\(^{nd}\) candidate is the minimum alternative path of the 1\(^{st}\) candidate, because as shown in section E.4.1.2, the nearest sequence to another one is a sequence which only diverges and merges once from the original sequence. And the minimum alternative path fulfills this requirement. The 3\(^{rd}\) candidate is searched from the 1\(^{st}\) and 2\(^{nd}\) candidates for two reasons. First, the 1\(^{st}\) candidate forbids the search of any sequence merging at the state where the 1\(^{st}\) and 2\(^{nd}\) candidate merged. Second, nothing avoids that the 2\(^{nd}\) candidate minimum alternative path has a smaller accumulated distance than the accumulated distance of the new minimum alternative path of the 1\(^{st}\) candidate. In fact, the 2\(^{nd}\) candidate travels some states which are different from the states of the 1\(^{st}\) candidate. Therefore, the state difference of the different states of 2\(^{nd}\) candidate from the 1\(^{st}\) candidate can be quite lower than any state difference of the states of the 1\(^{st}\) candidate. Figure E-12 shows a graphical explanation.

![Figure E-12: a) Possible candidates from the 1\(^{st}\) Candidate. b) Possible candidates from the 2\(^{nd}\) Candidate.](image)

The same justification can be applied for the search of the following candidates. When the initial and final states are imposed, the search of the \(k^{th}\) candidate is achieved following two steps. First, we search for the minimum alternative path of the candidates 1 to \((k-1)\). Second, we choose among the minimum alternative path of the candidates, the minimum alternative path having the smallest accumulated distance as the \(k^{th}\) candidate. Note that each time that a candidate is selected, the merging state between the new candidate and the candidates from which this new candidate is created is eliminated from the search of the next candidates.
The explanation of the case where the initial state is unknown but the final state is known is presented next. In this case, the candidates can begin at any Viterbi state but they still have to end at the same known state. Therefore, there is still only one initial survivor which is selected as the 1st candidate. From this 1st candidate, the 2nd candidate is searched following exactly the same process as in the known initial and final states case. The reason is the following. The minimum alternative path of the 1st candidate continues to be the nearest sequence in accumulated distance terms to the 1st candidate. Besides, the initial state knowledge does not change the manner of searching a candidate minimum alternative path. In fact, the part of a sequence which is previous to its merging state with a candidate is irrelevant from the accumulated distance point of view since its influence is already observed with the state difference. And the initial state only has influence on this part of the path. Therefore, it does not matter whether before the merging state the minimum alternative path diverged from the candidate or whether it simply was a completely different sequence; the only important element for the accumulated distance calculation is the state difference. Consequently, the initial state knowledge does not modify the 1st candidate search; it only increases the number of possible transmitted sequences.

The case where the initial and final states are unknown is presented next. The main difference between the case where the final state is known and the case where the final state is unknown is the number of initial survivors. In the former case, there is only one initial survivor, whereas in the latter case, the number of initial survivors is equal to the number of states. This means that the choice of the 1st candidate is made by choosing among the initial survivors, the survivor having the smallest accumulated distance. The search process of the 2nd candidate is slightly modified from the process presented before. In this case, this process search cannot only consists in searching the minimum alternative path of the 1st candidate, since it cannot be guaranteed that this sequence is the closest one in accumulated distances terms to the 1st candidate. In fact, as said in section E.4.1.2, any of the remaining initial survivors can be nearer to the 1st candidate. Note that, an initial survivor can travel a 1st candidate path part or can follow a completely different path. Therefore, the search process needs to find the minimum alternative path of the 1st candidate (the nearest path to the 1st candidate if the final state was known) and needs to compare its accumulated distance to the accumulated distance of the other initial survivors. Then, the path having the smallest accumulated distance between the remaining initial survivor having the smallest accumulated distance and the minimum alternative path of the 1st candidate is the path which has the closer accumulated distance to accumulated distance of the 1st candidate. Therefore, this path is claimed as the 2nd candidate.

One important remark is that if an initial survivor is chosen as the 2nd candidate, the search of the future candidates employs this initial survivor as any other candidate. The reason is that we cannot guarantee that the minimum alternative path of the 2nd candidate has a larger accumulated distance than any minimum alternative path calculated from the 1st candidate or other candidates generated from it. The same is true for any other initial survivor chosen as candidate.

Finally, the justifications of the last two paragraphs can be applied for of any candidate from the 3rd candidate.

**E.4.2. Generation of the i\textsuperscript{th} candidate**

The generation process of the i\textsuperscript{th} candidate consists in forcing the i\textsuperscript{th} candidate survival during the Viterbi decoding process.
In this section, the generation process of the $i^{\text{th}}$ candidate is presented. Nevertheless, before presenting the exact method, the objectives of the method and some definitions are given. Moreover, a detailed definition of the key element named restriction is also given.

E.4.2.1. Generation of the $i^{\text{th}}$ candidate objectives

The main objective of the $i^{\text{th}}$ candidate generation is to find the states travelled by the $i^{\text{th}}$ candidate. From the $i^{\text{th}}$ candidate travelled states, the information bits of the sequence can be extracted and the outer channel code can be applied in order to determine if this sequence is the transmitted one.

The generation of the $i^{\text{th}}$ candidate has another fundamental objective. This objective is to segment the sequence space in order to simplify the search of the next candidates. We define the sequence space as the ensemble of possible coded sequences. This means that when the $i^{\text{th}}$ candidate is generated, the sequence space associated to the merging state of the $i^{\text{th}}$ candidate and the candidate from which the $i^{\text{th}}$ candidate is generated is divided into two parts. One part is associated to the $i^{\text{th}}$ candidate. The other part is associated to the candidate from which the $i^{\text{th}}$ candidate is generated. Therefore, the sequences belonging to the sequence space associated to the $i^{\text{th}}$ candidate can only be found by inspecting the $i^{\text{th}}$ candidate. This means that the generation process of the $i^{\text{th}}$ candidate has to introduce some mechanism in order to avoid the repetition of the same segmentation of the sequence space and in order to avoid the invasion of the sequence space of the $i^{\text{th}}$ candidate by another candidate.

Nevertheless, note that some candidates can generate the same sequence since they can have the same minimum alternative path. And this fact can lead to the false impression that both candidates share the same segmented sequence space. More specifically, a sequence can be recovered from the $n^{\text{th}}$ candidate and from the candidate from which the $n^{\text{th}}$ candidate was generated; however, not all the state differences are the same for the sequence generated by the $n^{\text{th}}$ candidate as for the sequence generate by the candidate from which the $n^{\text{th}}$ candidate was generated. In fact, the segmentation of a sequence space refers to segment the sequence space of the sequences travelling a given state at a given time. This means that two or more candidates share a part of the general sequence space whereas they hold individually another part of the sequence space: no other candidate has this individual part of the segmented sequence space.

Apart from the two previous main objectives, there are two more requirements of the generation process of the $i^{\text{th}}$ candidate which are also necessary in order to allow the correct functioning of this new proposed method.

First, in order to achieve the search process of the $i^{\text{th}}$ candidate, the receiver had to employ the state differences of the states of the previous different candidates as specified in section E.4.1. However, the search process of the $i^{\text{th}}$ candidate never explained how the state differences of the candidates were calculated. It simply assumed that their values were known beforehand. In fact, the $i^{\text{th}}$ candidate state differences can only be calculated when the $i^{\text{th}}$ candidate is forced to survive the Viterbi decoding process since it is the only moment when the accumulated distances of the paths merging with the $i^{\text{th}}$ candidate can be compared to the accumulated distance of the $i^{\text{th}}$ candidate. Therefore, it is during the generation of the $i^{\text{th}}$ candidate when the state difference of the states of the $i^{\text{th}}$ candidate must be stored.

Second, the generation of the $i^{\text{th}}$ candidate is not easily achieved. In fact, if the previous candidates from $i$ equal to 1 to $i$ equal to $i-1$ are not eliminated during the generation of the $i^{\text{th}}$ candidate, these candidates can cause two major complications. First, some of these previous
candidates can eliminate the $i^{th}$ candidate because they merge at some state $X$ with the $i^{th}$ candidate and they have a smaller accumulated distance. Therefore, the sequence recovered after the Viterbi decoding process is not the $i^{th}$ candidate. In other words, the method needs to correctly place itself inside the desired segmented sequence space in order to recover the desired $i^{th}$ candidate. Second, some of these previous candidates can eliminate some sequence which could be a potential candidate because these previous candidates have a smaller accumulated distance. Therefore, in order to recover the correct $i^{th}$ candidate and correctly store the differences of the states travelled by the $i^{th}$ candidate, the previous candidates should be eliminated. In other words, the method has to avoid the elimination of any sequence belonging to the sequence space associated to the $i^{th}$ candidate.

The elimination of previous candidates from the generation process of the $i^{th}$ candidate generation can lead to the false conclusion that potential candidates merging at any state with these previous candidates are lost. This is not true because these potential candidates of the previous candidates are inspected during the search process of the minimum alternative path of the previous candidates. In other words, each possible candidate specifically belonging to a determined sequence space of a candidate is only inspected through the candidate.

**E.4.2.2. Definitions of the generation of the $i^{th}$ candidate**

We call the candidate from which the $i^{th}$ candidate is generated the *father* of the $i^{th}$ candidate $x$. In other words, the $i^{th}$ candidate is a minimum alternative path of its father. We also call a candidate generated from the $i^{th}$ candidate the *descendant* of the $i^{th}$ candidate. This means that an initial survivor does not have a father but can have zero, one or more than one descendants.

Additionally, the *ancestors* of the $i^{th}$ candidate are the father of the $i^{th}$ candidate, the father of the father of the $i^{th}$ candidate, and so on until the initial survivor from which the candidates have been generated. Therefore, the father of the father of the $i^{th}$ candidate is called the *grandfather* of the $i^{th}$ candidate.

Finally, a *restriction* is an element associated to a couple state-time which chooses the survivor sequences between the two sequences merging at the state and at the time marked by the restriction. The restrictions are the elements used to segment a sequence space and to associate each resulting segment to a determined candidate. A more detailed definition is given in section E.4.2.5.

**E.4.2.3. Generation process of the $i^{th}$ candidate**

The generation process of the $i^{th}$ candidate is achieved by applying the Viterbi decoding process but forbidding the survival of the previous generated candidates. In other words, the receiver conducts a new Viterbi decoding process which removes the paths having a smaller accumulated distance than the $i^{th}$ candidate and which does not affect at all the paths having a larger accumulated distance than the $i^{th}$ candidate.

The method proposed consists in applying the Viterbi algorithm while:

1. Reproducing the exact conditions that allowed the generation of the father of the $i^{th}$ candidate.
2. Adding some restrictions about other descendants of the father of the $i^{th}$ candidate which were created between the generation of the father and the $i^{th}$ candidate.
3. Forcing the receiver to choose the $i^{th}$ candidate instead of its father at their merging state.

Note that the application of the previous points is in reality a segmentation of the sequence space. The justifications of the 3 points are presented next.

First, the exact generation of the father of the $i^{th}$ candidate at the moment of its generation is necessary in order to guarantee that the $i^{th}$ candidate is equal to the path anticipated by the search process of the minimum alternative path of the father. In fact, if the father is not rigorously reproduced, it is possible that previous candidates -which were already eliminated at the generation of the father- merge with the $i^{th}$ candidate. Therefore, these previous candidates eliminate the $i^{th}$ candidate due to their smaller accumulated distances which means that a previous candidate is reproduced instead of the $i^{th}$ candidate.

Moreover, if the generation of the father of the $i^{th}$ candidate at the moment of its generation is not exactly reproduced, some other sequences with larger accumulated distances than the $i^{th}$ candidate can be erased by these previous candidates. Therefore, these other sequences are never examined since these previous candidates are eliminated later. This means that these other sequences will never be inspected even if they are the transmitted sequence. Besides, if the paths merging with the father are not the expected ones, the previous calculated state differences are false and the final accumulated distances are wrongly estimated.

In other words, the method needs to recover the original segmented sequence space of the father of the $i^{th}$ candidate in order to find the $i^{th}$ candidate. If the exact segmented space is not recovered, the $i^{th}$ candidate and any other sequence can be modified or eliminated. Therefore, the further segmentation of the examined sequence space is not the correct one. And this leads to the loss of some sequence space segmentations and to the no inspection of some sequences.

To sum up, the exact generation of the father of the $i^{th}$ candidate at the moment of its generation is required. Or, in other words, the exact regeneration of the conditions allowing the generation of the father of the $i^{th}$ candidate is necessary in order to obtain the $i^{th}$ candidate, in order to inspect any possible path, in order to obtain the correct state differences of the father and the $i^{th}$ candidate, and in general, in order to recover the segmented sequence space of the father at the moment of its generation.

Additionally, note that in order to exactly generate the father at the moment of its generation, it is necessary to regenerate rigorously the grandfather at the moment of its generation too; more specifically, the moment where the grandfather state differences were calculated. Therefore, the generation of the $i^{th}$ candidate is achieved by reproducing all the ancestors of the $i^{th}$ candidate at the moment of their generation: the father, the grandfather, the father of the grandfather, etc, until an initial survivor is reached. Or, in other words, the desired segmented sequence space of the father is reproduced by starting in the general sequence space and by descending later to the specified segment.

Second, the restrictions imposed by other descendants of the father of the $i^{th}$ candidate which were generated between the generation of the father and the generation of the $i^{th}$ candidate, are required in order to avoid unnecessary candidate repetitions and in order to prevent the elimination of the $i^{th}$ candidate by the other descendants of the father. First, when a new candidate is generated, the receiver has to mark on its father the state where the new candidate and the father merge. The marking consists in introducing a restriction in the determined couple state-time. In fact, if this state is not marked, the receiver will select again the merging state of the new candidate as the new minimal state during the search of the next candidate. Therefore, an unnecessary candidate repetition occurs and the process enters in a deadlock. Second, when the merging state of a previous descendant of the father is marked but it is not
taken into account in the generation of the $i^{th}$ candidate, the $i^{th}$ candidate is eliminated when it merges with the previous descendant at the previous marked merging state. The reason is that the previous descendant has a smaller accumulated distance than the $i^{th}$ candidate. Therefore, the $i^{th}$ candidate generation fails since the recovered sequence is not the desired one. Figure E-13 shows an example where if $X_1$ path is not restricted during generation of $X_2$ path from $X_0$ path, path $X_2$ is erased by $X_1$ path.

![Figure E-13: $i^{th}$ candidate elimination from previous $i^{th}$ candidate father descendant](image)

Moreover, note that this same type of restriction has to be also added to the generation of the grandfather in order to obtain the father at the moment of its generation since it cannot be guaranteed that the father is the first descendant of the grandfather. This means that the restrictions imposed by the generation of the previous descendants to the father of the grandfather of the $i^{th}$ candidate have to be imposed to the father in order to allow its generation. These restrictions should be included into the first type of restrictions since they are necessary to reproduce the father, grandfather, etc.

Another form to interpret the second requirement is to think about the sequence space segmentation conducted with each descendant. In fact, each time that the father of the $i^{th}$ candidate creates a descendant, the segmented sequence space associated to the father is divided into two parts. One part is associated to the father and the other to the descendant. Therefore, the segmented space associated to the father when the $i^{th}$ candidate is generated has to be recovered from the father segmented space when the father had none descendants. And from this latter sequence space, the method needs to find the exact part associated to the father after the space sequence division among the previous descendants of the father of the $i^{th}$ candidate.

Third and last, we force the Viterbi decoding process to make survive the $i^{th}$ candidate instead of its father. This action is equivalent to divide the segmented space of the father into two and to associate one part to the $i^{th}$ candidate and the other to its father.

### E.4.2.4. Generation algorithm of the $i^{th}$ candidate

The generation algorithm of the $i^{th}$ candidate can be divided into two parts. The first part consists in executing the Viterbi algorithm with the restrictions commented in section E.4.2.3 in order to obtain the correct state differences of each travelled state. The second part consists in adding to the $i^{th}$ candidate and to its father the new restrictions imposed by the $i^{th}$ candidate in order to prepare the search and generation processes of the $(i+1)^{th}$ candidate. In other words, this second part defines the new segmented sequence spaces associated to $i^{th}$ candidate and to its father. A more detailed explanation is given below.
The first part conducts the traditional Viterbi decoding method but with the particularity of making survive at the couple state-time marked by a restriction, the sequence indicated by this restriction. This means that all the restrictions conditioning the $i^{th}$ candidate generation must be found before applying the Viterbi decoding method with restrictions. These restrictions are the following:

- Restrictions allowing the selection of the $i^{th}$ candidate instead of the selection of any previous descendant of the father of the $i^{th}$ candidate.

- Restrictions used to generate the father of the $i^{th}$ candidate at the moment of its generation:
  - Restrictions allowing the selection of the father of the $i^{th}$ candidate instead of the selection of any previous descendant -to the father- of the grandfather of the $i^{th}$ candidate.
  - Restrictions used to generate the grandfather of the $i^{th}$ candidate at the moment of its generation:
    - …
    - Restrictions allowing the selection of the father of father… of the $i^{th}$ candidate instead of the selection of any previous descendant -to the father of the father…- of the initial survivor of the $i^{th}$ candidate.
    - All restrictions used to generate the initial survivor; which are none.
- Restriction selecting the $i^{th}$ candidate instead of its father at the merging state.

Nevertheless, the previous scheme can be simplified by sorting out the previous restrictions into two groups, where each group of restrictions is individually stored inside a candidate. These groups are the restrictions defining the generation conditions of a candidate and the restrictions marking the descendants of a candidate.

First, the restrictions defining the generation conditions of a candidate are in reality the second group of restrictions of the father of the candidate at the moment of generation of the candidate. Therefore, the first group of restrictions of a candidate is only stored at the moment of its generation. Additionally, note that the restrictions defining the generation conditions of the father are the second group of restrictions of the grandfather of the candidate at moment of generation of the father. The same can be said for the restrictions defining the generation conditions of the grandfather and son on until arriving at the candidate who is an initial survivor. Therefore, if whenever a candidate is generated, the current second group of restrictions of its father is stored as the first group of restrictions of the candidate, the regeneration of any candidate is easily accomplished since all its ancestors have already stored the restrictions allowing their own regeneration.

In other words, the first group of restrictions defines the segmented sequence space to be divided between the $i^{th}$ candidate and its father. And this segmented sequence space to be divided is defined from the segmented sequence space of the father at the moment of its generation.

Second, the restrictions marking the candidate descendants have two different origins. The first origin is that a descendant cannot be generated at a given state and at a given time because the state of the candidate is already marked as a merging state with an ancestor. Or, in other words, the receiver can only generated an ancestor from this state. Consequently, these restrictions are extracted from the second type of restrictions of the father of the candidate at the moment of generation of the candidate, but note that not all the restrictions are stored. In
fact, if the father has a restriction imposed at a certain time \( t \) at a state \( A \) and the candidate is not at this time \( t \) at the state \( A \), this restriction does not have to be applied to the candidate. The reason is that the candidate at this time \( t \) is at another state \( B \) which merges with a still not inspected sequence; a sequence which can be a future descendant of the candidate. Therefore, this inclusion of this restriction in the candidate removes in reality this un inspected sequence instead of an ancestor of the candidate. Moreover, since it is possible that the father of the candidate does no travel all the state of the candidate although some ancestors can, the restrictions of the first origin have to be also searched among all the candidate ancestors first group of restrictions in order to avoid the unnecessary regeneration of any ancestor.

The restrictions of the second origin are the descendants generated by the candidate. In this case, the receiver seeks to avoid future unnecessary candidate repetitions. Therefore, the restrictions of the second origin are stored along the generation of the descendants of the candidate.

In other words, the second group of restrictions defines the current segmented sequence space associated the \( i^{th} \) candidate. This current segmented sequence space is defined from the segmented sequence space of the \( i^{th} \) candidate at the moment of generation of the \( i^{th} \) candidate.

Finally, the search of the restrictions for the generation of the \( i^{th} \) candidate is thus reduced to:

- Second group of restrictions of the father of the \( i^{th} \) candidate.
- First group of restrictions of the father of the \( i^{th} \) candidate.
- First group of restrictions of the grandfather of the \( i^{th} \) candidate.
- …
- Restriction selecting the \( i^{th} \) candidate instead of its father at the merging state.

Once the restrictions allowing the generation of the \( i^{th} \) candidate, the second part of the generation algorithm of the \( i^{th} \) candidate is explained. This part consists in storing the two previous defined groups of restrictions each time that a new candidate is generated. More specifically, the receiver has to store the following restrictions:

- **On the second group of restrictions of the father of the \( i^{th} \) candidate:** the restriction signaling the \( i^{th} \) candidate-father couple merging state-time. We associate to the father its new segmented sequence space after the generation of the \( i^{th} \) candidate.

- **On the first group of restrictions of the \( i^{th} \) candidate:** the current second group of restrictions of the father. We store into the \( i^{th} \) candidate the segmented sequence space of its father before dividing the sequence space and associating a part to the \( i^{th} \) candidate. These restrictions are never modified and none restriction can be added to them.

- **On the second group of restrictions of the \( i^{th} \) candidate:** the restriction signaling the \( i^{th} \) candidate-father couple merging state-time. We associate to the \( i^{th} \) candidate its first segmented sequence space.

- **On the second group of restrictions of the \( i^{th} \) candidate:** the current second group of restrictions of the father which has a couple state-time travelled by the \( i^{th} \) candidate. We associate to the \( i^{th} \) candidate its first segmented sequence space.

- **On the second group of restrictions of the \( i^{th} \) candidate:** The first group of restrictions of the father and of the ancestors of the \( i^{th} \) candidate which has a couple state-time travelled by the \( i^{th} \) candidate. We associate to the \( i^{th} \) candidate its first segmented sequence space.
E.4.2.4.1. Example

This example illustrates the algorithm of the previous section. First, we assume that a \( k^{th} \) candidate is generated from the \( 1^{st} \) candidate and that the restrictions of the \( 1^{st} \) candidate marking its previous descendants to the \( k^{th} \) candidate -which have been used to generate the \( k^{th} \) candidate- are stored into the \( k^{th} \) candidate. Second, we assume that another \( n^{th} \) candidate is generated from the \( k^{th} \) candidate where \( n > k \). In this case, the restrictions of the \( 1^{st} \) candidate stored into the \( k^{th} \) candidate have been used to recreate the exact \( k^{th} \) candidate generation and the own restrictions of \( k^{th} \) candidate previous to the generation of the \( n^{th} \) candidate have been employed to mark the previous descendants of the \( k^{th} \) candidate. Then, once the \( n^{th} \) candidate has been generated, these restrictions of the descendants of the \( k^{th} \) candidate previous to the \( n^{th} \) candidate are stored inside the \( n^{th} \) candidate. Finally, we assume the generation of the \( m^{th} \) candidate from the \( n^{th} \) candidate where \( m > n > k \). In this case, the employed restrictions have been first the restrictions of the \( 1^{st} \) candidate stored into the \( k^{th} \) candidate. These restrictions allow the exact regeneration of the \( k^{th} \) candidate. Second, the restrictions of the \( k^{th} \) candidate stored into \( n^{th} \) candidate are used in order to generate the \( n^{th} \) candidate. In fact, once the \( k^{th} \) candidate has been regenerated, these previous restrictions allow the exact regeneration of the \( n^{th} \) candidate from the \( k^{th} \) candidate. Third and last, the restrictions of the descendants of the \( n^{th} \) candidate previous to the \( m^{th} \) candidate are used in order to generate the \( m^{th} \) candidate.

E.4.2.5. Restrictions

A restriction is an element that determines which sequence between two sequences entering a specific couple state-time has to survive. Or, in other words, a restriction privileges one path among all the others paths entering a state. Therefore, the restrictions are used in order to allow the survival of a path with a larger accumulated distance than another path entering into a state.

The restriction is the element employed by the proposed method which is responsible for dividing the initial sequence space into smaller sequence space segments. Therefore, the structure of a restriction is defined in order to segment the sequence space. In fact, the segmentation of the sequence space is applied to each individual couple state-time. This means that the method seeks to separate the sequence which enters into state \( S \) at time \( t \) coming from state \( S' \), from the sequence which enters state \( S \) at time \( t \) coming from state \( S'' \). Therefore, a restriction consists of 3 elements: the time \( t \) when the restriction is applied, the state \( S \) where the restriction is applied, and the state \( S^-1 \) from which the survival path enters the merging state.

E.4.3. Duration of the Viterbi decoding and bit recovery

A convolutional code can have as input an infinite stream of bits. This means that the receiver cannot wait until the transmission end, which does not exist, before beginning the decoding process.

The application of the traditional Viterbi algorithm can guarantee with a high probability the convergence of a part of all the survivors sequences at a determined transmission instant as said in annex E.3.2.2. More specifically, the Viterbi algorithm can guarantee with a high probability that the \( i^{th} \) symbol of all the survivors sequences is the same when the \( (i+5\cdot L)^{th} \) symbol is being received. Obviously, this probability increase for received symbols previous to the \( i^{th} \) symbol.
Therefore, since the proposed method is basically a Viterbi modification, the proposed method has to extend its Viterbi decoding until all the bits belonging to the same coded word have been determined with a high certainty. In other words, for the GPS L2C and GPS L5 navigation messages, the number of received symbols to process in order to ensure that the value of all the bits of a coded word have been provided with a high certainty by the Viterbi decoding process is \( M + 5 \cdot L \), where \( M \) depends on the decoded message. More specifically, \( M \) is equal to 300 bits for GPS L5 and GPS L2C mode CNAV data, and \( M \) is equal to 30 bits for GPS L2C mode NAV data with FEC.

