POLITECNICO
DI TORINO

Department of Electronics and Telecommunications
Master's Degree in Communications and Computer Networks Engineering
Master Thesis

Code Design for IOT Satellite Applications

Supervisor
Prof. Roberto Garello

Candidate
Nima Golamooee
Matricola:236438

October 2020

Acknowledgments

I want to express my sincere appreciation to my thesis advisor, Professor Roberto Garello, for his patience, continuous support, guidance, and useful evaluation of this master thesis. I am very grateful for all of the things that you have done for me, and I could not have imagined a better advisor and mentor.

Countless gratitude to Politecnico di Torino for giving me this opportunity; and for providing me the chance to accomplish this master's degree.

Heartfelt appreciations to my beloved fiancé, Gretchen Margaret, for all your love, support, and encouragement. I am indebted towards you for everything that you have done for me, and for always believing in me. I wholeheartedly dedicate this work to you.

Last but not least, I must express my profound gratitude to my family for supporting me in all the steps of my life with their continuous encouragement.

I could not have achieved this without all of you, so thank you to all from the bottom of my heart.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

AC	**A**uto-**C**orrelation
ACF	**A**uto-**C**orrelation **F**unction
ACF^e	**E**ven **A**uto-**C**orrelation **F**unction
ACF^o	**O**dd **A**uto-**C**orrelation **F**unction
AELW	**A**verage **E**xcess **L**ine **W**eight
AMEWSD	**A**verage **M**ean **E**xcess **W**elch **S**quare **D**istance
AMF	**A**verage **M**erit **F**actor
C/A	**C**oarse /**A**cquisition
CC	**C**ross -**C**orrelation
CCF	**C**ross -**C**orrelation **F**unction
CC^e	**E**ven **C**ross -**C**orrelation
CC^o	**O**dd **C**ross -**C**orrelation
CDMA	**C**ode **D**ivision **M**ultiple **A**ccess
DFT	**D**iscrete **F**ourier **T**ransform
ELW	**E**xcess **L**ine **W**eight
EWSD	**E**xcess **W**elch **S**quare **D**istance
GNSS	**G**lobal **N**avigation **S**atellite **S**ystem
GPS	**G**lobal **P**ositioning **S**ystem
LFSR	**L**inear **F**eedback **S**hift **R**egister
MEWSD	**M**ean **E**xcess **W**elch **S**quare **D**istance
MF	**M**erit **F**actor
MLS	**M**aximum **L**ength **S**equence
MPSL	**M**aximum **P**eak **S**ide **L**obe
PRN	**P**seudo **R**andom **N**oise
PSD	**P**ower **S**pectral **D**ensity
SIS	**S**ignal-**I**n-**S**pace
SNR	**S**ignal -to -**N**oise **R**atio
TTFF	**T**ime-**T**o-**F**irst-**F**ix

Abstract

The signal broadcast by the navigation satellites is usually represented as Signal-In-Space (SIS). These signals must: allow the user to estimate the pseudo distance user-satellite, carry some useful data, be robust to the transmission through the atmosphere, and identify uniquely the satellites [3]. GPS and Galileo use CDMA (Code Division Multiple Access) to identify the satellites without ambiguity. Each satellite transmits a different code (ranging codes) using the same carrier frequencies without time division. Codes are mutually orthogonal in order to permit the receiver to separate the signal of the satellite of interest from the others, and the data signals are modulated by the ranging codes. To design a new GNSS signal, there is always a trade-off between improving performance and increasing complexity or even between improving different performance benchmarks such as position accuracy, receiver sensitivity, or the Time-To-First-Fix (TTFF) [1]. To improve the performance of GNSS receivers concerning these benchmarks, adding a new signal component devoted to assist the acquisition process can be helpful. To generate this new component, we have to take into account various features such as the spreading modulation, the data navigation content, the channel coding, or the Pseudo Random Noise (PRN) sequences. Pseudo Random Noise (PRN) sequences are a fundamental element in any CDMA system such as GPS, Galileo, Compass, and widely used for satellite navigation systems. They are similar to noise and lack a definite pattern but are still deterministic and repeat themselves after their period. PRN sequence is a bit stream of ones and zeros occurring almost randomly. In this Thesis project, the goal is to design PRN codes (Random sequences) that can be suitable for this new signal component to optimize performance benchmarks. To carry out the optimization process, a detailed evaluation framework based on the welch bound including three key metrics has been recommended: performance in the acquisition stage, performance in the tracking stage, and the robustness against interfering narrowband signals. Besides, in a final figure or cost function, these three metrics can be combined to compare various candidate sequence sets. The sequence families are optimized only according to sequence properties. This means that in this project the real modulation characteristics of the signal, i.e., its particular spreading waveform, are not considered in the sequence design. The optimization process is mainly focused on analyzing auto-correlation and cross-correlation properties of the full length spreading sequences and their truncated versions to select the best among them. For the CDMA application, the cross-correlation issue is very important. Our primary solution is based on Gold sequences analysis because they have good correlation properties. We truncate the Gold sequences by removing the last few bits. Truncation can be done by removing a few bits from the beginning or at the end of the sequence. Then we compare Gold sequences and their truncated versions. The goal is to design a sequence that minimizes auto-correlation and cross-correlation function side lobe as much as possible to get closer to the ideal spreading sequences which are orthogonal with all of their delayed versions and with all the family codes.

Chapter 1

Introduction

To select the best chip spreading sequence, we have to consider different properties such as the autocorrelation, the cross-correlation, and the power spectral density. Theoretically, the ideal spreading sequence is orthogonal with all of its delayed versions and with all the family codes. This means that for any relative non-zero chip delay the autocorrelation function is zero and for any relative delay value the cross-correlation is zero. Practically ideal spreading codes are not easy to achieve. Imperfect spreading codes which are currently used, clearly underperform concerning a hypothetical ideal code, and this affects the final performance of the receiver [1]. One of the most noticeable limitations in developing a new spreading sequence family is the length of the code that must be as short as possible since the new component shall allow a faster acquisition process. The smallest length which is an entire divisor of the chip frequency 1.023 MHz is a 1023 chips code length and can be generated by a mathematical method [1]. There are several mathematical spreading sequences of families with a 1023 chips length. Some of them, like the Gold codes family, have already been used in the design of GNSS signals. Other families of sequences such as large Kasami can potentially be suitable spreading sequences family candidates. Other techniques generate efficient memory sequences; these techniques apply a cost function with limitations in order to optimize some properties.

1.1 Auto-Correlation and Cross-Correlation:

Let $C_{[n]}$ be an infinite sequence (or code) of complex numbers with period N. We suppose it has unitary normalized energy.

$$E_c = \frac{1}{N}\sum_{n=0}^{N-1} |c[n]|^2 = 1 \qquad (1)$$

Auto-Correlation and Cross-Correlation: Correlation is a measure of similarity between two sequences. Cross-correlation is when we compare two different sequences and auto-correlation is when we compare a sequence with itself. We define the (periodic,non normalized) auto-correlation funcion of the sequence C as :

$$r_c(\tau) = \sum_{n=0}^{N-1} c[n]\, c^*[n-\tau] \qquad 0 \le \tau \le N-1 \qquad (2)$$

We define the (periodic,non normalized) cross-correlation function between two sequences C_i and C_j as :

$$r_{c_i c_j}(\tau) = \sum_{n=0}^{N-1} c_i[n]\, c_j^*[n-\tau] \qquad 0 \le \tau \le N-1$$

(3)

The ideal auto-correlation behavior would be :

$$\begin{cases} r_c(\tau) = N & \tau = 0 \\ r_c(\tau) = 0 & \tau \neq 0 \end{cases}$$

(4)

The ideal cross-correlation behavior would be:

$$\left\{ r_{c_i c_j}(\tau) = 0 \qquad \forall \tau \right.$$

(5)

It is easy to show that it is impossible to satisfy these two conditions at the same time.

1.2 Bounds:

1.2.1 Correlation Maximal Magnitude :

Given a set of K sequences (C_1 , C_2 ,…., C_k) of length N, we define the maximal out-of-phase auto-correlation magnitude, the maximal cross-correlation magnitude, and maximal correlation magnitude respectively as :

$$r_A = \max\{|r_{c_i}(\tau)| \quad 1 \le \tau \le N-1 \quad 1 \le i \le K \quad \}$$

(6)

$$r_C = \max\{|r_{c_i c_j}(\tau)| \quad 0 \le \tau \le N-1 \quad 1 \le i,j \le K \quad i \neq j\}$$

(7)

$$r_M = \max\{r_C, r_A\}$$

(8)

1.2.2 Welch bound:

for a set of K complex sequences, welch bound is considered as a lower bound on the sum of the squares of the magnitudes of the inner products between all pairs of these sequences.it holds for complex sequences:

$$r_M^{2s} \geq \frac{1}{KN-1}\left[\frac{KN}{\binom{N+s-1}{s}} - 1\right] \qquad \forall s \in \mathbb{Z}^+ \tag{9}$$

1.2.3 Sidelnikov bound:

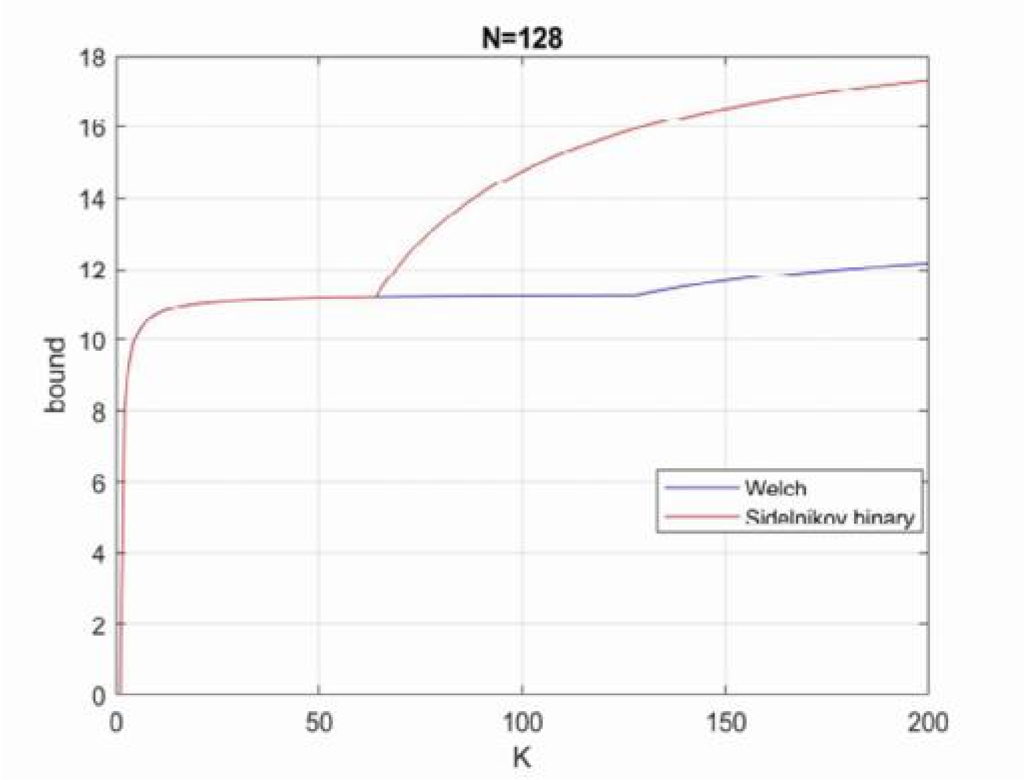
It holds for binary sequences.

$$r_M^2 > \frac{1}{N^2}\left[(2s+1)(N-s) + \frac{s(s+1)}{2} - \frac{2^s N^{2s+1}}{K(2s)!\binom{N}{s}}\right] \tag{10}$$

$$\forall s \in \mathbb{N} \qquad 0 \leq s < \frac{2N}{5}$$

Since the Sidelnikov bound applies to binary sequences, it is tighter than the welch bound.

Figure 1.1



Chapter 2

Overview and Construction of family of Spreading Codes

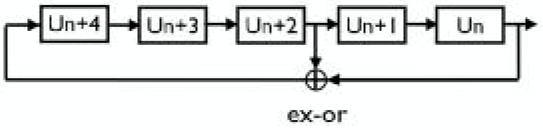
2.1 m-sequences:

m-sequences (maximum length sequences) are sort of pseudo-random binary sequences. They are sequences of bits (zeros and ones). The process of their generation is through maximal linear feedback shift registers (LFSRs). The philosophy behind their name (maximum length sequences) backs to their periodic behavior and ability to regenerate every binary sequence (except for the all zero vector) that can be characterized by the shift registers. They generate a sequence of length $2^n - 1$ where n is the number of registers. Maximum length sequence generator with a shift register of length 5 is shown in figure 2.1.

Figure 2.1 [3]

$$u_n^{(k)} \in \{0,1\} \leftrightarrow b_n^{(k)} \in \{1,-1\}$$

$$\oplus \leftrightarrow \cdot$$



In figure 2.1, n represents the time index and + is modulo-2 addition which corresponds to the ex-or when bit values are 0 or 1. Initial state of the registers can be any state except for the all zero state. The reason is that m-sequences are periodic and shift registers are able to cycle through every possible binary value (except for the all zero vector). It is common to describe the design of linear sequence generators by means of polynomials of the form: [3]

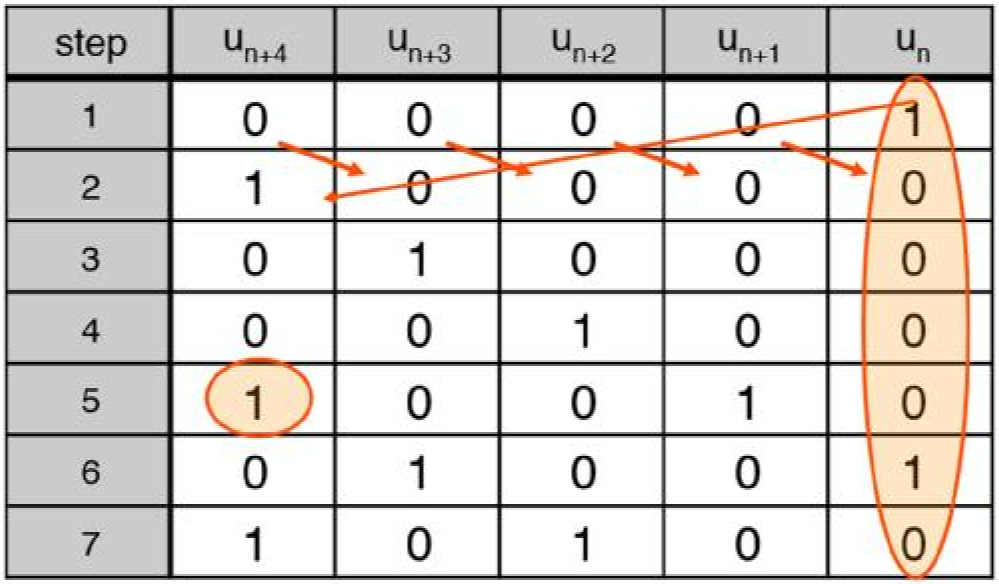
$$1 + \sum_i x^i \tag{11}$$

Where x^i means that the output of the i-th cell of the shift register is used at the input of the modulo-2 adder (ex-or). Only polynomials with specific properties generate m-sequences. The polynomial degree is actually equal to the length of the shift register. The polynomial factors are either zero or one and they correspond to the taps of the register that feed the ex-or gate. For instance, the polynomial related to Figure 2.1 is 1+x^3+x^5.

To generate a maximal length sequence by a linear feedback shift register, the related polynomial should be primitive. The shift register in the example can assume $2^n - 1$ states, where n=5 (The all zero state has to be avoided). If the register assumes all the possible states a maximum length sequence is obtained. Different positions of the ex-or give (or not) maximum length sequences.

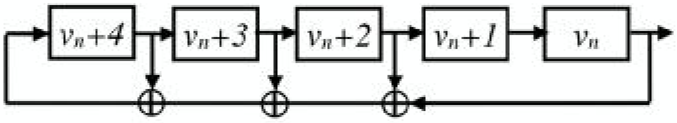
The process of code generation with initial state [0 0 0 0 1] is shown in figure 2.2.

Figure 2.2 [3]



step	u_{n+4}	u_{n+3}	u_{n+2}	u_{n+1}	u_n
1	0	0	0	0	1
2	1	0	0	0	0
3	0	1	0	0	0
4	0	0	1	0	0
5	1	0	0	1	0
6	0	1	0	0	1
7	1	0	1	0	0

Example of shift register which generates a m-sequence with corresponding polynomial :



$$1 + x + x^2 + x^3 + x^5$$

[3](12)

m-sequences properties: Cyclically shifted versions of the sequence are orthogonal.

