UNIVERSITÀ
DEGLI STUDI
DI BRESCIA

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

Corso di Laurea in
Ingegneria Informatica

Relazione Finale

Analysis and Characterization of Wi-Fi Channel State Information

Relatori: Prof. Renato Lo Cigno
Dott. Lorenzo Ghiro

Laureanda:
Elena Tonini
Matricola n. 727382

Anno Accademico 2021/2022

Sommario

Questa tesi è una relazione sull'analisi di Wi-Fi Channel State Information (CSI) — ossia Informazioni sullo Stato di un Canale Wi-Fi — che mira a caratterizzare il canale *wireless* e fornirne un modello matematico, lavorando sulla scala temporale della trasmissione di trame di traffico e non dei singoli simboli che compongono la trasmissione delle trame.

Le CSI permettono di derivare una descrizione ad alto livello del comportamento di un segnale che si propaga da un trasmettitore a un ricevitore. Esse possono essere utilizzate per effettuare *ambient sensing*, una tecnica che permette di estrarre da un canale *wireless* alcune informazioni relative all'ambiente in cui si propaga il segnale. L'*ambient sensing* ricopre già un ruolo essenziale nello sviluppo di nuove reti *wireless*: ad esempio, le reti 5G New Radio e Beyond 5G lo impiegano quando effettuano *Joint Communication and Sensing* — letteralmente: Comunicazione e Rilevamento Congiunti — e quando ottimizzano la propagazione del segnale attraverso *beamforming*.

Nonostante l'uso diffuso dell'*ambient sensing*, vi è ancora una mancanza di ricerca relativa alla caratterizzazione dei canali *wireless* quando si tratta di utilizzarli per scopi diversi dalla sola comunicazione, quindi su una scala temporale più lunga della trasmissione di una sequenza di simboli.

Questo lavoro analizza la struttura delle tracce CSI raccolte in un laboratorio e ne descrive il comportamento così da fornire un'interpretazione matematica che possa essere utilizzata per modellizzare l'impatto delle variazioni ambientali sul segnale.

Il modello ottenuto può contribuire a migliorare gli aspetti tecnici dell'implementazione di *Joint Communication and Sensing* nelle nuove reti, in modo che sia possibile sfruttare efficacemente le proprietà dei canali per raggiungere gli obiettivi che le tecnologie future fisseranno.

Summary

This thesis is a report on the analysis of Wi-Fi Channel State Information (CSI) aiming at characterizing the wireless channel and providing a mathematical model for it working at the timescale of frame transmission.

Channel State Information allows a high-level description of the behaviour of a signal propagating from a transmitter to a receiver. It can be used to perform *ambient sensing*, a technique that extracts relevant information about the surroundings from the signals received. Ambient sensing can already play an essential role in the development of new wireless networks: for instance, 5G New Radio and Beyond 5G networks use it when performing *Joint Communication and Sensing* and when optimizing signal propagation through *beamforming*.

Despite the widespread use of ambient sensing, there is still a lack of research about the characterization of wireless channels when it comes to using them for purposes other than communication alone, especially focusing on a timescale longer than the typical signal and symbol analysis.

This work analyzes the structure of CSI traces collected in a laboratory and describes their behaviour so as to provide a mathematical interpretation for it that can be used to model the channel at the frame level.

The obtained model can help improve the technical aspects of the implementation of Joint Communication and Sensing in future networks so that it will be possible to efficiently take advantage of the properties of the channels to fulfil the goals that upcoming technologies will set.

Contents

1 Introduction

With the development and evolution of 5G networks and the ongoing research dedicated to future generations of wireless networks, the importance of guaranteeing both adequate coverage and the high bit rates promised to the users has become more and more relevant.

To improve the performance of Fifth (and later) Generation networks, researchers have intensified their focus on what is known as Channel State Information or CSI.

CSI is the information that allows the description of the behaviour of a signal propagating from a transmitter to a receiver. Newly-developed technologies benefit from its use when it comes to channel equalization and the implementation of Multiple Input Multiple Output (MIMO) techniques.

Moreover, CSI can be used to perform *ambient sensing*, a technique that extracts relevant information about the surroundings and the environment where the signal propagation occurs. The information contained in the CSI depends on the behaviour of the signal because it varies depending on how the signal itself is reflected, scattered, and absorbed by the objects in the environment.

Through the analysis of the content of a CSI trace, we can reconstruct the static structure of the environment (i.e., the precise location of furniture and other appliances) as well as its dynamic entities (e.g., location and movements of the people in the environment). It is also possible to locate the devices situated in the environment.

Knowing that all surfaces and objects somehow interact with the signal

propagation by reflecting, scattering, and absorbing it, thus altering the content of the collected CSI, it is no surprise that it is possible to perform sensing to locate people in the environment even though they do not carry communication devices. To do so, only a fixed transmitter is needed, besides clearly the sensing receiver that analyzes the CSI. The transmitter and the sensing receiver must be fixed, because otherwise their movement would change the CSI, hiding the modifications induced by the ambient. Because the human body itself acts as an obstacle to signal propagation at high frequencies, whenever there is a person in the area covered by propagation, the content of the CSI gets modified according to the physical interaction between the signal and that person's body. This phenomenon gives an observer the chance to obtain relevant information on where people are in the room based solely on the properties that the signal displays at the receiver.

Although the final goal is to be able to perform ambient sensing anywhere and with any new wireless technology, there have not yet been relevant studies on CSI either taking place outdoors or using wireless communication technologies other than Wi-Fi (such as Long Term Evolution or 5G). Nonetheless, there is no reason why performing ambient sensing using Channel State Information should not be possible in outdoor environments or with technologies different from Wi-Fi. In fact, 5G New Radio and Beyond 5G networks use ambient sensing when performing what is known as *Joint Communication and Sensing* (JCAS), which is a staple for new services.

Right now, capture and analysis of CSI can already find multiple useful applications and actually play an essential role in the technological development of wireless networks.

The ability to perform ambient sensing alongside communication is an interesting property for all mobile communication networks using high frequencies (such as 5G, whose lowest frequency range starts as low as 600 MHz).

Whereas up to 5-6 GHz there are no difficulties linked to the use of omnidirectional antennae and Line of Sight (LoS) between antenna and receiver is not strictly required, it is impossible to have efficient transmitters at much higher frequencies without applying *beamforming*.

Beamforming is a technique that makes it possible to increase the directionality of the transmitter radiation pattern to cover a specific geographical area where the targeted receiver is. At high frequencies (20-30 GHz and higher) Non-Line of Sight (NLoS) communications are impossible to obtain, meaning that beamforming is required to compensate for the signal attenuation introduced by such frequencies. Without implementing beamforming, under these conditions, it would be both expensive and technologically challenging to provide network coverage to the whole area, as we would need to guarantee LoS between any antenna and any receiver. Instead, thanks to ambient sensing, it is possible to identify obstacles that would obstruct signal propagation, thus facilitating beam steering or allowing to move the transmission to a better-placed transmitter, avoiding a loss in the quality of service.

The two functionalities involved in JCAS are combined to provide support in multiple contexts other than wireless communication. For instance, JCAS can find many applications in autonomous vehicle networks - where sensing of the environment plays a key role in identifying obstacles and safely modifying the trajectory of vehicles -, robot movements management, motion-based device (de)activation in smart homes, and sensor-based in-home health monitoring. This leads to the possibility of employing JCAS in Beyond 5G networks to create holographic and multi-sense communications, connectivity for all things, and time-sensitive applications requiring high data rates [1]–[3].

Considering the benefits brought by the possibility of performing both sensing and communication simultaneously, studies are being conducted aimed at designing JCAS systems that can support both features and are able to perform

them using shared frequency band and hardware devices — thus improving spectrum efficiency and reducing hardware cost [4], [5].

As happens with all technological innovations, the risk of their exploitation for malicious purposes has to be taken into consideration when studying the applications of CSI analysis; however, by expanding the studies on the extraction of information about the environment through a Wi-Fi connection, we can develop new ways of securing the users' privacy.

The exploitation of Wi-Fi communications to extract information about the environment can lead to the users being in a vulnerable position, given that they cannot defend themselves from possible attacks that would violate their privacy, such as unauthorized sensing of their location or specific movements. This is why research has been and is currently being conducted on the development of strategies that can be used to prevent such violations of the users' privacy. Some strategies that have been studied include jamming signals to make it hard for a malicious user to obtain relevant information from CSI or obfuscating the sensitive information carried by it [6].

To this day, sensing is efficiently achieved only through Artificial Intelligence and Machine Learning techniques, but there is a lack of research about the characterization of wireless channels when it comes to using them for purposes other than communication itself: thanks to an effective channel characterization, it would be possible to obtain valuable and interesting information from sensed data and elaborate it to make JCAS more efficient. Moreover, it would be possible to provide new insights that could improve the efficiency of mechanisms aimed at enhancing users' privacy.

This thesis's goal is to analyse the behaviour of a Wi-Fi channel used for sensing and define its mathematical and, most importantly, statistical characterization.

2 Motivation and Goals

As seen in Chapter 1, whereas communication — as one can expect — has always been the principal goal of telecommunication networks, the possibility of performing sensing in parallel to it has only more recently been discovered in its full potential, so much so that ambient sensing is already seen as a fundamental feature for 5G New Radio and Beyond 5G wireless networks.

Alongside the rapid evolution of new wireless networks, new privacy issues linked to their possible weaknesses arise. Researchers aim at studying the functionalities of the networks and, at the same time, help prevent future innovative ways to attack them by creating appropriate means and tools that could improve protection.

This thesis focuses on deepening our understanding of how ambient sensing works in Wi-Fi networks, paving the road to future research aimed at enhancing the features of such networks and encouraging their implementation in upcoming technologies. Simultaneously, we would like to help prevent these technologies from falling victim to newly-developed attacks that could violate ethical and legal boundaries.

Surprisingly enough, to this day, no relevant study has been performed yet on the structure and behaviour of a Wi-Fi channel when it is used for sensing; there is an almost complete absence of scientific literature and documentation about such research. All experiments and implementations of localization systems are currently performed through Artificial Intelligence and Machine Learning techniques; very little interest has been shown regarding the structure of the underlying problem that convolutional networks are asked to solve.

As effective as they are, current applications of JCAS do not take into consideration the characteristics of the wireless channel when it is used for sensing. Having a mathematical and, most importantly, statistical description of how the channel behaves could simplify the research for new ways to protect users' privacy as well as provide meaningful insights on the specific aspects of how new high-performance networks work.

Through the analysis of Wi-Fi CSI, we expect to identify patterns or regularities in trace behaviour. Our goal is to describe the channel as it is used while performing sensing so that, based on the outcome derived from sampled data, we can provide a reusable, mathematically simpler model; essentially, we want to identify a known statistical distribution whose behaviour corresponds to that of the channel.

This model could be seen as a benchmark for future developments: it could then be used for different purposes, going from its concrete employment in applications used to detect the presence of a person within an environment — which would be the same as performing ambient sensing backed up by a mathematical model — to the generation of artificial traces comparable to those empirically collected. The ability to artificially engineer traces opens the possibility for experimentation and testing of how the parameters of the distribution that describes the channel influence the transmission itself, allowing researchers to alter such parameters and immediately see the consequences without needing to test their physical implementation.

These considerations lead us to believe that conducting this study could significantly contribute to research in this field and provide valuable results that could be employed to ensure the validity of the mathematical or statistical models underlying future experiments and new technologies implementations.

3 Wi-Fi fundamentals

Knowing that this thesis focuses on the analysis of Wi-Fi CSI, it is appropriate to provide a summary of its technical aspects before delving into our analysis.

Wi-Fi is now renowned as one of the most widespread means of wireless connection; as a matter of fact, it cannot properly be considered a technology *per se* because it is in fact a trademark applied to different devices by their manufacturers to guarantee interoperability. Regardless, Wi-Fi is tied to the IEEE 802.11 standard inseparably, as the IEEE standard serves as a foundation for wireless network devices associated with the Wi-Fi brand.

IEEE 802.11 is the standard for wireless LANs (Local Area Networks) or WLANs. It belongs to the same family of standards as Ethernet (IEEE 802.3) and relies upon a set of Medium Access Control (MAC) and Physical Layer (PHY) specifications that describe the implementation of WLANs. Different modulation techniques are implemented in the various versions of the 802.11 standard, although they are all derived from the same basic protocol. The protocol version used to sample the datasets in our study is 802.11a/g. Still, an 802.11ax trace with an 80 MHz channel width was used too to explore if CSI characterization is version-dependent or can be generalized.