Finally note that, due to the inclusion of the additional 5 \( \cdot L \) bits in the Viterbi decoding process, the decoding process of the \( n \)th coded word requires the use of the bits of the next coded word. More specifically, for GPS L5 and GPS L2C mode CNAV data, some \((n+1)\)th coded word bits are required whereas for GPS L2C mode NAV data with FEC all the \((n+1)\)th coded word bits are used plus some \((n+2)\)th coded word bits. Nevertheless, remember that only the \( M \) bits of the \( n \)th coded word are actually decoded.

### E.4.4. The Algorithm

The detailed algorithm of the inner and outer channel code combination method is presented next. The part of the algorithm concerning the verification of the outer code is not presented since this process is explained in annex E.1 and E.2. The maximum number of searched candidates before stopping the research and discarding the received word is set to \( K \).

The algorithm is described below:

- **Initialization:**
  1. **Array of Distances:** \( 2^{L-1} \) double values to store the accumulated distances of the initial survivors.
  2. **Array of Survivors:** \( 2^{L-1} \) Booleans to mark the initial survivors which have already been used as candidates.
  3. **Matrix of Differences:** \((M + 5 \cdot L)\) columns, \( K \) rows. Double values to store the state differences of the candidates.
  4. **Matrix of Candidates:** \((M + 5 \cdot L)\) columns, \( K \) rows. Integer values to store the \( K \) travelled states of the candidates.
  5. **Matrix of Descendants:** \((M + 5 \cdot L)\) columns, \( K \) rows. Booleans to mark the descendants of the \( K \) candidates.
  6. **Matrix of Generated:** \((M + 5 \cdot L)\) columns, \( K \) rows. Booleans to mark the descendants of the father of the candidate at the moment of generation of the candidate.
  7. **Array of Accumulated Distances:** \( K \) doubles values to store the accumulated distances of the \( K \) candidates.
  8. **Array of Fathers:** \( K \) integer values to store the identifier of the father of the candidate.

- **Searching and creating the 1st Candidate:**
  1. Complete Viterbi decoding process execution: \((M + 5 \cdot L)\) received symbols.
  2. Store the accumulated distance of the initial survivors at the Distances array.
3. Select the initial survivor with the smallest accumulated distance as the 1\textsuperscript{st} candidate.
4. Store the state differences of the 1\textsuperscript{st} candidate into Differences matrix row 1.
5. Store the travelled states of the 1\textsuperscript{st} candidate into Candidates matrix row 1.
6. Store the accumulated distance of the 1\textsuperscript{st} candidate into Accumulated Distances array cell 1.
7. Store the value -1 into Fathers array cell 1; marking a no valid father.

- **Searching the i\textsuperscript{th} Candidate**
  1. Loop: k = 1 to (i-1)
  1.1. Search for the minimum alternative path of the k\textsuperscript{th} candidate which is not yet a descendant. In other words, searching the minimal state of the k\textsuperscript{th} candidate.
    1.1.1. Initialize variable MIN to a big number e.g. 1000000000.
    1.1.2. Loop: j = 1 to (M + 5\cdot L)
      1.1.2.1. Verify that the state travelled by the k\textsuperscript{th} candidate at time j has not already produced a descendant (inspect Descendants matrix row k column j).
        1.1.2.1.1. Verification fails: the system jumps to explore the next state.
        1.1.2.1.2. Verification succeeds: the system compares the state difference value stored into Differences matrix row k column j to variable MIN. If (MIN > state difference), the state and its state difference are stored, and MIN = state difference.
    1.1.3. The minimum alternative path of the k\textsuperscript{th} candidate is defined by:
      1.1.3.1. Accumulated distance: Accumulated Distances matrix cell k value plus MIN variable value.
      1.1.3.2. Merging state S: Candidates matrix row k column j.
      1.1.3.3. Previous merging state S\textsuperscript{-1}: Candidates matrix row k column (j-1).
      1.1.3.4. Candidate final state S\textsubscript{f}: Candidates matrix row k column (M+5\cdot L).
  2. Search for the minimum alternative path with the smallest accumulated distance among all previously found minimum alternative paths in step 1.
  3. Search for the initial survivor with the smallest accumulated distance among all the initial survivors still not being a candidate: the Survivors array cell marks if the initial survivor is already a candidate.
  4. Choose the sequence with the smallest final accumulated distance between the step 2 and step 3 paths as the i\textsuperscript{th} candidate.

- **Generate the i\textsuperscript{th} Candidate from the minimum alternative path of the k\textsuperscript{th} candidate:**
  Each time that a restriction is recovered, the system has to recover the time t, the merging state S and the state previous to the merging state S\textsuperscript{-1} from the corresponding sequence states stored inside the Candidates matrix.
  1. Recover the restrictions used to generated the k\textsuperscript{th} candidate:
E. Decoding methods

a. Active-Father is the father of the $i^{th}$ candidate (the $k^{th}$ candidate).

b. 1) If Active-Father is not the $1^{st}$ candidate, the system recovers all the restrictions of the father of Active-Father marked on the Active-Father Generated matrix row. The states S and S-1 are searched inside the father of Active-Father Candidates matrix row.

2) If Active Father is the $1^{st}$ candidate, the restrictions search is over and the system goes to step 2.

c. Active-Father is the Active-Father father.

d. Going to step b.

2. Recover the restrictions of the $k^{th}$ candidate marked into Descendants matrix row k. The states $S'$ and $S'^{-1}$ are searched inside the $k^{th}$ candidate Candidates matrix row. Add only the restrictions that have not been recovered in step 1.

3. Add the restriction where the $i^{th}$ candidate eliminates its father: time $t$, merging state $S$ and previous state $S'^{-1}$.

4. Store the father identifier $id_f$. This identifier marks the row where the father characteristics are stored into the different several matrices.

5. Store the father final state $S_f$.

6. Complete execution of the modified Viterbi decoding process: $(M + 5\cdot L)$ received symbols. This execution differs from the normal one due to the restrictions imposed by the previous points:

6.1. Restrictions used to generate the $k^{th}$ candidate (Step 1 and 2 restrictions): The restrictions impose that the survivor path entering the merging state $S$ at time $t'$ is the sequence that comes from the previous state $S'^{-1}$.

6.2. $i^{th}$ candidate restriction (Step 3 restriction): The restriction imposes that the survivor path entering the merging state $S$ at time $t$ is the sequence that does not come from the previous state $S'^{-1}$.

7. Save into Candidates matrix row i the states of the path arriving at state $S_f$ ($i^{th}$ candidate).

8. Save into Differences matrix row i the state differences of the $i^{th}$ candidate.

9. Marking into Descendants matrix row k column t that state S has been used to generate the $i^{th}$ candidate. Row k stores the information of the descendants of the father of the $i^{th}$ candidate.

10. Copy Descendants matrix row k, the restrictions of the descendants of the father of the $i^{th}$ candidate, to Generated matrix row k.

11. Mark into Descendants matrix row i all the cells corresponding to a couple state-time travelled by the $i^{th}$ candidate where a restriction has been applied. Note that the time specifies the cell inside the row i where the restriction has to be marked.

12. Storing into the Accumulated Distances array cell i the accumulated distance of the path arriving at the state $S_f$ ($i^{th}$ candidate).

13. Storing into Father array cell i the identifier of the father of the $i^{th}$ candidate, $id_f$. 


E. Decoding methods

- Generate the $i^{th}$ Candidate from an initial survivor:
  1. Identify the final state of the initial survivor, $S_f$.
  2. Complete execution of the Viterbi decoding process: $(M + 5 \cdot L)$ received symbols.
  3. Store the travelled states of the initial survivor into Candidates matrix row $i$.
  4. Store the states differences of the states travelled by the initial survivor into Differences matrix row $i$.
  5. Store into Accumulated Distances array cell $i$ the accumulated distance of the initial survivor.
  6. Store into Father array cell $i$ the value -1 marking that the initial survivor, or $i^{th}$ candidate, does not have a father.
  7. Mark into Survivors array cell $S_f$ that the initial survivor having as final state $S_f$ has been used as candidate.

E.4.5. Variations of the decoding method

In this section, the 6 variations of the decoding method based on the inner and outer channel codes variation are detailed.

E.4.5.1. Viterbi method with a priori bit probabilities

The inner and outer channel code combination method implements the Viterbi algorithm as the inner channel code. Therefore, any method improving the Viterbi algorithm decoding should improve the proposed method performance. In this case, we propose to introduce the use of the a priori bit probabilities as explained in section 5.2.1.3.

Remember that the use of a priori bit probabilities helps the Viterbi decoding to obtain a lower BER than the BER obtained with the traditional technique. Therefore, in this case, the a priori bit probabilities increase the probability of finding the transmitted sequence among the inspected candidates. In other words, due to the use of a priori bit probabilities, after the Viterbi decoding process the receiver observes, in average, fewer sequences than before which have a smaller accumulated distance than the actual transmitted sequence. Therefore, the proposed method has a higher probability of finding the transmitted sequence in addition to finding the sequence more quickly.

This variation can only be included in the GPS L2C mode NAV data with FEC signal since it is the only signal which has the navigation data message encoder equal to the GPS L1 C/A navigation data message before being fed to the convolutional.

E.4.5.2. Sliding Window

The variation of the algorithm proposed in this section is relative to the Viterbi decoding length and the decoded bits. Therefore, before presenting the modification, the reason why the change of the Viterbi decoding length improves the method decoding performance is given.

The main algorithm proposes to apply the Viterbi decoding over a code word length, $M$ symbols, plus $5 \cdot L$ additional symbols. Remember that a code word is a set of bits protected by the outer channel code. The reason of this Viterbi decoding length is to reach the convergence
of all the survivor sequences over the M symbols to be decoded before estimating the symbols values. However, we could search for another decoding length which optimizes the decoding performance in addition to guaranteeing the symbols convergence.

In fact, after examining the source of errors of the proposed method, we propose to search for the most probable sequence spanning N coded words which verifies the outer channel code of each one of the N code words. Note that the expansion of the candidate from 1 code word to N code words is possible since at the end of each code word the calculation of the accumulated distance of the sequence is not interrupted and the final state is not fixed. This new decoding length, \((N \cdot M + 5 \cdot L)\) symbols, implies an increase of the computational cost of the process since the number of possible sequences grows exponentially with the number of decoded symbols \((N \cdot M)\). This means that this algorithm variation depends on the trade-off between the decoding performance gain and the extra computational cost.

The justification of why the decoding performance is improved is explained by analyzing the source of errors of the method. The source of errors of the method is the limitation of the outer channel code to detect any possible pattern of errors contained by a candidate, i.e. its detection capacity. Note that the Viterbi algorithm does not introduce single independent errors but bursts of errors. Consequently, if we consider a candidate as the transmitted sequence plus a pattern of errors, we can have wrong sequences verifying the outer channel code due to the previous commented limitation. This means, that the search for the transmitted sequence is stopped and the wrong code word is provided to the GPS receiver.

Two different patterns of errors can be distinguished depending on their position inside the wrong code word: at the middle of the code word or at the code word edges. First, the pattern of errors located at the code word edges implies that either the initial state of the code word, the final state of the code word or both states are different from the correct states of the code word. Moreover, this pattern of errors also affects the previous or posterior code word. Figure E-14 shows an example of a burst of errors located at two consecutive code words edges.

![Figure E-14: Pattern of errors situated at the edge of 2 consecutive code words](image)

From Figure E-14, it can be seen that if the proposed decoding method is applied at the same time for both codewords, the shared pattern of errors can be corrected. In fact, when the pattern of errors is situated at the code word edge, it is quite probable that the candidate which spans on 1 codeword and which verifies the outer channel code of this 1\(^{st}\) code word fails the verification for the 2\(^{nd}\) codeword when it is extended to the 2\(^{nd}\) code word; the candidate is now the union of two code words. The justification of this previous statement is that the probability of two erroneous code words passing the outer channel code verification is much smaller than the probability of only one code word verifying the outer channel code.

Second, the patterns of errors located at the middle of the code word are the patterns of errors which do not have any influence on the initial and final code word states. Therefore, the initial and final states of the code are the same for the correct code word as for the erroneous code words. This means that the decoding of an extra word cannot help the correction of the pattern of errors. The justification is that since the final state of the 1\(^{st}\) code word is the same for the correct coded word as for the erroneous code word, the candidate extension until the end of
the 2\textsuperscript{nd} code word is the same for the correct code word as for the erroneous code word. Therefore, although the 1\textsuperscript{st} code word is wrong, the 2\textsuperscript{nd} word cannot contain any error.

One remark that can be made about the extension of the decoding process to for example 2 code words concerns the situation where the 1\textsuperscript{st} code word is correct but the 2\textsuperscript{nd} word is erroneous. In this case, the error on the 2\textsuperscript{nd} code word can avoid the correct decoding of the 1\textsuperscript{st} code word and thus the decoding performance gets worse. Nevertheless, it has been shown through simulations that the extension of the candidate to more than one word increases the decoding performance of the method. And this means that the decoding performance gain obtained through the increase of the probability of detection is higher than decrease of decoding performance due to the impossibility of recovering the 1\textsuperscript{st} correct code word if the 2\textsuperscript{nd} one cannot be corrected.

One last remark to make about the modification of the Viterbi decoding length is the number of bits that can be extracted from the single decoding of a candidate. In fact, each time that the decoding length is extended to N words and a candidate verifies the outer channel code of each one of the N code words, the algorithm variation should only recover the bits of the first N-1 code words. The reason is that the N\textsuperscript{th} code word can still have a pattern of errors located at its right edge. Therefore, in order to decode these bits of the N\textsuperscript{th} code word, the algorithm implements a sliding window containing several code words where the last code word is never decoded but is used for the search of the extended candidates. More specifically, N code words define a candidate but only the bits of the first (N-1) code words are decoded; after, the N\textsuperscript{th} code word plus the next (N-1) words define a new candidate and the process begins again.

The sliding window algorithm is given below:

- \textit{i\textsuperscript{th} window:}
  - Words (i-1)-(N-1)+1 to (i-1)-(N-1)+N define the new candidate
  - Search of the candidate defined by these N words which verifies the outer channel code of each one of the N code words.
  - Decoding the bits of the code words (i-1)-(N-1)+1 to (i-1)-(N-1)+(N-1)

Finally, this variation of the decoding method increases significantly the navigation message GPS L2C mode NAV data number of correct decoded words; however its implementation has been discarded for GPS L2C mode CNAV data and GPS L5 due to the low CRC-24Q probability of non-detection [ARINC, 2004].

\section*{E.4.5.3. Sequence initial state imposition}

This algorithm variation consists in imposing the initial state of the candidates. In other words, each time that a candidate passes the outer channel code verification, its final state is used as the initial state of the next candidate-to-search. However, if the previous candidate is not found, the initial state of the next candidate is not imposed.

The imposition of the initial state of a candidate implies that the number of sequences which can be candidates is reduced because only the sequences having their initial state equal to the imposed state are considered. Therefore, several advantages and one drawback are introduced by this fact.

The first advantage is that the possibility of providing the outer channel code with an erroneous code word passing its verification is lower. This means an improvement of the BER performance. The second advantage is that the probability of finding the transmitted sequence
among the K inspected candidates is increased. This also means an improvement of the BER performance. The third and final advantage is that since there are less erroneous code words between the 1\textsuperscript{st} candidate and the emitted sequence, the time of the decoding process is reduced.

The drawback of the algorithm variation is due to the impossibility of the channel outer code to detect all the erroneous code words. This means that when an erroneous code word passes the verification of the outer channel code, the imposed initial state to the next candidate-to-search is false. Therefore, the next transmitted sequence can never be found by the proposed decoding method since all the candidates to be inspected will have the same different initial state from the initial state of the transmitted sequence. This means an increase of the BER. Nevertheless, this drawback is neglected for GPS L2C mode CNAV data and GPS L5 due to the low CRC-24Q probability of missing a wrong word [ARINC, 2004].

This variation of the decoding method is implemented for all the signals, GPS L2C mode NAV data with FEC, GPS L2C mode CNAV data and GPS L5.

E.4.5.4. Algorithms variations speeding up the execution of the proposed method

The 3 algorithm variations presented in this section increase the execution speed of the combination of channel codes method. The first and most important consequence of applying any of the 3 variations is the impossibility of guaranteeing that the method is still able to find the correct transmitted sequence when assuming an infinite number of candidates and a perfect detection capacity of the outer channel code.

More specifically, the application of any speed improvement is equivalent to reduce the sequence space in which the transmitted sequence is searched. In other words, the receiver no longer search for the transmitted sequence among all the possible sequences, the receiver limits its search to a reduced quantity of sequences. These sequences discarded are considered too improbable to be the transmitted sequence. And this means that if the transmitted sequence is one of these beforehand discarded sequences, the receiver cannot find the transmitted sequence.

Fortunately, there is always more than one sequence which contains the bits of the transmitted codeword. And this means that not all the sequences containing the bits of the transmitted codeword are lost due to the application of the speed improvements. In fact, since the last 5-L bits used during the Viterbi decoding process are part of a candidate but only the previous M bits represent the bits of the transmitted code word, all the sequences having the first M bits equal to the bits of the transmitted code word are considered as valid sequences regardless of their last 5-L bits. Therefore, not only the transmitted sequence contains the bits of the transmitted code word. Remember that only the first M bits have to pass the outer channel code verification. Nevertheless, note that the actual transmitted sequence should be the first sequence to be inspected among all the sequences containing the bits of the transmitted code words since this sequence should have the smallest accumulated distance.

Moreover, due to the possible elimination of all the sequences containing the bits of the transmitted code word, the application of any speed improvement can imply the increase of the probability of providing an erroneous code word to the receiver. In fact, when the sequences containing the bits of the transmitted code word are eliminated, the method has to reach the maximum number of inspected candidates before determining that any sequence containing the bits of the transmitted code word cannot be recovered. And this means that there are more erroneous sequences being tested by the outer channel code verification which
leads to more erroneous code words than usual provided to the higher layers of the receiver. Nevertheless, if the transmitted sequence is not eliminated, the effect is the opposite because there are probably few erroneous sequences between the 1st candidate and the transmitted sequence. Moreover, in this case, a decrease of the number of erroneous sequences between the 1st candidate and the transmitted sequence lead to the increase of the probability of having the transmitted sequence among the K inspected candidates.

Finally, three different types of speed optimizations are proposed in this section: the elimination of some sequences having the same last 5-L symbols, the elimination of some initial survivors and the elimination of candidates with the same final accumulated distance. Additionally, their application on the different GPS messages is commented.

E.4.5.4.1. Sequences having the same last 5-L symbols elimination

This algorithm variation consists in eliminating the sequences which diverge from a candidate at some of its last 5-L states and which merge again with the candidate at the a later state. In other words, if a sequence diverging from i-th candidate at state S, where S belongs to the last 5-L states, merges again to the i-th candidate at a later state S’, where S’ belongs to the last 5-L states and S < S’ in time, the sequences can be eliminated from the search of the minimum alternative path of the i-th candidate.

This implementation of this algorithm variation is applied during the generation of each candidate. More specifically, once the i-th candidate has been generated, the receiver first searches for the sequences fulfilling the previous characteristics. And second, the receiver marks as created descendants the states where the candidate merges with the previous determined sequences. Note that this implementation can only be made during the generation of each candidate because it is the only moment where the states of the sequences merging with the candidate are available.

The justification of the implementation of this algorithm variation is given next. The last 5-L bits of a candidate only are used to obtain the convergence of the first M bits of the sequence; the bits which have to pass the outer channel code verification. In fact, the values of these last 5-L bits do not determine the bits of the transmitted code word. However, since these bits are part of the candidate, the proposed method can waste some time searching different candidates with the same first M bits but with different last 5L bits. More specifically, the proposed method can search minimum alternative paths which do not change the candidate previous M bits and thus the outcome of the outer channel code verification is not modified. This means that method is wasting time and resources.

Nevertheless, remember that the application of any speed algorithm variation implies that not all the original sequence space is inspected. Therefore, the proposed algorithm modification cannot inspect the transmitted sequence. An example is given next. Although an X₀ sequence merging at some of the last 5-L states of the candidate do not modify the first M bits, it cannot be guaranteed that a X₁ sequence merging with X₀ sequence does modify these bits. Therefore, X₁ sequence could pass the outer channel code verification if X₁ sequence was not eliminated due to this improvement. A graphical example is given below in Figure E-5:
Finally, we tested this speed improvement and we determined that it is highly improbable to lose the correct transmitted sequence due to this improvement.

E.4.5.4.2. Elimination of initial survivor

This algorithm variation consists in eliminating the initial survivors having the same first M states as another initial survivor already inspected as candidate. In other words, if an initial survivor diverges from another initial survivor, already inspected as a candidate, at state S belonging to the same of the last 5\(L\) states, the initial survivor can be eliminated.

This implementation of this algorithm variation is made during the generation as candidate of each initial survivor. More specifically, the receiver executes the complete Viterbi decoding without any restriction and selects the desired initial survivor as a candidate. Then, the receiver searches for the remaining initial survivors, still not being candidates, which have the same M states as the initial survivor being generate as candidate. Finally, these selected initial survivors are marked as generated candidates in order to avoid their future generation.

Moreover, note that this modification can be implemented each time that a candidate is generated. In this case, since the Viterbi decoding has been executed with some restrictions, the initial survivors do not have to be the same initial survivors obtained when the Viterbi decoding was executed without restrictions. Nevertheless, the principle is the same and thus the initial survivors obtained by a Viterbi decoding with restrictions can also be eliminated if they have the same first M states as the generated candidate.

The justification of the implementation of this algorithm variation is the same as the speed improvement commented in annex E.4.5.4.1. Since some initial survivors do not change the first M bits of a candidate which has already failed the outer channel code verification, the receiver does not have to waste time and resources inspecting more sequences which do not change these first M bits.

Nevertheless, this speed improvement also removes some segments of the sequence space from the search of the transmitted sequences and thus, the correct transmitted sequence can again be lost. The same example as the example presented in section E.4.5.4.1 can be applied here. The only difference is that the sequence diverging from the candidate does not merge again with the candidate. Figure E-16 presents a graphical example.
Finally, we tested this speed improvement and we determined that it is highly improbable to lose the correct transmitted sequence due to this improvement.

**E.4.5.4.3. Elimination of sequences having the same accumulated distances**

This algorithm variation consists in eliminating the sequences having the same accumulated distance as a candidate. In other words, if a sequence has exactly the same accumulated distance as a candidate, it means that the sequences and the candidate are in reality the same sequence: the candidate and the sequence having the same accumulated distance as the candidate are the same sequences but generated from different fathers. Therefore, the receiver eliminates the new sequence since this sequence has already been generated as candidate.

This implementation of this algorithm variation is made during the minimum alternative path search of the \(k^{th}\) candidate. More specifically, if the receiver saves the accumulated distance of the previous candidate, the receiver can compare this value with the accumulated distance of the minimum alternative path of the \(k^{th}\) candidate. Therefore, if the distances are equal, this means that both sequences are the same and thus, the receiver marks into the \(k^{th}\) candidate that a descendant has already been generated from the merging state of \(k^{th}\) candidate with the minimum alternative path.

Nevertheless, this application of this algorithm variation, as well as the two previous speed improvements, implies that some segment of the sequence space is not inspected during the sequence search. Therefore, the transmitted sequence can be lost. The reason is given next. Although the inspected sequences are equal to a candidate because they have the same accumulated distances, their generation conditions or, in other words, the restrictions of their ancestors are different. Consequently, some states differences of each sequence having the same accumulated distance can be different. This means that at each different state difference, each sequence having the same distance has a different sequence merging in this state. And each different sequence can be a different future candidate. Therefore, this technique does not explore the complete sequence space.

An example illustrating the previous explanation is shown in Figure E-17.
E. Decoding methods

Figure E-17 shows the following scenario. First, two candidates are created, one from the other, and the restriction which divides their sequence space segment is marked in green. Second, the algorithm searches for and creates the 3rd candidate either from the 1st candidate or from the 2nd candidate. We can see that the states of the 3rd candidate do not depend on the identity of the father of the 3rd candidate. Therefore, if the proposed modification is applied, the new restriction is marked over the 1st candidate as well as over the 2nd candidate regardless of the identity of the father of the 3rd candidate. Third, two different 4th candidates can be generate depending on the father of the 3rd candidate. More specifically, we can observe that the 4th candidate generated by the 3rd candidate when the 2nd candidate was the father of the 3rd candidate can never be generated by the 3rd candidate when the 1st candidate was the father of the 3rd candidate. And the opposite is also true. In fact, the restriction imposed by the 1st candidate does not allow that the part of the 4th candidate until point A is flat. Consequently, since the sequences which can be generated from the 3rd candidate when its father is the 1st candidate cannot be flat before the A point, the transmitted sequence cannot ever be recovered.

Finally, we observed that the cases when this algorithm variation implies the no inspection of the transmitted sequence are really infrequent. Moreover, this algorithm variation is really significant in speed terms because in some cases the receiver tends to spend a lot of time generating the same candidate. Therefore, due to the favorable relationship between the speed improvement and the number times that the transmitted sequence is not inspected this improvement becomes very attractive.

