1 GNSS Signals and Receivers

José A. López-Salcedo, Ignacio Fernández-Hernández, M. Zahidul H. Bhuiyan, Elena Simona Lohan and Kai Borre

1.1 Introduction

This chapter introduces the fundamentals of signals and receivers used in Global Navigation Satellite Systems (GNSS). We will start with a general overview of the existing GNSS signals and their main characteristics in Section 1.2, in order to set the grounds for the subsequent sections and chapters. We will continue with Section 1.3, presenting in more detail the structure of GNSS signals and the theoretical models that will be used throughout the book. Section 1.4 will review the GNSS link budget to better understand how GNSS signals are received in the way they are. Then, Section 1.5 will briefly introduce the architecture of GNSS receivers, while their acquisition and tracking modules will be discussed in more detail in Sections 1.6 and 1.7, respectively. Sections 1.8 and 1.9 will cover the navigation message and pseudorange errors, respectively, and finally, Section 1.10 will explain how to calculate a position fix.

1.2 Overview of GNSS Signals

The main purpose of GNSS signals is to provide an accurate ranging measurement to the receiver, while at the same time, to provide the necessary data for the receiver to compute its position. Therefore, well-designed GNSS signals must allow ranging measurements as accurate as possible, as well as data reception without errors, whenever possible. To do so, several constraints and considerations must be borne in mind, as discussed next.

First of all, for both accurate ranging and good data reception, the carrier frequency must be chosen so the signal can propagate well through the atmosphere and be received in all possible weather and visibility conditions. The L band, between 1 and 2 GHz, with a wavelength between 30 and 15 cm is a good candidate for this purpose. In fact, current GNSS signals are transmitted in this band. In their journey from the satellite to the receiver, signals are delayed by the ionosphere, a layer of the atmosphere at around some hundreds of kilometers above the Earth's surface, and then by the troposphere, a lower and thinner layer that determines our weather. Unlike the tropospheric delay, most of the ionospheric delay is difficult to model, but it is related to the carrier frequency of the signal, so having two synchronized frequencies transmitted from the same satellite allows removing it and having more accurate ranging measurements. Transmitting signals in several frequencies also allows more services, increased resilience against interference and precise point positioning (PPP) improvements. Table 1.1 lists the features of GNSS signals used in this book and in the software accompanying it: the American Global Positioning System (GPS), the Russian GLONASS, the European Galileo, the Chinese BeiDou and the Indian Navigation Indian Constellation (NavIC). The first four are global systems, while the latter is a regional system.[1]

In addition to choosing an adequate carrier frequency, the signals need to be powerful and have a broad-enough frequency bandwidth to provide accurate ranging. Unfortunately, generating a powerful signal increases the satellite weight, making it more costly to put it in orbit. GNSS signals are transmitted at a power of some tens of watt (W) and received on Earth at an extremely low power, around 10^{-16} W, which makes the performance of GNSS even more remarkable. Apart from signal power restrictions, the frequency bandwidth is also constrained by satellite and receiver technology. Modern signals such as Multiplexed Binary Offset Carrier (MBOC) used by GPS and Galileo, or Alternative Binary Offset Carrier (AltBOC) used by Galileo and BeiDou, have been designed so that receivers can process a narrower part of it, at a lower complexity, or in its entirety, for a better ranging accuracy. Also, the signal power and bandwidth are constrained by international organizations such as the International Telecommunications Union, which allocates frequencies to services and guarantees that services do not interfere with each other.

For accurate ranging, the signals transmitted by all satellites need also to be synchronized with a common time reference. The satellites have very precise atomic clocks onboard, even though they are not perfect. For a meter-level accuracy, we need to measure time at the level of a few nanoseconds (a nanosecond is 10^{-9} s), since an electromagnetic signal travels 30 cm in 1 ns at the speed of light. The time offset between the individual clocks in the satellites and the common time reference is determined by the continuous tracking of the satellites by stations in the GNSS ground segment. The GNSS ground segment also tracks where the satellites are and sends this information back to the satellites through uplink stations. The satellites, in turn, embed this time offset and their position information into the navigation data that is conveyed by the transmitted GNSS signal. This ensemble of data is called the satellite *ephemeris* (*ephemerides* in plural).[2] The need for transmitting data along with the provision of accurate ranging has driven the design of GNSS signals. However, in order to achieve these same goals, designers prioritized different aspects. Some signals were designed to carry more data than just the satellite ephemerides, other signals were designed to support military signals, some were designed decades later than others, assuming better receiver capabilities, some have been designed to minimize interference with legacy signals, and so on.

In summary, GNSS signals are composed of at least three elements: (i) the data symbols conveying the bits of the navigation message, which contains the satellite

[1] Note that the book also indirectly addresses the Japanese Quasi-Zenith Satellite System (QZSS), as it uses GPS L1 C/A-like signals.

[2] Before GPS, this term referred to the positions of celestial bodies used by navigators.

Table 1.1 GNSS signals treated in this book and in the accompanying software.

Signal	L1 C/A	L1OF	E1-B	E1-C	B1I	L5I	L5Q	L5
System	GPS	GLONASS	Galileo	Galileo	BeiDou	GPS	GPS	NavIC
Service	SPS	SPS	OS	OS	OS	SPS	SPS	SPS
Carrier Freq. [MHz]	1575.42	1598.0625–1605.375	1575.42	1575.42	1561.098	1176.45	1176.45	1176.45
Polarization	RHCP	RHCP	RHCP	RHCP	RHCP	RHCP	RHCP	RHCP
Channel Access	CDMA	FDMA	CDMA	CDMA	CDMA	CDMA	CDMA	CDMA
Modulation	BPSK (1)	BPSK (0.5)	MBOC (6,1,1/11)	MBOC (6,1,1/11)	BPSK (2)	BPSK (10)	BPSK (10)	BPSK (1)
Component	Q	Q	I	I	I	I	Q	I
Chip Rate [Mcps]	1.023	0.511	1.023	1.023	2.046	10.23	10.23	1.023
Code Len [chips]	1,023	511	4,092	4,092	2,046	10,230	10,230	1,023
Code Len [ms]	1	1	4	4	1	1	1	1
Code Family	Gold	M-sequence	Memory codes	Memory codes	Gold	2 M-seq	2 M-seq	Gold
Symbol/Bit Rate [sps/bps]	50/50	50/50	250/125	–	50/≈37	100/50	–	50/25
Data Encoding	None	None	FEC 1/2, *c*:7; interleaving	–	BCH (15,11,1), interleaving	FEC 1/2, *c*:7	–	FEC 1/2; CL:7; interleaving
Nav Ephemeris	Keplerian	P, V, A	Keplerian	–	Keplerian	Keplerian	–	Keplerian
Nav Iono model	Klobuchar	None	NeQuick	–	Klobuchar	Klobuchar	–	Grid-based; GIVE / GIVD
Time Reference	GPST	UTC	GST	–	BDT	GST	–	IST
Geodetic System	WGS84	PZ-90	GTRF	–	BDCS	WGS84	–	WGS84

ephemerides; (ii) the spreading or pseudo-random noise (PRN) code, which facilitates the distance measurement between the satellite and the receiver and (iii) the carrier wave on which the former two are modulated.[3] The configuration of these three elements for different GNSS signals is shown in Table 1.1. For instance, data symbols are transmitted at a rate of 50 sps in GPS L1 C/A, while for Gailleo E1-B, they are transmitted at 250 sps. Their spreading code length is also different, 1,023 chips for GPS L1 C/A and 4,092 chips for Galileo E1-B. However, both signals share the same carrier frequency at 1575.42 MHz. The impact of using these values will be discussed when specifically addressing each system.

It is interesting to note that all satellites of a given GNSS system transmit similar signals, but their spreading codes are unique and thus each satellite can be univocally identified. Furthermore, since the spreading codes are orthogonal sequences, all satellites can transmit messages using the same carrier frequency and hence simultaneously access the medium without interfering with each other. This leads to a Code Division Multiple Access (CDMA) scheme, as indicated in Table 1.1, for most of the systems. If each of the satellites of the system used a different carrier frequency, we would then have a Frequency Division Multiple Access (FDMA) scheme, as it is the case with GLONASS.

Some of the signals listed in Table 1.1 contain components without navigation data bits, and therefore, their data rate is omitted in the corresponding cell. For instance, this is the case of the E1-C component of the Galileo E1 signal. Unmodulated or *pilot* signal components constitute a new feature of modernized GPS and Galileo signals. They are transmitted with the data components, sometimes orthogonally, and they have their own set of spreading codes including a primary and a secondary code on top of the former. The advantage of this new signal component comes from allocating the two main properties of GNSS signals, i.e., transmitting navigation data and providing ranging information, into two separate channels. The data channel conveys the data for locating the satellites in the constellation. However, this information is encoded in binary $-1, +1$ symbols, whose sign transitions reduce the energy that the receiver can accumulate to obtain a ranging measurement. On the contrary, the pilot channel does not contain unexpected symbol transitions and can be integrated over longer periods, which can drastically enhance the signal-to-noise ratio (SNR) of the received signals.

1.3 Structure of GNSS Signals

1.3.1 Signal Modulation

In Section 1.2, it has been introduced that GNSS signals are composed of three main constituent elements: the navigation data bits, the spreading code and the carrier. The data bits contain binary information that must be converted into signal levels to allow

[3] Modulation is the process that allows a signal to be successfully sent through a propagation medium. It will be discussed in Section 1.3.1.

their transmission over the propagation medium. Binary Phase Shift Keying (BPSK) modulation is the simplest way to do so, whereby each bit is converted into a binary symbol with amplitude $\{-1, +1\}$ that modulates a carrier. But before this, the data symbols are multiplied by a spreading code that is unique to each satellite and thus allows identifying each satellite signal at the receiver. The spreading code has a much higher rate than the data symbols, which means that its power content is spread across a much wider bandwidth. The same effect appears when multiplying the data symbols with the spreading code. The result is a signal with a much wider bandwidth than the original one. This effect can be observed in the upper plots of Figure 1.1, which show the frequency representation of the signals at each point of a simplified transmission chain. As can be seen, the spectrum of the signal after multiplication by the spreading code is much wider than that of the original signal.

The process of spreading the power of a signal across a wider range of frequencies is known as *spread spectrum* (SS) modulation. When such spreading is done by multiplying a low-rate signal with a high-rate one, the technique is known as direct-sequence spread spectrum (DS-SS). This is the case in GNSS where the original signal (either containing data or pilots) is multiplied by the spreading code. The ratio between the rate of the spreading code and the rate of the original signal is called the *processing gain*, and it typically runs from 10 to 60 dB. At the GNSS receiver, the original signal can be recovered back by multiplying the received spread signal with the same spreading code signal used at the transmitter. This technique dates back to the 1980s, and it is popular in applications involving radio links in hostile environments. Many radio frequency interference (RFI) signals can easily be rejected because they do not

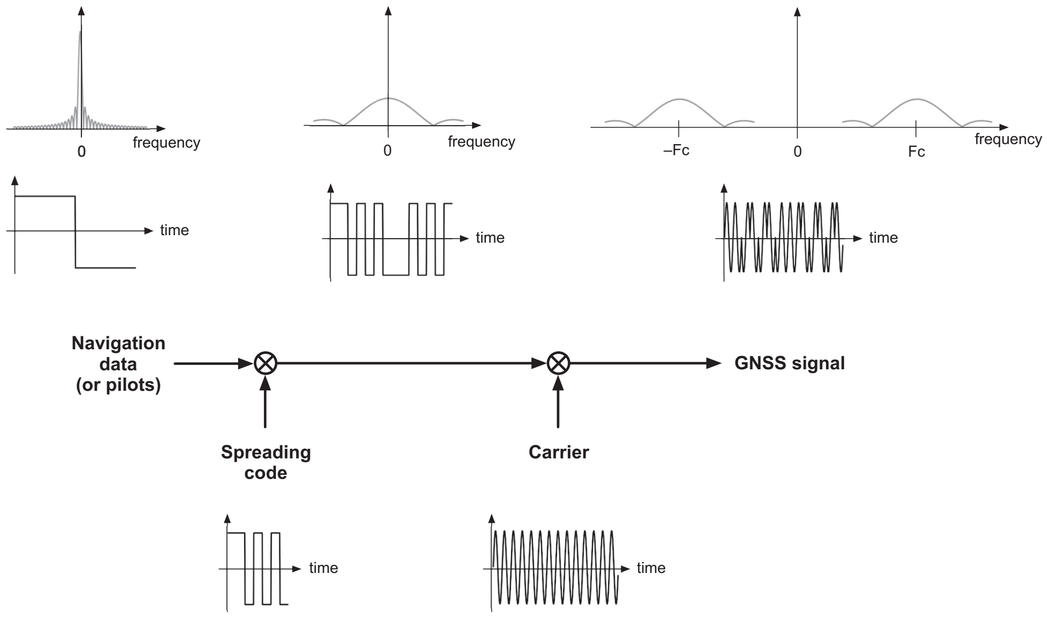


Figure 1.1 Illustration of how GNSS signals are generated, indicating the time and frequency representation of the resulting signal at each point of a simplified transmission chain.

contain the spreading code in their transmitted signal. Therefore, interference is actually spread at the receiver output because the interference signal is only multiplied once by the spreading signal. This feature is the real beauty of SS technology. Only the desired signal, which was generated using the same spreading code, will be seen at the receiver when the despreading operation is applied.

BPSK modulations are generally denoted in the context of GNSS as BPSK(n), where n stands for the rate of the spreading code in multiples of 1.023 MHz.[4] For example, GPS L1 C/A is a BPSK(1) signal. However, most new GNSS signals are modulated in *subcarriers*, which are nothing but a square signal with a rate on the order of the spreading code, or higher, that multiplies the input signal.

Subcarriers were first publicly proposed for GNSS in [1], and they are widely used nowadays in most modernized GNSS signals under the name of BOC signals. They were introduced in order to move the spectral content of the signal away from the carrier frequency, following the well-known modulation principle. By doing so, new signals can share the same carrier frequency as already used by the existing ones, while reducing the spectral overlap. An example comparing the spectrum of BPSK(1) with that of a subcarrier-modulated signal such as BOC(1,1) used in Galileo E1-B is shown in Figure 1.2. As can be seen, both the BPSK(1) and the BOC(1,1) spectra share the same central frequency. However, the spectrum of the BOC(1,1) signal is shifted to both sides of the central frequency so that the overlap with the BPSK(1) is significantly reduced.

The overall transmission and reception scheme of a GNSS signal is therefore the one schematically shown in Figure 1.3, including the possibility that subcarrier

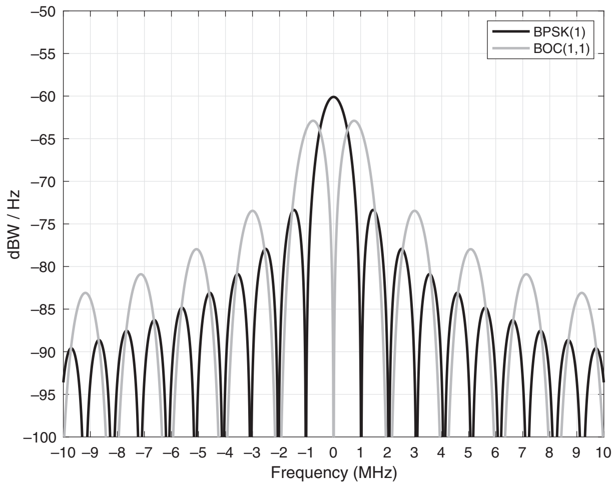


Figure 1.2 Comparison between the spectrum of the BPSK(1) signal used in GPS L1 C/A and that of a subcarrier-modulated signal such as the BOC(1,1) used in Galileo E1-B. The center frequency $f = 0$ corresponds to 1575.42 MHz once the signal is modulated by the carrier.

[4] The rate of the spreading code is also known as the *chip rate* because the spreading code is a sequence of binary elements that are referred to as *chips*.

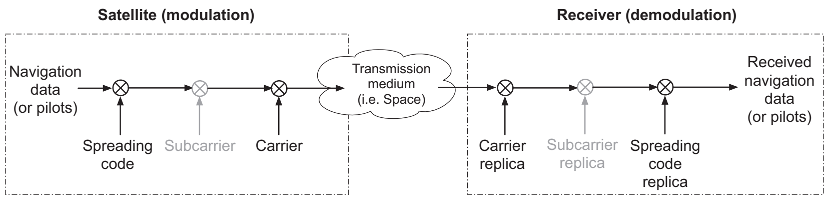


Figure 1.3 Fundamental components in GNSS signal modulation and demodulation.

modulation is present. As previously mentioned, the input signal is composed of data symbols or pilots, depending on the GNSS signal. This low-rate input signal is multiplied by the spreading code so that the resulting spectrum becomes spread in the frequency domain. This operation protects the symbols/pilots against harmful interference, makes the resulting signal exhibit low probability of interception (LPI), allows different satellites to be simultaneously transmitted in the same frequency band and, most importantly, facilitates the use of these signals for ranging. The DS-SS resulting signal is then multiplied by the subcarrier (when present, since not all GNSS signals are subcarrier-modulated) and finally multiplied by the carrier that places the resulting signal at the L band. Once transmitted through the propagation medium, the signal arrives at the receiver and the opposite operations need to be done in order to unbox the data symbols and to obtain the ranging information that is required to position the user.

Bandpass Signal Representation

The modulated signal transmitted by the GNSS satellites is a so-called *bandpass* signal because its frequency representation, or spectral content, is concentrated on a small neighborhood of a high frequency. For example, the BPSK(1) spectrum shown in Figure 1.2 has most of its power concentrated on its main lobe, which spans for approximately 2 MHz bandwidth. This signal is then modulated on a carrier at 1575.42 MHz for the GPS L1 C/A signal, so the spectrum becomes centered at that frequency. This is a clear example of a bandpass signal because most of the power is concentrated in just a 2 MHz bandwidth around a frequency of 1575.42 MHz. Bandpass signals are needed whenever radio transmission is taking place. This is because the size of the antennas needed for transmission and reception is inversely proportional to the central frequency of the signal, so the larger the carrier frequency, the smaller the required antenna. In the communications domain, it is often claimed that the carrier frequency bears no information at all and that is why it is removed at the receiver once its purpose of propagating the radio wave through the medium has been achieved. This is not precisely true in the GNSS domain because part of this carrier, actually, the carrier phase, does bear information on the distance from the satellite to the user's position. So keeping track of the carrier phase is needed at a GNSS receiver, even though this can be done once the carrier frequency has been removed anyway.

In its simplest form, a bandpass signal $s(t)$ can be expressed as follows:

$$s(t) = \sqrt{2P}x(t)\cos(2\pi F_c t), \tag{1.1}$$

where P is the bandpass or radio frequency (RF) transmitted power[5] and $x(t)$ is a unit power and real-valued signal. In our case, $x(t)$ is a train of $\{-1, +1\}$ amplitudes, including the navigation data or pilot symbols, the spreading code and eventually the binary subcarrier as per Figure 1.3. The bandpass nature of the signal s(t) in Eq. (1.1) is obtained after multiplying our signal $x(t)$ with the cosine term at some carrier frequency F_c.

The expression in Eq. (1.1) is the simplest representation of a bandpass signal, but another real-valued data-modulated or pilot signal can be, and is usually, transmitted at the same time using a sine term. This leads to a more general signal model whereby the bandpass signal results in[6]

$$s(t) = \sqrt{2P_I}x_I(t)\cos(2\pi F_c t) - \sqrt{2P_Q}x_Q(t)\sin(2\pi F_c t), \tag{1.2}$$

which contains two signal components, namely the *in-phase* component $x_I(t)$ and the *quadrature* component $x_Q(t)$, whose bandpass or RF transmitted power is P_I and P_Q, respectively. Note that the term *quadrature* refers to the fact that this component is placed at 90°, that is, in quadraphase, with respect to the in-phase one. It is also interesting to note that $x_I(t)$ and $x_Q(t)$ are two independent real-valued signals that are transmitted at the same carrier frequency. This is possible in virtue of the orthogonality between the cosine and the sine functions, and it is exploited in GNSS, as well as in many other wireless systems, for transmitting two different real-valued signals over a single carrier frequency, thus increasing the spectral efficiency. Being part of the same carrier, these signals are referred to as the signal *components*, while the whole ensemble is just referred to as the GNSS signal. GNSS signals use the in-phase and quadrature components in various ways. Here are some examples based on the signals processed by the software accompanying this book:

(1) The GPS L1 legacy signal conveys data-modulated symbols in both signal components $x_I(t)$ and $x_Q(t)$. $x_I(t)$ is used by the military P(Y) code, out of the scope of this book, and $x_Q(t)$ is the open GPS L1 C/A component that we all use.
(2) GPS L5 includes the two components L5I and L5Q carried by the $x_I(t)$ and $x_Q(t)$ in our model, respectively. L5I includes data-modulated symbols, and L5Q modulates the *pilot* component, used to improve tracking performance. Both components are conceived to be used by civil receivers, for a better performance.
(3) Galileo E1 includes two components in $x_I(t)$: The E1-B, with the data, and the E1-C, with the pilot, which are subtracted from one another. In $x_Q(t)$, the signal modulates the E1-A component, aimed for the Public Regulated Service, encrypted and out of the scope of this book.

[5] For a GNSS bandpass signal with amplitude A, its power becomes $P = \frac{1}{T}\int_0^T s^2(t)dt = \frac{1}{T}\int_0^T A^2x^2(t)\cos^2(2\pi F_c t)dt = \frac{A^2}{2}$, for $T \gg \frac{1}{F_c}$. So we need $A = \sqrt{2P}$ for the bandpass power to be P, as stated in the text.

[6] In some GNSS signals, the two carrier-modulated signal components are explicitly added with a + sign. This involves that the in-phase and quadrature components are sign-reversed one with each other when down converted to the baseband, as it is the case in GPS L1 [2, Section 3.3.1.5.1].

Note that here we are considering two BPSK signal components. If the cosine and sine are processed as a single signal, that is, the signal is processed by looking at the cosine and sine peaks altogether, the signal is defined as QPSK (Quadrature Phase Shift Keying). Higher order PSK signals are possible and, in fact, used by GNSS. For example, the Galileo constant envelope E5 AltBOC signal is processed as an 8-PSK signal [3].

Baseband Signal Representation

The information content of the bandpass signal in Eq. (1.2) lies in the two signal components $x_{\mathrm{I}}(t)$ and $x_{\mathrm{Q}}(t)$. For this reason, it is often convenient to focus on these two signal components only and to remove the cosine and the sine waves that are merely present to make the propagation through the medium possible. If the carrier is perfectly removed, and thus the cosine and sine waves, the resulting signal exhibits two key properties. The first one is that the spectral content is placed back into the base of the frequency axis, in the neighborhood of the zero frequency. So we call the resulting signal a *baseband* signal, denoted herein as $b_s(t)$. The second property is that the baseband signal is composed of two real-valued components, $x_{\mathrm{I}}(t)$ and $x_{\mathrm{Q}}(t)$, which need to be conveniently expressed as an ensemble signal because they are both part of Eq. (1.2). To do so, the following notation is used for the baseband signal:

$$b_s(t) = \sqrt{2P_{\mathrm{I}}}x_{\mathrm{I}}(t) + j\sqrt{2P_{\mathrm{Q}}}x_{\mathrm{Q}}(t), \tag{1.3}$$

where the complex number notation j is used to represent both signal components $x_{\mathrm{I}}(t)$ and $x_{\mathrm{Q}}(t)$, simultaneously. The in-phase component $x_{\mathrm{I}}(t)$ is placed in the real part, while the quadrature component $x_{\mathrm{Q}}(t)$ is placed in the imaginary part, thus being consistent with the fact that the quadrature component is placed at 90° with respect to the in-phase one, as in Eq. (1.2).

The baseband signal in Eq. (1.3) contains essentially the same information as the bandpass signal in Eq. (1.2), understanding *information* as the content conveyed by the signal components $x_{\mathrm{I}}(t)$ and $x_{\mathrm{Q}}(t)$. It is for this reason that the baseband signal in Eq. (1.3) is referred to as the baseband *equivalent* signal of Eq. (1.2). It makes use of the well-known Euler's formula (i.e. $e^{jz} = \cos z + j \sin z$), widely used in electrical engineering.[7] In particular, the reader can check that the bandpass signal in Eqs. (1.2) and (1.3) are related as follows:

$$\begin{aligned} s(t) &= \sqrt{2P_{\mathrm{I}}}x_{\mathrm{I}}(t)\cos(2\pi F_{\mathrm{c}}t) - \sqrt{2P_{\mathrm{Q}}}x_{\mathrm{Q}}(t)\sin(2\pi F_{\mathrm{c}}t) && (1.4)\\ &= \mathrm{Re}\Big[\big(\sqrt{2P_{\mathrm{I}}}x_{\mathrm{I}}(t) + j\sqrt{2P_{\mathrm{Q}}}x_{\mathrm{Q}}(t)\big)e^{j2\pi F_{\mathrm{c}}t}\Big] && (1.5)\\ &= \mathrm{Re}\Big[b_s(t)e^{j2\pi F_{\mathrm{c}}t}\Big] && (1.6)\end{aligned}$$

with Re[·] being the real part operator. The relationship between the baseband and bandpass signals will be further elaborated in Section 1.4.3.

[7] We will use complex-plane diagrams to show the correlation results of our signal of interest with a complex replica, modulated in both the in-phase and quadrature components, in order to ensure a proper tracking of the signal. See, for example, the discrete-time scatter plot for GPS tracking in Figure 2.4. This plot is also known as a *constellation* plot.

The baseband signal $b_s(t)$ has its spectral content in the neighborhood of the zero frequency, but an exception is made for GNSS signals whose components are subcarrier modulated. In this case, we will see that the baseband equivalent signal has its spectral content around the subcarrier frequency, similarly to what happens for a bandpass signal. In fact, a subcarrier-modulated signal can actually be understood as a bandpass signal rather than a baseband one because it is still modulated by either a squared-wave cosine or sine function at a given subcarrier frequency. The plots shown in Figure 1.2 are actually the baseband spectra of the GPS L1 C/A component and the Galileo E1-B component, the latter accounts for the contribution due to the BOC(1,1) modulation only. As can be seen, the main lobes of the BOC(1,1) spectrum are located at ± 1.023 MHz, which is the subcarrier frequency for this modulation.

Tiered Structure of GNSS Signal Components

The present section is intended to set the grounds for understanding the inner structure of GNSS signals and being able to answer questions like how data symbols are conveyed, how different satellites are distinguished or how the frequency representation of GNSS signals looks like. To do so, a general formulation will be introduced to represent either the in-phase or the quadrature components, which may carry either data-modulating symbols or a pilot signal, depending on the GNSS signal. Since we will consider a general model, the $_{\mathrm{I,Q}}$ subindex will be dropped for the sake of simplicity and we will simply refer by $x(t)$ to any of the GNSS signal components.

According to Figure 1.1, GNSS signal components consist of the product between a data/pilot signal and a spreading code signal. This is the simplest interpretation, and it often suffices for understanding the underlying nature of GNSS signals, which is nothing but a spread spectrum signal. As such, one can easily understand that the receiver must then implement the inverse operation and multiply the received signal with a local replica of the spreading code. In this way, despreading takes place, and the original data/pilot signal is recovered back. However, this pragmatic interpretation often makes it difficult to analyze the inner structure of the signal and the impact of the different constituent elements and parameters on the signal, for instance, when computing the correlation and power spectral density (PSD) of the resulting signal. It is for this reason that a more in-depth analysis will be derived in this section. To do so, a tiered approach will be followed where GNSS signal components are considered to be composed of layers or tiers of different signals.

With these considerations in mind, it is found that any GNSS signal component can be expressed as[8]

$$x(t) = \sum_{i=-\infty}^{\infty} d[i]g(t - iT_{\mathrm{d}}), \tag{1.7}$$

where $\{d[i]\}_{i=-\infty}^{\infty}$ is a sequence of data-modulating symbols, each of them transmitted through the propagation medium by means of a tiered waveform $g(t)$. The time

[8] Note that sequences are indexed with the discrete-time notation $[\cdot]$, while signals are indexed with the continuous-time notation $(\cdot)$ to better emphasize their distinct nature.

duration of this waveform is limited to the symbol period T_d. Note that the signal in Eq. (1.7) is actually a pulse amplitude modulated (PAM) signal because $g(t)$ is a pulse (actually, it is not a simple pulse but a tiered waveform) whose amplitude is modulated by the symbols $d[i]$. PAM signal properties have been widely studied in the context of digital communications, and they will facilitate the understanding of some key features of GNSS signals, such as their correlation and spectral representation. Pilot GNSS signal components transmitting no symbols can be modeled as well with Eq. (1.7) by setting $d[i] = 1$ for all i.

Regarding the tiered symbol waveform $g(t)$, it is composed of N_r concatenated spreading code signals $c(t)$ whose amplitudes are modulated by the binary ± 1 values of the so-called *secondary code* sequence, $\{u[k]\}_{k=0}^{N_\mathrm{r}-1}$. That is,

$$g(t) = \sum_{k=0}^{N_\mathrm{r}-1} u[k]c(t - kT_\mathrm{code}). \tag{1.8}$$

The result is another PAM signal, where the shaping pulse is now given by the spreading code signal $c(t)$, which has a limited time duration equal to the spreading code period, T_code.

The spreading code signal $c(t)$ is in turn another tiered waveform. It is composed of N_c concatenated chip pulses $p(t)$ whose amplitudes are modulated by the binary ± 1 values of the spreading code, or so-called *primary code* sequence $\{v[m]\}_{m=0}^{N_\mathrm{c}-1}$. That is,

$$c(t) = \sum_{m=0}^{N_\mathrm{c}-1} v[m]p(t - mT_\mathrm{c}). \tag{1.9}$$

The result is again another PAM signal, where the shaping pulse is now given by the chip pulse $p(t)$, which is a real-valued pulse with a limited time duration equal to the chip period, T_c. Putting together all these terms, we can see that GNSS signal components are formed by encapsulating different layers of pulses and sequences, one inside another like in a Matryoshka doll. A summary of these signals and sequences is schematically illustrated in Figure 1.4 for the sake of clarity.

On the basis of this structure, we can see that the symbol period has a duration $T_\mathrm{d} = N_\mathrm{r}T_\mathrm{code}$, with N_r the length of the secondary code sequence and T_code the spreading code period. The symbol rate is then defined as $R_\mathrm{d} = 1/T_\mathrm{d}$. In turn, the code period is given by $T_\mathrm{code} = N_\mathrm{c}T_\mathrm{c}$, with N_c being the length of the primary code sequence and T_c be the chip period. The chip rate is then defined as $R_\mathrm{c} = 1/T_\mathrm{c}$. As an example, we

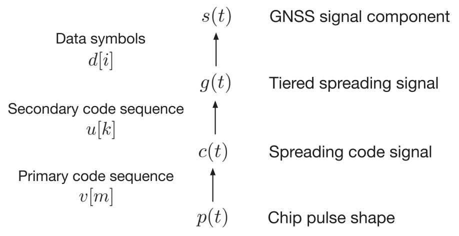


Figure 1.4 Schematic representation of the constituent elements upon which the tiered structure of a GNSS signal component is built.

can consider the GPS L1 C/A signal component whose parameters are indicated in Table 1.1. This component has a symbol rate as low as $R_\mathrm{d} = 50$ sps, which means that the symbol period is $T_\mathrm{d} = 20$ ms[9]. The chip rate is $R_\mathrm{c} = 1.023$ MHz, and the primary code length is $N_\mathrm{c} = 1{,}023$ chips, which means that the primary code period is $T_\mathrm{code} = N_\mathrm{c}/R_\mathrm{c} = 1$ ms. There is explicitly no secondary code in GPS L1 C/A, as stated in its interface control document (ICD), so these 1 ms primary codes are repeated one after another within the 20-ms symbol period. But we could understand this repetition as being brought by an all-ones secondary code code with length $N_\mathrm{r} = 20$. This interpretation will be useful for determining the PSD of GPS L1 C/A signals.

It is interesting to note that the signal model presented so far is valid for either BPSK- or BOC-modulated signals. The only difference between both is at the chip pulse. For instance, BPSK modulations typically adopt a rectangular chip pulse, occupying the whole chip period. This is the case of GPS L1 and GPS L5 signals that use

$$p_\mathrm{BPSK}(t) = \Pi\left(\frac{t - T_\mathrm{c}/2}{T_\mathrm{c}}\right), \tag{1.10}$$

which stands for a rectangular pulse of width T_c, delayed here for convenience by $T_\mathrm{c}/2$ to make it causal (i.e. so that the pulse starts at $t = 0$). The resulting chip pulse is shown in Figure 1.5(a). BOC-modulated signals, instead, are multiplied by the sign of a subcarrier waveform. This can be encompassed in the present formulation by having the chip pulse be the subcarrier waveform observed during one chip period. This results into a chip pulse composed of alternating rectangular pulses whose individual duration is a fraction of the chip period. The general formulation is given by

$$p_\mathrm{BOC}(t) = \sum_{q=0}^{N_\mathrm{scc}-1} (-1)^q \Pi\left(\frac{t - qT_\mathrm{sc}/2 - T_\mathrm{sc}/4}{T_\mathrm{sc}/2}\right), \tag{1.11}$$

where T_sc is the subcarrier period and $N_\mathrm{scc} = \frac{T_\mathrm{c}}{T_\mathrm{sc}/2}$ is the number of half subcarrier periods within one chip period. The term $T_\mathrm{sc}/4$ in the numerator is introduced to ensure that the pulse is causal, as the BPSK chip pulse in Eq. (1.10) is. A similar approach to the model used in Eq. (1.11) is adopted for instance in [4, Section 9.10], where the interested reader is referred to for more details. To put an example of Eq. (1.11), let us consider a BOC(1,1) signal such as the one whose spectrum is shown in Figure 1.2. For a BOC(1,1), there is one subcarrier period per chip period so that $T_\mathrm{sc} = T_\mathrm{c}$ and $N_\mathrm{scc} = 2$. This results into the following chip pulse:

$$p_{\mathrm{BOC}(1,1)}(t) = \Pi\left(\frac{t - T_\mathrm{c}/4}{T_\mathrm{c}/2}\right) - \Pi\left(\frac{t - 3T_\mathrm{c}/4}{T_\mathrm{c}/2}\right), \tag{1.12}$$

which consists of two consecutive and sign-reversed rectangular pulses, as shown in Figure 1.5(b). It is also worth noting that rectangular pulses in the BOC(1,1) chip pulse are narrower than that in BPSK(1), which intuitively means that BOC(1,1) should provide better accuracy in the time-delay estimation of the received signal. This topic will be discussed when addressing the PSD of both signals.

[9] Since GPS L1 C/A is using BPSK modulation, each BPSK symbol conveys one bit. This means that the bit rate and symbol rate are the same, as well as the bit period and the symbol period.

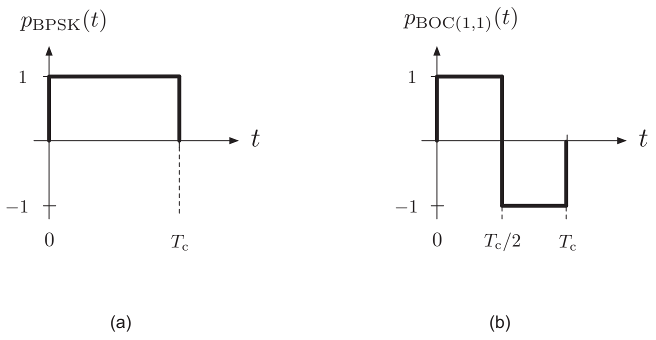


Figure 1.5 Illustration of (a) a BPSK chip pulse and (b) a BOC(1,1) chip pulse.

Merging together all the constituent elements introduced so far, Eq. (1.7) can equivalently be expressed as:

$$x(t) = \sum_{i=-\infty}^{\infty} d[i] \sum_{k=0}^{N_r-1} \sum_{m=0}^{N_c-1} u[k]v[m]p(t - mT_c - kT_{\mathrm{code}} - iT_d) = \sum_{m=-\infty}^{\infty} a[m]p(t - mT_c), \tag{1.13}$$

where the right-hand side is expressed as a function of some equivalent symbols $a[m]$ whose values depend on the combination between data/pilot symbols, secondary code and primary code of the GNSS signal. This latter expression corresponds to that of a conventional PAM.

An example of a BPSK-modulated GNSS signal component is shown in Figure 1.6 to illustrate the tiered structure that has just been presented. A generic signal has been assumed for the sake of simplicity. The top plot in the figure shows the primary spreading code $v[m]$, which is a discrete-time sequence composed of $N_c = 15$ chips at a chip rate of $R_c = 1.023$ MHz. The primary spreading code sequence is then shaped with a rectangular pulse of duration $T_c = 1/R_c$. The resulting spreading code signal is shown in the second top plot, corresponding to the continuous-time signal $c(t)$ introduced in Eq. (1.9). It can easily be seen that $c(t)$ is composed of rectangular pulses, each of them having an amplitude given by the sign of the primary spreading code sequence $v[m]$. Next, N_r replicas of the spreading code signal $c(t)$ are concatenated to form the tiered spreading signal $g(t)$ introduced in Eq. (1.8). This is done in this example using a secondary code of length $N_r = 4$, whose values are all equal to 1, that is, $u[k] = 1$ for $k = 0, 1, 2, 3$ in Eq. (1.8). The resulting signal $g(t)$ can be seen in the middle plot of Figure 1.6, where alternate dark and light colors have been used to facilitate the visual identification of each of the replicas of $c(t)$ that form $g(t)$. The resulting signal $g(t)$ becomes the waveform conveying the data-modulated symbols following the expression in Eq. (1.7), and similar to what occurs in a conventional PAM signal. This can be seen in the bottom plot of Figure 1.6, where the data-modulated signal $x(t)$ is shown for an observation period of two symbols. It can be seen how each of these symbols is shaped with the waveform $g(t)$ and then concatenated to form the resulting data-modulated signal $x(t)$. Alternate dark and light

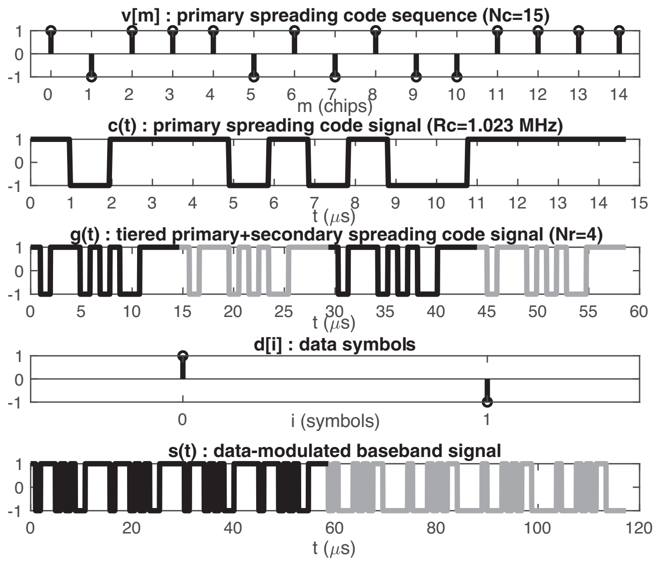


Figure 1.6 Illustration of the tiered structure of a generic GNSS signal. Example assuming a primary code sequence with $N_c = 15$ chips, chip rate $R_c = 1.023$ MHz a rectangular chip pulse with duration $T_c = 1/R_c$, and a secondary code sequence with length $N_r = 4$. The resulting signal is shown for an observation period where two binary data-modulated symbols are present.

colors are used here again to facilitate the visual identification of the two repeated waveforms.

1.3.2 Spreading Codes

The primary code sequence $v[m]$ is the cornerstone of GNSS signals. It is selected in such a way that provides a pseudorandom sequence of ± 1 values with orthogonality properties. At the same time, these antipodal values guarantee that once the sequence is pulse shaped, the resulting continuous-time signal becomes zero-mean regardless of the chip pulse, thus ensuring a power-efficient signal transmission. As for the orthogonality property, this is the key underlying principle behind CDMA. It allows signals from different satellites to coexist in the same frequency band without interfering one with each other. Perfect orthogonality has been shown to be possible for synchronous systems where the spreading sequences of different users are all perfectly aligned. Unfortunately, this is not the case in GNSS, where even if the signals were transmitted synchronously from all satellites, they travel different distances to the receiver and, therefore, they experience different propagation delays. Because of this asynchronous behavior, no perfect orthogonality can be achieved in GNSS, thus sparking the interest for *near*-orthogonal spreading codes. Some examples finally selected for GNSS are the following:

(1) Gold codes, used in the GPS L1 C/A signal and further discussed in Chapter 2.