3.1 802.11a/g Packet Format

A generic 802.11a/g packet is made of two main sections: a *preamble* and a *payload*. The preamble contains information that allows time and frequency

synchronization and channel estimation, together with some data regarding payload length and rate of the transmission. Essentially, the preamble contains information describing how to decode the content of the payload [7].

The following diagram features an outline of the packet structure.

The preamble consists of the first three fields:

- `STF` (`Short Training Field`): contains initial information that allows synchronization and frequency tuning
- `LTF` (`Long Training Field`): contains more precise information allowing synchronization, frequency tuning, and channel response estimation
- `SIGNAL`: 24 bits of configuration data needed to correctly interpret the payload content. They are divided in:
  - `RATE`: 4 bits used for *forward error correction* (or *channel coding*, a technique used in case of an unreliable or disturbed channel to correct errors in the received data without the need for re-transmission) and modulation
  - `LENGTH`: 12 bits that specify the length of the payload in bytes
  - `PARITY`: 1 parity bit calculated on `RATE` and `LENGTH`
  - `TAIL`: 7 bits used for `SIGNAL` symbol *forward error correction* decoding

The field named `Data` is the actual packet payload, corresponding to the user's data that we want to send over the channel.

As one can expect, with the different versions of the 802.11 standard comes a variety of Physical Layer Modulation techniques, whose characteristics determine how the user's data get sent over the channels. The more recent the version of the protocol, the higher data rates the chosen modulation technique allows.

Standard 802.11a/g uses OFDM (Orthogonal Frequency-Division Multiplexing) modulation technique.

3.2 Modulation techniques and OFDM

Modulation is a procedure that allows the mapping of information on a physical dimension.

The easiest way to perform modulation is *amplitude modulation*, which consists of mapping the information on the amplitude of the chosen dimension (e.g., mapping binary values onto voltage values).

If we increase the number of physical dimensions to map information onto, we can change the logic behind modulation and switch to a *phase modulation*. The name suggests that it was introduced based on a complex electromagnetic unit; it represents information using the phase of the exponential of the complex value. Phase modulation is obtainable by combining two non-interfering dimensions: such dimensions are orthogonal, meaning that their scalar product equals zero, which allows their representation on a Cartesian plane defined as the signal space.

Mapping onto more than two linearly independent dimensions is possible: in this case, the representation of the values depends on phase and amplitude on both Cartesian axes; this modulation technique is called Quadrature Amplitude Modulation (QAM).

OFDM is a multi-carrier modulation and multiplexing system that trans-

mits a data stream as multiple orthogonal narrowband signals named *sub-carriers* [8], each of them using QPSK (Quadrature Phase Shift Keying — i.e., Phase Modulation) or QAM modulation.

Two main advantages come from using OFDM:

- In case of a disturbed channel, interference, noise or fading phenomena only affect a portion of the sub-carriers without impairing the whole communication process
- It gives the possibility to reduce used bandwidth by partially overlapping adjacent sub-carriers.

To avoid interference from adjacent sub-carriers without adding a guard band between them, it is a fundamental requirement that the sub-carriers are mathematically orthogonal. If the symbol period is T, the sub-carriers are linearly independent if spaced by a multiple of $1/T$. If the sub-carriers are spatially distributed this way, we obtain that their combination shows sub-carrier nulls in correspondence to peaks of adjacent sub-carriers. This phenomenon only occurs if the sub-carriers are orthogonal and implies the removal of inter-carrier interference. This phenomenon is visually described in Figure 3.1.

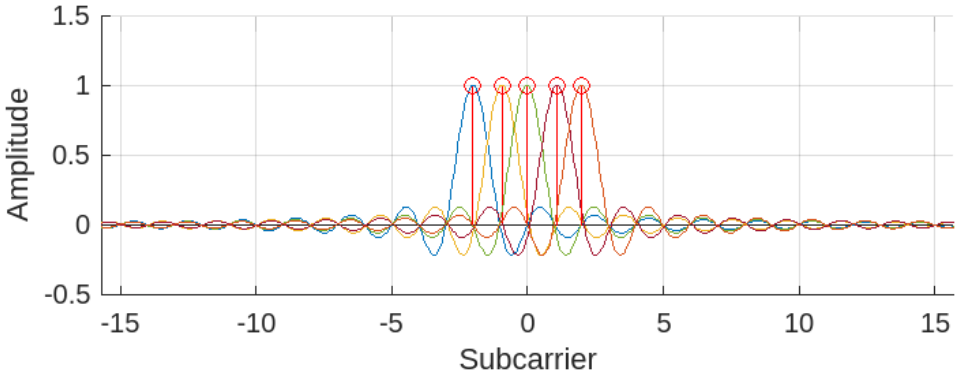

Figure 3.1: Visual representation of sub-carriers orthogonality

3.3 802.11ax Standard Version

Version 802.11ax is a version of IEEE 802.11 whose implementation grants the users a higher throughput compared to previous versions of the same standard [9].

The main advantage brought by this standard is its usability in outdoor and indoor environments characterized by a large number of users, such as stadiums, airports, train stations, etc. This feature depends on the introduction of Uplink Multi-User MIMO and OFDMA (Orthogonal Frequency-Division Multiple Access), a channel access technique that can be interpreted as an evolution of OFDM.

Despite the introduction of the Uplink Multi-User operation mode, 802.11ax still maintains the legacy Single-User mode active. When operating in Multi-User mode, multiplexing can be performed in two different ways:

- MU-MIMO: when used in Downlink, beamforming techniques are employed to direct packets simultaneously to users in different locations, which is why this technique is also known as *spatial multiplexing*. When used in Uplink, spatially separate users request channel access by sending a *trigger frame*; this is followed by a response from the Access Point, after which users can finally send their packets over their channel.

- OFDMA: by extending the way OFDM works, OFDMA allocates Resource Units (RUs) to the users for a limited time. Essentially, it divides the channel into sub-channels, each having its assigned sub-carriers, allowing the users to send their packets on a specific sub-channel for a designated amount of time. The minimum dimension of a sub-channel (i.e., the single RU) that can be used for transmission is a set of 26 sub-carriers. According to the specific needs, the whole channel can be either

allocated in its entirety to a single user or partitioned into sub-channels whose RUs are allocated separately.

The structure of a generic 802.11ax frame used in Single User mode respects the following model [10]:

Legacy preamble	RL-SIG	HE-SIG	HE-STF	HE-LTF	HE-Data	Packet Extension

The content of the `Legacy preamble` is there to guarantee backwards compatibility with previous versions of the protocol. The structure of this preamble is the same as that of the 802.11a/g packet preamble.

`RL-SIG` (`Repeated Legacy Signal`) field is used to repeat the content of the `SIGNAL` field of the `Legacy preamble`.

The rest of the preamble consists of fields whose names start with `HE` (`High Efficiency`): these fields can only be decoded by 802.11ax devices; their content is equivalent to that of the fields listed in Section 3.1, only in a different order. The `HE-Data` field contains the actual user's data and is followed by a `Packet Extension` field.

When used in Multi-User mode, the packet takes on the following structure, which is only slightly different from the Single-User mode packet structure:

Legacy preamble	RL-SIG	HE-SIG-A	HE-SIG-B	HE-STF	HE-LTF	HE-Data	Packet Extension

In this case, `HE-SIG-B` is introduced to the sole scope of managing Multi-User communication, whereas the remaining fields stay the same as in Single-User mode.

3.4 Wi-Fi Channel Frequency and Bandwidth

Together with the modulation technique, it is necessary to specify the channel frequency and bandwidth used in standards 802.11a/g and 802.11ax.

Most versions of the 802.11 standard utilize the $2.4 - 2.5$ GHz spectrum or the $4.915 - 5.825$ GHz band, which are often more simply referred to as the 2.4 GHz and the 5 GHz frequency bands. Both spectra are divided into multiple channels characterized by a centre frequency. The number of channels significantly differs depending on the selected spectrum.

When the 802.11 standard was first released, all channels had 20 MHz bandwidth; later, 40 MHz and 80 MHz bandwidths were introduced. Specifically, the 80 MHz bandwidth can be obtained by juxtaposing two adjacent non-overlapping 40 MHz channels.

The 2.4 GHz frequency band is subdivided into 14 channels, whereas the 5 GHz band has a more complicated partition because regulations vary from one Nation to another, causing the channels not to be utilizable the same way in all Countries.

Both standards we employ in our study can work on either the 2.4 GHz or the 5 GHz frequency band.

When using 802.11a/g, we utilize channel 157 at 5 GHz with 20 MHz bandwidth, whose centre frequency is 5785 MHz; channel 157, in Europe, can only be used with a 20 or 40 MHz bandwidth for Short Range Devices (SRD), whose allowed highest power level is 25 mW [7].

In this thesis, we will write 802.11a instead of 802.11a/g, given that this is the protocol that describes the 5 GHz spectrum, which is the only band we use empirically. Contrarily, 802.11g is dedicated to characterising the 2.4 GHz band.

3.5 CSI Structure

In Wi-Fi systems using OFDM modulation, every sub-carrier Channel State Information can be mathematically represented by a complex number using

the following mathematical formula [11]:

$$H(k) = ||H(k)||e^{j\angle H(k)} \tag{3.1}$$

In this formula, $H(k)$ is a CSI of the k-th sub-carrier, $||H(k)||$ corresponds to its amplitude and $\angle H(k)$ to its phase.

As one can imagine, Formula 3.1 is equivalent to:

$$H(k) = ||H(k)||(\cos(\angle H(k)) + j\sin(\angle H(k))) \tag{3.2}$$

Depending on the activity performed in the environment where the CSI is captured, $||H(k)||$ and $\angle H(k)$ take different values: based on their variations, the graphs showing CSI amplitude and phase will present a unique pattern that depends on the detected activity.

An example of this variation can be seen by comparing Figure 3.2 and Figure 3.3. The two figures represent the amplitude of CSI traces collected on sub-carriers on channel 157 at different times during the day with a person standing in two different spots within the same environment.

In Figures 3.2 and 3.3 we consider the different sub-carriers (i.e., the different frequencies) on the x axis and the packets sent over time on the y axis. Sub-carrier and packet numbers are indicated on the axes. As shown by the legend on the right side of the graphs, different colours indicate different amplitudes: a lighter colour (i.e., closer to lime green) indicates a lower value of the amplitude, whereas a darker colour (i.e., closer to blue) indicates a higher value.

Remarkably, the amplitude of both CSI traces shows an overall somewhat constant behaviour when observed in its evolution in time, but it notably changes if observed on the frequencies. Moreover, it can be noticed that the amplitude significantly varies when the person moves from one spot to another.

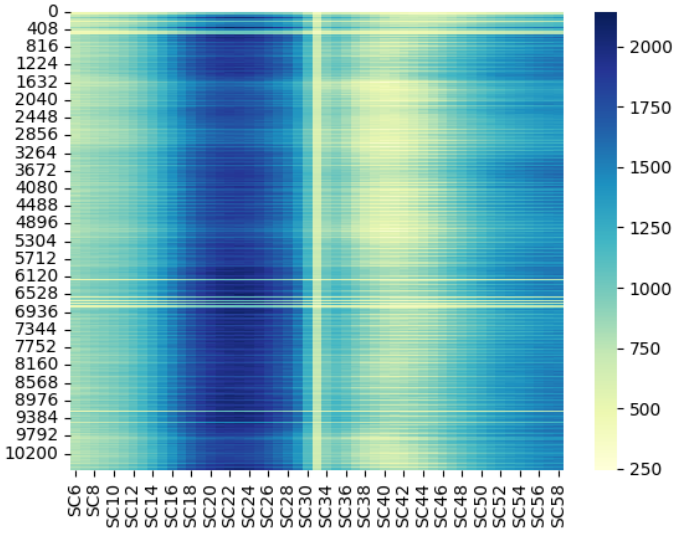

Figure 3.2: CSI amplitude relative to a trace collected with a person standing still within the environment

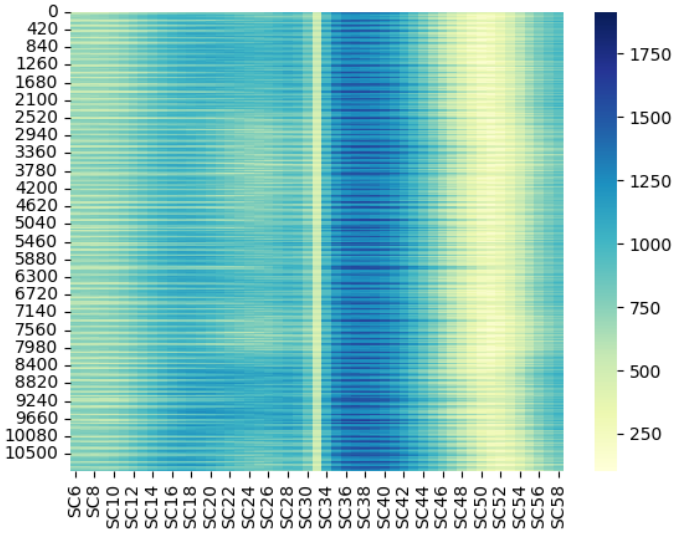

Figure 3.3: CSI amplitude relative to a trace collected with a person standing still in a different spot than Figure 3.2 within the environment

4 CSI Traces Collection

The CSI traces that we work on in this study have been collected in the indoor laboratory within the Department of Information Engineering at the University of Brescia. All those collected on the 802.11a channel were sampled on February 24th, 2022, whereas the 802.11ax trace was sampled on July 14th, 2022.