E.4.5.4.4. Application of the speed improvements on the GPS signals

The effects of the application of the speed up improvements on the GPS navigation messages having a different outer channel code have to be differentiated.

E.4.5.4.4.1. Application of the speed improvements on the GPS signals having the CRC-24Q as outer channel code

The CRC-24Q channel code has a very low probability of non-detection [ARINC, 2004]. This means that although the application of the speed up improvements can make the receiver not to inspect the transmitted sequence, the receiver hardly accepts any erroneous code word as
the transmitted code word due to the high detection probability of the CRC-24Q. In fact, none of the conducted simulation has ever accepted an erroneous word and the total number of tested words is about 4 million words. In other words, for the GPS message implementing the CRC-24Q channel code, the only source of error is that the maximum number of candidates, $K$, is reached without finding a sequence containing the bits of the transmitted code word.

To sum up, since the proposed speed modifications increase a lot the execution speed of the method, help the introduction of the transmitted sequence among the $K$ candidates and the number of times that a sequence space containing the transmitted sequence is discarded beforehand is very low, the GPS messages implementing the CRC-24Q outer channel code should always apply all the speed improvements.

**E.4.5.4.4.2. Application of the speed improvements on the GPS signals having the extended Hamming as outer channel code**

The main difference between the CRC-24Q case and the extended Hamming case is the detection capacity of the channel code. In this case, the detection capacity is much lower than the detection capacity of the CRC-24Q channel code. This means that the receiver will select as the transmitted sequence an erroneous sequence more often than in CRC-24Q case. In fact, for the CRC-24Q case, the receiver will usually inspect the $K$ candidates without finding a sequence which passes the outer channel code verification whereas for the extended Hamming case, it cannot be guaranteed.

Another difference is the size between the words when the CRC-24Q code is applied and the words when the extended Hamming code is applied. For the CRC-24C code, the code word size is equal to 300 bits whereas for the extended hamming code this number is reduced to 30. Therefore, the number of sequences is drastically reduced, and the probability of discarding beforehand the transmitted sequence when the speed improvements are applied is increased for the extended hamming channel code with respect to the CRC-24Q channel code.

Nevertheless, despite these two significant drawbacks, the improvement based on the elimination of the sequences having the same accumulated distances is still worthy and even necessary. More specifically, there are a lot more cases where the receiver reaches the maximum number of candidates, $K$, without finding the transmitted sequence than cases where the receiver accepts an erroneous sequence because the transmitted sequence is discarded beforehand.

However, the application of the other two speed improvements is not so evident since too many sequences are discarded beforehand with respect to the total number of sequences. Therefore, we only apply the speed improvement based on the elimination of the paths having the same last 5-L bits elimination improvement since it provides a better increase of the execution speed of the algorithm than the other improvement.

To sum up, for the extended hamming case, we apply the improvements based on the elimination of the sequence having the same last 5-L bits and on the elimination of the sequences having the same accumulated distances, but not the improvement based on the eliminate of the initial survivors.
E.5. Low Density Parity Check (LDPC) channel code

Two different LDPC channel codes are implemented on the subframes 2 and 3 of the GPS L1C signal [ARINC, 2006].

A LDPC code is a linear channel code block which can be systematic by construction. The GPS L1C LDPC subframe 2 and 3 channel codes are systematic and different for each subframe. The LDPC channel codes provide a very high performance: the BER versus $E_b/N_0$ relationship is very close to the theoretical limit calculated by Shannon, and known as the Shannon Limit [PROAKISd, 2001]. This high performance is obtained when the channel codes are applied on very long messages but this is not the case for subframe 2 and 3 sizes. However, the generation matrix, $G$, of each channel code subframe has been optimized in order to achieve a high decoding performance. In fact, the factors establishing the channel code performance, BER versus $E_b/N_0$, are the $G$ matrix of the LDPC code altogether with the chosen decoding process. Therefore, since subframe 2 and 3 sizes are different, the $G$ matrices are different, the LDPC codes are different and thus their performance is also different.

The coding process is equal to any other linear block channel code: it is an algebraic multiplication between the generation matrix and the information word expressed as a vector. However, for the GPS L1C navigation message, another method is defined in [ARINC, 2006].

In this work, the typical LDPC decoding method developed by D.J.C. MacKay and R.M. Neal is used. The main principle is commented but the reader is directed to [MACKAY and NEAL, 1995] [MACKAY, 1999] for a detailed explanation.

The main principle of the decoding method is to calculate and propagate probabilities. There are two types of probabilities, the bits probabilities and the checks probabilities. They are calculated using the parity matrix, $H$ (orthogonal matrix to the generation matrix). Remember that the checks of a parity matrix are the rows of this matrix, where the multiplication between a check and the received code word has to be equal to 0 in order to decide that the coded word does not contain any error. Therefore, a received code word has to verify all the checks before deciding that the word is error free.

The bits probabilities depend on the checks where the bits appear and the checks probabilities depend on the bits values which are used to calculate the check. Obviously, these two types of probabilities depend one to the other and thus an iterative algorithm of propagation has been established. Finally, the decoded information word is found when all the checks are verified (are equal to 0) or the maximum number of iterations is reached without deciding any word. In this last case, the last iterated word is delivered but indicating that it is not error free.

The complete algorithm can be found in [MACKAY and NEAL, 1995] [MACKAY, 1999].
Annex F. Probabilities of the binary prediction and influence of the bits forming the ephemeris to the described orbit

In this annex, all the probabilities analyses related to the binary prediction studied in Chapter 4 are presented in detail. Moreover, the influence of the bits forming the different Keplerian parameters on the orbit described by them is inspected.

F.1. Theoretical calculations of the required performance level of the binary prediction

In this section, the theoretical analyses conducted by this dissertation in order to determine the performance level of the binary prediction required to fulfill the conditions imposed by the Chapter 4 case of study are presented.

First the probability definitions are presented. Second, their mathematical expressions are calculated.

F.1.1. Probability definitions

In order to find the error probability of the first position error bit, this annex develops the equation already given in section 4.2.4.3.

\[ P(0e \text{ aR TP}) + P(1e \text{ aR TP}) \geq 99\% \]  

(\text{F-1})

Where

- \( P(0e \text{ aR TP}) \): Probability of having 0 user position errors after the RAIM application during TP time
- \( P(1e \text{ aR TP}) \): Probability of having 1 user position error after the RAIM application during TP time
- \( TP \): Time period defined by the case of study

Once the equation is reminded, this annex continues by dividing each individual probability into components, and continues to divide the resulting components into more elements until these elements can be calculated.

However, before beginning with the inspection of the terms, two important parameters have to be introduced. The first parameter is the number of satellites in sight and the second one is the number of independent positions that have to be calculated during the case of study period of time. Note that each independent user position is estimated with different satellite ephemeris data sets at different instants:

- Number of satellites: \( M \)
- Number of estimated positions: \( N = 12 \text{ (ephem/day)} \times \text{Number of days inside the defined time period} \)
Therefore, the previous expression (F-1) is equivalent to:

\[ P(0e \ aR \ Npr) + P(le \ aR \ Npr) \geq 99\% \]  

(F-2)

Where:

- \( P(Xe \ aR \ Npr) \): Probability of having \( X \) user position errors after the RAIM application when estimating \( N \) user positions

The first term of equation (F-2) is equal to:

\[ P(0e \ aR \ Npr) = P(0e \ bR \ Npr) + \sum_{k=1}^{N} P(ke_{-1}se \ bR \ Npr) \cdot P(Rd_{-ke}ke) \]  

(F-3)

Where

- \( P(0e \ bR \ Npr) \): Probability of having 0 user position errors before the RAIM application when estimating \( N \) user positions
- \( P(ke_{-1}se \ bR \ Npr) \): Probability of having \( k \) user position errors, each one caused by 1 individual satellite position error before applying the RAIM when estimating \( N \) user positions
- \( P(Rd_{-ke}ke) \): Probability of RAIM detecting \( k \) user position errors, where each user position error is caused by only 1 satellite position error. The total amount of existing user position errors is \( k \).

And each term \( P(ke_{-1}se \ bR \ Npr) \) can be separated into the sum of several other terms. Nevertheless, in order to simplify the study, before giving a general expression, each term is analyzed individually.

\[ P(le_{-1}se \ bR \ Npr) = \sum_{i=1}^{N} \sum_{n=1}^{M} P(Sat_{n_{-1}e} Sat_{n_{-0}e} Npr) \]  

(F-4)

Where:

- \( P(Sat_{n_{-1}e} Sat_{n_{-0}e} Npr) \): Probability of having a position error of the satellite \( n \) during the \( i^{th} \) user position estimation. The position of the satellite \( n \) is correct for the other user position estimations (\( \neq i \)). The positions of the remaining satellites (\( \neq n \)) are always correct. The total number of user positions estimated are \( N \).

\[ P(2e_{-1}se \ bR \ Npr) = \sum_{i=1}^{N} \sum_{j=1}^{N} \sum_{n=1}^{M} \sum_{m=1}^{M} P(Sat_{n,m_{-1}e} Sat_{n,m_{-0}e} Npr)_{i,j} \]  

(F-5)

- \( P(Sat_{n,m_{-1}e} Sat_{n,m_{-0}e} Npr)_{i,j} \): Probability of having a position error of the satellite \( n \) during the \( i^{th} \) user position estimation and of having a position error of the satellite \( m \) during the \( j^{th} \) user position estimation (\( m \) can be equal to \( n \)). The position of the satellite \( n \) is correct for the other user position estimation (\( \neq i \)) and the position of the satellite \( m \) is correct for the other user position estimation (\( \neq j \)) (except when \( m = n \)). The remaining satellites positions (\( \neq n,m \)) are always correct. The total number of user positions estimated are \( N \).

Therefore, the general expression can be extrapolated from the previous (F-4) and (F-5) mathematical equations.
F. Probabilities of the binary prediction and influence of the bits forming the ephemeris to the described orbit

Once the first term of equation (F-2) has been detailed, the second term can be also defined as a sum of terms.

\[
P(1e \text{ aR Npr}) = \sum_{k=1}^{M} P(ke \text{ _1se bR Npr}) \cdot P(Rd \text{ (k-1)e} | ke) + \\
+ \sum_{k=1}^{M} P(1e \text{ _kse bR Npr}) + P(>2e \text{ bR Npr})
\]  

(F-6)

Where:

- \(P(ke \text{ _1se bR Npr})\): Probability of having \(k\) user position errors, each one caused by only one satellite error position before applying the RAIM when estimating \(N\) user positions.

- \(P(Rd \text{ (k-1)e} | ke)\): Probability of RAIM detecting and correcting \(k-1\) user position errors, where each user position error is caused by only 1 satellite position error. The total amount of existing user position errors is \(k\) and the total number of user positions estimated are \(N\).

- \(P(1e \text{ _kse bR Npr})\): Probability of having 1 user position error caused by \(k\) satellite position errors before applying the RAIM when estimating \(N\) user positions.

- \(P(>2e \text{ Npr})\): Probability of having 2 or more user position errors, where 1 user position error among the total number of user position errors is caused by at least two satellite position errors and the remaining user position errors are caused by only one satellite position error before the applying the RAIM. This probability takes into account the probability of the RAIM detecting and correcting all the user position errors caused by only one satellite position error. The total number of user positions estimated are \(N\).

The last term \(P(>2e \text{ Npr})\) can be expressed as.

\[
P(>2e \text{ Npr}) = \sum_{n=1}^{M} P(2e \text{ _1se _nse bR Npr}) \cdot P(Rd \text{ _1e} | 2e) + \\
+ \sum_{n=1}^{M} P(3e \text{ _1se _1se _nse bR Npr}) \cdot P(Rd \text{ _2e} | 3e) + \ldots + \\
+ \sum_{n=1}^{M} P(Ne \text{ _1se...1se _nse bR Npr}) \cdot P(Rd \text{ _N} | Ne) 
\]  

(F-7)

Where:

- \(P(2e \text{ _1se _nse bR Npr})\): Probability of having 2 user position errors before applying the RAIM. One user position error is caused by only one satellite position error, and the other error is caused by \(n\) satellite position errors.

- \(P(3e \text{ _1se _1se _nse bR Npr})\): Probability of having 3 user position errors before applying the RAIM. Two user position errors are caused by only one satellite position error, and the remaining user errors are caused by \(n\) satellite position errors.

- \(P(Ne \text{ _1se...1se _nse bR Npr})\): Probability of having \(N\) user position errors before applying the RAIM. \(N-1\) user position errors are caused by only one satellite position error, and the remaining user position errors are caused by \(n\) satellite position errors.

Finally, note that the previous probabilities given in equations (F-6) and (F-7) can also be expressed as equations (F-4) and (F-5).
F. Probabilities of the binary prediction and influence of the bits forming the ephemeris to the described orbit

F.1.2. Probability calculations

Once the probabilities definitions involved in the case of study have been presented, the mathematical expression of each probability, or the addition of some of them, is presented.

Nevertheless, before giving the mathematical expressions of the probabilities given in section F.1.1, the RAIM detection probabilities are presented since they are more easily modeled. Taking into account the assumption saying that each satellite position prediction is independent from the prediction of the other satellite positions and the prediction of the same satellite position but in a different instant of time, the probabilities can be modeled as a binomial variable.

\[
P(R_{\text{d}_k}) = P_{dR}^k \tag{F-8}
\]

\[
P(R_{\text{d}_{k-1}}) = k \cdot P_{dR}^{(k-1)} \cdot (1-P_{dR}) \tag{F-9}
\]

Where:

- \( P_{dR} \): Probability of RAIM detecting the user position error.

Therefore, since the algorithm fault detection function is able to detect any failure equal or larger than 600m with a non-detection probability of \( 10^{-3} \) for a mono-frequency receiver in a urban environment [ESCHER, 2003], the detection probability \( P_{dR} \) is equal to:

\[
P_{dR} = 1-10^{-3} \tag{F-10}
\]

Now, in order to express erroneous user position probabilities, we use the independence among user position errors in time and in space, as said in section 4.2.4.2. Therefore, a user position estimation is an event independent of these other user position estimations. This means that the ensemble of the user position estimations during the specified time period can be modeled as a binomial variable.

\[
P(n_{\text{e}_{\text{Npr}}}) = \binom{N}{n} \cdot P_{\text{no\_error}}^{N-n} \cdot (1-P_{\text{no\_error}})^n \tag{F-11}
\]

Where:

- \( P(n_{\text{e}_{\text{Npr}}}) \): Probability of having \( n \) user position errors when estimating \( N \) user positions.
- \( P_{\text{no\_error}} \): Probability of correctly estimating a user position

A user position error can be produced by different causes, each of them defined in annex F.1.1. Therefore, we can modify the previous mathematical expression in order to adapt it to the previous defined probabilities.

\[
P(n_{\text{e}_{\text{X}_{\text{Npr}}}}) = \binom{N}{n} \cdot P_{\text{no\_error}}^{N-n} \cdot P_{\text{e}_{X}}^{n} \tag{F-12}
\]

Where:

- \( P(n_{\text{e}_{\text{X}_{\text{Npr}}}}) \): Probability of having \( n \) user position errors caused by \( X \), when estimating \( N \) user positions.
- \( P_{\text{e}_{X}} \): Probability of a user position estimation error caused by \( X \).
F. Probabilities of the binary prediction and influence of the bits forming the ephemeris to the described orbit

Therefore, for all the cases, we need to determine the possibility of correctly estimating a user position and to determine the probability of wrongly estimating a user position due to a given cause in order to express all the probabilities defined in annex F.1.1.

First, the probability of a correct user position estimation is equal to the probability that all the satellite position predictions are correct.

\[ P_{\text{no\_error}} = P_{\text{sat\_no\_error}}^M \]  \hspace{1cm} (F-13)

Where:
- \( P_{\text{sat\_no\_error}} \): Probability of correctly predicting the satellite position.
- \( M \): Average number of satellites.

Besides, the probability of well predicting a satellite position is equal to the probability of well predicting all its Keplerian parameters. And the probability of well predicting a Keplerian parameter depends on the number of erroneous bits accepted by each Keplerian parameter.

\[ P_{\text{sat\_no\_error}} = \left(1 - P_{\text{error}_1}\right)\left(1 - P_{\text{error}_2}\right)\left(1 - P_{\text{error}_3}\right)\ldots\left(1 - P_{\text{error}_n}\right) \]  \hspace{1cm} (F-14)

Customizing this expression for different bad Keplerian parameters predictions, we can obtain different results.

Finally, expression (F-12) is customized in order to describe all the probabilities defined in annex F.1.1. We begin with the simple probabilities and continue with the complicated ones using the simplifications and the results of the first probabilities.

First, the event of not having any error before applying the RAIM during all the time period is simply a binomial random variable where all cases are the same:

\[ P(0\text{e bR\_Npr}) = P_{\text{no\_error}}^N = P_{\text{sat\_no\_error}}^{M \cdot N} \]  \hspace{1cm} (F-15)

Second, we can regroup all the probabilities of having one user position error before the RAIM processing in one compact expression. Note that this expression is the sum of having one user position error caused by 1 satellite position error or more.

\[ P(1\text{e bR\_Npr}) = \sum_{n=2}^{M} P(1\text{e nse bR\_Npr}) + P(1\text{e 1se bR\_Npr}) \cdot P(Rd\_1\text{e}\_1\text{e}) + \sum_{n=1}^{M} P(1\text{e 1se bR\_Npr}) \cdot P(Rd\_0\text{e}\_1\text{e}) \]  \hspace{1cm} (F-16)

\[ P(1\text{e bR\_Npr}) = \sum_{n=1}^{M} P(1\text{e nse bR\_Npr}) \]  \hspace{1cm} (F-17)

Each individual term of equation (F-17) can be expressed as:

\[ P(1\text{e 1se bR\_Npr}) = \binom{N}{1} \cdot P_{\text{no\_error}}^{N-1} \cdot P_{\text{error}_1\_sat} \]  \hspace{1cm} (F-18)

\[ P_{\text{error}_n\_sat} = \binom{M}{n} \cdot P_{\text{sat\_no\_error}}^{M-n} \cdot \left(1 - P_{\text{sat\_no\_error}}\right)^n \]  \hspace{1cm} (F-19)

\[ P(1\text{e 1se bR\_Npr}) = N \cdot M \cdot P_{\text{sat\_no\_error}}^{N-M-1} \cdot \left(1 - P_{\text{sat\_no\_error}}\right) \]  \hspace{1cm} (F-20)
Therefore, the total probability of expression (F-17) is:

\[
P\left(1e \ bR \ Npr\right) = N \cdot \sum_{n=1}^{M} \binom{M}{n} \cdot P_{\text{sat_no_error}}^{N-M-n} \cdot \left(1 - P_{\text{sat_no_error}}\right)^n
\]  \quad \text{(F-21)}

Once the probability of obtaining one user position estimation error before the RAIM processing is given, this study can follow the same analysis to find the remaining probabilities. First, we give the probability of having either 0 or 1 user position error after the RAIM processing. We call this probability \(P_{2..Ne_{lsat}}\).

\[
P_{2..Ne_{lsat}} = \sum_{k=2}^{N} P\left(ke_{1se} bR \ Npr\right) \left[ P\left(Rd_{k \neq k}\right) + P\left(Rd_{(k-1)e_{k}}\right) \right]
\]  \quad \text{(F-22)}

We first simplify the probability expression by customizing the probability with \(k = N\).

\[
P_{Ne_{lsat}} = \binom{N}{N} \cdot \binom{M}{1}^N \cdot \left(1 - P_{\text{sat_no_error}}\right)^N \cdot \left(1 - P_{\text{sat_no_error}}\right)^N \cdot \left(1 - P_{\text{sat_no_error}}\right)^N \cdot \left(1 - P_{db}\right) \cdot \left(P_{dR} + N \cdot P_{dR}^{N-1} \cdot (1 - P_{db})\right)
\]  \quad \text{(F-23)}

The same development can be made but collecting the event where at least two user position errors occur, where one user position error is caused by more than one wrong satellite position, whereas the other user position errors are caused by only one wrong satellite position. This probability has been previously called \(P(>2e \ Npr)\). Nevertheless, in order to simplify the process, the probability of having 2 user position errors when 1 user position error is caused by \(n\) satellite errors is expressed first.

\[
P_{2e_{nsat}} = \binom{2}{1} \cdot \binom{N}{2} \cdot \binom{M}{1} \cdot \binom{M}{n} \cdot P_{\text{sat_no_error}}^{N-M-1-n} \cdot \left(1 - P_{\text{sat_no_error}}\right)^{n+1} \cdot P_{db}
\]  \quad \text{(F-25)}

The main difference between this expression and the previous probability equations is the first combinatorial number. This number is necessary because we have to distinguish the user position errors caused by at least two wrong satellite positions from the user position errors caused by only one wrong satellite position. Indeed, it is not the same case to have the first user position erroneous due to two satellite position errors and the second user position erroneous due to one satellite position error compared to the case where we have the first user position erroneous due to one satellite position error and the second user position erroneous due to two satellite position errors.

Therefore, if we regroup all the previous probability expressions for the cases where 2 user position errors occur:

\[
P\left(2e \ Npr\right) = 2 \cdot \binom{N}{2} \cdot P_{dR} \cdot \binom{M}{n} \sum_{n=2}^{M} P_{\text{sat_no_error}}^{N-M-1-n} \cdot \left(1 - P_{\text{sat_no_error}}\right)^{n+1}
\]  \quad \text{(F-26)}

The expression for 3 user position errors is thus.

\[
P\left(3e \ Npr\right) = 3 \cdot \binom{N}{3} \cdot P_{dR}^2 \cdot \binom{M}{n} \sum_{n=2}^{M} P_{\text{sat_no_error}}^{N-M-2-n} \cdot \left(1 - P_{\text{sat_no_error}}\right)^{n+2}
\]  \quad \text{(F-27)}
Finally, the expression of the probabilities of all the cases with more than one user position error, where one user position error is caused by at least two satellite position errors whereas the other user position errors are caused by only one satellite position error is:

\[
P(\geq 2e \mid Npr) = \sum_{i=2}^{N} \binom{N}{i} \cdot P_{2e}^{i-1} \cdot M^{i-1} \cdot f(M,i)
\]  

\[
f(M,i) = \sum_{n=2}^{M} \binom{M}{n} \cdot P_{\text{sat}_n\text{no_error}}^{N-M-n+i+1} \cdot (1 - P_{\text{sat}_n\text{error}})^{n+i+1}
\]

Finally, with this last equation, all the probabilities defined in annex F.1.1 have been given. Therefore, the sum of expressions (F-15), (F-21), (F-24) and (F-28) are equivalent to the left part of equation (F-1).

\[
P(\theta \leq b \mid bR \mid Npr) + P(1e \mid b \mid Npr) + P_{2e_{\leq \text{Ne}_{\text{sat}}} Npr} + P(\geq 2e \mid Npr) \geq 99\%
\]  

\[
(F-30)
\]

**F.2. Relationship between the binary domain and the decimal domain**

In this section, the theoretical calculations conducted to establish the relationship between a number expressed in binary format and the same number expressed in decimal format are presented. More specifically, these calculations search for the first and second percentages associated to a decimal number in order to obtain the prediction of a given number of bits. See section 4.2.5 for a definition of the first and second percentage.

First of all, it has to be reminded that if the bit in position n is not well predicted by a physical prediction method, see section 4.2.2 for a definition of a physical prediction method, the method cannot guarantee the correct prediction of any bit value less significant than the bit in position n.

The theoretical analysis conducted to find the relationship between the decimal and the binary precision can be divided into four main parts. The first part searches for the worst prediction able to correctly predict a given bit. The second part searches for the best prediction which does not correctly predict a given bit. The third part calculates from each prediction the decimal difference expressed in binary terms between the predictions and the real values. These differences are called distances and there are several values to predict having the worst and best prediction distances. Note that the distance of the worst predictions is the maximal acceptable decimal distance and that the distance of the best predictions is the minimal acceptable decimal distance. Finally, the fourth part calculates from the worst prediction distance the first percentage and from the best prediction distance the second percentage described in section 4.2.5.

In order to accomplish the first and second parts, a preliminary step has to be made. This preliminary step consists in sorting all the range of values of the field to predict into two groups, one group gathering the values having the n\textsuperscript{th} bit equal to 0, and the other group gathering the values having the n\textsuperscript{th} bit equal to 1. Each set is called bit-value-set. From these sets, the search of the worst and the best prediction is simplified.

In order to find the worst prediction able to correctly predict a given bit, we search for the two values belonging to the same bit-value-set which are separated by the largest distance. Therefore, since there is no other pair of values farther apart, each value represents the worst prediction when the other value is the value to be predicted.
F. Probabilities of the binary prediction and influence of the bits forming the ephemeris to the described orbit

In order to find the best prediction unable to correctly predict a given bit, we search for the two values belonging to different bit-value-sets which are separated by the smallest distance. Therefore, since there is no other closest pair of values, each value represents the best prediction when the other value is the value to be predicted.

An example of the worst and best predictions of the first bit of a binary value represented by 5 bits is presented in Table F-1. In this case, the bit-value-set of the first bit equal to 0 consists of the binary numbers between 00000 and 01111, and the bit-value-set of the first bit equal to 1 consists of the binary numbers between 10000 and 11111.