(2) Maximum-length sequences (M-sequences), used in GLONASS and GPS L5 and outlined in Chapters 3 and 8.
(3) Memory codes, which are hand-selected, predefined codes stored in large memory tables used in Galileo signals and introduced in Chapter 4.
(4) Weil codes, based on Legendre sequences introduced in a need for finding codes of length $N_c = 10{,}230$ chips, used in the modernized GPS L1C civil signal.

Autocorrelation and Cross-Correlation

Near orthogonality means that the cross-correlation between different spreading codes is not perfectly zero but small enough to be fairly neglected.[10] It is important to note that orthogonality is assessed through the *circular* cross-correlation[11] so that for two different spreading codes, namely $v^{(p)}[m]$ and $v^{(q)}[m]$ with $p \neq q$, we have

$$R_{v^{(p)}v^{(q)}}[k] = \frac{1}{N_c}\sum_{m=0}^{N_c-1} v^{(p)}[m]v^{(q)}[m-k]_{N_c} \approx 0, \qquad \text{for } p \neq q, \tag{1.14}$$

and through the circular autocorrelation of the same spreading code, making sure that it is nearly zero outside of the central correlation lag. That is,

$$R_{v^{(p)}}[k] = \frac{1}{N_c}\sum_{m=0}^{N_c-1} v^{(p)}[m]v^{(p)}[m-k]_{N_c} \approx \delta[k], \qquad \text{for any } p, \tag{1.15}$$

with $\delta[k]$ being the discrete-time Kronecker delta, namely $\delta[k] = 1$ for $k = 0$ and $\delta[k] = 0$ for $k \neq 0$. In the above expressions, the subscript $_{N_c}$ indicates that the time indexation of the sequence $v[m-k]$ must be cyclically shifted with the period N_c whenever $m - k < 0$. Finally, it is also worth noting that the correlation definitions in Eqs. (1.14)–(1.15) assume that $N_c \gg 1$ so that the spreading code sequence can fairly be approximated by a power-type signal.[12] This will facilitate the mathematical manipulations in the limit for $N_c \to \infty$, which is often invoked to achieve the perfect orthogonality that practical sequences with finite N_c cannot provide. This assumption will help in simplifying the results when computing the correlation and the PSD of the pulse-shaped spreading code signal.

An example of the circular auto- and cross-correlation of a primary spreading code sequence is shown in Figure 1.7. The results were obtained for two of the Gold codes used in GPS L1 C/A corresponding to PRNs 3 and 7. The lack of perfect orthogonality can be seen by the presence of a small noise-like contribution in both the auto- and cross-correlation functions. Note that despite the lack of perfect orthogonality, Gold codes serve well for the purpose of selecting the signal from a given satellite while

[10] This approximation needs to be revisited when many satellites are present, since it may happen that the aggregation of all their residual cross-correlations ends up being relevant. This is actually a key metric to be assessed in the design of new GNSS signals that may share the same frequency band with legacy ones.
[11] The correlation operation is often defined with one of its factors being complex conjugated, but this is ignored herein for the sake of simplicity under the assumption that spreading codes are real-valued.
[12] A signal is power-type if its average power is finite and greater than zero, even when observed over an infinite period of time [4, Section 8.1.4]. Sinusoids are examples of power-type signals.

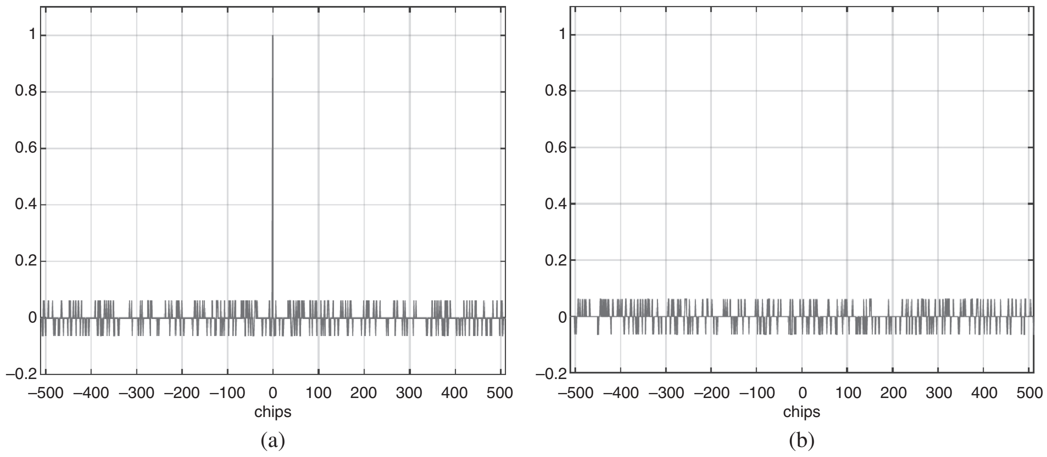


Figure 1.7 Correlation properties of the Gold spreading code with PRN $p = 3$ used in GPS L1 C/A. (a) Circular autocorrelation $R_{v^{(p)}}[k]$. (b) Circular cross-correlation $R_{v^{(p)}v^{(q)}}[k]$ for PRN $p = 3$ and $q = 7$.

disregarding all the rest. This can be seen in the high peak obtained at the autocorrelation lag $m = 0$ when correlating with the same Gold code and in the small residual for any correlation lag m between two different codes.

Details about Gold codes will be provided in Section 2.2.2, but it is interesting to note that for a generic random code the main correlation peak is on average $\sqrt{N_c}$ times larger than the residual (i.e. secondary) peaks. In terms of power, this represents an average distance ratio of N_c. This means that for a random code with the same $N_c = 1{,}023$ length as the one depicted in Figure 1.7, the main peak would have 30 dB more power than the secondary peaks. The result is a useful indication for the value that may be achieved with other spreading codes, such as the Gold codes to be described later in Chapter 2. The interested reader may refer to [4, Section 9.2.1] for further details on the analysis of random code sequences for GNSS.

The autocorrelation properties discussed so far are related to the primary spreading code, which is nothing but a discrete-time sequence. However, this sequence is actually not transmitted as it is by the GNSS satellites. Instead, it is first pulse-shaped in order to obtain a continuous-time signal that can actually modulate the carrier wave. To do so, the chips of the spreading code are pulse amplitude modulated using a pulse shape with typically the same duration as the chip period. The result was shown in Eq. (1.9), and it is reproduced below for the sake of clarity:

$$c(t) = \sum_{m=0}^{N_c-1} v[m]p(t - mT_c). \tag{1.16}$$

The spreading code signal in Eq. (1.16) is actually the one transmitted by the GNSS satellites, so it is of interest to obtain its autocorrelation. After some manipulations and assuming the chip shaping pulse $p(t)$ to have a finite duration T_c, it can be found that[13]

$$R_c(\tau_t) = N_c \sum_{k=-N_c+1}^{N_c-1} R_v[k] R_p(\tau_t - kT_c), \tag{1.17}$$

where $R_p(\tau_t)$ stands for the autocorrelation of the time-limited chip pulse $p(t)$ defined as

$$R_p(\tau_t) = \int_{-\infty}^{\infty} p(t)p(t-\tau_t)dt. \tag{1.18}$$

An example of Eq. (1.17) is shown below in Figure 1.8 for the Gold code with PRN 3 used in GPS L1 C/A.

As can be seen in Figure 1.8, the main peak of $R_c(\tau_t)$ is determined by the autocorrelation of the chip pulse shape $p(t)$. Since we are considering the GPS L1 C/A signal, a rectangular chip pulse is considered having one chip duration and the autocorrelation becomes a triangular pulse spanning from $-1 \leq \tau_t/T_c \leq 1$ chips (see zoomed view in the bottom plot of Figure 1.8). Note that the secondary peaks in $R_c(\tau_t)$ observed in the upper plot are due to the lack of perfect orthogonality in the spreading code sequence. This effect could be mitigated by letting $N_c \to \infty$, as often done in mathematical derivations to obtain a more manageable expression for $R_c(\tau_t)$. In that case, we have that $\lim_{N_c\to\infty} R_v[k] = \delta[k]$ and then $R_c(\tau_t)$ in Eq. (1.17) simplifies to

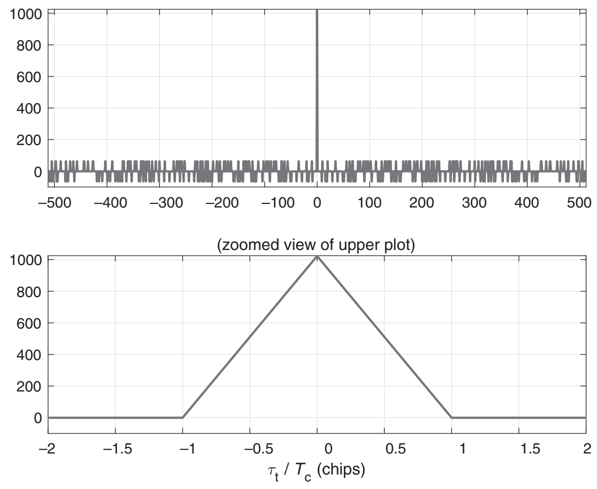


Figure 1.8 Autocorrelation of the spreading code signal, $R_C(\tau_t)$, for a GPS L1 C/A signal using the PRN 3 and a rectangular chip pulse with unit energy. Infinite bandwidth is assumed.

[13] The subindex $_t$ is used in τ_t to emphasize that it is a continuous-time delay measured in seconds, so it can be distinguished from its discrete-time version denoted simply by τ and measured in samples, which will appear when the signal is sampled.

$$\lim_{N_c \to \infty} \frac{1}{N_c} R_c(\tau_t) = R_p(\tau_t). \tag{1.19}$$

1.3.3 Power Spectral Density

The PSD provides information on how the power content of a signal is distributed in the frequency domain. This feature is important to understand how different signals behave, whether they have low- or high-frequency content (i.e. whether they exhibit slow or fast time variations, respectively), whether they incur a significant overlap or not (i.e. inter-band interference), whether they are strictly confined to the allocated frequency band or not (i.e. out-of-band interference), etc. But the PSD also plays a key role in determining how good a given signal is for ranging purposes. The reason is that the accuracy of time-delay measurements used for ranging depends on the mean square or *Gabor* bandwidth of the signal, which is actually measured through the second-order spread of its PSD [5, Section 3.11].

The PSD of a signal is related to its autocorrelation function through the Fourier transform, so for a signal $x(t)$, its PSD denoted by $S_x(f)$ is given by

$$S_x(f) = \mathcal{F}[R_x(\tau_t)], \tag{1.20}$$

with $\mathcal{F}[\cdot]$ being the Fourier transform operator and $R_x(\tau_t)$ being the autocorrelation of $x(t)$. The solution to Eq. (1.20) can easily be found for a GNSS signal component by understanding this signal as a PAM one, as already discussed in Section 1.3.1. To do so, we can take advantage of the following property, which states that for a generic PAM signal in the form

$$x(t) = \sum_{m=-\infty}^{\infty} \alpha[m] q(t - mT), \tag{1.21}$$

with some amplitude-modulating symbols $\alpha[m]$, shaping pulse $q(t)$ and symbol period T, the PSD is given by [6, Section 8.2]

$$S_x(f) = \frac{1}{T} \left[\sum_{k=-\infty}^{\infty} R_\alpha[k] e^{-j2\pi f k T} \right] S_q(f), \tag{1.22}$$

where $R_\alpha[k] = \mathrm{E}\left[\alpha[m]\alpha^*[m-k]\right]$ is the autocorrelation of the amplitude-modulating symbols and $S_q(f)$ is the energy spectral density (ESD) of the pulse, which is defined as

$$S_q(f) = \mathcal{F}[R_q(\tau_t)] = |Q(f)|^2 \tag{1.23}$$

with $Q(f) = \mathcal{F}[q(t)]$ being the Fourier transform of the pulse $q(t)$.

The spectral density in Eq. (1.23) is expressed in terms of *energy* because the pulse has a limited time duration and thus it becomes an energy-type signal.[14] In contrast, the whole PAM signal is ideally an infinite-length signal with random symbols being

[14] A signal $q(t)$ is energy-type if it has finite energy, that is, $\int_{-\infty}^{\infty} |q(t)|^2 dt < \infty$, even when observed over an infinite period of time [4, Section 8.1.4]. Rectangular pulses are examples of energy-type signals.

transmitted periodically every T seconds, and it is thus a random signal. Energy-type signals have ESD, while power-type and random signals have PSD. However, and for the sake of simplicity, we often refer to both spectral densities indistinguishably by simply the *spectrum*. Note as well that the definition of autocorrelation used for $R_\alpha[k]$ with the expectation operator E[·] is different from that used in Eq. (1.18) for the autocorrelation of the pulse. This is because $\alpha[m]$ is a (discrete-time) random signal, while $p(t)$ in Eq. (1.18) is an energy-type signal. For a discussion on the different definitions of autocorrelation for energy-type, power-type or random signals, the interested reader is referred to [6, Sections 2.3, 4.2].

The result in Eq. (1.22) is very insightful because it shows that the PSD of a GNSS signal depends on two terms. The first one is given by the autocorrelation of the amplitude-modulating symbols, which are composed of the data bits or pilots, the secondary code, the primary code, and eventually the alternating signs within the chip pulse of a subcarrier-modulated signal. The second one is given by the spectrum of the pulse. So both terms contribute in shaping the overall PSD, not only the spectrum of the pulse. Nevertheless, the latter is the only contribution for signals with uncorrelated symbols.

It is also worth emphasizing that the result in Eq. (1.22) facilitates the derivation of the PSD for GNSS signals. Two different approaches can be followed to do so depending on whether the starting point is the PAM expression in Eq. (1.7) or the one in Eq. (1.13). The latter is often simpler to substitute in Eq. (1.22) provided that the aggregated amplitude-modulated symbols $a[k]$ are zero-mean and uncorrelated. The uncorrelatedness assumption fits well when considering the spreading codes to be arbitrarily long. While this is not true in practice, one can assume that it is, approximately, in order to obtain a simplified and pedagogical result on how the PSD of a GNSS signal looks like. This approach is the easiest one for deriving the PSD of a GNSS signal, and it is often exposed first for an easy-to-understand explanation.

The second approach to take advantage of the result in Eq. (1.22) is by using the PAM expression in Eq. (1.7) as the starting point. In that case, the pulse is given by a tiered waveform, and therefore, the pulse ESD $S_q(f)$ in Eq. (1.22) is more tedious to obtain because one needs to recursively determine the ESD of each constituent waveform. The interested reader is referred to [7] for further details on this second approach.

Next, we will briefly discuss both of them in order to shed light on their advantages, their disadvantages and the difference in the final result that is obtained from each approach.

PSD for Arbitrarily Long Spreading Codes

This first approach assumes that the GNSS signal is expressed as in Eq. (1.13) so that the result in Eq. (1.22) can readily be applied by setting $\alpha[k] = a[k]$ and $q(t) = p(t)$. Considering the spreading codes to be arbitrarily long and with equiprobable random binary ±1 values, the amplitude-modulating symbols become uncorrelated due to the orthogonality properties of spreading code sequences. This means that they are totally random with no relationship between one symbol and any other, so that their autocorrelation becomes $R_a[0] = 1$ and $R_a[k] = 0$ for any other $k \neq 0$. The resulting PSD simplifies to

$$S_x(f) = \frac{1}{T_\mathrm{c}} S_p(f), \tag{1.24}$$

and therefore, it is completely determined by the spectrum of the chip pulse $p(t)$.

For the case of GNSS signals relying on BPSK modulation, the chip pulse is a rectangular one with duration equal to the chip period T_c, as discussed in Eq. (1.10). The spectrum of this pulse, applying the squared modulus of the Fourier transform as indicated in (1.23), is given by

$$S_p(f)_{|\mathrm{BPSK}} = T_c^2 \mathrm{sinc}^2(fT_\mathrm{c}), \tag{1.25}$$

where the sinc$(\cdot)$ function is defined herein as[15] $\mathrm{sinc}(x) = \frac{\sin(\pi x)}{\pi x}$. The PSD of a BPSK-modulated GNSS signal is then given by

$$S_x(f)_{|\mathrm{BPSK}} = T_c \mathrm{sinc}^2(fT_\mathrm{c}). \tag{1.26}$$

In case a BOC(1,1) modulation was considered instead, the chip pulse would be given by Eq. (1.12) and its spectrum would become

$$S_p(f)_{|\mathrm{BOC}(1,1)} = T_\mathrm{c}^2 \mathrm{sinc}^2(fT_\mathrm{c}/2)\sin^2(\pi f T_\mathrm{c}/2), \tag{1.27}$$

which comes from the direct application of the Fourier transform to Eq. (1.12). Therefore, the PSD of a BOC(1,1)-modulated GNSS signal would become

$$S_x(f)_{|\mathrm{BOC}(1,1)} = T_\mathrm{c} \mathrm{sinc}^2(fT_\mathrm{c}/2)\sin^2(\pi f T_\mathrm{c}/2). \tag{1.28}$$

It is interesting to note how the $\sin^2(\cdot)$ term in Eqs. (1.27)–(1.28), due to the alternating sign of the two short pulses forming the BOC(1,1) chip, is responsible for the spectral null that the BOC(1,1) PSD exhibits at the zero frequency. This effect can be clearly observed in the PSD plot shown in Figure 1.2. It is also worth pointing out that despite having a slightly different expression, the result in Eq. (1.28) is exactly the same as that often used by some authors and originally derived in [1], which is given by

$$S_x(f)_{|\mathrm{BOC}(1,1)} = T_\mathrm{c} \mathrm{sinc}^2(fT_\mathrm{c})\tan^2(\pi f T_\mathrm{c}/2). \tag{1.29}$$

PSD for Finite-Length Spreading Codes

Finite-length spreading codes are implemented in real-life GNSS signals, so strictly speaking, the simplifications discussed in the previous section cannot be applied. Finite-length spreading codes fit well into the tiered structure of GNSS signals introduced so far, since these codes repeat sooner or later along the GNSS signal, and thus a periodic correlation pattern does appear. In these circumstances, it is preferable to compute the PSD of the GNSS signal using Eq. (1.22) and substituting with the ele-

[15] This definition follows the convention mostly used in the literature (e.g. [6, 8]), even though the alternative definition $\mathrm{sinc}(x) = \frac{\sin(x)}{x}$ is preferred by some authors (e.g. [4]).

ments of $x(t)$ in Eq. (1.7). Then we can proceed in a tier-by-tier basis with the ESD of the finite-length tiered waveform $g(t)$, the spreading code waveform $c(t)$ and finally the chip pulse $p(t)$.

Regarding the first tier, we can start with the expression for $x(t)$ in Eq. (1.7), which is a PAM signal with binary random symbols being transmitted every T_d seconds with a waveform $g(t)$. According to Eq. (1.22), the PSD of such GNSS signal becomes

$$S_x(f) = \frac{1}{T_d} S_g(f) = \frac{1}{T_d}|G(f)|^2. \tag{1.30}$$

In order to find $G(f)$, the Fourier transform of the tiered waveform $g(t)$, it is interesting to express $g(t)$ as a function of the basic element being repeated, which is the spreading code $c(t)$. This can be done by expressing $g(t)$ as $g(t) = c(t) * u(t)$ with $u(t) = \sum_{k=0}^{N_r-1} u[k]\delta(t - kT_\mathrm{code})$ the signal containing the secondary code values only, and $\delta(t)$ the Dirac delta function. Its Fourier transform can easily be obtained as

$$G(f) = \mathcal{F}\left[g(t)\right] = \sqrt{N_r} C(f) U(f), \tag{1.31}$$

where $C(f)$ is the Fourier transform of the spreading code signal $c(t)$ and $U(f)$ is the Fourier transform of the secondary code sequence,

$$U(f) = \frac{1}{\sqrt{N_\mathrm{r}}} \sum_{k=0}^{N_r-1} u[k] e^{-j2\pi f k T_\mathrm{code}}. \tag{1.32}$$

The same procedure can be followed for the spreading code signal $c(t)$, whereby we obtain $c(t) = p(t) * v(t)$ with $v(t) = \sum_{m=0}^{N_c-1} v[m]\delta(t - mT_\mathrm{c})$ the signal containing the primary code values only. This results in

$$C(f) = \mathcal{F}\left[c(t)\right] = \sqrt{N_c} P(f) V(f), \tag{1.33}$$

where $V(f)$ is the Fourier transform of the primary code sequence,

$$V(f) = \frac{1}{\sqrt{N_\mathrm{c}}} \sum_{m=0}^{N_c-1} v[m] e^{-j2\pi f m T_\mathrm{c}}. \tag{1.34}$$

Replacing these results into the preliminar PSD expression in Eq. (1.30), we finally get

$$S_x(f) = \frac{1}{T_c}|U(f)|^2 |V(f)|^2 |P(f)|^2, \tag{1.35}$$

with $U(f)$ being the Fourier transform of the secondary code sequence, $V(f)$ the Fourier transform of the spreading code sequence and $P(f)$ the Fourier transform of the chip pulse that can be obtained as previously explained.

The result in Eq. (1.35) is very insightful. It states that the PSD of the GNSS signal does not only depend on the spectrum of the chip shaping pulse, $|P(f)|^2$, but also on the spectrum of the primary and secondary code sequences. For instance, GPS L1 C/A signals could be understood as having a secondary code with $u[k] = 1$ for $k = 0, 1, \ldots, N_\mathrm{r} - 1$ and $N_\mathrm{r} = 20$ even though, formally speaking, no secondary code is mentioned in the ICD of this signal. But for practical purposes, we could think of the repetition of primary codes within the GPS L1 C/A symbol period as the aforementioned all-ones secondary code. This involves that $|U(f)|^2$ is then given by

$$|U(f)|^2 = \frac{1}{N_r}\left|\frac{\sin(\pi f N_r T_{\text{code}})}{\sin(\pi f T_{\text{code}})}\right|^2, \tag{1.36}$$

which is known as the Fejér kernel. It is a comb-like periodic function in the frequency domain, with period $1/T_{\text{code}} = 1$ kHz for the case of GPS L1 C/A. It introduces spectral lines at 1 kHz due to the repetition of 1 ms length spreading codes every symbol period. This is an interesting observation that was ignored in the PSD of a BPSK modulated signal previously derived in Eq. (1.26), and it has some implications, for instance, on the impact of continuous-wave interference in GPS L1 C/A signals (see [4, Section 9.7.2]). Actually, frequencies where these periodic spectral lines are placed are more sensitive to the presence of a continuous wave (CW) signal. For a wider analysis on the spectral properties of finite-length spreading codes, the reader is referred to [7].

Impact of PAM Modulation on PSD

As previously mentioned, the interpretation of a GNSS signal in terms of a PAM modulation provides a compact expression that facilitates the mathematical derivation of the correlation and PSD. As for the latter, it is well-known that the PSD of a PAM signal has a shape that depends on two different contributions: the first one is provided by the spectrum of the shaping pulse, while the second one is provided by the correlation of the amplitude-modulating symbols [6, Section 8.2]. In our case, we followed two different approaches to apply this result. One of the approaches made direct use of Eq. (1.7), which in the GNSS case, depends on an intricate pulse $g(t)$ and the navigation data symbols as the amplitude-modulating symbols. Despite being a bit intricate at first glance, we showed how to proceed for the case of finite-length spreading codes, which led to the result in Eq. (1.35). The other approach made use of the expression in Eq. (1.13) that depends on the chip shaping pulse $p(t)$ and some aggregated amplitude-modulating symbols. We could easily apply this result to the case of arbitrarily long spreading codes, due to the fact that these aggregated symbols could be assumed to be all uncorrelated, thus leading to the PSD in Eq. (1.24).

This second approach helped us to emphasize that the PSD of a GNSS signal primarily depends on the spectrum of the chip shaping pulse, plus some additional contribution due to the correlation of the aggregated modulating symbols $a[m]$ in Eq. (1.13). Therefore, not only the chip shaping pulse but also the aggregated symbols have an impact on how the resulting power is distributed in the frequency domain [7].

This principle was actually driving the design of BOC signals by taking into account that the BOC chip pulse itself could be understood as a PAM signal as well, but with correlated symbols. Actually, alternating ones as shown in Eqs. (1.11)–(1.12). Such pattern leads to a zero-mean pulse, at least for even N_{scc}, and a zero-mean signal has always a null in the frequency domain at the zero frequency. This statement demonstrates why the BOC(1,1) spectrum shown in Figure 1.2 has a null at the zero frequency, which is compensated by shifting the power content at both sides of that frequency.

1.3.4 BOC Modulation

This section briefly introduces Binary Offset Carrier (BOC) modulations, already mentioned in this chapter, and which have become an essential part of modern GNSS signals. The key feature of BOC modulations is their ability to move signal power to specific parts of the allocated frequency band. This can be used to reduce interference with other signals and to improve the timing estimation accuracy by increasing the resulting mean square bandwidth. Redundancy in the upper and lower sidebands of BOC modulations also offers practical advantages for signal acquisition, code tracking, carrier tracking and data demodulation. However, some signal processing complexity and peak ambiguities are introduced as well, as discussed in detail in Chapter 10.

BOC signals are the result of modulating a GNSS signal component with a square wave subcarrier obtained from the sign of a sinusoid. The BOC modulation of the signal component in Eq. (1.7) would result in either of the two following implementations,

$$x_{\text{sinBOC}}(t) = x(t)\text{sign}\left(\sin(2\pi F_{\text{sc}}t)\right) \tag{1.37}$$

$$x_{\text{cosBOC}}(t) = x(t)\text{sign}\left(\cos(2\pi F_{\text{sc}}t)\right), \tag{1.38}$$

depending on whether the subcarrier is adopting a sine or a cosine waveform, which leads to the so-called sinBOC and cosBOC, respectively. A "BOC" signal, for short, often refers to a sinBOC signal. This was also the convention used in Eqs. (1.11) and (1.12), where a sinBOC was actually considered. Two parameters (m,n) control the generation of BOC signals, which are therefore referred in general as BOC(m,n) signals. The parameter

$$m = F_{\text{sc}}/F_0, \tag{1.39}$$

is the ratio between the subcarrier frequency F_{sc} and the reference frequency F_0 = 1.023 MHz. In turn, the parameter

$$n = R_{\text{c}}/F_0, \tag{1.40}$$

is the ratio between the chip rate R_{c} and the reference frequency. For instance, a BOC(6,1) means a 6.138 MHz subcarrier frequency and a 1.023 MHz chip rate. This signal together with a BOC(1,1) is transmitted in the E1-C component of the Galileo E1 signal. Their PSD are illustrated in Figure 1.9 and compared with the PSD of the signal transmitted in the GPS L1 C/A component.

As can be seen in Figure 1.9, even for a low-order BOC signal-like BOC(1,1), one can see that the power is split into two lobes slightly away from the L1 central frequency where the main lobe of the GPS L1 C/A PSD is placed. This confirms the ability of BOC signals to move the power content in the frequency domain. This feature can intuitively be understood by taking the ratio m/n, which provides the number of subcarrier cycles contained within a chip period. For a BOC(1,1), this ratio is equal to one, which means that the chip period contains one full subcarrier cycle. For a sinBOC(1,1), this would mean that the chip shaping pulse $p(t)$ is actually composed by

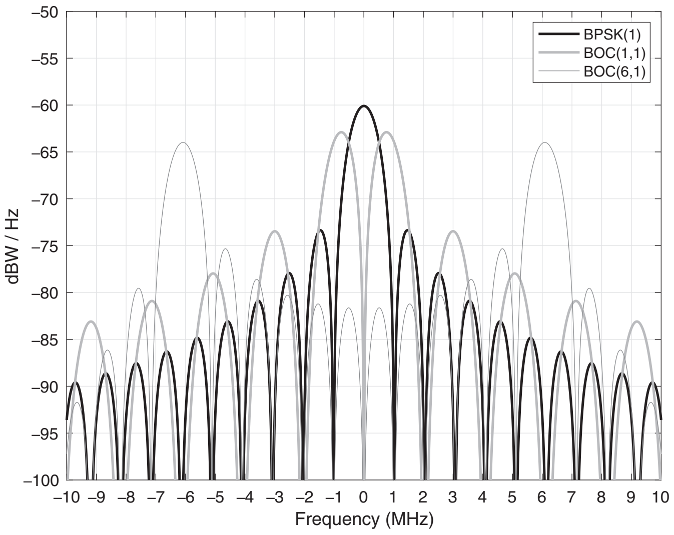


Figure 1.9 Power spectral density of the BPSK(1) modulation used in GPS L1 C/A and the BOC(1,1) and BOC(6,1) modulations used in Galileo E1. The center frequency $f = 0$ corresponds to 1575.42 MHz.

two shorter rectangular pulses, each one with a duration of half the chip period and sign-reversed. The result is a composite pulse $p(t)$ with zero-mean, which translates into the frequency domain to a pulse spectrum with a null at $f = 0$, at the expense of slightly increasing its power content at higher frequencies, as shown in the two sidelobes of the BOC(1,1) signal in Figure 1.9.

1.4 GNSS Signal Propagation, Reception and Conditioning

Before the signal is processed at the receiver, it has to be generated and then transmitted by the satellite antenna, travel a long distance until reaching the receiver antenna and be conditioned and digitally sampled in the receiver's Radio Frequency Front End (RFFE). This section briefly describes these steps.

1.4.1 Signal Propagation, Link Budget and Received Signal Strength

Link budget is the term used to calculate the power gains and losses from signal transmission to reception. A signal transmitted at a certain power P_T is then amplified by the satellite antenna, pointing to the Earth with a gain G_T. Usually, the power resulting from the term $P_T G_T$ is known as EIRP (Effective Isotropic Radiated Power, sometimes Equivalent Isotropic Radiated Power), where G_T is the maximum antenna gain, usually at *nadir*. Note that, in GNSS, the radiated power is concentrated on the Earth surface, and approximately, the same power should arrive to all users on Earth. This

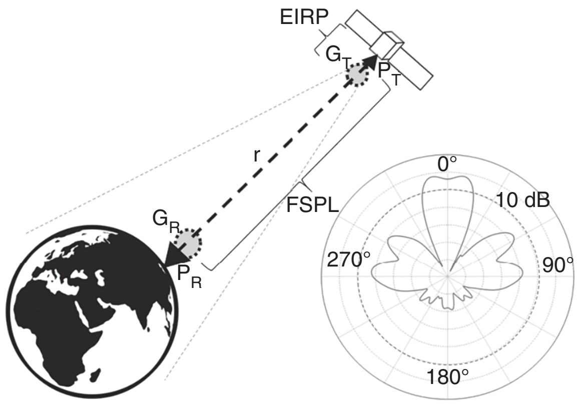


Figure 1.10 GNSS link budget. (Left) Signal path and link budget terms, including EIRP and FSPL. (Right) 2D antenna gain pattern of a Galileo satellite in the E1 (L1) frequency, 1575.42 MHz, from −15 to 20 dB, where the gain in the angle covering the Earth surface is between 10 and 15 dB [9].

leads to the antenna gain pattern shown in Figure 1.10 (right), with a tooth shape in its main lobe.

Signal polarization is another relevant property of the signals. It defines the direction of propagation of the electrical field. If we depict the electromagnetic wave as a spring between the transmitter and the receiver, if the spring departs to the right, the signal is Right-Hand Circularly Polarized (RHCP), and if it departs to the left, it is Left-Hand Circularly Polarized (LHCP). GNSS signals are RHCP. For an optimum transmission, the transmitting and receiving antennas should have the same polarization. Many receiver antennas are linearly polarized, instead of RHCP. This leads to a 3-dB loss. On the other hand, receiver antennas are often hemispherical, offering an approximate 3-dB gain for signals above the horizon, especially above a certain elevation.

After being amplified in the satellite antenna, the signal travels toward the Earth incurring free-space propagation losses, and then, it impinges on the receiver antenna, which also has some gain (G_R). This link budget calculation is expressed in the following formula:

$$P_R = \frac{P_T G_T G_R \lambda^2}{(4\pi r)^2}, \tag{1.41}$$

where λ is the wavelength and the term $(\lambda/4\pi r)^2$ is usually called free space path loss (FSPL). It encompasses two effects: the power spatial density, which decreases by a factor of $1/4\pi r^2$ as the distance r increases, and the effective area of the receiving antenna, which adds a factor of $\lambda^2/4\pi$. For example, a signal transmitted in the L1 band at 30 W, with a satellite antenna gain of 10, from a satellite at 20,200 km altitude (i.e. the approximate altitude of GPS at the zenith) and no receiver antenna gain will be received with the following power:

$$P_R = \frac{30 \cdot 10 \cdot 1 \cdot 0.19^2}{(4\pi \cdot 20.2 \cdot 10^6)^2} = 1.68 \cdot 10^{-16}\ \text{W}. \tag{1.42}$$

Usually, power amounts are expressed in dB watts, as follows:

$$\begin{aligned} P_R(\text{dBW}) &= 10\log_{10}(30) + 10\log_{10}(10) + 10\log_{10}(1) + 10\log_{10}((0.19/4\pi 20.2\cdot 10^6)^2) \\ &= 14.77 + 10 + 0 - 182.5 = -157.74\ \text{dBW}, \end{aligned} \tag{1.43}$$

which is within the range of typical GNSS signals.[16] Note that 30 W is a very low power, in the order of that of a small traditional bulb, and thus, the received power is extremely low, slightly higher than a billionth of a billionth of the transmitted power. We do not take into account signal propagation losses other than FSPL, nor receiver processing losses of around 0.5 dB. They are studied later. For a wider description of signal transmission, including GNSS antennas and propagation losses, see [9], [10] or [11].

In this section, we will also explain three relevant magnitudes of GNSS receivers for measuring the strength of the received GNSS signals: SNR, Carrier-to-Noise Density Ratio (C/N_0) and Bit Energy-to-Noise Density Ratio (E_b/N_0). We will explain how they relate to each other at the different stages of the receiver.

Signal-to-Noise Ratio (SNR)

SNR is the ratio between the signal power and the noise power, the latter mostly dominated by the thermal noise introduced by the same receiver. Sometimes it also referred to as S/N, although here we will use SNR. It is thus defined as:

$$\text{SNR} = \frac{P_s}{P_n}, \tag{1.44}$$

where P_s is the signal power and P_n is the noise power. We will show how to calculate the SNR before the correlation takes place at the signal processing stage. In our example, we assume the power of current GNSS signals, as received on Earth, is $P_s = -160$ dBW. In order to estimate the noise power P_n, we need the noise power *density*, or N_0. The noise power density expresses the noise power per frequency Hz and can be calculated as $N_0 = K_{\text{boltz}} \cdot T_e$, where K_{boltz} is the Boltzmann constant ($1.3806488 \cdot 10^{-23}$) and T_e is the effective temperature at the receiver. The effective temperature at the receiver is mainly driven by the ambient temperature T_0 (290° K). In reality, T_e is higher than T_0 due to the receiver antenna temperature and the different front-end stages, which include amplifiers and filters, especially the first stage. These stages increase the T_e with respect to T_0, and this leads to a noise power increase in the order of 1–2 dBs. However, for this example, we assume $T_0 = T_e$. A detailed explanation of how the receiver front-end affects the noise level is provided in [12, Chapter 6].

If $T_e = T_0$, $N_0 = 4 \cdot 10^{-21}$ W/Hz, which expressed in dB is −204 dBW/Hz. To get the noise power in the receiver, P_n, we just have to multiply by the receiver two-sided bandwidth:

[16] Note that, in this section, we use magnitudes in dB, as it is customary for power measurements expressed in a logarithmic scale. We assume that the reader is familiar with conversions to and from dB. In this section, they consist of using the $10\log_{10}$ conversion. We use dBW for expressing watt in a logarithmic scale and dBm for expressing milliwatt in a logarithmic scale. For example, −160dBW = −130 dBm.

$$P_n(\mathrm{W}) = N_0(\mathrm{W/Hz}) \cdot B(\mathrm{Hz}). \tag{1.45}$$

If we have a two-sided bandwidth of $B = 2$ MHz to process GPS L1 C/A BPSK(1) signal, we multiply the noise power density by a bandwidth of $2 \cdot 10^6$ Hz, which involves adding 63 dB, thus obtaining a noise power P_n of -141 dBW. Therefore, the SNR becomes

$$\mathrm{SNR[dB]} = -160\ \mathrm{dBW} - (-141\ \mathrm{dBW}) = -19\ \mathrm{dB}. \tag{1.46}$$

In contrast to most communication systems, where a positive SNR is typically obtained, here we obtain a negative SNR when the signal reaches the receiver. This means that GNSS signals, when received and sampled, are buried below the noise level. This can be apparent by looking at the samples after the ADC, where there is no trace yet of a GNSS signal. This is a consequence of using an SS modulation, and the reason why we need to correlate the signal with its replica to raise it above the thermal noise level.

Carrier-to-Noise Density Ratio

C/N_0 is just the ratio between the signal power and the noise power density,

$$C/N_0 = \frac{P_s}{N_0}. \tag{1.47}$$

As we said before, $P_n = N_0 \cdot B$, so therefore,

$$C/N_0 = \mathrm{SNR} \cdot B. \tag{1.48}$$

If we start with a noise density power of -204 dBW/Hz, as just explained, and a signal power of $C \equiv P_s = -160$ dBW at the receiver's antenna, this yields a C/N_0(dB-Hz) of $-160 - (-204) = 44$ dB-Hz.

In order to raise the signal above the noise level, we have to perform the correlation described in detail in the next section. Following our example, we will assume that a full 1-ms GPS L1 C/A code of 1,023 chips is correlated. We can depict the correlation process as the multiplication sample by sample of the signal with the replica. For example, assuming that 2,000 samples are correlated, obtained by sampling the code at approximately $N_{\mathrm{sc}} = 2$ samples per chip, we would have a $10\log_{10}(2000) \approx 33$-dB gain. Note, however, that noise samples within the chip period may be correlated, and the gain does only depend on the number of truly uncorrelated samples [12]. So the actual correlation gain will be close to the 30 dB obtained by using $N_{\mathrm{sc}} = 1$ sample per chip.[17] This 30-dB gain raises the signal above the noise level, leading to a positive post-correlation ratio ($\mathrm{SNR_{corr}}$), and allowing processing of the signal. Note that the values in this example are theoretical and do not take into account the increase in the noise level due to the receiver front-end stages mentioned earlier, and the correlation

[17] Note also that this 30-dB gain is the result of two factors: the signal power gain, of $(1{,}023)^2$, of around 60 dB, and the noise gain, of 1,023, or about 30 dB, assuming that the noise power of our signal is normalized, or $\sigma_n = 1$. The resulting correlation gain is therefore $60 - 30 = 30$ dB.

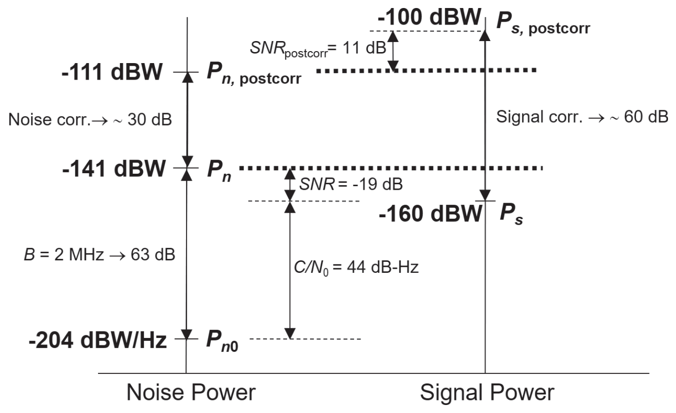


Figure 1.11 Example of noise and signal power at different stages of the receiver.

losses due to filtering, sampling and quantization, which subtract a few dBs from the correlation gain.

Carrier-to-noise density ratio (C/N_0) is used to measure the strength of a received GNSS signal with respect to the noise level, including that generated by the receiver itself in the front end. It is typically preferred to SNR in GNSS, contrary to what happens in communication systems, because SNR depends on the receiver bandwidth, while C/N_0 is bandwidth-independent. Since the bandwidth of a GNSS receiver is not fixed (e.g. it may range from whatever value from 2 MHz to 20 MHz or more), using the SNR as a performance metric can be confusing and difficult to compare among different receivers.