Most CSI traces are extracted from OFDM-modulated Wi-Fi frames transmitted over a channel regulated by the 802.11a protocol. The used channel is channel 157 within the 5 GHz frequency band with 20 MHz bandwidth. A CSI trace is also collected on an 80 MHz bandwidth channel regulated by 802.11ax protocol.

The traces have been extracted using Nexmon Channel State Information Extractor [12], [13]. All frames in the traces collected on channel 157 consist of 316 bytes and are sampled over nearly 18 seconds, depending on the number of frames in each trace. Instead, the trace collected on the 80 MHz bandwidth channel is much shorter and consists of only 1,825 packets.

The analyzed traffic is generated by a board communicating with a receiving device: Figure 4.1 displays a map of the laboratory settings where all device locations are indicated.

Let us consider the following landmarks to facilitate location identification of the used devices:

- Corner 0 (c0): lab corner adjacent to the door
- Corner 1 (c1): lab corner straight ahead of the door

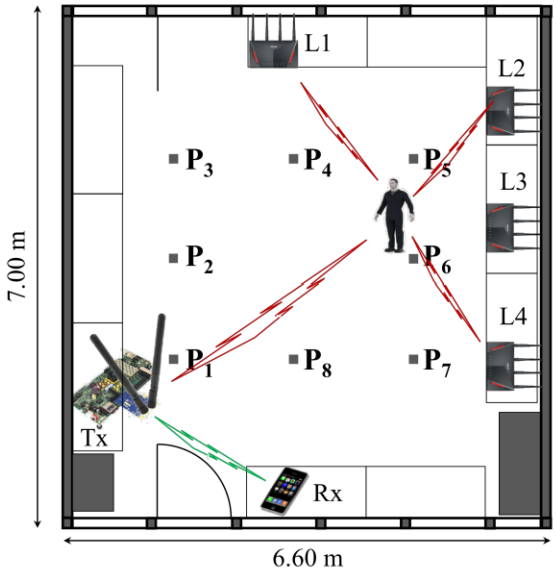

Figure 4.1: Map of the laboratory — Image taken from [14] with permission from the authors

- Corner 2 (c2): lab corner diagonally opposite Corner 0
- Corner 3 (c3): lab corner diagonally opposite Corner 1

The devices to generate, receive, and collect traffic are thus placed:

- Board (Transmitter): two metres away from Corner 0 along wall c0 - c1
- Receiver: halfway along wall c0 - c3, about three metres from the board
- Three Collectors: evenly distributed along wall c2 - c3
- Fourth Collector: two metres away from c1 along wall c1 - c2

The traces have been collected with a person standing still in eight different spots within the laboratory, all of which can be identified in Figure 4.1 as they are indicated by the letter `P` followed by a number from 1 to 8.

The data have been collected by Professor Francesco Gringoli, Dr Lorenzo Ghiro, and Dr Marco Cominelli.

5 Traces Analysis

Our examination of CSI traces is performed separately trace by trace, to monitor their behaviour and detect possible inconsistencies or mistakes as we go.

As we have already seen using Equation 3.1, a CSI is nothing but a complex number, therefore it can be described by amplitude and phase. The focus of our analysis of CSI traces is strictly on the behaviour of trace amplitude values; future research may lead to more in-depth studies on the evolution and behaviour of CSI phases too, but this ulterior analysis currently goes beyond the scope of this thesis. To this day, existing scientific literature has always been centred on amplitude analysis because it is the most widespread means of characterization of the channel and the environment, as it carries the most relevant pieces of information that allow location identification within an environment [15]–[18].

Despite their predictable and undeniable importance, phase values of CSI traces have been left out of our analysis because they have not yet been used in meaningful studies to characterize the channel and extract features from the surroundings.

It should be noted that all plotting and data processing has been performed on the datasets obtained from capturing CSI traces on the 80 MHz bandwidth 802.11ax channel and the 20 MHz bandwidth 802.11a channel.

The structure of the code that was used to carry out this analysis is summarized in Appendix A, where we also provide brief code documentation.

Of course, it would not be feasible to show all the graphs that have been plotted during this study. Nonetheless, it should be noted that, every time

graphs were produced, they were generated for all sub-carriers. The graphs presented in this thesis display the index of the corresponding sub-carrier (SC) in their title.

5.1 Amplitude Evolution In Time

From every CSI trace, we extract the values of the amplitude of the packets sent on each sub-carrier (256 sub-carriers for the data collected on the 80 MHz bandwidth channel, 64 sub-carriers for those collected on the 20 MHz one). Some sub-carriers have been discarded as they are not used for modulation; trying to fit a statistical distribution on data that do not derive from transmitted information would produce unreliable and incorrect results.

The initial objective is to verify whether there is any correlation in time between the amplitudes of packets sent on the same sub-carrier. As the first step in our analysis, we plot graphs showing the amplitude evolution in time: on the x axis we use time as the unit of measurement, whereas on the y axis we present the amplitude variations.

We consider this process as having discrete time values because each packet is separated from the previous and the following one by a time interval. Considering that the frames we work on have been collected in bursts, the distance between consecutive packets belonging to the same burst is very short (about 1 ms). For a perfectly precise evaluation of the behaviour of the channel, we would have to consider the time of arrival of each received packet and examine how the system behaves between consecutive packets while no data transmission takes place.

Some examples of the resulting graphs can be seen in Figures 5.1, 5.2, and 5.3.

The fluctuations displayed in the graphs in Figures 5.1, 5.2, and 5.3 may

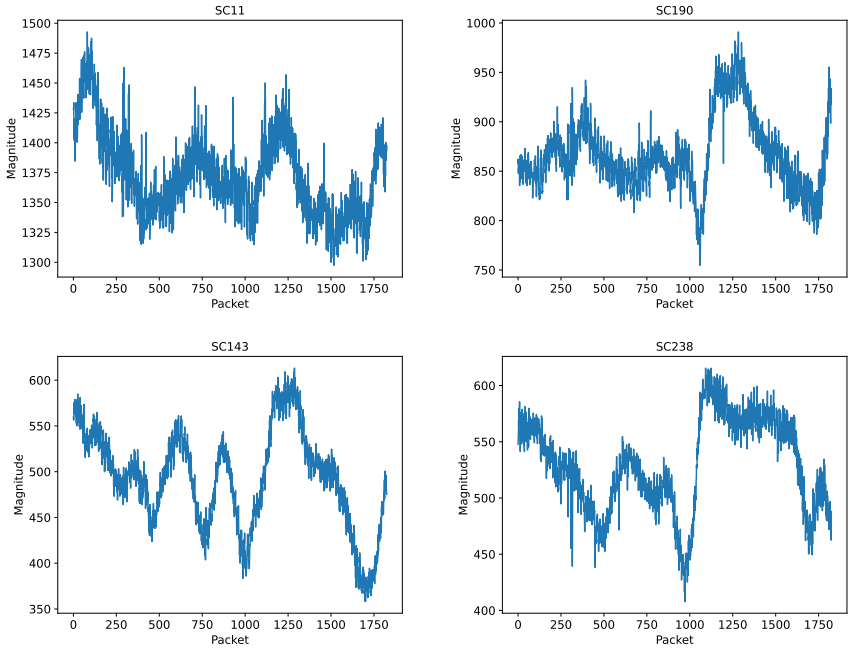

Figure 5.1: Time evolution of packets amplitude on an 80 MHz bandwidth 802.11ax channel

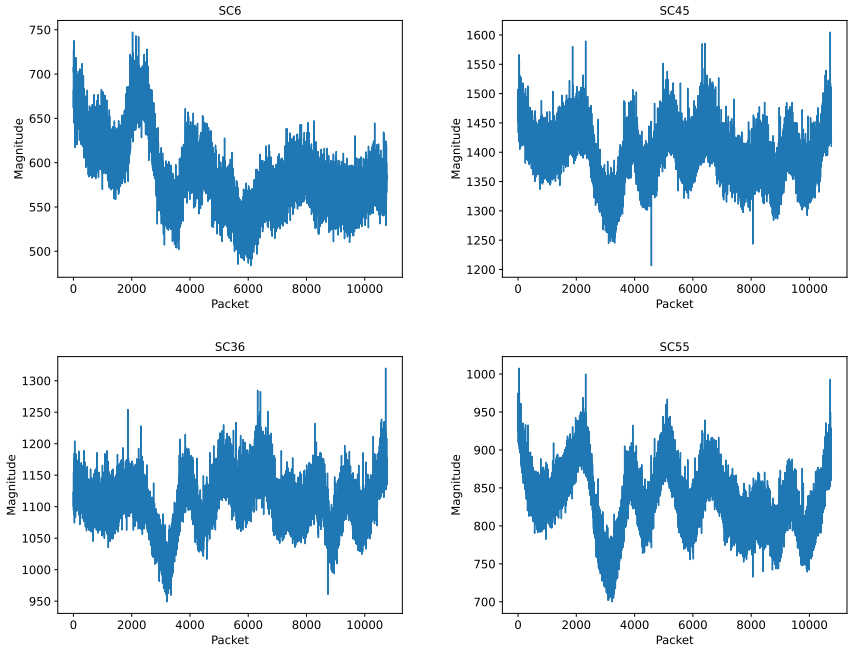

Figure 5.2: Time evolution of packets amplitude on a 20 MHz bandwidth 802.11a channel

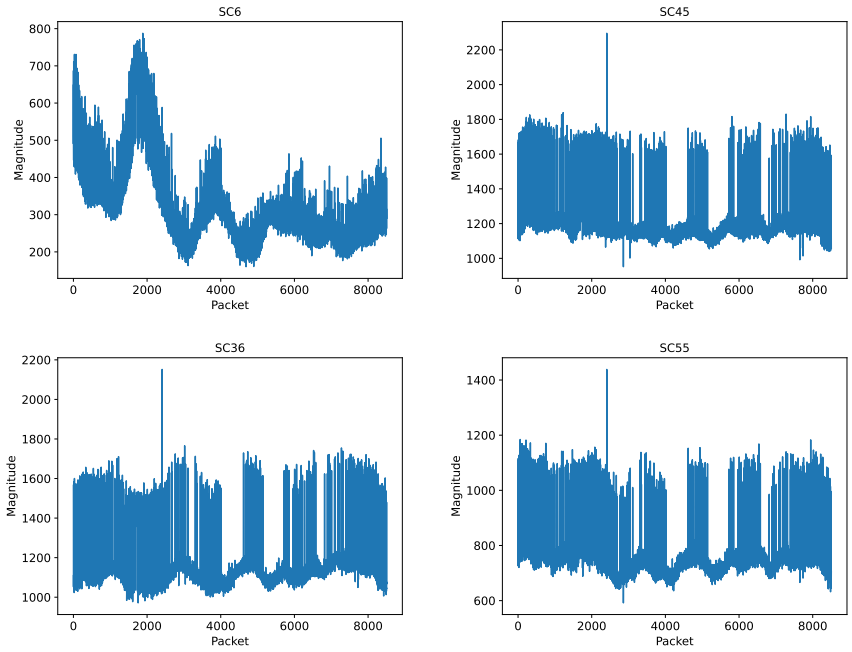

Figure 5.3: Time evolution of packets amplitude on a 20 MHz bandwidth 802.11a channel - Data taken from a different trace than Figure 5.2

be due to the short period for which the sub-carriers have been observed on both the 80 MHz bandwidth 802.11ax and the 20 MHz bandwidth 802.11a channel. The CSI traces collected on both channels are too short to allow accurate observations of the long-term characteristics of the process. We can also observe how, in most instances, the trend eventually tends to stabilize and show a more constant behaviour; this leads us to think that indeed observing this process only for as long as it takes to transmit a few thousands of packets is too short an observation time.