<table>
<thead>
<tr>
<th>Value of the First bit</th>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Real value to predict associated to the best prediction</td>
<td>01111</td>
<td>10000</td>
</tr>
<tr>
<td>Best prediction erroneously predicting the associated value</td>
<td>10000</td>
<td>01111</td>
</tr>
<tr>
<td>Real value to predict associated to the worst prediction</td>
<td>00000/01111</td>
<td>10000/11111</td>
</tr>
<tr>
<td>Worst prediction correctly predicting the associated value</td>
<td>01111/00000</td>
<td>11111/10000</td>
</tr>
</tbody>
</table>

Table F-1: First bit best and worst predictions for a 5 bit number

Note that since the worst prediction is searched between pairs of numbers belonging to the same bit-value-set, the worst prediction and the real value to be predicted are interchangeable.

The table presenting the binary values of the worst and best prediction distances is given below.

<table>
<thead>
<tr>
<th>Value of the First bit value</th>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Best prediction distance</td>
<td>00001</td>
<td>00001</td>
</tr>
<tr>
<td>Worst prediction distance</td>
<td>01111</td>
<td>01111</td>
</tr>
</tbody>
</table>

Table F-2: First bit best and worst prediction distances for a 5 bits number

One important remark about Table F-2 is that the best and worst distances are equal for both bit-value-sets. Therefore, the study can be reduced to the analysis of only one of the bit-value-sets.

At this point, we should use the worst and best prediction distances expressed in decimal format in order to calculate the percentages with respect to the maximal range value, i.e. the first and second percentages. However, the conversion process of these binary distances into the decimal format is complex. The accuracy in decimal format for a value converted from a binary format is determined by this LSB of the value. This means that the worst and best predictions distances have an accuracy which depends on this bit, and this limit of accuracy is harmful for our calculation since it limits the precision of our percentages to the LSB accuracy. Therefore, in order to remove this limit of accuracy and in order to have very accurate decimal distance values, we have to artificially extend the accuracy of the binary format of the distances. This extension is easily achieved if we consider a decimal number as a binary number with an infinite number of bits. This means that if we insert a sufficient number of extra bits of lower weight than the LSB of the original binary format of the binary value to the binary value, we are approaching the binary value to the ideal binary
F. Probabilities of the binary prediction and influence of the bits forming the ephemeris to the described orbit

representation of a decimal value. Finally, the values of these extra bits are smartly determined in order to adapt the new precision to our requirements. An example of this process is given next when 10 low weight bits are inserted.

The insertion of 10 low weight bits to the initial number of bits of the binary field in order to modify the worst and best prediction distances and the maximal range value is explained next. The values of the bits inserted to the maximal range value are equal to 1. The values of the bits inserted to the worst distance are 1 for the bit with more weight and 0 for the others. The reason is that the worst decimal maximal distance with the new accuracy is the sum of two terms. The first term is the original binary worst prediction distance expressed in decimal format. The second term is half the decimal value of the LSB of the original binary format of the distance, where the original binary format is the binary format before inserting the 10 extra bits. And this half decimal value of the original binary format LSB is represented by the 10 additional bits where the first bit is equal to 1 and the last 9 bits are equal to 0. Finally, the same analysis can be made for the best prediction distance with new accuracy. In this case, we have to subtract half the decimal value of the LSB of the original binary format from the distance. This means that first we have to change the last bit of the original binary format of the best prediction distance to 0, and second we have to add the 10 extra bits with the same values as the values of the bits of the worst prediction distance.

Finally, the formula of the prediction relative distance is given below.

\[
\text{Distance precision (\%) = 100 \times \frac{\text{Dist}}{\text{Max Range Value}}} \tag{F-31}
\]

Where:

- Dist: Prediction distance expressed in decimal format
- Max Range Value: Maximal range value expressed in decimal format

Therefore, applying equation (F-31) and expanding the value precision with 10 bits as explained in the previous paragraph, the results obtained for the prediction of the first bit are presented below.

<table>
<thead>
<tr>
<th>Prediction accuracy</th>
<th>Maximal Precision Distance</th>
<th>Minimal Precision Distance</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>48.4375 %</td>
<td>1.5625 %</td>
</tr>
</tbody>
</table>

**Table F-3: First bit prediction accuracy for a 5 bit number**

<table>
<thead>
<tr>
<th>Prediction accuracy</th>
<th>Maximal Precision Distance</th>
<th>Minimal Precision Distance</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>49.8 %</td>
<td>0.2 %</td>
</tr>
</tbody>
</table>

**Table F-4: First bit prediction accuracy for an 8 bit number**

Table F-3 shows that if we want the physical prediction method to be able to predict the first bit of a 5 bit number, the prediction has to be at a distance from the real value lower than \(\approx 50\%\) of the maximal range value. Nevertheless, this table shows that despite obtaining an almost perfect prediction, such as a distance between the prediction and the real value of 2\% of the maximal range value, the prediction of the first bit cannot be guaranteed. The same
conclusions can be made from Table F-4. Moreover, from both tables, we can conclude that the number of bits has little influence on the final results.

Another example is given before presenting a general equation of the evolutions of the maximal and minimal precision distances. The example searches for the accuracy necessary to predict the second bit. In this case, there are four different bit-value-sets because we assume that if the second bit is to be predicted, the first bit has to be correctly predicted. Therefore, it is impossible for a physical prediction method to predict a decimal value expressed as 01XXX when the first bit is known to be 1. Consequently, the four bit-value-sets for a 5 bit number are 00XX, 01XX, 10XX, 11XX. And from these sets, the worst and best predictions and their associated distances are found. The distances are shown below in Table F-5.

<table>
<thead>
<tr>
<th>First bit value</th>
<th>0</th>
<th>1</th>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Second bit value</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Best prediction distance</td>
<td>00001</td>
<td>00001</td>
<td>00001</td>
<td>00001</td>
</tr>
<tr>
<td>Worst prediction distance</td>
<td>00111</td>
<td>00111</td>
<td>00111</td>
<td>00111</td>
</tr>
</tbody>
</table>

Table F-5: Second bit best and worst prediction distances for a 5 bit number

And the distance precision for a 5 bit number, which is valid for a value expressed by any number of bits, is shown below on Table F-6.

<table>
<thead>
<tr>
<th>Prediction accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximal Precision Distance</td>
</tr>
<tr>
<td>23.4375 %</td>
</tr>
</tbody>
</table>

Table F-6: Second bit prediction accuracy for a 5 bit number

Therefore, the minimal precision distance (MnPD) necessary to guarantee a bit prediction remains constant for all the bits of the field to predict. Nevertheless, the maximal precision distance (MxPD) necessary to try to predict a bit depends on the bit position inside the field. The formula of the MxPD is given below.

\[
MxPD \approx \frac{100}{2^n} - \frac{2^n \cdot MnPD}{100} \quad (\%)
\]

Where:
- \( n \) = position of the bit to be predicted

**F.3. Probabilities a priori of the ephemeris data**

In 0, we have presented a modification of the traditional decoding Viterbi algorithm. This modification employed the a priori probabilities of the bits belonging to the ephemeris data. In this section, the method used to calculate the a priori probabilities is presented. These a priori probabilities have been calculated from the history of the ephemeris data which encloses the ephemeris data set broadcasted between 2004 and 2007. The data is recovered from the webpage http://igscb.jpl.nasa.gov.
F. Probabilities of the binary prediction and influence of the bits forming the ephemeris to the described orbit

We have obtained the probability of the \( n^{th} \) bit of the \( i^{th} \) Keplerian parameter by dividing the number of \( i^{th} \) Keplerian parameters having the \( n^{th} \) bit equal to 1 by the total number of \( i^{th} \) Keplerian parameters. Moreover, during the bit probabilities calculation, we remarked that some Keplerian parameters such as the \( M_0 \), \( C_{uc} \) or \( C_{rc} \) could be divided into two cases. In fact, when these Keplerian parameter bits probabilities were directly calculated, the results were quite disappointing with probabilities about 50%. However, when we separated first the negative and positive Keplerian parameter values, the probabilities, or at least the probabilities of the more significant bits were more satisfying. These probabilities had values about 80%. Therefore, we have separated the Keplerian parameters encoded 2’s complement into positive value probabilities and negative value probabilities.

Finally, each satellite has its own singular orbit, as has been shown in section 4.3.2 which means that each satellite has to have its own ephemeris bit probabilities. These probabilities can be recalculated along the time since more samples or broadcasted ephemeris will be available. Therefore, the receivers can store the a priori bit probabilities and receive their actualizations by external platforms such as mobile telephone stations.

One example table summarizing the a priori bit probabilities is shown at the end of this annex.

F.4. Influence of an erroneous bit inside a Keplerian parameter on the satellite position

The combination of two of the main assumptions of section 4.2.4.2 specifies that the binary prediction error of only bit of a Keplerian parameter bit results into an erroneous satellite position prediction. Moreover, a third assumption specified that an erroneous satellite position prediction leads to a user position prediction error. Therefore, in order to validate these assumptions, the influence of an erroneous bit inside a Keplerian parameter on the satellite position is inspected.

The method used for analyzing the influence of the Keplerian parameters bits on the satellite position consists in inspecting the satellite position error introduced by the variation of the Keplerian parameters bits. The method first searches for the error introduced by the variation of the bits of each individual Keplerian parameter and second, it searches for the error introduced by the variation of the bits of two simultaneous Keplerian parameters.

Moreover, since the weight of the bit certainly determines the influence of the bit on the satellite orbit position, the method searches the bit of lowest weight which changes the satellite position farther than a determined threshold distance. Therefore, this method begins the inspection by the LSB of the Keplerian parameter and finishes, if necessary, by the MSB. Moreover, since we use physical methods to predict the satellite position, the wrong prediction of a bit of a Keplerian parameter implies the wrong prediction of this bit and all the bits having less weight inside the Keplerian parameter. Therefore, the method considers a wrong bit prediction as a wrong prediction of this bit and its previous LSB.

More specifically, the method searches the satellite position error in the worst scenario, which is the situation where a Keplerian parameter bit prediction causes the largest change on the Keplerian parameter value. The worst scenario for a positive Keplerian parameter value when its \( n^{th} \) bit is to be tested is simulated by creating a new ephemeris data set where the Keplerian parameter \( n^{th} \) bit value and the bits having a lower weight are set to 0. This new ephemeris set is called the reference ephemeris data set. From the orbit defined by the reference ephemeris set, we can calculate the maximal error distance introduced by the wrong prediction of the \( n^{th} \) bit of this Keplerian parameter. This calculation is done by measuring the distance between
F. Probabilities of the binary prediction and influence of the bits forming the ephemeris to the described orbit

the orbit defined by the reference ephemeris data set and the orbits defined by the different ephemeris data sets which are equal to the reference ephemeris data set but with different values for the $n$th bit and its bits of lower weight of the Keplerian parameter. A similar process is conducted for a negative Keplerian parameter value.

Table F-7 shows the bit of more weight of a Keplerian parameter which causes a satellite position error larger than a determined threshold. This table has been calculated using 400 different ephemeris sets and varying the value of only one Keplerian parameter.

<table>
<thead>
<tr>
<th>Dist. (m)</th>
<th>$\sqrt{A}$</th>
<th>e</th>
<th>$\omega$</th>
<th>IDOT</th>
<th>$\Omega'$</th>
<th>$\Delta n$</th>
<th>M0</th>
<th>cuc</th>
<th>cus</th>
<th>crc</th>
<th>crs</th>
<th>cic</th>
<th>cis</th>
<th>$\Omega_0$</th>
<th>i</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>31</td>
<td>28</td>
<td>31</td>
<td>13</td>
<td>23</td>
<td>15</td>
<td>31</td>
<td>15</td>
<td>15</td>
<td>14</td>
<td>14</td>
<td>15</td>
<td>15</td>
<td>31</td>
<td>31</td>
</tr>
<tr>
<td>1</td>
<td>28</td>
<td>25</td>
<td>28</td>
<td>11</td>
<td>21</td>
<td>13</td>
<td>28</td>
<td>12</td>
<td>12</td>
<td>11</td>
<td>11</td>
<td>12</td>
<td>12</td>
<td>28</td>
<td>28</td>
</tr>
<tr>
<td>Num. Bits</td>
<td>32</td>
<td>32</td>
<td>32</td>
<td>14</td>
<td>24</td>
<td>16</td>
<td>32</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>32</td>
<td>32</td>
</tr>
</tbody>
</table>

Table F-7: First keplerian parameter bit position allowed to be wrong predicted when one and only one keplerian parameter is incorrectly predicted

From Table F-7, it can be observed that the bits of some Keplerian parameters have a larger influence on the satellite position than the bits of other Keplerian parameters. For example, the orbit semi-major axis accepts only a variation of its last 4 bits before modifying the satellite position by more than 1m, whereas the eccentricity allows a variation of its last 7 bits.

Table F-8 shows the bit of more weight which causes a satellite position error larger than a determined threshold when the bits of two different Keplerian parameters are modified at the same time. This table has been calculated using 400 different ephemeris sets.

<table>
<thead>
<tr>
<th>Distance (m)</th>
<th>$\sqrt{A}$</th>
<th>E</th>
<th>$\omega$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>32</td>
<td>---</td>
<td>32</td>
</tr>
<tr>
<td>1</td>
<td>32</td>
<td>32</td>
<td>---</td>
</tr>
<tr>
<td>1</td>
<td>---</td>
<td>32</td>
<td>32</td>
</tr>
</tbody>
</table>

Table F-8: First Keplerian parameter bit position allowed to be wrong predicted when two Keplerian parameters are incorrectly predicted

From Table F-8, it can be observed that the variation of the LSB of two different Keplerian parameters at the same time causes a satellite position error larger than 1 meter. Therefore, two or more Keplerian parameters cannot be incorrectly predicted at the same time if the GPS receiver seeks to implement a prediction method yielding a satellite position error lower than 1 meter.

One possible combination of wrong predicted bits of all the Keplerian parameters, except the $t_{sec}$, which cause a satellite position error larger than 600 meters is presented below.
Finally, we can conclude that the specifications of the case of study are only fulfilled when one or more bits are erroneously predicted for one Keplerian parameter. The erroneous prediction of bits belonging to different Keplerian parameters leads to a satellite position error larger than 1 meter and thus it is not accepted by our case of study. Moreover, it has been shown that in order to use the RAIM function, we cannot predict the LSBs of the Keplerian parameters since the satellite position error will be smaller than 600 meters and thus it will not be detected.
### Probabilities of having bits equal to 1

<table>
<thead>
<tr>
<th>SAT 01</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14</th>
<th>15</th>
<th>16</th>
<th>17</th>
</tr>
</thead>
<tbody>
<tr>
<td>sqrtA</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>E</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>O</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0.613</td>
<td>0.487</td>
<td>0.579</td>
<td>0.461</td>
<td>0.412</td>
<td>0.614</td>
<td>0.522</td>
<td>0.476</td>
<td>0.49</td>
<td>0.501</td>
<td>0.509</td>
<td>0.519</td>
<td>0.494</td>
</tr>
<tr>
<td>Id_first</td>
<td>0.654</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Id_pos</td>
<td>---</td>
<td>0.0</td>
<td>0.0</td>
<td>0.167</td>
<td>0.376</td>
<td>0.446</td>
<td>0.461</td>
<td>0.486</td>
<td>0.485</td>
<td>0.492</td>
<td>0.5</td>
<td>0.506</td>
<td>0.497</td>
<td>0.501</td>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>Id_neg</td>
<td>---</td>
<td>1</td>
<td>1</td>
<td>0.984</td>
<td>0.736</td>
<td>0.594</td>
<td>0.55</td>
<td>0.526</td>
<td>0.499</td>
<td>0.515</td>
<td>0.49</td>
<td>0.496</td>
<td>0.488</td>
<td>0.496</td>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>OI</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0.905</td>
<td>0.775</td>
<td>0.457</td>
<td>0.502</td>
<td>0.498</td>
<td>0.501</td>
</tr>
<tr>
<td>Ms</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0.004</td>
<td>0.855</td>
<td>0.383</td>
<td>0.493</td>
<td>0.504</td>
<td>0.499</td>
<td>0.501</td>
<td>0.508</td>
<td>0.499</td>
<td>0.497</td>
<td>0.502</td>
<td>0.502</td>
<td>0.504</td>
<td>---</td>
</tr>
<tr>
<td>M_pos</td>
<td>---</td>
<td>0.499</td>
<td>0.501</td>
<td>0.499</td>
<td>0.498</td>
<td>0.501</td>
<td>0.497</td>
<td>0.498</td>
<td>0.503</td>
<td>0.496</td>
<td>0.499</td>
<td>0.499</td>
<td>0.503</td>
<td>0.508</td>
<td>0.506</td>
<td>0.499</td>
<td>0.406</td>
</tr>
<tr>
<td>M_neg</td>
<td>---</td>
<td>0.497</td>
<td>0.499</td>
<td>0.499</td>
<td>0.499</td>
<td>0.499</td>
<td>0.501</td>
<td>0.5</td>
<td>0.503</td>
<td>0.495</td>
<td>0.502</td>
<td>0.504</td>
<td>0.505</td>
<td>0.491</td>
<td>0.499</td>
<td>0.5</td>
<td>0.498</td>
</tr>
<tr>
<td>Cuc_first</td>
<td>0.4438</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cuc_pos</td>
<td>---</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.28</td>
<td>0.407</td>
<td>0.461</td>
<td>0.472</td>
<td>0.486</td>
<td>0.488</td>
<td>0.492</td>
<td>0.491</td>
<td>0.501</td>
<td>0.51</td>
<td>0.499</td>
<td>0.493</td>
<td>---</td>
</tr>
<tr>
<td>Cuc_neg</td>
<td>---</td>
<td>1</td>
<td>1</td>
<td>0.971</td>
<td>0.595</td>
<td>0.539</td>
<td>0.53</td>
<td>0.518</td>
<td>0.518</td>
<td>0.502</td>
<td>0.507</td>
<td>0.496</td>
<td>0.512</td>
<td>0.503</td>
<td>0.503</td>
<td>0.497</td>
<td>---</td>
</tr>
<tr>
<td>CucFirst</td>
<td>0.9721</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cucpos</td>
<td>---</td>
<td>0</td>
<td>0</td>
<td>0.417</td>
<td>0.42</td>
<td>0.461</td>
<td>0.493</td>
<td>0.496</td>
<td>0.501</td>
<td>0.5</td>
<td>0.503</td>
<td>0.5</td>
<td>0.5</td>
<td>0.493</td>
<td>0.501</td>
<td>0.493</td>
<td>---</td>
</tr>
<tr>
<td>Cucneg</td>
<td>---</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0.9868</td>
<td>0.8785</td>
<td>0.6525</td>
<td>0.6073</td>
<td>0.5292</td>
<td>0.5678</td>
<td>0.5207</td>
<td>0.4981</td>
<td>0.5</td>
<td>0.4802</td>
<td>0.4463</td>
<td>---</td>
</tr>
<tr>
<td>Crc</td>
<td>0</td>
<td>0</td>
<td>0.5574</td>
<td>0.483</td>
<td>0.5284</td>
<td>0.503</td>
<td>0.4993</td>
<td>0.4965</td>
<td>0.4951</td>
<td>0.4965</td>
<td>0.498</td>
<td>0.4998</td>
<td>0.4996</td>
<td>0.4936</td>
<td>0.4981</td>
<td>0.5044</td>
<td>---</td>
</tr>
<tr>
<td>Crc_pos</td>
<td>---</td>
<td>0</td>
<td>0</td>
<td>0.0119</td>
<td>0.3441</td>
<td>0.4383</td>
<td>0.4726</td>
<td>0.4719</td>
<td>0.4909</td>
<td>0.4925</td>
<td>0.501</td>
<td>0.4981</td>
<td>0.5008</td>
<td>0.5047</td>
<td>0.5005</td>
<td>0.4999</td>
<td>---</td>
</tr>
<tr>
<td>Crc_neg</td>
<td>---</td>
<td>1</td>
<td>1</td>
<td>0.9094</td>
<td>0.5769</td>
<td>0.551</td>
<td>0.528</td>
<td>0.5102</td>
<td>0.516</td>
<td>0.5037</td>
<td>0.4955</td>
<td>0.5014</td>
<td>0.4938</td>
<td>0.5041</td>
<td>0.5007</td>
<td>0.4969</td>
<td>---</td>
</tr>
<tr>
<td>CrcFirst</td>
<td>0.4869</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Crcpos</td>
<td>---</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.2113</td>
<td>0.4593</td>
<td>0.4846</td>
<td>0.4883</td>
<td>0.498</td>
<td>0.488</td>
</tr>
<tr>
<td>Crcneg</td>
<td>---</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0.8487</td>
<td>0.4845</td>
<td>0.5306</td>
<td>0.5127</td>
<td>0.5124</td>
<td>0.5072</td>
</tr>
<tr>
<td>Cis_first</td>
<td>0.5268</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cis_pos</td>
<td>---</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.001</td>
<td>0.1924</td>
<td>0.34</td>
<td>0.4313</td>
<td>0.4624</td>
<td>0.4878</td>
</tr>
<tr>
<td>Cis_neg</td>
<td>---</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1.0006</td>
<td>0.78</td>
<td>0.6322</td>
<td>0.5533</td>
<td>0.5301</td>
<td>0.5105</td>
</tr>
<tr>
<td>O_first</td>
<td>0.493</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>O_pos</td>
<td>0.4921</td>
<td>0.5066</td>
<td>0.5066</td>
<td>0.4908</td>
<td>0.4916</td>
<td>0.4983</td>
<td>0.5039</td>
<td>0.5805</td>
<td>0.498</td>
<td>0.5007</td>
<td>0.5004</td>
<td>0.5002</td>
<td>0.5007</td>
<td>0.5018</td>
<td>0.5047</td>
<td>0.5039</td>
<td>---</td>
</tr>
<tr>
<td>O_neg</td>
<td>0.4937</td>
<td>0.4943</td>
<td>0.5086</td>
<td>0.5226</td>
<td>0.5086</td>
<td>0.5024</td>
<td>0.492</td>
<td>0.4127</td>
<td>0.4884</td>
<td>0.5052</td>
<td>0.5018</td>
<td>0.5007</td>
<td>0.5004</td>
<td>0.5007</td>
<td>0.5013</td>
<td>0.5005</td>
<td>---</td>
</tr>
<tr>
<td>i</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.5328</td>
<td>0.3511</td>
<td>0.5336</td>
<td>0.6742</td>
<td>0.5133</td>
<td>0.5154</td>
</tr>
</tbody>
</table>

---

F. Probabilities of the binary prediction and influence of the bits forming the ephemeris to the described orbit.
Annex G. Temporal methods of the binary prediction of the ephemeris data

In this annex, the temporal methods used to predict the bits of the ephemeris data are presented in detail.

G.1. Method allowing the use of the DIFF difference results

In this section, a method which is able to obtain the DIFF difference as result of the simplification process instead of the XOR difference is presented. Remember that the simplification process using the almanac data or the long term prediction ephemeris data has as result of the prediction the XOR difference, where the XOR difference determines the bit of most weight which cannot be perfectly predicted. Therefore, since the XOR difference is worse than the DIFF difference as verified in section 4.3.1.1.3 and 4.3.1.1.4, we are interested in applying the simplification process and in obtaining the DIFF difference as its result.

The proposed method consists in first making 3 predictions of the future ephemeris data set instead of only one, and second in choosing the correct prediction through the application of a determined criterion.

The explanation of the proposed is divided in 3 parts. First, the problems with the DIFF difference are identified and analyzed. Second the 3 predictions method used to solve the DIFF difference problem is presented. Third and last, the criterion used to select the correct prediction is explained.

G.1.1. DIFF difference problem

The DIFF difference is calculated by first subtracting the decimal reference value to the decimal broadcasted ephemeris data set. Second, we transform the absolute value of the resulting decimal difference into binary format. Therefore, the DIFF difference does not represent the bits of the reference value having the same value as the broadcasted Keplerian parameters bits. In fact, the DIFF difference represents the value missing from the reference value to reach the value of the broadcasted Keplerian parameter converted into binary format.

From now on, we call DIFF difference value the binary value to be added or subtracted from the reference value in order to obtain the value of the broadcasted Keplerian parameter. Therefore, since the DIFF difference represents a binary format of the value to be added or subtracted, we cannot guarantee the prediction of any bit from the DIFF difference value. The reason is that even the binary addition of the LSB bit to a binary number can modify all the bits of the number as shown in section 4.3.1.1.2. However, if we are able to determine in advance the modification of the predicted bits by the addition or the subtraction of the DIFF difference value, we could make a prediction which will obtain as a result of the simplification process a XOR difference equal to the DIFF difference.

To sum up, in this section, we analyze the structure of the DIFF difference value in order to determine the effects of the binary addition and subtraction. Therefore, knowing these effects, we obtain as result of the simplification process the DIFF difference instead of the XOR difference.
The DIFF difference value can be divided into two parts. The first part of this difference is formed by the DIFF difference constant bits, or, in other words, from the MSB bit until the last bit with probability of error equal to 0. Remember that the probability of error of a bit, as defined in section 4.3.1.1.3, determines the percentage of times that the bit of the reference value has been the bit of most weight not equal to the bit of the broadcasted Keplerian parameter which is at the same position. Therefore, in the DIFF difference case, if the probability of error of a bit is equal to 0, this means that this bit has a constant value equal to 0 in the DIFF difference value.

The second part of this difference is formed by the non-constant or unknown bits, or, in other words, from the first bit with a probability of error not equal to 0 to the LSB bit. Therefore, since the constant bits of this difference are equal to 0, the decimal DIFF difference value is completely determined by these non-constant and unknown bits and thus it is not fixed.