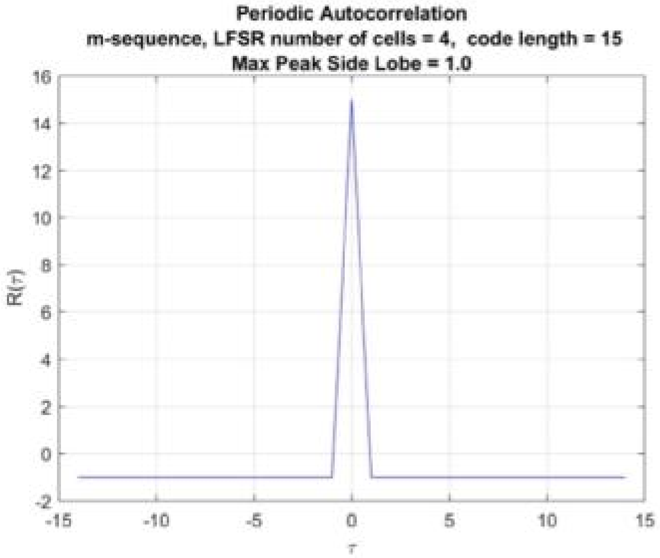
$$\sum_{n-1}^{p} b_n b_{(n+m)} \mod p \simeq 0$$

[3] (13)

Each sequence has length P=$2^n - 1$ bits. Each sequence contains 2^{n-1}ones and $2^{n-1} - 1$ zeros. The ex-or of a sequence and of a cyclically shifted version is another shifted version of the same sequence.

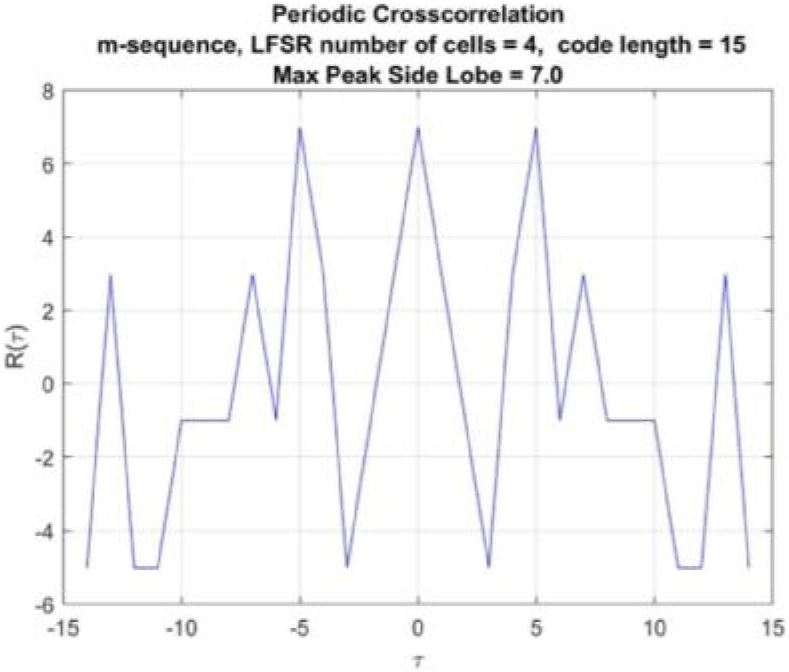
Figure 2.3 shows the periodic auto-correlation of the m-sequence with LFSR number of cells equal to 4 and code length equal to 15.

Figure 2.3



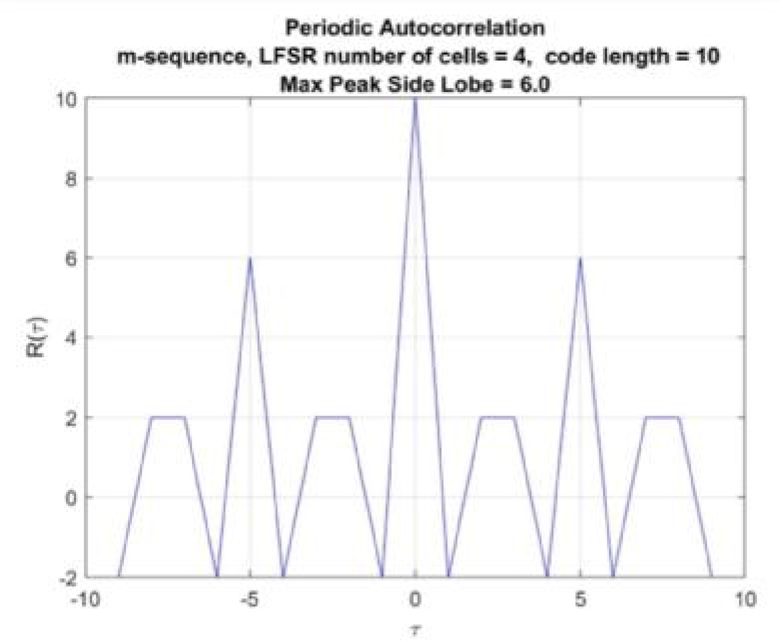
In Figure 2.4, we can observe the periodic cross-correlation of the same m-sequence which is not so good.

Figure 2.4



In Figure 2.5, we can observe the behavior of period auto-correlation of the **Truncated m-sequence** which is not more ideal (the sequence length is reduced to L which is smaller than N=$2^n - 1$).

Figure 2.5



2.2 Gold Codes:

A family of codes with good correlation properties can be generated using two different sequences of the same length.

Gold codes are the main class of periodic sequences, providing relatively broad sets of codes with strong periodic cross-correlation and autocorrelation properties.

For a length N, a family of N+2 Gold sequences can be generated. Gold codes have a code period of N = $2^n - 1$ chips and have $2^n + 1$ codes in the set. These codes are made of selected m-sequences, and in particular of preferred pairs of m-sequences of length, N. Preferred pairs of m-sequences a and b of period N = $2^n - 1$ generated by primitive binary polynomials with no common factor and where ≠0 mod 4 is determined [1].

The set of sequences defined by $G\ (a, b)$ is called Gold codes.

$$G\ (a, b) = \{a, b, a + b, a + Tb, \ldots, T^{N-1}b\} \qquad [1]\ (14)$$

$T^i a$ signifies the operator generating the sequence in which the k^{th} element is a_{k+i}. Linear Feedback Shift Registers (LFSRs) generate gold sequences as in their construction two binary polynomials are involved.

Cross-Correlation of Gold sequences has three possible values: $-t, -1$, $t-2$.

$t = 2^{(n+1)/2} + 1$ if n is an odd number (15)

$t = 2^{(n+2)/2} + 1$ if n is an even number (16)

A complete set of Gold sequences has length N = $2^n - 1$, Cardinality K= $2^n + 1$ and maximal cross-correlation magnitude $r_c = t$.

The comparison against Sidelnikov Bound is shown in Table 2.1:

Table 2.1

n	N	K	r_c	SB
3	7	9	5	4
4	15	17	9	7
5	31	33	9	8
6	63	65	17	12
7	127	129	17	16
8	255	257	47	23

2.3 Gold codes in GPS:

In GPS C/A code N=1023 (n=10); sequences for the satellites are chosen among the 1025 Gold sequences. A Gold code, also known as Gold sequence, is a type of binary sequence, used in satellite navigation systems. PRN ("Pseudo-Random Noise") sequences are used as part of the CDMA scheme in GPS. In fact, it's not random, it's a kind of pseudorandom number. One of the reasons we want a spreading sequence that looks random is to ensure we have good correlation properties.

We also want the different orthogonal codes for each satellite, so the auto-correlation of one code with the signal coming from the same satellite, will be very high even when many satellite codes are mixed in and cross-correlation of PRN codes with the signal not containing codes from corresponding satellite, would be nearly zero. This is the scheme used to create pseudo-random sequences with all these properties for GPS. A set of Gold codes consists of sequences in which, each one has a length equal to $2^n - 1$. The Gold Codes are generated using a pair of shift registers with the feedback out of two pre-defined maximum length sequences (MLS) and their shifted versions. The ex-or of two different Gold codes from the same set generates another Gold code in the same phase.

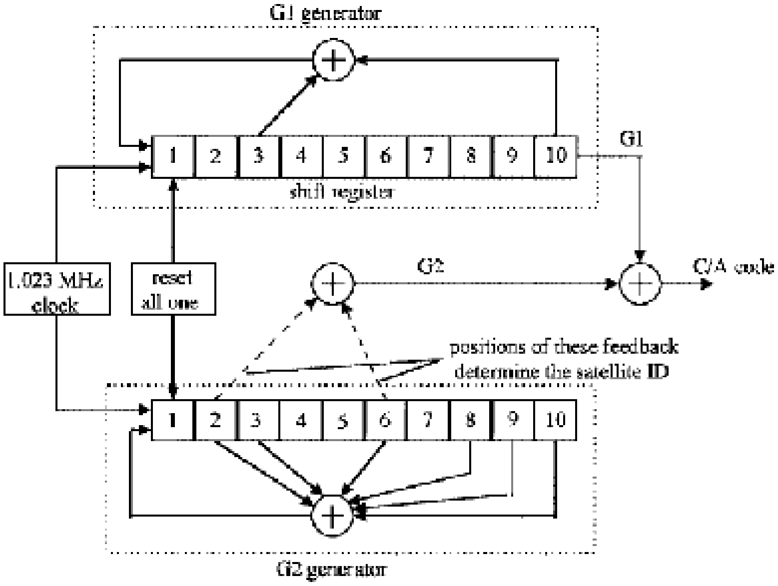
In GPS Two sequences generate all the C/A codes for the satellites:

G1: $1+x^3+x^{10}$

G2: $1+x^2+x^3+x^6+x^8+x^9+x^{10}$

Each satellite is characterized by a different shifted version of G2. The delay effect of the G2 output is obtained by ex-or of selected positions of two taps. This is possible thanks to the properties of the m-sequences that the sum of a sequence and a shifted version of it, is another shifted version of the same sequence.

Figure 2.6



We pick two maximum length sequences of the same length $2^n - 1$, where n is the size of the LFSR used to generate the maximum length sequence. It can be written also as two polynomials, where the powers correspond to the index of registers, having output to ex-or function. 32 different permutations of this indexes are taken to generate 32 unique PRN codes for each of 32 satellites that transmit data in GPS.

C/A codes are almost uncorrelated with each other (cross-correlation is almost zero) and with themselves (auto-correlation is nearly zero, except for zero latency). This property makes it easy whether two identical codes are matched perfectly.

2.4 Balanced Gold Codes:

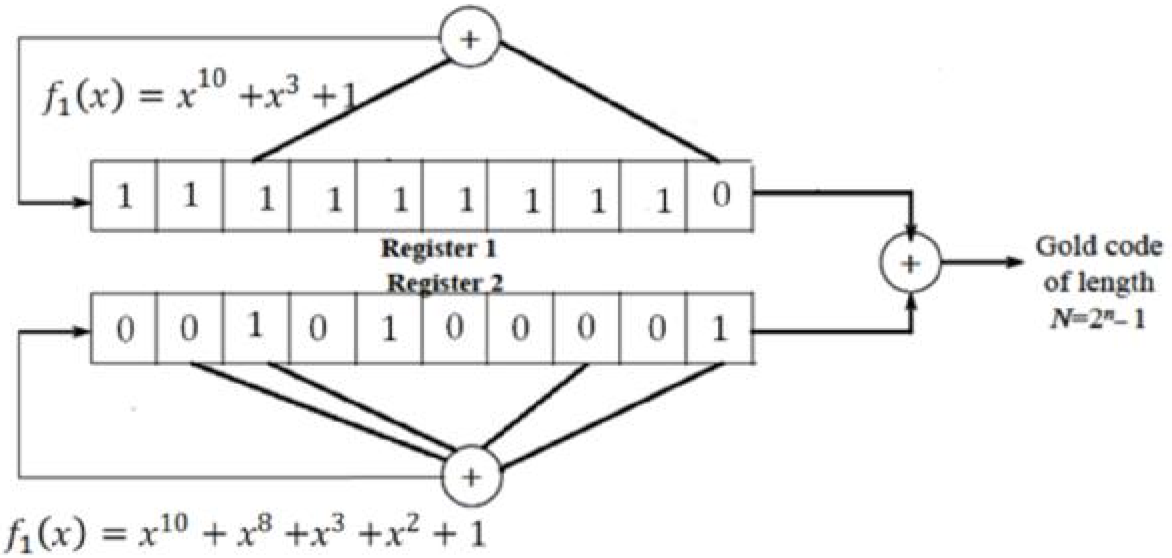
In balanced sequence the number of "ones" in the sequence is equal to number of "zeros" of the corresponding sequence plus one. Such sequences have favorable spectral characteristics, but not all Gold sequences are balanced sequences. To construct the family of balanced gold codes, we need to consider following steps:

First of all, we have to select a preferred pair of m-sequences a and b of length N=$2^n - 1$. Then for shift register 2, initial conditions are obtained by long division of the ratio $g(x)/f(x)$, where $f(x)$ is the characteristic polynomial of sequence b and $g(x)$ is defined as: $g(x) = f(x) + d[xf(x)]$ dx and for shift register 1, initial conditions correspond to all ones and finally the set of Gold codes is formed by ex-or of the two registers 1 and 2. [1]

First polynomial is 1+x^3+$x^{10}$and second one is 1+x^2+x^3+x^8+x^{10}.

In Figure 2.7, we can observe the scheme of balanced gold codes: [1]

Figure 2.7



So from the sequences a and b, we can obtain the family as:

$$G\,(a, b) = \{a, b, a + b, a + Tb, \ldots, T^{N-1}b\} \tag{17}$$

As we can observe in Figure 7, our initial condition for sequence a is 1111111110 and for sequence b is 0010100001.

2.5 Kasami codes:

Kasami codes are binary codes. Their length is N=$2^n - 1$, where N is an even integer number. They have good cross-correlation values that approach the Welch's lower bound.

We classify kasami codes as follows: Small kasami codes and Large kasami codes.

2.6 Large Kasami codes:

Like Gold codes, Large kasami codes are a collection of periodic sequences with strong correlation properties. Under the condition of $d(n,\ 4)$ = 2, they have a code period of N=$2^n - 1$. Their maximum cross-correlation is $2^{(n+2)/2}$. For instance, for $n = 10$ the maximum cross-correlation value will be 64. They contain both Gold codes and Small kasami codes as subset.

In addition, the size of this family of codes is equal to $(N + 2).\ (N + 1)^{1/2}$. [1] (18)

For 1023 chips code length, the family size is equal to 32800 codes and we select the most remarkable ones in terms of correlation properties.

Small set of kasami codes are needed in order to create large kasami codes.

In the case of code length equal to 1023 chips, Small kasami sequences have the most remarkable correlation properties. Small kasami sequences are optimal because they match Welch's lower bound for correlation functions. Nevertheless, the family size is only $(N+1)^{1/2}$=32 codes, which is not large enough to support all of the satellites of one GNSS constellation.

According to the theory and for sequence length equal to 1023 chips, we define a preferred pair of m-sequences a and c. we have also b sequence, the decimation of a sequence and a periodic m-sequence with period equal to $2^{(\frac{n}{2})}-1$ which is produced by characteristic polynomial of degree $n/2$.

$$b = a\ [2^{(\frac{n}{2})}+1] = a[33]$$ [1] (19)

For the set of sequences defined by $KL(a)$, the Large Kasami codes are: [1]

$$K_L(a) = G(a,c) \cup [\bigcup_{i=0}^{2^{(\frac{n}{2})}-1} \{T^i b + G(a,c)\}]$$ (20)

First polynomial is 1+x^3+x^{10}, second one is 1+x^2+x^3+x^8+x^{10} and third one is 1+x^2+x^3+x^4+x^5.

2.7 Weil codes:

Weil codes have strong pseudo-random and auto-correlation properties. Their linear complexity is almost high. They are based on Legendre sequences, which are generated from quadratic residues. The quadratic residue is defined as follows:

If the greatest common divisor of α and β is 1,which in mathematical form will be (α, β) =1; if β divided by $(x^2-\alpha)$ has a solution, α is called the quadratic residue of module β. They are called Weil codes because their correlation bound depends on Weil's classical bound of the magnitude of sums of Legendre symbols [2]. The Legendre symbol (x/N) is defined by:

$$\frac{x}{N} = \begin{cases} 1 \text{ if } x \text{ is the quadratic residue of } N \\ 0 \text{ if } x \text{ is not the quadratic residue of } N \end{cases}$$ [2] (21)

Where N is the length of Legendre sequence.