Such a limited time range does not allow a thorough analysis because, although the process seems stationary, there are not enough packets to prove it with absolute certainty. Longer traces would let us estimate the stationary nature of the amplitude variations more confidently.

The unusual fluctuations visible in Figure 5.3 may also be due to the presence of Automatic Gain Control (AGC) at the receiver, which moves the ob-

served frequencies into the desired range. Unfortunately, it is not possible to alter the operation performed by AGC, meaning that we cannot control its influence on the received data.

Our stationarity hypothesis is proven by considering the packet amplitudes collected on each sub-carrier separately and calculating their mean value. Then, sub-carrier by sub-carrier, we divide each dataset into multiple batches of the same length and calculate their mean value. To ensure stationarity, we set a limit so that any variation of a batch average from the corresponding dataset mean value does not cause the process to be discarded as non-stationary if it takes place within the imposed range. The 10% limit we use to tell stationary and non-stationary processes apart is based on empirical measures.

If a stationary process is involved, the average of a sample has the same value, whether it be calculated on the whole dataset using Equation 5.1 or computed in batches by splitting the sample of dimension n into k smaller subsets. The two versions of the operation are allowed by the linearity of the average operator. The second way of calculating the average of a dataset can be implemented using Equation 5.2.

$$\overline{X} = \sum_{i=1}^{n} \frac{x_i}{n} = \frac{1}{n} \sum_{i=1}^{n} x_i \tag{5.1}$$

$$\overline{X} = \frac{1}{k} \sum_{i=1}^{k} \left[\frac{k}{n} \sum_{j=1}^{n/k} x_{(ki+j)} \right] = \frac{1}{k} \sum_{i=1}^{k} \left[\frac{k}{n} \overline{X_i} \right] \tag{5.2}$$

Although non-stationary according to the imposed limit in our study, some processes are plotted anyway, because identifying the "anomalies" and the sub-carriers on which they occur can still provide some interesting insights.

Aside from the analysis of the amplitude evolution on the single sub-carrier,

we can also see that, although the considered sub-carrier changes significantly, the trend followed by amplitudes remains consistent: some minor variations occur that clearly depend on the used sub-carrier, but we can identify some features that stay more or less unaltered.

In Figure 5.1, we can see that all graphs have a local minimum around packet 1,000, just like there is a downward trend followed by an upward one after packet 1,250, regardless of the considered sub-carrier. Similarly, in Figure 5.2 we can spot some shared key features: all graphs have a downward trend in the first 1,000 packets, followed by a local maximum around packet 2,000, after which some fluctuations take place, while later on the trend tends to stabilize.

It is significant to specify that these characteristics are not proper only to the CSI traces displayed in this thesis but, albeit different from the ones described above, multiple features can be identified in other CSI traces showing similar trends on different sub-carriers. As one can imagine, it would not be feasible to show all the graphs that have been plotted during this study.

These considerations lead us to think that there may be a form of correlation that goes beyond time correlation alone: the graphs are often qualitatively similar even though the sub-carriers they refer to are not adjacent, therefore it would be interesting to search for more regularities in their behaviours. A meaningful representation of how closely comparable the graphs of adjacent sub-carriers are is shown in Figures 5.4, 5.5, and 5.6.

Given these observations, it is likely that CSI amplitudes are subject to a time-frequency correlation rather than time correlation alone. We leave the study of this more complex dependency for future research, limiting ours to the analysis of time correlation on the individual sub-carrier.

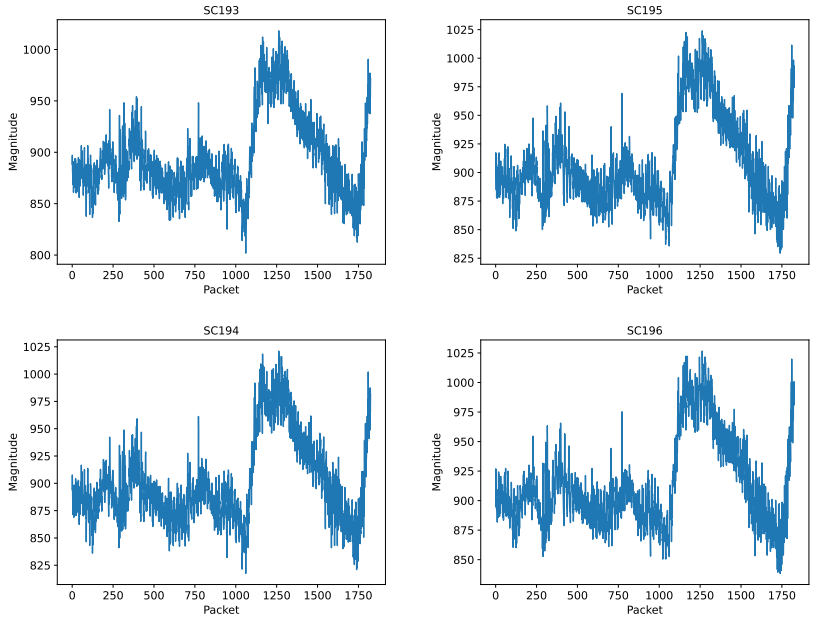

Figure 5.4: Time evolution of packets amplitude on an 80 MHz bandwidth 802.11ax channel

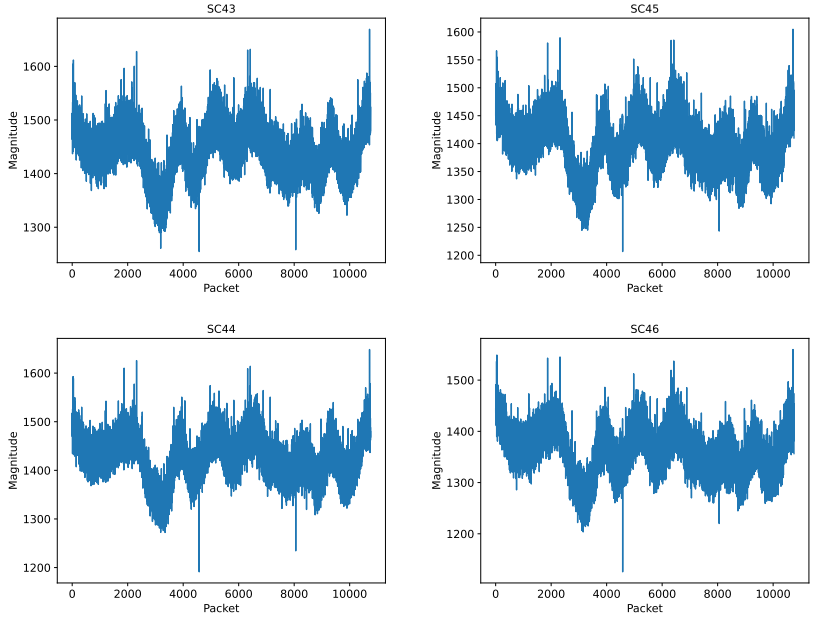

Figure 5.5: Time evolution of packets amplitude on a 20 MHz bandwidth 802.11a channel

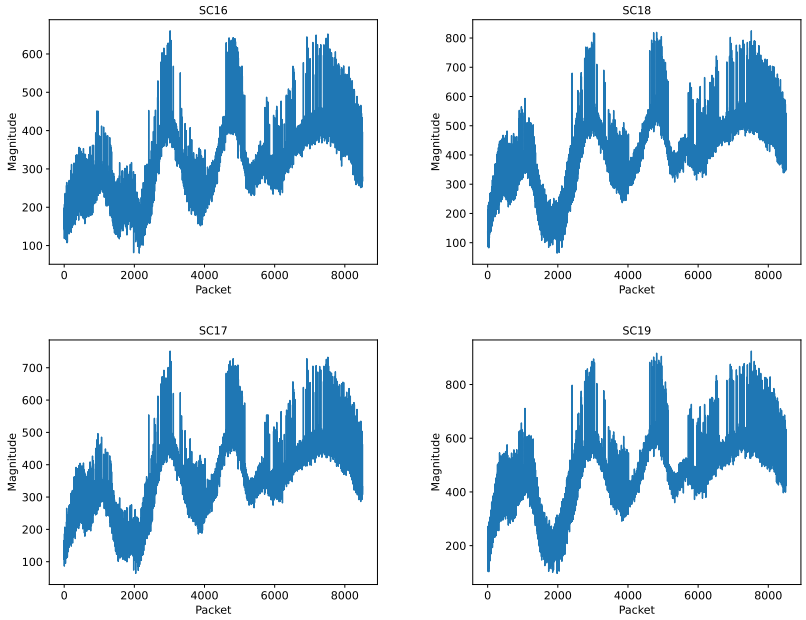

Figure 5.6: Time evolution of packets amplitude on a 20 MHz bandwidth 802.11a channel - Data taken from a different trace than Figure 5.5

5.2 Amplitude Relative Frequency Observation

The packet amplitudes on the different sub-carriers can be shown using histograms having amplitude on the x axis and its relative frequency on the y axis. Some examples of histograms derived from data collected using the 80 MHz bandwidth 802.11ax channel are showcased in Figure 5.7: every graph is relative to the subcarrier specified in its title.

Further examples of histograms derived from data collected using the 20 MHz bandwidth 802.11a channel are shown in Figures 5.8 and 5.9.

By normalizing the plotted amplitudes, as has been done in Figures 5.7, 5.8, and 5.9, we obtain a histogram showing their distribution, which allows us to provide a first hypothesis on the family of statistical distributions that could fit our data. We clearly see that, although the considered sub-carrier changes significantly, the distributions of the normalized frequencies of the amplitudes

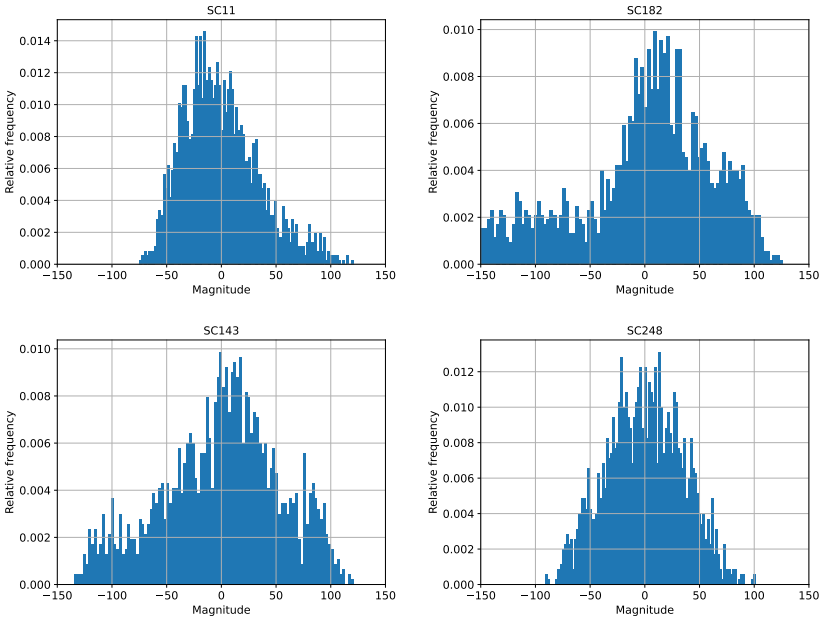

Figure 5.7: Relative frequency of packets amplitude on an 80 MHz bandwidth 802.11ax channel