The DIFF difference value can be defined as:

\[
\text{DIFF difference value: } \underbrace{Df_1}_0 \ldots \underbrace{Df_2}_{0x} \ldots x
\]

Where:
- \(Df_1\): First part of the DIFF difference – Constant bits
- \(Df_2\): Second part of the DIFF difference – Not constant or unknown bits

From the structure shown in equation (G-1), we can draw some conclusions. First, the first part of the DIFF difference does not have any influence on the addition between DIFF difference value and the reference value. Second, the influence of the unknown bits of the second part of the DIFF difference on this addition is unpredictable. Therefore, since this influence is unpredictable, all the bits of the addition between the DIFF difference value and the reference value are also unpredictable. And this means that the DIFF difference value cannot be used as indication of which bits can be predicted by the simplification process. An example is given below.

<table>
<thead>
<tr>
<th>Reference Value (RV)</th>
<th>DIFF Difference</th>
<th>Df1</th>
<th>Df2</th>
<th>RV + Df1 + Df2</th>
</tr>
</thead>
<tbody>
<tr>
<td>101101</td>
<td>0000xx</td>
<td>\textbf{000000}</td>
<td>0000xx</td>
<td>xxxxx</td>
</tr>
</tbody>
</table>

Table G-1: DIFF difference numerical example

Where:
- \(x\): Represent the value of an unknown bit

In order to be able to exploit the DIFF difference value as the vector indicating the bits which can be predicted by the simplification process, we have to find the influence of \(Df_2\) on the MSB bits of the addition of the reference value with the DIFF difference value. Remember that the DIFF difference value allows the prediction of more bits than the XOR difference value.

In order to do so, we calculate all the possible \(Df_2\) contributions on the binary addition between the DIFF difference value and the reference value so that we can find a predictable pattern of the contribution on the MSB bits of the addition. The contributions of \(Df_2\) are divided into their contribution on the bits of the binary addition which are at the same position than the \(Df_2\) bits and their contribution on the bits of the binary addition which are at the same position than the \(Df_1\) bits. Table G-2 shows the contribution of different positive values
of the unknown bits. The vertical line separates the bits belonging to Df1 from the bits belonging to Df2.

| Reference value | 0..0|111 | 0..0|011 | 0..0|111 | 0..0|001 |
|-----------------|-----|-----|-----|-----|-----|-----|-----|
| Positive DIFF difference (Df1 | Df2) | 0..0|011 | 0..0|100 | 0..0|111 | 0..0|010 |
| Df2 (Denoted as (2)) | | 011 | 100 | 111 | 010 |
| Reference value bits at the same position as Df2 (Denoted as (1)) | | 111 | 011 | 111 | 001 |

\[
(1)+(2) =
\]
Contribution of Df2 on the bits of the binary addition which are at the same position than the Df2 bits

\[
(1)+(2) =
\]
Contribution of Df2 on the bits of the binary addition which are at the same position than the Df1 bits

| Reference value | 0..0|111 | 0..0|011 | 0..0|111 | 0..0|001 |
|-----------------|-----|-----|-----|-----|-----|-----|-----|
| Negative DIFF difference (Df1 | Df2) | 1..1|100 | 1..1|010 | 1..1|011 | 1..1|001 |
| Df2 (Denoted as (2)) | | 100 | 010 | 011 | 001 |
| Reference value bits at the same position as Df2 (Denoted as (1)) | | 111 | 011 | 111 | 011 |

\[
(1)+(2) =
\]
Contribution of Df2 on the bits of the binary addition which are at the same position than the Df2 bits

\[
(1)+(2) =
\]
Contribution of Df2 on the bits of the binary addition which are at the same position than the Df1 bits

| Reference value | 0..0|111 | 0..0|011 | 0..0|111 | 0..0|001 |
|-----------------|-----|-----|-----|-----|-----|-----|-----|
| Negative DIFF difference (Df1 | Df2) | 1..1|100 | 1..1|010 | 1..1|011 | 1..1|001 |
| Df2 (Denoted as (2)) | | 100 | 010 | 011 | 001 |
| Reference value bits at the same position as Df2 (Denoted as (1)) | | 111 | 011 | 111 | 011 |

\[
(1)+(2) =
\]
Contribution of Df2 on the bits of the binary addition which are at the same position than the Df2 bits

\[
(1)+(2) =
\]
Contribution of Df2 on the bits of the binary addition which are at the same position than the Df1 bits

Table G-2: Df2 contribution of a positive DIFF difference value on the binary addition between the DIFF difference and the reference value

It can be observed that whereas the contribution of Df2 bits on the bits of the binary addition situated at the same position as the Df2 bits is unpredictable, the Df2 contribution on the bits of the binary addition situated at the same position as the Df1 bits is easily predictable. In fact, there are only two types of contribution. The Df2 bits do not contribute at all on the bits situated at the same position as the Df1 bits, or the Df2 bits add 1 bit to the LSB bit of the bits situated at the same position as the Df1 bits.

Once a Df2 contribution of a positive DIFF difference value is analyzed, we continue with the Df2 contribution of a negative DIFF difference value. The contribution of a negative value is shown in complement-2 format. Table G-3 shows the examples.
The Df2 contribution of a negative DIFF difference value is also easily predicted. The contribution on the bits of the binary addition situated at the same position as the Df2 bits is unpredictable and there are two types of the Df2 contributions on the bits of the binary addition situated at the same position as the Df1 bits. The Df2 bits do not contribute at all on the bits situated at the same position as the Df1 bits, or the Df2 bits add 1 bit to all the bits situated at the same position as the Df1 bits.

Once the Df2 contributions of a negative and a positive DIFF difference value have been analyzed, we know all the possible variations of the bits of the binary addition between the reference value and the DIFF difference value which are situated at the same position as the Df1 bits. Therefore, we can implement a method allowing the exploitation of the DIFF difference value as the vector indicating the bits which can be predicted. This method consists in generating a new reference value such that when it is used in the simplification process yields a XOR difference value equal to the DIFF difference value obtained with the original reference value.

G.1.2. DIFF difference method

The method allowing the creation of the new reference value commented at the end of the previous section consists in 3 steps. The first step identifies the different parts of the DIFF difference value, Df1 and Df2. The second step creates 3 possible new reference values which cover all the possible Df2 contributions on the bits of the binary addition between reference value and the DIFF difference which are situated at the same position as the Df1 bits. Finally, the third step applies a criterion in order to determine among the 3 new reference values the one that correctly predicts the broadcasted ephemeris data set. Note that this prediction does not predict the bits situated at the same position than the Df2 bits.

In this section, only step 2 is presented since step 1 is easily accomplished and step 3 is presented in annex G.1.3. Step 2 describes the 3 new reference values and the justification of why these predictions cover all the possible Df2 contributions. The 3 new reference values are equal to the binary addition of the original reference value with the Df2 bits of either a positive or negative DIFF difference value. Each new reference value represents a different contribution of the Df2 bits on the binary addition. Remember that the bits situated at the same position as the Df2 bits are always unknown and thus, they are never predicted.

The 3 new reference values are given next.

1. **Normal prediction:** This reference value assumes that the Df2 contribution is 0 for either a positive or a negative DIFF difference value.
   
   Example: \( \text{mod}(010011xxx + 000001xxx, 2) = 010110xxx \)

2. **Addition prediction:** This reference value assumes that the Df2 contribution is equal to the addition of the LSB of the Df1 part. This contribution can be brought by a positive DIFF difference value.
   
   Example: \( \text{mod}(010011xxx + 000001xxx, 2) = 010110xxx \)

3. **Subtraction prediction:** This prediction reference value assumes that the Df2 contribution is equal to the sum of a vector of ones on the bits of the binary addition which are situated at the same position as the Df1 bits. This contribution can be brought by a negative DIFF difference value.
   
   Example: \( \text{mod}(010011xxx + 111111xxx, 2) = 010010xxx \)
It can be observed that these 3 predictions cover all the possible contributions of the Df2 part of the DIFF difference value. Therefore, it can be guaranteed that one of them correctly predicts the bits of the broadcasted ephemeris data set which are situated at the same position as the Df1 bits.

G.1.3. Selecting criterion and algorithm

The selection consists in choosing as the correct new reference value, the new reference value formed by the bits which pass the channel code verification after the reorganization of the bits in code words. The channel code verification is defined in annex E.1.2. Besides, note that this verification has to use the bits of the new reference value which cannot be predicted, the bits situated at the same position as the Df2 bits. The values of these non-predictable bits are determined by the typical demodulation process.

In order to optimize the selection process, the receiver can test the new reference values using the following method. First, the receiver correlates the predicted bits with the values of these predicted bits but obtained by the typical demodulation process. Second and last, since there are 3 new reference values for each predicted Keplerian parameter, the code words of the message are constructed by first using the predicted Keplerian parameters providing the greatest correlation peaks. If the verification of the channel word fails for any of the constructed code words, the method uses the next predicted Keplerian parameter having the greatest correlation peak in order to construct a new code word. Therefore, the first group of Keplerian parameters trying to pass the channel code verification is the group formed by the new reference value of each Keplerian parameter obtaining the greater correlation peak.

G.2. Process of simplification using the almanac data

The process of simplification using the almanac data consists in first choosing the proper almanac data set to be compared to the ephemeris data set, because the almanac data sets are broadcasted about every day and the ephemeris data sets are broadcasted about each 2 hours. Once the selection has been done, the process continues by calculating the XOR and DIFF difference, and the bit error probabilities.

The association between almanac data and ephemeris data is not trivial, because there are more ephemeris data samples than almanac data samples, although the almanac data samples have a longer validity.

- 1 ALMA file every day (1 sample/day) – 6 days of validity
- 1 EPHEMERIS file every 2 hours (1 sample/2h) – 4h of validity

Therefore, we subtract the ephemeris data of day (n+1)th to the almanac data transmitted in day n th, because the almanac data validity is 6 days. Consequently, the orbit represented by the n th almanac data corresponds to the orbits represented by the (n+1) th ephemeris data. The resulting difference signal, y, can be modeled as:

\[ y[x] = \text{alma}(x/12) - \text{ephemeris}(x+12) \]  

(G-2)

Where

- y: XOR or DIFF difference between the n th day almanac data set and the (n+1) th day ephemeris data sets.

371
G. Temporal methods of the binary prediction of the ephemeris data

- **x**: Index of 2-hour delta set
- **alma(t)**: Almanac data of day t
- **ephemeris(t)**: Ephemeris data of t set of 2 hours
- **\([t]\)**: Lower integer part of number t

Nevertheless, whereas the ephemeris data are normally broadcasted at the same time every day, the transmission time of the almanac data varies with each day. Moreover, the orbit described by either the ephemeris data or by the almanacs data has an interval of validity such that at their middle point, the described orbit is closer to the real orbit. Therefore, instead of the calculating the difference signal as has been specified in (G-2), we can build another difference signal where each ephemeris data is subtracted to the almanac with interval center closest to the interval center of the ephemeris data. The optimal difference signal can be modeled as:

\[
y'[x] = \text{alma}(f[x]) - \text{ephemeris}(x+12)
\]  
\[(G-3)\]

Where:

- **y'[x]**: XOR or DIFF difference between the ephemeris data set n and the almanac data set m with interval center the closest to the ephemeris data set n interval center.
- **f[x]**: Function which searches for the almanac data set with interval center the closest to the ephemeris data set (x+12) interval center

This last difference signal should provide better results than the first defined difference signal since it is based on the optimal subtraction. However, the results with the optimal subtraction are not significantly better than the results with the first defined signal. Moreover, the last difference signal introduces a periodicity in the residual data which is more complex to predict than the periodicity introduced by the first difference signal. Therefore, we only analyze the first defined signal performance.

Once the association almanac data – ephemeris data pair has been made, the XOR and DIFF distance calculation can be done. However, the almanac data and the ephemeris data are represented with a different number of bits. Therefore, in order to have both parameters with the same binary representation and thus in order to allow the XOR and DIFF difference calculation, the almanac data has to be transformed. This transformation consists in converting the almanac data to decimal format and in quantifying the decimal format with the same number of bits as the ephemeris data.

### G.3. Process of the simplification using long term prediction ephemeris data

In this section, the process of the simplification using long term prediction ephemeris data is presented. This process should only compare broadcasted ephemeris data to the ephemeris data provided by the part of the TAS program responsible for transforming the XYZ coordinates of a satellite orbit into a set of Keplerian parameters as specified in section 4.3.1.1.4. Nevertheless, in this section, the process of the simplification using the complete program is presented and later is customized for the transforming part of the program.

The process of the simplification using long term prediction ephemeris data is similar to the process of the simplification using the almanac data. First, we construct a difference signal, which consists in selecting the ephemeris data - long term predicted ephemeris data pairs having the closest interval of validity centers. Second, we convert the two values into the
same binary format. Third and last, we calculate the probability of error of each Keplerian parameter as specified in section 4.3.1.1.3. The first step of the process is further detailed below because it varies depending on the used long term orbital prediction program.

In this dissertation, we use a long term orbital prediction program provided by one of the thesis sponsors, THALES ALENIA SPACE - France (TAS-F). This program outputs one ephemeris data set every 4h whereas the GPS L1 C/A signal broadcasts one ephemeris data set every 2h. Therefore, we subtract the same long term predicted ephemeris data set to 2 consecutive broadcasted GPS L1 C/A ephemeris data sets. In other words, a TAS ephemeris data set having a validity period going from \( t \) to \( t+4h \), is compared with the broadcasted ephemeris data sets having a validity period going from \( t \) to \( t+4h \) and from \( t+2h \) to \( t+6h \). The reason of choosing these two specific broadcasted ephemeris data sets is that their theoretical time of application goes from \( t \) to \( t+2h \) for the first set and from \( t+2h \) to \( t+4h \) for the second set. And the combined time of application of the broadcasted ephemeris data sets is equal to the interval of validity of the TAS ephemeris set. A mathematical model for the difference signal is given below:

\[
y(x) = TAS\_ephemeris(\lfloor x/2 \rfloor) - ephemeris(x)
\]  

\text{(G-4)}

Where

- \( y \): XOR or DIFF difference between a TAS ephemeris data set and two consecutives broadcasted ephemeris data sets with interval centers closest to the TAS ephemeris data set interval center.
- \( x \): Index of 2-hour delta set
- \( TAS\_ephemeris(t) \): TAS ephemeris data of \( t \) set of 2 hours
- \( ephemeris(t) \): Ephemeris data of \( t \) set of 2 hours
- \( \lfloor t \rfloor \): Integer part of number \( t \) data sets

The method chosen to inspect the second source of error is the same method as the method used for the analysis of the simplification process using long term prediction ephemeris data. The only difference is the generation of the difference signal which no longer uses the TAS ephemeris data sets as one of its inputs. On the contrary, this method uses ephemeris data sets output by the XYZ-orbit-coordinates-to-Keplerian-parameters TAS conversion program when using as inputs to this program the orbits defined by the GPS L1 C/A broadcasted ephemeris data sets. More specifically, first, we obtain the orbit XYZ coordinates defined by the GPS L1 C/A broadcasted ephemeris data sets and second, from these XYZ orbit coordinates, we calculate by applying the TAS program part in charge of the XYZ-ephemeris transformation the new ephemeris data sets representing the broadcasted ephemeris data sets. Note that the TAS program continues to provide one ephemeris set each 4h which means that the difference signal is constructed as indicated in (G-4) but using the new defined input.

\section*{G.4. Justification of the results of the simplification using TAS long term prediction ephemeris data}

The simplification process using the TAS program of long term prediction ephemeris data is not satisfying enough. In fact, it is surprising to observe such a significant difference between the GPS L1 C/A broadcasted ephemeris data sets and the ephemeris data sets extrapolated
from the XYZ orbit ordinates defined by the same GPS L1 C/A broadcasted ephemeris data sets.

The first plausible explanation is that the transforming program of TAS does not work properly. However, the distance between an orbit defined by the broadcasted ephemeris data set and an orbit defined by the transformed ephemeris data set proves the contrary. In fact, Figure G-1 shows a really small distance between the orbits, and thus confirms that the program works very well.

There are two other possible explanations that justify this large binary divergence. The first one is that although the THALES program is able to obtain a really good accuracy in the decimal domain, the accuracy of the program is not enough to obtain a satisfactory binary performance as it has been explained in section 4.2.5. However, this explanation is inconsistent with the analysis conducted in annex F.4 about the influence on the satellite estimated position of the last bits of the Keplerian parameters. This analysis showed that the wrong prediction of the last bits of the Keplerian parameters changed the orbit defined by the Keplerian parameters by a distance far larger than the distances observed in Figure G-1. Therefore, the first explanation cannot be applied in this case.

The second explanation is that the ephemeris set found by the TAS converter software is not the same ephemeris data set as the broadcasted ephemeris data set. In fact, the objective of the XYZ-ephemeris converter part of the TAS long term orbital prediction program is to find an ephemeris data set which has a minimal distance to the orbit defined by the broadcasted ephemeris data set. Or, in other words, the program does not seek to optimize the binary distance between the converted Keplerian parameters and the broadcasted Keplerian parameters but rather the orbit distance. Therefore, whereas the distance between orbits is small, the Keplerian parameters binary difference is significant.

The justification of the second explanation can be found by inspecting the operations conducted by the TAS converter program and its inputs. First, the system is overestimated because the number of input points describing the orbit during the 4h period is larger than 16, the number of Keplerian parameters. Second, the function relating the ephemeris data to the XYZ orbit positions is not invertible; therefore the program is not ideal and an approximated model which relates the XYZ orbit position to the ephemeris data set has to be used. Third, the program implements a converging model based on iterations. Fourth, since the program is
not ideal and a unique solution cannot be determined, a new criterion has to be established in order to choose an acceptable solution. This criterion is that the distance between the broadcasted ephemeris data orbit and the converted ephemeris data orbit is smaller than a given threshold. Moreover, some other complementary criterions are applied such as the maximal number of iterations, the relative gain between iterations, etc. Finally, since there are several ephemeris data sets fulfilling the previous conditions, the program provides one of them. Therefore, the program has not calculated an ephemeris data set equal to the broadcasted data set, but rather an ephemeris data set fulfilling other conditions. Consequently, the second explanation is justified.

Another justification of this explanation is presented on the following tables. The first table shows some distances between broadcasted ephemeris data orbit positions and converted ephemeris data orbit positions. The second table shows the bit of most weight which is not equal between the broadcasted ephemeris data and the converted ephemeris data.

These distances shown in Table G-2 are larger for the left part of this table than for its right part. However, Table G-3 shows that the converted ephemeris data sets of its left part have more correctly predicted bits than the converted ephemeris data sets of its right part.

<table>
<thead>
<tr>
<th>Ephemeris Set Id</th>
<th>488</th>
<th>725</th>
<th>6369</th>
<th>6392</th>
<th>6583</th>
<th>124</th>
<th>130</th>
<th>878</th>
<th>1092</th>
<th>1432</th>
</tr>
</thead>
<tbody>
<tr>
<td>Orbit Distance (meters)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2,107</td>
<td>1,288</td>
<td>0,564</td>
<td>2,687</td>
<td>0,347</td>
<td>0,387</td>
<td>0,066</td>
<td>0,102</td>
<td>0,076</td>
<td>0,383</td>
<td></td>
</tr>
<tr>
<td>0,986</td>
<td>0,061</td>
<td>0,072</td>
<td>1,392</td>
<td>0,150</td>
<td>0,209</td>
<td>0,112</td>
<td>0,344</td>
<td>0,272</td>
<td>0,261</td>
<td></td>
</tr>
<tr>
<td>0,177</td>
<td>0,657</td>
<td>0,402</td>
<td>0,395</td>
<td>0,381</td>
<td>0,367</td>
<td>0,106</td>
<td>0,319</td>
<td>0,246</td>
<td>0,417</td>
<td></td>
</tr>
<tr>
<td>0,437</td>
<td>0,828</td>
<td>0,421</td>
<td>0,312</td>
<td>0,344</td>
<td>0,242</td>
<td>0,066</td>
<td>0,113</td>
<td>0,088</td>
<td>0,244</td>
<td></td>
</tr>
<tr>
<td>0,655</td>
<td>0,466</td>
<td>0,138</td>
<td>0,637</td>
<td>0,048</td>
<td>0,055</td>
<td>0,063</td>
<td>0,157</td>
<td>0,136</td>
<td>0,075</td>
<td></td>
</tr>
<tr>
<td>0,577</td>
<td>0,408</td>
<td>0,433</td>
<td>0,630</td>
<td>0,494</td>
<td>0,264</td>
<td>0,073</td>
<td>0,354</td>
<td>0,283</td>
<td>0,329</td>
<td></td>
</tr>
<tr>
<td>0,245</td>
<td>1,64</td>
<td>1,214</td>
<td>0,318</td>
<td>1,216</td>
<td>0,310</td>
<td>0,102</td>
<td>0,356</td>
<td>0,275</td>
<td>0,334</td>
<td></td>
</tr>
<tr>
<td>0,328</td>
<td>3,14</td>
<td>2,146</td>
<td>0,261</td>
<td>2,064</td>
<td>0,0760</td>
<td>0,088</td>
<td>0,111</td>
<td>0,021</td>
<td>0,095</td>
<td></td>
</tr>
</tbody>
</table>

Table G-4: Distances between the broadcasted ephemeris data orbit and the TAS converted ephemeris data orbit
Therefore, the comparison of these two tables shows that the objective of the TAS converter program of searching the ephemeris data set which defines the closest orbit to the orbit defined by the broadcasted ephemeris data set does not take into consideration the binary distances between the ephemeris data set. And this means that this program is not ideal to make a binary prediction of the Keplerian parameters of the broadcasted ephemeris data sets.

### G.5. Blackman-Tuckey method

In this section, first, the mathematical expressions of the Blackman-Tuckey method used to relate the spectral components of the observed signal with the amplitude and frequency of the sinusoids forming the observed signal on the time domain are presented. Second, the method used to estimate the phase of the sinusoids forming the observed signal is presented. Third and last, the complete process using the Blackman-Tuckey spectral estimation is given.

#### G.5.1. Time-frequency mathematical models of the sinusoids amplitude and frequency

In order to find the mathematical relationships between the amplitude and frequency of a sinusoid in the time domain and the characteristics of the spectral component peak of the same sinusoid in the frequency domain, we apply the Blackman-Tuckey estimation to a generic sinusoid. This sinusoid is considered as a random signal since, although the amplitude and frequency are determined, the phase is uniformly distributed over $[0, 2\pi]$. 

<table>
<thead>
<tr>
<th>Ephemeris Set Id</th>
<th>488</th>
<th>725</th>
<th>6369</th>
<th>6392</th>
<th>6583</th>
<th>124</th>
<th>130</th>
<th>878</th>
<th>1092</th>
<th>1432</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\sqrt{A}$</td>
<td>24</td>
<td>24</td>
<td>24</td>
<td>29</td>
<td>25</td>
<td>21</td>
<td>21</td>
<td>21</td>
<td>21</td>
<td>21</td>
</tr>
<tr>
<td>$E$</td>
<td>20</td>
<td>22</td>
<td>21</td>
<td>24</td>
<td>24</td>
<td>19</td>
<td>19</td>
<td>19</td>
<td>18</td>
<td>22</td>
</tr>
<tr>
<td>$\omega$</td>
<td>17</td>
<td>17</td>
<td>20</td>
<td>24</td>
<td>18</td>
<td>15</td>
<td>15</td>
<td>15</td>
<td>16</td>
<td>15</td>
</tr>
<tr>
<td>IDOT</td>
<td>6</td>
<td>9</td>
<td>8</td>
<td>11</td>
<td>13</td>
<td>6</td>
<td>8</td>
<td>6</td>
<td>7</td>
<td>11</td>
</tr>
<tr>
<td>$\dot{\Omega}$</td>
<td>19</td>
<td>18</td>
<td>21</td>
<td>15</td>
<td>21</td>
<td>15</td>
<td>17</td>
<td>15</td>
<td>17</td>
<td>16</td>
</tr>
<tr>
<td>$\Delta n$</td>
<td>11</td>
<td>10</td>
<td>12</td>
<td>7</td>
<td>12</td>
<td>9</td>
<td>9</td>
<td>9</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>$M0$</td>
<td>1</td>
<td>17</td>
<td>20</td>
<td>3</td>
<td>18</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>CUC</td>
<td>9</td>
<td>9</td>
<td>12</td>
<td>10</td>
<td>13</td>
<td>7</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>11</td>
</tr>
<tr>
<td>CUS</td>
<td>9</td>
<td>10</td>
<td>9</td>
<td>13</td>
<td>10</td>
<td>12</td>
<td>12</td>
<td>11</td>
<td>9</td>
<td>7</td>
</tr>
<tr>
<td>CRC</td>
<td>9</td>
<td>10</td>
<td>10</td>
<td>13</td>
<td>11</td>
<td>11</td>
<td>11</td>
<td>11</td>
<td>9</td>
<td>8</td>
</tr>
<tr>
<td>CRS</td>
<td>9</td>
<td>10</td>
<td>13</td>
<td>13</td>
<td>14</td>
<td>7</td>
<td>7</td>
<td>8</td>
<td>7</td>
<td>11</td>
</tr>
<tr>
<td>CIC</td>
<td>11</td>
<td>16</td>
<td>14</td>
<td>9</td>
<td>14</td>
<td>10</td>
<td>13</td>
<td>11</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>CIS</td>
<td>10</td>
<td>11</td>
<td>11</td>
<td>14</td>
<td>16</td>
<td>9</td>
<td>10</td>
<td>10</td>
<td>13</td>
<td>13</td>
</tr>
<tr>
<td>$\Omega_o$</td>
<td>17</td>
<td>26</td>
<td>26</td>
<td>17</td>
<td>31</td>
<td>17</td>
<td>17</td>
<td>17</td>
<td>17</td>
<td>17</td>
</tr>
<tr>
<td>$I$</td>
<td>24</td>
<td>26</td>
<td>31</td>
<td>22</td>
<td>29</td>
<td>22</td>
<td>26</td>
<td>22</td>
<td>24</td>
<td>23</td>
</tr>
</tbody>
</table>

Table G-5: First different bit between the broadcasted ephemeris set and TAS converted ephemeris set
First of all, the definition of a sinusoid signal to observe, \( x[n] \), with a random phase modeled by a uniform distribution over \([0, 2\pi]\) is given below:

\[
x[n] = A \cdot \sin(2\pi f_0 n + \theta_0) \quad \forall n \in \mathbb{Z}
\]  

(G-5)

Where:
- \( A \): sinusoid amplitude
- \( f_0 \): sinusoid frequency
- \( \theta_0 \): random variable uniformly distributed over \([0, 2\pi]\)

The determination of the power density spectrum of the observed signal using the Blackman-Tuckey estimator is achieved by first estimating the signal autocorrelation. The signal autocorrelation is obtained by using an autocorrelation estimator. In this dissertation, we use the ‘biased’ autocorrelation estimator, \( R_{xx}[m] \), and its definition is given below [MARPLEa, 1987]:

\[
R_{xx}[m] = \frac{1}{N} \sum_{n=0}^{N-1} x[n+m] \cdot x[n] \quad 0 \leq |m| \leq N
\]  

(G-6)

Where:
- \( N \): Number of samples of the observed signal used in the autocorrelation estimation. Ideally chosen as \( N \cdot f_0 = k, k \in \mathbb{Z} \).