Weil sequence can be generated by ex-or of Legendre sequence and its cyclic shift. t in (23) is the shift value of Legendre sequence.

$$Leg_N(i) = \left(\frac{i}{N}\right), \quad 1 \le i \le N-1$$

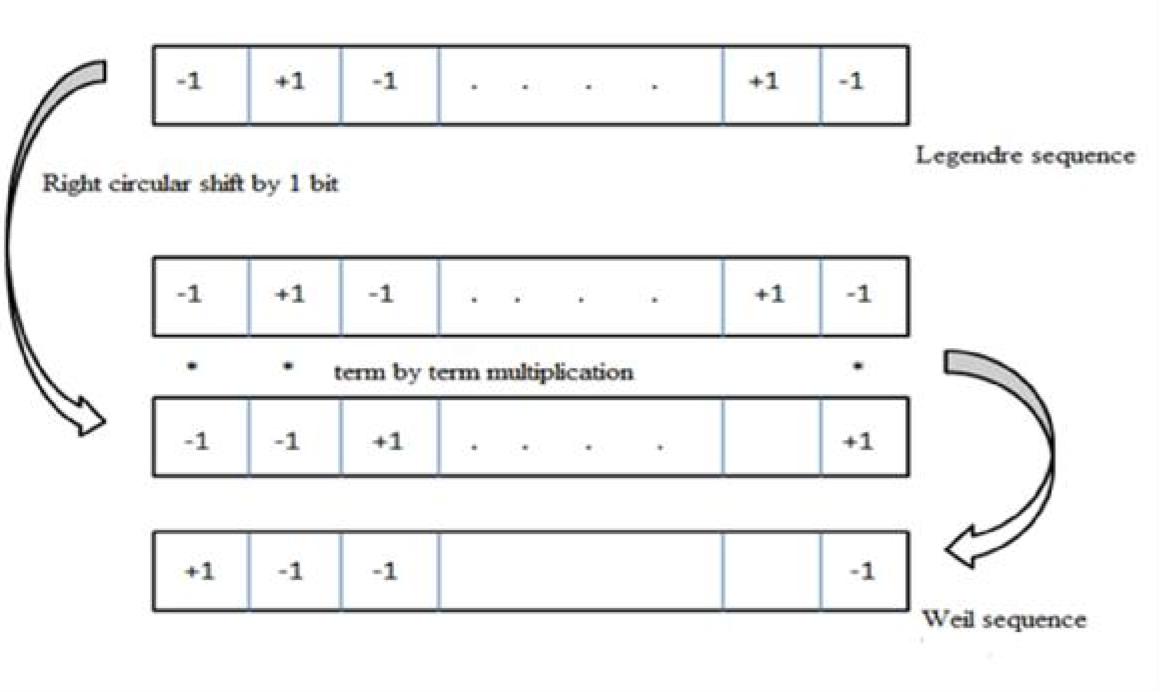
[2] (22)

$$Weil_N^t(i) = Leg_N(i) \oplus Leg_N(i+t), \quad 1 \le i \le (N-1)/2$$

[2] (23)

Figure 2.8 shows the generation process of Weil sequence [2].

Figure 2.8



2.8 Random Codes:

A procedure for generating a collection of spreading sequences with strong correlation properties includes: creating a primary collection of random bits' patterns, in which each of them can be our possible spreading sequence and gives improved performances against the primary collection of bits, with the ultimate aim of choosing an optimized final collection of spreading sequences.

To see if the current iteration offers a better solution than the previous one, we need to define the cost function. This function evaluates undesirable correlation peaks (those which enhance the error probability of the acquisition stage). Consequently, any correlation value surpassing the welch bound degrades our system performance.

The following equation is the cost function: [1]

$$F_i = \sum_{\substack{l=1 \\ ACF^e(l)>\Phi_{min}}}^{N-1} (ACF^e(l) - \Phi_{min})^2 + \sum_{j\neq i} \sum_{\substack{l=1 \\ CC_j{}^e(l)>\Phi_{min}}}^{N-1} (CC_j{}^e(l) - \Phi_{min})^2 \tag{24}$$

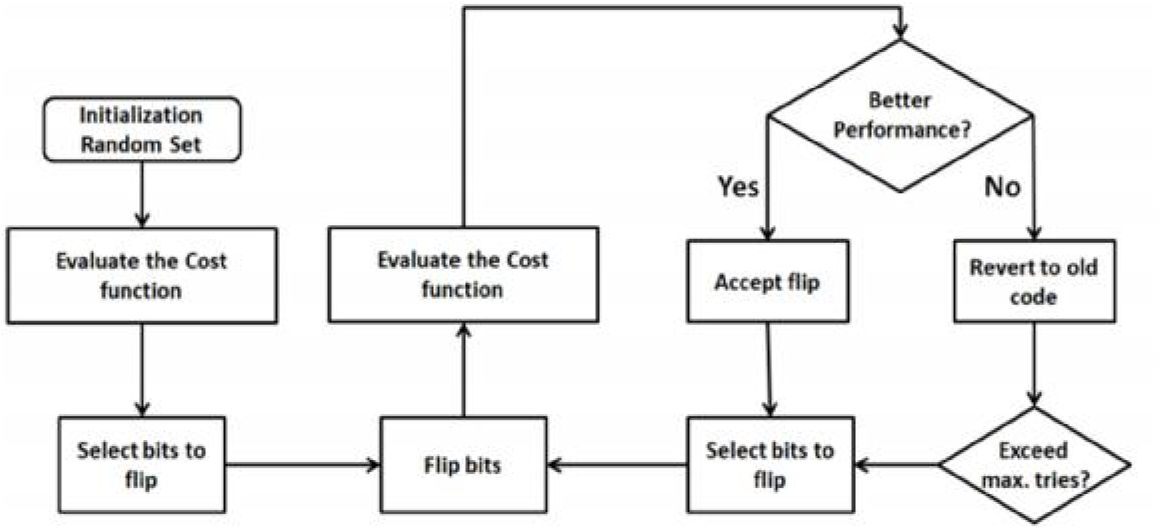
Where ACF^e is even auto-correlation function, CCj^e is even cross-correlation function and φ min is welch bound.

Every iteration of the algorithm consists of two steps: a chip flip within the spreading sequence and cost function assessment.

The chip flip is accepted if minimizes the cost function regarding the preceding iteration otherwise will be rejected.

The algorithm's flow diagram for optimization is shown in figure 2.9: [1]

Figure 2.9



We want both balanced and minimum auto-correlation side-lobe properties for our spreading sequences. We set these properties as initial necessities for the initial set of sequences but after some of the bits have been flipped, those characteristics could not be kept.

To guarantee them, we impose two conditions on the flipping bit step.

The first condition guarantees that the sequence remains balanced and in order to satisfy it, we always flip the bits in pairs, i.e., one bit with zero value and one bit with value 1 are flipped.

The second condition is to minimize auto-correlation function side-lobe and through the following equation, we satisfy it:

$$a_{k-1} + a_{k+1} = a_{j-1} + a_{j+1} \qquad [1]\ (25)$$

Where a_k and a_j are the bits that are flipped.

Chapter 3

Performance Evaluation of Spreading codes based on the Welch bound

In this chapter, we briefly overview the acquisition and tracking stages in GNSS receivers. The reason is that first of all both of these stages rely on the estimation of correlation functions (auto-correlation and cross-correlation functions) and secondly in order to select the best spreading sequence, we evaluate a weighted cost function according to a set of benchmarks based on autocorrelation, cross-correlation, and power spectral density. These benchmarks show the effect of the non-ideal spreading sequences over the acquisition and tracking stages as well as over the robustness against narrow-band interferences. Then we will introduce the weighted performance evaluation method for spreading sequences based on the welch bound.

3.1 Acquisition and Tracking stages:

The navigation receiver should be capable to perfectly align spreading sequences used by transmitter and receiver. It is necessary to align the receiver local sequence recovering the delay between a locally generated and the received code. It is also necessary to recover the frequency and phase of the locally generated carrier. Small misalignments of the sequence can lead to large pseudo-range errors [3].

The synchronization procedure of the sequences consists of two distinct steps:

3.1.1 Acquisition:

When a receiver is turned on, a set of operations must be done before the navigation position can be evaluated. All the operations from the receiver start-up to the detection and signal confirmation are usually called the acquisition stage. To track the signal, it is necessary to first find the signal and its parameters through the acquisition stage. The acquisition is based on the correlation of the received signal with a local matching replica [3].

The acquisition stage consists of two parts: detection and estimation. First of all, we detect the n-th satellite (if present) and then we estimate the parameters of the detected satellites. The initial rough estimate of the delay and Doppler shift between the incoming sequence and the local replica will be done at this stage [3].

3.1.2 Tracking:

In this stage we keep the sequences synchronized so we can dynamically recover the delay between them (fine alignment) [3].

As we mentioned before both stages highly rely on the estimation of auto-correlation and cross-correlation functions.

3.2 A Weighted Performance Evaluation Method based on the Welch bound:

As the GNSS signal structure improves continuously, we need to design new spreading sequences for the different signals. To compare various candidates of code families, selection benchmarks have been set up. These benchmarks model the effect of the spreading sequence characteristics (in particular their auto- correlation, and cross-correlation functions) onto the receiver performance. A detailed evaluation framework based on the welch bound, including three key metrics, has been recommended for this reason: performance in the acquisition stage, performance in the tracking stage, and the robustness against interfering narrowband signals. Besides, in a final figure or cost function, these three metrics can be combined to compare various candidate sequence sets. The user groups should evaluate the corresponding weighted coefficients since the sequence effect varies within each type of application. These three performance metrics and cost function will be explained in the following categories:

3.2.1 Acquisition performance metric:

The goal of the acquisition performance metric is to find undesirable correlation peaks during the acquisition stage. These high out of phase peak values for both auto-correlation and cross-correlation functions increase the probability of false acquisition and as a result, decrease the detection performance. Such correlation peaks occur in aggressive environments, where the correlation characteristics of the signal are affected by enormous attenuations or strong multipath. These unwanted correlation peaks degrade the cross-correlation characteristics of the signal in presence of other GNSS signals. Several mathematical models are recommended to quantify this effect. The problem is that since their assumptions are based on the environment (losses) or the architecture of the receiver, it will be hard to distinguish the pure effect of correlation characteristics of a particular set of sequences from the effects of these metrics. So a simple mathematical model based on the welch bound (φ min) has been designed. This simple metric just depends on the sequence correlation characteristics and evaluates the effect of the sequence onto the acquisition performance. Theoretically, the welch bound is the minimum of the maximum value of auto-correlation or cross-correlation functions. It can be obtained according to (26).

$$\phi_{\min} = N\sqrt{\frac{M-1}{MN-1}}$$

[2] (26)

Where M is the number of sequences and N is the sequence length.

We consider correlation values that exceed (φ min) as an acquisition performance degradation risk.

The greater the distance between these correlation values and the welch bound, the greater will be the performance deterioration. So our metric based on the welch bound should consider this distance. This metric is called Excess Welch Square Distances (EWSD) and has two sub-metrics:

$EWSD^{MP}$: this metric is used to assess the effects of multipath on the desired signal and is mainly focused on analyzing the auto-correlation function (ACF). To justify the introduction of this metric, we can consider a significant attenuation of a direct path signal while other reflection components arrive without degradation at the receiver. To avoid the receiver lock onto one of the side lobes of the auto-correlation function during the acquisition stage, the Mean Excess Welch Squared Distance $MEWSD^{MP}$ has been introduced as follows [2] :

$$\text{MEWSD}^{\text{MP}} = \text{mean}\left(\sum_{nf_{offs}} \frac{1}{2}\left(\underbrace{\sum_{l=1}^{N-1}\left(AC^{e}\left(l,f_{offs}\right)-\phi_{\min}\right)^{2}+}_{AC^{e}(l,f_{offs})>\phi_{\min}} \underbrace{\sum_{l=1}^{N-1}\left(AC^{o}\left(l,f_{offs}\right)-\phi_{\min}\right)^{2}}_{AC^{o}(l,f_{offs})>\phi_{\min}}\right)\right) \quad (27)$$

$AC^{e}(l,\ f_{offs}\ )$ and $AC^{0}(l,\ f_{offs}\ )$ are the even and odd auto-correlation functions calculated for a relative delay lag -times Tc and for a doppler frequency offset f_{offs}, and the sample number of doppler frequency offset is nf_{offs} (n is the number of frequency points). The $MEWSD^{MP}$ shows the average value (over all possible delays and doppler frequency offsets) of the squared distances between the welch bound and all even or odd auto-correlation function elements that exceed this bound.

Equation (27) is for one sequence. But we have M sequences in our set so, to assess the impact over all sequence family, we need to take an average among them. The formula (28) shows the Average Mean Excess Welch Square Distance $AMEWSD^{MP}$:

$$\text{AMEWSD}^{\text{MP}} = \frac{1}{M}\sum_{i=1}^{M}\text{MEWSD}_{i}^{\text{MP}} \quad [2]\ (28)$$

Another sub-metric of the general EWSD benchmark is based on cross-correlation function.

In other words, again our direct path suffers from significant attenuation but we assume that signals with the sequences from other satellites are not attenuated. So we have to consider the cross-correlation peaks between the replica and all of these sequences.

$EWSD^{CT}$: This metric is used to assess the effects of the non-desired signals onto the acquisition performance (effect of non-desired signals from other satellites on the direct path). The expression is similar to (27). We just need to substitute the auto-correlation terms with the cross-correlation ones.

$$\mathrm{MEWSD}_{i,j}^{\mathrm{CT}} = \mathrm{mean}\left(\sum_{nf_{\mathit{offs}}} \frac{1}{2}\left(\underbrace{\sum_{l=0}^{N-1}\left(CC_{i,j}^{e}(l,f_{\mathit{offs}}) - \phi_{\min}\right)^2}_{CC_{i,j}^{e}(l,f_{\mathit{offs}})>\phi_{\min}} + \underbrace{\sum_{l=0}^{N-1}\left(CC_{i,j}^{o}(l,f_{\mathit{offs}}) - \phi_{\min}\right)^2}_{CC_{i,j}^{o}(l,f_{\mathit{offs}})>\phi_{\min}}\right)\right)$$

[2] (29)

$CC_{i,j}e(l,\ f_{offs}$) and $CC_{i,j}o(l,\ f_{offs}$) are the even and odd cross-correlation functions for the sequences i, j calculated for a relative delay lag- times Tc and for a doppler frequency offset f_{offs}. Equation (29) calculates the $MEWSD^{CT}$ between two different sequences. We have to take into account the whole set of M sequences and take an average among them. The formula (30) shows the Average Mean Excess Welch Square Distance $AMEWSD^{CT}$:

$$\mathrm{AMEWSD}^{\mathrm{CT}} = \frac{1}{M(M-1)}\sum_{i=1}^{M}\sum_{j=1}^{M}\mathrm{MEWSD}_{i,j}^{\mathrm{CT}}$$

[2] (30)

3.2.2 Tracking performance metric:

In the acquisition stage, to avoid any performance degradation, we considered only the highest correlation peaks but in the tracking stage, we have to consider all correlation function elements to evaluate the ultimate tracking performance of the receiver. Any non-ideal spreading sequence introduces an aggregate perturbation denominated average interference parameter. This parameter directly affects the average SNR in the receiver. To assess those effects, a tracking metric called Merit Factor (MF) is introduced.

We have two cases here: One for multipath based on auto-correlation function and the other one for cross-talk based on cross-correlation function (similar to MEWSD in the acquisition stage).

MF^{MP}:This metric is used to assess the multipath effect on the tracking performance and is based on the auto-correlation function. It is calculated through equation (31) as follows:

$$\mathrm{MF}_i^{\mathrm{MP}} = \frac{1}{n_{f_{offs}}} \sum_{f_{offs}} \left\{ \frac{1}{2} \left(\begin{array}{c} \sum\limits_{l=1,2,N-2,N-1} \left(AC_i^e\left(l,f_{offs}\right)\right)^2 + \\ \sum\limits_{l=1,2,N-2,N-1} \left(AC_i^o\left(l,f_{offs}\right)\right)^2 \end{array} \right) \right\}$$

[2] (31)

To assess the effect over all the sequence family, we have to take an average for all set of M sequences (average of M values of MF^{MP}). AMF^{MP} is computed as follows:

$$\mathrm{AMF}^{\mathrm{MP}} = \frac{1}{M} \sum_{i=1}^{M} \mathrm{MF}_i^{\mathrm{MP}}$$

[2] (32)

MF^{CT} : This metric is used to assess the effect of all non-desired signals from other satellites onto the tracking performance. It is computed through equation (33) as follows:

$$\begin{aligned} \mathrm{MF}_{i,j}^{\mathrm{CT}} = \frac{1}{n_{f_{offs}}} \sum_{f_{offs}} \Bigg(& \frac{1}{2} \sum_{l=0}^{N-1} \left(CC_{i,j}^e\left(l,f_{offs}\right)\right)^2 \\ & + \frac{1}{2} \sum_{l=0}^{N-1} \left(CC_{i,j}^o\left(l,f_{offs}\right)\right)^2 \Bigg) \end{aligned}$$

[2] (33)

In MF^{CT}, we consider all possible cross-correlation function peaks (for any delay or doppler shift) without any limitation concerning the bound of these doppler frequency offsets. Similar to what we had for MEWSD in acquisition performance metric, the sampling interval for MF^{CT} is a fifth of the inverse of the sequence period.

To assess the effect over all the sequence family, we have to take an average for all set of M sequences which is equal to the average of the M (M-1) elementary MF^{CT}). $AMF^{CT}$is computed as follows:

$$\mathrm{AMF}^{\mathrm{CT}} = \frac{1}{M(M-1)} \sum_{i=1}^{M} \sum_{j=1}^{M} \mathrm{MF}_{i,j}^{\mathrm{CT}}$$

[2] (34)

3.2.3 Robustness against narrowband interfering signals:

When the GNSS signal is generated by an ideal spreading sequence with an infinite sequence period, the power spectral density (Ideal PSD) must match with the exact pulse shape envelope of the spreading modulation. In reality, our spreading sequences have finite sequence periods. When we reduce the spreading sequence period (can be also translated to reducing the length of the sequence or truncation of the sequence), the power spectral density (PSD) related to these signals is not matched with the exact pulse shape envelope anymore and the peaks exceed this envelope. As a consequence, the receiver sensitivity against continuous wave or narrowband signals will increase.