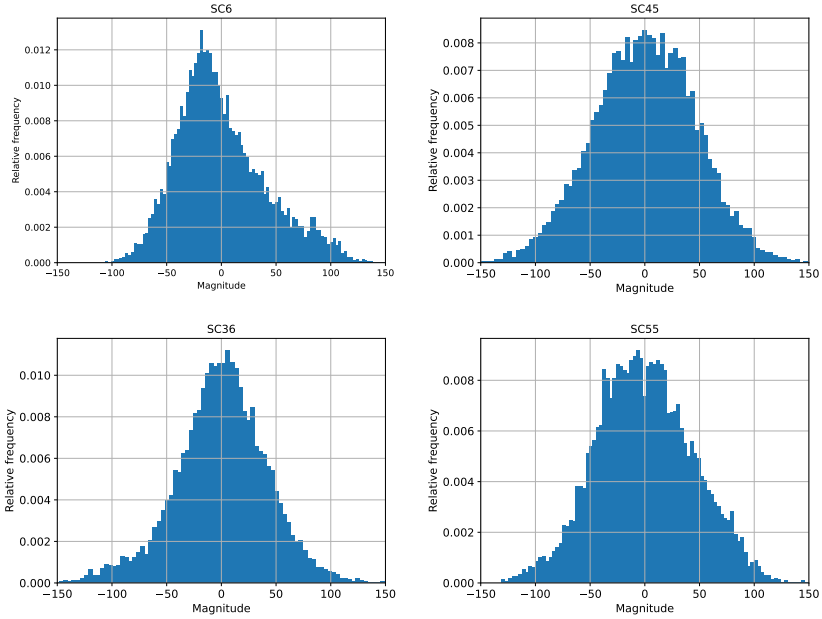

Figure 5.8: Relative frequency of packets amplitude on a 20 MHz bandwidth 802.11a channel

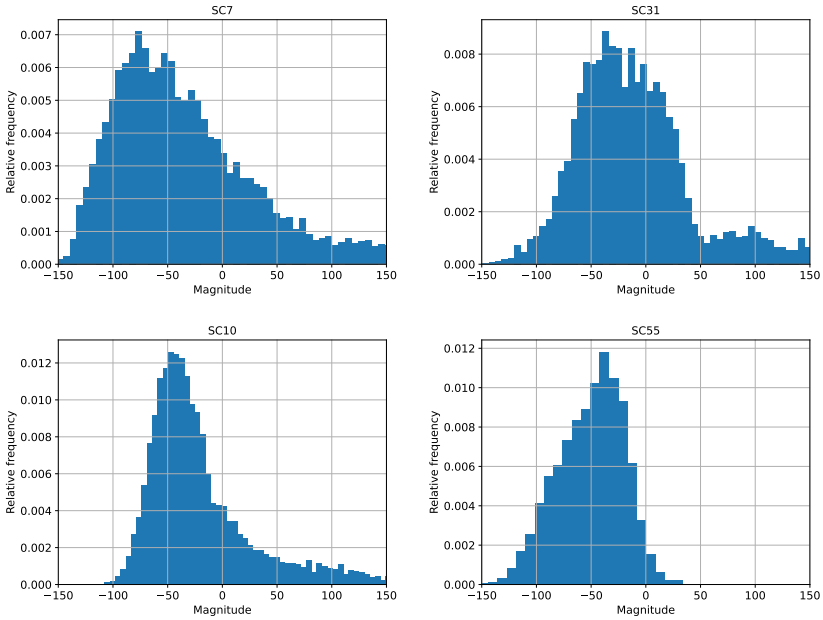

Figure 5.9: Relative frequency of packets amplitude on a 20 MHz bandwidth 802.11a channel - Data taken from a different trace than Figure 5.8

do not differ substantially. This contributes to reinforcing the idea that an analysis of a very likely time-frequency correlation is the natural continuation of this study and it would be interesting to see it carried on in future research.

Rather than only analyzing amplitude relative frequency, which does not let us understand the properties of the process in their entirety, we also take into consideration the increments of the amplitude on each sub-carrier. This part of the analysis is further investigated in Section 5.3.

5.3 Amplitude Increments and Auto-Correlation

To calculate the values of the increments, we simply subtract the amplitude of packet $i-1$ from the amplitude of packet i. This way, we are left with a set of values that can be plotted using histograms that show the distribution of the increments.

The value X of the amplitude associated with the i-th packet on each sub-

carrier depends on the value of the amplitude of the previous packet (packet $i-1$) and a variable increment I, according to Equation 5.3:

$$X_{i+1} = \alpha X_i + I_i \tag{5.3}$$

If $\alpha = 1$, the process whose amplitude values are described by Equation 5.3 has no memory: this means that predictions can be made about the next amplitude value based solely on the present state. Moreover, any prediction has the same probability of being accurate, whether it is based on the current state or the system's history. A process that displays this property is known as a *Markovian* process and is also characterized by the property of "*memorylessness*". In essence, the value of the amplitude of packet $i+1$ depends only on the amplitude of packet i and not on the amplitude of packets $i-1$, $i-2$, and previous ones.

We plot a histogram of the amplitude increments for each sub-carrier (excluding those not used for modulation). Some of these graphs are shown in Figures 5.10, 5.11 and 5.12: the sub-carrier each graph refers to is indicated in the title of the graph.

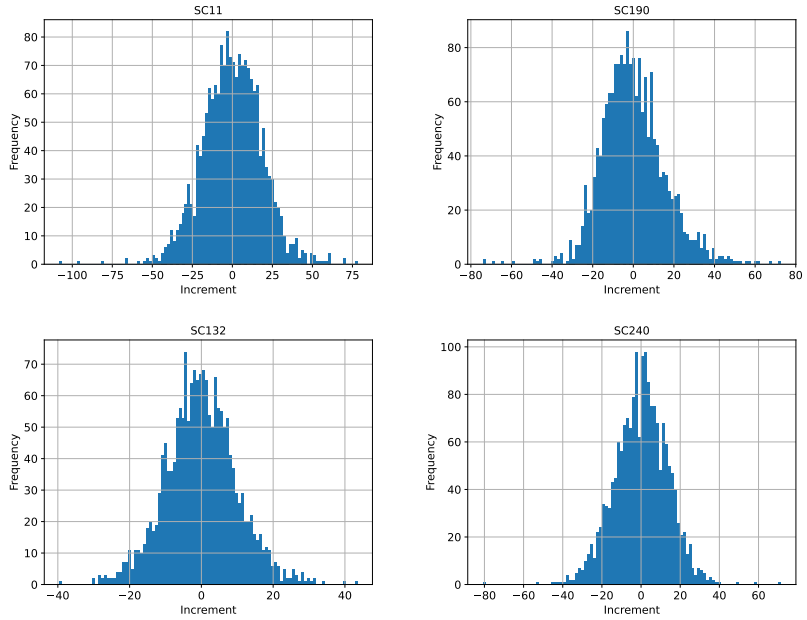

Figure 5.10: Distribution of the increments of packets amplitude on an 80 MHz bandwidth 802.11ax channel

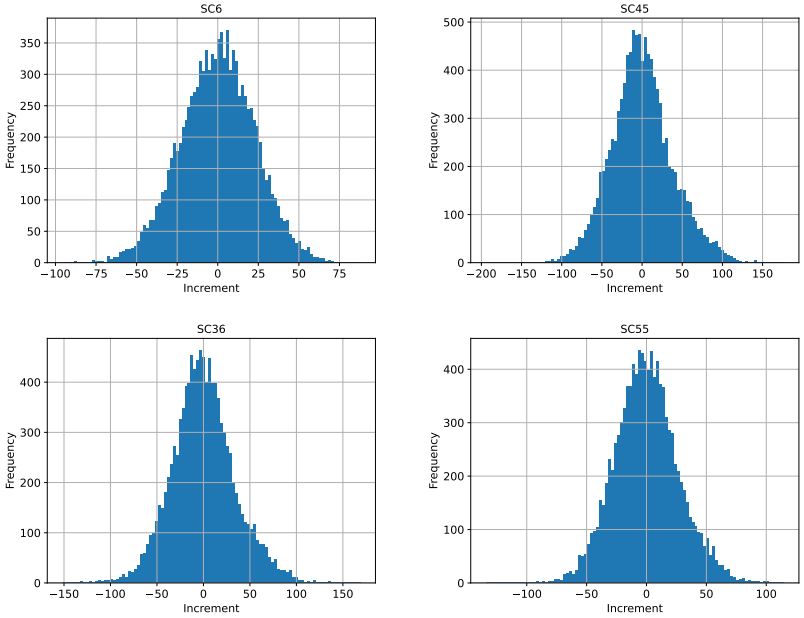

Figure 5.11: Distribution of the increments of packets amplitude on a 20 MHz bandwidth 802.11a channel

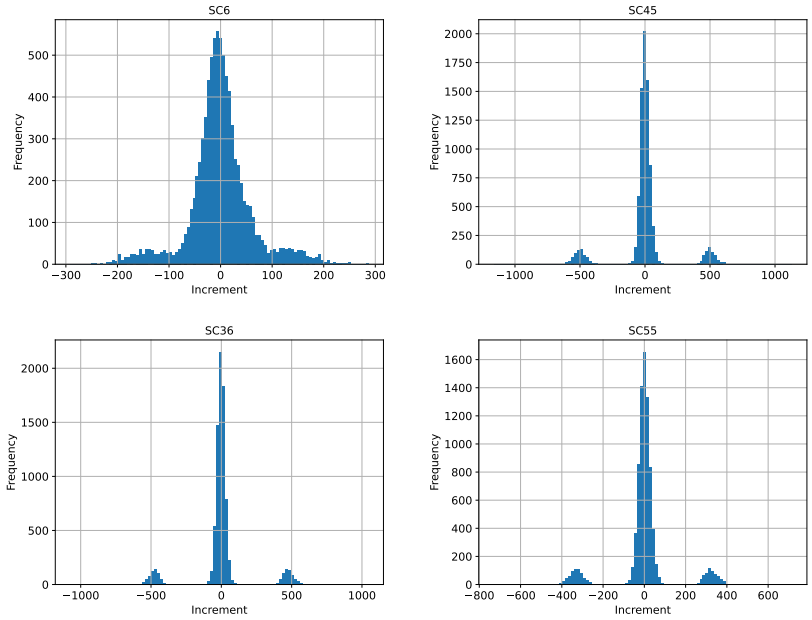

Figure 5.12: Distribution of the increments of packets amplitude on a 20 MHz bandwidth 802.11a channel - Data taken from a different trace than Figure 5.11. Data may be subject to AGC that generates the two smaller peaks on the sides of the one centred in zero.

These histograms highlight that the mean value of the increments calculated sub-carrier by sub-carrier is close to 0 and that the increments are distributed symmetrically to the y axis, suggesting that the process described by these increments may be *Markovian*.

To practically prove the hypothesis enunciated above, we calculate and plot the values of the auto-correlation.

Given a dataset $\{x_i\}$ of size n, we limit the τ variation to a reasonable finite value $\tau_{max} << n$, which is the "window" where we estimate the autocorrelation; then we can evaluate the *sample covariance* as the average of all the $n - \tau_{max}$ possible couples of samples at distance $0 \leq \tau \leq \tau_{max}$

$$Covs(x_i, x_{i+\tau}) = \frac{1}{n - \tau_{max}} \sum_{j=1}^{n-\tau_{max}} (x_j - \overline{X}) \cdot (x_{j+\tau} - \overline{X}) \tag{5.4}$$

Normalizing with respect to S^2 we obtain the *sample normalized auto-covariance*:

$$R'(\tau) = \frac{Covs(x_i, x_{i+\tau})}{S^2} \tag{5.5}$$

Using Equations 5.4 and 5.5 on the dataset obtained by calculating the increments of the amplitude values, we expect the data to display a noise-like behaviour: considering that each increment is a variable value I that is independent of previous increments — i.e., the increments process does not have memory — , we predict that the values displayed in the graph will fluctuate around the x axis. This hypothesis is confirmed by Figures 5.13 and 5.14.