Bit Energy-to-Noise Density Ratio

A useful metric related to the C/N_0 is the bit energy to noise density, E_b/N_0. Since it comprises the energy per bit, it is obtained by multiplying the C/N_0 with the bit period, T_b, thus leading to

$$\frac{E_\text{b}}{N_0} = \frac{C}{N_0} T_\text{b}. \tag{1.49}$$

It gives the ratio between the energy in a bit and the noise density. The energy can be expressed in joules (or watts multiplied by seconds). The noise density is measured in watts per hertz, which can be expressed also in joules, so E_b/N_0 is dimensionless. An important feature of E_b/N_0 is that it allows us to calculate the Bit Error Rate (BER) of the received bits, before taking into account any coding scheme, through the following formula:

$$\text{BER} = \frac{1}{2}\text{erfc}\left(\sqrt{\frac{E_\text{b}}{N_0}}\right), \tag{1.50}$$

where erfc is the complementary error function.

Because of the relationship in Eq. (1.49), the BER depends on the product between the C/N_0 and the bit period. This is an interesting interplay because it means that no matter how weak the received signal power is, the data bits can still be recovered with an arbitrarily low error probability by increasing the bit period accordingly. This is a

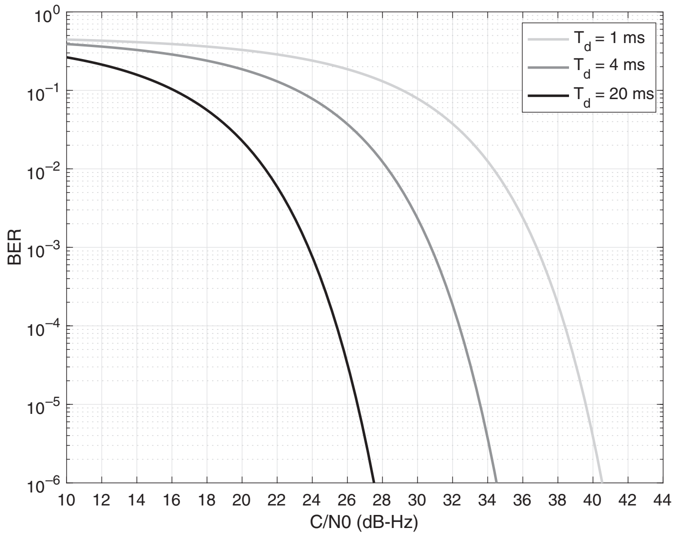


Figure 1.12 Uncoded bit error rate (BER) for different bit periods.

well-known principle in deep Space missions, where probes sending data to the Earth use very long bit periods in order to compensate for the very large propagation losses that cause the signals to be received on Earth with extremely weak power levels. As a result, the data rate is very small, just a few bits or a few tenths of bits per second.

The situation in GNSS is kind of similar and the bit period is actually quite long, for instance, 20 ms in GPS L1 C/A thus leading to a bit rate of just 50 bps. The resulting BER is shown in Figure 1.12. It corresponds to the *uncoded* BER because no encoding has been considered. In spite of this, one can see that for a low C/N_0 such as 28 dB-Hz, the BER is already below 10^{-6} or one bit in error per one million bits. If we consider the nominal C/N_0 of 44 dB-Hz already discussed in the previous section, corresponding to outdoors reception with open-sky conditions, then the BER is actually negligible if we extrapolate the results in Figure 1.12. It is so small that one could even think of deciding whether a bit is either -1 or $+1$ before the end of the bit period. It is for this reason that Figure 1.12 provides the resulting BER using a shorter period of time for deciding the bit sign, namely 4 and 1 ms. It can be seen that even when observing the received signal for just 1 ms, which would correspond to one code period in GPS L1 C/A, the BER in outdoors open-sky conditions is still very good, less than 10^{-6}.

1.4.2 Received Signal Model

As discussed in the previous section, the GNSS received signal is severely attenuated in its journey from the GNSS satellites down to the user receiver. This makes the received signal an extremely weak one, so specific countermeasures are needed to compensate for that at the receiver end. But apart from severe attenuation, the signal is received with a time delay due to the distance travelled during its journey, with a frequency shift due to the relative movement between the satellite and the user, and with a phase shift caused by both the time delay and the frequency shift onto the carrier. Additional disturbances such as ionospheric scintillation, multipath replicas or potential interference signals might be encountered as well, but they will be omitted here for the sake of simplicity. The main goal at this point is to focus on the model representing the signal received from a given satellite of interest.

We will first consider the Doppler effect experienced by the signal due to the satellite movement. This effect has an impact on the whole spectrum of the signal, with every frequency f being shifted to $(1+\frac{s_{\rm r}}{c})f$, with $s_{\rm r}$ the radial speed along the line of sight (LOS) between the satellite and the receiver, or relative speed of the satellite as seen by the receiver, and where c is the speed of light. This means that for a signal placed at the central frequency $F_{\rm c}$, such as the one in Eq. (1.2), the signal ends up being placed at the new central frequency $(1+\frac{s_{\rm r}}{c})F_{\rm c}$. Thus, it experiences a Doppler *frequency* shift given by[18]

$$\nu_t = \frac{s_{\rm r}}{c}F_{\rm c}. \tag{1.51}$$

In the sequel, we will follow the convention whereby the Doppler shift is considered to be positive when the satellite is approaching the user receiver, while it is considered to be negative when the satellite is moving away.

The impact of the Doppler effect onto the central or carrier frequency is the most visible one, but actually the same effect is experienced by every single frequency of the signal spectrum. For a signal $s(t)$ with frequency representation $S(f)$, the presence of Doppler makes the latter be converted into $S(f/(1+\frac{s_{\rm r}}{c}))$. This means that the spectral content originally at $f = F_{\rm c}$ for $S(f)$ is now found at the Doppler-shifted frequency $f = (1+\frac{s_{\rm r}}{c})F_{\rm c}$ for $S(f/(1+\frac{s_{\rm r}}{c}))$. This confirms that the spectral content was actually moved.

The effect is translated into the time domain by having the signal $s(t)$ now given by $s((1+s_{\rm r}/c)t)$, or equivalently $s((1+\nu_t/F_{\rm c})t)$ in terms of the Doppler frequency shift.[19] The result involves that the time-domain signal is compressed (i.e. when $\nu_t > 0$) or expanded (i.e. when $\nu_t < 0$) as a function of the Doppler shift. This effect will need to be accounted for at the receiver because the local replicas of the code and carrier waveforms to be generated need to match the compression or expansion of the received signal accordingly.

Let us now incorporate the Doppler effect onto the received signal. To do so, we will consider the bandpass or RF signal in Eq. (1.2) transmitted by a GNSS satellite. Only one of the two signal components of $s(t)$ is processed by the receiver at a time. For instance, GPS satellites transmit in the L1 signal $s(t)$ both the C/A and the P(Y) components, one in quadraphase with the other. But in our receiver, we are likely interested in the C/A component only. It is for this reason that we consider henceforth

[18] The subindex $_t$ is used in ν_t to emphasize that it is a frequency shift in the continuous time measured in Hz, so it can be distinguished from its discrete-time version denoted simply by ν and unitless, which will appear once the signal is sampled.

[19] The result is obtained in virtue of the property of the Fourier transform whereby $s(at) \overset{\mathcal{F}}{\leftrightarrow} \frac{1}{|a|}S(\frac{f}{a})$ for some constant a.

only one signal component in the GNSS received signal, the one to be processed by the receiver. This leads to the following received signal model:[20]

$$r'(t) = \sqrt{2C}x((1 + \nu_t/F_c)t - \tau_t) \cos(2\pi(F_c + \nu_t)t + \theta) + w'(t), \tag{1.52}$$

where C is the received power P_R in Eq. (1.41) including the receiver implementation losses, τ_t is the time delay due to the signal propagation from the satellite to the user receiver and $w'(t)$ is some bandpass noise that encompasses the contribution of the rest of satellites in the same band as well as the background and receiver noise. We can see that the signal is affected by three main parameters: the Doppler frequency shift ν_t, the propagation time delay τ_t and the carrier phase θ, the latter encompassing the effect of the time delay onto the carrier.[21]

It is also worth noting that the signal model in Eq. (1.52) is applicable to any of the two signal components in Eq. (1.2) just by adding a 90° phase offset to θ. So we are considering the carrier using the cosine function, but we could have used the sine function irrespectively, since we do not know the value of θ either. Note as well that the Doppler effect is present in both the signal component $x(t)$ and the carrier. At the carrier level, we already mentioned that Doppler introduces a frequency shift ν_t, and at the signal component level, it makes the signal waveform compress or expand depending on the sign of the Doppler shift. This effect is referred to as code Doppler, and it is very subtle because the radial speed of the satellite is orders of magnitude smaller than the speed of light, so that $\nu_t/F_c \approx 0$. Effects of code Doppler are only perceived when the received signal is observed for a long period of time. This is not the case in most GNSS applications, where just a few milliseconds of received signal are often enough to determine the user's position. In this case, the received signal model can be further simplified to

$$r'(t) = \sqrt{2C}x(t - \tau_t) \cos(2\pi(F_c + \nu_t)t + \theta) + w'(t). \tag{1.53}$$

Code Doppler will be revisited later when addressing the use of long observation periods for high-sensitivity applications, that is, when the received signal power is extremely weak. For the time being, the expression in Eq. (1.53) serves well for most applications, so we will stick to it unless otherwise stated. The interested reader can find further details on the impact of code Doppler in [13, Section 16.3.5].

1.4.3 Received Signal Conditioning

The signal impinging onto the user receiver antenna is modeled according to Eq. (1.53), which is a bandpass signal that needs to be properly conditioned so that it can be processed by a GNSS receiver. Signal conditioning takes place at the RF front end where filtering, amplification, down conversion and digitization are carried out. The output of the RF front end is typically a stream of digital samples that can readily

[20] The upperscript $'$ is used in $r'(t)$ to indicate that this is the bandpass received signal, so it can be distinguished from its down converted version denoted simply by $r(t)$.

[21] The signal model assumes that all parameters are fixed for the sake of simplicity, but in practice, they all vary due to the satellite movement.

be processed by a digital GNSS receiver, either in a hardware or software implementation. A brief summary of the tasks carried out by the RF front end is described next.

Down Conversion

As already described in Section 1.3.1, the GNSS signal components are multiplied by a carrier wave to enable radio transmission over the L band allocated to radionavigation services. Once the signal arrives at the receiver, the carrier must be removed so that the GNSS-modulated signals are recovered back. Removal of the carrier frequency is done through a process known as *down conversion*. The received carrier is converted from a high frequency down to a low frequency, which is known as *intermediate frequency* (IF), or even down to the zero frequency, which means that no carrier frequency is present anymore and the resulting signal is purely a baseband signal. Down conversion can be done gradually, in consecutive steps, or directly in a single step. Gradual down conversion is the conventional approach in most RF front ends because at each step, the resulting signal is filtered and amplified. Step-by-step filtering and amplification helps in a better rejection of the undesired image replicas that appear in the down conversion process, a better isolation between weak (input) and strong (output) signals and a relaxation of the requirements for the hardware components. The resulting architecture is known as a *superheterodyne* receiver, while the one for the single-step down conversion is known as a *direct conversion* receiver.

The fundamental concept behind down conversion is the same already used for signal modulation. It relies on the fact that multiplication of a signal by either a cosine or a sine term causes the spectrum of the signal to be shifted (i.e. *modulated*) in the frequency domain as shown in Eq. (1.54)[22]. We will adopt the cosine case even though a similar principle applies for the sine as well. Let us assume a real-valued signal component $x(t)$ with Fourier transform $X(f)$ that has a generic amplitude A and it is multiplied by a cosine term. The resulting frequency-domain representation becomes:

$$Ax(t)\cos(2\pi F_c t) \overset{\mathcal{F}}{\longleftrightarrow} \frac{A}{2}X(f+F_c) + \frac{A}{2}X(f-F_c). \tag{1.54}$$

This operation is also referred to as *up conversion*, and it results in a bandpass signal whose spectral content is placed in the neighborhood of the carrier frequency F_c, as seen in the upper part of Figure 1.13.

At the receiver side, we have two options to compensate for the carrier frequency. The first one is to shift the spectrum of the bandpass signal in Eq. (1.54) down to a smaller carrier frequency, F_{IF}, which makes the resulting signal much easier to manipulate than with the high frequency of the initial carrier. This can be done by multiplying the signal in the left-hand side of Eq. (1.54) by $B\cos(2\pi(F_c - F_{IF})t)$ for a generic amplitude B, thus leading to

$$\begin{aligned} Ax(t)\cos(2\pi F_c t)B\cos(2\pi(F_c - F_{IF})t) &\overset{\mathcal{F}}{\longleftrightarrow} \frac{AB}{4}X(f+2F_c-F_{IF}) \\ + \frac{AB}{4}X(f-2F_c+F_{IF}) &+ \frac{AB}{4}X(f+F_{IF}) + \frac{AB}{4}X(f-F_{IF}). \end{aligned} \tag{1.55}$$

[22] Down-conversion can also be understood as an example of the general trigonometric formula $\cos(a)\cos(b) = \frac{1}{2}[\cos(a-b) + \cos(a+b)]$.

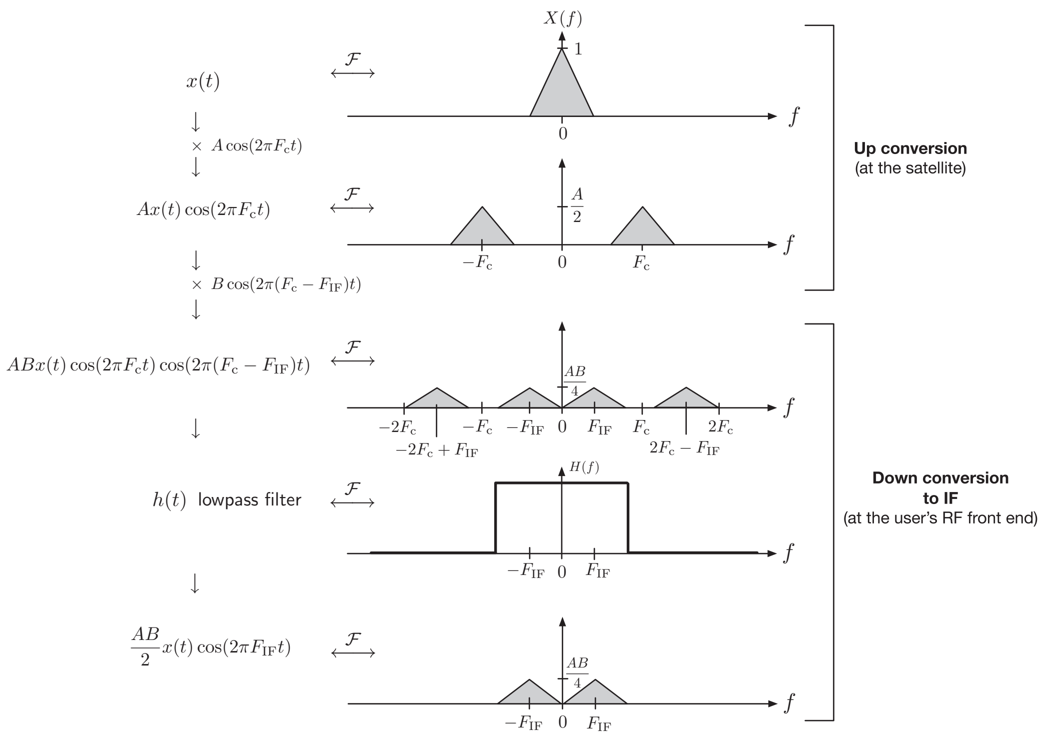
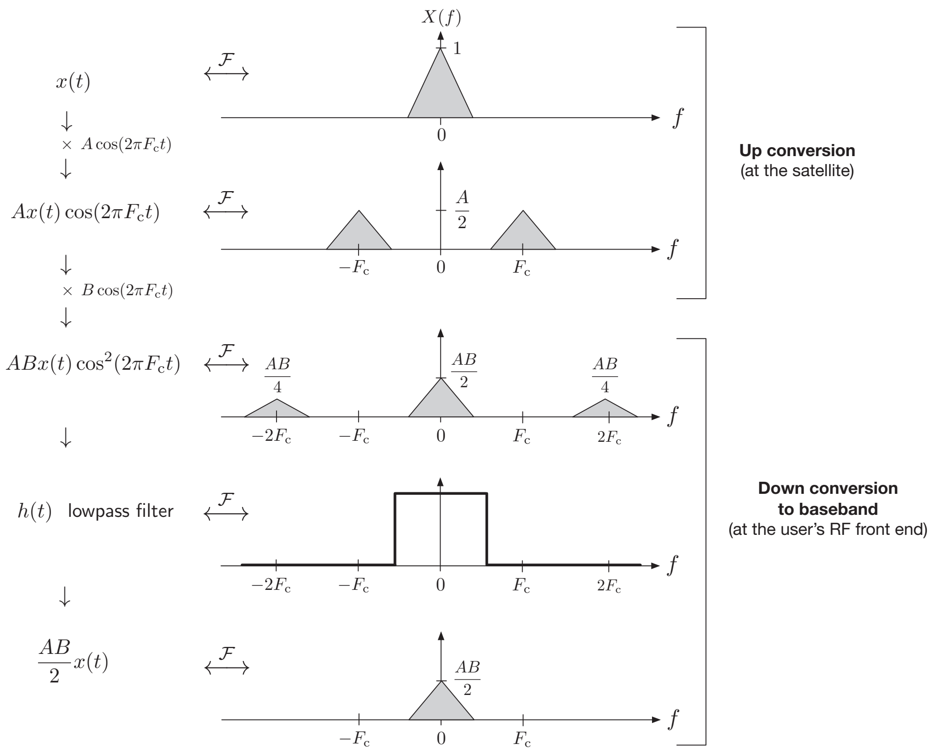

$x(t)$
$\times A\cos(2\pi F_c t)$
$Ax(t)\cos(2\pi F_c t)$
$\times B\cos(2\pi(F_c - F_{IF})t)$
$ABx(t)\cos(2\pi F_c t)\cos(2\pi(F_c - F_{IF})t)$
$h(t)$ lowpass filter
$\frac{AB}{2}x(t)\cos(2\pi F_{IF} t)$
$\mathcal{F}$
$X(f)$
$H(f)$
$\frac{A}{2}$
$\frac{AB}{4}$
$-2F_c$, $-F_c$, $-F_{IF}$, 0, F_{IF}, F_c, $2F_c$
$-2F_c + F_{IF}$
$2F_c - F_{IF}$
f
Up conversion (at the satellite)
Down conversion to IF (at the user's RF front end)

Figure 1.13 Illustration of the down conversion to IF through the frequency representation of the signal at each step of the process.

The result is illustrated in the central part of Figure 1.13 where the frequency representation in the right-hand side of Eq. (1.55) is shown. As can be seen, two replicas of the signal spectrum are placed at $\pm F_{IF}$, which is typically a much lower frequency (i.e. in the order of a few MHz) compared to the initial carrier in the L band. A lowpass filter is then applied to remove the aliased replicas at high frequency and to keep the low-frequency replicas only, as shown in the bottom part of Figure 1.13.

Down conversion to IF is an intermediate step to recover the signal of interest $x(t)$ because the result is not $x(t)$ itself but a bandpass version of it, at a smaller carrier frequency than the initial one. Many RF front ends provide at their output an already digitized down-converted signal at IF. One of the advantages of doing so is that the IF signal continues to be real-valued and therefore can be sampled with a single analog-to-digital converter (ADC). This was the case of the RF front end accompanying the preceding book, where the output signal was digitized as real-valued samples at an IF of either 1.364 or 4.092 MHz [14].

The second option for down conversion is to fully down convert to baseband so that the spectrum of the signal of interest is placed at the zero frequency. This would be the reverse operation to the up conversion carried out in Eq. (1.54), and it involves multiplying the received signal again by a cosine term with the same frequency, $B\cos(2\pi F_c t)$, where a generic amplitude B is considered. The result becomes



$x(t)$
$\times\ A\cos(2\pi F_c t)$
$Ax(t)\cos(2\pi F_c t)$
$\times\ B\cos(2\pi F_c t)$
$ABx(t)\cos^2(2\pi F_c t)$
$h(t)$ lowpass filter
$\frac{AB}{2}x(t)$
$\mathcal{F}$
$X(f)$
1
$\frac{A}{2}$
$\frac{AB}{4}$
$\frac{AB}{2}$
$-2F_c$
$-F_c$
0
F_c
$2F_c$
f
Up conversion (at the satellite)
Down conversion to baseband (at the user's RF front end)

Figure 1.14 Illustration of the down conversion to baseband through the frequency representation of the signal at each step of the process.

$$ABx(t)\cos^2(2\pi F_c t) \overset{\mathcal{F}}{\longleftrightarrow} \frac{AB}{4}X(f+2F_c) + \frac{AB}{2}X(f) + \frac{AB}{4}X(f-2F_c), \quad (1.56)$$

where the original spectrum $X(f)$ is recovered and two high-frequency replicas are obtained as shown in the central part of Figure 1.14. These replicas are then removed by applying a lowpass filter, and the result is shown in the bottom part of Figure 1.14. The result in Eq. (1.56), once the lowpass filtering is applied to remove the high-frequency aliases, provides the signal of interest.

The previous example serves well for illustration purposes and it assumes that the original signal was up converted using a cosine term, and then down converted using the same cosine term aligned in frequency and phase. But this situation hardly ever occurs in practice. In particular, the received signal in Eq. (1.53) is affected by a Doppler frequency shift and a phase offset that are both unknown to the receiver. So it is impossible for the front end to perfectly know beforehand how the cosine term to be implemented should be. If down conversion is implemented considering the nominal frequency, that is, using the term $\cos(2\pi F_c t)$, the output would then be given by the expression at the output of the upper branch in Figure 1.15. As can be seen, the down-converted signal is still affected by a cosine term given by $\cos(2\pi\nu_t + \theta)$. To illustrate the situation, let us assume that no Doppler shift is present such that $\nu_t = 0$. Then, if

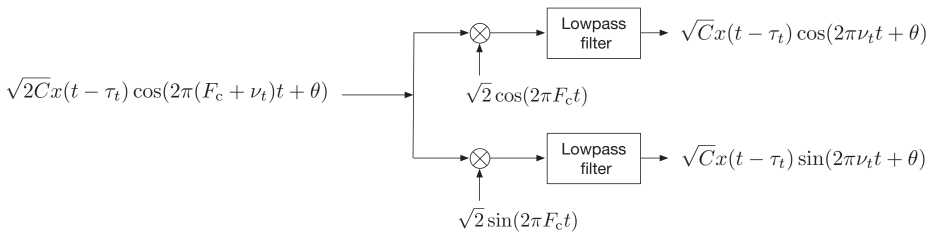

$\sqrt{2C}x(t-\tau_t)\cos(2\pi(F_c+\nu_t)t+\theta)$
Lowpass filter
$\sqrt{C}x(t-\tau_t)\cos(2\pi\nu_t t+\theta)$
$\sqrt{2}\cos(2\pi F_c t)$
Lowpass filter
$\sqrt{C}x(t-\tau_t)\sin(2\pi\nu_t t+\theta)$
$\sqrt{2}\sin(2\pi F_c t)$

Figure 1.15 Implementation of down conversion to baseband where both cos and sin branches are considered. An amplitude $\sqrt{2}$ has been used in the cos and sin multipliers so that the power of the complex baseband signal coincides with that of the bandpass or RF received power C.

θ happens to be zero or π, the signal of interest will appear in the upper branch, but if θ happens to be equal to $\pi/2$ or $-\pi/2$, the signal at the output of this upper branch would be zero. It is for this reason that both the cosine and sine branches need to be implemented in practice when performing down conversion to baseband as shown in Figure 1.15, regardless of the fact that the original signal in Eq. (1.54) had only a cosine term. Once the outputs of both branches are available, they are merged together and a complex baseband signal is obtained, as already discussed in Eq. (1.3). Since the baseband down-converted signal is complex, in contrast to the IF down-converted one, two ADC are needed to sample the in-phase (I) and quadrature (Q) components of the resulting signal.

Taking the aforementioned considerations into account, and following the down conversion approach in Figure 1.15, the received signal in Eq. (1.53) would be down converted by the RF front end and result in the following complex-valued baseband signal,

$$r(t) = \sqrt{C}x(t-\tau_t)\cos(2\pi\nu_t t+\theta) + j\sqrt{C}x(t-\tau_t)\sin(2\pi\nu_t t+\theta). \tag{1.57}$$

Adding the contribution of the noise, which has been omitted so far for the sake of simplicity, the complex-valued baseband signal can be expressed for convenience as

$$r(t) = \sqrt{C}x(t-\tau_t)e^{j(2\pi\nu_t t+\theta)} + w(t), \tag{1.58}$$

where $w(t)$ contains the complex-valued baseband contribution of the noise, which is obtained following a similar reasoning to that of the signal obtained at the output of Figure 1.15. The formulation in Eq. (1.58) follows the same approach as in [4, Eq. (11.20)], where the received and down-converted signal is formulated as a complex-valued signal, in contrast to some other contributions in the GNSS literature where the same signal is formulated as two independent and real-valued signals (i.e. the in-phase and quadrature components) that are processed separately.

The presentation provided herein is just a glimpse of the fundamentals behind the down conversion process going on at the RF front end. The interested reader will find a complete and comprehensive discussion on the topic in [4, Section 11.1], [14, Chapter 4] or [13, Section 14.3].

Sampling and Quantization

Once down conversion takes place, the next step at the RF front end is to convert the resulting signal into a *digital* signal so that it can be processed by either a digital hardware or a software receiver implementation. Digital hardware implementations have been the *de facto* standard for a long time already due to their reliability, simplicity and flexibility compared to analog implementations. Rapid advances in very large-scale integration (VLSI) circuits during the last decades have also contributed to dramatically reducing the manufacturing cost and the device size. But when it comes to a software receiver implementation, as we are targeting in this book, digital signal processing is implicitly required since all operations are carried out in a digital signal processor (DSP) or a multipurpose personal computer, and thus, implemented in the digital domain. It is for this reason that the input signal to a software receiver is always a file with digital signal samples that are obtained at the output of an RF front end.

A *digital* signal is actually the result of two different operations. The first one involves sampling a continuous-time signal so that it is converted into a discrete-time one. The latter is a signal having samples of the original signal at discrete-time instants only, and thus, being discrete in the time domain. The second operation involves quantizing the value of each sample so that it can be represented with fixed precision using a finite number of bits. Sampling and quantization are both carried out by an analog-to-digital (ADC) converter at the receiver front end, and they are controlled by the sampling rate and the number of bits.

As for the first operation, sampling is typically done on the down-converted signal. Two main considerations must be borne in mind. The first one is that the selected sampling frequency should always fulfill the Nyquist criterion whereby $F_s \geq 2B_{bb}$, with B_{bb} the baseband (i.e. one-sided) signal bandwidth or $F_s \geq B$, with B the bandpass (i.e. two-sided) signal bandwidth. The second consideration is that one should carefully choose F_s such that it is not an integer multiple of the chip rate R_c in order to achieve sub-sample accuracy in the time-delay estimation of the received signal [15, 16].

In the recent years, some RF front ends have been implementing both down conversion and sampling into the same step. This is the so-called *IF or bandpass sampling*, a process whereby the bandpass-received signal is directly sampled at a wisely chosen sampling frequency [4, Section 11.1.3], [13, Section 14.3.3], [17]. This is done in order to benefit from the spectral aliases that naturally appear as a result of the sampling process, which actually constitute down-converted versions of the original signal obtained for free. That is, without having to explicitly implement a down conversion stage with an RF mixer. The reason why such spectral aliases appear is due to the well-known property in digital signal processing whereby sampling a signal in a given domain (i.e. time or frequency) results in aliased replicas in the other domain (i.e. frequency or time, respectively). This means that replicas of the spectrum of a GNSS signal at a central frequency F_c will appear at frequencies $F_c \pm mF_s$, for any integer m, once the signal is sampled at a sampling frequency F_s.

The key point of bandpass sampling is then to wisely choose F_s such that: (i) it avoids aliasing between consecutive replicas of the signal spectrum and (ii) it places one of these replicas as close as possible to the zero frequency. This would provide a discrete-time signal without aliasing while at the same time already down-converted, ready to be processed by the subsequent stages of the receiver. The penalty to be paid is that some degradation is incurred by the superposition of noise aliases in the frequency domain, which lead to a slight increase in the noise floor. The interested reader can find a detailed description of bandpass sampling for GNSS in [4, Section 11.1.3].

Regarding the quantization process, which is the second task carried out by the ADC, the key parameter is the number of bits to represent the value of each sample. Due to the extremely weak received power of GNSS signals, actually only one bit would suffice. The losses would be just 2 dB with respect to the case without quantization [18, Section III.C.4]. But in practice it is often customary to work with two bits per sample (e.g. sign and magnitude), which incurs just 0.5-dB losses, and at the same time, it is easy to handle by storing four signal samples into one byte. Beyond four bits per samples, the quantization losses are negligible. The only reason to move further beyond would be the case when interference signals are present. In that case, since terrestrial interferences have power levels much higher than the received GNSS signals, a larger dynamic range is needed at the ADC to avoid overload distortion. In these situations, it is also interesting to disable the automatic gain control (AGC) of the RF front end to avoid that the interfered GNSS signals might be further attenuated by the AGC when accommodating the dynamic range of the interference plus GNSS-received signal.

1.4.4 Discrete-Time Received Signal Model

In the sequel, and for the sake of simplicity, we will ignore quantization in the mathematical formulation unless otherwise stated. We will also assume that the signal is fully down-converted and placed at baseband and that a proper sampling frequency has been chosen. This results in the discrete-time counterpart of the complex-valued baseband received signal already introduced in Eq. (1.58), leading to:

$$r[n] = \sqrt{C}x[n-\tau]e^{j(2\pi\nu n+\theta)} + w[n], \tag{1.59}$$

where $r[n] = r(nT_s)$ and $x[n] = x(nT_s)$ are the discrete-time versions of the received signal and the signal component of interest, respectively, whereas $w[n] = w(nT_s)$ is the discrete-time complex-valued noise contribution. The parameters $\{\tau, \nu\}$ are now the discrete-time counterparts of the continuous-time delay and the aggregated frequency error including the Doppler effect and the frequency offset of the local oscillator. That is,

$$\tau = \tau_t/T_s, \qquad \nu = \nu_t/F_s, \tag{1.60}$$

with T_s being the sampling time and $F_s = 1/T_s$ the sampling frequency.

In the discrete-time domain, the data symbol period in Eq. (1.7), the code period in Eq. (1.8) and the chip period in Eq. (1.9) become the number of samples per data symbol N_{sd}, the number of samples per code N_{scode} and the number of samples per chip N_{sc}, respectively, which are defined as:

$$N_{\mathrm{sd}} = T_{\mathrm{d}}/T_{\mathrm{s}}, \tag{1.61}$$

$$N_{\mathrm{scode}} = T_{\mathrm{code}}/T_{\mathrm{s}}, \tag{1.62}$$

$$N_{\mathrm{sc}} = T_{\mathrm{c}}/T_{\mathrm{s}}. \tag{1.63}$$

Finally, a piece of received signal of T seconds will be processed at the receiver, first for signal acquisition, and then for each measurement generated at the signal tracking stage. This leads to a snapshot of N_T received signal samples in the discrete-time domain with

$$N_T = T/T_{\mathrm{s}}. \tag{1.64}$$

In contrast to the previous parameters in Eqs. (1.61)–(1.63), the snapshot length N_T will always be an integer number in practice because we cannot get a fraction of sample from the ADC output of the RF front end. In practice, we will be using $\lfloor N_T \rfloor$ samples, where $\lfloor x \rfloor$ stands for the *floor* of x. That is, the greatest integer smaller than or equal to x.

1.5 Receiver Architecture

Once the received signal has been conditioned and converted into the discrete-time domain by the RF front end, the output stream of digitized samples is ready to be processed by the GNSS receiver. The first task to be carried out by the receiver is to align its local code and carrier replicas to the actual code and carrier of the received signal. Such alignment is done following a two-stage approach. In the first stage, the correlation between the received signal and the local spreading codes is implemented for each satellite using coarse tentative delays and Doppler frequencies. This stage is considered to be successful when the satellite number, identified by its PRN, and a coarse estimate of its delay and Doppler frequency, are identified. In the second stage, the receiver implements a fine alignment to precisely track any possible time variation of the aforementioned parameters due to the satellite's movement and the user's dynamics. These two stages correspond to the so-called acquisition and tracking, whose blocks are shown next in Figure 1.16 as part of the architecture of a GNSS receiver.

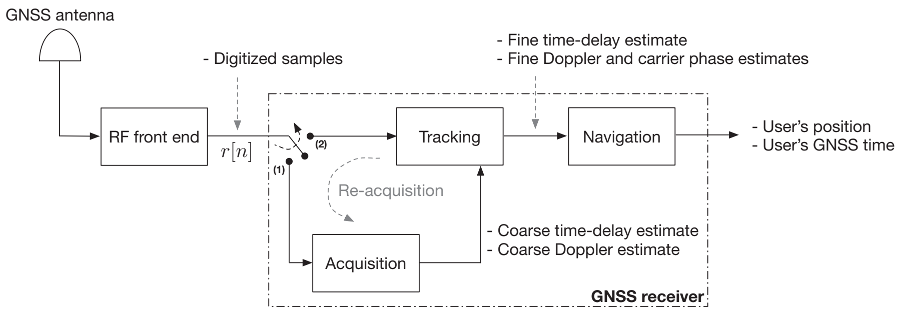


Figure 1.16 Schematic block diagram of a GNSS receiver.

Acquisition is performed first on a satellite-by-satellite basis, meaning that typically only one satellite is searched at a time. When the satellite of interest is successfully acquired, the receiver switches to tracking mode where it remains until this satellite is lost. In some situations, the satellite may still be present in the sky, but the reception is blocked due to an obstacle in the LOS. In that case, the acquisition stage is restarted for this satellite and its signal is reacquired, if still visible. After the tracking stage, the signal received from the satellite is decoded and the navigation data bits are extracted. Hardware receivers implement the acquisition and tracking of each satellite using parallelized hardware elements or *channels*, such that each channel acquires and tracks one satellite and many channels are simultaneously run in parallel. The acquisition and tracking stages will be discussed in Section 1.6 and Section 1.7, respectively.

Finally, when the observables of time delay and Doppler frequency are available and the navigation data have been decoded, the receiver is able to determine the satellite positions and then to estimate the user's position based on solving the so-called navigation equations. This task is carried out at the navigation module shown in Figure 1.16 where the user's position as well as the GNSS time are obtained. The procedure to do so will be described in Sections 1.8 and 1.10.

1.6 Acquisition

In order to compute the range measurements and to get the satellite data, the receiver first needs to be synchronized in frequency and time with the signal coming from the visible satellites. This involves determining, at least coarsely, the frequency shift ν and the propagation delay τ in Eq. (1.59), a task that is referred to as *acquisition*. The procedure is similar to that of tuning the radio of our car to a radio station. But in contrast to radio signals, GNSS signals are buried below the noise floor and exhibit large frequency shifts due to the satellite's motion, so the acquisition problem is much more difficult to solve.

Discrete-time formulation will be considered henceforth motivated by the fact that all GNSS signal processing is nowadays implemented in the digital domain. That is, using either digital hardware, such as an ASIC (Application-Specific Integrated Circuit) or an FPGA (Field-Programmable Gate Array), or digital software implementations (e.g. software receiver running on a host computer), the latter being the case of the present book. The discrete-time received signal $r[n]$ in Eq. (1.59) is processed by the acquisition stage following a three-step approach as illustrated in Figure 1.17:

1. First, the received signal is correlated with a local replica (i.e. each sample is multiplied and the result is summed to the previous multiplications) using tentative values of time delay and frequency shift. These tentative values are denoted as $\{\tau', \nu'\}$, respectively, in order to distinguish them from the true values $\{\tau, \nu\}$ being sought. When the received signal power is too low, the correlation time must be increased accordingly in order to be able to detect the signal. This is due to the fact that signal detection depends on the received energy, and energy is the product of power by time. So for a fixed received power, the collected energy

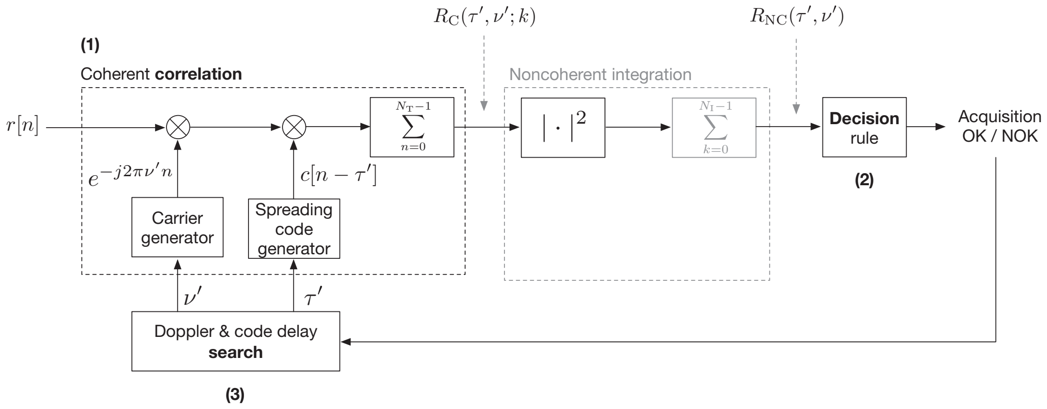


Figure 1.17 Architecture of the acquisition stage where the three steps discussed above are indicated: correlation with the received signal (either just coherently or in combination with noncoherent accumulations); decision rule to decide whether acquisition is successful for the pair of tentative $\{\tau', \nu'\}$ values; and finally, the search for a new pair of tentative values when those being used did not lead to a successful acquisition.

is increased by lengthening the observation time as well. Extending the correlation over a long time period suffers from problems that will be discussed later, so in practice the correlation time cannot be increased without bound. It is for this reason that the total correlation time must be split into a set of N_I shorter coherent correlations $R_C(\tau', \nu'; k)$ for $k = 0, 1, \ldots, N_I - 1$ that are noncoherently accumulated, that is, squared and summed, as indicated in Figure 1.17, resulting in the noncoherent correlation $R_{NC}(\tau', \nu')$. This feature will be described later and allows the receiver to increase the sensitivity. Receivers implementing this feature are known as *high-sensitivity* GNSS receivers. For receivers operating in good visibility conditions and thus receiving a high enough power, a short correlation time is enough and noncoherent accumulations are not needed. This is the reason why this block is shaded in Figure 1.17, to highlight that it is optional.

2. When the tentative $\{\tau', \nu'\}$ are close enough to the true values, the correlation of the received signal with the local replica results in a large value compared to the case when the local replica is poorly aligned. A result similar to the central peak in Figure 1.7(a) should be obtained. So the second step of the acquisition stage involves detecting whether the correlation output is large enough to consider that the acquisition is successful. The latter means that the satellite being searched is present, and the local replica is well enough aligned to it. To do so, a decision rule must be applied on the correlation output value, which essentially boils down to comparing this value with a predefined threshold.
3. In case acquisition is not successful, which means that the output correlation value is not large enough, then the receiver must come back to step (1) and try again by correlating the received signal with a new pair of tentative $\{\tau', \nu'\}$ values. The search for the right values for $\{\tau', \nu'\}$ can be done sequentially for both time and frequency, or in parallel for time but sequentially for frequency. This search is the

most consuming part of the acquisition stage because many tentative values for $\{\tau', \nu'\}$ need to be tested. These values are taken by discretizing the search space into bins of $\Delta\tau'$ width on the time axis and $\Delta\nu'$ width on the frequency axis. Significant savings can be obtained in case assistance information is available, so the receiver can narrow down the search space, reduce the number of time–frequency bins to test, and consequently speed up the acquisition time [12].

Once the three main steps of the acquisition stage have been introduced, we will discuss in more detail each of them, while providing some insights into their practical implementation.