So the spreading sequences with good robustness against narrowband interfering signals must have as few peaks which exceed the ideal PSD for the pulse shape envelope as possible.

To evaluate the above- mentioned characteristic, the Excess Line Weight (ELW) is defined (35). The goal of this definition is to reduce the peak exceeding. In other words, this metric measures the robustness against narrowband signals. Similar to what we have already seen in previous metrics, average values are computed to assess the effect over all the sequence family (36).

$$\mathrm{ELW} = 10\lg\left(\frac{1}{n}\underbrace{\sum_{k=-n/2}^{n/2}\left(A_k-\sqrt{n}\right)^2}_{A_k>\sqrt{n}}\right)$$

[2] (35)

A_k is the k^{th} value of the discrete Fourier transform (DFT) of the spreading sequences, C_i. The ELW calculates the averaged squared distances between the spectral line values and the value $(n)^{1/2}$, which would be obtained for an ideal random sequence of the infinite period. These averaged distances are only calculated for those spectral lines which exceed $(n)^{1/2}$.

The AELW for all set of M sequences is equal to the average of the M elementary ELW:

$$\mathrm{AELW} = \frac{1}{M}\sum_{i=1}^{M}\mathrm{ELW}_i$$

[2] (36)

3.3 Weighted Cost Function of Spreading Sequences:

Generally speaking, spreading sequence performance analysis should not focus entirely on a certain index. We need to assess the performance with several indexes thoroughly, and each of them is assigned a weighted value. In order to handle an indisputable framework, we can use a weighted cost function to assess and compare the sequence sets.

We define this weighted cost function R_i as follows: [2] (37)

$$R_i = \sum_{j=1}^{5} \rho_j \frac{cv_{i,j} - \overline{cv_j}}{\overline{cv_j}}, \quad i = 1, 2, \cdots \tag{37}$$

i is the index for different sequence sets and j is the index for different metrics.

$\overline{cvj}$ is the mean value of the metric j over all different sequence sets, $cv_{i,j}$ is the value of metric j and sequence set i and ρ_j is the weighted factor of metric j. The sum of $\rho_1, \rho_2, \rho_3, \rho_4, \rho_5$ is equal to 1. R_i , is the result of performance in percentage or weighted cost function.

The value of ρ_j or weighted value for each metric depends on the degree of importance of that metric among different metrics. Usually, the acquisition performance metric is more important. To assess and compare families of spreading sequences in our project, we take into account five performance metrics as follows:

$AMEWSD^{MP}$, $AMEWSD^{CT}$, AMF^{MP}, AMF^{CT}, AELW

For instance, the vector of weights equal to [0.2,0.2,0.2,0.2,0.2] means that the degree of importance of all above-mentioned performance metrics is equal. It is called a uniform weight vector. Since the acquisition performance metric is more important, we can increase corresponding elements in the vector of weights and define a new vector as follows: [0.35,0.35,0.1,0.1,0.1]. The first two weights correspond to $AMEWSD^{MP}$, $AMEWSD^{CT}$ which are acquisition performance metrics and the other three weights correspond to AMF^{MP}, AMF^{CT}, AELW respectively. As we mentioned before, the sum of all weights in the vector should be equal to 1.

Chapter 4

Performance Evaluation of the Gold Sequence and Truncated Gold Sequence

Our final goal in this thesis project is the optimization of the performance benchmarks by designing PRN codes (Random sequences) that can be suitable for the aforementioned new signal component considering different properties such as the autocorrelation, the cross-correlation, and the power spectral density. In chapter 2, we studied some popular spreading codes and the process of their generation.

In this chapter, we analyzed gold sequences periodic cross-correlation properties considering different code lengths and the different number of sequences. We reduced the length of the sequence by removing few bits from the end of the sequence (we truncated the sequence, so the sequence length, L, is less than N = $2^n - 1$) and computed the maximal cross-correlation absolute value between any possible pair of the sequences. We have already seen in chapter 2, that cross-correlation has three possible values: $-t, -1$, $t - 2$.

$t = 2^{(n+1)/2} + 1$ if n is an odd number

$t = 2^{(n+2)/2} + 1$ if n is an even number

A complete set of Gold sequences has length N = $2^n - 1$, cardinality K= $2^n + 1$, and maximal cross-correlation magnitude (maximum peak side lobe or MPSL), $r_c = t$.

We verified when the sequence length is N = $2^n - 1$, the maximal cross-correlation absolute value between any possible pair of the sequences can be computed through the formulas above.

In the case of full sequence length 31, the value of MPSL will be 9. After running the program for this length, we obtained a matrix with 33 rows and 33 columns and each element of this matrix except the main diagonal (the main diagonal elements are zero because the cross-correlation of a sequence with itself is zero) represents the absolute cross-correlation value between two different sequences. Since we have a full-length sequence, all these values are equal to 9.

In the case of full sequence length 127, the value of MPSL will be 17. After running the program for this length, we obtained a matrix with 129 rows and 129 columns and each element of this matrix except the main diagonal (the main diagonal elements are zero because the cross-correlation of a sequence with itself is zero) represents the absolute cross-correlation value between two different sequences. Since we have a full-length sequence, all these values are equal to 17.

In the case of full sequence length 1023, the value of MPSL will be 65. After running the program for this length, we obtained a matrix with 1025 rows and 1025 columns and each element of this matrix except the main diagonal (the main diagonal elements are zero because the cross-correlation of a sequence with itself is zero) represents the absolute cross-correlation value between two different sequences. Since we have a full-length sequence, all these values are equal to 65.

We have already seen that truncated m-sequences were not ideal and they did not have good correlation properties. When we truncate the sequence, the matrix of absolute cross-correlation will have several values for different pairs (the main diagonal elements are all zero) and these values are not necessarily equal (in full-length gold sequence all values were equal). The next step is, to find the maximum value among all elements of the matrix and compare it with the corresponding value of the full-length sequence. Obviously, the smaller the maximum absolute value of cross-correlation, the better will be the performance.

Since our primary solution could affect both time convergence and final solution performance, we started from gold sequences analysis (under the minimum cross-correlation function side lobe condition).

4.1 Gold Sequence Construction with MATLAB:

We can generate gold sequences in MATLAB as follows:

H = comm.GoldSequence creates a Gold sequence generator System object, H. This object generates a pseudo-random Gold sequence. [4]

```
goldseq = comm.GoldSequence('FirstPolynomial','x^3+x+1',...
        'SecondPolynomial','x^3+x^2+1', ...
    'FirstInitialConditions', [0 0 1], ...
    'SecondInitialConditions', [ 0 0 1], ...
    'Index',4,'SamplesPerFrame', frame);
x1 = goldseq ()';
```

Properties:

First Polynomial: [4]

Generator polynomial for first preferred PN sequence.

First Initial Conditions: [4]

Initial conditions for first PN sequence generator. Specify the initial conditions for the shift register of the first preferred PN sequence generator. The default is [0 0 1]. The initial conditions require a numeric, binary scalar, or a numeric, binary vector with length equal to the degree of the first generator polynomial specified in the first polynomial property.

Second Polynomial: [4]

Generator polynomial for second preferred PN sequence.

Second Initial Conditions: [4]

Initial conditions for second PN sequence generator. Specify the initial conditions for the shift register of the second preferred PN sequence generator.

The default is [0 0 1]. The initial conditions require a numeric, binary scalar, or a numeric, binary vector with length equal to the degree of the second generator polynomial specified in the second polynomial property.

Index: [4]

Index of output sequence of interest. Specify the index of the output sequence of interest from the set of available sequences as a scalar integer. The default is 0. The scalar integer must be in the range [–2 , $2^n - 2$], where n is the degree of the generator polynomials you specify in the first polynomial and second polynomial properties.

The index values -2 and -1 correspond to the first and second preferred PN sequences as generated by the First Polynomial and Second Polynomial, respectively.

Samples Per Frame: [4]

Number of output samples per frame

Specify the number of Gold sequence samples that the step method outputs as a numeric, integer scalar value. The default is 1. If you set this property to a value of M, then the step method outputs M samples of a Gold sequence with a period of N =$2^n - 1$. The value of n represents the degree of the generator polynomials that you specify in the first polynomial and second polynomial properties.

4.2 Periodic Cross-Correlation of the Gold sequence with MATLAB:

The program computes the periodic cross-correlation for gold sequences.

First polynomial and second polynomial depend on the polynomial's degree (3,5,7,10).

Table 4.1 shows these polynomials for the gold sequence generation process.

Table 4.1

Degree	First polynomial	Second polynomial
3	$1+x^1+x^3$	$1+x^2+x^3$
5	$1+x^2+x^5$	$1+x^2+x^3+x^4+x^5$
7	$1+x^3+x^7$	$1+x^1+x^2+x^3+x^7$
10	$1+x^3+x^{10}$	$1+x^2+x^3+x^8+x^{10}$

```
% Periodic Cross-Correlation for Gold sequences%
clc;
clear all;
close all;

printf ("periodic cross correlation for Gold sequences \n \n");

cell=10; % number of cells, but we could have the same number of cells
with different polynomial or seed

frame = 1000; % code length

%%%%%%%%%%%%%%%

if (cell==3)
L=3;
goldseq = comm.GoldSequence('FirstPolynomial','x^3+x+1',...
        'SecondPolynomial','x^3+x^2+1’, ...
    'FirstInitialConditions’, [0 0 1], ...
    'SecondInitialConditions’, [ 0 0 1], ...
    'Index',4,'SamplesPerFrame’, frame);
x1 = goldseq ()';

goldseq = comm.GoldSequence('FirstPolynomial','x^3+x+1',...
        'SecondPolynomial','x^3+x^2+1’, ...
    'FirstInitialConditions’, [0 0 1], ...
    'SecondInitialConditions’, [ 0 0 1], ...
    'Index',10,'SamplesPerFrame’, frame);
x2 = goldseq ()';

end

%%%%%%%%%%%%%%%

if (cell==5)
L=5;
goldseq = comm.GoldSequence('FirstPolynomial','x^5+x^2+1',...
        'SecondPolynomial','x^5+x^4+x^3+x^2+1',...
    'FirstInitialConditions’, [0 0 0 0 1], ...
    'SecondInitialConditions’, [0 0 0 0 1], ...
    'Index',4,'SamplesPerFrame’, frame);
x1 = goldseq ()';

goldseq = comm.GoldSequence('FirstPolynomial','x^5+x^2+1',...
        'SecondPolynomial','x^5+x^4+x^3+x^2+1',...
    'FirstInitialConditions’, [0 0 0 0 1], ...
    'SecondInitialConditions’, [0 0 0 0 1], ...
    'Index',10,'SamplesPerFrame’, frame);

x2 = goldseq ()';
```

```
 end
%%%%%%%%%%%%%%%

if (cell==7)
L=7;
goldseq = comm.GoldSequence('FirstPolynomial','x^7+x^3+1',...
        'SecondPolynomial','x^7+x^3+x^2+x+1',...
    'FirstInitialConditions’, [0  0  0  0  0  0  1], ...
    'SecondInitialConditions’, [0  0  0  0  0  0  1], ...
    'Index',4,'SamplesPerFrame’, frame);
x1 = goldseq ()';

goldseq = comm.GoldSequence('FirstPolynomial','x^7+x^3+1',...
        'SecondPolynomial','x^7+x^3+x^2+x+1',...
    'FirstInitialConditions’, [0  0  0  0  0  0  1], ...
    'SecondInitialConditions’, [0  0  0  0  0  0  1], ...
    'Index',10,'SamplesPerFrame’, frame);

x2 = goldseq ()';

end

%%
if (cell==10)
L=10;
for nn1=1:(2^L+1)
    nn1
goldseq = comm.GoldSequence('FirstPolynomial','x^10+x^3+1',...
    'SecondPolynomial','x^10+x^8+x^3+x^2+1',...
    'FirstInitialConditions’, [0  0  0  0  0  0  0  0  0  1], ...
    'SecondInitialConditions’, [0  0  0  0  0  0  0  0  0  1], ...
    'Index’, nn1-3,'SamplesPerFrame’, frame);
x1 = goldseq ()';

for nn2=(nn1+1): ((2^L+1))
    nn1
    nn2
goldseq = comm.GoldSequence('FirstPolynomial','x^10+x^3+1',...
    'SecondPolynomial','x^10+x^8+x^3+x^2+1',...
    'FirstInitialConditions’, [0  0  0  0  0  0  0  0  0  1], ...
    'SecondInitialConditions’, [0  0  0  0  0  0  0  0  0  1], ...
    'Index’, nn2-3,'SamplesPerFrame’, frame);

x2 = goldseq ()';

%end

x1b=2*x1-1; % 0 --> -1    1 --> +1
x2b=2*x2-1; % 0 --> -1    1 --> +1

cc=ifft(fft(x1b).*conj(fft(x2b))); %Normalized periodic autocorrelation
```

```
cc2= [fliplr (cc (2: frame)), cc]; %Same but symmetric
tau=(-(frame-1):1:(frame-1)); % tau axis

MPSL=max(abs(cc));
Res (nn1, nn2) =MPSL;
Res (nn2, nn1) =MPSL;

figure
plot (tau, cc2,'b-o');
hold on;
grid on;
ylabel('R(\tau)');
xlabel('\tau');
title (sprintf ('Periodic Cross-Correlation\n Gold-sequence, LFSR number
of cells = %d, code length = %d\nMax Peak Side Lobe = %3.1f’,
L,frame,MPSL));
end
end
end
```

4.3 Simulation Results and Analysis

4.3.1 The Matrix of Absolute Cross-Correlation Values:

The following part of the program shows how to obtain the matrix of cross-correlation absolute values between any pair of sequences. The polynomial degree is equal to 10, sequence length is 1000 chips (truncated by the last 23 bits), cardinality or number of sequences in the set is 1025, the indexes of sequences are from -2 to 1022, and the matrix of absolute cross-correlation values has 1025 rows and 1025 columns whereas, the main diagonal values are all zero. We have to find the maximum value of all the elements of this matrix. The maximum absolute value among all the elements of this matrix is 170. By comparing this value with the corresponding MPSL value of the full-length sequence we can say that truncated gold sequences are not ideal, and they do not have good correlation properties. The above-mentioned matrix was obtained after running the program for several hours.

```
if (cell==10)
L=10;
for nn1=1:(2^L+1)
    nn1
goldseq = comm.GoldSequence('FirstPolynomial','x^10+x^3+1',...
    'SecondPolynomial','x^10+x^8+x^3+x^2+1',...
    'FirstInitialConditions’, [0 0 0 0 0 0 0 0 0 1], ...
    'SecondInitialConditions’, [0 0 0 0 0 0 0 0 0 1], ...
    'Index’, nn1-3,'SamplesPerFrame’, frame);
x1 = goldseq ()';
for nn2=(nn1+1): ((2^L+1))
    nn1
    nn2
```

```
goldseq = comm.GoldSequence('FirstPolynomial','x^10+x^3+1',...
    'SecondPolynomial','x^10+x^8+x^3+x^2+1',...
    'FirstInitialConditions’, [0 0 0 0 0 0 0 0 0 1], ...
    'SecondInitialConditions’, [0 0 0 0 0 0 0 0 0 1], ...
    'Index’, nn2-3,'SamplesPerFrame’, frame);

x2 = goldseq ()';

x1b=2*x1-1; % 0 --> -1    1 --> +1
x2b=2*x2-1; % 0 --> -1    1 --> +1

cc=ifft(fft(x1b).*conj(fft(x2b))); %Normalized periodic autocorrelation

cc2= [fliplr (cc (2: frame)), cc]; %Same but symmetric
tau=(-(frame-1):1:(frame-1)); % tau axis
MPSL=max(abs(cc));
Res (nn1, nn2) =MPSL;
Res (nn2, nn1) =MPSL;
```

The following matrix as a whole is huge (1025 rows and 1025 columns) but, the first 10 rows and 10 columns for this sequence length which equals to 1000 chips are shown as follows:

```
0     116   96    98    114   104   102   96    106   100
116   0     116   110   110   104   106   100   110   104
96    116   0     126   114   104   102   128   122   92
98    110   126   0     108   98    96    94    108   102
114   110   114   108   0     102   144   110   100   110
104   104   104   98    102   0     94    116   102   120
102   106   102   96    144   94    0     98    108   106
96    100   128   94    110   116   98    0     94    124
106   110   122   108   100   102   108    94   0     106
100   104   92    102   110   120   106   124   106    0
```

The first 10 rows and 10 columns for full-length sequence (1023 chips) are shown as follows:

```
0     65    65    65    65    65    65    65    65    65
65    0     65    65    65    65    65    65    65    65
65    65     0    65    65    65    65    65    65    65
65    65    65    0     65    65    65    65    65    65
65    65    65    65    0     65    65    65    65    65
65    65    65    65    65    0     65    65    65    65
65    65    65    65    65    65    0     65    65    65
65    65    65    65    65    65    65    0     65    65
65    65    65    65    65    65    65    65    0     65
65    65    65    65    65    65    65    65    65    0
```

It is clear that whenever we truncated the sequence, we had several values for the peak side lobe but, in the case of full-length sequences (1023,127,31), we could easily verify that all the elements of the matrix (except the main diagonal which were always zero) were equal to t and could be calculated through (15), (16).