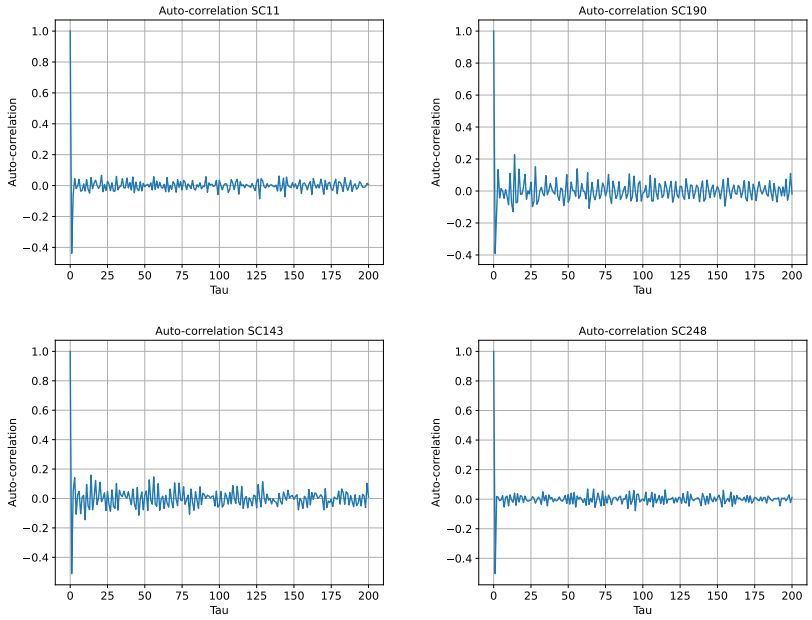

Figure 5.13: Auto-correlation of packets amplitude increments on an 80 MHz bandwidth 802.11ax channel

We can verify that the increments process displays Markovian auto-regression, meaning that the current state depends only on the last state and not on previous ones. By using Equation 5.3, we can find a first description of the state of our system by assigning parameter α value of 1.

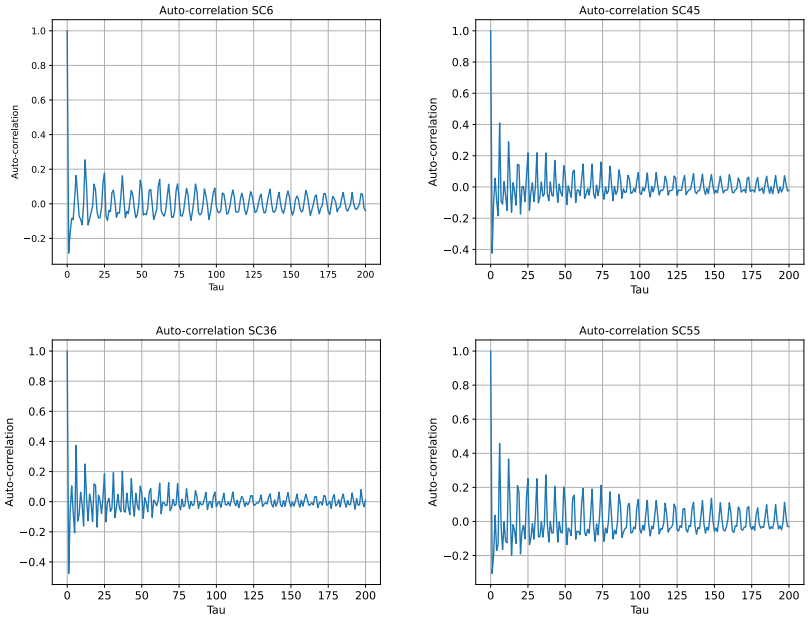

Figure 5.14: Auto-correlation of packets amplitude increments on a 20 MHz bandwidth 802.11a channel

Said equation turns into:

$$X_{i+1} = X_i + I_i \tag{5.6}$$

To better observe the phenomenon, we also calculate the auto-correlation coefficient on the amplitude of the packets. The used formulae are again Equations 5.4 and 5.5. The resulting graphs are presented in Figures 5.15 and 5.16.

As we can see, there is an evident correlation between packets sent over the same sub-carrier. The graphs let us see that auto-correlation displays an uncommon, rippled behaviour. Such a trend may be due to the short time taken to observe the process: a larger number of packets would grant a period that would be long enough to state the stationarity of the process, which instead sometimes appears to be non-stationary in the short term. It would be ideal to have a trace that is much longer than the maximum size of the lags on which auto-correlation is calculated.

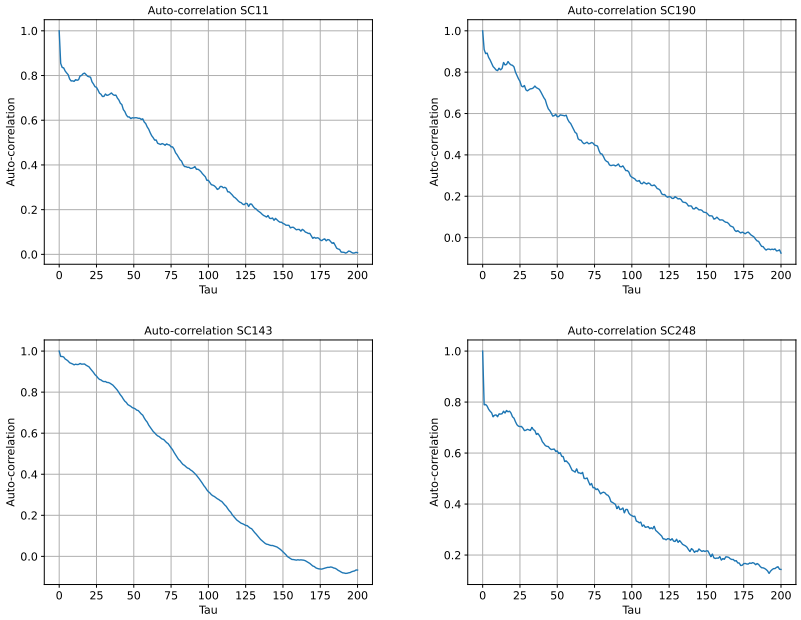

Figure 5.15: Auto-correlation of packets amplitude on an 80 MHz bandwidth 802.11ax channel

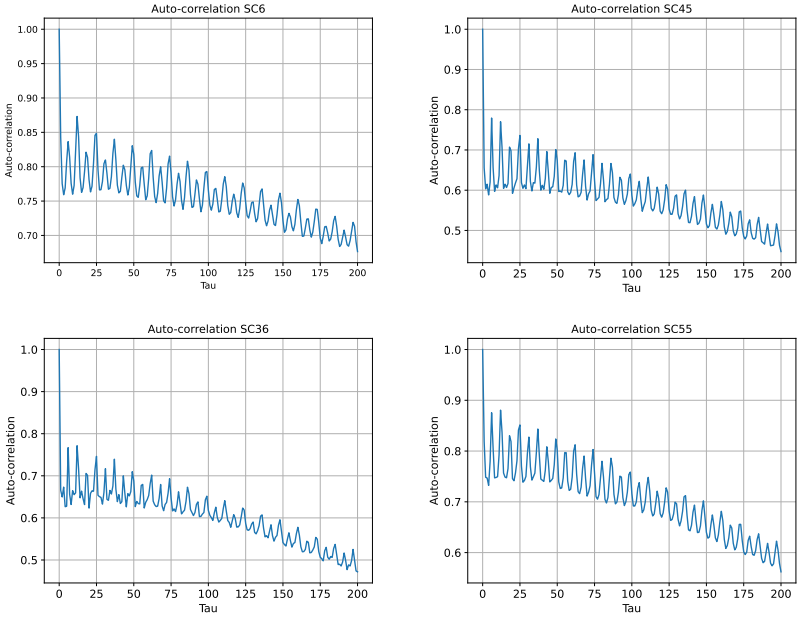

Figure 5.16: Auto-correlation of packets amplitude on a 20 MHz bandwidth 802.11a channel

5.4 Amplitude Increments Analysis

Based on the graphs obtained in Figures 5.10, 5.11, 5.12, and those derived from the same computations performed on other traces we want to identify a statistical distribution that fits the data and mathematically models the process.

Through Python scripts — some of which are described in Appendix A —, we perform a broad analysis to test different distributions. Trace by trace, for each used sub-carrier, we check if there is a distribution belonging to the list below that fits the increments:

- Beta
- Cauchy
- Chi
- Chi-squared
- D-Gamma
- F-distribution
- Folded Cauchy
- Folded Normal
- Gamma
- Generalized Normal
- Half-Cauchy
- Half-Normal
- Inverted Gaussian
- Inverted Gamma
- LogGamma
- LogNormal
- Normal
- Power Law
- Rayleigh

This test is performed using the *Fitter* library [19], which allows the selection of a list of distributions that have to be tested on the data and returns, among other parameters, the values of the parameters that characterize each

distribution.

By identifying the five best-fitting distributions for each sub-carrier, we can narrow down our research field and perform the test once again on the following distributions only (i.e., the ones that come up as the best fits most frequently):

- Gamma
- Generalized Extreme Value
- Generalized Gamma
- Generalized Logistic
- Generalized Normal
- Logistic
- Normal
- Inverted Weibull
- Weibull

These distributions come from the four families of Gamma, Logistic, Gaussian, and Weibull.

It appears that, although many complex distributions appear in the lists above, the Normal distribution shows up most frequently as the best-fitting one in both tests.

As a further means to check the goodness of fit of the Gaussian distribution on our data, we bear the values of its location parameter μ. The values of μ obtained by fitting this distribution on the increments of the data collected on the 802.11ax channel are displayed in the following table:

SC0 -	SC14 $-9.3\cdot10^{-16}$	SC28 $-9.2\cdot10^{-16}$	SC42 $-6.0\cdot10^{-16}$	SC56 $-3.3\cdot10^{-16}$
SC1 -	SC15 $8.8\cdot10^{-16}$	SC29 $-4.3\cdot10^{-16}$	SC43 $1.3\cdot10^{-16}$	SC57 $4.7\cdot10^{-17}$
SC2 -	SC16 $9.7\cdot10^{-17}$	SC30 $-9.3\cdot10^{-16}$	SC44 $2.4\cdot10^{-16}$	SC58 $4.3\cdot10^{-16}$
SC3 -	SC17 $-5.2\cdot10^{-16}$	SC31 $-6.3\cdot10^{-16}$	SC45 $-1.2\cdot10^{-17}$	SC59 $-6\cdot10^{-16}$
SC4 -	SC18 $-1.4\cdot10^{-16}$	SC32 $1.8\cdot10^{-16}$	SC46 $-1.1\cdot10^{-15}$	SC60 $-1.7\cdot10^{-16}$
SC5 -	SC19 $-7.7\cdot10^{-16}$	SC33 $6.2\cdot10^{-16}$	SC47 $8.2\cdot10^{-17}$	SC61 $-6.8\cdot10^{-16}$
SC6 $8.7\cdot10^{-16}$	SC20 $2.4\cdot10^{-16}$	SC34 $-4.1\cdot10^{-16}$	SC48 $5.7\cdot10^{-16}$	SC62 $3.6\cdot10^{-16}$
SC7 $7.9\cdot10^{-16}$	SC21 $-1.8\cdot10^{-16}$	SC35 $-2.3\cdot10^{-16}$	SC49 $-6.5\cdot10^{-16}$	SC63 $3\cdot10^{-16}$
SC8 $-3.2\cdot10^{-16}$	SC22 $7.1\cdot10^{-16}$	SC36 $6.1\cdot10^{-16}$	SC50 $-2.1\cdot10^{-16}$	SC64 $-2.5\cdot10^{-16}$
SC9 $7.1\cdot10^{-16}$	SC23 $5.0\cdot10^{-16}$	SC37 $3.7\cdot10^{-16}$	SC51 $-1.2\cdot10^{-15}$	SC65 $4.4\cdot10^{-16}$
SC10 $3.8\cdot10^{-16}$	SC24 $3.7\cdot10^{-16}$	SC38 $4.6\cdot10^{-16}$	SC52 $2.5\cdot10^{-16}$	SC66 $-1.6\cdot10^{-16}$
SC11 $-4.7\cdot10^{-16}$	SC25 $-1.7\cdot10^{-16}$	SC39 $2.9\cdot10^{-16}$	SC53 $1.6\cdot10^{-17}$	SC67 $4.4\cdot10^{-16}$
SC12 $-8.4\cdot10^{-16}$	SC26 $-1.0\cdot10^{-15}$	SC40 $-8.9\cdot10^{-17}$	SC54 $-2.8\cdot10^{-16}$	SC68 $-3.4\cdot10^{-16}$
SC13 $-1.5\cdot10^{-15}$	SC27 $-6.1\cdot10^{-16}$	SC41 $4.3\cdot10^{-16}$	SC55 $-5.8\cdot10^{-16}$	SC69 $3.2\cdot10^{-16}$