1.6.1 Correlation of the Received Signal with the Local Replica

As already introduced, the first step of the acquisition stage is to correlate the received signal with a local replica of the satellite signal being searched. The replica is generated by the receiver at both code and carrier levels. For the code level, let us omit at this time the presence of data bits and the secondary code, so that we can concentrate essentially on the correlation with the primary spreading code only. This would lead the receiver to implement the following correlation,

$$R_C(\tau', \nu') = \frac{1}{N_T} \sum_{n=0}^{N_T-1} r[n] c^*[n-\tau']_{N_{\text{scode}}} e^{-j2\pi\nu' n} \tag{1.65}$$

where we can see the replica of the discrete-time spreading code $c[n] = c(nT_s)$ at a tentative discrete-time delay τ', and the replica of the discrete-time complex carrier at a tentative discrete-time frequency shift ν'. Note that since the received signal $r[n]$ and the local carrier replica in Eq. (1.65) are both complex-valued, so is the result of the correlation $R_C(\tau', \nu')$. Furthermore, and because of the finite length of the spreading code, the indexation of its replica $c^*[n-\tau']$ in Eq. (1.65) must ensure that $0 \leq |n-\tau'| < N_{\text{scode}} - 1$. This is achieved by indexing the code replica in a cyclic or modular manner with period N_{scode}. That is why the subindex $_{N_{\text{scode}}}$ is indicated in the code replica in Eq. (1.65), as done for the circular correlation in Eqs. (1.14)–(1.15). For the same reason, the time delay τ that we are looking for can only be resolved within a spreading code period. This means that our estimates for τ are just the fractional part of the true (absolute) time delay within one code period.

In the sequel, we will define the error between the tentative and the true values of the time delay and frequency shift as

$$\begin{array}{llcll} \tau_{t,\epsilon} = \tau_t - \tau_t' & \text{(seconds)} & \rightarrow & \tau_\epsilon = \tau - \tau' & \text{(samples)} \\ \nu_{t,\epsilon} = \nu_t - \nu_t' & \text{(Hz)} & \rightarrow & \nu_\epsilon = \nu - \nu' & \text{(unitless).} \end{array} \tag{1.66}$$

It is important to mention that the correlation in Eq. (1.65) can either be implemented in the time domain as in most hardware implementations (e.g. following a matched filter approach as in Figure 1.17) or in the frequency domain as in most software implementations. The latter takes advantage of the property of the Fourier transform whereby the correlation in the time domain can be implemented as a multiplication in the frequency domain. The Fast Fourier Transform (FFT) is used in practice to speed up the computations, as discussed in Section 1.6.3. Examples of time-domain implementations can be found in [19, 20, 21], whereas examples of frequency-domain implementations such as the classical FFT-based acquisition, the half-bit acquisition method or the double block zero padding method can be found in [22] and [23], respectively.

The squared mean value of Eq. (1.65) leads to the so-called *ambiguity* function, which provides information on the signal power that is obtained at the correlator output as a function of the tentative values τ' and ν'. Therefore, it provides information on how *sensitive* the correlator output is to changes on these tentative values. Ideally, we would like the correlator output to be very sensitive to τ' and ν' because this would allow us to precisely determine the best (i.e. the most accurate) tentative value to be used. It is for this reason that the ambiguity function is widely used in signal design, as it is the case, for instance in radar signal processing, for determining the best time–frequency properties of new signal waveforms. For our application at hand, where the signal waveform is already given to us, the ambiguity function is used as an indicator function to determine how time–frequency misalignments of the local replica do affect the received signal power. In this way, we can use the ambiguity function as a cost function to be maximized as a function of τ' and ν'. The tentative values of τ' and ν', where the ambiguity function becomes maximum, are those values for which the local replica is properly aligned with the received signal and thus the maximum received signal power is obtained. In other words, the values of τ' and ν' for which the ambiguity function is maximized become the estimates of the actual time delay and frequency shift of the received signal.

We will address the incurred losses later on, but for the time being, let us concentrate on the ambiguity function. The latter is nothing but the squared magnitude of the complex-valued correlation in Eq. (1.65) as a function of the difference between the true and the tentative time delay and frequency shift. That is,

$$\Psi(\tau_\epsilon, \nu_\epsilon) = \left| R_\mathrm{C}(\tau', \nu') \right|^2, \tag{1.67}$$

where τ_ϵ and ν_ϵ are the time delay and frequency shift errors defined in Eq. (1.66). Substituting $R_\mathrm{C}(\tau', \nu')$ and ignoring the noise contribution, we obtain

$$\Psi(\tau_\epsilon, \nu_\epsilon) = \frac{C}{N_T^2} \left| \sum_{n=0}^{N_T-1} c[n-\tau] c^*[n-\tau']_{N_\mathrm{scode}} e^{j2\pi\nu_\epsilon n} \right|^2 \tag{1.68}$$

$$= \frac{C}{N_T^2} \left| \frac{R_p(\tau_\epsilon)}{R_p(0)} \frac{\sin(\pi\nu_\epsilon N_T)}{\sin(\pi\nu_\epsilon)} \right|^2. \tag{1.69}$$

A long enough spreading code has been assumed when deriving Eq. (1.69) so that the autocorrelation of the spreading code signal could be approximated by the autocorrelation of the chip shaping pulse, only. Note that for $\nu_\epsilon = 0$, the ambiguity function simplifies to

$$\Psi(\tau_\epsilon, 0) = C \left| R_p(\tau_\epsilon) / R_p(0) \right|^2, \tag{1.70}$$

which means that in the absence of frequency errors, the accuracy for determining the time delay of the received signal is determined by the shape of the chip pulse autocorrelation, R_p. This is a very important result because it means that the sharper the chip pulse autocorrelation, the easier it will be to detect small variations on $\tau_\epsilon \approx 0$ and thus, the more accurate the time-delay estimation will be. A sharper autocorrelation involves a wider spectrum, so this observation links time-delay accuracy with the bandwidth of the signal.[23]

On the other hand, even when the local code replica is time aligned to the received signal, the presence of residual frequency errors degrades the correlation in Eq. (1.65). This can be observed by setting $\tau_\epsilon = 0$ in Eq. (1.69), which leads to[24]

$$\Psi(0, \nu_\epsilon) = \frac{C}{N_T^2}\left|\frac{\sin(\pi\nu_\epsilon N_T)}{\sin(\pi\nu_\epsilon)}\right|^2 \overset{|\nu_\epsilon|\ll 1/2}{\simeq} C\,\mathrm{sinc}^2(\nu_{t,\epsilon}T). \tag{1.71}$$

The result in Eq. (1.71) indicates that whenever the product $\nu_\epsilon N_T$, or its continuous-time counterpart $\nu_{t,\epsilon}T$, becomes an integer number, then $\Psi(0, \nu_\epsilon) = 0$ and the correlation output in Eq. (1.65) collapses. This observation demonstrates that the coherent sum being done in Eq. (1.65) is strongly affected by the presence of residual frequency errors.

An example of ambiguity function is illustrated in Figure 1.18 to better show how this function looks like in practice.

The time domain is presented in continuous time through the code delay error $\tau_{t,\epsilon}$ in chips (i.e. $\tau_{t,\epsilon}/T_c$), while the frequency domain is presented in continuous time as well, through the frequency error $\nu_{t,\epsilon}$ in Hz. This is done for the sake of clarity because continuous-time magnitudes are easier to understand than discrete-time ones, which many times do not even have units (e.g. as the discrete-time frequency). Notwithstanding, one can move from one domain to the other by using the relationships in Eq. (1.60). The example in Figure 1.18 considers a GPS L1 C/A signal using a coherent integration time of $T = 1$ ms. One can see that the envelope on the time-domain axis is given by the autocorrelation of the chip pulse, as indicated in Eq. (1.70). GPS L1 C/A uses rectangular chip pulses thus giving rise to a triangular autocorrelation shape. On the frequency-domain axis, one can see that the envelope corresponds to the Fejér kernel in Eq. (1.71) having a sinc-like shape. Nulls appear when the time–frequency product $\nu_\epsilon N_T$, or equivalently $\nu_{t,\epsilon}T$, is an integer number. In this example where $T = 1$ ms is considered, it corresponds to integer multiples of $1/T = 1$ kHz. This suggests that the residual frequency error $\nu_{t,\epsilon}$ should be smaller than 1 kHz to prevent the correlation from collapsing. As a rule of thumb, typically one half this value is considered, so the receiver should make sure that $\nu_{t,\epsilon} \leq 1/2T = 500$ Hz. As mentioned

[23] Actually, the mean square bandwidth rather than the conventional bandwidth plays a role here, as discussed in detail in [5, Example 3.13].

[24] The result coincides with the Fejér kernel, which is the squared modulus of the Dirichlet kernel given by $\sin(\pi N x)/\sin(\pi x)$. The representation is very similar to that of the sinc function already defined as $\mathrm{sinc}(x) \doteq \sin(\pi x)/\pi x$, except for the fact that the Fejér and Dirichlet kernels are periodic functions of x, while the $\mathrm{sinc}(x)$ function is not.

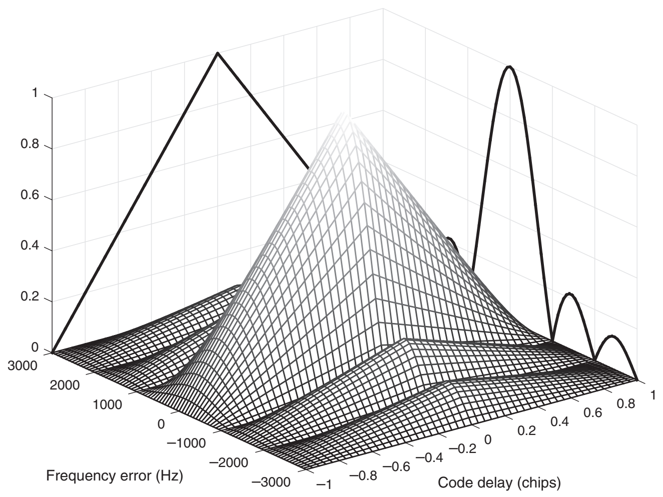


Figure 1.18 Ambiguity function normalized to unit amplitude for a GPS L1 C/A signal using $T = 1$ ms. The plot shows the square root of the ambiguity function to facilitate the visual identification of the triangular shape of the GPS L1 C/A correlation function. Infinite bandwidth is assumed.

earlier, the interplay between coherent integration time and residual frequency errors will be further discussed next when addressing the limitations of coherent integration.

It is worth mentioning that the result in Figure 1.18 assumes that the received signal has an infinite bandwidth and therefore the correlation with the local replica leads to a perfect triangle. In practice, GNSS receivers have a band-limited front end that cuts the received signal PSD down to a certain bandwidth. For example, for GPS L1 C/A signals, this bandwidth can be as low as 2 MHz. This provides significant computational savings at the expense of slightly degrading the sharpness of the correlation peak, and thus, slightly degrading the time-delay estimation accuracy. Some examples on the resulting correlation function using a band-limited GPS L1 C/A signal are shown in Figure 1.19. Two different bandwidths have been considered, $B = 2.046$ MHz (i.e. $B = 2R_c$) and $B = 4.092$ MHz (i.e. $B = 4R_c$). The bandwidth is left in general as $B = 2R_c$ and $B = 4R_c$ because the results for BPSK signals are the same regardless of the specific value of R_c, either it is $R_c = 1.023$ as in GPS L1 C/A or $R_c = 10.23$ MHz as in GPS L5. The correlation with infinite bandwidth is also shown in Figure 1.19 as a benchmark.

SNR Gain of Coherent Integration

The presence of a residual frequency shift has already been mentioned as a source of degradation for the coherent correlation in Eq. (1.65). Such a degradation can easily be assessed by observing the losses incurred in the ambiguity function in terms of ν_ϵ. For instance, when both the code and carrier replicas are perfectly aligned to the input signal, we have $\nu_\epsilon = 0$ and $\tau_\epsilon = 0$, and the ambiguity function becomes $\Psi(0, 0) = C$. The result does not depend on the coherent integration length because the coherent correlation $R_C(\tau', \nu')$ was already normalized by the integration length

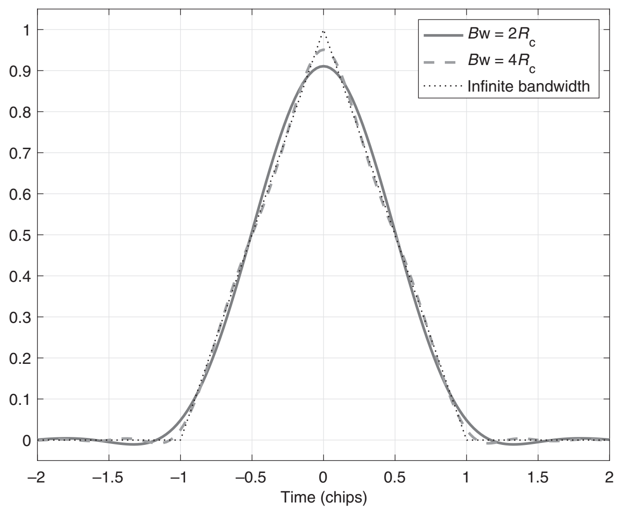


Figure 1.19 Correlation of a band-limited received GNSS signal using rectangular chip pulses at chip rate R_c with its ideal local replica. An example would be the GPS L1 C/A signal, which has $R_c = 1.023$ MHz, or the GPS L5 signal, which has $R_c = 10.23$ MHz.

N_T in its definition in Eq. (1.65). If we take the noise contribution into account, ignored so far, it is not difficult to see that the output noise power decreases as a function of N_T/N_{sc}, which is the total number of uncorrelated samples at the coherent correlation output.[25].

Based on these two results, we can compute the resulting SNR at the coherent correlation output. On the one hand, we have just mentioned that the signal power remains constant as a function of the coherent integration length N_T, according to the way that the coherent correlation output has been defined in Eq. (1.65), and as shown in $\Phi(0,0)$. On the other hand, we have that the noise power decreases linearly with N_T/N_{sc}. So merging both results together, we have that the SNR at the coherent correlation output increases linearly with N_T/N_{sc}, which is the number of uncorrelated samples being coherently integrated. For this result, we are assuming the optimistic case where the local replica is perfectly aligned. However, we can extend it for any τ_ϵ and ν_ϵ resulting in the following SNR gain at the coherent correlation output,

$$\Delta\text{SNR}_{\text{coh}}(\tau_\epsilon, \nu_\epsilon, N_T) = 20\log_{10}\left|\frac{\sin(\pi\nu_\epsilon N_T)}{\sin(\pi\nu_\epsilon)}\right| + 10\log_{10}\frac{N_T}{N_{\text{sc}}}, \qquad \text{(dB)} \qquad (1.72)$$

which is always contained within the range

$$0 \leq \Delta\text{SNR}_{\text{coh}}(\text{dB}) \leq 10\log_{10}\frac{N_T}{N_{\text{sc}}}. \qquad (1.73)$$

[25] This is because for a set of N uncorrelated random variables $x_0, x_1, \ldots, x_{N-1}$ with the same variance σ_x^2, the sample mean $\bar{x} = \frac{1}{N}\sum_{n=0}^{N-1} x_n$ has a variance $\sigma_{\bar{x}}^2 = \sigma_x^2/N$ [24, p. 188].

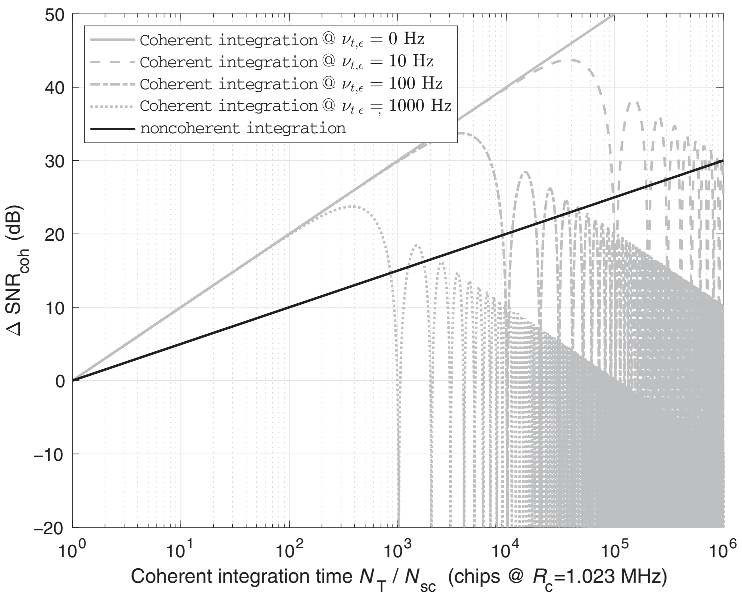


Figure 1.20 SNR gain at the coherent correlation output as a function of the integration time in units of chips, at $R_c = 1.023$ MHz.

An example of the SNR gain in Eq. (1.72) is shown in Figure 1.20 for different values of the frequency shift error. The SNR gain is shown as a function of the coherent integration length N_T normalized to the chip period N_{sc}, so that the result becomes the coherent integration length in units of chips. A chip rate of $R_c = 1.023$ MHz has been assumed in this example. The results in Figure 1.20 first show the ideal case when no frequency shift errors are present, so that $\nu_{t,\epsilon} = 0$. In this case, we can see how the SNR gain increases linearly without bound. Notwithstanding, when $\nu_{t,\epsilon} \neq 0$, the linear increase of the SNR gain is limited up to a point where it starts to decrease and it actually collapses. The collapse occurs whenever $\nu_{t,\epsilon}T$ or equivalently $\nu_\epsilon N_T$ are integer numbers, as previously discussed when introducing the ambiguity function. So for the case of $\nu_{t,\epsilon} = 1{,}000$ Hz, it means that the SNR first collapses for $T = 1/1{,}000 = 1$ ms, as confirmed in Figure 1.20. This integration length corresponds to 1,023 chips at the chip rate $R_c = 1.023$ MHz considered in Figure 1.20.

Finally, it is interesting to note that Figure 1.20 does also consider the case of noncoherent integration, where the accumulation in Eq. (1.65) would be carried out using the squared absolute value of each sample. This operation has no practical sense because we do need to integrate coherently at least for one code period in order to despread the received signal. But the curve in Figure 1.20 for noncoherent integration serves well for illustrating the advantage of coherent versus noncoherent integration. When integrating for 1,000 chips, one can see that there is a 30-dB gain with coherent integration, while just a 15-dB gain with the noncoherent one. So in practice there is a trade-off between coherent and noncoherent integration. The longer the coherent integration time, the higher the SNR gain but the smaller the residual frequency errors that can be tolerated. Noncoherent integration provides a means to compensate for the latter problem, but it comes at the expense of a worse capability to reject noise, as unveiled by the smaller SNR gain.

Constraints on the Coherent Integration Time

The minimum coherent integration time is usually set to one code period, so we have $T = T_{\text{code}}$ and thus $N_T = N_{\text{scode}}$ in Eq. (1.65). By doing so one is able to despread the received signal and to benefit from the orthogonality properties already discussed in Section 1.3.2. The maximum coherent integration time, though, could ideally be unbounded. Actually, the larger this value, the better the sensitivity of the receiver because more energy is accumulated. But one cannot let the coherent integration time increase without bound because of some limiting factors such as: (i) the presence of data modulating symbols; (ii) residual frequency errors still present in Eq. (1.65); and (iii) clock instabilities introduced by the front-end local oscillator.

The presence of data-modulating symbols restricts the coherent integration time down to the symbol period, so that $T \leq T_{\text{d}}$. Otherwise, integrating beyond this value may face the presence of symbol transitions that lead to the sum of destructive pieces of samples. So instead of having a gain by integrating over a longer time, we may end up having a severe signal loss whenever sign-reversed symbols are contained within the integration time. The only way to increase the coherent integration time beyond the symbol period is by knowing in advance what these symbols are, so one can compensate for them while doing the coherent accumulation in Eq. (1.65). This approach is actually the one that motivates the introduction of the so-called pilot or data-less components in modernized GNSS signals. These signals carry no data-modulated symbols, and thus, they are completely known to the receiver by the time they are received. It is for this reason that once the starting point of both the primary and secondary code is found, the rest of the signal is known and thus long coherent integration times can be implemented without the risk of suffering from unexpected symbol transitions. This involves, however, determining the starting point of the secondary code present in the pilot signal. The extension of Eq. (1.65) to cover this case will be briefly discussed next, but the interested reader can find more details in [4, Section 16.4].

The second limiting factor for the coherent integration time is the presence of residual frequency errors, which appear when wiping off the input carrier at frequency ν with the local replica at the tentative frequency ν'. Because of this frequency mismatch, the resulting signal still has some residual frequency error $\nu_\epsilon = \nu - \nu'$ that causes the ambiguity function in Eq. (1.69) to collapse whenever $\nu_\epsilon N_T$, or $\nu_{t,\epsilon} T$ in continuous time notation, is an integer number. This effect was already discussed in the previous section, and it could easily be seen in Figure 1.18 and also as a degradation of the SNR gain in Figure 1.20. The rule of thumb to deal with residual frequency errors is to limit the coherent integration time up to $T \leq \frac{1}{2|\nu_{t,\epsilon}|}$. This condition can be achieved by two different means: either by shortening T for a fixed frequency error $\nu_{t,\epsilon}$ or by reducing $\nu_{t,\epsilon}$ for a fixed coherent integration time T. The latter approach is the one usually considered in practice because T is a design parameter that is set to meet certain performance requirements, such as the receiver sensitivity. Once T is set, the acquisition stage must be designed so that the frequency search is fine enough to ensure that the remaining $\nu_{t,\epsilon}$ is compliant with the aforementioned condition.

Finally, the third limiting factor to deal with is the clock stability. This is often measured through the Allan variance $\sigma^2(T_\text{A})$, a dimensionless measure that provides the variance of the random frequency fluctuations of the clock over a time period T_A [25]. When the Allan variance is referred to a given carrier frequency F_c in Hz, one can obtain the one-sigma frequency deviation of the clock as $\sigma_\text{A}(T)F_\text{c}$ in Hz. The coherent integration of this fluctuating frequency error results in an effect similar to that of the residual frequency error just mentioned above, causing the coherent integration in Eq. (1.65) to collapse when T is too long. Applying a similar principle, one should make sure that $\sigma_\text{A}(T)F_\text{c} \ll \frac{1}{2T}$ to minimize the impact of clock instabilities.

The three conditions to be fulfilled for setting the correlation time T are therefore:

$$T \leq T_\text{d}, \qquad T \leq \frac{1}{2|\nu_{t,\epsilon}|}, \qquad T \ll \frac{1}{2\sigma_\text{A}(T)F_\text{c}}. \tag{1.74}$$

The first condition depends on the GNSS signal, and there is nothing the receiver can do unless it has access to the navigation data symbols via an alternative means (e.g. using an assistance GNSS server), or a pilot component is processed. In contrast, the second and third conditions depend on the receiver implementation and can thus be improved at the expense of increasing complexity and cost. Complexity is due to the need to make $\nu_{t,\epsilon}$ small, thus increasing the number of tentative values for ν' covering the same frequency range. Cost is due to the need to replace the clock with a better one, in case the stability does not allow for a long enough correlation. To have an idea of the order of magnitude, conventional temperature-controlled crystal oscillators allow T to be up to one hundred milliseconds, chip-scale atomic clocks (CSAC) up to a few hundred milliseconds and oven-controlled crystal oscillators (OCXO) up to one second [26]. Further details on the clock at the RF front end are provided at the end of Section 11.2.2.

Noncoherent Extension of the Coherent Integration Time

It now becomes clear that the coherent integration time cannot be increased without bound due to the presence of symbol transitions, residual frequency errors and clock instabilities that hinder the integration in Eq. (1.65). Among all of them, symbol transitions are often the most limiting factor because it is difficult for a standalone receiver to know in advance these symbols, so that they could be compensated. Assuming that symbol alignment has already been achieved at the receiver, the symbol period leaves just a few milliseconds of margin to safely perform coherent integration without the risk of experiencing a sign transition. For instance, the symbol period for GPS L1 C/A is just 20 ms long, which provides a sensitivity of ~35 dBHz of C/N_0. This value may not be enough for receivers operating in urban canyons, soft indoor environments or even in Space, so extending the integration time beyond the symbol period becomes a real need. In essence, the longer the correlation time, the larger the energy that is collected, and thus, the weaker the signals that can be detected.

The solution to this problem is to extend the correlation time by noncoherently accumulating N_{I} coherent correlations [12, 27]. The extended noncoherent correlation can thus be computed as follows:

$$R_{\mathrm{NC}}(\tau', \nu') = \sum_{k=0}^{N_{\mathrm{I}}-1} \left| R_{\mathrm{C}}(\tau', \nu'; k) \right|^2, \tag{1.75}$$

where the k-th coherent correlation $R_{\mathrm{C}}(\tau', \nu'; k)$ is derived from Eq. (1.65) as

$$R_{\mathrm{C}}(\tau', \nu'; k) = \sum_{n=0}^{N_T-1} r[n + kN_T] c^*[n - \tau']_{N_{\mathrm{scode}}} e^{-j2\pi\nu' n}. \tag{1.76}$$

Note that the total correlation time in Eq. (1.75) is now:

$$T_{\mathrm{tot}} = N_{\mathrm{I}} T \quad \text{(seconds)}, \quad N_{\mathrm{tot}} = N_{\mathrm{I}} N_T \quad \text{(samples)}. \tag{1.77}$$

The problem with noncoherent integration is that it is not as effective as the coherent one when it comes to mitigating the noise. This can be seen in Figure 1.20 where the solid black line shows the SNR gain when performing the full integration noncoherently. This will never be the case in practice because we need to coherently correlate at least one code period, so that we can despread the code in the received signal. But the example illustrates well how less efficient noncoherent integration is. While coherent integration has an SNR gain growing linearly with the integration time, noncoherent integration has an SNR gain that barely grows with the square root of the integration time. It is for this reason that one should always try to use the longest possible coherent integration time before noncoherent integration comes into action. At least, when sensitivity is being targeted. For accuracy purposes, this is not the case because the only important parameter for precise time-delay estimation is the total integration time, no matter how it is split into coherent or noncoherent integrations. But for detecting the satellites with the best possible sensitivity, one should always set the longest possible coherent integration time and then extend this time if needed, noncoherently.

It now becomes clear that the receiver design must face a trade-off between coherent and noncoherent integration at the correlation implemented by the acquisition stage. On the one hand, the longer the coherent integration, the better the noise rejection (i.e. the higher the receiver sensitivity), but the more sensitive the correlation is to residual frequency shifts. On the other hand, using short coherent integrations relaxes the requirement on the frequency search, because larger residual frequency shifts are tolerated. But this comes at the expense of a poor noise rejection and thus a poor sensitivity, so the receiver will need to operate in better sky visibility conditions than when long coherent integrations are used.

In practice one needs to carefully assess the receiver requirements and the implementation constraints in order to find the proper combination of coherent integration time and number of noncoherent integrations. This is because for the same total integration time T_{tot}, the performance can be quite different. To demonstrate this statement, Figure 1.21 shows an example for the sensitivity or minimum C/N_0 that would be required for detecting a GPS L1 C/A signal with a relatively high confidence (probability of detection $P_{\mathrm{d}} = 0.9$ and probability of false alarm $P_{\mathrm{FA}} = 10^{-4}$). Details on

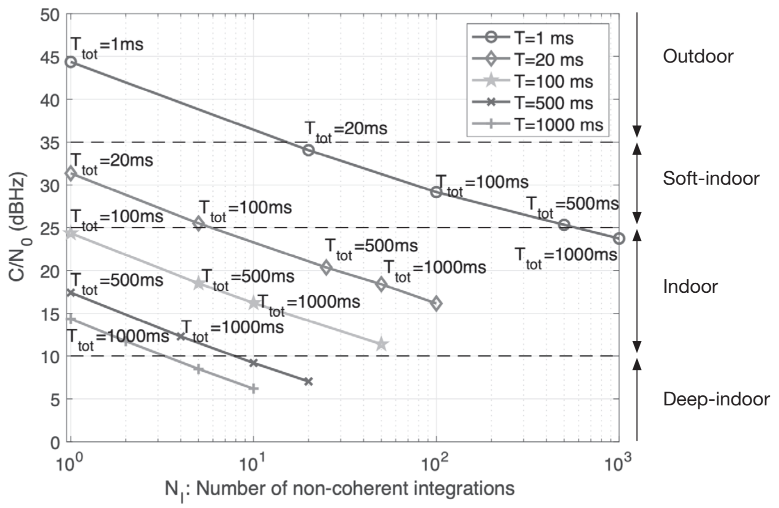


Figure 1.21 Minimum required C/N_0 (i.e. sensitivity) to detect a GPS L1 C/A signal with probability of detection $P_{\text{d}} = 0.9$ and probability of false alarm $P_{\text{FA}} = 10^{-4}$ for different combinations of coherent and noncoherent integrations. It is interesting to observe how the same total integration time T_{tot} provides different sensitivities when different combinations of T and N_{I} are used.

how to obtain these metrics will be discussed later in step 3 of the acquisition stage and with more detail in Section 9.2.4.

The results in Figure 1.21 show, for instance, that an integration time of 20 ms has ~3 dB better sensitivity when it is performed as a single coherent integration rather than as a noncoherent integration of 20 coherent integrations of 1 ms each. The difference is even larger when we move to longer integration times. For example, a single coherent integration of 1,000 ms length has ~10 dB better sensitivity than when the same integration is carried out by noncoherently accumulating 1,000 coherent integrations of 1 ms each. These results are illustrative of how different the sensitivity can be for the same total correlation time T_{tot}, when different combinations of T and N_{I} are set.

Finally, it is worth mentioning that based on the results shown so far, one may think that noncoherent integration is the answer to let the integration time increase without bound. It solves the presence of symbol transitions by removing the sign changes and it solves the clock issues by letting the coherent integration be performed on a short enough interval, short enough to prevent clocks instabilities to significantly degrade the coherent correlation output. But despite these advantages, the total integration cannot be increased without bound either. The last impairment to be coped with is actually the first one already introduced in Eq. (1.52), namely the code Doppler. It causes the received signal to be compressed or expanded depending on the experienced Doppler shift, thus causing the received spreading code not to match with the local replica being used at the receiver. Actually, the Doppler shift is still unknown at this stage of the receiver, so we cannot compensate for it beforehand. Assuming that the frequency search is done in steps of $\Delta\nu'$ width, the maximum residual frequency shift is then given by half this bin width, $\Delta\nu'/2$. So similarly to code Doppler contribution in Eq. (1.52), the residual frequency shift will introduce an additional time delay of $\delta\tau_t = \frac{\Delta\nu'_t/2}{F_c} T_{\text{tot}}$ seconds when the signal is observed over an interval of T_{tot} seconds long. If such additional time delay wants to be kept smaller than a given maximum value, $\delta\tau_{\max,t}$, the total correlation time must then be limited to

$$T_{\text{tot}} \leq 2\delta\tau_{\max,t} \frac{F_c}{\Delta\nu'_t} \tag{1.78}$$

which becomes the upper bound on the maximum total correlation time that can be used by the receiver.

Correlation in the Presence of Data Bits or Secondary Code

The coherent correlation in Eq. (1.65), which is also the basis for the noncoherent correlation in Eq. (1.75), was derived under the assumption that no data symbols or secondary code. If any of them are present, then the starting point of the symbol transition or the starting point of the secondary code must be searched as well. For instance, the symbol period in the GPS L1 C/A signal is 20 ms long, and it is composed of the repetition of $N_r = 20$ spreading codes with 1-ms duration each. Determining the starting point of the symbol period allows the receiver to implement a safe coherent integration up to $T = 20$ ms, without the risk of experiencing sign transitions in-between. Symbol or bit synchronization is often implemented by collecting the correlation peaks every 1 ms and then checking their signs to confirm whether they belong or not to the bit pattern of 20 consecutive equal signs. Variations of this approach can be found in [28, 29], and an implementation using a sliding window is in Section 9.2.8.

Pilot signals do not carry data-modulating symbols but incur a similar problem because they are implementing a modulated sequence on top of the spreading code, which is actually the secondary code. Even though the secondary code sequence is known, the starting point of this sequence in the received signal is unknown to the receiver. So eventually, the problem is the same as determining the data-modulated symbol transition, except for the fact that we are now searching for a known pattern of alternating signs. It is for this reason that many of the previous methods for symbol synchronization can be adapted for secondary code synchronization as well, just by changing the repetition pattern. The interested reader will find a detailed discussion on the topic in [11, Section 14.3.6] and [13, Section 16.5].

Alternatively, one could decide not to search for the symbol transition and proceed with the coherent integration in Eq. (1.65) using the received signal samples straightaway. In that case, there is a certain probability of experiencing a symbol transition somewhere within the coherent integration period. The symbol transition will split this period into two blocks that will add destructively in Eq. (1.65). The result can either collapse when the two blocks have equal size or barely suffer any degradation when one block is much larger than the other. In the end, the presence of symbol transitions can be modeled as a loss in signal power. This problem was analyzed in [30] where it was shown that the average losses for a total integration time $T_{\text{tot}} = TN_I$ amount at

$$L_{\text{bit,mean}} = -10\log_{10}\left(1-\frac{T}{3T_{\text{d}}}\right), \qquad \text{(dB)} \tag{1.79}$$

with T_{d} the symbol period, while the worst-case losses can reach up to

$$L_{\text{bit,max}} = -10\log_{10}\left(1-\frac{T}{T_{\text{d}}}\right). \qquad \text{(dB)} \tag{1.80}$$

These results are valid for any total integration time $T_{\text{tot}} \geq T_{\text{d}}$ provided that the coherent integration time is $T < T_{\text{d}}$. As can be seen, the results do not depend on the number of noncoherent integrations but just on the coherent integration time. The worst-case losses asymptotically increase as $T \to T_{\text{d}}$, but for a shorter coherent integration time of just 10 or 5 ms, the maximum losses are limited down to 3 or 1.25 dB, respectively. This means that in the absence of symbol synchronization, it is convenient to use a shorter coherent integration time T in order to keep the losses due to such transitions under control. This in turn will incur a sensitivity loss, but it can be compensated up to a great extent by increasing N_{I} and thus having a longer total integration time.

Otherwise, one always has the option of trying to detect the sign of the spreading codes present in the coherent integration period T. This would be like understanding spreading codes as if they were data-modulated symbols, and trying to detect and compensate their sign before adding them all together. The results in Figure 1.12 indicate that this is feasible even for spreading codes of 1 ms provided that the C/N_0 is above 35 dB-Hz, leading to a probability of error smaller than 10^{-2}.

1.6.2 Signal Detection

So far it has been considered that the noncoherent correlation $R_{\text{NC}}(\tau',\nu')$ constitutes the basis for deciding whether the local replica is correctly aligned to the received signal or not. As a by-product, it serves for deciding whether the satellite being searched is present or not, as well. These two possible hypotheses are often denoted as $\mathcal{H}_1$, that is, satellite present and well-aligned replica, and $\mathcal{H}_0$, that is, satellite absent or badly aligned replica. Different variations on $R_{\text{NC}}(\tau',\nu')$ can alternatively be used for this purpose. For instance, by using the absolute value or a fractional exponent instead of the square shown in Eq. (1.75) [31]. In order to formulate the signal detection problem, let us in general denote by Z_i the resulting metric, using either Eq. (1.75) or any other variant. Let us also assume that the search space is composed of $N_{\tau'}$ bins of width $\Delta\tau'$ in the time axis, and $N_{\nu'}$ bins of width $\Delta\nu'$ in the frequency axis. So we have a total of $N_{\tau'}N_{\nu'}$ bins to test, and thus Z_i spans for $i = 1,\ldots,N_{\tau'}N_{\nu'}$. Because of the bi-dimensional time–frequency search, it is often customary to store all the resulting Z_i values in a matrix with dimensions $(N_{\tau'} \times N_{\nu'})$. Each Z_i value is therefore often referred to as a test *cell*. Finally, the decision test statistic is formed through some function $\Phi(\cdot)$ of all the Z_i results,

$$Z = \Phi(Z_1, Z_2, \ldots, Z_{N_{\tau'}N_{\nu'}}). \tag{1.81}$$

Since the Z_i values are ultimately based on the correlation between the received signal and a tentative local replica, it is not difficult to see that a large value will be

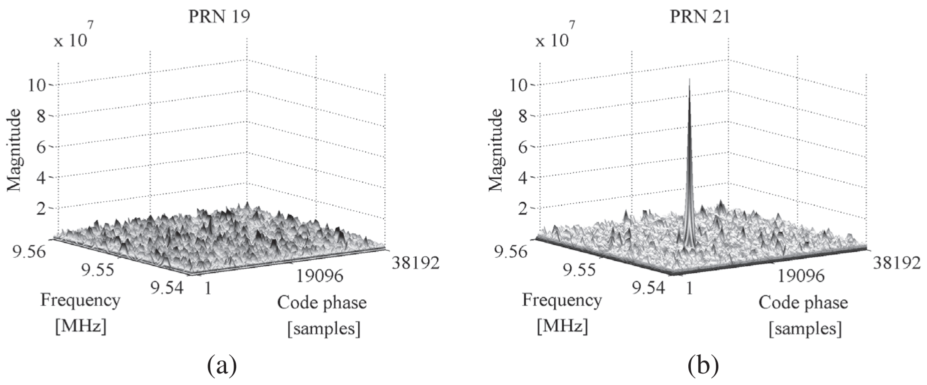


Figure 1.22 Sample acquisition output (a) PRN 19 is not visible, so no peak is present. (b) PRN 21 is visible, so a significant peak is present. The peak is at code delay 13404 samples and frequency 9.5475 MHz.

obtained whenever the local replica is well-enough aligned to the received signal. So finding the best tentative local replica, and thus the best pair of (τ', ν') values, boils down to finding which of the Z_i values is large enough. It is for this reason that the function $\Phi(\cdot)$ can be the maximum of its arguments, the ratio between the largest value and the second largest value separated apart by a certain interval, or any other criterion that may help in finding the correct cell. In the end, the procedure consists of comparing the decision test statistic Z with a given value or threshold γ. That is,

$$Z \underset{\mathcal{H}_0}{\overset{\mathcal{H}_1}{\gtrless}} \gamma, \tag{1.82}$$

being the outcome of this comparison, the answer to the decision question on whether either $\mathcal{H}_1$ or $\mathcal{H}_0$ is the correct hypothesis. An example of the $(N_{\tau'} \times N_{\nu'})$ matrix containing all Z_i values under analysis is shown in Figure 1.22. On the left-hand side, we can see the case when the satellite signal of interest is not present, so all cells contain noise only. On the right-hand side, we can see the case when the signal is present, observing a cell having a much larger value than the rest of cells in the search space.

As mentioned previously, finding the correct cell provides twofold information. First, that the tentative local replica is well-enough aligned to the received signal. Second, as an obvious consequence of the former, that the satellite being searched is actually present in the received signal. In that case, the acquisition is said to be *successful* and the tracking stage is then initialized with the corresponding pair of (τ', ν') as coarse estimates of the code delay and frequency shift. So we can declare that a given satellite is present, while at the same time we can provide rough estimates on the time delay and frequency shift values. The decision may be correct or wrong, but in the way the threshold γ is designed, the probability of correct decision and the probability of false alarm are both under control.

The test statistic can be based on either a data or a pilot signal component, or a combination of both data and pilot components. The combining methods can be either noncoherent or coherent. Few research papers have addressed the issue on how to combine the data and pilot components in forming the decision statistic [32, 33, 34]. The general approach is the noncoherent one where the correlation outputs from the data and pilot components are squared and then accumulated in order to form the decision variable. Alternatively, coherent combining methods are proposed in [32] for both E1 and E5 Galileo signals. The main idea of coherent combining methods is to add and subtract the pilot and data components to form two decision variables Z_{p} and Z_{d}, and then form a decision variable based on the maximum between the two outputs. According to the results reported in [32], the coherent combining scheme slightly outperforms the noncoherent one, but only if the signal level is strong enough. At moderate and low signal levels, the noncoherent and coherent combinations show similar performance.

The problem of detecting a signal in noise has been extensively presented in [35], where the reader is referred to for further information on the underlying principles. The application to GNSS involves some additional considerations because multiple test cells, on the order of thousands, need to be evaluated and this requires some search strategy planning, as discussed in the next step on the search process. At high level, signal detection in GNSS can be performed either in parallel by evaluating all the testing cells at the same time, that is, as indicated in Eqs. (1.81)–(1.82), or sequentially by evaluating one test cell at a time. In the latter case, one moves from one test cell to the next until detection is declared successful. When this occurs, the search stops and acquisition is declared successful.