In order to have a better idea, we repeated the calculation of the matrix of cross-correlation absolute values. This time, the polynomial degree was 10, cardinality or number of sequences was 1025, the indexes of sequences were from -2 to 1022, and the matrix of cross-correlation absolute values had 1025 rows and 1025 columns whereas, the main diagonal values were all zero. The only difference was the sequence length which we set it to 700 chips (truncated by the last 323 bits). We obtained 150 for the maximum value among all the elements of the matrix. This value was reduced by 20 units with respect to the previous case, which was 170 (full-length sequence was truncated by the last 23 bits). We may say that when we truncated more bits (300 bits more), the maximum peak side lobe value decreased but we have to consider the fact that at the same time the sequence length was decreased too. In other words, in the case of full-length sequence with 1023 bits this value was 65, which means in the worst case which is fixed for any pair of the sequences in our set, after comparison between any possible pair of the sequences of this length, 65 bits are correlated. In terms of percentage, around 6.3% of bits are correlated along the length of the sequences. When we truncated our sequence by the last 23 bits, we obtained that in the worst-case scenario 170 out of 1000 bits are correlated. So in terms of percentage, 17% of bits are correlated along the length of the sequence. Then we truncated the sequences by the last 323 bits and we obtained that in the worst case 150 out of 700 bits are correlated which in terms of percentage will be around 21.4%. So comparing to the full-length code's MPSL value, the truncated sequences are not ideal again.

The following matrix as a whole is huge (1025 rows and 1025 columns) but, the first 10 rows and 10 columns for this sequence length which equals to 700 chips are shown as follows:

```
0     82    86    96    86    98    92    80    74    88
82    0     80    82    80    92    86    90    92    98
86    80     0    90    88    88    86    98    88    82
96    82    90     0    90    94    108   80    90    84
86    80    88    90    0     88    94    82    88    82
98    92    88    94    88    0     98    78    76    98
92    86    86    108   94    98    0     112   82    84
80    90    98    80    82    78    112   0     86    80
74    92    88    90    88    76    82    86    0     98
88    98    82    84    82    98    84    80    98    0
```

By analyzing the above matrix, we can observe that almost all cross-correlation absolute values are less than the previous experiment (when sequence length was 1000 bits) but at the same time the sequence length is 700 which is 300 bits less than 1000.

The most important result: whenever we truncated the full-length gold sequence for more bits, the final performance degraded more as well.

At this point, we repeated the calculation of the matrix of cross-correlation absolute values between any pair of the truncated sequences. This time, the polynomial degree was 10, cardinality or number of sequences was 1025, the indexes of sequences are from -2 to 1022, and the matrix of cross-correlation absolute values had 1025 rows and 1025 columns whereas, the main diagonal values were all zero. The only difference was the sequence length which we set it to 400 chips (truncated by the last 623 bits). We obtained 120 for the maximum value among all the elements of the matrix. This value was reduced by 30 units concerning the previous case, which was 150 (the full-length sequence was truncated by the last 323 bits). We may say that when we truncated more bits (300 bits more), the maximum peak side lobe value decreased but we have to consider the fact that at the same time the sequence length was decreased too. We truncated our sequence by the last 623 bits. We obtained that in the worst-case scenario 120 out of 400 bits are correlated. In terms of percentage 30% of bits are correlated along the length of the sequence.
So comparing to the full-length code's MPSL value, the truncated sequences are not ideal again.

The following matrix as a whole is huge (1025 rows and 1025 columns) but, the first 10 rows and 10 columns for this sequence length which equals to 400 chips are shown as follows:

```
0     64    60    68    68    66    76    72    66    60
64    0     68    60    72    62    64    64    62    60
60    68     0    72    56    62    64    60    58    68
68    60    72     0    72    62    72    76    66    56
68    72    56    72    0     66    60    64    58    56
66    62    62    62    66    0     78    54    64    66
76    64    64    72    60    78    0     64    70    64
72    64    60    76    64    54    64    0     50    60
66    62    58    66    58    64    70    50    0     62
60    60    68    56    56    66    64    60    62    0
```

By analyzing the above matrix, we can observe that almost all cross-correlation absolute values are less than the previous experiment (when sequence length was 700 bits) but at the same time the sequence length is 400 which is 300 bits less than 700.

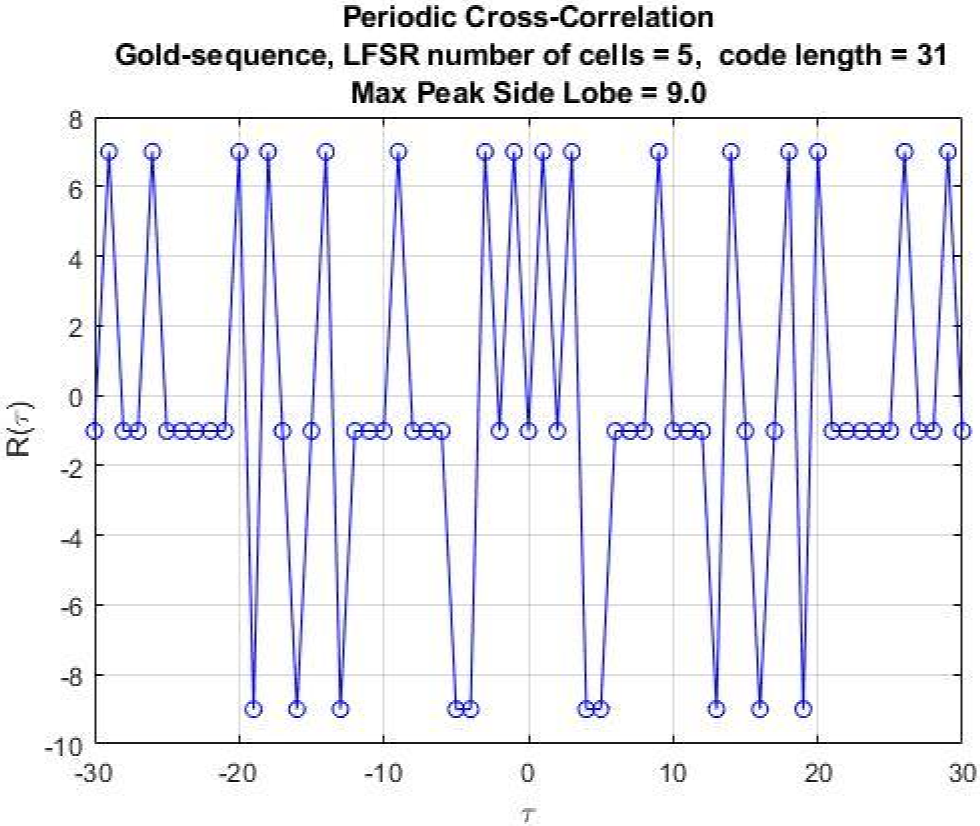
The most important result: whenever we truncated the full-length gold sequence for more bits, the final performance degraded more as well.

4.3.2 Cross-Correlation Properties of the Full Length Gold Sequence and Truncated Gold Sequence

Figure 4.1 shows the periodic cross-correlation of Gold sequences with LFSR number of cells=5 and code length =31.

In Figure 4.2, we reduced the length of the sequence to 20, with the same number of cells to analyze periodic cross-correlation behavior for Truncated Gold-sequence (the sequence length is reduced to L smaller than N=$2^n - 1$).

Figure 4.1



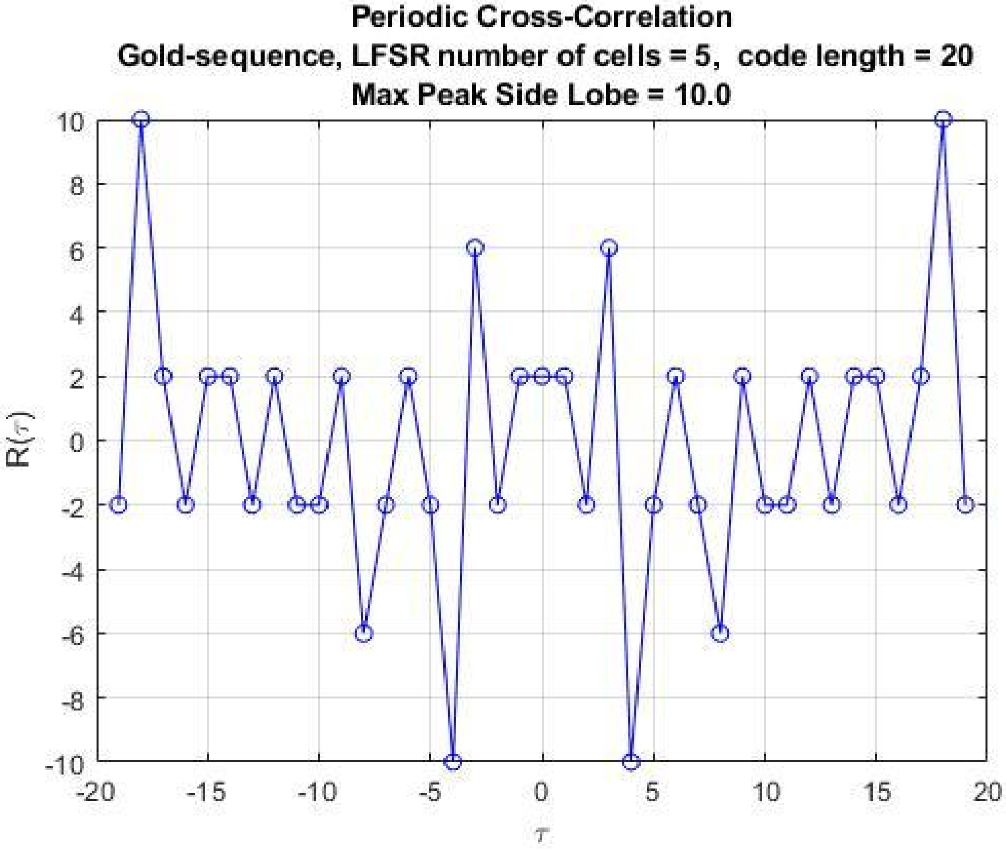
The Maximum Peak Side Lobe for periodic cross-correlation function is 9.

Maximal Cross-correlation magnitude (maximum peak side lobe or MPSL) = *t*.

Since n is 5, $t = 2^{(n+1)/2} + 1$. After calculations, we obtained the value of 9.

So we verified the cross-correlation property for the full-length sequence when the number of cells is 5.

Figure 4.2

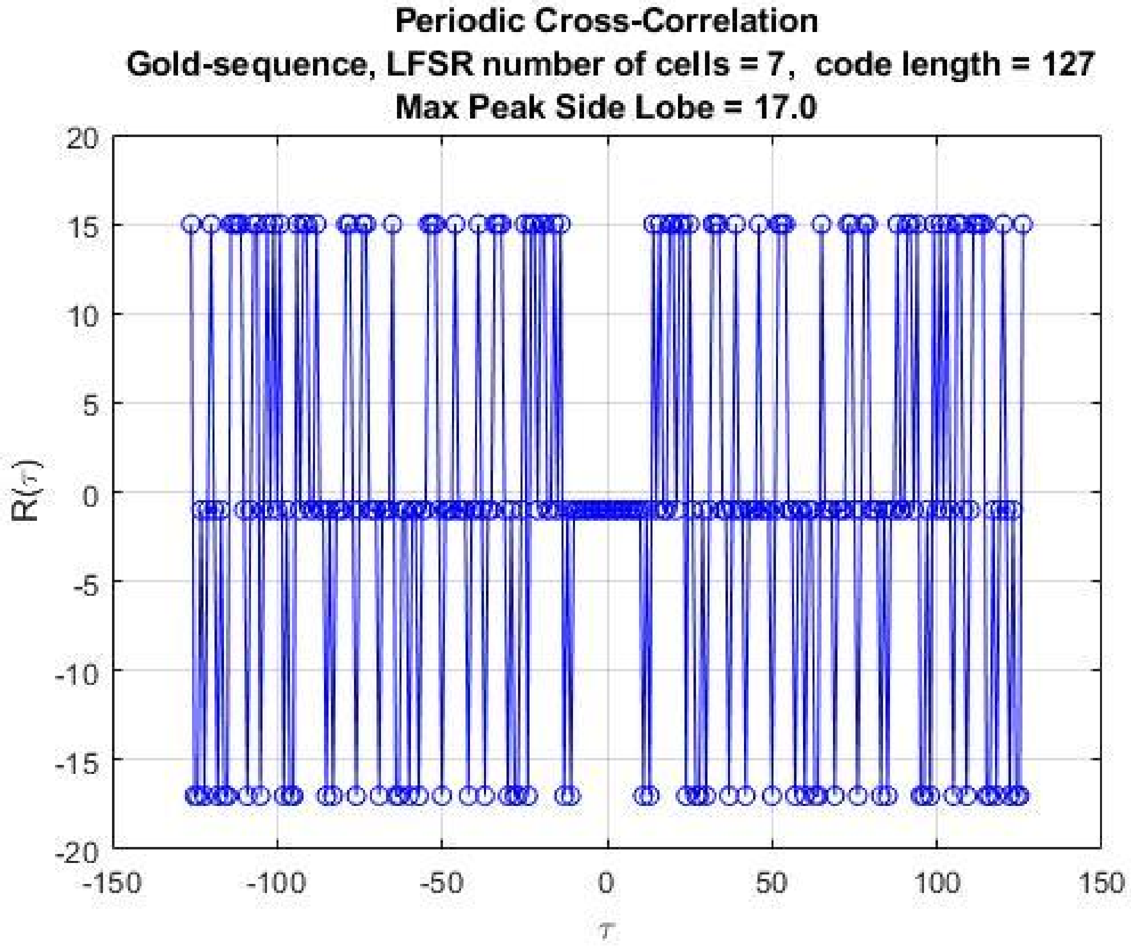


We can observe after the sequence is truncated by the last 11 chips, the Maximum Peak Side Lobe increased to 10 (in case of full-length sequence was 9), which means that by reducing the sequence length, the target performance degraded. The optimization goal is to design a sequence which minimizes the cross-correlation function side lobe as much as possible. In the case of full-length sequence with 31 bits, MPSL is 9 which means in the worst case which is fixed for any pair of the sequences in our set, after comparison between any possible pair of the sequences of this length, 9 bits are correlated. In terms of percentage, around 29% of bits are correlated along the length of the sequences which is much worse than 6.3% (the value we obtained when the sequence length was 1023 bits). When we truncated our sequence by the last 11 bits, we obtained that in the worst-case scenario 10 out of 20 bits are correlated. So in terms of percentage 50% of bits are correlated along the length of the sequence which is not good at all.
As a result, whenever we truncated the full-length gold sequence for the last few bits, the final performance degraded as well.

Figure 4.3 shows the periodic cross-correlation of Gold sequences with LFSR number of cells=7 and code length =127.

In Figure 4.4, we reduced the length of the sequence to 100, with the same number of cells to analyze periodic cross-correlation behavior for Truncated Gold-sequence (the sequence length is reduced to L smaller than N=$2^n - 1$).

Figure 4.3



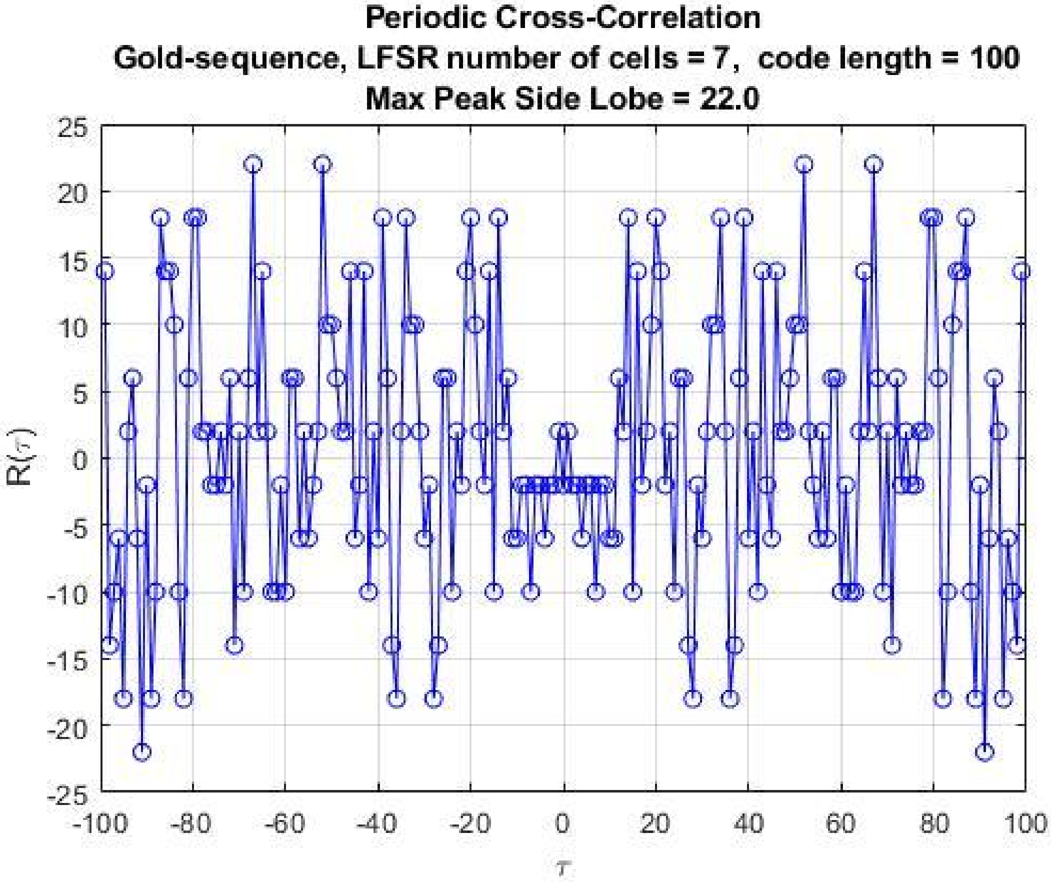
The Maximum Peak Side Lobe for periodic cross-correlation function is 17.