The estimated autocorrelation has a length of 2N-1 samples when the input signal has a length of N samples. This estimator can also be expressed as:

\[
R_{xx}[m] = \frac{1}{N} \sum_{n=1}^{\infty} x[n+|m|] \cdot \rho_N[n+|m|] \cdot x[n] \cdot \rho_N[n] \quad 0 \leq |m| \leq N
\]  

(G-7)

Where:
- \( \rho_N[n] = \) Rectangular window of length \( N \) starting at \( n=0 \) and ending at \( n=N-1 \)

Once we have an estimation of the autocorrelation of the observed signal, we apply the Blackman-Tuckey method in order to obtain an estimation of the power density function of the observed signal. The Blackman-Tuckey method consists in applying a Bartlett window of half the estimated autocorrelation size to the ‘biased’ autocorrelation estimator. Therefore, the Blackman-Tuckey signal, \( BT[m] \), can be modeled as:

\[
BT[m] = R_{xx}[m] \cdot v_{(N/2)}[m] \quad 0 \leq |m| \leq N
\]  

(G-8)

Where:
- \( v_{(N)}[n] = \) Triangular window of length \( 2N-1 \) centered at 0. The window starts at \( n=-\frac{(N-1)}{2} \) and ends at \( n=N-1 \).

Applying the DFT operator on the Blackman-Tuckey signal, we obtain the power spectrum density function of the observed signal when estimated by the Blackman-Tuckey method. And from this power spectrum density function, we can relate the amplitude and frequency of the observed sinusoid with the characteristics of the estimated spectral components of the signal.

However, the mathematical expression of the power spectrum density function is not easy to interpret since the signal, and the window are mixed. Therefore, in order to simplify the
interpretation, we analyze the mean of the Blackman-Tuckey signal and we transform the mean of the Blackman-Tuckey signal in order to obtain an approximation of the power spectrum density function of the observed signal which is much easier to analyze. Remember that the observed signal is a random signal since its phase is uniformly distributed over \([0, 2\pi]\).

The mean of the Blackman-Tuckey signal, \(BTm[m]\), is [MARPLEa, 1987]:

\[
BTm[m] = E[BT[m]] \quad 0 \leq |m| \leq N \quad (G-9)
\]

\[
BTm[m] = E[R_x[m]] \cdot v_{T(N/2)}[m] \quad 0 \leq |m| \leq N \quad (G-10)
\]

And since the observed signal is random and the windows are deterministic, the mean of the ‘biased’ autocorrelation estimator is:

\[
E[R_x[m]] = r_x[m] \cdot v_{T(N)}[m] \quad 0 \leq |m| \leq N \quad (G-11)
\]

Therefore, the mean of the Blackman-Tuckey signal, \(BTm[m]\), is:

\[
BTm[m] = r_x[m] \cdot v_{T(N)}[m] \cdot v_{T(N/2)}[m] \quad 0 \leq |m| \leq N \quad (G-12)
\]

The power spectrum density function of the observed signal when estimated with the Blackman-Tuckey method can be approximated by:

\[
\hat{S}_{BT}[k] = DFT[BT[m]] \approx \frac{1}{(2N)^2} S_x[k] \otimes DFT[v_{T(N)}[n]] \otimes DFT[v_{T(N/2)}[n + N/2]] \quad (G-13)
\]

Where:

- \(DFT[f[m]]\): Discrete Fourier transform of \(f[m]\)
- \(\otimes\): Circular convolution of signals of length equal to 2N samples

From expression (G-13), we can see that although the peaks of the ideal power density function of the observed sinusoid signal are lost due to the 2N circular convolution of the windows, the maximum value is still situated at ±\(f_0\).

Moreover, we can evaluate expression (G-13) at \(f_0\), if \(f_0 = (k_0/2N)\), in order to relate the peak value to the sinusoid amplitude. Therefore, if we approximate the power density function of the signal estimated by the Blackman-Tuckey method with another expression:

\[
\hat{S}_{BT}[k] = \sum_{n=-N/2+1}^{N/2-1} \left( \frac{A^2}{2} \cos(2\pi f_0 n) \cdot v_{T(N)}[n] \cdot v_{T(N/2)}[n] \cdot e^{-j2\pi f_0 n} \right) \quad (G-14)
\]

And if we only inspect the positive frequencies part, the expression is simplified to:

\[
\hat{S}_{BT}^+ [k] = \sum_{n=-N/2+1}^{N/2-1} \left( \frac{A^2}{4} \cdot v_{T(N)}[n] \cdot v_{T(N/2)}[n] \cdot e^{-j2\pi \left( \frac{k_0}{2N} - f_0 \right) n} \right) \quad (G-15)
\]

Finally, evaluating the expression at \((k_0/2N) = f_0\):

\[
\hat{S}_{BT}^+ [k_0] \approx \frac{A^2}{2} \sum_{n=0}^{N/2} \left( 1 - \frac{n}{N} \right) \left( 1 - \frac{2n}{N} \right) \quad (G-16)
\]
A table summarizing the relationship between the amplitude and the frequency of the observed sinusoid and the spectral components of the power density spectrum of the signal estimated by the Blackman-Tuckey method are presented below.

<table>
<thead>
<tr>
<th>Sinusoid Frequency</th>
<th>Sinusoid Amplitude</th>
</tr>
</thead>
<tbody>
<tr>
<td>$f_0 = \text{Ordinate X of the peak}$</td>
<td>$A = \sqrt{\frac{48 \cdot \text{Peak Amp}(f_0)}{5 \cdot N}}$</td>
</tr>
</tbody>
</table>

Table G-6: Time-frequency sinusoid amplitude and frequency mathematical relations

Nevertheless, note that we have assumed that $f_0 = (k_0/2N)$; however this is hardly the case. Therefore, neither the frequency nor the amplitude is perfectly estimated.

G.5.2. Method of the estimation of the phase

The method implemented to estimate a sinusoid phase is explained next. This method needs to have estimates of the amplitude and the frequency of the sinusoid in order to estimate its phase.

The base of this method consists in generating a new signal which is equal to the subtraction of a sinusoid to the observed signal. The subtracted sinusoid is the sinusoid which phase is being estimated. The sinusoid is generated several times, each time with a different phase. The method chooses the phase that minimizes the peak amplitude of the spectral component of the residual spectrum (difference between the observed signal and the estimated signal) situated at the same frequency as the frequency of the sinusoid which phase is being estimated.

G.5.3. Blackman-Tuckey process

The complete Blackman-Tuckey process implemented to identify all the spectral components is presented next. The process consists in 5 steps:

1- Identification of the spectral component having the maximum peak amplitude.
2- Estimate of the amplitude and frequency of the sinusoid from theoretical expressions.
3- Estimate of the sinusoid phase from section G.5.2 method.
4- Subtract the new identified sinusoid to the signal being analysed in order to create a new signal without this spectral component.
5- Back to step 1 until reaching the desired number of identified sinusoids.

G.6. PRONY model

In this annex, the mathematical expression of the PRONY model as well as the method used to determine the order of the PRONY model, or, in other words, the number of sinusoids composing of the signal, are presented.


G. Temporal methods of the binary prediction of the ephemeris data

G.6.1. PRONY model mathematical expressions

The PRONY method models a signal with N data samples with the following mathematical expression [MARPLEd, 1987].

\[ \hat{x}[n] = \sum_{k=1}^{p} A_k \cdot e^{(\alpha_k + j2\pi f_k)nT + j\theta_k} \quad n = 0..N - 1 \]  

Where:
- \( \hat{x}[n] \): Signal Estimation
- \( T \): Sample interval
- \( A_k \): Exponential amplitude
- \( \alpha_k \): Damping factor (1/sec)
- \( f_k \): Sinusoid frequency (Hz)
- \( \theta_k \): Sinusoid initial phase (radians)
- \( p \): Number of exponentials

Nevertheless, for this study, this expression is simplified because the signal is real and the damping factor is equal to 0 for all the exponentials.

\[ \hat{x}[n] = \sum_{k=1}^{p/2} A_k \cdot \cos(2\pi f_k nT + \theta_k) \]  

Therefore, we need to find the amplitude, frequency and phase of each existing sinusoid. In order to do so, we express the original signal without taking into account any assumptions as:

\[ x[n] = \sum_{k=1}^{p} h_k \cdot z_k^n \quad n = 0..N - 1 \]  

\[ h_k = A_k \cdot e^{j\theta_k} \]  

\[ z_k = e^{(\alpha_k + j2\pi f_k)T} \]

Consequently, we can determine the characteristics of the sinusoids by solving the following equation obtained from (G-20):

\[
\begin{pmatrix}
z_1^0 & z_2^0 & \cdots & z_p^0 \\
z_1^1 & z_2^1 & \cdots & z_p^1 \\
\vdots & \vdots & \ddots & \vdots \\
z_1^p & z_2^p & \cdots & z_p^p \\
\end{pmatrix}
\begin{pmatrix}
h_1 \\
h_2 \\
\vdots \\
h_p \\
\end{pmatrix} =
\begin{pmatrix}
x[0] \\
x[1] \\
\vdots \\
x[p-1] \\
\end{pmatrix} \Rightarrow h = Z^{-1} \cdot x
\]

Therefore, we first determine the \( z_k \) coefficients, and second, we use equation (G-23) to determine the \( h_k \) coefficients.

In order to determine the \( z_k \) coefficients, we define a homogeneous linear constant-coefficient difference equation with the \( z_k \) coefficients as roots.
G. Temporal methods of the binary prediction of the ephemeris data

\[ \phi(z) = \prod_{k=1}^{p} (z - z_k) \]  

(G-24)

\[ \phi(z) = \sum_{m=0}^{p} a[m] z^{-m} \quad a[0] = 1 \]  

(G-25)

Where:

- \( z_k \): Polynomial roots
- \( a[k] \): Polynomial Coefficients

Therefore, we find the polynomial coefficients and we use them in order to find the \( z_k \) coefficients. The method selected in order to identify the \( a[k] \) coefficients is the modified covariance method [MARPLEd, 1987] because this method uses the forward and backward prediction. Therefore, the modified covariance method uses more points than the normal methods and thus uses more signal samples to find the \( a[k] \) coefficients. For a better explanation of the modified covariance method, the reader is directed to [MARPLEd, 1987]. Therefore, the determination of the \( a[k] \) coefficients by the modified covariance method is achieved by solving the following equation:

\[ a = -R^{-1}r \]  

(G-26)

\[ a = [a[1] \quad a[2] \ldots a[p]]' \]  

(G-27)

\[ R(i, j) = \sum_{n=p+1}^{N} [x[n-i]x[n-j] + x^*[n-p+i]x[n-p+j]] \quad i, j = 1..p \]  

(G-28)

\[ r(i) = \sum_{n=p+1}^{N} [x[n-i]x[n] + x^*[n-p+i]x[n-p]] \quad i = 1..p \]  

(G-29)

Where:

- \( x[n] \): analyzed signal at instant \( n \)

Therefore, using expression (G-26), we can find the \( a[k] \) coefficients defining a polynomial whose roots are the \( z_k \) coefficients. This means, that from the \( a[k] \) coefficients, we can determine the \( z_k \) coefficients, and using equation (G-23), we can find the \( h_k \) coefficients from the \( z_k \) coefficients.

Finally, once all the \( h_k \) and \( z_k \) coefficients have been determined, the parameters defining the exponentials components are identified using the following expressions [MARPLEd, 1987].

\[ A_i = |h_i| \]  

(G-30)

\[ \theta_i = \arctan \left( \frac{\text{Im}(h_i)}{\text{Re}(h_i)} \right) \quad \text{radians} \]  

(G-31)

\[ \alpha_i = \frac{\ln |z_i|}{T} \quad 1/\text{sec} \]  

(G-32)

\[ f_i = \frac{\arctan \left( \frac{\text{Im}(z_i)}{\text{Re}(z_i)} \right)}{2\pi T} \quad \text{Hz} \]  

(G-33)

Remember that we have previously said that our signal is such that the damping factor of all the exponentials is equal to 0; therefore, equation (G-32) is not used.
Moreover, since the analysed signal is real, the number of exponentials has to be an even number. Therefore, from equations (G-30), (G-31) and (G-33), we must find twice the same amplitude, one positive frequency and one negative frequency with the same absolute value and one positive phase and one negative phase with the same absolute value.

G.6.2. PRONY model order selection

In the previous section, it has been shown how to determine the sinusoids characteristics from the analyzed signal. However, the previous section assumed that the number of exponentials is known. However, the number of exponentials is usually unknown, and it is one of the model parameters to be determined. Therefore, the first action to undertake before beginning with the estimation of the $a[k]$ coefficients is to determine the number of sinusoids forming the analyzed signal. Note that twice the number of sinusoids is equal to the number of exponentials, also called the model order. Therefore, we need first to determine the model order.

There are different methods which can be used in order to find the number of exponentials forming a signal. And these methods can be sorted out into two main groups, the methods which base their criterion on the signal statistics properties [MARPLEc, 1987] and the methods which uses the signal singular values (SVD or singular value decomposition) [MARPLEd, 1987]. Each group of methods has its advantages and drawbacks such as the percentage of times that the number of exponentials is exactly found, upper or under estimations, or their processing time.

In this case, the group of methods best suiting our needs is the second type of methods. The reason is that the first type of methods provides a good performance when the number of exponentials generating the signal is low, between 3 and 5, but these methods provide an underestimation when the number of exponentials is larger. Moreover, the methods based on signal statistics properties, in addition of being less robust to the influence of the signal noise, obtain an abnormal estimation when the difference signal resulting from the application of the simplification process using almanac data is inspected. However, the method based on the signal singular values is quite slower than the methods based on signal statistics properties.

The main idea of the method based on the signal singular values is to differentiate the singular values associated to the signal exponentials from the singular values generated by the signal additive noise. Therefore, since the singular values of the exponentials are some orders of magnitude larger than the singular values of the noise, a threshold can separate the two types of singular values. And thus, this separation allows the identification of the total number of exponentials forming the signal.

Therefore, this method consists of two steps. First, we calculate the singular values of the matrix of the signal associated to the forward prediction, $X_f$, the singular values of the matrix of the signal associated to the backward prediction, $X_b$, or the singular values of the union of both matrices. The mathematical expression of the matrixes is given below:

$$
X_f = \begin{pmatrix}
x[q] & x[q-1] & \cdots & x[1] \\
x[q+1] & x[q] & \cdots & x[2] \\
\vdots & \vdots & \ddots & \vdots \\
x[N-1] & x[N-2] & \cdots & x[N-q]
\end{pmatrix}
$$

(G-34)
Second, we determine the number of exponentials forming the signal as the number of singular values larger than a fixed threshold. Note that $q$ is the maximum model order possible when using the matrixes $X_q^f$ and $X_q^b$. Therefore, if the method application results into a number of exponentials $p$ equal to the maximum considered model order minus one, $p = q - 1$, the method has to be applied again with a bigger value $q$. And this process is repeated until $p$ is lower than $q - 1$. Nevertheless, for the analyzed signal, if $p$ is too big, the application of the method results into the dummy solution, $p = q - 1$.

**G.7. Artificial Neural Network method**

In this section, the neural network structure is given and the learning phase of a neural network is described.

**G.7.1. Neural Network structure**

The fundamental and smaller element of a neural network is the neuron. Each neural network is formed by several neurons distributed in different layers, where each layer is hidden from the user. For example, Figure 4-3 presents a neural network with only one hidden layer. Nevertheless, we implemented only 1 layer neural networks.

A general neuron structure is given below [BHADESHIA, 1999] [YEGNANARAYANAa, 2006].

![Figure G-2: Generic neuron model](image)

From Figure G-2, it can be seen that a neuron is formed by a $g$ function, also called combination function, a neuron bias, also called activation threshold and represented by the $b$ variable, and a $f$ function, also called the activation function. The function of these elements is presented next.

The combination function is responsible for manipulating the neuron inputs in order to create a suitable input to the activation function. Moreover, the choice of the combination function defines the type of implemented neural network. There are two main types of neural networks, the Multi-layer Perceptron (MLP) and the Radial Basis Function (RBF). The MLP
neural networks combination function implements a linear combination of the neuron inputs [YEGNANARAYANAc, 2006].

\[
g_j(P) = \sum_{i=1}^{N} w_{ij} \cdot p_i
\]  

(G-36)

Where:
- \( g_j(x) \): \( j^{th} \) neuron combination function
- \( P \): vector of neuron inputs
- \( p_i \): \( i^{th} \) neuron input
- \( w_{ij} \): \( i^{th} \) neuron input weight of the \( j^{th} \) neuron allowing the correct output prediction

Note that the neuron input is in our case the known samples of the difference signal to be predicted. The last known samples of the difference signal before the first unknown sample to be predicted.

The RBF neural networks combination function calculates the distance or norm between the neuron inputs, interpreted as a vector, and a given vector [CHEN et al, 1991].

\[
g_j(P) = \| P - P_j \|
\]  

(G-37)

Where:
- \( P_j \): \( j^{th} \) neuron vector of delays allowing the correct output prediction

The activation function is responsible for introducing a non-linearity inside the neuron model. There are several activation function choices such as the sigmoid function, the Heaviside function and some trigonometric functions [YEGNANARAYANAb, 2006]. We chose to implement the Arctangent function.

The activation threshold is used to determine if the combination function output has reached a fixed value or threshold. Therefore, the activation threshold determines the input sign of the activation function [YEGNANARAYANAb, 2006].

Until now, we described the generic structure of a neuron; however, any neural network is formed by several neurons. Therefore, the neuron outputs are either linearly or non-linearly combined in order to provide one output or several outputs of the neural network which suit the needs of the problem. In our case, we implemented a linear combination of the neuron outputs in order to obtain a single final sample. A scheme of a generic neural network of 1 layer is given below [BHADRESHIA, 1999].
Therefore, the mathematical expression of the neural network output at instant n is [YEGNANARAYANAc, 2006]:

\[ \hat{x}[n] = d + \sum_{j=1}^{S} c_j \cdot f \left( g_j(p) + b_j \right) \]

(G-38)

Where:
- \( \hat{x}[n] \): neural network output at instant n
- d: external constant
- \( c_j \): neuron weights

Customizing expression (G-38) first for a MLP neural network and second for a RBF neural network, we obtain the following equations [YEGNANARAYANAc, 2006] [CHEN et al, 1991].

\[ \hat{x}[n] = d + \sum_{j=1}^{S} c_j \cdot f \left( b_j + \sum_{i=1}^{R} w_j \cdot x[n-i] \right) \]

(G-39)

\[ \hat{x}[n] = d + \sum_{j=1}^{S} c_j \cdot f \left( \|X - X_j\| \right) \]

(G-40)

Where:
- \( x[n-i] \): signal input at instant n-i
- X: vector of signal inputs from \( x[n-1] \) to \( x[n-R] \)
- \( X_j \): constant \( j^{th} \) neuron vector of delays
G.7.2. Neural Network learning phase

All the values of the parameters defined in the previous section, such as the neuron weights, the activation thresholds, the input weights of each neuron and the external constant, have to be correctly determined in order to allow the desired signal prediction. Therefore, the first step to take in order to predict a signal by using a neural network is to launch the neural network learning phase which is responsible for determining the values of the neural network parameters.

More specifically, this phase consists in varying the values of the parameters in order to find the values which minimize the power of the error between the value of the predicted sample and the value of the sample which is being predicted. The power of the error is denoted as \( E \).

Since we are applying the learning phase and the value of the sample which is being predicted has to be known, the values of the neural network output and the neuron inputs are the following. The sample of the neural network output is a known sample of the difference signal and the neuron inputs are the known delayed samples of the signal which should be used to predict the output sample if it was unknown. The variation of the value of the constant parameter is achieved as specified below [YEGNANARAYANAb, 2006].

\[
parameter[n+1] = parameter[n] + \mu \frac{\partial E}{\partial parameter} 
\]

\[
E = \frac{1}{2} \sum_{i=1}^{L} (e_i[n])^2 
\]

\[
e_i[n] = x_i[n] - \hat{x}_i[n] 
\]

\[
\frac{\partial E}{\partial parameter} = \frac{1}{\partial parameter} \left( \frac{1}{2} \sum_{i=1}^{L} \left( x_i[n] - d - \sum_{j=1}^{S} c_j \cdot f [g_j(P) + b_j] \right)^2 \right) 
\]

Where:

- \( e_i(n) \): Prediction error between the \( i^{th} \) known signal sample \( x_i[n] \) and the \( i^{th} \) predicted sample \( \hat{x}_i[n] \) at instant \( n \). \( \hat{x}_i[n] \) is modeled as indicated in equation (G-38).
- \( L \): number of samples to predict/estimate. In our case, \( L \) is equal to 1.
- \( \mu \): learning rate.

The learning rate must not be too large because the neural network would not converge. And it must not be too small because the converging process can be too slow.

Finally, the specific steps of the learning process are explained below.

1. Random selection of a group of consecutives signal samples.
2. The oldest signal samples are used as the neural network inputs, or neuron inputs, to calculate a signal sample prediction of the latest signal sample of the group.
3. The signal sample prediction is subtracted to the latest signal sample of the group in order to calculate the prediction/estimation error.
4. The variations of the parameter values, neuron weights, the activation thresholds, the input weights of each neuron and the external constant, are calculated from the prediction error and the inputs.
5. Back to step 1 until a certain number of iterations is reached.
Annex H. Transmission channels and simulators schemes

In this annex, the simulator schemes used in this dissertation in order to simulate the signal transmission through an AWGN channel and through a mobile channel are presented. Moreover, the mathematical expression of the autocorrelation function of a mobile channel is given and the mathematical model of a frequency-selective channel is described.

H.1. Autocorrelation function of a mobile channel

In order to determine the autocorrelation function of a channel, first the received signal and the channel impulse response has to be modeled.

The received signal is modeled as the addition of the LOS signal and the multipath component. The multipath component or the echoes can be modeled as a succession of transmitted signals arriving at the receiver antenna at different delays since they are the refractions, reflections, etc, of the transmitted signal arriving at the receiver antenna by another path different from the LOS path. Moreover, due to the variability of the surroundings, the movement of the transmitting satellite and the receiver, the delay, the phase and the attenuation of the echoes vary at each instant. Therefore, the received band-pass signal at the receiver antenna output can be expressed as [PROAKISg, 2001]:

\[ v(t) = \sum_n \alpha_n(t)s[t - \tau_n(t)] \]  \hspace{1cm} (H-1)

With:
- \( v(t) \): received band-pass signal
- \( s(t) \): transmitted band-pass signal
- \( \alpha_n(t) \): Complex attenuation factor for the \( n^{th} \) path
- \( \tau_n(t) \): Delay of the \( n^{th} \) path

The transmitted signal can be also expressed using its equivalent baseband complex envelope expression, \( s(t) \): 

\[ s(t) = \text{Re}[s_c(t) \cdot \exp\{j2\pi f_c t\}] \]  \hspace{1cm} (H-2)

With:
- \( f_c \): Carrier frequency of the signal

Thus, the equivalent baseband complex envelope received signal, \( v_c(t) \), can be expressed as:

\[ v_c(t) = \sum_n \alpha_n(t) \cdot \exp[-j2\pi f_c \tau_n(t)] \cdot s_c[t - \tau_n(t)] \]  \hspace{1cm} (H-3)

Finally, the equivalent baseband complex envelope channel impulse response, \( c(\tau, t) \), can be described as:

\[ c(\tau; t) = \sum_n \alpha_n(t) \exp[-j2\pi f_c \tau_n(t)] \delta[\tau - \tau_n(t)] \]  \hspace{1cm} (H-4)
From equation (H-4), parameter $t$ is used to indicate the instant of time at which the attenuation factor, $\alpha_n(t)$, and the propagation delay, $\tau_n(t)$, of the $n$th echo are evaluated. Parameter $\tau$ is used to determine the echo which has a propagation delay equal to $\tau$.