The search strategy has an impact on the decision rule in Eq. (1.81) because the statistics of a large set of values are different from the statistics of just a single value. Parallel detection involves evaluating all the testing cells simultaneously. It is often implemented in software receivers because it fits well with the use of the FFT. It will be analyzed in detail in Sections 1.6.3 and 9.2.4 for snapshot receivers. The alternative is sequential detection, which involves evaluating a single cell at a time. The evaluation of the decision rule in Eq. (1.81) can be done in a single stage or in multiple stages with various coherent and noncoherent integration times. The single-stage detection is known as single-dwell detection, see [36, 37]. Multiple-stage detection is also known as multidwell detection [36, 38, 39], where we gain more confidence on the presence or absence of the satellite under analysis as we move on to subsequent stages. In this way, we do not need to wait until the whole signal is processed, but rather take partial decisions dwell by dwell. By doing so we can stop whenever we are confident enough on the decision to be made, or continue to the next dwell in case we are not confident enough yet and would like to double check. The concept of multidwell detection is illustrated in Figure 1.23. Each dwell is characterized by the so-called *dwell time*, which is simply the integration time used in the correlation of that dwell. Then a decision statistic $Z_{i,k}$ is formed at the k-th dwell for the i-th time–frequency bin, and the result is compared with a threshold γ_k. If the decision statistic is larger than the threshold, we move to the next $(k+1)$-th dwell, with a larger dwell time in order to cross-check the decision just taken. At each dwell or decision stage, the same steps are undertaken. The final delay and frequency estimates are obtained after the last

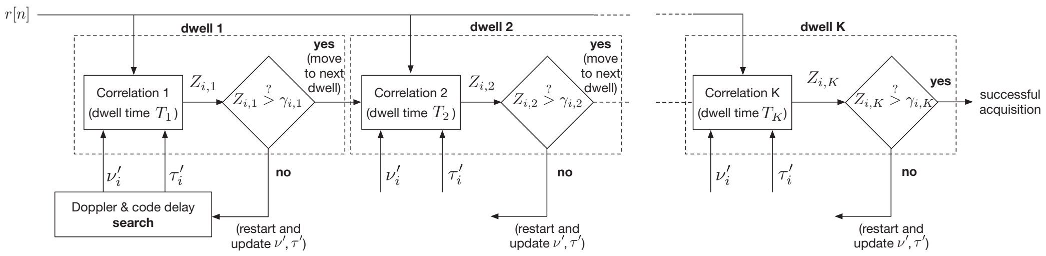

$r[n]$
dwell 1
dwell 2
dwell K
Correlation 1 (dwell time T_1)
Correlation 2 (dwell time T_2)
Correlation K (dwell time T_K)
$Z_{i,1}$
$Z_{i,2}$
$Z_{i,K}$
$Z_{i,1} \overset{?}{>} \gamma_{i,1}$
$Z_{i,2} \overset{?}{>} \gamma_{i,2}$
$Z_{i,K} \overset{?}{>} \gamma_{i,K}$
yes (move to next dwell)
yes
successful acquisition
no
(restart and update ν', τ')
ν_i'
τ_i'
Doppler & code delay search

Figure 1.23 Multidwell structure for the acquisition of GNSS signals. Detail showing the dwells for the i-th testing cell using the pair of tentative values (ν_i', τ_i').

dwell. Ideally, the larger the number of dwells, the better the trade-off between a high detection probability and a small false alarm probability we can get, though this might not always be the case [40]. Moreover, a higher number of dwells means an increased complexity of the acquisition stage.

Coming back to the global decision rule in Eq. (1.81), the probability of detection is obtained as

$$P_{\mathrm{D}}(\gamma) = \int_{\gamma}^{\infty} f(Z|\mathcal{H}_1)dZ, \tag{1.83}$$

whereas the probability of false alarm, that is, the probability of declaring acquisition successful when actually no signal but only noise was present, is obtained as

$$P_{\mathrm{FA}}(\gamma) = \int_{\gamma}^{\infty} f(Z|\mathcal{H}_0)dZ, \tag{1.84}$$

with $f(Z|\mathcal{H}_m)$ the probability density function (PDF) of Z under hypotheses $\mathcal{H}_m$, where $m = \{0, 1\}$. Note that both the probability of detection and the probability of false alarm have been expressed as a function of the detection threshold γ, which is the parameter that allows us to adjust these probabilities to match the receiver requirements. In case of sequential detection using multiple dwells, the decision rule at each individual dwell is characterized by the corresponding probability of detection $P_{\mathrm{d},k}(\gamma_k) = \int_{\gamma_k}^{\infty} f(Z_k|\mathcal{H}_1)dZ_k$ and probability of false alarm $P_{\mathrm{fa},k}(\gamma_k) = \int_{\gamma_k}^{\infty} f(Z_k|\mathcal{H}_0)dZ_k$, for $k = 1, \ldots, K$. Based on these individual probabilities, the global detection and false alarm probabilities are computed as $P_{\mathrm{D}}(\gamma_k) = \prod_{k=1}^{K} P_{\mathrm{d},k}(\gamma_k)$ and $P_{\mathrm{FA}}(\gamma_k) = \prod_{k=1}^{K} P_{\mathrm{fa},k}(\gamma_k)$, respectively. Multidwell structures have not been so extensively studied in GNSS, even though some references on the topic can be found in [41, 42, 43].

For the detection problem in Eq. (1.81), the common practice is to be given a requirement in terms of P_{D} for a given working condition (i.e. at some minimum C/N_0) and in terms of P_{FA}. Based on these inputs, we will need to tune the detection stage so that these probabilities are ultimately achieved. The tuning is done by adjusting the detection threshold γ in Eqs. (1.83)–(1.84) accordingly.

The relationship between the probability of detection, probability of false alarm and the detection threshold is schematically illustrated in Figure 1.24. For instance, the probability of false alarm is the probability of declaring acquisition successful when

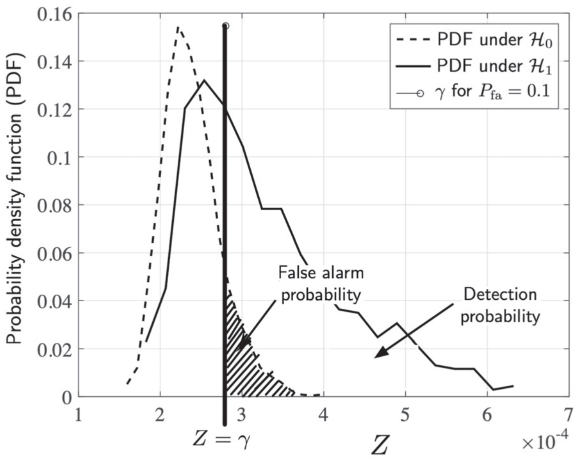


Figure 1.24 Threshold choice at the acquisition stage. $\mathcal{H}_0$ is the null hypothesis (no signal), and $\mathcal{H}_1$ is the alternative hypothesis (signal present).

it is actually not. This corresponds to the integral in Eq. (1.84), which is graphically illustrated in Figure 1.24 by the area below the right tail of the PDF under $\mathcal{H}_0$, beyond the point $Z = \gamma$. Since the PDF under $\mathcal{H}_0$ does only depend on the noise statistics, and not on the signal one, the probability of false alarm is directly related to the detection threshold γ. So once $P_{\rm FA}$ is provided as a requirement to be met, we can obtain γ from either Eq. (1.84), in case an analytical expression is available, or from the histogram of the empirical PDF for $\mathcal{H}_0$ by moving γ until the shaded area in Figure 1.24 coincides with $P_{\rm FA}$. Once the detection threshold has been set, then we need to make sure that for the minimum C/N_0 that has been specified, the area below the right tail of the PDF under $\mathcal{H}_1$ is greater than or equal to the specified $P_{\rm D}$. Finally, it can easily be seen in Figure 1.24 that the larger the detection threshold γ, the smaller the false alarm probability, which is good, but the smaller as well the detection probability, so a trade-off needs to be found.

A performance criterion for the acquisition stage is the mean acquisition time (MAT), which refers to the average time needed to be spent in the acquisition stage until the delay and frequency estimates are obtained [44]. Naturally, different combinations between search and detection mechanisms will provide different MAT. Alternative performance criteria can also be found in [45]. Typically, the trade-off is between the complexity of the acquisition unit (number of parallel correlators) and the acquisition times. Generic methods to compute MAT for DS-SS signals can be found in [37, 38]. For Galileo, the MAT values have been analyzed in [39, 43, 46, 47]. Some techniques to improve MAT both at the search and at the detection stages have been discussed in [48].

More details on the process of detecting a GNSS signal will be discussed in Chapter 9 when addressing snapshot receiver implementations.

1.6.3 Time–Frequency Search

This step is in charge of evaluating the correlation $R_{\mathrm{NC}}(\tau', \nu')$ for all tentative values of time delay and frequency shift within the search space. As mentioned previously, the search space is composed of $N_{\tau'}$ bins in the time domain with resolution or bin width $\Delta\tau'$, and $N_{\nu'}$ bins in the frequency domain with resolution or $\Delta\nu'$. The total search space is thus composed of $N_{\tau'}N_{\nu'}$ bins.

Time Search

Regarding the time-delay search, the received signal comes with samples equispaced every T_s seconds, as given by the sampling rate. The local code replica is thus sampled at the same rate in order to perform the correlation. Despite the one sample time resolution, it is important to keep in mind that the time resolution of the correlation output, in terms of τ', does not depend on the sampling rate but on the resolution of the actual τ' used for generating the local replica. This resolution, $\Delta\tau'$, can actually be as fine as desired. However, it has the disadvantage that, by doing so, the number of tentative time-delay bins $N_{\tau'}$ dramatically increases. In practice it is often preferred to use a conservative time-delay resolution, typically half the one-sided width of the chip autocorrelation peak. By doing so one can trade off with the number of tentative delay bins that are required to explore. It is interesting to note that once acquisition is successful and the correlation peak of the received signal is found, sub-sample accuracy can always be obtained by interpolating the samples in the neighborhood of the maximum peak [49] or by recomputing the correlation peak once an estimate of the frequency shift is available as well [50]. In the end, one should keep in mind that the accuracy of time-delay estimation does only depend on the received signal bandwidth and the received C/N_0, but not on the sampling rate of the receiver, provided that it fulfills the Nyquist criterion. The impact of the time–frequency resolution in terms of C/N_0 losses at the correlation output is thoroughly discussed in [13, Section 16.2.2].

So in practice, the time-delay resolution $\Delta\tau'$ is required to be smaller than the one-sided width of the chip pulse autocorrelation peak. This, in turn, determines the minimum sampling rate needed. Indicative values for $\Delta\tau$ are provided in Table 1.2 for some representative GNSS signals.

The impact of the time-delay mismatch between the local replica and the received signal is shown in Figure 1.25(a). The results are shown in terms of the incurred loss in signal power. A rectangular chip pulse with duration equal to the chip period, such

Table 1.2 Maximum time delay resolution $\Delta\tau'$ at the acquisition search step, for some representative GNSS signals.

Modulation	One-sided width of the chip pulse autocorrelation main peak [chips]	Time-delay resolution $\Delta\tau'$ of the acquisition search [chips]
BPSK (any order)	1	0.5
BOC(1,1)	0.34	0.175
sinBOC(10,5)	0.14	0.07
cosBOC(10,5)	0.11	0.055

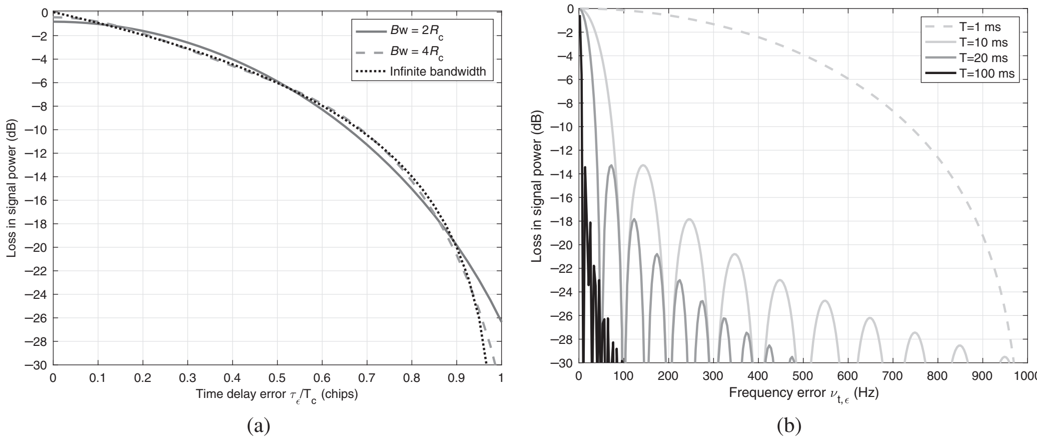


Figure 1.25 Losses in signal power due to either (a) a time mismatch or (b) a frequency mismatch in the correlation between the received signal and the local replica. A rectangular chip pulse with duration equation to the chip period $T_c = 1/R_c$ is considered.

as in GPS L1 C/A, is considered. One can verify that when the local replica is offset by the half chip period with respect to the received signal, the correlation output corresponds to the mid-point of one side of the triangular peak. So being the mid-point, its amplitude should be one half of that of the peak, thus incurring in a 6-dB loss in signal power. This is actually the value observed in Figure 1.25 for a time-delay error of 0.5 chip. Now let us come back to the results in Table 1.2. The time-delay resolution $\Delta\tau'$ for the time-delay search was mentioned to be (at least) half of the one-sided chip pulse autocorrelation peak width. Therefore, for the example at hand, this means using at least $\Delta\tau'/N_{sc} = \Delta\tau'_t/T_c = 0.5$. That is, two samples per chip. On the other hand, searching in steps of $\Delta\tau'$ incurs a maximum time delay error of $\tau_{\epsilon,\max} = \Delta\tau'/2$. This translates into a maximum time delay error of $\tau_{\epsilon,\max}/N_{sc} = \tau_{t,\epsilon,\max}/T_c = 0.25$ chip for this example, which is found to incur a ~2-dB loss according to Figure 1.25. The conclusion is that in the worst case, the losses due to the bin size of the time delay search amount up to ~2 dB. Finally, once the time resolution is known, the time-delay search spanning within one code period of N_{scode} samples results in a total of $N_{\tau'}$ time bins where the correlation peak may appear,

$$N_{\tau'} = \left\lfloor \frac{N_{\text{scode}}}{\Delta\tau'} \right\rfloor. \tag{1.85}$$

Frequency Search

Regarding the frequency search, it is interesting to review first the three main elements contributing to the frequency shift in the received signal. The first one is the satellite motion, which amounts to a maximum of ±5 kHz frequency shift, as per GPS satellites at L1.[26] The second one is the frequency offset of the receiver oscillator, which

[26] The frequency shift of other GNSS satellites in ordinary MEO orbits is similar, but this is not the case for satellites in IGSO orbits, where Doppler shift is approximately half of that of MEO orbits; GEO orbits,

adds ±1.5 kHz per each ppm (part per million) of frequency stability when observed at L1. The third one is the user's velocity, which adds ±1.5 Hz per each 1 km/h of speed [12, Chapter 3].

When no a-priori knowledge is available and an oscillator with 1 ppm frequency stability is used, the acquisition stage should account for a maximum frequency search of $\nu_{t,\max} = 6.5$ kHz. If the frequency offset of the receiver oscillator was known, for instance, by some prior calibration[27] means, then the maximum frequency search could be reduced down to $\nu_{t,\max} = 5$ kHz. If on top of the oscillator calibration we also had a-priori information on the satellite Doppler, for instance, thanks to the use of assistance information, then there would be no need to perform the coarse frequency search. In that case, the only remaining frequency shift would be due to the user's dynamics, and this can normally be accommodated within the span of a single frequency bin, which reaches up to $\pm\Delta\nu'_t/2$.

While the maximum span of the frequency search is now clear, it remains to determine the actual bin size $\Delta\nu'$. We will use the continuous-time notation $\Delta\nu'_t$ instead, because it is a more intuitive metric than its discrete-time counterpart $\Delta\nu'$, which is actually unitless. This will facilitate the discussion hereafter.

Some details on the tolerable frequency error were already unveiled when introducing the coherent correlation in Eq. (1.65), which was found to collapse whenever $\nu_{t,\epsilon}T = 1$ or multiples of it. The effect was clearly visible in the ambiguity function in Eq. (1.69) and shown therein in Figure 1.18. As a rule of thumb, the maximum frequency error was set at $\nu_{t,\epsilon,\max} = 1/2T$ with T the coherent integration time. This would correspond to a frequency search using a bin width of $\Delta\nu'_t = 1/T$, assuming that the errors are uniformly distributed within this width. The impact of having such frequency error can now be seen in Figure 1.25(b). For a coherent integraton time of $T = 1$ ms, the frequency bin width would be 1 kHz and the maximum frequency error would then become 500 Hz. Such error incurs a ~4-dB loss in signal power as observed in Figure 1.25, which may be too high to afford in certain working conditions. A more conservative approach is to set the frequency bin width to

$$\Delta\nu'_t = 1/2T, \tag{1.86}$$

which then leads to a maximum frequency error,

$$\nu_{t,\epsilon,\max} = 1/4T. \tag{1.87}$$

For the same example above, this now would lead to a frequency search bin width of $\Delta\nu'_t = 500$ Hz, a maximum frequency error of 250 Hz, and just a ~1-dB loss in signal power. Finally, once the frequency resolution is known, the frequency search comprised in the range within $\pm\nu_{t,\max}$ is then divided into $N_{\nu'}$ frequency bins to test, where

which is much lower; or the eccentric Galileo eccentric satellites (E18 and E14), which reach a higher Doppler shift than standard MEO satellites.

[27] Note that such calibration may need to be repeated from time to time to account for temperature variations and ageing of the oscillator.

$$N_{\nu'} = \left\lfloor \frac{2\nu_{t,\max}}{\Delta\nu'_t} \right\rfloor + 1 \tag{1.88}$$

or equivalently using the frequency bin width in Eq. (1.86),

$$N_{\nu'} = \lfloor 4T\nu_{t,\max} \rfloor + 1. \tag{1.89}$$

The results in Figure 1.25(b) provide many insights on the interplay between coherent integration time and maximum tolerable frequency error at the correlation stage. One can see how increasing the coherent integration time, as required in high-sensitivity applications, places stringent demands on the frequency search, since the residual frequency error must be decreased accordingly. For instance, let us assume that we are fine with the ~4-dB loss of signal power when using $T = 1$ ms and a frequency bin width of $\Delta\nu'_t = 1$ kHz. Let us now assume that we would like to preserve the same loss but when implementing a long coherent integration time of $T = 100$ ms, as it could be needed for instance in some high-sensitivity applications. The 4-dB loss for $T = 100$ ms is found in Figure 1.25 to be obtained for a frequency error of just 5 Hz, which implies a frequency search with a bin width of $\Delta\nu'_t = 10$ Hz. This is an extremely demanding requirement. To make it clear, let us assume that the receiver at hand is operated in standalone mode, so assistance information is not available. According to the discussion above, the frequency search should span at least for $\nu'_t \in [-\nu_{t,\max}, +\nu_{t,\max}] = [-6.5, +6.5]$ kHz. Covering this range in steps of $\Delta\nu'_t = 10$ Hz leads to $N_{\nu'} = 1{,}301$ frequency bins to evaluate. On the time-delay search, assuming a spreading code with 1,023 chip and thus a time delay bin width of $\Delta\tau'_t/T_c = 0.5$ chip, we would have $N_{\tau'} = 2{,}046$ time bins to evaluate. Putting everything together, we would need a time–frequency search space composed of a total of $N_{\tau'}N_{\nu'} {\sim} 2.6 \cdot 10^6$ bins, which is extremely demanding. This example shows how high-sensitivity receivers requiring long coherent integration times, and thus, a very fine frequency search must operate with assistance information in order to make the acquisition process feasible in practice. The same principle applies to any receiver in the sense that assistance information does always help in reducing the frequency search span, and thus, in reducing the search space. But for demanding applications where long coherent integration times are needed, assistance information becomes a must.

Search Strategies

So far we have set the grounds for the time–frequency search, determining the search space, the search bin width and the incurred losses. The next step is to determine how to actually go through the search space and evaluate all the time–frequency cells. One or several cells can be analyzed at each trial, resulting in three different approaches for implementing the time–frequency search:

– *Serial search.* In this case, the decision metric of each cell, which is given by the correlation output in Eq. (1.75), is computed and tested individually right away [51, 52, 14]. Decisions are taken on a cell-by-cell manner, and thus, the search process follows sequentially with the rest of the cells in the search space as illustrated in Figure 1.26(a).

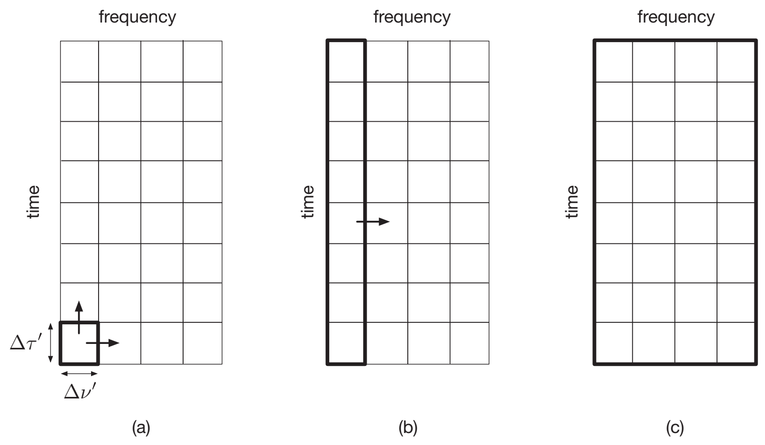


Figure 1.26 Illustration of different search strategies for going through the whole time–frequency acquisition matrix: (a) serial search, (b) hybrid search and (c) full parallel search.

– *Hybrid search.* All time-domain cells are used together to form a decision metric. Typically, this is done when using the FFT to compute the code correlation for a given tentative frequency. The result of the FFT simultaneously provides all the correlation samples, which are used to detect whether the signal is present or not, as illustrated in Figure 1.26(b). Then the search proceeds sequentially by implementing the FFT for a new tentative frequency, and the detection process is repeated again until the whole frequency search space is scanned [53, 54, 39, 40].
– *Full parallel search.* This is an extension of the hybrid search where the FFT implementing the correlation is computed for all the tentative frequencies, and the results are stored in a time–frequency matrix. Then, all cells of this matrix are simultaneously used to detect whether the signal is present or not. This is done for instance by taking the largest value of the whole time–frequency acquisition matrix, and then comparing this maximum value with a given detection threshold [55, 56, 57].

Example of Acquisition Technique: Full Parallel Search

We provide next an algorithmic description of the full parallel search, which is actually the technique implemented in the multi-GNSS software of this book, and therefore, it serves well for the purpose of illustrating an example of acquisition technique. From a high-level perspective, the implementation of the parallel search comprises the following steps:

1. Generate the local spreading code replica $c[n]$ for the satellite to be acquired.
2. Apply the FFT to $c[n]$ (see Section 9.2.9 for more details) and take the complex conjugate of the result. The complex conjugate of the FFT is needed because we seek to implement the correlation operation in the frequency domain. This is possible in virtue of the convolution theorem of the Fourier transform, whereby

the product of Fourier transforms corresponds to the convolution in the time domain [8, Section 2.9.6]. The convolution of two discrete-time signals $x_1[n]$ and $x_2[n]$, defined as $x_1[n] * x_2[n] = \sum_{k=-\infty}^{\infty} x_1[k]x_2[n-k]$, is very similar to the correlation we are looking for, but not identical. The correlation of these two signals is defined as $R_{x_1,x_2}[n] = \sum_{k=-\infty}^{\infty} x_1[k]x_2^*[k-n]$ where the complex conjugate $(\cdot)^*$ has been considered for the sake of completeness, in case the signals were complex-valued. So we can see that convolution and correlation are related as follows, $R_{x_1,x_2}[n] = x_1[n] * x_2^*[-n]$. That is, we need one of the signals to be complex conjugated and time reversed so that by doing the convolution we actually obtain the correlation. Both complex conjugate and time reversal in the time domain imply the complex conjugate only of the Fourier transform [8, Section 2.8]. This is the reason why we are taking the complex conjugate of the code replica FFT in the present step, so we can obtain the correlation of the received signal with the local replica by means of implementing their convolution in the frequency domain.

3. Two options are possible here:
   (a) Multiply the received signal $r[n]$ with a local carrier wave $e^{-j2\pi\nu' n}$ at the tentative frequency value ν' to tentatively remove the frequency shift ν present in $r[n]$. Take the FFT of the result.
   (b) Alternatively, the code replica $c[n]$ could be the one multiplied by the local carrier, in this case, by $e^{j2\pi\nu' n}$. This could be done in the frequency domain by circularly rotating the FFT samples of the code replica, which are already available from step 2. In that case, the frequency shift bin width $\Delta\nu'$ is determined by the frequency resolution of the FFT. Care must be taken when choosing the size of the FFT in order to match the FFT frequency resolution with the required $\Delta\nu'$.
4. Obtain $R_C(\tau', \nu'; k)$ in Eq. (1.76). If step 3(a) was implemented, $R_C(\tau', \nu'; k)$ is obtained by multiplying the result of step 2 with that of step 3(a) and applying the IFFT. If step 3(b) was implemented, $R_C(\tau', \nu'; k)$ is obtained by multiplying the FFT of $r[n]$ with the result of step 3(b) and applying the IFFT.
5. Repeat steps 2-4 for $k = 0, \ldots N_I - 1$ and accumulate $|R_C(\tau', \nu'; k)|^2$ in order to form $R_{NC}(\tau', \nu')$ in Eq. (1.75).
6. The satellite is detected whenever $\max_{\tau',\nu'} R_{NC}(\tau', \nu') > \gamma$ for a given threshold γ. The coarse estimates for the time delay and frequency error, denoted by $(\hat{\tau}, \hat{\nu})$, are obtained as the pair of time–frequency values (τ', ν') where the maximum above is achieved. That is,

$$(\hat{\tau}, \hat{\nu}) = \arg\max_{\tau',\nu'} R_{NC}(\tau', \nu') > \gamma. \tag{1.90}$$

As can be seen, the implementation makes extensive use of the FFT operation in order to efficiently implement the correlation operation, and it is therefore the preferred approach in software receiver implementations. In the subsequent chapters, we will refer to this technique when addressing the implementation details for each specific GNSS signal. Further details on the FFT implementation are discussed in Section 9.2.9.

1.7 Tracking

The acquisition stage is completed once the satellite of interest is detected and coarse estimates of its code delay and frequency shift are obtained. These values are then fed as an input to the tracking stage, whose role is to provide fine estimates of both the code and carrier phase and to keep track of any potential variation over time. If the receiver has a snapshot architecture, no tracking stage is then implemented. Instead, the coarse estimates coming from the acquisition stage are provided as the output observables straightaway. Snapshot architectures are specifically addressed in Chapter 9. In the rest of this chapter, we will focus on conventional GNSS receivers, which implement both acquisition and tracking stages following the architecture shown in Figure 1.16. Thanks to the tracking stage, the receiver can continuously retrieve the bits from the navigation message and measure the satellite ranges that are needed to compute the user's position.

The tracking stage is composed of two separate modules, each of them in charge of code tracking and carrier tracking, respectively. Both modules are implemented using a closed-loop architecture. This involves comparing the input signal with a local replica, determining their relative error, and then using this error to correct the local replica that will be compared again with the incoming signal in the next time instant. Both code and carrier tracking will briefly be described next with the aim of providing a high-level overview of their operation principle.

1.7.1 Carrier Tracking

Carrier tracking is required at the receiver in order to synchronize the local carrier replica with the incoming signal, and thus to completely wipe off the carrier. Once the carrier is properly removed, all subsequent operations such as despreading the PRN code of the satellite under analysis, measuring the time-delay and retrieving the navigation data bits can all be safely carried out as long as the SNR allows us to do so. As can be inferred, carrier tracking is intimately related to code tracking because actually one is depending on the other. Carrier tracking allows the carrier to be wiped off and thus to proceed safely with the coherent correlation for code despreading. But at the same time, carrier tracking needs the code to be despread first, because otherwise the signal of interest is buried well below the noise floor and becomes imperceptible. It is for this reason that both carrier and code tracking operate simultaneously, and indeed, carrier tracking is often used to assist code tracking under the so-called *carrier aiding*. This strategy takes advantage of the capability of carrier tracking to precisely keep track of the user dynamics. Assisting the code tracking stage with the carrier dynamics allows a simpler implementation of the code tracking, since user dynamics are already provided and the emphasis can be placed on just filtering out the thermal noise.

Carrier tracking is implemented following the rationale of the block diagram in Figure 1.27. When carrier tracking is first initiated, the incoming signal is correlated with the local code and carrier replicas using the coarse code delay and frequency shift estimates $(\hat{\tau}, \hat{\nu})$ provided by the acquisition stage. Later, once tracking is ongoing, code delay and carrier phase values are recursively computed within the tracking loop. These values are used to align the local code and carrier replicas to those in the received signal, and the goodness of such alignment is assessed through the correlation of both terms as previously done at the acquisition stage with Eq. (1.65). This leads to the so-called *prompt* correlation, whose expression is given by

$$y_{\text{P}}[k] = \sum_{n=0}^{N_T-1} r[n + kN_T]c^*[n - \hat{\tau}]_{N_{\text{scode}}} e^{-j\hat{\theta}[n;k]}, \tag{1.91}$$

for the k-th prompt epoch, with the locally generated carrier phase given by $\hat{\theta}[n;k] = 2\pi\hat{\nu}[k]n + \hat{\theta}[k]$. Substituting the received signal $r[n]$ with the signal model in Eq. (1.59), the expected value of the prompt correlator results in

$$\text{E}\left[y_{\text{P}}[k]\right] = \sqrt{C} \sum_{n=0}^{N_T-1} c[n + kN_T - \tau]_{N_{\text{scode}}} c^*[n - \hat{\tau}]_{N_{\text{scode}}} e^{j\theta_\epsilon[n;k]}, \tag{1.92}$$

where data symbols or the values of the secondary code in pilot signals have been omitted for simplicity. The carrier phase error $\theta_\epsilon[n;k]$ results from the difference between the carrier phase of the received signal $\theta[n;k]$ and the carrier phase of the local replica $\hat{\theta}[n;k]$. Note that k is indexing the block of N_T received samples being processed at the coherent correlation, whereas n is the sample by sample indexing. Note also that the carrier phase of the received signal can be further decomposed as $\theta[n;k] = 2\pi\nu(n + kN_T) + \theta = 2\pi\nu n + \theta[k]$ for some carrier phase offset $\theta[k]$. So the error between both carrier phases that appears in Eq. (1.92), namely $\theta_\epsilon[n;k]$, is thus given by

$$\theta_\epsilon[n;k] = \theta[n;k] - \hat{\theta}[n;k] = 2\pi\nu_\epsilon[k]n + \theta_\epsilon[k], \tag{1.93}$$

where $\nu_\epsilon[k] = \nu - \hat{\nu}[k]$ stands for the frequency error and $\theta_\epsilon[k] = \theta[k] - \hat{\theta}[k]$ stands for the phase offset error at the k-th prompt correlator epoch. When the summation in Eq. (1.91) takes place, the presence of a residual frequency error $\nu_\epsilon[k]$ leads to the Fejér kernel already mentioned when introducing the ambiguity function in Eq. (1.69), causing the output amplitude of the prompt correlator to degrade. The expected value in Eq. (1.92) thus results in

$$\text{E}\left[y_{\text{P}}[k]\right] = \sqrt{C}\frac{R_p(\tau_\epsilon)}{R_p(0)} \frac{\sin(\pi\nu_\epsilon[k]N_T)}{\sin(\pi\nu_\epsilon[k])} e^{j\theta_\epsilon[k]}, \tag{1.94}$$

which is a complex value. In the sequel, we will refer to the real (i.e. in-phase) and imaginary (i.e. quadrature) components of the prompt correlator output by $I_{\text{P}}[k]$ and $Q_{\text{P}}[k]$, respectively, so that

$$I_{\text{P}}[k] = \text{Re}[y_{\text{P}}[k]], \quad Q_{\text{P}}[k] = \text{Im}[y_{\text{P}}[k]]. \tag{1.95}$$

Once some preliminaries have been introduced, we are now in position to describe how carrier tracking is implemented in a GNSS receiver following the architecture in Figure 1.27. As can be seen, the aforementioned prompt correlator output samples are fed to a carrier discriminator. This block is in charge of providing an error signal $\epsilon[k]$ that, on average, is proportional to the (frequency or phase) error between the

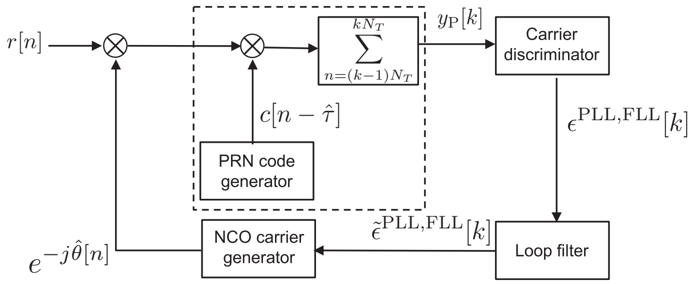


Figure 1.27 Generic block diagram for carrier tracking in a GNSS receiver.

received signal and the local carrier replica. A tracking loop making use of a *phase* discriminator is referred to as a Phase-Locked Loop (PLL), while a tracking loop making use of a *frequency* discriminator is referred to as a Frequency-Locked Loop (FLL). Regardless of the type of discriminator, the output error signal needs to be filtered by a loop filter to obtain a smoothed error signal. The latter is finally used to control the generation of the carrier replica for the next epoch, which is carried out by a numerically controlled oscillator (NCO). The resulting closed-loop architecture is able to align the local carrier to the received one by subsequently iterating until the resulting error signal is driven to zero.

Phase-Locked Loop Discriminators

It is important to bear in mind that carrier phase tracking needs to be insensitive to the 180° phase shifts that, from time to time, are introduced by the symbol transitions in data-modulated GNSS signals. A way to circumvent this limitation is by using a *Costas* discriminator, which is a type of discriminator insensitive to sign transitions. The baseline Costas phase discriminator relies on the atan of the prompt correlation samples so that the signal error is generated as

$$\epsilon_{\mathrm{atan}}^{\mathrm{PLL,Costas}}[k] = \mathrm{atan}\left(\frac{Q_P[k]}{I_P[k]}\right) = \theta_\epsilon[k] \in [-90°, 90°]. \tag{1.96}$$

The result in Eq. (1.96) actually provides the phase error to be corrected regardless of the presence of data-modulating symbols. The main drawback is that it relies on a nonlinear function (atan), and thus, its hardware implementation may pose some troubles. Alternative Costas discriminators have been proposed to circumvent this problem at the expense of shrinking the linear region of their input-output response. Some examples are shown below along with the region in which they operate linearly. That is, the region where their output coincides with the phase error $\theta_\epsilon[k]$ that is present at their input and that we are looking for.

$$\begin{aligned} \epsilon_{I\times Q}^{\mathrm{PLL,Costas}}[k] &= I_P[k]Q_P[k] &&= \sin(2\theta_\epsilon[k]) &&\approx \theta_\epsilon[k] \in [-22.5°, 22.5°] \\ \epsilon_{\mathrm{DD}}^{\mathrm{PLL,Costas}}[k] &= \mathrm{sign}(I_P[k])Q_P[k] &&= \sin(\theta_\epsilon[k]) &&\approx \theta_\epsilon[k] \in [-45°, 45°]. \end{aligned} \tag{1.97}$$

Note that $\epsilon_{\mathrm{DD}}^{\mathrm{PLL}}[k]$ is designed for BPSK-modulated symbols. It implements a Decision Directed (DD) strategy for deciding the data-modulated symbol through sign$(I_P[k])$

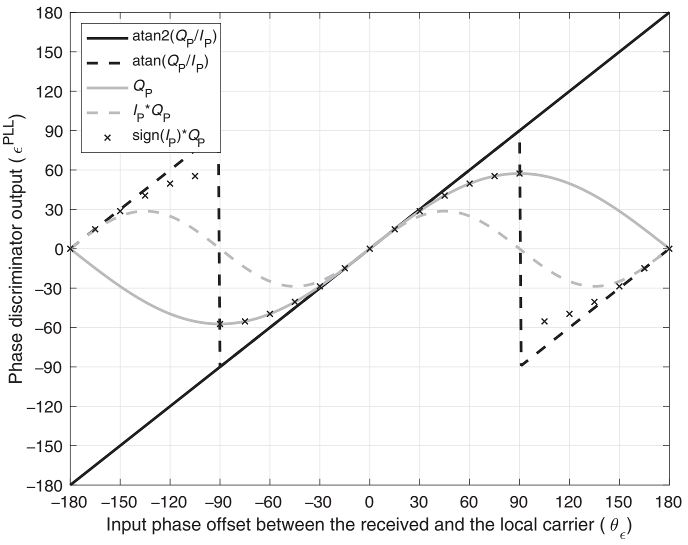


Figure 1.28 Output $\epsilon[k]$ provided by different PLL discriminators as a function of the input phase difference $\theta_\epsilon[k]$ between the received carrier and the local replica. Phase in units of degrees.

and then uses this decision for removing the contribution of that symbol on $Q_P[k]$. The expressions above assume that an AGC is applied to the prompt correlator in Eq. (1.91) so that it is normalized to unit power, $P_{y_P} = \mathrm{E}\left[|y_P[k]|^2\right] = 1$. Otherwise, the discriminators in Eq. (1.97) turn out to depend on P_{y_P} and $\sqrt{P_{y_P}}$, respectively, so they should be normalized accordingly using an estimate of the prompt correlator power. Note also that for small arguments it is reasonable to assume $\sin(x) \approx x$, so the low-complexity Costas discriminators in Eq. (1.97) actually provide a valid correction for the NCO at the expense of a reduced input linear operation region. This can be clearly seen in Figure 1.28 where the input-output response of some Costas phase discriminators is shown, normalized so that their slopes are equal to one in the linear region. Such input-output representation is often referred to as the *S-curve*,[28] typically having a linear region for small inputs, then a zero slope for larger inputs, and sometimes, a negative slope for much larger inputs that may cause the curve to exhibit additional zero-crossings.

It is worth noting that in the absence of data-modulating symbols, such as when tracking pilot signals, the limited input range of Costas discriminators leads to an unnecessary degraded performance in the presence of severe noise. In these circumstances, it is preferred to avoid Costas discriminators and go for a PLL discriminator based on the full-range, or four-quadrant version atan2, which is widely used in software receiver implementations. In that case,

$$\epsilon^{\mathrm{PLL}}_{\mathrm{atan2}}[k] = \arg\left(y_P[k]\right) = \mathrm{atan2}\Big(\frac{Q_P[k]}{I_P[k]}\Big) = \theta_\epsilon[k] \in [-180^\circ, 180^\circ], \tag{1.98}$$

[28] The S-curve is also used for code tracking discriminators, as later shown in Figure 1.32.

where $\arg(\cdot)$ stands for the complex argument (i.e. angle) of a complex number. A simpler implementation is possible recalling that $\arg(x) \approx \text{Im}[x]$ for a small and complex number x. Then we have

$$\epsilon_Q^{\text{PLL}}[k] = \text{Im}\left[y_\text{P}[k]\right] = Q_\text{P}[k] \approx \theta_\epsilon[k] \in [-45^\circ, 45^\circ]. \tag{1.99}$$

Similarly to the previous low-complexity Costas discriminators in Eq. (1.97), the PLL discriminator in Eq. (1.99) also needs to be normalized to avoid fluctuations caused by variations on the received signal strength. In this case, normalized by $\sqrt{P_{y_\text{P}}}$.

Phase-Locked Loop Performance and Filter Bandwidth

The carrier phase tracking performance can be assessed through the variance of the estimated phase at the NCO output. Assuming that the tracking loop operates in the linear region, such variance is given by (see for instance [4, Section 12.3.6])

$$\sigma_{\hat{\theta}}^2 = \frac{B_\text{L}}{\frac{C}{N_0}}\left(1 + \frac{1}{2\frac{C}{N_0}T}\right) \quad (\text{rad}^2), \tag{1.100}$$

with B_L the bandwidth of the loop filter in charge of smoothing the discriminator output and T the coherent correlation time corresponding to the N_T samples used in Eq. (1.91) to obtain the prompt correlation output. Note that the term $\frac{C}{N_0}T$ in Eq. (1.100) becomes the SNR at the output of the prompt correlator. The overall term in brackets in Eq. (1.100) is known as the squaring losses and accounts for the performance degradation that is experienced at medium- to low-SNR due to nonlinear effects caused by the increased noise. Carrier phase tracking is a well-known topic that has extensively been addressed in the existing literature. An exhaustive coverage of this topic is thus out of the scope of the present book. Just the fundamental concepts have been addressed here to support the understanding and operation of the companion software to this book. The interested reader can find further details on this topic in [58, 59, 60] or [10] for a comprehensive overview. For details on the loop filter design and performance analysis, the interested reader is referred to [4, Section 12.3].