Maximal cross-correlation magnitude (maximum peak side lobe or MPSL) = *t*.

Since n is 7, $t = 2^{(n+1)/2} + 1$. After calculations, we obtained the value of 17.

So we verified the cross-correlation property for the full-length sequence when the number of cells is 7.

Figure 4.4

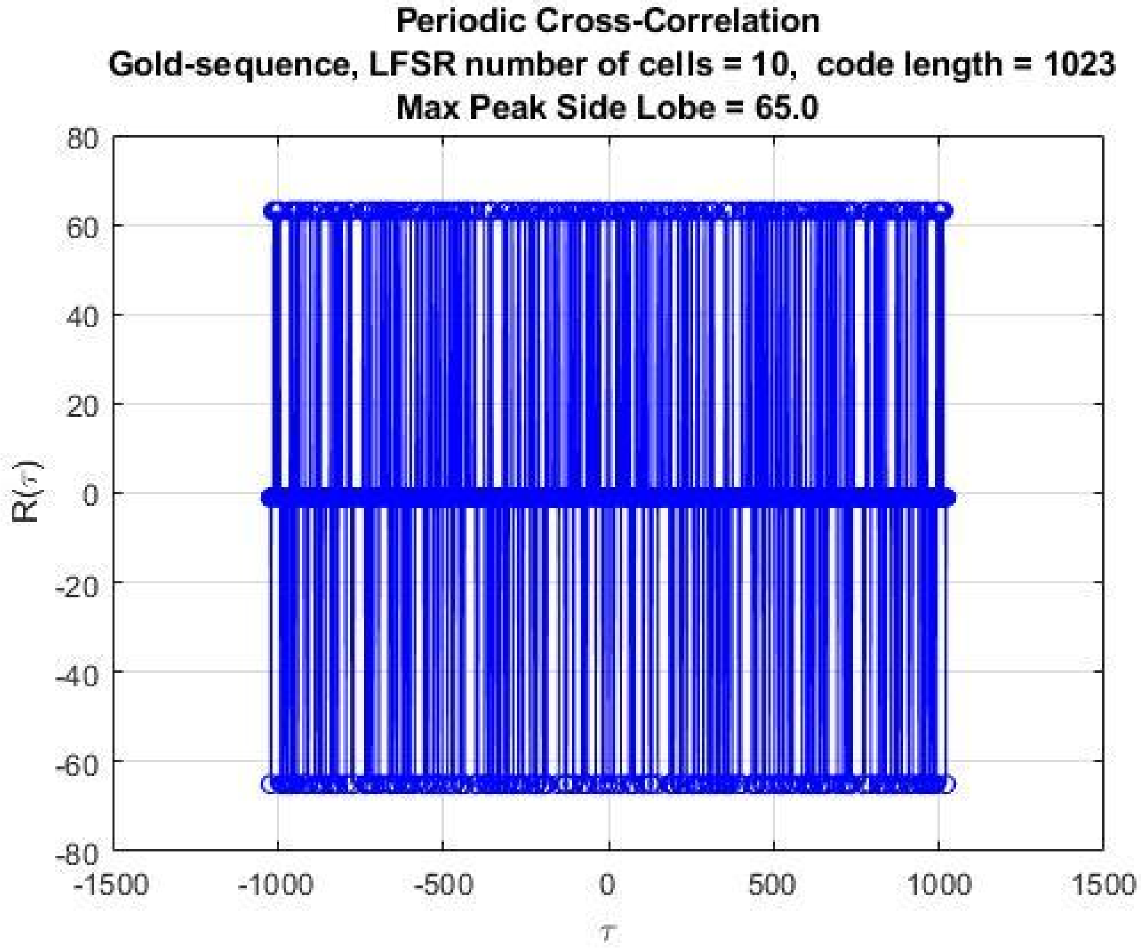


We can observe after the sequence is truncated by the last 27 chips, the Maximum Peak Side Lobe increased to 22 (in the case of full-length sequence was 17), which means that by reducing the sequence length, the target performance degraded. The optimization goal is to design a sequence which minimizes the cross-correlation function side lobe as much as possible. In the case of full-length sequence with 127 bits, MPSL is 17 which means in the worst case which is fixed for any pair of the sequences in our set, after comparison between any possible pair of the sequences of this length, 17 bits are correlated. In terms of percentage, around 13.4% of bits are correlated along the length of the sequences which is worse than 6.3% (the value we obtained when the sequence length was 1023 bits) and better than 29% (the value we obtained when the sequence length was 31 bits). When we truncated our sequence by the last 27 bits, we obtained that in the worst-case scenario 22 out of 100 bits are correlated. So in terms of percentage 22% of bits are correlated along the length of the sequence which is not good at all.

As a result, whenever we truncated the full-length gold sequence for the last few bits, the final performance degraded as well.

Figure 4.5 shows the periodic cross-correlation of Gold sequences with LFSR number of cells=10 and code length =1023.

Figure 4.5



The Maximum Peak Side Lobe for periodic cross-correlation function is 65.

Maximal cross-correlation magnitude (maximum peak side lobe or MPSL) = t.

Since n is 10, $t = 2^{(n+2)/2} + 1$. After calculations, we obtained the value of 65.

So we verified the cross-correlation property for the full-length sequence when the number of cells is 10.

According to the results which we obtained from previous sequence lengths (31,127 and then their truncated versions 20 and 100 respectively), we expect that for the sequence length equal to 1023 chips truncated versions will have a larger maximum cross-correlation side lobe (we reduced the length of the sequence to 1000 bits by removing the last 23 bits and then to 700 bits by removing the last 323 bits). Note: In 4.3 we have already seen the cross-correlation matrix of the full-length sequence of 1023bits and its truncated versions (1000,700 and 400 bits).

In Figure 4.6, we reduced the length of the sequence to 1000, with the same number of cells to analyze periodic cross-correlation behavior for Truncated Gold-sequence (the sequence length is reduced to L smaller than N=$2^n - 1$ =1023). The maximum peak side lobe is 170. The value in the plot which is 104, is obtained after applying cross-correlation function between just two different gold sequences (2 indexes selected randomly among all 1025 possible indexes) with the same initial seed value and same sequence length equal to 1000 bits each.

In Figure 4.7, we reduced the length of the sequence even more to 700, with the same number of cells to analyze periodic cross-correlation behavior for Truncated Gold-sequence (the sequence length is reduced to L smaller than N=$2^n - 1$ =1023). The maximum peak side lobe is 150. The value in the plot which is 92, is obtained after applying cross-correlation function between just two different gold sequences (2 indexes selected randomly among all 1025 possible indexes) with the same initial seed value and same sequence length equal to 700 bits each.

Figure 4.6

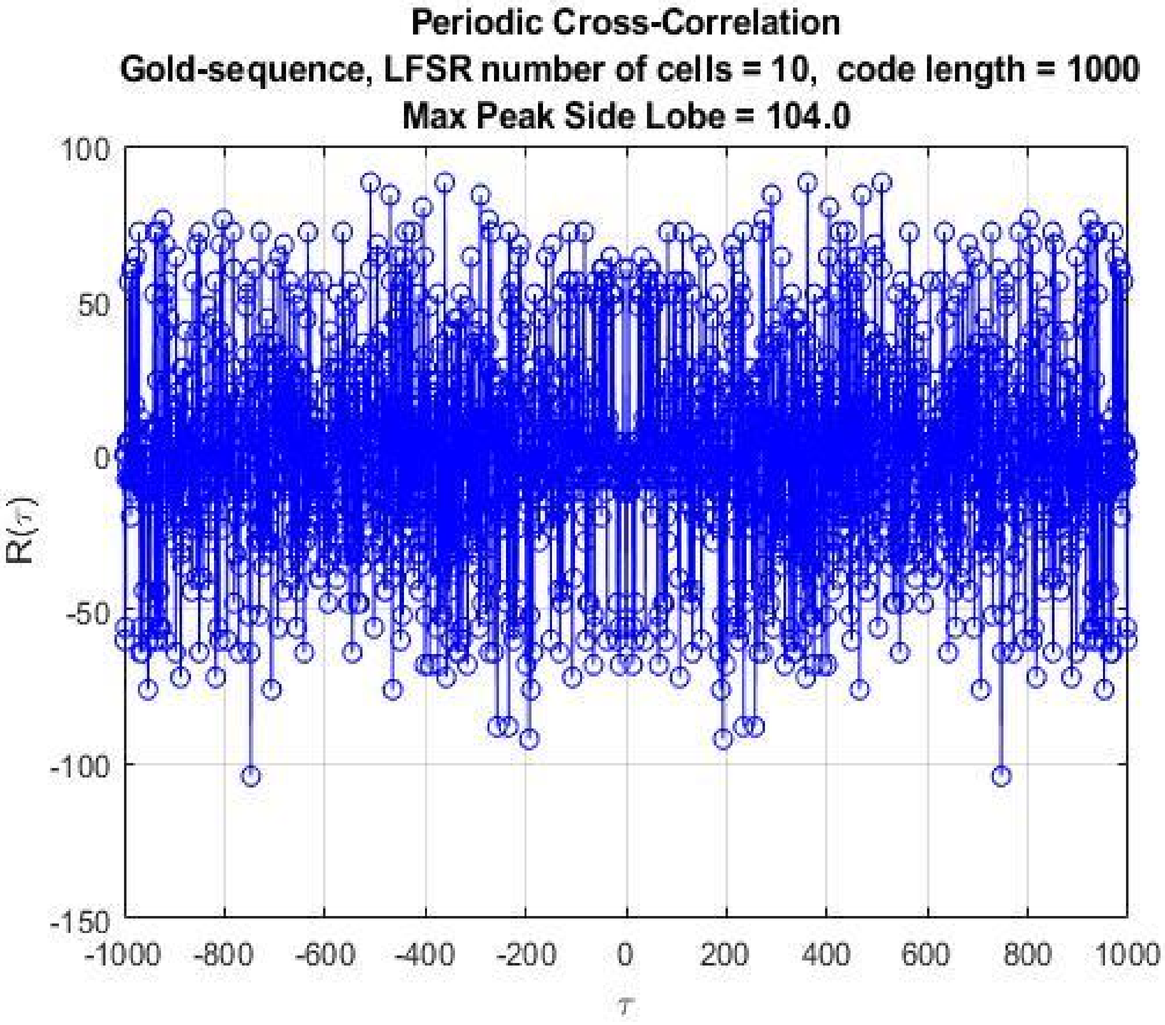
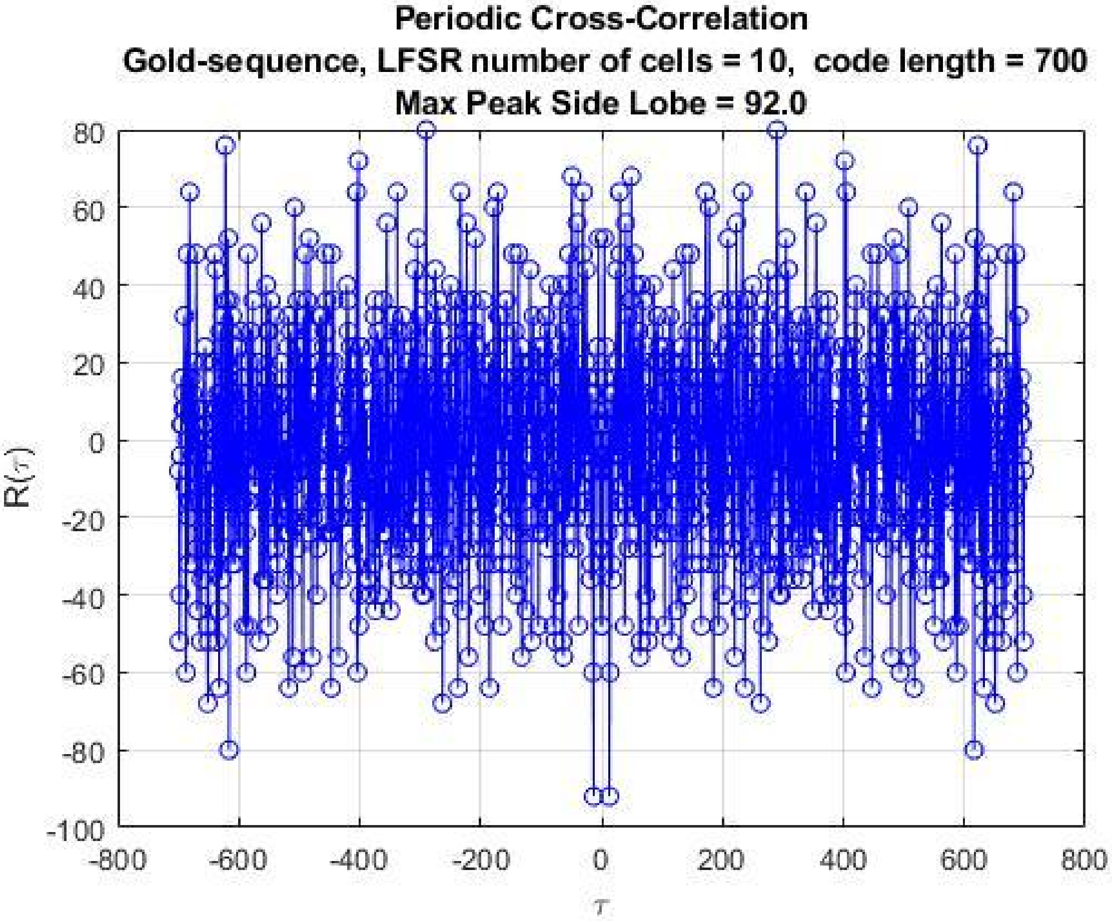


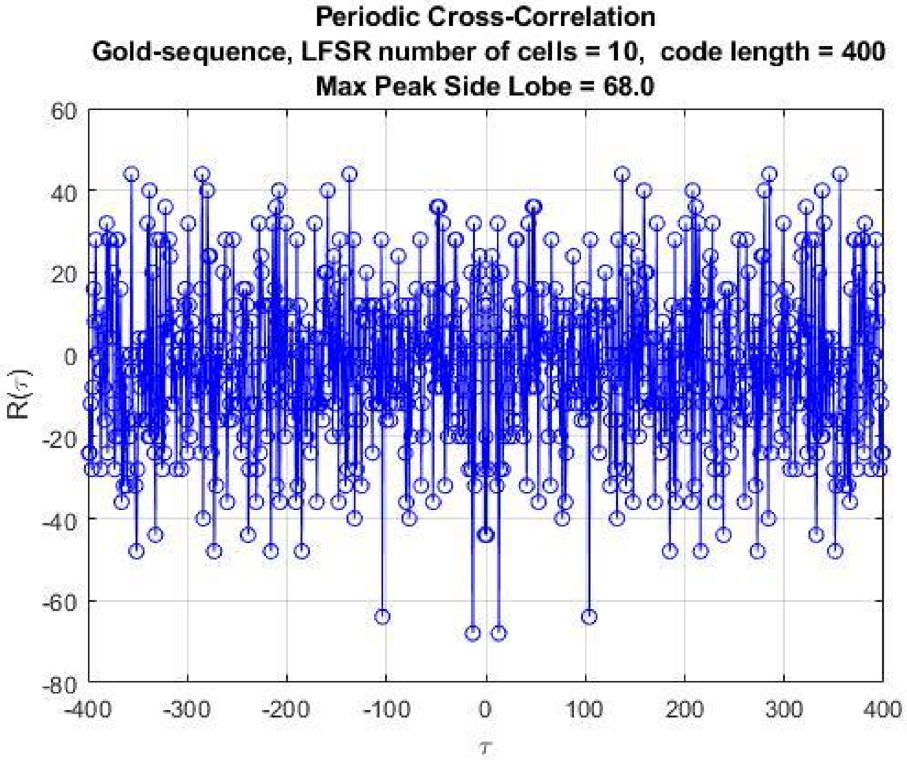
Figure 4.7



We can observe that the maximum peak side lobe value decreased after we reduced the length for another 300 chips (total 323 bits were truncated from the end of the sequence). It is very important to notice that as we mentioned before several times, the sequence length decreased and we discussed this sequence length specifically on page 29. We reduced the sequence length again for another 300 chips and computed the periodic cross-correlation for this new truncated version with sequence length equal to 400 chips.