SC70 $4.2{\cdot}10^{-16}$	SC108 $1.1{\cdot}10^{-16}$	SC146 $-2.5{\cdot}10^{-16}$	SC184 $-10{\cdot}10^{-16}$	SC222 $6.2{\cdot}10^{-17}$
SC71 $3{\cdot}10^{-16}$	SC109 $-2.8{\cdot}10^{-16}$	SC147 $2.3{\cdot}10^{-16}$	SC185 $4.4{\cdot}10^{-16}$	SC223 $-5.8{\cdot}10^{-16}$
SC72 $-4.2{\cdot}10^{-16}$	SC110 $6.8{\cdot}10^{-16}$	SC148 $-3.7{\cdot}10^{-16}$	SC186 $-1.7{\cdot}10^{-16}$	SC224 $-6.2{\cdot}10^{-16}$
SC73 $-3.2{\cdot}10^{-16}$	SC111 $5.7{\cdot}10^{-16}$	SC149 $-8.2{\cdot}10^{-17}$	SC187 $3.4{\cdot}10^{-16}$	SC225 $-4.2{\cdot}10^{-16}$
SC74 $-3.7{\cdot}10^{-16}$	SC112 $4.2{\cdot}10^{-16}$	SC150 $3.4{\cdot}10^{-16}$	SC188 $5.5{\cdot}10^{-16}$	SC226 $-9.5{\cdot}10^{-16}$
SC75 $2.3{\cdot}10^{-16}$	SC113 $2.6{\cdot}10^{-16}$	SC151 $-1.6{\cdot}10^{-16}$	SC189 $7.1{\cdot}10^{-16}$	SC227 $-3.9{\cdot}10^{-16}$
SC76 $-5.1{\cdot}10^{-16}$	SC114 0.0	SC152 $-2.5{\cdot}10^{-16}$	SC190 $-8.5{\cdot}10^{-16}$	SC228 $6{\cdot}10^{-16}$
SC77 $2.9{\cdot}10^{-16}$	SC115 $-4.9{\cdot}10^{-16}$	SC153 $-8.3{\cdot}10^{-16}$	SC191 $-4.7{\cdot}10^{-17}$	SC229 $2.2{\cdot}10^{-16}$
SC78 $4.7{\cdot}10^{-16}$	SC116 $-3.2{\cdot}10^{-16}$	SC154 $-4.4{\cdot}10^{-16}$	SC192 $-8.0{\cdot}10^{-16}$	SC230 $5.8{\cdot}10^{-16}$
SC79 $-2.6{\cdot}10^{-16}$	SC117 $-5.4{\cdot}10^{-16}$	SC155 $1.6{\cdot}10^{-17}$	SC193 $4{\cdot}10^{-16}$	SC231 $-3.8{\cdot}10^{-16}$
SC80 $-1.3{\cdot}10^{-16}$	SC118 $-1.1{\cdot}10^{-16}$	SC156 $-2.3{\cdot}10^{-17}$	SC194 $2.3{\cdot}10^{-16}$	SC232 $-1.1{\cdot}10^{-15}$
SC81 $3{\cdot}10^{-16}$	SC119 $4.1{\cdot}10^{-16}$	SC157 $1.2{\cdot}10^{-16}$	SC195 $-6.9{\cdot}10^{-16}$	SC233 $7.1{\cdot}10^{-16}$
SC82 $-4.3{\cdot}10^{-16}$	SC120 $5.6{\cdot}10^{-16}$	SC158 $3.9{\cdot}10^{-16}$	SC196 $7.3{\cdot}10^{-16}$	SC234 $-5{\cdot}10^{-16}$
SC83 $3.7{\cdot}10^{-16}$	SC121 $-5{\cdot}10^{-16}$	SC159 $-1.8{\cdot}10^{-16}$	SC197 $1.9{\cdot}10^{-16}$	SC235 $1.6{\cdot}10^{-16}$
SC84 $2.0{\cdot}10^{-16}$	SC122 $6.4{\cdot}10^{-16}$	SC160 $-4.4{\cdot}10^{-16}$	SC198 $3.2{\cdot}10^{-16}$	SC236 $-3{\cdot}10^{-16}$
SC85 $-1.2{\cdot}10^{-16}$	SC123 $-7.8{\cdot}10^{-17}$	SC161 $-5.0{\cdot}10^{-16}$	SC199 $2.0{\cdot}10^{-16}$	SC237 $-3.2{\cdot}10^{-16}$
SC86 $-4.7{\cdot}10^{-17}$	SC124 $2.5{\cdot}10^{-16}$	SC162 $1.6{\cdot}10^{-16}$	SC200 $3.0{\cdot}10^{-16}$	SC238 $-3.9{\cdot}10^{-16}$
SC87 $5.8{\cdot}10^{-16}$	SC125 $8.8{\cdot}10^{-16}$	SC163 $-4.7{\cdot}10^{-17}$	SC201 $5.1{\cdot}10^{-16}$	SC239 $-6.1{\cdot}10^{-16}$
SC88 $4.7{\cdot}10^{-16}$	SC126 $-5.8{\cdot}10^{-16}$	SC164 $1.8{\cdot}10^{-16}$	SC202 $-1.2{\cdot}10^{-16}$	SC240 $-1.8{\cdot}10^{-16}$
SC89 $-2{\cdot}10^{-16}$	SC127 -	SC165 $9.8{\cdot}10^{-16}$	SC203 $-6.8{\cdot}10^{-16}$	SC241 $6.7{\cdot}10^{-16}$
SC90 $8.8{\cdot}10^{-16}$	SC128 -	SC166 $-4.5{\cdot}10^{-16}$	SC204 $1.4{\cdot}10^{-15}$	SC242 $4.8{\cdot}10^{-16}$
SC91 $-4.3{\cdot}10^{-16}$	SC129 -	SC167 $-6.3{\cdot}10^{-16}$	SC205 $-1.2{\cdot}10^{-15}$	SC243 $-2.5{\cdot}10^{-16}$
SC92 $-1.7{\cdot}10^{-16}$	SC130 $-2.0{\cdot}10^{-16}$	SC168 $-8{\cdot}10^{-16}$	SC206 $1.4{\cdot}10^{-15}$	SC244 $1.5{\cdot}10^{-15}$
SC93 $5.5{\cdot}10^{-16}$	SC131 $-2.6{\cdot}10^{-16}$	SC169 $-5{\cdot}10^{-16}$	SC207 $5.3{\cdot}10^{-16}$	SC245 $-9.9{\cdot}10^{-16}$
SC94 $-5.6{\cdot}10^{-16}$	SC132 $-2.5{\cdot}10^{-16}$	SC170 $-1.1{\cdot}10^{-16}$	SC208 $6.6{\cdot}10^{-16}$	SC246 $1.1{\cdot}10^{-15}$
SC95 $-1.5{\cdot}10^{-17}$	SC133 $-7.8{\cdot}10^{-18}$	SC171 $-5.5{\cdot}10^{-16}$	SC209 $-1.2{\cdot}10^{-16}$	SC247 $5.2{\cdot}10^{-16}$
SC96 $-1.2{\cdot}10^{-16}$	SC134 $-2.3{\cdot}10^{-16}$	SC172 $-6.6{\cdot}10^{-16}$	SC210 $-2.5{\cdot}10^{-16}$	SC248 $1.8{\cdot}10^{-16}$
SC97 $4.4{\cdot}10^{-16}$	SC135 $-5.2{\cdot}10^{-16}$	SC173 $6.3{\cdot}10^{-16}$	SC211 $7.1{\cdot}10^{-16}$	SC249 $5.5{\cdot}10^{-16}$
SC98 $1.7{\cdot}10^{-16}$	SC136 $-1.4{\cdot}10^{-16}$	SC174 $5.8{\cdot}10^{-16}$	SC212 $8.9{\cdot}10^{-17}$	SC250 $-1.9{\cdot}10^{-15}$
SC99 $-1.1{\cdot}10^{-16}$	SC137 $-3{\cdot}10^{-16}$	SC175 $3.7{\cdot}10^{-16}$	SC213 $7.1{\cdot}10^{-16}$	SC251 -
SC100 $-5.5{\cdot}10^{-16}$	SC138 $3.6{\cdot}10^{-16}$	SC176 $8.3{\cdot}10^{-16}$	SC214 $-6{\cdot}10^{-16}$	SC252 -
SC101 $3.6{\cdot}10^{-16}$	SC139 $7.8{\cdot}10^{-17}$	SC177 $-1.0{\cdot}10^{-16}$	SC215 $-8.7{\cdot}10^{-16}$	SC253 -
SC102 $5{\cdot}10^{-16}$	SC140 $1.5{\cdot}10^{-16}$	SC178 $1.7{\cdot}10^{-16}$	SC216 $-5.5{\cdot}10^{-16}$	SC254 -
SC103 $3.4{\cdot}10^{-16}$	SC141 $4.1{\cdot}10^{-16}$	SC179 $5.5{\cdot}10^{-16}$	SC217 $7.8{\cdot}10^{-18}$	SC255 -
SC104 $2.3{\cdot}10^{-16}$	SC142 $1.1{\cdot}10^{-16}$	SC180 $3.5{\cdot}10^{-16}$	SC218 $5.5{\cdot}10^{-17}$	
SC105 $3.5{\cdot}10^{-16}$	SC143 $-3.3{\cdot}10^{-16}$	SC181 $5.5{\cdot}10^{-16}$	SC219 $6.7{\cdot}10^{-16}$	
SC106 $2.9{\cdot}10^{-16}$	SC144 $-4.4{\cdot}10^{-16}$	SC182 $2.3{\cdot}10^{-16}$	SC220 $1.1{\cdot}10^{-15}$	
SC107 $-5.3{\cdot}10^{-16}$	SC145 $-1.2{\cdot}10^{-16}$	SC183 $-4.9{\cdot}10^{-16}$	SC221 $-3.1{\cdot}10^{-16}$	

The results displayed above let us see that the mean value of the increments is already extremely close to zero, being in the order of 10^{-16} for most sub-carriers, despite only having a few packets available on each sub-carrier. By increasing the number of packets we would most likely have mean values even closer to zero, reinforcing the hypothesis that the Normal distribution is the best-fitting one.

Alongside the table displayed above, we also show one containing the values of the scale parameter σ obtained by fitting the Gaussian distribution on the same data as the ones used to find the values of μ.