Frequency-Locked Loops

Regarding FLL implementations, they provide an alternative way of implementing carrier tracking by using frequency measurements instead of phase ones. By doing so, FLLs provide much more robust performance than PLLs in the presence of high user dynamics, owing to the fact that frequency variations are always much smaller than phase variations. This allows the tracking loop to operate in a more relaxed manner and avoid the abrupt jumps that are typically observed in phase measurements. The price to be paid is that there is no access to the actual phase of the received signal, but only to its frequency. This means that the phase of the locally generated carrier replica is always behind the phase of the received signal by a constant phase offset. This is not a major problem because such offset can be resolved at the demodulation stage of the receiver. However, the common practice in many GNSS receiver implementations is to first start the tracking stage using an FLL and once in steady-state, switch to a PLL to perform the finer tracking and resolve the pending phase offset ambiguity.

Most FLL discriminators are based on the observation that the product between two consecutive prompt correlator outputs provides a complex value whose argument is proportional to the frequency error. This motivates the following baseline FLL discriminator,

$$\epsilon_{\text{atan2}}^{\text{FLL}}[k] = \frac{1}{2\pi N_T} \arg\left(y_{\text{P}}[k]y_{\text{P}}^{*}[k-1]\right), \tag{1.101}$$

which is often expressed in terms of in-phase and quadrature formulation as follows:

$$\epsilon_{\text{atan2}}^{\text{FLL}}[k] = \frac{1}{2\pi N_T} \operatorname{atan2}\left(\frac{I_{\text{P}}[k-1]Q_{\text{P}}[k] - I_{\text{P}}[k]Q_{\text{P}}[k-1]}{I_{\text{P}}[k]I_{\text{P}}[k-1] + Q_{\text{P}}[k]Q_{\text{P}}[k-1]}\right). \tag{1.102}$$

The four-quadrant arctangent discriminator above (atan2) can also be implemented in a decision-directed manner by using the two-quadrant arctangent discriminator instead (atan), as done in the Costas PLL shown in Eq. (1.96). By doing so, we get rid of the 180° phase shifts due to symbol transitions in the prompt correlator samples.

In the sequel, we will consider Eq. (1.101), though, because it allows a more intuitive interpretation. Actually, Eq. (1.101) is providing the phase difference taking place in the complex product of two consecutive prompt correlator outputs, namely $y_{\text{P}}[k]y_{\text{P}}^{*}[k-1]$. Note that being a phase difference, it has the sense of frequency, and thus, it is consistent with being a frequency discriminator. It is not difficult to see that the expression in Eq. (1.101) is equivalent to:

$$\epsilon_{\text{atan2}}^{\text{FLL}}[k] \equiv \frac{\theta_{\epsilon}[k] - \theta_{\epsilon}[k-1]}{2\pi N_T} = \nu_{\epsilon}[k], \tag{1.103}$$

which is the frequency error between the received signal and the local carrier replica, $\nu_{\epsilon}[k]$. Actually, what we were looking for.

Similarly to the PLL, low-complexity FLL discriminators have been proposed to circumvent the nonlinear function for retrieving the angle of a complex number. Some examples are the following:

$$\begin{aligned} \epsilon_{I\times Q}^{\text{FLL}}[k] &= \operatorname{Im}\left[y_{\text{P}}[k]y_{\text{P}}^{*}[k-1]\right] = I_{\text{P}}[k-1]Q_{\text{P}}[k] - I_{\text{P}}[k]Q_{\text{P}}[k-1] \\ \epsilon_{\text{DD}}^{\text{FLL}}[k] &= \operatorname{sign}\left(\epsilon_{I\times Q}^{\text{FLL}}[k]\right)\epsilon_{I\times Q}^{\text{FLL}}[k]. \end{aligned} \tag{1.104}$$

Finally, it is interesting to note that the phase difference taking place in $y_{\text{P}}[k]y_{\text{P}}^{*}[k-1]$ can be extended to the product of two prompt correlations separated apart by more than one epoch. The general case would be products in the form $y_{\text{P}}[k]y_{\text{P}}^{*}[k-m]$ for $m \geq 1$. The combination of several of these terms helps in mitigating the noise increase that is experienced when two noisy samples are multiplied, as it is the case in the FLL discriminators presented so far. This approach has its roots on autocorrelation-based frequency estimators such as the Kay, Fitz and Luise & Reggiannini methods widely adopted in digital communication receivers. The interested reader is referred to [61, Section 3.2] for a detailed while comprehensive discussion on the topic.

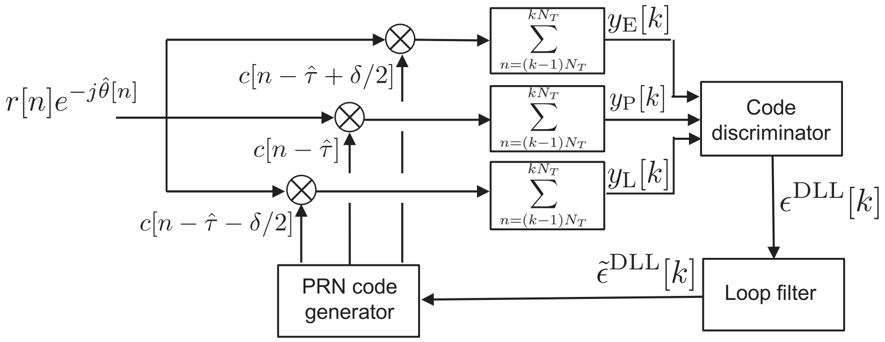
1.7.2 Code Tracking

Code tracking shares the same principle as carrier tracking, but now the magnitude being tracked is not the carrier phase or carrier frequency but the code offset of the local code replica with respect to the incoming signal. Contrary to carrier tracking, which uses only the prompt correlator, code tracking requires more than one correlator to ascertain whether the local code replica is truly aligned with the received one or not. In practice, two more correlators, each of them referred to as the *early* and the *late* correlator, often suffice for this purpose apart from the prompt one already used for carrier tracking. Each of these two additional correlators is placed on each side of the correlation time, at a distance $\delta/2$ from the central correlation peak. The separation δ is therefore referred to as *early-late separation*.

The generic block diagram of a code tracking architecture is shown below in Figure 1.29, where the three aforementioned correlators are denoted by $y_{\mathrm{E}}[k]$, $y_{\mathrm{P}}[k]$ and $y_{\mathrm{L}}[k]$ for the early, prompt and late, respectively. The output of these correlators is a complex value indicating how well each specific code replica is aligned with the received signal. It is for this reason that code tracking architectures are often referred to as Delay-Locked Loops (DLL) . Following the same convention as for carrier tracking, their in-phase and quadrature components are referred herein as $I_{\mathrm{E,P,L}}[k]$ and $Q_{\mathrm{E,P,L}}[k]$ depending on the specific correlator under analysis. Note that the input signal for code tracking is the received signal $r[n]$ with the carrier already wiped off using the carrier replica generated by the carrier tracking module. This indicates that both carrier and code tracking must operate simultaneously, since the output of each of them has an impact on the other.

Delay-Locked Loop Correlator Spacing

Regarding the correlator spacing, typically $\delta = 1$ chip is used in most GNSS receiver implementations, especially for BPSK signals. A narrower separation provides better accuracy in determining the code delay of the input signal and is more robust to the presence of multipath. A narrower separation is needed as well to track BOC signals in order to monitor the central correlation peak, whose width is typically a fraction of chip. However, using a too narrow spacing is troublesome when rapid variations of the code delay are experienced, either because of the user's dynamics or because of rapidly



$r[n]e^{-j\hat{\theta}[n]}$
$c[n-\hat{\tau}+\delta/2]$
$c[n-\hat{\tau}]$
$c[n-\hat{\tau}-\delta/2]$
$\sum_{n=(k-1)N_T}^{kN_T}$
$y_{\mathrm{E}}[k]$
$y_{\mathrm{P}}[k]$
$y_{\mathrm{L}}[k]$
Code discriminator
$\epsilon^{\mathrm{DLL}}[k]$
Loop filter
$\tilde{\epsilon}^{\mathrm{DLL}}[k]$
PRN code generator

Figure 1.29 Generic block diagram for code tracking in a GNSS receiver.

fluctuating multipath. In these circumstances, it is beneficial to use a wider early-late separation. Modern GNSS receivers allow the early-late spacing to be adjusted while the receiver is tracking the signal. This has the advantage of providing an additional degree of freedom for the receiver to adapt the tracking stage to the working conditions, in order to cope with situations where the C/N_0 suddenly drops, and thus to reduce the risk of loss of lock.

To better illustrate the underlying concept of code tracking let us consider the following example. In Figure 1.30, right, the late code has the highest correlation, so the estimated code delay $\hat{\tau}$ must be increased. In Figure 1.30, left, however, the highest peak is located at the prompt replica, and the early and late replicas have equal correlation. It is in this situation where one can claim that the code delay is being properly tracked. The estimated code delay $\hat{\tau}$ is now aligned with the true delay τ of the received signal, and the resulting delay error τ_ϵ goes to zero. From Figure 1.30, one can also infer the correlation process as a multiplication of the sampled replica and signal, and accumulation. The black samples in the replicas contribute to the correlation gain, as they multiply a $+1$ by a $+1$, or a -1 by a -1. The grey samples do not, as they multiply a $+1$ by a -1, which happens when the replica and the code are not well aligned. Note that, in reality, the signal is buried in noise, so the correlation of several samples is needed to raise the signal above the noise level. Figure 1.30 also shows why it is convenient to select a sampling frequency F_s that is not a multiple of the spreading code frequency, to avoid aliasing, as otherwise several contiguous replicas would have the same gain, leading to an accuracy reduction.

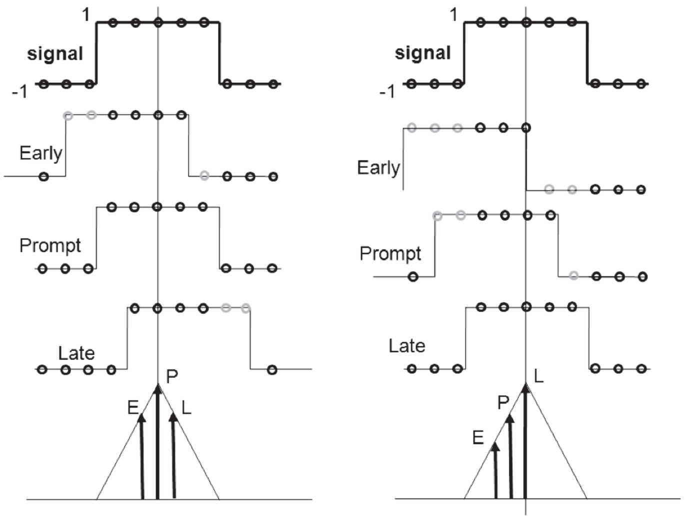


Figure 1.30 Example where the early, prompt and late replicas are correlated with the incoming signal. Left: The prompt code has the highest correlation and $I_E[k] \approx I_L[k]$, so the loop is perfectly tuned. Right: The late replica has the highest correlation, so the code delay must be increased, that is, the code must be delayed. Note that T_s is purposely not a multiple of T_c.

Another example is shown in Figure 1.31 with the evolution of the early, late and prompt correlator outputs as a function of time. Figure 1.31(a) corresponds to the case where the PLL is not locked. As can be seen, the early, late and prompt outputs of the DLL are not locked either. They are actually rotating in the complex plane, and their energy is moving from the in-phase (upper subplot) to the quadrature component (bottom subplot) and vice versa. In contrast, Figure 1.31(b) corresponds to the case where the PLL is locked so that the carrier is properly wiped off from the received signal. In that case, all the energy is now on the in-phase component of the DLL, as shown in the upper plot of Figure 1.31(b). The DLL is also found to operate as expected, with the early and late correlators having a similar magnitude, indicating that the code replica is well aligned with the received signal. The square value of the early, late and prompt correlators is shown in Figure 1.31. It is for this reason that the upper plot in Figure 1.31(b) has a magnitude of 0.25 for the early and late correlators, which is 0.5^2, being 0.5, half, the height of a unit-amplitude correlation peak when a rectangular chip pulse is considered.

Delay-Locked Loop Discriminators

Similarly to carrier tracking, the way in which the error signal is obtained from the correlation between the received signal and the local code replica plays an important role in the code tracking performance. For a DLL, such error signal is obtained by means of a code or DLL discriminator whose output $\epsilon^{DLL}[k]$ is fed back to the code replica generator, in order to compensate for the misaligment of the local replica with respect to the received signal. There are essentially two different types of DLL discriminators depending on their sensitivity to carrier phase mismatches. The first type is composed of the so-called *coherent* discriminators, which operate straight-away on the correlation output samples assuming that the carrier phase is perfectly tracked. This is the case of the early-minus-late (EML) discriminator given by

$$\epsilon^{\mathrm{DLL}}_{\mathrm{EML}}[k] = \frac{1}{2}\left(I_{\mathrm{E}}[k] - I_{\mathrm{L}}[k]\right), \tag{1.105}$$

which is the simplest form of DLL discriminator. It has the problem, though, that it depends on the signal amplitude and thus, variations in the received signal power cause the EML output to vary as well. In order to avoid this problem, the EML discriminator is often normalized by the magnitude of the early plus the late correlator outputs. This leads to the so-called normalized EML (NEML) discriminator as follows:

$$\epsilon^{\mathrm{DLL}}_{\mathrm{NEML}}[k] = \frac{1}{2}\frac{I_{\mathrm{E}}[k] - I_{\mathrm{L}}[k]}{|I_{\mathrm{E}}[k]| + |I_{\mathrm{L}}[k]|}. \tag{1.106}$$

Another problem of the EML discriminator is that it is sensitive to the presence of data-modulating symbols. To circumvent this limitation, the EML is often replaced by the *dot-product* (DP) discriminator, which takes the same expression in Eq. (1.105) but multiplies by the in-phase component of the prompt correlator. The latter contains the sign of the data-modulated symbols and thus multiplying by it, the symbol is effectively removed from the discriminator output. The expression of the DP discriminator is therefore,

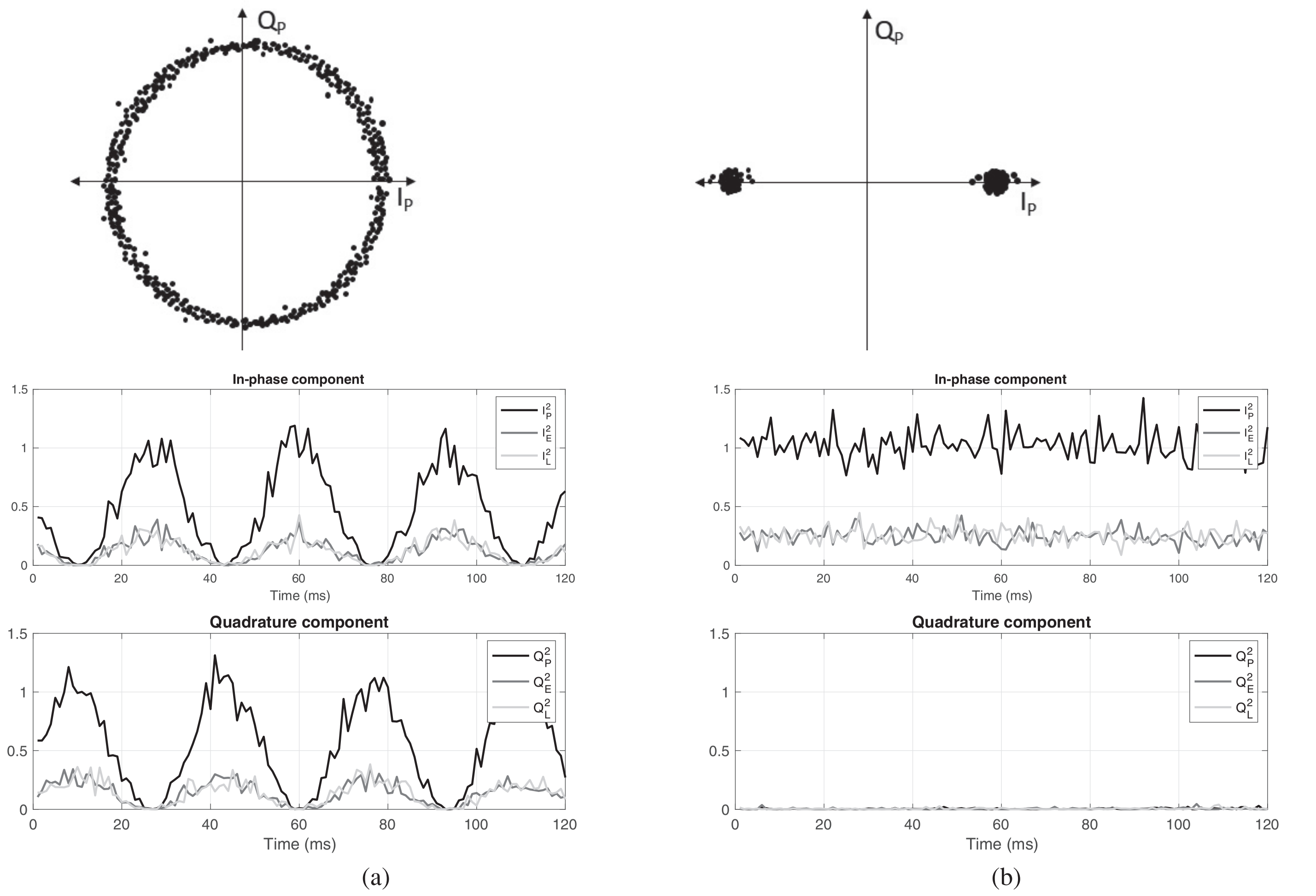


Figure 1.31 (a) Output of the in-phase and quadrature arms of the DLL when the PLL is not locked yet. (b) Output of the DLL when the PLL is locked. The upper part shows the prompt correlator outputs in the complex plane. A BPSK(1) signal is considered with an early-late separation of $\delta = 1$ chip.

$$\epsilon_{\mathrm{DP}}^{\mathrm{DLL}}[k] = \frac{1}{2}\left(I_{\mathrm{E}}[k] - I_{\mathrm{L}}[k]\right) I_{\mathrm{P}}[k]. \tag{1.107}$$

The result is again affected by variations of the received signal power, so a normalized version of the DP discriminator is often used by dividing Eq. (1.107) with the square of the prompt correlator, which acts as an estimate of the instantaneous power. This, however, is only valid when the code delay error is small. Otherwise, the square of the prompt correlator is no longer a reliable estimate of the received power because it is affected by the code delay error, and thus, the output of the discriminator would become biased. According to the aforementioned discussion, the resulting normalized DP (NDP) discriminator is given by

$$\epsilon_{\mathrm{NDP}}^{\mathrm{DLL}}[k] = \frac{1}{2}\frac{I_{\mathrm{E}}[k] - I_{\mathrm{L}}[k]}{I_{\mathrm{P}}[k]}. \tag{1.108}$$

The second type of DLL discriminators are the *noncoherent* ones, which typically take the square or the magnitude of the correlation outputs in order to become insensitive to residual phase errors and to the presence of data symbols. As can be inferred, such robustness comes at the price of increasing the noise due to the nonlinear operation on the noisy input samples. The impact is often acceptable and the fact of being insensitive to the two aforementioned effects typically pays off. One of the most popular noncoherent DLL discriminators is the normalized early minus late power (NEMLP) discriminator,

$$\epsilon_{\mathrm{NEMLP}}^{\mathrm{DLL}}[k] = \frac{1}{4}\frac{(I_{\mathrm{E}}^2[k] + Q_{\mathrm{E}}^2[k]) - (I_{\mathrm{L}}^2[k] + Q_{\mathrm{L}}^2[k])}{(I_{\mathrm{E}}^2[k] + Q_{\mathrm{E}}^2[k]) + (I_{\mathrm{L}}^2[k] + Q_{\mathrm{L}}^2[k])}, \tag{1.109}$$

which relies on the squared modulus of the early and late correlators. An alternative using just the modulus instead of the squared value is the normalized noncoherent EML (NNEML), which can be seen as the noncoherent implementation of the EML in Eq. (1.105), and it is the discriminator actually used in the software receiver accompanying the present book. It is given by[29]

$$\epsilon_{\mathrm{NNEML}}^{\mathrm{DLL}}[k] = \frac{1}{2}\frac{\sqrt{I_{\mathrm{E}}^2[k] + Q_{\mathrm{E}}^2[k]} - \sqrt{I_{\mathrm{L}}^2[k] + Q_{\mathrm{L}}^2[k]}}{\sqrt{I_{\mathrm{E}}^2[k] + Q_{\mathrm{E}}^2[k]} + \sqrt{I_{\mathrm{L}}^2[k] + Q_{\mathrm{L}}^2[k]}}. \tag{1.110}$$

The output of the six discriminators discussed above is shown in Figure 1.32 for the case of an infinite bandwidth rectangular chip pulse. As can be seen, they all exhibit a linear behavior with a unitary slope when the input code delay error is small, that is, $\tau_\epsilon \to 0$. For larger values of the input code delay error, the widest linear region is provided by the EML-type discriminators, namely the EML, NEML and NNEML discriminators, spanning within the interval $\tau_\epsilon/T_c \in [-0.5, +0.5]$ chips. At least, in

[29] The $1/2$ scaling factor in Eq. (1.110) is merely used here for the sake of convenience to guarantee a unitary slope within the linear region of the discriminator, as shown in Figure 1.32. This scaling factor, as those in the rest of discriminators discussed herein, is actually not needed provided that its effect is accounted for in the design of the DLL tracking loops, as it was done in our software receiver.

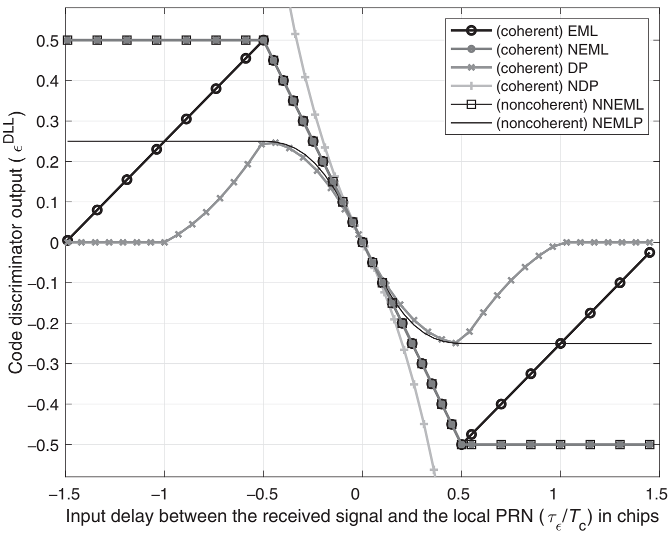


Figure 1.32 Output provided by different code discriminators as a function of the code delay difference τ_ϵ at their input. Infinite bandwidth is assumed.

the absence of noise, as considered herein. In contrast, the NDP and NEMLP discriminators do only provide a linear response over a much narrower interval of barely $\tau_\epsilon/T_c \in [-0.15, +0.15]$ chips. This involves that the two latter discriminators are more sensitive to the presence of input noise, since variations on the early and late correlators can easily lead the discriminator output to lie outside of the linear region and thus to incur in bias.

Delay-Locked Loop Performance and Filter Bandwidth

Regarding the performance of code tracking, the variance of the closed-loop estimated code delay for a coherent DLL discriminator such as the EML can be approximated for a BPSK signal by

$$\sigma^2_{\hat{\tau},\mathrm{EML}} \approx \frac{B_L \delta}{2\frac{C}{N_0}} \qquad (\text{chips}^2), \tag{1.111}$$

with B_L the loop filter bandwidth. Similarly, the variance of the closed-loop estimated code delay for a noncoherent DLL discriminator such as the NEMLP can be approximated by

$$\sigma^2_{\hat{\tau},\mathrm{NEMLP}} \approx \frac{B_L \delta}{2\frac{C}{N_0}}\left(1 + \frac{2}{\frac{C}{N_0}T}\right) \qquad (\text{chips}^2), \tag{1.112}$$

as shown in [4, Eq. (12.34)]. Finally, it is interesting to note that, as a rule of thumb, code tracking is assumed to be in lock when $\sigma_{\hat{\tau}} < \delta/12$ [13, Section 19.4]. This is a simple guideline that can be used to ascertain whether the receiver is in track at code level or the tracking has been lost and then re-acquisition may be needed. The interested reader can find the derivation of Eqs. (1.111) and (1.112) in [62, Chapters 7–8], [4, Chapter 12] or [13, Chapter 19].

1.7.3 C/N_0 Estimation

As already introduced in Section 1.4.1, the C/N_0 is a key metric to measure the strength of the received GNSS signal, and it is usually measured in the tracking block. However, the receiver cannot measure this ratio directly and it has to estimate it. A well-known technique for (C/N_0) estimation will be described here. It is called Narrow-band Wide-band Power Ratio (NWPR),[30] and is based on the ratio of the signal wideband power to its narrow-band power as mentioned in [62, Chapter 8]:

$$\left(\frac{C}{N_0}\right)[k] = 10\log_{10}\left(\frac{1}{T}\cdot\frac{\hat{\mu}_{\mathrm{NP}}[k]-1}{M-\hat{\mu}_{\mathrm{NP}}[k]}\right), \tag{1.113}$$

where $\hat{\mu}_{\mathrm{NP}}$ is the mean normalized power, as expressed in the following equation:

$$\hat{\mu}_{\mathrm{NP}}[k] = \frac{1}{K}\sum_{k=1}^{K}\mathrm{NP}[k], \tag{1.114}$$

where NP$[k]$ is the normalized power between narrow-band power and wide-band power, given by

$$\mathrm{NP}[k] = \frac{\mathrm{NBP}[k]}{\mathrm{WBP}[k]}, \tag{1.115}$$

with NBP$[k]$ and WBP$[k]$ defined as follows:

$$\mathrm{NBP}[k] = \left(\sum_{n=1}^{M} I_{\mathrm{P}}[n+kM]\right)^2 + \left(\sum_{n=1}^{M} Q_{\mathrm{P}}[n+kM]\right)^2, \tag{1.116}$$

$$\mathrm{WBP}[k] = \sum_{n=1}^{M} I_{\mathrm{P}}^2[n+kM] + Q_{\mathrm{P}}^2[n+kM]. \tag{1.117}$$

In the equations above, M stands for the number of correlation output samples that are accumulated to get one sample of the ratio in Eq. (1.115). Typically $M = 20$ in GPS L1 C/A signals in order to implement a one symbol length coherent integration, assuming that symbol synchronization has already been achieved. Then, K is usually set to $K = 50$ in order to have one smoothed C/N_0 estimate at a rate of one value per second.

The NWPR estimation is useful for receivers tracking signals that can be coherently integrated for relatively long periods. However, this is not the case for tracking signals with frequent symbol transitions or snapshot receivers. For these two cases, this book proposes alternative C/N_0 estimators. Section 7.2.1 proposes a C/N_0 estimator where there are frequent symbol transitions, based on the signal-to-noise power ratio (SNPR), and Section 9.2.6 presents an estimator for snapshot positioning based on noncoherent integrations.

[30] This technique is implemented in the FGI-GSRx software companion of this book.

1.8 Navigation Message

The previous sections have explained how the signals can be acquired and tracked, allowing us to generate the range measurements and demodulate the symbols. This section focuses on the navigation message. The navigation message provides the information to calculate the satellite position, among other data. This section presents how the receiver can synchronise to the signal and start interpreting the symbols, how these symbols encode the navigation bits to enhance reception even in case of signal impairments and what are the main navigation message parameters.

1.8.1 Synchronization to the Signal

We have described how to track the carrier phase and the code delay through the PLL/FLL and DLL, respectively. However, none of these outputs allow, strictly speaking, calculation of the satellite-receiver range. The carrier phase measurement provides an estimation of the signal phase at a given cycle, but the number of cycles, or wavelengths, between the satellite and the receiver is unknown. Since GNSS wavelengths are in the order of 18 to 25 cm, this carrier phase integer ambiguity will remain unknown unless the receiver has very accurate error information and uses special techniques such as real-time kinematics (RTK) or PPP, which are out of the scope of this book (see, for example, Part B of [63] for an updated guide on these techniques). Similarly, the DLL output provides an estimate of the *code delay*, or the fractional measurement of the satellite-to-receiver signal delay within one code length. That is, a value between 0 and 1 ms (0 and 300 km) for GPS L1 C/A, and between 0 and 4 ms (0 and 1200 km), for Galileo E1-C or E1-B. In order to compute the full range measurement, the receiver needs to know the integer number of full codes to be added to the fractional code delay measurement, that is, the remaining part of the total distance. In order to do that, classic receivers need to synchronize to the navigation message. By this synchronization, the receiver can determine the *transmit time*, or when the code associated with the measured code delay was transmitted. This is achieved because the message time-tags symbol transitions in the data stream, and this allows determining the number of codes, or time, between our code delay measurement, and the symbol transition associated with the time reported in the signal, as explained in the next section.

The synchronization process starts with the receiver processing symbols until a fixed sequence is received. For GPS L1 C/A, this sequence is composed of the bits (10001011) and it is called *preamble*. It is within a field called Telemetry Word that is followed by another field called Hand-Over Word, that provides the GPS Time (GPST). These fields appear once every GPS L1 C/A *subframe*, or every 6 seconds. Often, after recognizing the preamble, receivers check the parity of the message portion in which it is embedded, to verify that the sequence is not due to a combination of bits from other sections of the message. From that point onward, the receiver can interpret the bits of the navigation message. The next step is, by interpreting these bits, to synchronize with the beginning of a 30-second *frame*. Once

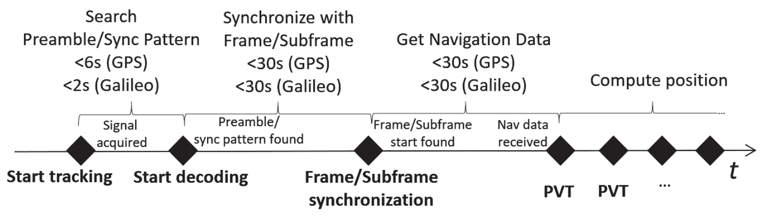


Figure 1.33 Receiver synchronization with the navigation message for GPS L1 C/A and Galileo I/NAV.

this is achieved, the receiver can wait for another 30 seconds to decode the satellite position data, or *ephemeris*, described later in this chapter. Receivers can save some time by combining data from two contiguous frames so the total decoding time can be around 30 seconds. The GPS L1 C/A navigation message lasts 12.5 minutes due to the *almanacs*, also explained later, but the receiver does not need this information to have a position. We call the time to compute the first position *Time To First Fix* (TTFF). The process for Galileo E1-B I/NAV is very similar. The message provides a 10-bit *synchronization pattern* (0101100000), every 2 seconds, similar to the GPS preamble. From that sequence onward, the receiver can process the navigation data. Galileo I/NAV also transmits the satellite position data and time every 30 seconds, or Galileo I/NAV *subframe*, so the TTFF process is similar. Figure 1.33 illustrates the receiver synchronization process for GPS L1 C/A and Galileo I/NAV.

1.8.2 Channel Encoding and Decoding

The navigation data are often received with errors due to the impairments in its path from the satellite to the receiver. Therefore, *channel encoding* schemes are added at the transmitter to encode the data bits, allowing the receiver to detect corrupted bits, and even recover them in some cases. The main error detection and correction principles used in GNSS are outlined in this section.

Error Detection

The navigation message has some extra bits allowing the receiver to check if it has been corrupted. This is commonly defined as a *checksum*. For example, the GPS L1 C/A signal provides six bits of parity, or check bits, for every 24 bits of information, through a (32,26) *Hamming code* [64]. Most messages, as for example the Galileo I/NAV, F/NAV and C/NAV, provide a cyclic redundancy check (CRC). The principle of CRC generation is to predefine a *generator polynomial* G of degree equal to the size of the CRC, create a polynomial including the bit stream to check, divide it by G and check that the reminder R obtained from the division is the same as the transmitted CRC. For example, in the case of Galileo I/NAV, M is the stream of 196 bits, that is expressed as a polynomial of degree 196. For I/NAV, F/NAV and C/NAV, $G(X)$ is a predefined polynomial of degree 24 which depends on the prime, primitive or irreducible polynomial $P(X)$:

$$\begin{aligned} G(X) &= (1+X)P(X) \\ P(X) &= X^{23} + X^{17} + X^{13} + X^{12} + X^{11} + X^{9} + X^{8} + X^{7} + X^{5} + X^{3} + 1. \end{aligned} \tag{1.118}$$

Then, the polynomial $m(X)X^{24}$, that is, the original message $m(X)$ with 24 zeros appended, is divided by $G(X)$, and the 24-bit reminder is the CRC.

Error Correction

Modern coding techniques allow not only error detection but also error correction. There are multiple error detection algorithms which are described in the various ICDs. For example, Galileo, GPS L5, NAVIC L5 and satellite-based augmentation system (SBAS) signals include forward error correction (FEC) convolutional encoding, by which n bits are encoded into k symbols, where $r = n/k$ is the code rate. The convolutional property implies that each symbol depends on a number of bits, which is the *constraint length* of the encoder. The encoding process for most of these signals is depicted in Figure 1.34,[31] where $r = 1/2$, the constraint length $c = 7$ and the encoding sequence is performed through two polynomials ($G1 = 171$ octal, or 1111001 binary, and $G2 = 133$ octal, or 1011011 binary; they are equivalent to the arrows pointing up and down, respectively, in Figure 1.34, where D expresses one bit shift, or delay). The encoded sequence alternates symbols from $G1$ and $G2$. Other error detection techniques are low-density parity check, for GPS L1C signals, and Reed Solomon, used for Galileo [65] and QZSS [66]. See Figure 1.34 as an example of FEC encoding.

As regards decoding in the receiver, the most common decoder used for convolutional codes is the *Viterbi decoder*. The main property of the Viterbi decoder is that it estimates the bits recursively until the sequence with the maximum likelihood is found. The Viterbi decoder can be implemented through *hard-decision* or *soft-decision* decoding, where the former takes as an input only binary symbol values and the latter takes as an input intermediate values. For more details on Viterbi decoders, see [67] or [68].

Interleaving

Due to the perturbations in the signal transmission channel, bit errors often come in bursts, affecting several continuous bits. In order to make the encoded sequence more

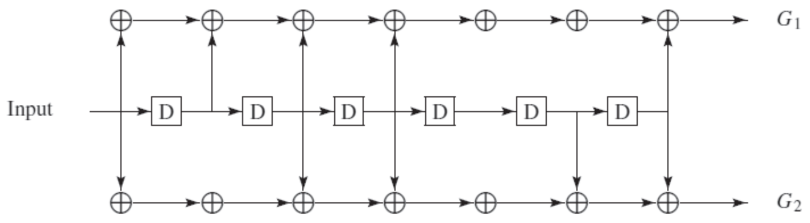


Figure 1.34 Convolutional encoder.

[31] For Galileo, the G_2 symbol is inverted at the end.

robust against bursts, the order of the symbols is changed before transmission through *interleaving*, which means that the encoded sequence is written in columns and then read in rows before transmission. Interleaving is normally implemented by using a two-dimensional buffer array. The data enter the array column by column and leave the array row by row. A burst of errors in the communication channel is spread out into single-symbol errors, which are easier to correct than consecutive errors.

We illustrate in the following example how an input data stream is written into an array column by column. Then, it is read out row by row, to achieve the bit interleaving operation:

– Data input: $x_1\ x_2\ x_3\ x_4\ x_5\ x_6\ x_7\ x_8\ x_9\ x_{10}\ x_{11}\ x_{12}\ x_{13}\ x_{14}\ x_{15}\ x_{16}$
– Interleaving array:

$$\begin{matrix} x_1 & x_5 & x_9 & x_{13} \\ x_2 & x_6 & x_{10} & x_{14} \\ x_3 & x_7 & x_{11} & x_{15} \\ x_4 & x_8 & x_{12} & x_{16} \end{matrix}$$

– Interleaved data output for transmission:

$$x_1\ x_5\ x_9\ x_{13}\ x_2\ x_6\ x_{10}\ x_{14}\ x_3\ x_7\ x_{11}\ x_{15}\ x_4\ x_8\ x_{12}\ x_{16}$$

In this simple example, three consecutive errors in the interleaved transmitted data, let us say x_1, x_5, x_9, are translated into three isolated, single errors when interleaving is reversed at the receiver end.

More details on navigation message encoding are provided in the next chapters for each GNSS. For a more general description of channel encoding techniques, see [69].

1.8.3 Message Parameters

Once synchronized, our receiver can start interpreting the data transmitted by the satellite. The satellite data are generally structured in a *navigation message* that is repeated continuously for all users worldwide. The most relevant information that the receiver needs are the satellite position and time. This information is conveyed through what GNSS call satellite *ephemeris*, which include the satellite orbital parameters.

Satellite Orbits

Most GNSS satellites are describing almost circular orbits around the Earth. Their trajectories, as with any other bodies in space, follow Newton's Law of Universal Gravitation[32]:

$$F_{1,2} = \frac{Gm_1m_2}{r^2}, \tag{1.119}$$

where $F_{1,2}$ is the gravitational force between the two bodies, m_1 and m_2 are their masses, r is the distance between them, and G is the universal gravitational constant ($G = 6.674 \cdot 10^{-11}\ \mathrm{m^3\ kg^{-1}\ s^{-2}}$). Dividing the force by the satellite mass, we obtain the satellite acceleration, and rearranging terms and expressing Newton's law in vectors we obtain the following:

[32] Particularized here for a two-body problem with masses m_1 and m_2.

$$\ddot{\mathbf{r}} + \frac{\mu\mathbf{r}}{r^3} = 0, \tag{1.120}$$

where μ is G times the Earth mass ($5.972 \cdot 10^{24}$ kg), or $3.98 \cdot 10^{14}$ m^3/s^2, also known as the *Earth's gravitational constant*; and **r** is the vector from the Earth center of mass to the satellite center of mass.[33] The expression in Eq. (1.120) is usually called the *fundamental orbital differential equation*. It already encompasses Kepler's three laws which, prior to Newton, described the planetary orbits as ellipses sweeping a constant area around one of its focus with a period proportional to $a^{3/2}$ (their semimajor axis at the power of 3/2). A body that follows a trajectory as per Newton's law will also fulfil Kepler's laws. However, for convenience, orbits are often described according to their so-called *Keplerian elements*, which are the following:

– the orbit semimajor axis (a),
– the orbit eccentricity (e),
– the orbit inclination with respect to the equatorial plane (i),
– the right ascension of the ascending node (RAAN, Ω), or angle between the point where the satellite orbit crosses to the northern hemisphere and the vernal equinox,[34] a reference point in the celestial sphere; it is also called *Longitude of the Ascending Node* (LAN),
– the argument of the perigee (ω), or the angle between the equatorial plane and the semimajor axis (perigee direction),
– the true anomaly (ν), which defines where the satellite is in the orbit; in particular, ν defines the angle between the perigee and the satellite, as seen from the Earth's center of mass.[35]

The first two parameters, a and e, describe the ellipse that the orbit follows. The second three (i, Ω, ω), describe how the orbit is positioned with respect to the Earth. The last parameter (ν) describes the location of the satellite in the orbit at a given *time*, which is sometimes considered as the seventh parameter.

The navigation message provides mostly these parameters, although with some alterations and additions. The complete table of parameters is provided in Table 1.3. This table provides first the time of applicability of the corrections, then the six (modified) Keplerian parameters, and finally, some linear and periodical corrections to the orbits.

[33] Note that, while the orbit parameters give the position at the center of mass of the satellite, the signal departs from the antenna phase center, which is some decimeters apart. The navigation message transmits the position of the antenna phase center, but high accuracy applications generally calculate the center of mass and correct this offset, and to correct it, they also need the satellite attitude and to compensate for antenna phase center offset variations.