Then, from this channel impulse response expression, the channel autocorrelation can be defined [PROAKISg, 2001]. The $c(\tau, t)$ is assumed wide-sense-stationary.

$$\phi_c(\tau_1, \tau_2; \Delta t) = \frac{1}{2} E\left[e^{j(\tau_1; t)} c(\tau_2; t + \Delta t)\right]$$

(H-5)

Where:

- $\tau_1, \tau_2$: Echoes propagation delays. Determine the propagation delay of the echoes which are compared.
- $\Delta t$: Channel variation time. Since the propagation delay, the phase shift and the attenuation factor of an echo vary with time, the channel impulse response also varies with time. Therefore the $\Delta t$ parameter is used to indicate the amount of time between the two channel impulse responses which are compared.

Therefore, assuming that the attenuation and phase shift of the channel associated with the echo having a propagation path delay equal to $\tau_1$ is uncorrelated with the attenuation and phase shift of the echo having a path delay equal to $\tau_2$, the Fourier transform of the autocorrelation function results in [PROAKISg, 2001]:

$$\phi_c(\Delta f; \Delta t) = \frac{1}{2} E\left[C^*(f_1; t) C(f_2; t + \Delta t)\right]$$

(H-6)

Where:

- $\Delta f$: Frequency difference between two frequencies of the mobile channel.
  
  $\Delta f = f_2 - f_1$

This Fourier transform is called spaced-frequency, spaced time correlation function of the channel [PROAKISg, 2001]. Therefore, imposing $\Delta t$ equal to 0, the channel autocorrelation as a function of the frequency is obtained from this expression and thus the channel coherence bandwidth can also be obtained. The channel coherence bandwidth, $(\Delta f)_c$, is the width of the region which is not null of the spaced-frequency, spaced time correlation function of the channel when $\Delta t$ is equal to 0 and its value only depends on the frequency difference, $\Delta f$. Then, since this autocorrelation specifies how the channel is correlated with itself for two different frequency values, we can assume that two sinusoids with frequency separation larger than $(\Delta f)_c$ are affected differently by the channel.

Moreover, from expression (H-6) and assuming $\Delta f$ equal to 0, the channel autocorrelation function variation along the time can be obtained. If the channel does not vary along the time, the channel is considered time-invariant and the autocorrelation function should be a constant value equal to 1. However, since the channel varies with time, the autocorrelation function has a limited time span, and the length of time region where the autocorrelation function is not zero is the called channel coherence time, $(\Delta t)_c$.

In other words, the delay, attenuation and phase of each received echo vary for each instant of time. More precisely, although the echoes are completely independent among them, they are not uncorrelated with themselves over time. This means that the values of delay, phase and attenuation of the $n$th echo at time $t_n$ are related with the same values at time $t_0$. Therefore, the $(\Delta t)_c$ represents the duration of time in which the channel remains about constant. This
variation of the channel is termed fading [PROAKISg, 2001]. Therefore, the longer the signal symbol duration is, the more the symbol amplitude and phase vary.

H.2. Frequency-selective and not slowly fading mathematical model

In the event that the transmission channel is frequency-selective and not slowly fading at the receiver correlator output, the final mathematical model can also be represented by the Perez-Fontan model. In this case, the only difference with the previous presented frequency non-selective channel model is the presence of the other tap delays –or main echoes- of Figure 3-7 and their complex coefficients $c_n(t)$. Therefore, if the distribution of the $c_n(t)$ coefficients is known, the frequency-selective channel may be easily reproduced

As previously said in section 3.2.2.2.2, the complex coefficient $c_n(t)$ models the contribution to the received signal amplitude and phase of a main echo and of all the received echoes between half the time to the previous main echo and half the time to the next main echo. Therefore, each complex coefficient $c_n(t)$ is independent from any other coefficient $c_n(t)$ but they all follow the same statistical distribution. Moreover, the complex coefficient $c_1(t)$, which models the contribution of the LOS signal and of the echoes received until half the time of the first main echo, is also different from any complex coefficients $c_n(t)$ but also follows a different statistical distribution. Nevertheless, the contribution made to a main echo by all the echoes received between half the time to the previous main echo and half the time to the next main echo is not statistically different from the contribution made to the LOS signal by the echoes received until half the time to the first main echo. In fact, the only difference is the statistic of the main echo with respect to the LOS signal. Therefore, since the complex coefficient $c_1(t)$ was modeled as a Loo distribution, the addition of a Log-normal variable representing the LOS signal and a Rayleigh variable representing the echoes, the complex coefficients $c_n(t)$ are directly modeled by a Rayleigh variable since they only represent the addition of echoes.

One last factor to take into account during the modeling of the complex coefficients $c_n(t)$ is the power carried by the LOS signal and each main echo. In fact, since the echoes travel more distance than the LOS signal plus the extra attenuation introduced by the obstacles, the average power of any echo is smaller than the average power of the LOS signal. And the same statement is verified for echoes received later than other echoes. Therefore, in order to fit this decrease of the average power to our $c_n(t)$ coefficients, we have to multiply each $c_n(t)$ by a factor which decreases with each growing n. These factors are extracted from the average power delay profiles database. Therefore, the complex coefficients $c_n(t)$ can be finally modeled as:

$$ c_n(t) = w \cdot e^{i\phi} \cdot P_m((n-1) \cdot T_c) $$

(H-7)

Where:
- $w$: Rayleigh variable with an average power equal to sqrt(MP) (see section 3.2.2.3.4)
- $\phi$: Uniform variable on [0, 2\pi]
- $P_m(\tau)$: Linear average power delay profile evaluated at $\tau$
An example of an average power delay profile is given below:

![Figure H-1: Typical average power delay profile for suburban and urban areas](image)

Where:
- $P_m(\tau)$: Linear average power delay profile

Finally, since the transmission channel is frequency-selective and not slowly fading at the correlator output, the correlation between the received signal PRN code and the generated local replica is not ideal and thus the main echoes are not eliminated. This means that if we want to implement a simulator for this kind of channel, the correlation between the received signal PRN code and the generated local replica has to be completely recreated.

To sum up, the mathematical model of a frequency-selective channel which is not slowly fading is the LOS signal plus $L$ main echoes separated from each other by the signal symbol duration, $T$, -in this case $T$ is equal to $T_c$- where $L = \lceil \frac{T_m}{T_c} \rceil + 1$. The LOS signal is multiplied by the complex coefficient $c_1(t)$ which follows a Loo distribution (defined in section 3.2.2.3.4) and each main echo is multiplied by a complex coefficient $c_n(t)$ which follows a Rayleigh distribution. The distribution parameters depend on the Markov state at which the model is at each instant. Moreover, each complex coefficient of the main echoes is multiplied by a factor simulating the channel average power delay profile. Finally, the correlation between the generated local replica and the complete receive signal –LOS signal plus main echoes- has to be completely calculated.

**H.3. Simulator schemes**

The simulator schemes of the transmission of a signal through an AWGN channel or a mobile channel are presented next.

**H.3.1. AWGN channel simulator**

The AWGN channel simulator depends on the tracking process. Therefore, two cases are detailed in the following subsections.

**H.3.1.1. Ideal carrier phase tracking process**

An ideal carrier phase tracking is assumed when the $C/N_0$ at the antenna output is so high that it induces a negligible PLL carrier phase estimation error. And this means that no useful
signal power is lost: the useful signal power at the receiver antenna output is the same as the useful signal power at the demodulator block input.

\[
C_{\text{RF/IF filter output}} = C_{\text{Input Demodulator}} \quad (H-8)
\]

Therefore, since all the useful power can be used to demodulate the signal, the C/N\textsubscript{0} and E\textsubscript{b}/N\textsubscript{0} figures of merit can be linked as it is done in annex B.3. Remember that the demodulation/decoding performance expressed as a function of the E\textsubscript{b}/N\textsubscript{0} provides different information that the demodulation performance expressed as a function of the C/N\textsubscript{0} as it is also said in annex C.5.

Finally, in an AWGN channel, in order to simulate the received signal at the correlator output when an ideal carrier phase tracking process is assumed, we have to generate the model presented in equation (3-14) but neglecting the cosine term. More specifically, we generate the data symbols with a power equal to 1 plus a Gaussian process representing the AWG noise whose generation is described in annex H.3.1.3. Note that for the reception of a GNSS signal when the subcarrier has been removed, the data symbol is a BPSK symbol which means that is either represented by a +1 or -1 value. Moreover, the simulated signal is a baseband signal since it not necessary to use a more complex model.

**H.3.1.2. Non ideal carrier phase tracking process**

A non-ideal carrier phase tracking process happens when the C/N\textsubscript{0} at the antenna output is not large enough to achieve a negligible PLL carrier phase estimation error but still high enough to allow the PLL lock. And this means that not all the useful signal power can be used to demodulate the signal.

There are two ways of obtaining the signal demodulation performance when the carrier phase tracking process is not ideal. The first one consists in averaging the demodulation performance values of an ideal carrier phase tracking process weighted by the probability of the carrier phase estimation errors. The second one consists in introducing the phase estimation errors into the simulation. Both methods are explained next. Nevertheless, the first method can only be applied to signals implementing an interleaver. In fact, the first method assumes that the errors of the received bits are independent, but the PLL introduces correlated errors in time. Therefore, only the signals which break this time correlation between errors through the application of an interleaver can be modeled with the first method. A more detailed justification is given in the method explanation (see next section).

**H.3.1.2.1. Ideal demodulation performance values modification method**

The effect of a non-ideal tracking process is the loss of part of the useful signal power. This loss can be modeled as a multiplying factor L to the data symbol power and it changes for each estimated carrier phase value. In this section, it is assumed that the PLL generates a different constant local carrier phase for each received data symbol.

More specifically, the parameter L represents the instantaneous useful signal power loss due to the non-ideal carrier phase tracking process and can be expressed as:

\[
L = \cos^2(\epsilon_\theta) \quad (H-9)
\]

Where:
- \(\epsilon_\theta\): Carrier phase estimation error.
L: Instantaneous Power Loss

The carrier phase estimation error at the correlator output is due to 4 sources of error as has been said in subsection 3.1.3.2 and shown in equations (3-5) and (3-6). The Allan deviation noise and the oscillator vibrations are considered negligible in front of the two other sources in this dissertation and thus, the carrier phase estimation error has been modeled as a Gaussian variable with an average determined by the signal dynamic stress error and with a variance determined by the thermal noise. Assuming that the higher dynamics of the signal are represented by a constant jerk and that the chosen PLL discriminator is the Q discriminator, the carrier phase estimation error is modeled as Gaussian variable with the following characteristics.

\[
\varepsilon_\theta \rightarrow N\left( m_\varepsilon = 2\pi \frac{T_i^3}{K_3} \cdot \frac{m \cdot g}{\lambda}, \sigma^2_\varepsilon = \frac{B_L}{C / N_0} \right)
\]  

(H-10)

Where:

- \(B_L\): PLL filter Bandwidth
- \(T_i\): Coherent integration Time
- \(m\): Constant number of jerks
- \(g\): gravity acceleration
- \(\lambda\): Wavelength of the carrier signal frequency
- \(T_i\): Integration Time
- \(K_3\): coefficient given by [STEPHENS and THOMAS, 1995], in their description of discrete-update PLL.

Moreover, the carrier phase estimation error samples are not independent among them since they represent the evolution in time of a white Gaussian noise filtered by the PLL close loop function equivalent filter.

From equation (H-9), it is easy to calculate the average loss of power by simply averaging the instantaneous power loss (L) at the correlator output weighted by their probability density function. And from the subtraction of the average loss of power divided by the symbol duration to the nominal \(E_b/N_0\) value, a new average \(E_b/N_0\) value can be calculated. Nevertheless, the new average BER cannot be calculated from the new average \(E_b/N_0\).

The reason is that the \(E_b/N_0\) averaging process is not equivalent to average the BER since there is not always a linear relationship between them. Therefore, the new BER has to be found through the average of the instantaneous BER and not through the average of the instantaneous \(E_b/N_0\) values.

An instantaneous BER value is simply the BER value which would be obtained by a \(E_b/N_0\) with the same value as the instantaneous \(E_b/N_0\) value if the tracking process was considered ideal. Therefore, the ideal tracking process relationship between the BER and the \(E_b/N_0\) is also used to link the instantaneous BER and \(E_b/N_0\) values. Nevertheless, note that this statement is only true when the carrier phase tracking errors at the correlator output are independent. The reason is that the correlation among the carrier phase tracking errors at the correlator output correlates the useful power of the received data symbol at the correlator output. Therefore, the data symbol errors are no longer independent as they were when the relationship between the BER and the \(E_b/N_0\) assuming an ideal carrier phase tracking was calculated. Therefore, in
order to apply this method, the signal needs to implement an interleaver. An interleaver is an element which redistributes the transmitted data symbols into a new order at the transmitter and thus allows breaking the correlation between consecutive samples. The original distribution is recovered at the reception by the desinterleaver. A better definition is given in section 6.2.4.

Finally, an instantaneous $E_b/N_0$ value is calculated by multiplying the nominal $E_b/N_0$ value by the instantaneous loss ($L$). And the probability of having an instantaneous $E_b/N_0$ at a given moment depends on the distribution of $L$. And the distribution of $L$ depends on the distribution of $\varepsilon_0$.

Equation (H-11) expresses the BER obtained for a given nominal $E_b/N_0$ when the tracking process is not assumed ideal.

$$ BER_{real} = \int_{-\infty}^{\infty} BER \left[ \frac{E_b}{N_0}_{ideal} \cdot \cos^2(\varepsilon_\theta) \right] \cdot p(\varepsilon_\theta) \cdot d\varepsilon_\theta $$ (H-11)

Where:

- $BER[x]$: Ideal carrier phase tracking BER obtained at an $E_b/N_0$ value equal to $x$.
- $p(x)$: Probability density function of $x$.

The BER as a function of the $E_b/N_0$ when assuming ideal carrier phase tracking can be recovered from the already published theoretical curves of the signal or channel code, or it can be obtained through simulations as it has been explained in section H.3.1.1.

Introducing the carrier phase estimation error probability, equation (H-11) results into:

$$ BER_{real} = \frac{1}{2\pi\sqrt{\sigma_{\varepsilon_\theta}^2}} \int_{-\infty}^{\infty} \exp \left[ \left( \varepsilon_\theta - m_{\varepsilon_\theta} \right)^2 / (2\sigma_{\varepsilon_\theta}^2) \right] \cdot BER \left[ \frac{E_b}{N_0}_{ideal} \cdot \cos^2(\varepsilon_\theta) \right] \cdot d\varepsilon_\theta $$ (H-12)

And from this equation, the BER as a function of the $E_b/N_0$ when the carrier phase tracking process is affected by thermal noise and a constant jerk can be calculated. Moreover, note that using the ideal conversion between $C/N_0$ and $E_b/N_0_{ideal}$, the result can be expressed as a function of the $C/N_0$.

Previously, it has been commented that the proposed method can only be applied when an interleaver is implemented on the signal since it allows breaking the correlation between consecutive data symbols power introduced by the carrier phase estimation errors at the correlator output. Nevertheless, not all the interleavers can break this correlation. In fact, the redistribution of the data symbols made by the interleaver has to separate two samples of the original distribution of at least a minimum determined time. This time is specified by the PLL filter impulsive response. And this time is the time between the first filter impulse response sample and the sample of which the amplitude is attenuated by a factor of $(1/e)$ with respect to the maximal filter impulse response amplitude. Note that this time is specified by the typical criterion of correlation loss of filtered samples. Moreover, this time is about the inverse filter bandwidth that is in this case of 10Hz. Therefore, the interleaver has to separate two consecutives samples of at least 0.1s.

Therefore, for GPS L1C signal transmitted at a rate of 100 symb/s and with an interleaver of 38 rows (data written) by 46 columns (data read), the time between the reception of two consecutives symbols is 0.38 seconds, which largely fulfills the criterion of correlation loss. For GALILEO E1 OS, the signal is transmitted at a rate of 250 symb/s with an interleaver of 8
rows (data read) and 30 columns (data written), which means 0.12 seconds between the
reception of two consecutive symbols. However, the reception time between two bits
separated by 9 bits in the original data symbol distribution is only 32ms, which means that
they are correlated since this time is smaller than 0.1s. To sum up, the carrier phase estimation
errors at the correlator output are independent among them for GPS L1C signal but they are
not entirely uncorrelated for GALILEO E1 OS signal.

H.3.1.2.2. Noise error sources implementation

The second method consists in implementing the different error sources affecting the signal in
the simulator and in demodulating the resulting signal in order to obtain the BER, the WER
and the EER.

There are two ways of implementing these error sources. The first technique is to implement
the error sources after the PLL processing, which implies the carrier phase estimation error
implementation defined in subsection 3.1.3.2. The second technique is to implement the
original error sources before their PLL processing and thus a PLL has also to be implemented.

H.3.1.2.2.1. Noise error sources implementation after the PLL processing

The simulator has to implement the phase estimation error sources after the PLL processing.
Therefore, the simulator has to generate and directly demodulate the signal described in
equation (3.15), which represents the I channel of the signal at the correlator output, \( r_i[k] \). It is
modeled as:

\[
 r_i[k] = \frac{A}{2} \cdot d[k] \cdot \cos(\epsilon_\theta[k]) + n[k] \quad (H-13)
\]

Where:

- \( d_m[k] \): Navigation data at epoch k
- \( \epsilon_\theta[k] \): Phase estimation error
- \( \epsilon_0[k] \): \( \Theta[k] - \hat{\Theta}[k] \)
- \( n[k] \): Filtered Gaussian noise

Consequently, the only difference with the simulator of an AWGN channel with ideal
tracking process is the phase estimation error \( \epsilon_\theta[k] \) generation. And the carrier phase
estimation error \( \epsilon_\theta[k] \) is equal to the addition of two factors: a constant factor representing the
dynamic stress error bias and a variable factor representing the thermal noise.

For practical purposes the generation of the carrier phase estimation error due to thermal noise
at the PLL output is generated from a white Gaussian noise filtered by the closed loop PLL
transfer function. The white Gaussian noise variance is defined by (3.8); therefore, the closed
PLL transfer function gain is set to 1 in order to conserve the final noise power.
The simulation scheme is the following:

\[ i[k] = \frac{1}{2} A d[k] \cos(\theta_k) + n[k] \]

Where:
- \( A \): useful signal amplitude
- \( d[k] \): Navigation data at epoch \( k \)
- \( n[k] \): discrete white Gaussian noise \( \sim N(0, \sigma_n^2) \)
- \( v[k] \): discrete white Gaussian noise \( \sim N(0, \sigma_v^2) \)
- \( \theta_k \): dynamic stress error bias

Note that \( n[k] \) and \( v[k] \) are generated as specified in the beginning of annex H.3.1.3 with \( A \cdot d_m[k]/2 = 1 \).

The coefficients of the filter representing the closed PLL transfer function are defined in [STEPHENS and THOMAS, 1995]. Nevertheless, these coefficients make that the closed PLL transfer function has a gain different from 1; therefore, in order to keep the desired input noise power, the output noise has to be multiplied by a corrector factor.

### H.3.1.2.2. Noise error sources implementation before the PLL processing

In this case, the simulator has to implement the phase error sources over the signal before applying the PLL. Therefore, the main complication in this method is the PLL implementation since the phase error sources are much easier represented. In fact, the influence of the thermal noise over the tracking process is simply generated by filtered white Gaussian noise addition to the transmitted signal, the same addition used to generate the noise over the demodulation performance (see section H.3.1.3).

Another important decision to make is to decide at which rate the simulator is going to generate the input samples. This rate can go from the signal sampling rate, which depends on the signal bandwidth, to any rate which results from the division of the data symbol rate by an integer number. In any case, each generated input sample has a different carrier phase value. However, the PLL can generate new carrier phase estimations at the same rate that the generated input samples or it can generate new carrier phase estimations at a slower rate. For this last option, the PLL carrier phase estimation is used to remove the carrier phase of more than one input sample.

Additionally, this simulator has to generate the correlation between the received signal PRN code and the generated local replica. The correlation result is well known when the code delay is determined and when the carrier phase estimation error is constant during the length of the
received PRN code. But, in this case, since the carrier phase estimation is not constant, the contribution of the correlation has to be calculated differently. This contribution can be determined by first calculating the partial correlations of the parts of the received signal PRN code which has the same carrier phase, by second multiplying the partial correlations by their carrier phase estimation error, and by, third and last, adding all the weighted partial correlations composing the total correlation between the received PRN code and the generated local replica.

The scheme of the simulator is given below.

\[ L_x = \frac{1}{M} \sum_{k=0}^{M-1} s_x[k] \cdot e^{j(\theta(k)-\hat{\theta}(k))} \]  \hspace{1cm} (H-14)
Where:
- \( s_x[k] \): Partial correlation between the received signal PRN code chips and the generated local replica with a delay between them equal to \( \epsilon \tau[k] \) at epoch \( k \) of the channel \( x \).
- \( M \): Total number of partial correlations composing the complete simulated PRN code.

If the number of chips between two generated input samples is an integer number, a partial correlation can be modeled as:

\[
 s_x[k] = \frac{1}{R} \sum_{r=0}^{R-1} p_x[r] \tag{H-15}
\]

Where:
- \( p_x[r] \): Integration result of the channel \( x \) from the multiplication of the received code chip \( r \) by the generated replica code chip \( r \) having a delay between them equal to \( \epsilon \tau[k] \).
- \( R \): Number of chips of the received PRN code used to make the partial autocorrelation.

Assuming that the RF/IF block allows a received signal with an infinite bandwidth, the term \( p_x[r] \) can be modeled as:

\[
 p_x[r] = \left\{ \begin{array}{ll}
 \frac{1}{1 - 2 \cdot \epsilon \tau[k]} & c_x^r = c_{r-1}^x \\
 1 & c_x^r = -c_{r-1}^x
\end{array} \right. \tag{H-16}
\]

Where:
- \( T_c \): Code chip duration.
- \( c_x^r \): Code chip \( r \) amplitude of channel \( x \) \( \in \{-1, +1\} \).

This model can be adapted to a real RF/IF block by filtering equation (H-16) by the RF/IF implemented filter. This is not done in this dissertation except for acquisition performance analysis in 0.

In this dissertation, the code delay estimation error \( \epsilon \tau[k] \) is assumed to be 0; therefore, \( s_x[k] \) is always equal to 1, and the scheme of the simulator can be simplified as shown below.

![Figure H-4: Simplified PLL simulator scheme for \( \epsilon \tau[k] = 0 \)](image-url)
Where:

$$L = \frac{1}{M} \sum_{k=0}^{M-1} e^{j(\theta(k)-\theta(k))}$$ \hspace{1cm} (H-17)

One important observation made from Figure H-4 is that the complete correlation between the received signal PRN code waveform and the generated local replica does not have to be simulated.

Another remark can be made about the pilot and data channel noises. In fact, both noises originate from the same thermal noise at the antenna output. This antenna output noise is linearly combined with the pilot PRN code to generate the pilot channel noise and is linearly combined with the data PRN code to generate the data channel noise. Therefore, a more realistic simulation should conduct these linear combinations in order to obtain the channel noises. Additionally, the noises are generated as indicated in annex H.3.1.3, where the Gaussian standard deviation is calculated using equation (H-23). In this equation, the simulator specifies the symbol transmission rate as the rate at which the noise samples are generated.

Moreover, note that the previous presented schemes have assumed that the pilot and data channel are in-phase as for GALILEO E1 OS. Nevertheless, this scheme is still valid for a time-multiplexed pilot and data channels, but some smaller changes should be introduced to adapt the schemes to an in-phase-quadrature construction.

Finally, the signal dynamics that affect the signal phase are generated from equation (A-21). Nevertheless, since the PLL implemented in our simulations is of the order 3, and the jerk is considered constant and equal to 1, the model of the incoming signal can be reduced to:

$$\theta(t) = t^3$$ \hspace{1cm} (H-18)

**H.3.1.3. Generation of the additive white or filtered Gaussian noise**

In subsection 3.1.2, the main characteristics of the AWGN channel mathematical model have been defined. However, the relations between all the theoretical formulas and real implementation for simulation purposes have not yet been described.

In the next paragraphs, the generation of a white Gaussian noise or a filtered white Gaussian noise is shown. This generation can be used to create the noise of any signal, where the noise fulfills the previous characteristics. This means that the following proposed method can be used to represent either a white Gaussian noise or a filtered white Gaussian noise at the antenna output, at the correlator input, at the correlator output, etc. The only difference is the value of the parameters needed to generate the noise.