In Figure 4.8, we reduced the length of the sequence even more (by removing the last 623 bits), with the same number of cells to analyze periodic cross-correlation behavior for Truncated Gold-sequence (the sequence length is reduced to L smaller than N=$2^n - 1$ =1023). The maximum peak side lobe is 120. It means in the worst case 120 out of 400 bits are correlated and in terms of percentage is equal to 30%. The value in the plot which is 68, is obtained after applying cross-correlation function between just two different gold sequences (2 indexes selected randomly among all 1025 possible indexes) with the same initial seed value and same sequence length equal to 400 bits each.

Figure 4.8



Note: In all four different lengths (1023,1000,700,400), we kept the number of cells fixed equal to 10 and a fixed initial seed value [0 0 0 0 0 0 0 0 0 1].

Table 4.2 shows the MPSLs values obtained for different sequence lengths and corresponding welch bounds (M number of sequences=1025 and N=sequence length)

Table 4.2

Sequence length	MPSL	φ min (Welch bound)	Worst-case analysis (correlated bits along the length of the sequence)
1023	65	31.96	6.3%
1000	170	31.6	17%
700	150	26.44	21.4%
400	120	19.99	30%

The values in Table 4.2 are obtained after applying cross-correlation function between any possible pair of gold sequences with the same initial seed value and same sequence length. In fact, for full-length sequences with 1023 chips, we have already seen that the absolute value of cross-correlation for any possible pair of them is 65. We also ran several simulations for truncated sequences by removing the last 23 bits, the last 323 bits, and the last 623 bits. We have seen that the values of the maximum peak side lobe are 170,150 and 120 respectively. (these values are the maximum among all peak side lobe or cross-correlation absolute values in the result matrixes). We also computed the welch bound for each sequence length according to (26) which was introduced in the previous chapter considering in all cases the number of sequences is 1025 (M=1025). After comparing the welch bound values and maximum peak side lobes for each sequence length in table 4.2, we could easily find that for the full-length Gold sequence with length equal to 1023 bits, the corresponding MPSL value was the closest to its Welch's lower bound. It means that the full-length Gold sequence exceeded the welch bound less than its truncated versions which is better in terms of performance. We also calculated the percentage of correlated bits along the length of the sequences in the worst case.

As a final result of all previous steps, different cell sizes, and different sequence lengths, we can say that peak side lobe values increase almost linearly with the number of truncated bits and at the same time we can observe more instability in plots whenever the number of truncated bits are increasing. We also observed that number of correlated bits with respect to the length of the sequence increased significantly. Many sequences have been generated after we changed the truncation size (23,323,623).

4.3.3 Maximum Peak Side Lobe Analysis with the Change of Initial Seed Values in Truncated Gold Sequences

It may be noted that truncated sequences also change quite significantly with the change of initial seed value (In all simulations the initial seeds were [0 0 0 0 0 0 0 0 0 1]). In the case of full-length sequence with 1023 bits, we changed the initial seed values and the result was 65. It means that for the full-length gold sequence maximum peak side lobe value did not change after feeding our polynomials with different initial seeds. To analyze if there are some variations of peak side lobe values with the change of initial seed values in truncated gold sequence (even at a particular value of bit truncation), we changed the initial seeds to [0 0 0 0 0 1 1 0 0 0] and [0 0 0 0 1 1 1 0 0 0].

The sequence length was set to 1000 with a polynomial degree equal to 10 (a truncated version of the full-length sequence by removing the last 23 bits). To have a precise result, we kept randomly selected indexes (-2 to 1022) the same (indexes 4 and 10). In order to plot the curves in previous parts, we chose those indexes in the program when all the initial seeds were [0 0 0 0 0 0 0 0 0 1]. After running the program, we obtained the result which is shown in figure 4.9.

Then the sequence length was set to 700 with a polynomial degree equal to 10 (a truncated version of the full-length sequence by removing the last 323 bits) with these new initial seeds. After running the program, we obtained the result which is shown in figure 4.10.

Finally, the sequence length was set to 400 with a polynomial degree equal to 10 (a truncated version of the full-length sequence by removing the last 623 bits) with these new initial seeds. After running the program, we obtained the result which is shown in figure 4.11.

Figure 4.9

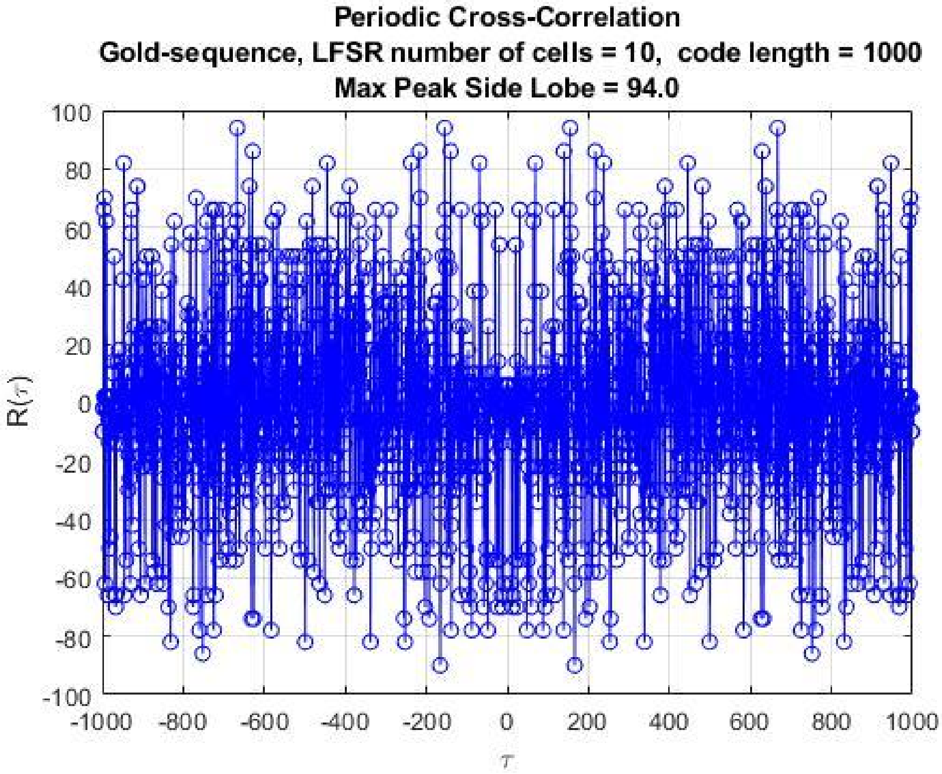


Figure 4.10

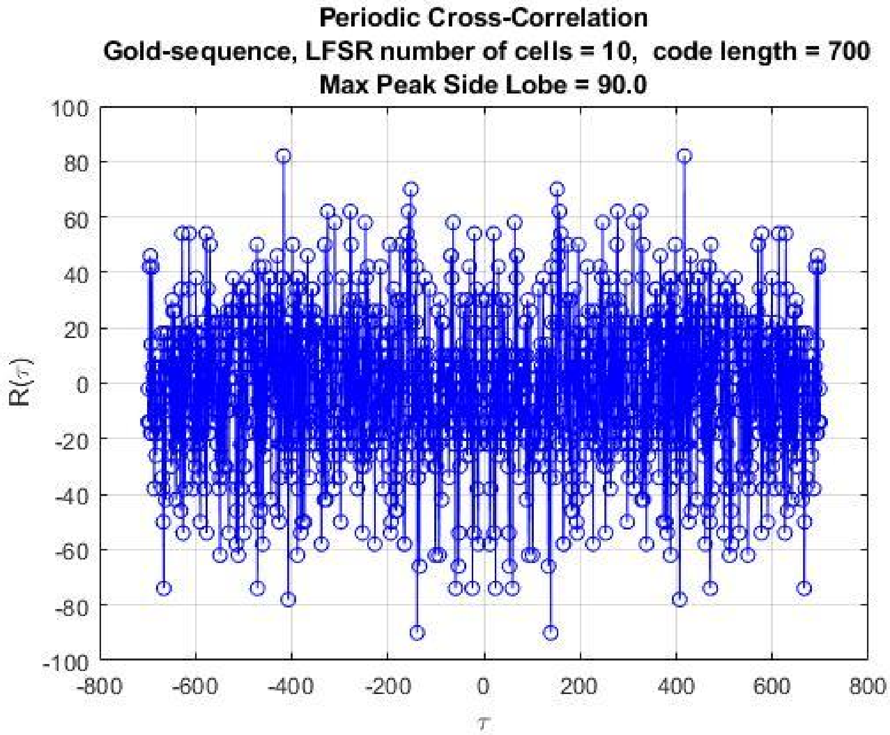
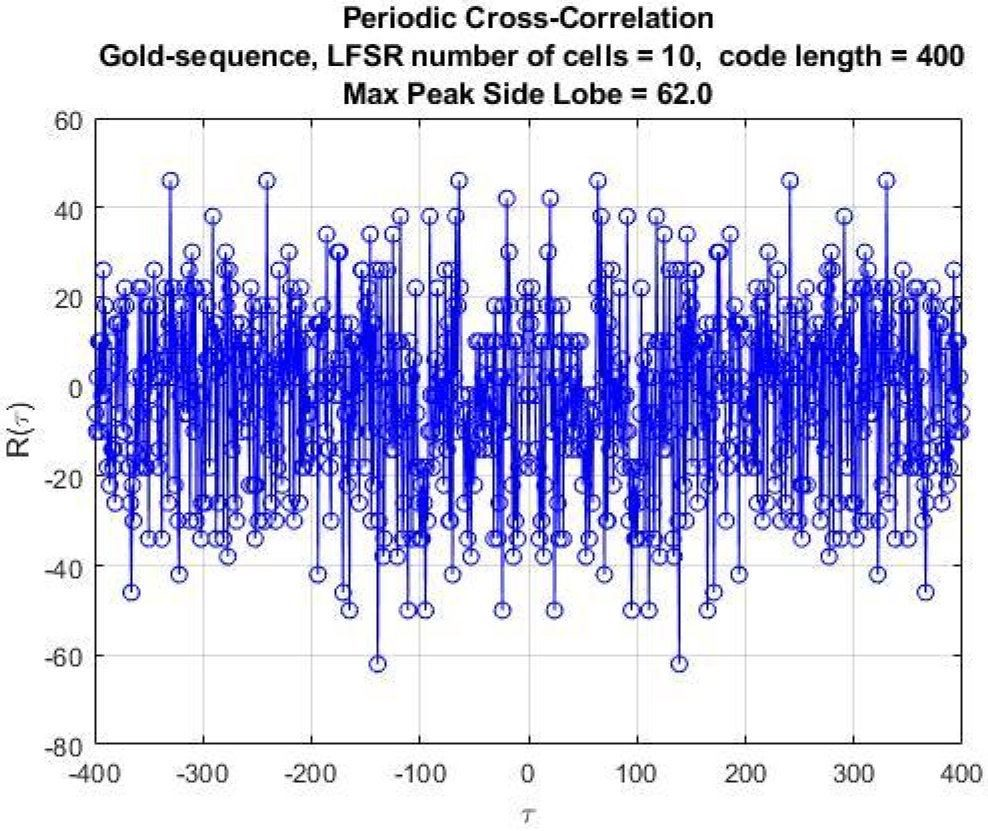


Figure 4.11



We can observe that in these particular cases after using different initial seeds, there are some variations of peak side lobe values with the change of initial seed values (even at a particular value of bit truncation).

The maximum peak side lobe for sequence length equal to 1000 bits, reduced by 10 units (from 104 to 94). Now we can say the performance improved by 1% (from 10.4% to 9.4% correlated bits along the length of the sequence). We fixed the length of the sequence in this case and just applied different initial seeds.

The MPSL value in figure 4.9 which is 94, is obtained after applying cross-correlation function between just two different gold sequences (indexes 4 and 10 selected randomly among all 1025 possible indexes) with the same initial seed value and same sequence length equal to 1000 bits each. We just fed the polynomials with different initial seeds.

The maximum peak side lobe for sequence length equal to 700 bits, reduced by 2 units (from 92 to 90). Now we can say the performance improved a little bit by 0.29% (from 13.14% to 12.85% correlated bits along the length of the sequence).

The MPSL value in figure 4.10 which is 90, is obtained after applying cross-correlation function between just two different gold sequences (indexes 4 and 10 selected randomly among all 1025 possible indexes) with the same initial seed value and same sequence length equal to 700 bits each.

The maximum peak side lobe for sequence length equal to 400 bits, reduced by 6 units (from 68 to 62). Now we can say the performance improved by 1.5% (from 17% to 15.5% correlated bits along the length of the sequence). It is the highest improvement among all these three cases.

The MPSL value in figure 4.11 which is 62, is obtained after applying cross-correlation function between just two different gold sequences (indexes 4 and 10 selected randomly among all 1025 possible indexes) with the same initial seed value and same sequence length equal to 400 bits each.

As a final remark for this part, we can say that for full-length gold sequences maximum peak side lobe values did not change after feeding our polynomials with different initial seeds. On the other hand, we have seen variations in maximum peak side lobe values for truncated gold sequences after feeding our polynomials with different initial seeds. In all three cases, we considered new initial seeds as follows: [0 0 0 0 0 1 1 0 0 0] and [0 0 0 0 1 1 1 0 0 0].

We studied truncated sequences because one of the most remarkable constraints in developing a new spreading sequence family is the length of the sequence that must be as short as possible since the new component shall allow a faster acquisition process. Unfortunately, we have seen that truncated gold sequences did not have good correlation properties and their maximum peak side lobe values exceeded the Welch's lower bound significantly compared to full-length gold sequences. So to compare families of sequences in the next chapter, full-length gold sequences will be the choice.

To see if truncated gold sequences are useful for CDMA, we analyzed the effect of truncation on correlation properties for cross-correlation function. We observed strong correlation properties at zero time lag in truncated sequence, regardless of its initial seed value and also the number of truncated bits [10]. This is a desirable aspect and also useful for tracking the sequence. We also observed peak side lobe variations by changing the initial seed value. However, this fact is not what we actually want but the strength of the peak side lobe is too weak to degrade the performance of the receiver. Another interesting aspect that allows us to use truncated sequences for CDMA is that there is no correlation in the truncated sequence with a different polynomial [10].

Chapter 5

Spreading Codes Evaluation

In chapter 2, we studied some popular families of spreading sequences. In order to evaluate the performance of spreading sequences, we introduced a weighted cost function based on the welch bound in chapter 3. The evaluation framework included three key metrics: performance in the acquisition stage, performance in the tracking stage, and the robustness against interfering narrowband signals. We can use this weighted cost function to evaluate and compare the sequence sets by taking into account five performance metrics as follows:

$AMEWSD^{MP}$, $AMEWSD^{CT}$,AMF^{MP},AMF^{CT}, AELW

$AMEWSD^{MP}$, $AMEWSD^{CT}$: are performance benchmarks in the acquisition stage.

AMF^{MP},AMF^{CT}: are performance benchmarks in the tracking stage.

AELW: is a benchmark for the robustness against interfering narrowband signals.

In this chapter, some popular families of spreading sequences are compared based on the above-mentioned metrics. Our primary solution for generating random sequences is very important because it affects the final solution performance. In chapter 4, we studied gold sequences and truncated gold sequences in detail. Our candidate as a primary solution is the gold sequence. It means that starting from gold sequences we generated random sequences according to the algorithm which was introduced in chapter 2. In every iteration of this algorithm, we flipped the bits within the spreading sequence then we evaluated the cost function according to equation (24). The bit flip was accepted if it minimized the cost function regarding the previous iteration otherwise was rejected. Also spreading sequences should be balanced with minimum auto-correlation side-lobe characteristics. We set these properties as initial necessities for the initial set of sequences but after some of the bits have been flipped, those characteristics could not be kept. We guaranteed those characteristics by imposing two conditions. The first condition guarantees that the sequence remains balanced and in order to satisfy it, we always flip the bits in pairs and the second condition is to minimize the auto-correlation function side-lobe and we satisfy it through equation (25).

5.1 Comparison Among Families of Spreading Sequences:

In this part, we compared three families of sequences under the benchmarks introduced in chapter 3. We chose sequence length equal to 1023 bits for each of them because the smallest length which is an entire divisor of the chip frequency 1.023 MHz is a 1023 chips sequence length and can be generated by a mathematical technique. Our code rate is 1.023 Mbit/s. All sequences are balanced and have minimum auto-correlation function side lobe.