[34] The vernal equinox can be defined by the intersection between the ecliptic, or rotation plane of the Earth, and the equatorial plane at the equinoxes. It used to point at the Ares constellation, therefore the symbol representing it in Figure 1.35. RAAN always refers to the vernal equinox, but the term *longitude* in LAN may relate to the usual longitude used in Earth-centered Earth-fixed systems, which relates to the Greenwich meridian, or another one, depending on how it is defined.

[35] Note that ν here does not refer to the signal Doppler shift used in the previous signal processing sections, but we use it here again in order to follow the standard Keplerian nomenclature.

Table 1.3 Ephemeris parameters.

t_{0e}	ephemeris reference time
$\sqrt{a}$	square root of the semimajor axis
e	eccentricity
i_0	inclination angle at reference time
Ω_0	right ascension of ascending node at reference time
ω	argument of perigee
M_0	mean anomaly at reference time
Δn	mean motion difference from computed value
$\dot{i}$	rate of change of inclination angle
$\dot{\Omega}$	rate of change of right ascension
C_{uc}, C_{us}	amplitude of cosine and sine corrections to argument of latitude
C_{rc}, C_{rs}	amplitude of cosine and sine corrections to orbit radius
C_{ic}, C_{is}	amplitude of cosine and sine corrections to inclination

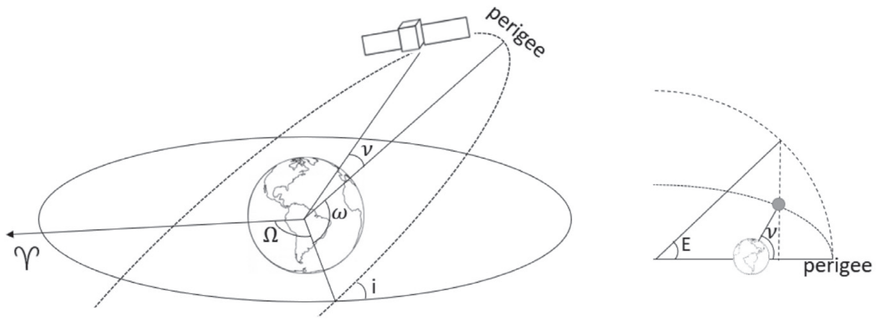


Figure 1.35 Satellite orbital parameters. Left: Keplerian parameters. Right: True anomaly versus eccentric anomaly.

Figure 1.35 shows the Keplerian parameters. The left plot depicts a satellite in its orbit, with respect to the Earth, while the right plot depicts the true anomaly ν and the eccentric anomaly E. For more details on satellite and GNSS orbits, see [70] or [4]. You can also consult the SIS ICDs of GPS [2], Galileo [71] or BeiDou [72], implemented in our MATLAB receivers.[36]

Satellite Position Computation

Here, we explain how to determine the satellite position at a given time t from the ephemeris in the navigation message (t_{0e} in Table 1.3). This is the process that is applied to GPS and Galileo ephemerides according to their signal specifications [2] [71].

Our first task is to calculate the true anomaly at the reference time t, or ν_t. Then we calculate the x, y satellite position in the orbital plane. Finally, we rotate the satellite position into an Earth-centered Earth-fixed (ECEF) coordinate system.

[36] GLONASS navigation message provides the satellite positions in a different format [73].

The GNSS Keplerian parameters do not transmit the true anomaly ν but the *mean anomaly* at reference time, or M_0. The advantage of M_0 is that it allows to linearly extrapolate to the mean anomaly, M, at our time of interest. The disadvantage is that it requires to convert to the *eccentric anomaly*, E, depicted in Figure 1.35, and then to ν. For this conversion, we need to use the semimajor axis a and the eccentricity $e = \sqrt{1 - a^2/b^2}$, where b is the semiminor axis. We also need to calculate the mean motion n, which is constant, but it is affected by Δn, provided in Table 1.3. We start by calculating the mean anomaly M_t at our target time t (1.121).[37] Then, we estimate the eccentric anomaly E_t by iteration in Eq. (1.122). With E_t we calculate the true anomaly ν_t:

$$M_t = M_0 + \sqrt{\frac{\mu}{a^3}}(t - t_{0e}) \tag{1.121}$$

$$M_t = E_t - e\sin(E_t) \tag{1.122}$$

$$\nu_t = 2\,\mathrm{atan}\Big(\sqrt{\frac{1+e}{1-e}}\tan\Big(\frac{E_t}{2}\Big)\Big). \tag{1.123}$$

Once the satellite is located in the orbital plane by ν_t, its position can be expressed in rectangular coordinates. For that, we first add the argument of perigee ω to get the argument of latitude Φ_t:

$$\Phi_t = \nu_t + \omega. \tag{1.124}$$

The receiver needs now to apply harmonic corrections from the message. These corrections are due to the gravitational force of third bodies not taken into account in our model, like the Sun and the Moon, the solar radiation pressure, and the nonuniformity of the Earth's gravitational field. We call the result the *corrected argument of latitude*, u_t:

$$\delta u_t = C_{us}\sin(2\Phi_t) + C_{uc}\cos(2\Phi_t) \tag{1.125}$$

$$u_t = \Phi_t + \delta u_t. \tag{1.126}$$

We now calculate the orbital radius r_t, to which we also add harmonic corrections from the broadcast message:

$$\delta r_t = C_{rs}\sin(2\Phi_t) + C_{rc}\cos(2\Phi_t) \tag{1.127}$$

$$r_t = a(1 - e\cos E_t) + \delta r_t. \tag{1.128}$$

With r_t and u_t we can express the satellite position in rectangular coordinates in the orbital plane, or *perifocal* coordinates:

$$x'_t = r_t\cos u_t \tag{1.129}$$

$$y'_t = r_t\sin u_t. \tag{1.130}$$

Now, the satellite position must be rotated into an Earth-centered inertial (ECI) reference frame, that is, a coordinate system centered at the Earth that does not take

[37] Note the appearance of Kepler's third law on the right side, second term.

Table 1.4 Clock correction parameters.

t_{0c}	clock correction reference time
a_{f0}	clock bias correction
a_{f1}	clock drift correction
a_{f2}	clock drift rate correction

Earth rotation into account, and then into an ECEF reference frame, solidary to the Earth rotation. For that, we need first to obtain the corrected inclination i and the corrected argument of latitude Ω_t, which already takes into account the Earth rotation angular speed, $\dot{\Omega}_e = 7.2921151467 \cdot 10^{-5}$ rad/s, to convert from ECI to ECEF:

$$\delta i = C_{is}\sin(2\Phi_t) + C_{ic}\cos(2\Phi_t) \tag{1.131}$$

$$i = i_0 + \dot{i}(t - t_{0e}) + \delta i \tag{1.132}$$

$$\Omega_t = \Omega_0 + (\dot{\Omega} - \dot{\Omega}_e)(t - t_{0e}) - \dot{\Omega}_e t_{0e}. \tag{1.133}$$

Finally, we can rotate our satellite position from perifocal coordinates and obtain our final ECEF position x_t, y_t, z_t at the time of interest t:

$$x_t = x_t' \cos(\Omega_t) - y_t' \cos(i)\cos(\Omega_t) \tag{1.134}$$

$$y_t = x_t' \sin(\Omega_t) + y_t' \cos(i)\cos(\Omega_t) \tag{1.135}$$

$$z_t = y_t' \sin(i). \tag{1.136}$$

Satellite Clock Corrections

As important as the satellite position is the satellite time. Even if the satellites have extremely accurate atomic clocks, they have deviations with respect to the GNSS time. These clock deviations usually remain low, but they get into the range measurement (a one-microsecond clock error would add an error of 300 m) and therefore need to be provided in the navigation message, so the receiver can correct them.[38] They are usually expressed as a polynomial with fixed, linear and quadratic parameters, according to Table 1.4. Notice that the time of applicability of the clock corrections, t_{0c}, usually coincides with that of the orbital parameters, t_{0e}.

The clock corrections are applied as follows:

$$dt(t) = a_{f0} + a_{f1}(t - t_{0c}) + a_{f2}(t - t_{0c})^2 + \Delta t_{\text{rel}}, \tag{1.137}$$

where $dt(t)$ is the clock correction to be applied at time t; $a_{f0,f1,f2}$ are the correction coefficients and Δt_{rel} is the *relativistic correction term*, an additional time error due to relativity.

Relativistic Effect

The theory of relativity affects GNSS in various ways: first, the satellite clocks must be adjusted to account for it. Due to their speed, and the special relativity, the clocks run slightly faster in orbit than on Earth, where they are built, but due to their lower

[38] For a performance characterization of clocks on-board GNSS satellites, see [74] or [11, Chapter 7].

gravitational force, they run at a slightly slower pace in orbit than on Earth. The net effect is a slight slowdown of a few tens of microseconds per day, depending on the GNSS orbit (e.g. 38 μs per day for GPS orbits [12]). This is already accounted for in the satellite clock manufacturing process, so the receiver does not have to care about it. However, this assumes a perfectly circular orbit. When the orbits have some eccentricity, the relativistic correction term needs to be taken into account, depending on where the satellite is located in its orbit:

$$\Delta t_{\mathrm{rel}} = F e \sqrt{a} \sin(E_t), \tag{1.138}$$

where e is the eccentricity, a is the semimajor axis, E_t is the eccentric anomaly at t calculated from the Keplerian orbital parameters and F is a constant:

$$F = \sqrt{\mu}/c^2 = -4.442807309 \cdot 10^{-10} \quad (\mathrm{s/m}^{1/2}), \tag{1.139}$$

where μ and c are the Earth gravitational constant and the speed of light, respectively.

Broadcast Group Delays

A sometimes omitted term in the position computation is the satellite broadcast group delay also referred to as Timing/Total Group Delay (TGD), instrumental delay, satellite inter-frequency bias (IFB), differential code bias or hardware bias. It encompasses the time biases in the generation of different signals due to the satellite hardware. Many GNSS, GPS and Galileo among them, provide their clock correction not referring to L1 C/A or E1, but to measurements generated in the receiver by combining two frequencies, L1 and L2 in GPS, and E1 and E5b in Galileo, into the so-called *ionosphere-free* combination. The reason is that the service is formally defined for a dual-frequency user, for a better performance. The clock correction is therefore transmitted to cancel the bias in the dual-frequency combination, and the clock correction for the single-frequency user (e.g. L1 C/A) is not straightforwardly available in the navigation message. As a consequence, in order to navigate with a single-frequency, the receiver needs to subtract the inter-frequency bias to the broadcast clock correction. Therefore, a user navigating with a frequency F_1 will need to apply this additional bias:

$$dt(t)_{F_1} = dt(t)_{F_1,F_2} - B_{F_1,F_2}, \tag{1.140}$$

where B_{F_1,F_2} is also broadcast in the navigation message, or a term allowing to calculate it following the signal specification; and $dt(t)_{F_1,F_2}$ is the term calculated in Eq. (1.137). This is because, as mentioned earlier, the clock corrections of Table 1.4 have been adapted so that these biases cancel when the receiver applies the F_1 and F_2 iono-free combination. For more details on multifrequency navigation and generating ionosphere-free measurements, see Eq. (8.11) and Eq. (8.12) in Chapter 8.

Other Formats and Navigation Data

These parameters and corrections are used nowadays by GPS and most GNSS. However, there are multiple ways to provide the satellite position. For example, GLONASS transmits the satellite position coordinates, velocity and acceleration perturbations, as discussed later in Chapter 3, and Galileo has recently started providing a short ephemeris message in similar, yet reduced format [75].

The navigation message also includes other parameters supporting the receiver. One of them is an ionospheric model (Klobuchar for GPS, NeQuick for Galileo), which corrects around half of the ionospheric error and is outlined later. Another part of the message is the *almanac*. The almanac is a long message by which each satellite transmits the rough (at the few-kilometer error level) position of all the satellites in the constellation, with the purpose of speeding up the acquisition time: If a receiver has a rough approximation of its own position, and that of the satellites, it will not search for satellite PRNs that are not visible. Almanacs are not widely used nowadays, but they are still transmitted for legacy reasons. Other information includes an indicator of the accuracy of the transmitted signal and its parameters, named URA (User Range Accuracy) or SISA (Signal-In-Space Accuracy) for GPS and Galileo, respectively, other satellite flags reporting general satellite and data health or the offset between the GNSS time and other time references such as UTC (Universal Time Coordinated).[39] Galileo will also transmit data authentication, also known as OSNMA (Open Service Navigation Message Authentication) [76].

1.9 Pseudorange Errors

Once the navigation data are decoded, the receiver can determine the coordinates of the satellite position as well as the time at which the signal is transmitted. The receiver can then estimate the exact time of arrival (ToA) of the signal. Based on the time difference between the transmission and reception time, the receiver can compute the distance between the satellite and the receiver by multiplying the time difference by the speed of light. However, this is not the actual distance, but a *pseudodistance*, as it includes also the offset in the receiver clock. The *pseudorange* is therefore the distance between a satellite and a receiver including the receiver clock offset. It is composed by the integer number of codes since a bit transition in the message, that the receiver can time-tag once it is synchronized, and the fractional code delay measurement. The reason we speak of pseudoranges rather than ranges is therefore because the quantity includes a receiver clock offset which is common to all pseudoranges. Together with the satellite position, it is one of the unknowns to be solved in our system of equations, as described later.

The pseudorange also contains, together with the true geometric distance and the receiver clock bias, some additional distance errors related to propagation effects

[39] UTC is useful to align the reference time with some standards, such as the National Marine Electronics Association (NMEA) standard, which is used by many applications far beyond the maritime domain.

due to the ionosphere, the troposphere, local effects in the vicinity of the receiver leading to multipath errors and receiver noise. This is expressed in Eq. (1.141) as follows:

$$\rho_m^{(p)} = r^{(p)} + c(dt_u - dt^{(p)}) + I^{(p)} + T^{(p)} + \varepsilon^{(p)}, \tag{1.141}$$

where $\rho_m^{(p)}$ is the *measured* pseudorange for satellite (p), before applying any correction; $r^{(p)}$ is the range, or actual distance between the satellite and the receiver; c is the speed of light; dt_u is the user receiver clock bias; $dt^{(p)}$ is the satellite clock bias $I^{(p)}$ is the range ionospheric error; $T^{(p)}$ is the tropospheric error and $\varepsilon^{(p)}$ contains other unmodelled measurement errors including multipath, interference and noise. These errors are described in the next subsections.

1.9.1 Ionospheric Error

The ionosphere, as the name suggests, is a layer of the atmosphere that contains electrically charged, or *ionized*, particles that affect the propagation of electromagnetic waves. It can be the most significant source of errors in GNSS measurements, up to a hundred meters. The ionospheric variations depend on the solar activity, which oscillates in cycles of approximately 11 years. The last solar maximum was in 2014 when writing this text. At a solar maximum, there is a higher amount of solar spots and flares, and a higher amount of ionized particles arrive to the Earth and remain in the ionosphere. Therefore, the ionosphere is active during the day, but in some cases, ionospheric perturbations occur at or after sunset. The particles are also affected by the Earth's magnetic field, and therefore, activity is higher close to the poles and the geomagnetic equator. Ionospheric activity is measured by the Total Electron Content (TEC), which is usually expressed in TECU (TEC Units), or total number of electrons per square meter.

Ionosphere affects the signals in two different ways. Ionospheric *refraction* induces a group delay to the ranging code modulation and a carrier phase advance of the same magnitude [4, Chapter 5]. Its effect is present everywhere within the atmosphere. In contrast, ionospheric *scintillation* causes rapid signal power fading and phase fluctuations, and it is usually encountered at equatorial and polar latitudes. In this section, we describe approaches to mitigate the ionospheric refraction; scintillation is not addressed here, but the interested reader may refer to [77] for more information. In the remainder of this section, we will refer to the ionospheric refraction effect as a delay, which corresponds to the refractive effect on the pseudorange measurement. The same concepts apply to the carrier phase measurement but with the sign reversed (see Eq. (8.7) for more details).

The ionospheric delay depends on the carrier frequency. The first-order ionospheric delay is proportional to TEC along the propagation path between the receiver and satellite (p) as

$$I^{(p)}(f) = \frac{40.3\ \mathrm{TEC}^{(p)}}{f^2}. \tag{1.142}$$

Higher order ionospheric delay effects exist, but their contribution on the ionospheric delay is in the order of centimeters so they can be neglected except for high accuracy applications [78].

There are three main approaches to compensate for the ionospheric delay: evaluating an ionosphere model using the parameters transmitted by the satellites, exploiting the frequency dependence of Eq. (1.142) with a dual-frequency receiver and using external correction data. The first method is outlined here. The second one is described in Chapter 8, which covers a dual-frequency receiver, and some more details about the ionosphere. Some references for the third one are provided in Chapter 7.

Broadcast Ionospheric Correction Models

For single-frequency users, the most straightforward way for mitigating ionospheric delays is to apply the correction parameters broadcast as part of the navigation message. GPS and BeiDou support the Klobuchar ionosphere model [79], whereas Galileo supports the NeQuick G algorithm [80]. This model is a variant of the NeQuick electron density model [81], which has specifically been adapted for GNSS. In addition to the ionosphere parameters, broadcast models require the receiver and satellite positions as well as the time of measurement as inputs.

The ionospheric data transmitted by GNSS satellites are based on predictions and not updated in real time. It is also transmitted in very few bits, for example, 41 and 64 bits for Galileo and GPS, respectively. Therefore, the user cannot expect them to compensate for the ionospheric error perfectly. For instance, the GPS ionospheric corrections are estimated to reduce the RMS error by at least 50% [2], whereas the NeQuick G model has been designed to correct 70% or more of RMS ionospheric delays under regular conditions [80]. Since the ionospheric delay in Eq. (1.142) depends only on the TEC value and the carrier frequency, any model to estimate the TEC value can be used with any GNSS constellation. Consequently, Galileo-based NeQuick G corrections can be used with GPS and BeiDou. Similarly, Klobuchar corrections can be applied to Galileo signals instead of NeQuick G.

The two different approaches are illustrated in Figure 1.36. The Klobuchar model takes eight input parameters $\{\alpha_i, \beta_i\}_{i=0\ldots3}$ that characterize the amplitude and period of the vertical ionospheric delay. It models the ionosphere as a thin shell at a height of 350 km above the Earth surface. The correction algorithm determines the point where the LOS ray from the satellite to the receiver pierces the shell, computes the vertical ionospheric delay at the piercing point and adjusts this estimate based on the satellite elevation angle to obtain the slant ionospheric delay. A detailed description of the algorithm is available in [2].

NeQuick G is fundamentally different from the thin-shell Klobuchar model supported by GPS and BeiDou. NeQuick G is a full model of the ionosphere consisting of three layers (E, F1, F2), allowing the evaluation of the local electron density at an arbitrary point. Therefore, the TEC estimate is obtained by numerically integrating the electron density along the path from the satellite to the receiver antenna. In general, the NeQuick G algorithm is computationally much heavier than the Klobuchar, and

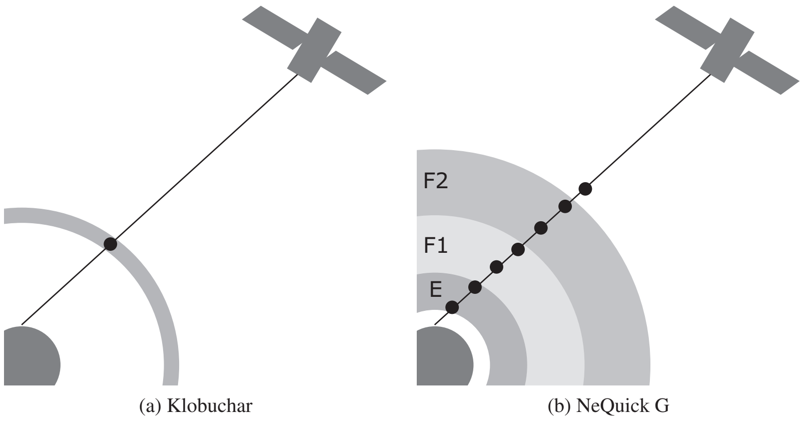


Figure 1.36 Principles of ionosphere compensation algorithms (illustrations not to scale). Black dots denote points where the delay is evaluated; note that in (b), the amount and spacing of evaluation points are determined by the numerical integration algorithm.

for this reason, it is recommended to update the TEC estimates at a low rate, such as every 30 seconds [80].

The Galileo navigation message provides three NeQuick G parameters (a_0, a_1, a_2) which are the coefficients of a second-order polynomial to evaluate the effective ionization level. In addition, the receiver must have access to two sets of precomputed data: a grid to compute the modified dip latitude and a set of spherical harmonic coefficients to characterize the monthly median transmission factor and critical frequency of the F2 layer. For details, please refer to the NeQuick G algorithm specification [80].

1.9.2 Tropospheric Error

The troposphere is the lowest layer of the atmosphere. It is between the Earth surface and about 9–10 kilometers high. Unlike the ionosphere, the troposphere is nondispersive, that is, there is no frequency-dependent effect. The temperature, pressure and humidity will, however, alter the velocity of radio waves. The resulting delay highly depends on the distance travelled through the troposphere. Since signals from low-elevation satellites travel a longer distance than a signal from a satellite at zenith, we often model the tropospheric delay using a zenith delay and a mapping function. The zenith tropospheric delay is about 2.4 m, and the tropospheric delay can be as high as 10 m for low-elevation satellites. Applying relatively simple tropospheric models can reduce these errors to below 0.5 m.

The total tropospheric delay in the direction of a satellite can be divided into two parts: the “dry,” hydrostatic component, and the “wet”, nonhydrostatic component. These two delays are then projected onto the zenith direction using some mapping functions, for example, using the Niell model [82]:

$$\mathrm{SPD}(el) = m_{\mathrm{HD}}(el)\,\mathrm{ZHD} + m_{\mathrm{WD}}(el)\,\mathrm{ZWD}, \tag{1.143}$$

where SPD(el) is the slant path delay in meters along elevation angle el, ZHD is the zenith hydrostatic delay in meters, ZWD is the zenith wet delay in meters and $m_{\mathrm{HD}}(el)$ and $m_{\mathrm{WD}}(el)$ are the mapping functions of the hydrostatic and wet path delay along elevation angle el. The model proposed in [82] is currently used as the mapping function in the multi-GNSS SDR receiver accompanying this book. Another tropospheric model commonly used is the UNB3 model [83], used in the SBAS Minimum Operational Performance Standards (MOPS) of the Radio-Technical Commission for Aeronautics (RTCA) [84]. Further information about the troposphere and its models is provided in [85].

1.9.3 Multipath

Multipath is the effect of the reflections of the satellite signal in objects around the receiver. These reflections may be received and tracked by the receiver and the consequence is that our correlation peak is distorted, and this distortion leads to an error in the code delay measurement. The multipath error depends on the amplitude and distance of the reflections. More than one reflection can be received by the receiver, and this is generally the case in reality. The multipath error observed in the measurement depends on many factors: the signal tracked, the amplitude, the distance and the carrier phase of the reflection, the receiver bandwidth, and the correlator spacing. Even if the multipath echoes arrive always later than the original signal, they can lead to a negative delay if the carrier phase is rotated 180° with respect to the original signal. This is usually measured in the multipath error envelope (MEE) plot, treated in more detail in Chapter 10, where an example of MEE is provided (Figure 10.12).

Generally, mitigation techniques are effective in the case of LOS multipath, that is, when both the satellite-to-receiver LOS and the echoes are received. When only the echoes are received, this situation is referred to as non-line of sight (NLOS) multipath, and its effect may be more detrimental in the receiver than LOS multipath, especially if the reflection arrives with a delay of tens, or even hundreds of meters. Further details on state-of-the-art multipath detection and mitigation techniques and their performance can be found in Chapter 10. A low-complexity multipath detection method is also proposed in Chapter 9 for snapshot receivers. For more details on multipath, see [10, Chapter 9].

1.9.4 Interference

GNSS RFI can be defined as unwanted energy that gets into the GNSS receiver, excluding natural causes such as electromagnetic radiation from the sky or the ground, which we consider as noise. Typical GNSS interference can come from multiple sources, such as radars, aviation equipment, jammers, spoofers, GNSS repeaters, solar bursts, energy spillover from signals in other bands, such as TV signals, or even self-interference if the receiver has several RF devices, as it is the case for smartphones. There are multiple ways to classify interference, depending on the interferer's intention (intentional, nonintentional, collateral), whether it's natural (e.g. solar bursts) or man-made, the signal waveform (e.g. pulsed, narrow-band, wideband, chirped, matched spectrum...), or the impact on the receiver (e.g. signal degradation, service denial, or service integrity failure) [86]. For more details, [87] presents a thorough description of interference, including their mathematical representation and countermeasures. In our book and related software, we will not develop specific interference mitigation or detection techniques, although SDRs are powerful platforms for developing and improving such techniques thanks to their flexibility.

1.9.5 Receiver Noise

We refer to receiver noise as all other effects that contribute to the measurement error which are not described before. They include the electromagnetic noise radiated by the sky and the ground that reach the antenna together with the signal, and the thermal noise added by the receiver. The combination of these noise sources are represented by the noise power in Eq. (1.44). The signal filters also add a delay to our measurement. This delay can be considered noise as well, although, if constant, it is estimated as part of the receiver clock bias. If it is frequency-dependent, it is considered as a receiver group delay or hardware bias and can be estimated separately or removed with the ionospheric delay. Other noise effects relate to the sampling and quantization of the signal, described previously in Section 1.4.3. More details on receiver noise components can be found in [12, Chapter 6].

1.9.6 Error Budgets

Once we have introduced all the errors, we present in Table 1.5 the error budgets or contributions to the pseudorange measurement. The purpose of the table is to provide the reader a grasp of the different error contributions, but the error magnitudes can vary significantly depending on many factors. Satellite orbit and clock error contributions are gradually reduced due to the improvements in satellite on-board clocks and ground orbit determination capabilities. Ionosphere and multipath are very variable, as described before, and multipath and noise highly depend on the receiver signal and measurement processing. Note also that the error contribution can be composed of a fixed term (bias) and a random term (variance). Finally, the table illustrates the error

Table 1.5 Approximate average error contributions to the pseudorange measurement. Some of them can significantly vary and depend on many factors.

Source	Typical Error (m)	Comments
Satellite orbit	0.2–0.5	Gradually reduced with GNSS modernization.
Satellite clock	0.2–0.5	Gradually reduced with GNSS modernization.
Ionosphere	2.5–5	Very variable. 5–10 m before navigation message correction.
Troposphere	0.5	Around 3–10 m before receiver model correction.
Multipath	1–2	In open-sky conditions. Very variable and up to around 20 m in degraded conditions or higher if NLOS.
Receiver noise	0.5	Includes receiver hardware noise, filtering, sampling and quantization.

in the *corrected* pseudorange ρ, while the error in the *measured* pseudorange ρ_m from Eq. (1.141) is quantified in the right column, where pertinent. Further considerations on error modeling are provided in section 1.10.4. For a more exhaustive measurement error characterization, see [4, Chapter 5], [10] or [11].

1.10 Computation of Position and Time

The principle of GNSS positioning is simple. By knowing the position of the satellites and the satellite-receiver distance, or range, as the signals travel at the speed of light, the receiver can compute its position. In order to calculate a 3D position, three exact range measurements would be required, but the receiver does not know its own time, so it cannot calculate the exact time of arrival (ToA). The receiver time uncertainty, or bias b, is common to all range measurements, so it is calculated as an additional unknown. This means we need at least four pseudoranges to build a system of equations, as follows:

$$\rho^{(p)} = \sqrt{(x^{(p)} - x)^2 + (y^{(p)} - y)^2 + (z^{(p)} - z)^2} + b + e^{(p)}, \tag{1.144}$$

where the unknonws are in the vector $\mathbf{x} = (x, y, z, b)$, that we call the *state vector*, $(x^{(p)}, y^{(p)}, z^{(p)})$ are satellite p's coordinates, and b is the receiver clock bias, in meters $(b = dt_u c)$. When we have four measurements, we can solve the system. Note that $\rho^{(p)}$ is the *corrected* pseudorange after removing the errors we can estimate: ionosphere and satellite clock bias from the navigation message, and troposphere from our model. However, $\rho^{(p)}$ still has an error $e^{(p)}$ that includes everything that has not been corrected yet.

1.10.1 Linearization

The system of equations from Eq. (1.144) when $(p) = 1, \ldots, 4$ is nonlinear, so it has to be linearized. We will explain the linearization process algebraically and then geometrically.

Algebraic Explanation

In order to solve a nonlinear system of equations, it is typical to use the *Newton*, or *Netwon-Raphson*, approximation, which consists of approaching the value of a function at a certain point by its value in the proximity and its derivative multiplied by an increment. A function can be expressed as a series of powers, according to a Taylor series, which also depend on the function derivatives. In our case, the function is the pseudorange $\rho^{(p)}$, which depends on our state vector $\mathbf{x} = (x, y, z, b)$, and the Taylor series is expanded as follows:

$$\rho^{(p)} = \rho^{(p)}(\mathbf{x}_0) + \frac{\partial \rho^{(p)}(\mathbf{x}_0)}{\partial \mathbf{x}} \Delta\mathbf{x} + \frac{1}{2!} \frac{\partial^2 \rho^{(p)}(\mathbf{x}_0)}{\partial^2 \mathbf{x}} \Delta\mathbf{x}^2 + \ldots \tag{1.145}$$

where $\rho^{(p)}(\mathbf{x}_0)$ is the pseudorange between the satellite position and our initial position estimate $\mathbf{x}_{0,(1:3)}$ (note that full initial state vector $\mathbf{x}_0$ includes also the bias in the fourth term). We consider sufficient to take the first derivative, as later we will perform several iterations. Thus, and developing Eq. (1.145) into the partial derivatives on the variables of our four-state vector, we have:

$$\rho^{(p)} \approx \rho^{(p)}(\mathbf{x}_0) + \frac{\partial \rho^{(p)}(x_0)}{\partial x}\Delta x + \frac{\partial \rho^{(p)}(y_0)}{\partial y}\Delta y + \frac{\partial \rho^{(p)}(z_0)}{\partial z}\Delta z + \frac{\partial \rho^{(p)}(b_0)}{\partial b}\Delta b. \tag{1.146}$$

If we perform the partial derivatives and slightly rearrange Eq. (1.146), we obtain the following:

$$\rho^{(p)} - \rho^{(p)}(\mathbf{x}_0) \approx \frac{x - x^{(p)}}{r^{(p)}}\Delta x + \frac{y - y^{(p)}}{r^{(p)}}\Delta y + \frac{z - z^{(p)}}{r^{(p)}}\Delta z + \Delta b, \tag{1.147}$$

where $r^{(p)} = r^{(p)}(\mathbf{x}_0) = \sqrt{(x^{(p)} - x_0)^2 + (y^{(p)} - y_0)^2 + (z^{(p)} - z_0)^2}$. The terms applying to the user position coordinates $(\Delta x, \Delta y, \Delta z)$ can be expressed as the unitary vector $\mathbf{e}^{(p)}$ that points from the estimated position solution $\mathbf{x}_0$ to satellite (p) position, $\mathbf{x}^{(p)}$, with opposite sign:

$$\mathbf{e}^{(p)} = \left(\frac{x^{(p)} - x}{r^{(p)}}, \frac{y^{(p)} - y}{r^{(p)}}, \frac{z^{(p)} - z}{r^{(p)}}\right) = \frac{\mathbf{x}^{(p)} - \mathbf{x}_{0,(1:3)}}{|\mathbf{x}^{(p)} - \mathbf{x}_{0,(1:3)}|}. \tag{1.148}$$

Finally, our system of equations can be expressed in a matrix form as follows:

$$\boldsymbol{\Delta\rho} = \begin{bmatrix} -\mathbf{e}^{(1)} & 1 \\ \dots & \end{bmatrix} \cdot \boldsymbol{\Delta}\mathbf{x}, \tag{1.149}$$

where $\boldsymbol{\Delta}\mathbf{x} = \mathbf{x} - \mathbf{x_0}$ and $\boldsymbol{\Delta\rho}$ is a vector of differences between the corrected pseudoranges and the theoretical estimation based on the initial position. For satellite (p):

$$\Delta\rho^{(p)} = \rho^{(p)} - \rho^{(p)}(\mathbf{x}_0). \tag{1.150}$$

Note that, in some other references, $\rho^{(p)}(\mathbf{x}_0)$ is referred as $\hat{\rho}^{(p)}(\mathbf{x}_0)$, or just $\hat{\rho}^{(p)}$, to highlight that it is a *theoretical* estimation of the pseudorange. The linear system in Eq. (1.149) leads to a recursion where the new receiver position is used as initial position $\mathbf{x_0}$ for the next iteration, and so on, until the position improvement converges. Usually, around four iterations are sufficient, and more iterations do not improve the result. We will call the matrix with the unitary vectors as the *geometry matrix*, also known as *observation matrix*, or *design matrix*, and it will be denoted by $\mathbf{A}$:

$$\boldsymbol{\Delta\rho} = \mathbf{A} \cdot \boldsymbol{\Delta}\mathbf{x}. \tag{1.151}$$

For more information on the linear algebra behind the satellite navigation positioning equations, see [88].

Geometric Explanation

Our geometric interpretation of the positioning problem is depicted in Figure 1.37, again for satellite (p). For our explanation, we reduce the state vector to the position vector (x, y, z). In the figure, we start from an initial position vector $\mathbf{x}_0$ and we want

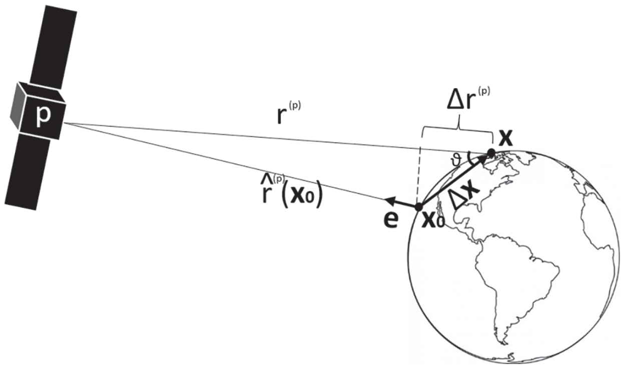


Figure 1.37 Linear relationship between measurements and position.

to obtain vector $\mathbf{\Delta x}$ toward $\mathbf{x}$, the true position, through a linear relationship with our pseudoranges. We define the delta range for satellite (p) as

$$\Delta r^{(p)} = r^{(p)} - \hat{r}^{(p)}(\mathbf{x}_0). \tag{1.152}$$

Then, as shown in Figure 1.37:

$$\Delta r^{(p)} \approx |\mathbf{\Delta x}| \cos\theta, \tag{1.153}$$

which can also be expressed as the dot product of vectors $-\mathbf{e}^{(p)}$ and $\mathbf{\Delta x}$,

$$\Delta r^{(p)} \approx -\mathbf{e}^{(p)} \cdot \mathbf{\Delta x} = |\mathbf{\Delta x}| \cos\theta. \tag{1.154}$$

We use $\approx$ in Eq. (1.153) because of the linearization error, which is reduced at each iteration.

We have now established our linear relationship between measurements and position. We add the bias term b to the expression and obtain the system of equations from Eq. (1.149).

1.10.2 Least Squares and Weighted Least Squares

When we have four pseudorange measurements, we can solve our four unknowns by calculating the inverse of matrix $\mathbf{A}$ for each iteration:

$$\mathbf{\Delta x} = \mathbf{A}^{-1} \cdot \mathbf{\Delta}\rho, \tag{1.155}$$

where $(\cdot)^{-1}$ is the inverse matrix operator. However, even after applying the ionospheric and tropospheric models and satellite clock corrections, our measurements still have errors, and these errors will propagate into the position error. In order to minimize this error, we want to use more measurements. When we have more than four measurements, our system is overdetermined, meaning that it has many possible solutions for our state vector. What we want is that the chosen solution minimizes the position error. In order to minimize this position error, we use the *least-squares* estimator, which computes the state vector that minimizes the sum of the squares of the residual measurement errors with respect to our solution, that is, the differences between the measured and theoretical pseudoranges for our solution ($\Delta\rho$ in the last iteration). When applying least squares, our iterative system of equations is expressed as

$$\mathbf{\Delta x} = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\Delta\rho, \tag{1.156}$$

where $(\cdot)^T$ is the transpose matrix operator. Sometimes the matrix $(\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T$ is referred to as the *pseudo-inverse* matrix.

Some measurements will be of higher quality than others. For example, those from a higher satellite elevation use to have a lower ionospheric and tropospheric delay, less multipath and possibly a higher signal strength due to the satellite and receiver antenna gain patterns. The estimator may *weight* the measurements so that those believed to be better have a higher weight in the position solution. In this case, we use

$$\mathbf{\Delta x} = (\mathbf{A}^T\mathbf{W}\mathbf{A})^{-1}\mathbf{A}^T\mathbf{W}\Delta\rho, \tag{1.157}$$

where $\mathbf{W}$ is the *weight matrix*, a diagonal $(m \times m)$ matrix, where m is the number of measurements and where each coefficient in the diagonal represents the weight. Usually, weights are determined depending on the satellite elevation or C/N_0, but other criteria can be used, such as the satellite type or information in the navigation message, like URA/SISA as explained later in this chapter. If the variances of the satellite errors could be estimated, its weight could be their inverse, so that $W^{(p)} = 1/\sigma^2_{(p)}$ (we will later refer to these variances as UERE^2).

Least squares is one of the simplest methods to estimate the receiver position, but it has the inconvenience that the resulting position is not smoothed in time, as it depends on the instantaneous errors of the measurements. Kalman filters [89] are often used for that purpose. Other standard algorithms use least squares, but over carrier-smoothed pseudoranges such as Hatch filters discussed in [90] or in Chapter 8 (8.18). Also, for the estimation of the receiver velocity based on the range rate measurements, see [4, Chapter 6].

1.10.3 Coordinates and Reference Frames

Our satellite position was calculated in ECEF Cartesian coordinates, so our position solution (x, y, z) is by default referred to ECEF as well. However, users are often interested in the longitude and latitude, or *geographic* coordinates. As shown in Figure 1.38, the transformation between latitude and longitude[40] (l_a, l_o) and (x, y, z) is as follows:

$$\begin{aligned} x &= R\cos(l_a)\cos(l_o) \\ y &= R\cos(l_a)\sin(l_o) \\ z &= R\sin(l_a), \end{aligned} \tag{1.158}$$

where R is the position vector module. However, we often want to express our coordinates with respect to a *reference frame*. A reference frame includes not only

[40] We avoid the usual notation of φ and λ symbols as both have been used before in this chapter.

latitude and longitude but also height, among other information. For example, GPS uses WGS84. WGS84 defines the specific values of the main constants used in the position computation: the Earth rotation rate $\dot{\Omega}_e$ and the Earth gravitational constant μ. WGS84 also defines an ellipsoid that models the Earth. This way, a position can be expressed in (l_a, l_o, h) where h is the height with respect to the ellipsoid. The ellipsoid is defined by the Earth semimajor axis (a = 6,378,137 m) and the flattening factor ($f = 1/298.257223563$). Then, l_a, l_o and h can be calculated with respect to the WGS84 ellipsoid as follows:

$$l_o = \mathrm{atan}\left(\frac{y}{x}\right) \tag{1.159}$$

$$N_\varphi = \frac{a}{\sqrt{1 - f(2-f)\sin^2(l_a)}} \tag{1.160}$$

$$l_a = \mathrm{atan}\left(\frac{z}{\sqrt{x^2+y^2}}\left(1 - \frac{(2-f)fN_\varphi}{N_\varphi + h}\right)^{-1}\right) \tag{1.161}$$

$$h = \frac{\sqrt{x^2+y^2}}{cos(l_a)} - N_\varphi. \tag{1.162}$$

Note that, once again, we have a nonlinear system to solve N_φ, l_a and h. This is usually solved by iteration.

In addition to the simple ellipsoid, WGS84 defines the *geoid* as a more irregular surface through some harmonic corrections. Further information on the WGS84 and all its parameters, which also include gravitational models, magnetic models and datum transformations, can be found in [91].

Topocentric coordinate systems are also relevant for receivers. They express coordinates with respect to the horizon, or a horizontal plane that is tangent to the Earth surface at the user location. They are particularly relevant to express errors and covariances in the horizontal plane and vertical component, as mentioned later. A usual right-handed topocentric coordinate system is ENU (East, North, Up). It is also depicted in Figure 1.38. Another one is NED (North, East Down).