To simulate the AWGN channel mathematical model, we have to be able to generate an additive white Gaussian noise or an additive filtered white Gaussian noise. The power of this noise depends on the $C/N_0$ figure of merit; therefore, the choice made in this Ph.D. is to set the useful signal power to 1, to model the AWG noise as a random Gaussian process and to set the power of this Gaussian process in relation to the desired $C/N_0$ when the useful signal power is equal to 1. This means that the increase/decrease of the received signal power and the increase/decrease of the noise source power, or in other words the $C/N_0$ variation, are achieved by modifying the standard deviation value of the random Gaussian process which models the noise.
A Gaussian noise can be modelled as

\[ n[k] = \sqrt{\sigma} \cdot g[k] \quad \text{(H-19)} \]

\[ P_n = \sigma^2 \quad \text{(H-20)} \]

Where:

- \( n[k] \): Gaussian variable: \( N(0, \sigma^2) \)
- \( g[k] \): Gaussian variable: \( N(0, 1) \)
- \( P_n \): Noise power

Using equations (3-1), (3-3) and (H-20), the random Gaussian variable standard deviation is linked to the \( C/N_0 \) value as it is shown below.

\[ \sigma = \sqrt{\frac{B}{(C/N_0)}} \quad \text{(H-21)} \]

Where:

- \( B \): Channel bandwidth

Finally, assuming that the channel bandwidth is equivalent to the symbol transmission rate (\( R_s \)) as it is the case when a matched filter is used at the reception [PROAKISc, 2001], equation (H-21) is equivalent to:

\[ R_s = B \quad \text{(H-22)} \]

\[ \sigma = \sqrt{\frac{R_s}{(C/N_0)}} \quad \text{(H-23)} \]

The symbol transmission rate is the parameter which allows the generation of the desired noise. For example, if we want a simulator generating one sample for each code chip at the correlator input, the chosen \( R_s \) will be the code chip rate. If we want a simulator generating two samples for each code chip at the correlator input, the chosen \( R_s \) will be twice the code chip rate. And if we want a simulator generating one sample for each data symbol at the correlator output, the chosen \( R_s \) will be the data symbol transmission rate.

Moreover, the conducted simulations can implement either a baseband signal or an equivalent baseband signal to a pass-band signal. In both cases, all the generated noises are based on equation (H-23). More specifically, the only difference between a baseband signal and an equivalent baseband signal to a pass-band signal is that for the latter option the generated noise is complex whereas for the first option it is real. This means that for the latter option, two noises, one for the real component and the other for the imaginary component, are generated. Note that the sum of the noises power has to be equal to \( R_s \cdot N_0 \) as it has been specified in equation (3-1); therefore the standard deviation of the noise of the real component and the standard deviation of the noise of the imaginary component are calculated using equation (H-23) but with a dividing 2 factor inside the square root. For the base-band signal option, the imaginary component is never used during the demodulation process but the noise of the real component is generated as it was done in the equivalent baseband signal to a pass-band signal case:
H. Transmission channels and simulators schemes

\[ \sigma = \sqrt{\frac{R_s}{2 \cdot \left( C / N_0 \right)}} \]  

(H-24)

Finally, in order to obtain a filtered Gaussian noise, as for example to model the influence of the thermal noise over the PLL phase estimation error, the generation is done following the same process as for an AWG noise but implementing a filter with the desired cut-off frequency: the input of the filter is the AWG noise and the output of the filter is the desired filtered Gaussian noise.

H.3.2. Mobile channel simulator

In this subsection, the simulator implemented to represent the transmission of a GNSS signal through a mobile channel is described. Using the simplified version of the second simulator of an AWGN channel (see Figure H-4), the mobile channel simulator scheme is presented below.

Figure H-5: Mobile channel simulator scheme

Where:
- \( \theta[k] \): Incoming signal carrier phase
- \( \hat{\theta}[k] \): PLL signal carrier phase estimation
- \( c_1[k] \): Loo variable at epoch k
- \( n_p[n] \): Pilot channel noise at epoch n.
- \( n_d[n] \): Data channel noise at epoch n.
- \( d[n] \): Navigation data at epoch n.
- \( k \): epoch at chip sampling rate
- \( n \): epoch integration output rate
- \( L[n] \): Correlator output at epoch n

\[ L[n] = \frac{1}{M} \sum_{k=0}^{M-1} e^{j(\theta[k] - \hat{\theta}[k])} \]  

(H-25)

The generation of the \( c_1[k] \) coefficient and the generation of the noise is presented next.

H.3.2.1. Generation of the \( c_1[k] \) coefficient

The \( c_1[k] \) coefficient follows a Loo distribution and thus its generation is accomplished by reproducing this distribution type. Therefore, since a Loo variable is the addition of two
Transmission channels and simulators schemes

H. Transmission channels and simulators schemes

terms, first a variable with a Rayleigh distributed module and a phase uniformly distributed over \([0,2\pi]\), and second a Log-normal distributed variable, the generation of a \(L_0\) variable is achieved by generating these two variables.

\[
c_i[k] = z[k] + w[k]
\]

(H-26)

Where:
- \(z[k]\): Log-normal distributed variable
- \(w[k]\): Variable with a module Rayleigh distributed and a phase uniformly distributed over \([0,2\pi]\)

The generation of a Rayleigh distribution variable, the generation of a log-normal distribution variable and the final \(c_i[k]\) generation scheme are presented in the following subsections.

**H.3.2.1.1. Generation of the variable with a Rayleigh distributed module and a uniform distributed phase**

A variable having a Rayleigh distributed module and uniform over \([0,2\pi]\) distributed phase can be modeled using two independent Gaussian variables \(N(0, \sigma^2)\): one variable representing the real component, and the other variable representing the imaginary component.

\[
w[k] = n_1[k] + jn_2[k]
\]

(H-27)

Where:
- \(n_1[k]\): Normal distribution variable \(N(0, \sigma^2)\)
- \(n_2[k]\): Normal distribution variable \(N(0, \sigma^2)\)

The generation of the Gaussian variables is made by following the method explained in annex H.3.1.3. This method consists in generating a Gaussian variable with a standard deviation equal to 1, and in multiplying this variable by a coefficient calculated from the received useful signal \(C/N_0\) in order to modify the Gaussian variable variance. However, in this case, the generation of the multiplying coefficient depends on the MP definition. This parameter represents the average multipath power with respect to the LOS signal having a power equal to 0dB. And since the LOS signal amplitude before the multiplication of the \(c_i[k]\) coefficient has been set to 1, the simulator is representing the case defined in the definition. Therefore, the standard deviation of the final Gaussian variables, or in other words, the multiplying coefficient, is equal to \(\sigma = \sqrt{\text{MP}/2}\) in order to impose a power of the multipath component equal to MP. The MP parameter, define in section 3.2.2.3.4, is used in its decimal format.

Finally, one last characteristic remains to be simulated. The samples generated from the previous Gaussian variables are independent among them whereas they should be correlated in time since these samples represent the multipath component evolution along the time. In fact, they should follow an evolution with a changing rate bounded by the channel coherence time. Therefore, in order to simulate this evolution, the independent Gaussian variables are filtered by a low-pass filter with a cut-off frequency equal to the Doppler spread. An adequate filter representing the satellite-mobile user communications is the Butterworth filter [BURZIGOTTI et al., 2008].

\[
|H_{\text{Butt}}(f)|^2 = \frac{B}{1 + \left(\frac{f}{f_{c-off}}\right)^{2K}}
\]

(H-28)
H. Transmission channels and simulators schemes

With:
- B: Constant setting the filter gain to 1
- $f_{\text{c-off}}$: cut-off frequency
- $k$: filter order

**H.3.2.1.2. Log-normal distribution variable generation**

The generation of a log-normal distribution variable is based on the generation of a Gaussian distribution variable. In fact, a log-normal variable has the same distribution as a Gaussian variable when a logarithm is applied to the log-normal variable. Therefore, in order to generate a log-normal variable, we are going to generate a Gaussian variable and from it, we are going to generate the log-normal variable applying an exponential to the Gaussian variable.

Therefore, this simulator is going to first generate a Gaussian variable with mean and standard deviation equal to the log-normal $\alpha$ and $\Psi$ parameters. Remember that these parameters define the log-normal variable mean and standard deviation of the desired mobile channel. Second, the simulator is going to divide the normal variable by 20. And third and last, the simulator is going to power by 10 the division result. See equation (H-29).

$$z[k] = 10^\frac{n[k]}{20}$$  \hspace{1cm} (H-29)

With:
- $n[k]$: Gaussian variable $N(\alpha, \Psi^2)$
  - $\alpha$: log-normal mean (dB)
  - $\Psi$: log-normal standard deviation (dB)

Note that the values of the mean and standard deviation used to generate the lognormal variable are expressed in dB instead of in decimal format.

Finally, the generation of the log-normal variable given in equation (H-29) provides independent log-normal samples. Therefore, these independent samples have to be related to the channel mathematical model presented by Perez-Fontan [PEREZ-FONTAN et al, 1998]. The mathematical model define that two independent log-normal samples are found separated by a distance called the channel correlation distance, $L_{\text{Corr}}$; and this distance altogether with the mobile receiver velocity determine the time between two independent log-normal samples. This means that if the simulator wants to generate more input samples between the two independent log-normal samples, the simulator has to correlate the new generated input samples with the previous independent log-normal samples. The reason is that in doing so, the simulator will obtain input samples following the desired log-normal distribution in time where two samples separated the channel correlation distance will still be independent.

In fact, the rate of the input samples generated by the simulator is defined by the multipath component and thus, the simulator needs to generate more input samples between the two independent log-normal samples. Therefore, the log-normal generation is completed by adding an interpolator which generates from two independent log-normal samples, a series of correlated log-normal samples at the desired sampling rate between the two independent log-normal samples.
The typical interpolator block scheme is given below.

![Generic Interpolator Diagram](image)

Note that if the required rate of the input samples is too high to be attained with only one interpolator –filter is too restrictive to be successfully implemented-, the simulator can always implement more than one interpolator.

### H.3.2.1.3. Generation scheme of the $c_1[k]$ coefficient

The final generation scheme of the $c_1[k]$ coefficient is shown in Figure H-7. This scheme shows all the elements presented in the previous two sections.

![Generation of the $c_1[k]$ coefficient Diagram](image)

Finally, note that whereas the log-normal mean and standard deviation are used in dB, the MP parameter is used in its natural form.

### H.3.2.2. AWGN generation

The generation of the AWG noise can be ambiguous because, in the mobile channel case, the received signal power varies. Moreover, the simulated signal is the received equivalent baseband signal which is complex. Therefore, the noise also has to be complex.
The complex noise generation consists in generating two white Gaussian variables as indicated in annex H.3.1.3. One Gaussian variable represents the real noise component and the other represents the imaginary one. And the standard deviation of the Gaussian variables is calculated by using equation (H-24). Finally, the $C/N_0$ used to calculate the standard deviation values is the $C/N_0$ value when it is not taken into account the possible signal attenuation introduced by the mobile channel, the $c_1[k]$ coefficient. In other words, the $C/N_0$ used to calculate the noise sigma is the $C/N_0$ that would be received if the transmission was made through an AWGN channel as has been specified in subsection 3.2.2.4.
Annex I. Code Shift Keying

In this annex, the CSK mapping between the input bits and the transmitted CSK symbol is presented and the calculation of the variance of the number of errors of a packet defined by a given CSK source packets mapping is described.

I.1. CSK mapping between the input bits and the transmitted symbol

In this section, the mapping between the k bits transmitted into a single CSK symbol and the CSK symbol selected to transmit the bits is presented. This mapping is different from the orthogonal CSK or M-ary orthogonal signaling to the bi-orthogonal CSK or M-ary bi-orthogonal signaling.

I.1.1. Orthogonal CSK mapping

The orthogonal CSK signaling method calls the symbol represented by the smallest circular shift of the fundamental code—even if this shift is equal to 0 chips-, the first circularly shifted version of the fundamental code or symbol 0. The transmission of this symbol would be reflected only at the $y_0$ component of Y vector, if no noise was introduced during the signal transmission through the channel. The other components of the vector should be 0. However, since the channel transmission introduces thermal noise to the received signal, the value of the $y_0$ component is corrupted and the values of the other Y vector components are not 0. The symbol represented by the second smallest circular shift of the fundamental code is called the symbol 1 and its transmission through a noiseless channel would be reflected only on the $y_1$ component of the Y vector. The remaining association between circular shifts of the fundamental code, symbols and components of the Y vector is done as explained with the two first symbols. Therefore, the reception of the CSK symbol represented by the largest circular shift of the fundamental code should be inspected at the last component of the Y vector, the $y_{M-1}$ component.

The mapping between the input information bits and the CSK symbols is a natural mapping, with the first bit fed to the demodulator considered as the bit with the smallest weight, and the last bit fed to the demodulator considered as the bit with the biggest weight. The input information bits are denoted as:

$$B^k = \{b_{k-1}, b_{k-2}, \ldots, b_0\}$$

(I-1)

Where:

- $B^k$: Set of $k$ input bits for a orthogonal CSK symbol
- $b_x$: Value of the $x$th bit of the set $B^k$

For example, if the bits values are $B = \{b_2=1, b_1=0, b_0=1\}$, the circularly shifted version of the fundamental code having the sixth smallest shift —symbol 5— is transmitted. Its reception is observed at the $y_3$ component of the Y vector.
### I.1.2. Bi-orthogonal CSK mapping

The mapping of the bi-orthogonal CSK signaling method between the input information bits and the bi-orthogonal CSK symbols is very similar to the orthogonal CSK mapping. The only difference is that for a bi-orthogonal mapping one of the input bits of the set of \( k \) bits determines the polarity—or sign—of the CSK symbol to transmit. The exact mapping process is given below.

The first \( k-1 \) bits fed to the modulator are used to select the symbol to transmit. The selection is made as if the \( k-1 \) bits have to be mapped for an orthogonal CSK signal. Therefore, the component of the \( Y \) vector in which the reception of a bi-orthogonal CSK symbol is observed is determined by the first \( k-1 \) bits fed to the modulator. The \( k^{th} \) bit fed to the modulator is used to select the polarity of the symbol. The mathematical notation for a bi-orthogonal CSK mapping is:

\[
R^k = \{ \beta, b_{k-2}, b_{k-2}, \ldots, b_0 \} \quad (I-2)
\]

\[
R^k = \{ \beta, B^{k-1} \} \quad (I-3)
\]

Where:

- \( R^k \): Set of \( k \) input bits for a bi-orthogonal CSK symbol
- \( \beta \): Value of the bit controlling the polarity of the bi-orthogonal CSK symbol
- \( B^{k-1} \): Set of \( k-1 \) input bits for an orthogonal CSK symbol
- \( b_x \): Value of the \( x^{th} \) bit of the set \( B^{k-1} \)

Note that a \( M \)-ary bi-orthogonal signaling method is equivalent to a \( M/2 \)-ary orthogonal signaling method where the two polarities—or signs—of the \( M/2 \) different symbols are used.

### I.2. Calculation of the variance of the number of bit errors of a packet defined by a CSK source packet mapping

The calculation of the variance of the number of errors of a packet defined by a given CSK source packet mapping is achieved by applying the typical formula of the variance of a random variable. This means that we need to find the probability of a packet having \( X \) errors after its transmission through the channel, where \( X \) goes from 0 to the packet size.

The only difficulty of this proposed calculation of the variance is that the theoretical formula of the probability of a packet having \( X \) errors after its transmission through a channel changes its mathematical expression depending on the used CSK source packet mapping. Therefore, the solution employed in this dissertation consists in encoding a program which provides the desired results for each given CSK source packet mapping rather than to determine a specific formula for each CSK source packet mapping. Nevertheless, note that this program does not consist of channel transmission simulations but it consists of applied theoretical formulas.

The probability of a packet having \( X \) errors after its transmission through a channel when the packet is transmitted following a given CSK source packet mapping depends on two main parameters. The first parameter is the number of bits mapped by each CSK symbol which belong to the same packet. In this case, in order to simplify the analysis, it has been assumed that the number of bits mapped by a CSK symbol which belong to the same packet is the same for all the packets. For example, if a CSK symbol maps 6 bits, the 4 different configurations are: 6 packets of 1 bit, 3 packets of 2 bits, 2 packets of 3 bits, and 1 packet of 6
bits. The second parameter is the packet size. The number of different probabilities or, in other words, the number of errors $X$ which can have a packet depends on the packet size. And the larger the packet is, the more number of errors has and the larger the variance is.

The method presented in the next subsections follows two steps. First, the probability of having $n$ erroneous bits out of $x$ bits belonging to the same packet when a CSK symbol mapping $K$ bits, $x \leq K$, is transmitted with a CSK symbol probability of error equal to $P_M$ is calculated. Second, the previous probabilities for different values of $n$ which are associated to a single CSK symbol are propagated to the total number of CSK symbols necessary to transmit a complete packet. Note that the parameter $x$ defines the different CSK source packet mappings.

### I.2.1. Probability of having $n$ erroneous bits among $x$ bits belonging to the same packet when the transmitted CSK symbol maps $K$ bits

The first step of the calculation of the variance consists in calculating the probability of having $n$ erroneous bits out of $x$ bits belonging to the same packet when the transmitted CSK symbol maps $K$ bits and has a probability of error equal to $P_M$.

The number of different probabilities to calculate in this first step depends on the number of bits, $x$, belonging to each packet. For example, the number of probabilities to calculate is 7, from 0 erroneous bits to 6 erroneous bits, when a CSK symbol maps 6 bits and all the bits belong to the same packet; and the number is 4, from 0 erroneous bits to 3 erroneous bits, when a CSK symbol maps 6 bits with 3 bits belonging to a packet and 3 bits belonging to another one. In this last case, the probabilities calculated are valid for both packets.

The method used to calculate the probabilities consists in first finding the probability that an erroneous estimated CSK symbol mapping $K$ bits results into the erroneous determination of the values of $n$ bits, and second, in weighting the previous probability with the probability of demodulating incorrectly a CSK symbol.

The probability is found by dividing the number of cases where the CSK symbol has $n$ errors by the total number of possible cases. A case is defined as a pattern of bit errors generated by the erroneous estimation of the received CSK symbol. Therefore, since a CSK symbol mapping $K$ bits can have from 0 to $K$ erroneous bits when the CSK symbol has been estimated incorrectly, the total number of cases is $2^K-1$. The number of cases where the erroneous CSK symbol results into $n$ erroneous bits is calculated by using a combinatorial number.

Nevertheless, the previous method has to be slightly modified in order to take into account that only $x$ bits out of $K$ bits mapped by the CSK symbol belong to the same packet. Therefore, the first analyzed CSK source packet mapping is the mapping where all the bits belong to the same packet since it is the simplest calculation and its calculation method is the method defined previously. This CSK source packet mapping is called the basic CSK source packet mapping from now on. Moreover, the results of the basic CSK source packet mapping are used to simplify the calculations of remaining CSK source packet mappings.

Assuming a CSK symbol carrying $K$ bits, the probabilities of the basic CSK source packet mapping of having $n$ erroneous bits when the CSK symbol is erroneously demodulated are:

$$p(0|K_k) = 1 - P_M$$  \hspace{1cm} (I-4)
I. Code Shift Keying

\[ p(n|K_K) = P_M \cdot \frac{\binom{K}{n}}{2^K - 1} \quad K \geq n \geq 1 \]  \hspace{1cm} (I-5)

Where:

- \( p(n|Y_y) \): Probability of having \( n \) bit errors belonging to the same packet when transmitting a CSK symbol mapping \( Y \) bits out of which \( y \) bits belong to this packet.

- \( P_M \): Probability of error of a CSK symbol

Once the probabilities of the basic CSK source packet mapping have been found, the probabilities for CSK source packet mappings defining packets which share bits mapped by the same CSK symbol are calculated. The process consists of two steps.

First, for each different number of bit errors resulting from the erroneous estimate of the CSK symbol and its associated probability -the probability of the main CSK source packet mapping-, the probabilities of the different CSK source packet mappings where the bits mapped by a CSK symbol belong to more than one packet are calculated. More specifically, each probability is calculated by dividing the number of patterns of \( m \) bits errors out of which \( n \) erroneous bits belong to the analyzed packet by the total number of patterns of \( m \) bit errors. For example, assuming that a CSK symbol carries 6 bits where 3 bits belong to one packet and the 3 other bits belong to another packet, some representative particular cases are: if the erroneous estimate of the CSK symbol results in to 1 bit error out of the \( K \) bits mapped by the CSK symbol, the particular cases are that the packet of 3 bits has either 0 or 1 errors; however, when the erroneous estimate of the CSK symbol results in to 4 bit errors, the particular cases are that the packet of 3 bits has from 1 to 3 errors. The mathematical expressions are:

\[ p(n,m|K_{x}) = \begin{cases} 
\frac{x}{\binom{x}{n}} \cdot \binom{K-x}{m-n} & m \geq n \geq (m-x) \\
0 & \text{otherwise}
\end{cases} \]  \hspace{1cm} (I-6)

Where:

- \( p(n,m|K_{x}) \): probability of having \( m \) erroneous bits out of which \( n \) bits belong to the same packet when the CSK symbol mapping \( K \) bits out of which \( x \) bits belong to the same packet, \( m \leq x \), has been erroneously estimated.

The second step consists of adding the probabilities of the particular cases having the same number \( n \) of bit errors but found from different numbers of \( m \) bit errors. The \( m \) bit errors are the result of the erroneous estimate of the CSK symbol.

\[ p(n|K_{x}) = (1 - P_M) \cdot \delta(n) + \sum_{m=1}^{K} p(n,m|K_{x}) \]  \hspace{1cm} (I-7)

Where:

- \( \delta(n) \): Dirac delta function

Once the calculation has been explained, the mathematical expression is simplified and adapted to include the basic CSK source packet mapping. The general expression is:
I. Code Shift Keying

\[ p(n|K_n) = (1 - P_m) \cdot \delta(n) + P_m \sum_{w=\delta(n)}^{K^x} \binom{K-x}{w} n \leq x \]  

(I-8)

Finally, this expression is valid and exact for an orthogonal CSK or MOS signaling method. Nevertheless, this expression can also be used for a bi-orthogonal CSK because, in this case, the expression is a very accurate approximation.

I.2.2. Calculation of the probability error of the bits of a packet from the probability of error of the bits mapped by a CSK symbol

The second step of the calculation of the variance of the number of errors of the packets defined by a given CSK source packets mapping consists in calculating the probability of having a given number of bit errors inside a packet from the probability of having \( n \) erroneous bits out of \( x \) bits belonging to the same packet when the transmitted CSK symbols maps \( K \) bits.

The bits mapped by a single CSK symbol which belongs to one or more packets do not have to be the only bits belonging to these packets. In fact, since the amount of bits which can be mapped by a single CSK symbol is rather small, the most common configuration is to group several CSK symbols in order to generate a packet. This means that the probabilities calculated in the previous section are not sufficient to calculate the variance of the number of bit errors of the packets of a given CSK source packet mapping. More specifically, if the packet has a size of \( P \) bits, the probabilities necessary to calculate the variance go from the probability of having 0 bit errors during the complete packet transmission to the probability of having \( P \) bit errors during the complete packet transmission. And all these probabilities can be calculated from the probabilities calculated in the previous section by propagating them. The definition of a generic propagation process is given below.

A generic propagation process consists in knowing the bit error probabilities of an element of \( L \) bits, denoted as the element-to-propagate, in order to calculate which will be the bit error probabilities of this element if its size is increased to \( T \). The new element of \( T \) bits of size is called the propagated-element, where \( T=L+L' \) and \( L' \) is the size of the element used to extend the original element of \( L \) bits of size. The element of \( L' \) bits of size is called the propagation-element. Therefore, applying the propagation process, the probability of having \( n \) erroneous bits belonging to the same packet when the bits of the packet are carried by one CSK symbol which maps \( L \) bits can be extended to the probability of having \( n' \) erroneous bits belonging to the same packet when the bits of the packet are carried by two CSK symbols where each one maps \( L \) bits. Moreover, note that the element-to-propagate and the propagation-element are not limited to a single CSK symbol but they can be any element of any possible size.

The propagation process is achieved by making a combination of all the different bit error probabilities of the element-to-propagated with all the different bit error probabilities of the propagation-element. More specifically, each bit error probability of the element-to-propagate is multiplied by each bit error probability of the propagation-element. The result is a partial bit error probability of the propagated-element where the number of bit errors associated to this probability is calculated by adding the number of bit errors of this bit error probability of the element-to-propagate to the number of bit errors of this bit error probability of the propagation-element. Finally, once all the multiplications have been made, the partial bit error probabilities of the propagated-element having the same number of bit errors are added together in order to obtain the total bit error probabilities of the propagated-element.
The algorithm implementing the propagation process is given below:

- **Element-to-propagate**: \{Size: L, \( P^e = (P^e_0, \ldots, P^e_L) \)\}
  - \( P^e_x \): Probability of having \( x \) bit errors of the element-to-propagate

- **Propagation-element**: \{Size: \( L' \), \( P^p = (P^p_0, \ldots, P^p_{L'}) \)\}
  - \( P^p_x \): Probability of having \( x \) bit errors of the propagation-element

- **Propagated-element**: \{Size: \( T \), \( P^{ep} = (P^{ep}_0, \ldots, P^{ep}_T) \)\}
  - \( P^{ep}_x \): Probability of having \( x \) bit errors of the propagated-element

- Initializing propagated-element probabilities to 0.

- **For** \( i = 0 \) to \( L \)
  - **For** \( j = 0 \) to \( L' \)
    - \( P^{(i+j)}_{(i+j)}^{ep} = P^{(i+j)}_{(i+j)}^{ep} + P^e_i \cdot P^p_j \)