These three families of sequences are as follows: Gold sequences, Random sequences with a random array as a primary solution, and Random sequences with Gold sequences as a primary solution. Obviously, in order to evaluate the performance, we needed to choose a subset of sequences. The size of the subset was set to 100 sequences for each of the above-mentioned families of sequences. Welch bound (φ min) is 31.824. This value was obtained through equation (26) where N=1023 and M=100. For gold sequences, we selected the most remarkable ones in terms of correlation properties according to (24). The cost function in equation (24) evaluates undesirable correlation peaks (those which enhance the error probability of the acquisition stage). Consequently, any correlation value surpassing the welch bound degrades our system performance. So the best sequences are the ones that minimize this cost function Fi.

For gold sequences, the best 100 sequences out of 1025 sequences were chosen. For random sequences families (random sequences with a random array as a primary solution and random sequences with gold sequences as a primary solution), we set a fixed number of iterations equal to 20000. This number is large enough for the convergence of the optimization process.

Another important issue that we had to take into account is the Doppler shift. When the Doppler shift is small, Excess welch Square Distance (EWSD) has a large value. If we consider EWSD of spreading sequences as a function of Doppler shift, the function is increasing at first then decreasing rapidly when the Doppler shift is varied from 0 to 1000 Hz. EWSD descends with the increase of Doppler shift and tends to be stable when the Doppler shift is more than 1000 Hz [2].

Obviously, we can analyze the behavior of EWSD much better when the number of frequency points is large enough but there is always a trade-off between increasing the sampling points and the complexity of our system. So we have to set the number of frequency points (nf_{offs}) properly. For instance, if the Doppler frequency shift is set from 0 to 6800 Hz with a sampling interval of 200 Hz, the number of frequency points (nf_{offs}) will be 35.

5.2 Simulation Results and Analysis

In this part, we used the weighted cost function to evaluate and compare the sequence sets by taking into account five performance metrics as follows:

$AMEWSD^{MP}$, $AMEWSD^{CT}$,AMF^{MP},AMF^{CT}, AELW

Note: The size of the subset was set to 100 sequences for each of the three families of sequences. To evaluate the impact over all sequence family, we needed to take an average among them.

Table 5.1 shows different values of these performance metrics for three families of sequences [1].

Table 5.1

Performance metric	Gold sequences	Random sequences with a random array as a primary solution	Random sequences with Gold sequences as a primary solution
$AMEWSD^{MP}$ (27),(28)	0.09550	0.0881	0.08840
$AMEWSD^{CT}$ (29),(30)	0.09629	0.09022	0.09035
AMF^{MP} (31),(32)	0.00096	0.00094	0.00099
AMF^{CT} (33),(34)	0.98985	0.98946	0.98944
AELW (35),(36)	19.5320	19.2296	19.2905

Table 5.1 shows that the parameters thus mentioned are not of the same order of magnitude. So to compare them fairly, we need to normalize them. This normalization can be the main reason for calculating the weighted cost function R_i according to equation (37) in chapter 3. However, without normalization and just by looking at the numbers in the table we can say that random sequences families (random sequences with a random array as a primary solution and random sequences with gold sequences as a primary solution) are very close to each other for all five performance metrics. It is also noticeable that random sequences families have better $AMEWSD^{MP}$, $AMEWSD^{CT}$ values (acquisition performance metrics) with respect to gold sequences family. The reason is that during the minimization process in random sequences families both $AMEWSD^{MP}$, $AMEWSD^{CT}$ were minimized.

Table 5.2 shows the weighted relative performance metrics in percent for three families of spreading sequences. In this table, the results of the cost function for each family of sequences are shown [1]. We can also observe the ranking place among the 3 families. We applied a weighting metric with a uniform weight vector, weight = [0.2,0.2,0.2,0.2,0.2]. When the vector of weights is [0.2,0.2,0.2,0.2,0.2] means that the degree of importance of all the above-mentioned performance metrics is equal. It is called a uniform weight vector.

Since the acquisition performance metric is more important, we can increase corresponding elements in the vector of weights and define a new vector as follows: [0.35,0.35,0.1,0.1,0.1]. The results for this weight vector is illustrated in table 5.3. The first two weights correspond to $AMEWSD^{MP}$, $AMEWSD^{CT}$ which are acquisition performance metrics and the other three weights correspond to AMF^{MP}, AMF^{CT}, AELW respectively. As we mentioned before, the sum of all weights in the vector should be equal to 1.

Table 5.2

Weighted relative metrics for 3 families of sequences with a uniform weight vector

	Gold sequences	Random sequences with a random array as a primary solution	Random sequences with Gold sequences as a primary solution
$AMEWSD^{MP}$ (%)	0.0565	-0.0258	-0.02198
$AMEWSD^{CT}$ (%)	0.0504	-0.0157	-0.01432
AMF^{MP} (%)	-0.0017	-0.0186	-0.03313
AMF^{CT} (%)	0.00015	-0.00024	-0.00026
AELW (%)	0.0081	-0.0074	-0.0043
Cost Function (R_i)	2.2721	-1.3599	-0.1551
Rank	3	1	2

After the cost function has been calculated for these 3 families, we could easily choose the best family of sequences among them. Obviously, the sequence with the smallest value of the cost function is the best. By looking at the rank row in table 5.2, random sequences with a random array as a primary solution have the smallest cost function value and as a result, they are considered as the best set of sequences among them.

Table 5.3 shows the weighted relative performance metrics in percent for three families of spreading sequences but with a weighting metric which is non-uniform weight vector, weight = [0.35,0.35,0.1,0.1,0.1]. In this case, the acquisition performance metric is more important and as a result, the families of sequences with better acquisition performance metrics improve their relative cost function value.

Table 5.3

Weighted relative metrics for 3 families of sequences with a non-uniform weight vector

	Gold sequences	Random sequences with a random array as a primary solution	Random sequences with Gold sequences as a primary solution
$AMEWSD^{MP}$ (%)	0.0565	-0.0258	-0.02198
$AMEWSD^{CT}$ (%)	0.0504	-0.0157	-0.01432
AMF^{MP} (%)	-0.0017	-0.0186	-0.03313
AMF^{CT} (%)	0.00015	-0.00024	-0.00026
AELW (%)	0.0081	-0.0074	-0.0043
Cost Function (R_i)	3.9141	-1.9081	-1.0004
Rank	3	1	2

From table 5.3, again random sequences with a random array as a primary solution have the smallest cost function value and as a result, they are considered as the best set of sequences among three families. Also, we can observe that for both random sequences with a random array as a primary solution and random sequences with gold sequences as a primary solution, relative cost function value improved significantly which means they have better performances on $AMEWSD^{MP}$, $AMEWSD^{CT}$ (acquisition performance metrics). At the same time, the cost function value increased for gold sequences (from 2.2721 to 3.9141) which means that gold sequences don't have better performances on acquisition performance metrics.

As a final result for this part, it can be proved that the performance of the random sequence family with a gold sequence as a primary solution is worse than the random sequence family with a random primary array. Random sequences with a random array as a primary solution have the smallest cost function value and as a result, they are considered as the best set of sequences among these three families. It may be noted that random sequence family with a gold sequence as a primary solution has better performance in tracking stage metrics concerning random sequence family with a random primary array. We also can expect that random sequence family with a random primary array is the best if we add other families of sequences like large kasami sequences or random sequence family with large kasami sequence as a primary solution. Another important issue is the number of iterations that we fix in the algorithm. It can be verified that when the number of iterations is large enough, a random sequence family with a random primary array stays in the first rank with a larger distance to other families of sequences. One of the reasons we want a spreading sequence that looks random is to ensure we have good correlation properties. Since the optimization process for random sequences families is done through the above-mentioned algorithm as well as the calculation of the cost function (24), we can minimize the performance metrics $AMEWSD^{MP}$, $AMEWSD^{CT}$, AMF^{MP}, AMF^{CT}, AELW, and weighted cost function to approach the Welch's lower bound as much as possible.

As a final remark in this chapter, we can say that random sequences families have better performance concerning other families of sequences and the reason is that the optimization process for them is done through the algorithm (Figure 2.9). Obviously, the larger the number of iterations, the better will be the final performance. We also have to consider that by increasing the number of iterations our complexity increases as well. There is always a trade-off between improving performance and increasing complexity.

Chapter 6

Final Remarks

6.1 Conclusion

After a brief introduction in the first chapter, we overviewed important definitions about periodic auto-correlation and cross-correlation as well as bounds: maximal out-of-phase auto-correlation magnitude, the maximal cross-correlation magnitude, maximal correlation magnitude, Welch bound, and Sidelnikov bound. The reason for this overview was that our selection benchmarks are based on different properties of spreading sequences such as auto-correlation, cross-correlation, side lobe values, and Welch bound.

In chapter 2, we studied some popular families of spreading sequences and the process of their generation. At the end of this chapter, we introduced the algorithm for random sequence generation with the final goal of optimizing the families of sequences. In every iteration of this algorithm, we flipped the bits within the spreading sequence then we evaluated the cost function according to equation (24). The bit flip was accepted if it minimized the cost function regarding the previous iteration otherwise was rejected. Also spreading sequences should be balanced with minimum auto-correlation side-lobe characteristics. We set these properties as initial necessities for the initial set of sequences but after some of the bits have been flipped, those characteristics could not be kept. We guaranteed those characteristics by imposing two conditions. The first condition guarantees that the sequence remains balanced and to satisfy it, we always flip the bits in pairs and the second condition is to minimize the auto-correlation function side-lobe and we satisfy it through equation (25).

In chapter 3, we briefly overviewed the acquisition and tracking stages in GNSS receivers. The reason for this overview was that first of all both of these stages rely on the estimation of correlation functions (auto-correlation and cross-correlation functions) and secondly, to select the best spreading sequence, we evaluate a weighted cost function according to a set of benchmarks, based on auto-correlation, cross-correlation, and power spectral density. These benchmarks show the effect of the non-ideal spreading sequences over the acquisition and tracking stages as well as over the robustness against narrow-band interferences. Then we introduced the weighted performance evaluation method for spreading sequences based on the Welch bound or in other words a weighted cost function based on the welch bound. This evaluation framework included three key metrics: performance in the acquisition stage, performance in the tracking stage, and the robustness against interfering narrowband signals. We could use this weighted cost function to evaluate and compare the sequence sets by taking into account five performance metrics as follows:

$AMEWSD^{MP}$, $AMEWSD^{CT}$, AMF^{MP}, AMF^{CT}, AELW

In chapter 4, we analyzed gold sequences periodic cross-correlation properties considering different sequence lengths. In each case, we reduced the length of the sequence by removing few bits from the end of the sequence (the length of the truncated sequence, is less than N = $2^n - 1$)

and computed the maximal cross-correlation absolute value between any possible pair of sequences. We verified for full-length gold sequences, all elements of the matrix of cross-correlation absolute values (except the main diagonal which were all zeros) are equal and can be computed according to (15), (16). The most important result was that whenever we truncated the full-length gold sequence for the last few bits, the final performance degraded as well. We also verified that the full-length gold sequence exceeded the Welch's lower bound less than its truncated versions which is better in terms of performance. As a result, for different cell sizes and different code lengths, we observed that peak side lobe values increased almost linearly with the number of truncated bits and at the same time we could observe more instability in plots whenever the number of truncated bits were increasing. In the worst-case analysis, we observed that the number of correlated bits concerning the length of the sequence increased significantly. Many sequences had been generated after we changed the truncation size (23,323,623) in the case of the full-length sequence. We also performed the maximum peak side lobe analysis with the change of initial seed values in gold Sequences. For full-length gold sequences, maximum peak side lobe values did not change after feeding our polynomials with different initial seeds. On the other hand, we observed variations in maximum peak side lobe values for truncated gold sequences. Truncated gold sequences did not have good correlation properties and their maximum peak side lobe values exceeded the Welch's lower bound significantly compared to full-length gold sequences. So to compare families of sequences in the final chapter, we chose the full-length gold sequence as a primary solution.

Finally, in chapter 5 we compared 3 families of sequences (Gold sequences, Random sequences with a random array as a primary solution, and Random sequences with Gold sequences as a primary solution) under the benchmarks introduced in chapter 3. The size of the subset was set to 100 and sequence length was set to 1023 bits. All sequences were balanced and had minimum auto-correlation function side lobe. Welch bound (φ min) was 31.824. This value was obtained through equation (26) in chapter 3, where N=1023 and M=100. To select the new spreading sequences family, we considered five selection performance metrics in chapter 3 based on the auto-correlation, cross-correlation, and Doppler frequency offset. Then we evaluated all those metrics based on a relative weighted cost function assessment. Gold sequence family was generated through mathematical models which we described in chapter 2 and random sequences families were generated based on the algorithm (Figure 2.9) and a cost function minimization process (24). We compared these three families' cost function values and we verified that random sequences with a random array as a primary solution have the smallest cost function value and as a result, they are considered as the best set of sequences among them. As a final result, random sequences families have much better performance in the acquisition stage concerning other families of sequences. However, there is no method that is hundred percent perfect and the performance evaluation method in this thesis project is not an exception and can be improved in future works.

Bibliography

[1] Espluga, Lorenzo Ortega, Poulliat, Charly, Boucheret, Marie-Laure, Aubault, Marion, Bitar, Hanaa Al, "New Solutions on the Design of a Galileo Acquisition-Aiding Signal to Improve the TTFF and the Sensitivity," *Proceedings of the 2018 International Technical Meeting of the Institute of Navigation*, Reston, Virginia, January 2018, pp. 810-825

[2] Dun Wang, Rui Xue2 and Yanbo Sun "A ranging code based on the improved Logistic map for future GNSS signals: code design and performance evaluation" Wang et al. EURASIP Journal on Wireless Communications and Networking (2017) 2017:57 DOI 10.1186/s13638-017-0840-4

[3] Fabio Dovis "Satellite Navigation System Course" Department of Electronics and Telecommunications, Polytechnic university of Turin, Academic Year 2017-2018

[4] https://it.mathworks.com/help/comm/ref/comm.goldsequence-system-object.html

[5] F. Soualle, M. Soellner, S. Wallner, J. Avila-Rodriguez, G. Hein, B. Barnes, T. Pratt, L. Ries, J. Winkel, C. Lemenager, and P. Erhard. "Spreading Code Selection Criteria for the future GNSS Galileo "

[6] D. V. Sarwate M. B. Pursley "Cross Correlation properties of Pseudorandom and related sequences" Proceedings of the IEEE vol. 68 no. 5 pp. 593-618 May 1980.

[7] James L. Massey and Thomas Mittelholzer "Welch's Bound and Sequence Sets for Code-Division Multiple-Access Systems " Signal and Information Processing Laboratory, Swiss Federal Institute of Technology CH-8092 Zürich

[8] Paonni, M., Bavaro, M., "On the Design of a GNSS Acquisition Aiding Signal," Proceedings of the 26th International Technical Meeting of the Satellite Division of the Institute of Navigation (ION GNSS+ 2013), Nashville, TN, September 2013, pp. 1445-1456.

[9] Marco Baldi, Franco Chiaraluce, Noureddine Boujnah, Roberto Garello, "On the Autocorrelation Properties o\f Truncated MaximumLength Sequences and Their Effect on the Power Spectrum" lEEE Trans Signal Processing, vol. 58, no. 12, December 2010.

[10] P. Banerjee, Geetanjali Raj, Kanupriya Attrey and Monica Kaushik, "Usefulness of Truncation of Full Length Pseudo Random Sequence for CDMA Communication" February 2015, Conference: SPIN 2015 "2nd International Conference on Signal Processing and Integrated Networks,At: Ämity University, Noida

[11] Jon Olafur Winkel, US 2008/0246655 A1, "Spreading Codes for a Satellite Navigation System".

[12] Oppenheim, Alan V, Alan S. Willsky, and Ian T. Young. Signals and Systems. Englewood Cliffs, N.J: Prentice-Hall, 1983.

[13] Elliott D. Kaplan. Understanding GPS: Principles and Applications. Boston: Artech House, 1996.

[14] Irsigler, Markus, Avila-Rodriguez, Jose Angel, Hein, Guenter W., "Criteria for GNSS Multipath Performance Assessment," Proceedings of the 18th International Technical Meeting of the Satellite Division of the Institute of Navigation (ION GNSS 2005), Long Beach, CA, September 2005, pp. 2166-2177.

[15] S. Wallner and J.-A. Ávila-Rodríguez. "Codes: The PRN Family Grows Again" Inside GNSS, Volume 1, September, 2011

[16] M. B. Pursley H. F. A. Roefs "Numerical evaluation of correlation parameters for optimal phases of binary shift register sequences" IEEE Trans. Commun vol. COM-27 no. 10 Oct. 1979.

[17] Dovis F., Mulassano P., Dominici F., "Overview of Global Navigation Satellite Systems," pp. 923-974

[18] "Spread Spectrum Systems for GNSS and Wireless Communications", Chapter 2, Jack K. Holmes

[19] " Error Correcting Codes" W. Wesley Peterson, MIT Press (1970) [16] Betz, John W., "Effect of Narrowband Interference on GPS Code Tracking Accuracy," Proceedings of the 2000 National Technical Meeting of the Institute of Navigation, Anaheim, CA, January 2000, pp. 16-27.