Reference frames, also known as *datums*, can be different for different GNSS. Reference frame parameters are defined by a set of monitor stations precisely located on Earth. Each terrestrial reference frame, like Galileo Terrestrial Reference Frame (GTRF) for Galileo, or WGS84 for GPS, uses its own set of stations. Both GTRF and WGS84 are based on International Terrestrial Reference Frame, produced by the International Earth Rotation and Reference Systems Service (IERS), and their differences are minimal, in the order of centimeters, so they are not relevant for code-based positioning. They may be relevant though for high accuracy centimeter-level positioning, which may require reference frame transformations to avoid this error. More details on reference frames can be found in [11, Chapter 2] and [92, Chapter 2].

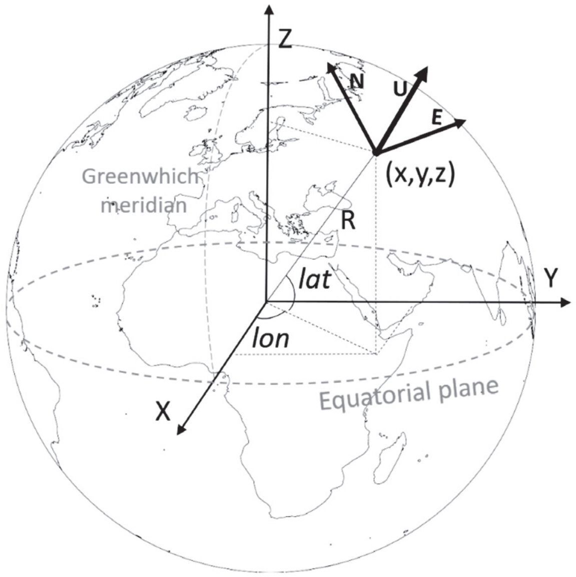


Figure 1.38 ECEF and ENU coordinate systems and conversion to geographic coordinates.

1.10.4 Dilution of Precision, Measurement Residuals and Position Accuracy

Dilution of Precision

Position accuracy depends on two independent factors: satellite geometry and measurement errors. If we have four satellites in view but they are aligned in the sky, our matrix **A** will be rank-deficient and no solution will be obtained, or at best our precision would be highly *diluted*. This is measured by the *Dilution of Precision* (DOP), and is reflected in the DOP matrix **D**:

$$\mathbf{D} = (\mathbf{A}^T\mathbf{A})^{-1} \tag{1.163}$$

D is a (4×4) matrix where the diagonal terms d_{11} to d_{44} express how good our geometry is. The advantage of DOP is that it gives an indication of the solution precision (or its dillution) without looking at the pseudorange measurements. Depending on our interest, we can look at different terms:

- $\sqrt{d_{11} + d_{22} + d_{33} + d_{44}}$: Geometric Dilution of Precision (GDOP). It expresses the overall impact of geometry in the position and time solution.
- $\sqrt{d_{11} + d_{22} + d_{33}}$: Position Dilution of Precision (PDOP). It expresses the geometry impact in the position solution.
- $\sqrt{d_{44}}$: Time Dilution of Precision (TDOP). It expresses the geometry impact in the time solution.

One limitation of these DOP indicators is that they do not allow expressing DOP in the receiver vertical and horizontal components, which is desirable. For this, we need to express **D** in a topocentric coordinate system. We can rotate our DOP matrix and express it in ENU coordinates as follows:

$$\mathbf{D}_{\text{enu},3\times3} = \mathbf{F}_{\text{xyz}}^{\text{enu}T} \cdot \mathbf{D}_{3\times3} \cdot \mathbf{F}_{\text{xyz}}^{\text{enu}}, \tag{1.164}$$

where $\mathbf{D}_{\text{enu},3\times3}$ is the (3×3) DOP matrix rotated to ENU including elements $d_{\text{enu},11}$ to $d_{\text{enu},33}$, and $\mathbf{D}_{3\times3}$ is the (3×3) DOP matrix in XYZ, including elements $d_{\text{enu},11}$ to $d_{\text{enu},33}$. We define here matrix $\mathbf{F}_{\text{xyz}}^{\text{enu}}$, that allows to convert from a Cartesian to an ENU coordinate system, as follows:

$$\mathbf{F}_{\text{xyz}}^{\text{enu}} = \begin{bmatrix} -\sin(l_o) & -\sin(l_a)\cos(l_o) & \cos(l_a)\cos(l_o) \\ \cos(l_o) & -\sin(l_a)\sin(l_o) & \cos(l_a)\sin(l_o) \\ 0 & \cos(l_a) & \sin(l_a) \end{bmatrix}. \tag{1.165}$$

Now we can estimate DOP in the horizontal plane and vertically:

- $\sqrt{d_{\text{enu},11} + d_{\text{enu},22}}$: Horizontal Dilution of Precision (HDOP). Geometry impact in the horizontal components of the position.
- $\sqrt{d_{\text{enu},33}}$: Vertical Dilution of Precision (VDOP). Geometry impact in the vertical component of the position.

VDOP and HDOP are widely used for many applications. For example, an airplane using GNSS for precision approach will be sensitive to VDOP, while a car will be more sensitive to HDOP. Generally, VDOP will be worse than HDOP, as satellites are only in the sky, that is, in the upper side in the vertical dimension. HDOP and VDOP usually range between almost one and two, where lower implies a better geometry. Thanks to multi-GNSS, geometry can be highly improved. Practical examples of HDOP and VDOP values, for different GNSS combinations, are shown in Chapter 7, Table 7.6.

User Equivalent Range Error and Covariance Matrix

The other element that drives position accuracy is the measurement error. In practice, when solving Eq. (1.149) we apply the troposphere, ionosphere and satellite clock corrections to our receiver measurement ρ_m and the satellite orbital errors to the estimated pseudorange $\hat{\rho}$ and **A**. We may apply multipath mitigation through specific tracking loop discriminators, or smoothing of the pseudorange measurements. Note also that no interference error is explicitly considered as, unlike the others, interference is sporadic or considered under the noise term. With these actions, we reduce the measurement errors as much as we can, but we still want to estimate how big these errors are. The estimations of error contributions introduced in Section 1.9 are aggregated into the *User Equivalent Range Error* (UERE), which represents the standard deviation of the total error:

$$\text{UERE} = \sqrt{\sigma_{\text{orb}}^2 + \sigma_{\text{clk}}^2 + \sigma_{\text{iono}}^2 + \sigma_{\text{tropo}}^2 + \sigma_{\text{mpath}}^2 + \sigma_{\text{noise}}^2}, \tag{1.166}$$

where σ_{orb}, σ_{clk}, σ_{iono}, σ_{tropo}, σ_{mpath}, and σ_{noise} are the standard deviations of error distributions of the satellite orbits and clocks, ionospheric and tropospheric delays, multipath and receiver noise. In Eq. (1.166), we are assuming that the variance of the sum of stochastic variables is the sum of the variances of each variable. This implies that all variables are independent, that is, each error source is uncorrelated with the rest.

Usually, UERE is divided in two components: signal-in-space ranging error (SISRE), including the error of the orbits, clocks and group delays when applied, and user equipment error (UEE), which includes everything else[41]:

$$\text{SISRE} = \sqrt{\sigma^2_{\text{orb}} + \sigma^2_{\text{clk}}} \tag{1.167}$$

$$\text{UEE} = \sqrt{\sigma^2_{\text{iono}} + \sigma^2_{\text{tropo}} + \sigma^2_{\text{mpath}} + \sigma^2_{\text{noise}}} \tag{1.168}$$

$$\text{UERE} = \sqrt{\text{SISRE}^2 + \text{UEE}^2}. \tag{1.169}$$

The assumption that errors are uncorrelated may not be correct sometimes. Also, if we want to characterize the errors by their standard deviation, we are assuming that they are zero-mean, which may also not be true. Therefore, UERE is not very representative in many cases. In spite of that, it is widely used. For example, it allows us to roughly characterize the horizontal user position error as UERE $\times$ HDOP, and the vertical position error as UERE$\times$VDOP. We can also use a generic UERE value to estimate the covariance matrix of our solution that shows the variances of the different components $(\sigma^2_x, \sigma^2_y ...)$ and their interrelationships, or *covariances*, $(\sigma_{xy}, \sigma_{xz} ...)$, as

$$\mathbf{COV}_{\text{xyz}} = \text{UERE}^2 \cdot \mathbf{D}. \tag{1.170}$$

Measurement Residuals

So far, DOP and UERE have been described independently from the actual measurements. In order to assess the goodness of our position in relation to the measurement errors, we use the measurement *residuals*. Once we obtain our solution, our residuals vector β is commonly expressed as

$$\beta = \mathbf{Ax} - \mathbf{\Delta}\rho. \tag{1.171}$$

The residuals are a good, yet limited, indicator of the position accuracy. Having a vector with low residuals is often an indicator of a good solution accuracy. It means that the measurements "agree" with the computed position. However, this is a good indicator only if the measurement errors are uncorrelated. If, for example, measurements are reflections from a building, the position may be off by much more than the residuals suggest. Therefore, measurement residuals are not a perfect indicator of the user accuracy either.

Measuring and Bounding Position Errors and Integrity

We have presented an overview of how the receiver can estimate the goodness of its position through its geometry (DOP), a-priori errors (UERE) and measurement residuals β. However, in the next chapters, the actual errors are measured with respect to the true position, which is known, instead of using UERE and DOP. When we measure

[41] Note that, in this convention, we include all propagation errors as part of the user equipment error. Note also that SISRE is sometimes referred to as signal-in-space error (SISE).

position errors, we have to do it through an unambiguous statistical characterization. For example, while RMS and standard deviation (σ) are used sometimes as synonyms, and the same occurs for 95 percentile and 2σ, there are subtleties that depend on the number of dimensions of the error used and other factors. In the next chapters, the position accuracy is provided through the horizontal RMS error, vertical RMS error and 3D RMS error. For more details on the differences on accuracy indicators, see the excellent analysis in [93]. When the user position is unknown but we want to better characterize our errors beyond an a-priori UERE, we can use the information provided by some GNSS or augmentation systems. For example, GPS and Galileo provide an indicator of the error statistical distribution in the user range accuracy (URA) and signal-in-space accuracy (SISA), respectively. URA bounds the user range error, or URE, which includes the orbit and clock errors[42] by 4.42 times with a probability of $1 - 10^{-5}$ every hour (or 5.73 times with a probability of $1 - 10^{-8}$ every hour, depending on the navigation flags). Similarly, SISA provides the standard deviation (1σ) of the zero-mean Gaussian distribution that bounds the URE. The exact definitions and further details are provided in the official GNSS service definition documents [94] [95].

Measurement errors are at the core of satellite navigation because they drive the position accuracy. Also, by modeling the error statistical distributions correctly, the user can have an estimate of the goodness of its position and the maximum bound of its error with a certain probability. This is important for critical applications, such as airplane precision approaches, which in addition to accuracy, require position *integrity*. Integrity can only be provided effectively with additional monitoring means, either on-board, through external information or combined. The GNSS integrity concept also requires that when the errors cannot be bounded, the receiver is alerted in a few seconds maximum. On-board integrity is usually called receiver autonomous integrity monitoring (RAIM). RAIM is usually based on looking at a priori and/or residuals information available, and detecting and excluding faulty measurements from the solution. Augmentations monitor the satellites and sometimes ionosphere and transmit error bounds and alert the users through satellite or local ground channels, in the case of SBAS or ground-based augmentation systems (GBAS), respectively. GPS III already provides an "integrity flag," that allows to switch from the $4.42, 10^{-5}$ to the $5.73, 10^{-8}$ URA characterization mentioned earlier. New SBAS provide the covariance matrix of each satellite, allowing a better computation of $\mathbf{COV}$ than that in Eq. (1.170). Other modern integrity systems, such as advanced RAIM (ARAIM), combine error bounds provided from the system with RAIM. Integrity is beyond the scope of this book, but for more details on RAIM, SBAS, GBAS and ARAIM, see [96], [97], [98] and [99] respectively.

[42] Note that before we defined SISRE/SISE as the one-sigma or RMS URE.

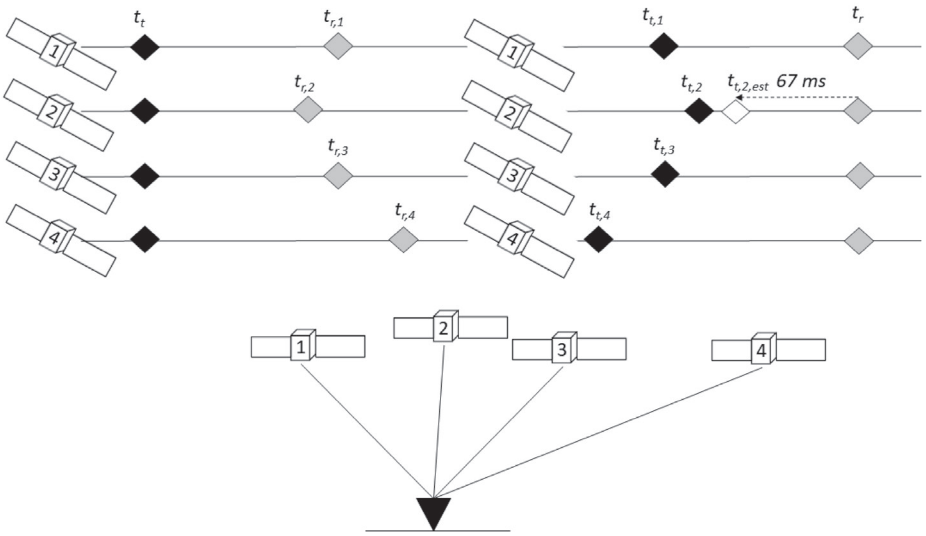


Figure 1.39 Satellite position computation for pseudoranges measured at the same time.

1.10.5 Other Practical Aspects

How to put our PVT engine to work is not so obvious in practical terms, so we will discuss some practicalities in this subsection.

Earth Rotation and Receiver Time Reference

Apart from the satellite clock offset, which can be neglected here, all satellites are synchronized and provide the time of the start of the frame in the transmitted signal t_t, with which the receiver can synchronize, as illustrated in Figure 1.39, left. As the satellites are located at different distances to the receiver, this time tag will be received at different instants $t_{r,1}$, $t_{r,2}$, etc. However, our receiver measures the pseudoranges at a given instant for *all* satellites, t_r, as shown in the top right part. What is measured is a satellite spreading code that was transmitted by each satellite at a different time, $t_{t,1}$, $t_{t,2}$, etc. The subtlety here is that, for our range estimation, we need to have the satellite positions in ECEF. To do that, we need to convert from an inertial reference frame, not accounting for the Earth rotation, to ECEF, but conversion to ECEF at $t_{t,2}$ is not the same as conversion to ECEF at $t_{t,3}$, as the Earth has rotated between the two timestamps. Therefore, we have to convert to ECEF separately for each satellite. This can be done through the following rotation matrix:

$$\mathbf{x}^{(p)} = \begin{bmatrix} \cos(\dot{\Omega}_e \hat{\tau}^{(p)}) & \sin(\dot{\Omega}_e \hat{\tau}^{(p)}) & 0 \\ -\sin(\dot{\Omega}_e \hat{\tau}^{(p)}) & \cos(\dot{\Omega}_e \hat{\tau}^{(p)}) & 0 \\ 0 & 0 & 1 \end{bmatrix} \mathbf{x}_{nr}^{(p)}, \tag{1.172}$$

where $\mathbf{x}^{(p)}$ is the rotated satellite position vector, $\dot{\Omega}_e$ is the Earth's rotation speed, $\hat{\tau}^{(p)}$ is the expected ToA for satellite (p) ($\hat{\tau}^{(p)} = r^{(p)}/c$) and $\mathbf{x}_{nr}^{(p)}$ is the satellite position vector before rotation. This per-satellite rotation is sometimes called *Sagnac effect*.

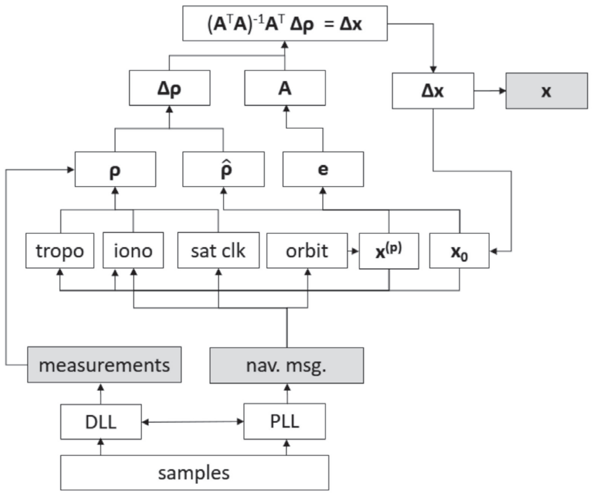


Figure 1.40 Position and time computation process, including inputs (measurements and navigation message), outputs (position and time) and dependencies.

The concept of receiver time reference in an SDR, and more generally, in postprocessing, may not be obvious. Our receivers make an estimation of the reference time based on the GNSS signal. A way to estimate its own time is to assume that the signal of the closest GPS satellite takes, for example, 67 ms, which corresponds to the approximate distance of a GPS satellite at the zenith and use this as an initial time reference that will be refined as part of the position solution. This is also illustrated in Figure 1.39, right. Also, the receiver can estimate the time just with one satellite, with a certain error, by subtracting a rough arrival time estimation from one satellite.

Data Flows and Dependencies

A data flow diagram with the dependencies of the position computation elements is presented in Figure 1.40. The pseudorange measurements are generated from the DLL, after the addition of the integer number of codes to the code delay, and the data bits from the PLL are decoded to form the navigation message. From the navigation message, we obtain the orbits, clocks and ionospheric information (note that signal synchronization is assumed to be achieved and removed from the figure for simplicity). The satellite positions $\mathbf{x}^{(\mathbf{p})}$ at the expected time are computed and used, together with the estimated receiver position in $\mathbf{x_0}$, for the predicted pseudoranges $\hat{\rho}$ and geometry matrix **A**. **A** requires the receiver-to-satellite unitary vector **e**, also using the estimated receiver position in $\mathbf{x_0}$. Our best initial estimation ($\mathbf{x_0}$ in Figure 1.40) can be the Earth center or the center between sub-satellite points. The measured pseudoranges (ρ_m) are corrected with the iono and tropo models and clock corrections to generate the corrected pseudoranges ρ. From these, we subtract the predicted pseudoranges $\hat{\rho}$ and plug the result $\Delta\rho$ into the navigation equation. The equation is iterated until the final solution, **x**, is obtained.

References

[1] J. W. Betz, “Binary Offset Carrier Modulations for Radio Navigation,” *Navigation: Journal of the Institute of Navigation*, vol. 48, no. 4, pp. 227–246, 2002.

[2] GPS Navstar Joint Program Office, “Navstar GPS Space Segment/Navigation User Segment Interfaces, IS-GPS-200,” Revision M, May 21, 2021.

[3] L. Lestarquit, G. Artaud, and J.-L. Issler, “AltBOC for Dummies or Everything You Always Wanted to Know about AltBOC,” in *Proceedings of ION GNSS, Savannah, GA*, pp. 961–970, 2008

[4] P. Misra and P. Enge, *Global Positioning System: Signals, Measurements and Performance, Revised Second Edition*. Ganga-Jamuna Press, 2011.

[5] S. M. Kay, *Fundamentals of Statistical Signal Processing, Volume I: Estimation Theory*. Prentice-Hall, 1993.

[6] J. Proakis and M. Salehi, *Communication Systems Engineering*. Prentice-Hall, 2002.

[7] J. W. Betz, “On the Power Spectral Density of GNSS Signals, with Applications,” in *Proceedings of ION International Technical Meeting (ITM), San Diego, CA,* pp. 859–871, January 2010.

[8] A. V. Oppenheim, A. S. Willsky, and S. Hamid, *Signals and Systems*. Prentice-Hall, 1996.

[9] S. Ramakrishnan, T. Reid, and P. Enge, “Leveraging the l1 Composite Signal to Enable Autonomous Navigation at Geo and beyond,” in *Proceedings of the 26th International Technical Meeting of The Satellite Division of the Institute of Navigation (ION GNSS’2013), Nashville, TN, USA*, pp. 16–20, 2013.

[10] E. D. Kaplan and C. J. Hegarty, *Understanding GPS: Principles and Applications*. Artech House, Inc., 2017.

[11] P. Teunissen and O. Montenbruck, *Springer Handbook of Global Navigation Satellite Systems*. Springer, 2017.

[12] F. van Diggelen, *A-GPS, Assisted GPS, GNSS and SBAS*. Artech House, 2009.

[13] J. W. Betz, *Engineering Satellite-Based Navigation and Timing: Global Navigation Satellite Systems, Signals, and Receivers*. Wiley IEEE Press, 2016.

[14] K. Borre, D. Akos, N. Bertelsen, P. Rinder, and S. H. Jensen, *A Software-Defined GPS and Galileo Receiver, Single Frequency Approach*. Birkhäuser, 2007.

[15] K. J. Quirk and M. Srinivasan, “PN Code Tracking Using Noncommensurate Sampling,” *IEEE Transactions on Communications*, vol. 54, no. 10, pp. 1845–1856, October 2006.

[16] V. Tran, N. Shivaramaiah, T. Nguyen, J. Cheong, E. Glennon and A. Dempster, “Generalised Theory on the Effects of Sampling Frequency on GNSS Code Tracking.” *Journal of Navigation*, vol. 71, no. 4, 257–280.

[17] D. M. Akos, M. Stockmaster, J. B. Y. Tsui, and J. Caschera, “Direct Bandpass Sampling of Multiple Distinct RF Signals,” *IEEE Trans on Communcations*, vol. 47, no. 7, pp. 983–988, July 1999.

[18] A. J. Van Dierendonck, “GPS receivers,” in *Part I: GPS Fundamentals* (J. J. Spilker Jr., P. Axelrad, B. W. Parkinson, and P. Enge, eds.), *Global Positioning System: Theory and Applications*, ch. 8, American Institute of Aeronautics and Astronautics, 1996.

[19] A. Lakhzouri, E. S. Lohan, and M. Renfors, “Reduced-Complexity Time-Domain Correlation for Acquisition and Tracking of BOC-Modulated Signal,” in *Proceedings of ESA NAVITEC, Noordwijk, The Netherlands*, December 2004.

[20] H. Sorokin, E. S. Lohan, and J. Takala, “Memory-Efficient Time-Domain Correlation for BOC-Modulated Signals,” in *Proceedings of ESA NAVITEC, Noordwijk, The Netherlands*, December 2006.

[21] T. Pany, B. Riedl, and J. Winkel, "Efficient GNSS Signal Acquisition with Massive Parallel Algorithms Using GPUs," in *Proceedings of ION Intl. Technical Meeting of the Satellite Division, Portland, OR*, pp. 1889–1895, 2010.

[22] M. L. Psiaki, "Block Acquisition of Weak GPS Signals in a Software Receiver," in *Proceedings of ION GPS*, pp. 2838–2850, *Salt Lake City, UT*, September 2001.

[23] B. Chibout, C. Macabiau, A. C. Escher, L. Ries, J. L. Issler, S. Corrazza, and M. Bousquet, "Comparison of Acquisition Techniques for GNSS Signal Processing in Geostationary Orbit," in *Proceedings of ION National Technical Meeting (NTM), San Diego, CA*, pp. 637–649, 2007.

[24] A. Papoulis, *Probability, Random Variables and Stochastic Processes*. McGraw-Hill, 1991.

[25] D. W. Allan, "Time and Frequency (Time-Domain) Characterization, Estimation and Prediction of Precision Clocks and Oscillators," *IEEE Transactions on Ultrasonics, Ferroelectrics and Frequency Control*, vol. 34, no. 6, pp. 647–654, 1987.

[26] E. Domínguez, A. Pousinho, P. Boto, D. Gómez-Casco, S. Locubiche, G. Seco-Granados, J. A. López-Salcedo, H. Fragner, F. Zangerl, O. Peña, and D. Jimenez-Baños, "Performance Evaluation of High Sensitivity GNSS Techniques in Indoor, Urban and Space Environments," in *Proceedings of ION GNSS+, Portland, OR*, pp. 373–393, September 2016.

[27] G. Corazza and R. Pedone, "Generalized and Average Likelihood Ratio Testing for Post Detection Integration," *IEEE Transactions on Communications*, vol. 55, no. 11, pp. 2159–2171, 2007.

[28] M. Kokkonen and S. Pietilla, "A New Bit Synchronization Method for a GPS Receiver," in *Proceedings of the IEEE Position Location and Navigation Symposium, Palm Springs, CA*, pp. 85–90, 2002.

[29] T. Ren and M. Petovello, "An Analysis of Maximum Likelihood Estimation Method for Bit Synchronization and Decoding of GPS L1 C/A Signals," *Journal on Advances in Signal Processing*, vol. 3, pp. 1–12, 2014.

[30] G. López-Risueño and G. Seco-Granados, "Measurement and Processing of Indoor GPS Signals Using One-Shot Software Receiver," in *Proceedings of ESA NAVITEC, Noordwijk, The Netherlands*, pp. 1–9, 2004.

[31] D. Gómez-Casco, J. A. López-Salcedo, and G. Seco-Granados, "Optimal Fractional Non-coherent Detector for High-Sensitivity GNSS Receivers Robust against Residual Frequency Offset and Unknown Bits," in *Proceedings of IEEE Workshop on Positioning, Navigation and Communications (WPNC), Bremen, Germany*, pp. 1–5, October 2017.

[32] D. Borio and L. L. Presti, "Data and Pilot Combining for Composite GNSS Signal Acquisition," *International Journal of Navigation and Observation*, pp. 1–12, 2008.

[33] B. A. Siddiqui, J. Zhang, M. Z. H. Bhuiyan, and E. S. Lohan, "Joint Data-Pilot Acquisition and Tracking of Galileo E1 Open Service Signal," in *Proceedings of Ubiquitous Positioning Indoor Navigation and Location Based Service, Kirkkonummi, Finland,* pp. 1–7, October 2010.

[34] J. Zhou and C. Liu, "Joint Data-Pilot Acquisition of GPS L1 Civil Signal," in *Proceedings of International Conference on Signal Processing (ICSP), Hangzhou, China*, pp. 1628–1631, October 2014.

[35] S. M. Kay, *Fundamentals of Statistical Signal Processing, Vol 2.: Detection Theory*. Prentice Hall, 1993.

[36] M. K. Simon and M. S. Alouini, *Digital Communication over Fading Channels*. John Wiley & Sons, 2000.

[37] G. J. Povey, “Spread Spectrum PN Code Acquisition Using Hybrid Correlator Architectures,” *Wireless Personal Communications*, vol. 8, pp. 151–164, 1998.

[38] B. J. Kang and I. K. Lee, “A Performance Comparison of Code Acquisition Techniques in DS-CDMA System,” *Wireless Personal Communications*, vol. 25, pp. 163–176, 2003.

[39] E. S. Lohan, A. Lakhzouri, and M. Renfors, “Selection of the Multiple-Dwell Hybrid-Search Strategy for the Acquisition of Galileo Signals in Fading Channels,” in *Proceedings of IEEE Personal and Indoor Mobile Radio Communications (PIMRC)*, vol. 4, *Barcelona, Spain*, pp. 2352–2356, September 2004.

[40] E. S. Lohan, A. Burian, and M. Renfors, “Acquisition of GALILEO Signals in Hybrid Double-Dwell Approaches,” in *Proceedings of ESA NAVITEC, Noordwijk, The Netherlands*, December 2004.

[41] W.-H. Sheen and S. Chiou, “Performance of Multiple-Dwell Pseudo-Noise Code Acquisition with IQ Detector on Frequency-Nonselective Multipath Fading Channels,” *Wireless Networks*, vol. 5, no. 1, pp. 11–21, 1999.

[42] L. E. Aguado, G. J. Brodin, and J. A. Cooper, “Combined GPS/Galileo Highly-Configurable High-Accuracy Receiver,” in *Proceedings of ION GNSS, Long Beach, CA*, September 2004.

[43] A. Konovaltsev, A. Hornbostel, H. Denks, and B. Bandemer, “Acquisition Tradeoffs for Galileo SW Receiver,” in *Proceedings of European Navigation Conference, Manchester, UK*, 2006.

[44] A. Polydoros and C. L. Weber, “A Unified Approach to Serial Search Spread Spectrum Code Acquisition—Part II: A Matched-Filter Receiver,” *IEEE Transactions on Communications*, vol. 32, pp. 550–560, 1984.

[45] N. M. O’Mahony, *Variable Dwell Time Verification Strategies for CDMA Acquisition with Application to GPS Signals*. PhD thesis, Department of Electrical and Electronic Engineering, University College Cork, Ireland, 2010.

[46] A. Konovaltsev, H. Denks, A. Hornbostel, M. Soellner, and M. Kaindl, “General Approach to Analysis of GNSS Signal Acquisition Performance with Application to GALILEO Signals,” in *Proceedings of ESA NAVITEC, Noordwijk, The Netherlands*, 2006.

[47] M. Villanti, C. Palestini, R. Pedone, and G. Corazza, “Robust Code Acquisition in the Presence of BOC Modulation for Future GALILEO Receivers,” in *Proceedings of IEE International Conference on Communications (ICC), Glasgow, UK*, pp. 5801–5806, 2007.

[48] W. Suwansantisuk and M. Z. Win, “Multipath Aided Rapid Acquisition: Optimal Search Strategies,” *IEEE Transactions on Information Theory*, vol. 53, pp. 174–193, 2007.

[49] G. López-Risueño and G. Seco-Granados, “Measurement and Processing of Indoor GPS Signals Using a One-Shot Software Receiver,” in *Proceedings of ESA NAVITEC, Noordwijk, The Netherlands* pp. 1–9, 2004.

[50] D. Jiménez-Baños, N. Blanco-Delgado, G. López-Risueño, G. Seco-Granados, and A. Garcia, “Innovative Techniques for GPS Indoor Positioning Using a Snapshot Receiver,” in *Proceedings of ION GNSS, Fort Worth, TX*, pp. 2944–2955, 2006.

[51] A. Polydoros and M. Simon, “Generalized Serial Search Code Acquisition: The Equivalent Circular State Diagram Approach,” *IEEE Transactions on Communications*, vol. 32, pp. 1260–1268, 1984.

[52] V. M. Jovanovic, “Analysis of Strategies for Serial-Search Spread-Spectrum Code Acquisition–Direct Approach,” *IEEE Transactions on Communications*, vol. 36, pp. 1208–1220, 1998.

[53] A. Burian, E. S. Lohan, and M. Renfors, “BPSK-Like Methods for Hybrid-Search Acquisition of GALILEO Signals,” in *Proceedings of ICC 2006, Istanbul, Turkey*, 2006.

[54] E. Pajala, E. S. Lohan, and M. Renfors, "CFAR Detectors for Hybrid-Search Acquisition of GALILEO Signals," in *Proceedings of ENC-GNSS, Munich, Germany*, pp. 1–7, 2005.
[55] W. Zhuang, "Noncoherent Hybrid Parallel PN Code Acquisition for CDMA Mobile Communications," *IEEE Transactions on Vehicle Technology*, vol. 45, pp. 643–656, 1996.
[56] Y. Zheng, "A Software-Based Frequency Domain Parallel Acquisition Algorithm for GPS Signal," in *Proceedings of IEEE International Conference on Anti-Counterfeiting Security and Identification in Communication (ICASID), Chengdu, China*, pp. 298–301, 2010.
[57] C. O'Driscoll, *Performance Analysis of the Parallel Acquisition of Weak GPS Signals*. PhD thesis, University College Cork, 2007.
[58] W. C. Lindsey and C. M. Chie, "A Survey of Digital Phase-Locked Loops," *Proceedings of the IEEE*, vol. 69, no. 4, pp. 410–431, 1981.
[59] G.-C. Hsieh and J. C. Hung, "Phase-Locked Loop Techniques. A Survey," *IEEE Transactions on Industrial Electronics*, vol. 43, no. 6, pp. 609–615, 1996.
[60] F. M. Gardner, *Phaselock Techniques*. John Wiley & Sons, 2005.
[61] U. Mengali and A. D'Andrea, *Synchronization Techniques for Digital Receivers*. Plenum Press, 1997.
[62] B. W. Parkinson and J. J. Spilker, *Global Positioning System: Theory and Applications Volume I*. Applied Mechanics Reviews American Institute of Astronautics, 1996.
[63] Y. J. Morton, van Diggelen, F. S. Jr, P. J. J., S. B. W., Lo, and G. E. Gao, *Position, Navigation, and Timing Technologies in the 21st Century: Integrated Satellite Navigation, Sensor Systems, and Civil Applications*. John Wiley & Sons, 2021.
[64] The US Government, Global Positioning Systems Directorate, *IS-GPS-200*, j ed., 2018.
[65] I. Fernández-Hernández, T. Senni, D. Borio, and G. Vecchione, "High-Parity Vertical Reed-Solomon Codes for Long GNSS High-Accuracy Messages," *Navigation: Journal of the Institute of Navigation*, vol. 67, no. 2, pp. 365–378, 2020.
[66] M. Miya, S. Fujita, Y. Sato, K. Ota, R. Hirokawa, and J. Takiguchi, "Centimeter Level Augmentation Service (clas) in Japanese Quasi-zenith Satellite System, Its User Interface, Detailed Design, and Plan," in *Proceedings of the 29th International Technical Meeting of the Satellite Division of the Institute of Navigation (ION GNSS+ 2016), Portland, OR*, pp. 2864–2869, 2016.
[67] A. Viterbi, "Error Bounds for Convolutional Codes and an Asymptotically Optimum Decoding Algorithm," *IEEE Transactions on Information Theory*, vol. 13, no. 2, pp. 260–269, 1967.
[68] G. D. Forney, "The Viterbi Algorithm," *Proceedings of the IEEE*, vol. 61, no. 3, pp. 268–278, 1973.
[69] A. Goldsmith, *Wireless Communications*. Cambridge University Press, 2005.
[70] O. Montenbruck, E. Gill, and F. Lutze, "Satellite Orbits: Models, Methods, and Applications," *Applied Mechanics Reviews*, vol. 55, no. 2, pp. B27–B28, 2002.
[71] European Union, "European GNSS (Galileo) Open Service, Signal-in-Space Interface Control Document, Issue 2.0," January 2021.
[72] China Satellite Navigation Office, "BeiDou Navigation Satellite System, Signal in Space Interface Control Document, Open Service Signal B1I," February 2019. version 3.0.
[73] Russian Institute of Space Device Engineering, "Global Navigation Satellite System GLONASS, Interface Control Document, Navigational Radiosignal in Bands L1, L2," 2008. Edition 5.1.

[74] G. Tobías, A. García, C. García, M. Laínez, P. Navarro, and I. Rodríguez, "Advanced GNSS Algorithms and Services Based on Highly-Stable On-board Clocks," in *Proceedings of the 28th International Technical Meeting of the Satellite Division of the Institute of Navigation (ION GNSS+ 2015), Tampa, FL*, pp. 2801–2808, 2015.

[75] M. Paonni, M. Anghileri, T. Burger, L. Ries, S. Schlötzer, B. Schotsch, M. Ouedraogo, S. Damy, E. Chatre, M. Jeannot, J. Godet, and D. Hayes, "Improving the Performance of Galileo E1-OS by Optimizing the I/NAV Navigation Message," in *Proceedings of the 32nd International Technical Meeting of the Satellite Division of the Institute of Navigation (ION GNSS+ 2019), Miami, FL, USA*, pp. 1134–1146, 2019.

[76] I. Fernández-Hernández, V. Rijmen, G. Seco-Granados, J. Simon, I. Rodríguez, and J. D. Calle, "A Navigation Message Authentication Proposal for the Galileo Open Service," *Navigation: Journal of the Institute of Navigation*, vol. 63, no. 1, pp. 85–102, 2016.

[77] P. M. J. Kintner, T. Humphreys, and J. Hinks, "GNSS and Ionospheric Scintillation: How to Survive the Next Solar Maximum," *Inside GNSS*, pp. 22–30, July/August 2009.

[78] A. Aragon-Angel, M. Hernandez-Pajares, P. Defraigne, N. Bergeot, and R. Prieto-Cerdeira, "Modelling and Assessing Ionospheric Higher Order Terms for GNSS Signals," in *Proceedings of ION GNSS+, Tampa, FL*, pp. 3511–3524, September 2015.

[79] J. A. Klobuchar, "Ionospheric Time-Delay Algorithm for Single-Frequency GPS Users," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 23, no. 3, pp. 325–331, 1987.

[80] European Commission, "European GNSS (Galileo) Open Service–Ionospheric Correction Algorithm for Galileo Single Frequency Users," Tech. Rep., September 2016.

[81] B. Nava, P. Coïsson, and S. M. Radicella, "A New Version of the NeQuick Ionosphere Electron Density Model," *Journal of Atmospheric and Solar-Terrestrial Physics*, vol. 70, no. 15, pp. 1856–1862, 2008.

[82] A. E. Niell, "Global Mapping Functions for the Atmosphere Delay at Radio Wavelengths," *Journal of Geophysical Research*, vol. B2, no. 101, pp. 3227–3246, 1996.

[83] P. Collins, R. Langley, and J. LaMance, "Limiting Factors in Tropospheric Propagation Delay Error Modelling for GPS Airborne Navigation," in *Proceedings of the 52nd Annual Meeting of The Institute of Navigation*, vol. 3, 1996.

[84] RTCA, "Minimum Operational Performance Standards for Global Positioning System/Wide Area Augmentation System Airborne Equipment," *RTCA/DO-229D*, 2006.

[85] T. Hobiger and N. Jakowski, "Atmospheric Signal Propagation," in *Springer Handbook of Global Navigation Satellite Systems*, Springer, pp. 165–193, 2017.

[86] I. Fernández-Hernández, T. Walter, K. Alexander, B. Clark, E. Châtre, C. Hegarty, M. Appel, and M. Meurer, "Increasing International Civil Aviation Resilience: A Proposal for Nomenclature, Categorization and Treatment of New Interference Threats," in *Proceedings ION International Technical Meeting (ITM), Reston, VA,* pp. 389–407, January 2019.

[87] F. Dovis, *GNSS Interference Threats and Countermeasures*. Artech House, 2015.

[88] K. Borre and G. Strang, *Algorithms for Global Positioning*. Cambridge University Press, 2012.

[89] M. S. Grewal, "Kalman Filtering," *International Encyclopedia of Statistical Science*. Springer, pp. 705–708, 2011.

[90] R. Hatch, "The Synergism of GPS Code and Carrier Measurements," in *Proceedings of International Geodetic Symposium on Satellite Doppler Positioning, Las Cruces, NM*, pp. 1213–1231, 1983.

[91] Defense Mapping Agency, “World Geodetic System 1984: Its Definition and Relationships with Local Geodetic Systems,” Tech. Rep., 2014.

[92] B. Hofmann-Wellenhof, H. Lichtenegger, and E. Wasle, *GNSS–Global Navigation Satellite Systems: GPS, GLONASS, Galileo, and More*. Springer Science & Business Media, 2007.

[93] F. van Diggelen, “Update: GPS accuracy: Lies, Damn Lies, and Statistics,” *GPS WORLD*, p. January, 2007.

[94] GPS Navstar Joint Program Office, “Global Positioning System Standard Positioning Service Performance Standard,” 5th Edition, April 2020.

[95] European Commission, “Galileo Open Service - Service Definition Document”, Issue 1.2, November 2021.

[96] R. G. Brown, “A Baseline GPS RAIM Scheme and a Note on the Equivalence of Three RAIM Methods,” *Navigation*, vol. 39, no. 3, pp. 301–316, 1992.

[97] P. Enge, T. Walter, S. Pullen, C. Kee, Y.-C. Chao, and Y.-J. Tsai, “Wide Area Augmentation of the Global Positioning System,” in *Proceedings of the IEEE*, vol. 84, no. 8, pp. 1063–1088, 1996.

[98] S. Pullen, T. Walter, and P. Enge, “1.5 System Overview, Recent Developments, and Future Outlook for WAAS and LAAS,” 2002.

[99] J. Blanch, T. Walter, P. Enge, S. Wallner, F. Amarillo Fernandez, R. Dellago, R. Ioannides, I. Fernandez Hernandez, B. Belabbas, A. Spletter, *et al.*, “Critical Elements for a Multi-constellation Advanced Raim,” *Navigation: Journal of the Institute of Navigation*, vol. 60, no. 1, pp. 53–69, 2013.