SC0 -
SC1 -
SC2 -
SC3 -
SC4 -
SC5 -
SC6 24.3
SC7 26.3
SC8 23.3
SC9 17.7
SC10 16.3
SC11 18.8
SC12 20.6
SC13 22.1
SC14 21.6
SC15 19.9
SC16 17.3
SC17 15.9
SC18 16.0
SC19 17
SC20 16.7
SC21 16.5
SC22 16.3
SC23 16.7
SC24 16.6
SC25 16.5
SC26 16.4
SC27 17.2
SC28 17.9
SC29 18.4
SC30 17.7
SC31 17.7
SC32 18
SC33 18.5
SC34 18.4
SC35 18.2
SC36 18.3
SC37 18.7
SC38 18.6
SC39 18
SC40 17.3
SC41 17.1
SC42 17.4
SC43 17.2
SC44 16.6
SC45 16.2
SC46 16.4
SC47 16.9
SC48 17.7
SC49 18.9
SC50 18.7
SC51 18.9
SC52 17.9
SC53 16.7
SC54 14.7
SC55 13.4
SC56 12.9
SC57 12.4
SC58 12.1
SC59 12.7
SC60 14.1
SC61 14.7
SC62 15.3
SC63 15.8
SC64 16.3
SC65 16.1
SC66 15.2
SC67 14.5
SC68 14.5
SC69 15.1
SC70 15.5
SC71 14.9
SC72 14.2
SC73 13.4
SC74 14.3
SC75 15.4
SC76 17.9
SC77 18.6
SC78 18.5
SC79 16.0
SC80 13.7
SC81 12.9
SC82 14.1
SC83 15.7
SC84 16
SC85 15.1
SC86 13.9
SC87 13.6
SC88 16.2
SC89 19.3
SC90 20.8
SC91 20.9
SC92 17.9
SC93 14.5
SC94 11.6
SC95 11.7
SC96 12.8
SC97 13.6
SC98 13.4
SC99 14.1
SC100 13.4
SC101 13.7
SC102 14.6
SC103 14.5
SC104 12.8
SC105 11.2
SC106 10.7
SC107 10.9
SC108 10.8
SC109 10.9
SC110 10.7
SC111 10.9
SC112 11.6
SC113 12.5
SC114 13.4
SC115 14.2
SC116 14.1
SC117 13.5
SC118 12.4
SC119 11.7
SC120 11.6
SC121 11.8
SC122 12.1
SC123 12.4
SC124 13.3
SC125 15.2
SC126 16.8
SC127 -
SC128 -
SC129 -
SC130 14.8
SC131 11.9
SC132 9.9
SC133 11.5
SC134 11.9
SC135 11.5
SC136 10.1
SC137 9.1
SC138 8.9
SC139 11.4
SC140 13.5
SC141 14.7
SC142 13.7
SC143 12.3
SC144 10.6
SC145 9.7
SC146 10.3
SC147 11.1
SC148 12
SC149 12.2
SC150 11.9
SC151 11.1
SC152 11.5
SC153 11.8
SC154 12.3
SC155 12.4
SC156 12.5
SC157 12.2
SC158 12.1
SC159 12.2
SC160 12.2
SC161 12.6
SC162 12.9
SC163 14.2
SC164 15
SC165 16.0
SC166 16.2
SC167 15.9
SC168 15.2
SC169 14.3
SC170 14.2
SC171 14.7
SC172 15
SC173 15.6
SC174 15.9
SC175 16.3
SC176 17.3
SC177 18.3
SC178 19.6
SC179 19.5
SC180 18.8
SC181 16.7
SC182 15.3
SC183 14.5
SC184 14.1
SC185 14.4
SC186 14.4
SC187 14.7
SC188 14.8
SC189 15.1
SC190 15.7
SC191 16.2
SC192 17.2
SC193 18.1
SC194 18.4
SC195 17.5
SC196 16.3
SC197 15.9
SC198 16.1
SC199 16.4
SC200 15.5
SC201 14.3
SC202 14.7
SC203 17.7
SC204 20.2
SC205 22.2
SC206 21.0
SC207 17.8
SC208 13.7
SC209 11.6
SC210 13.1
SC211 15.8
SC212 16.8
SC213 16.1
SC214 14.1
SC215 14.3
SC216 17.4
SC217 22.1
SC218 24.3
SC219 23.3
SC220 19.8
SC221 14.9
SC222 11.8
SC223 12.4
SC224 14.1
SC225 15.8
SC226 16.4
SC227 18.3
SC228 20.2
SC229 21.3
SC230 21.3
SC231 20.2
SC232 18.6
SC233 16.7
SC234 15.7
SC235 14.6
SC236 13.3
SC237 12.8
SC238 12.1
SC239 12.1
SC240 13.6
SC241 16.8
SC242 20.3
SC243 23.2
SC244 24.5
SC245 24
SC246 23.1
SC247 22.2
SC248 22.7
SC249 21.8
SC250 28.9
SC251 -
SC252 -
SC253 -
SC254 -
SC255 -

6 A Simple Markovian Model

In the previous chapter, we discussed some graphic examples of the outcomes of the calculations we performed: we have observed that a one-step memory can accurately describe the process of the increments, leading us to the conclusion that the amplitude variations on a single isolated sub-carrier behave according to a Gauss-Markovian process.

We now delve into the validation of this hypothesis by examining the behaviour of artificially generated traces. Distinct traces have been created to fit the specifics of the 802.11ax and the 802.11a channels. If the results we get from processing both kinds of artificial traces are similar to those we have obtained from processing real data, we can safely state that our potential model is an accurate description of the observed phenomenon.

Through a Python script, for each sub-carrier, we generate an artificial trace. The trace features increment values coming from a Gaussian distribution having zero as mean value μ, as the observed mean value for the increments collected on both channels is extremely close to zero, and the best-fitting distribution seems to be the Gaussian distribution. The value of σ to produce a specific distribution for each sub-carrier is taken from the table shown at the end of Chapter 5.

All processing performed on the artificial traces abides by the same algorithm as the one previously used on real data and all the analyses focus on the same aspects as those presented in Chapter 5.

6.1 Time Evolution

The amplitudes evolution in time obtained from the artificial trace highlights — as we can expect — that the sub-carriers created through Python code are independent of each other: this can be inferred from the fact that graphs obtained from adjacent sub-carriers (see Figure 6.1) do not show significantly related properties.

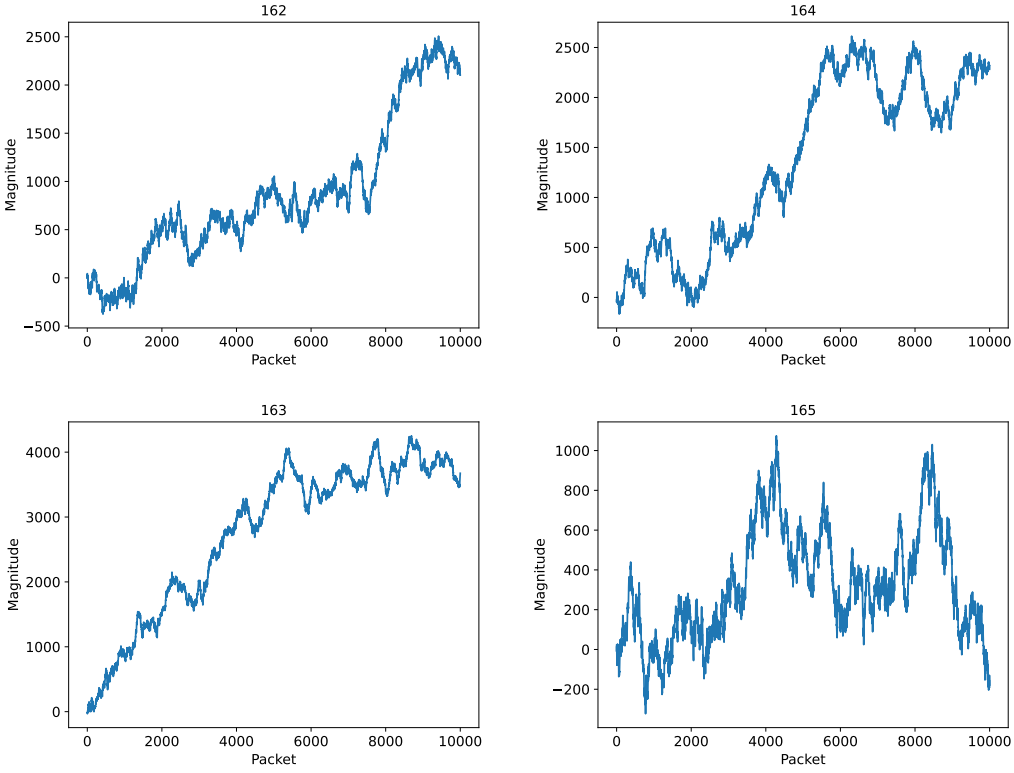

Figure 6.1: Time evolution of packets amplitude on an artificial trace fitting the specifics of an 80 MHz bandwidth 802.11ax channel

On the contrary, by interpreting the results displayed in Figures 5.1 and 5.2 — as has been done more thoroughly in the previous chapter — , we can safely state that there is some form of frequency correlation between adjacent sub-carriers, given that the displayed trends coming from subsequent sub-carriers

are easily comparable.

This main difference between real and artificial data demonstrates that research in this field still has many aspects to analyze and is open to future expansion.

6.2 Increments Distribution

By observing Figure 6.2 we can see that there actually are some visually evident similarities between real and artificial increments relative to an 80 MHz bandwidth 802.11ax channel. This confirms that the Normal distribution from which we derive our artificial data can indeed be accurately descriptive of the observed process, as it generates data confirming our previous analyses.

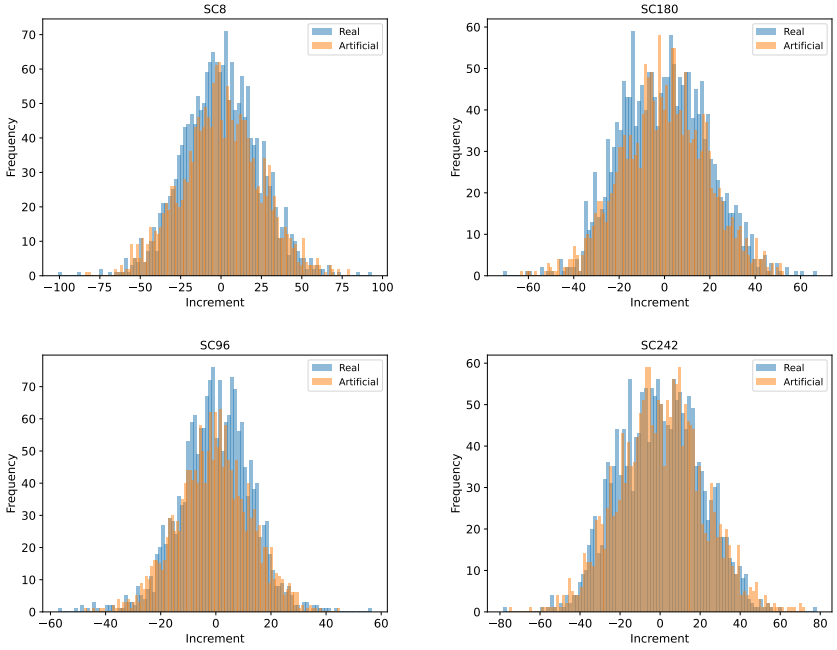

Figure 6.2: Distribution of the increments of packets amplitude on an artificial trace (orange) fitting the specifics of an 80 MHz bandwidth 802.11ax channel compared with the increments calculated on collected data (blue)

Alongside Figure 6.2, to prove the versatility of our model, we also com-

pare the distribution of amplitude increments on a 20 MHz bandwidth 802.11a channel with an artificial trace created to fit the specifics of this second channel. Said comparison can be made by observing the similarities highlighted in Figure 6.3.

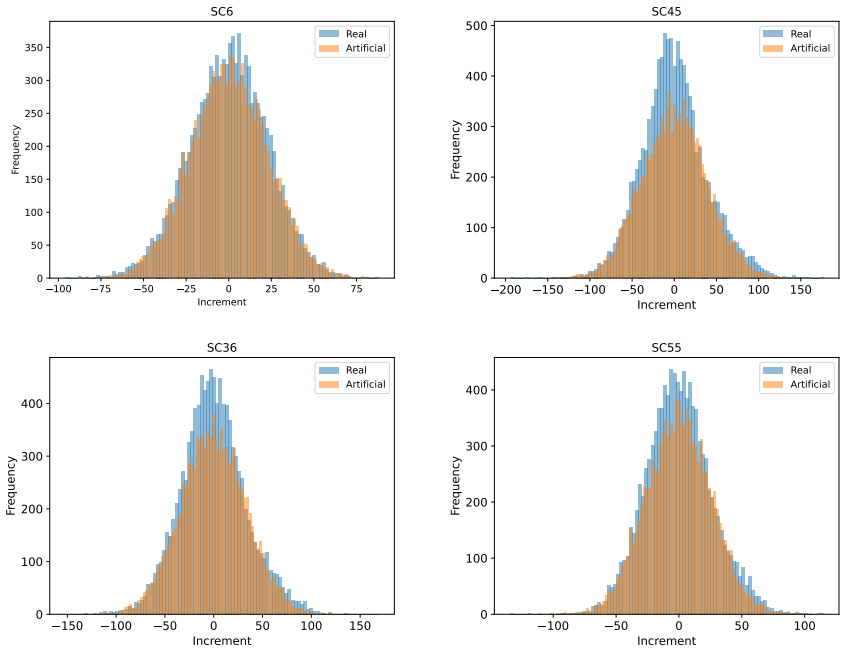

Figure 6.3: Distribution of the increments of packets amplitude on an artificial trace (orange) fitting the specifics of a 20 MHz bandwidth 802.11a channel compared with the increments calculated on collected data (blue)

6.3 Increments Auto-correlation

Apart from having a mean value of zero, which is granted by the fact that they have been generated through code to do so, increments display similar behaviour to that shown in Figures 5.13 and 5.14.

The *memorylessness* of the increments process is confirmed by the graphs representing the auto-correlation: the values of the auto-correlation coefficient display the noise-like behaviour (see Figure 6.4) we would expect data coming

from a memoryless distribution to have. This supports the results of our previous analysis that deduced the Markovian nature of the observed process from the fact that the obtained auto-correlation is similar to the typical one of processes with one-step memory.

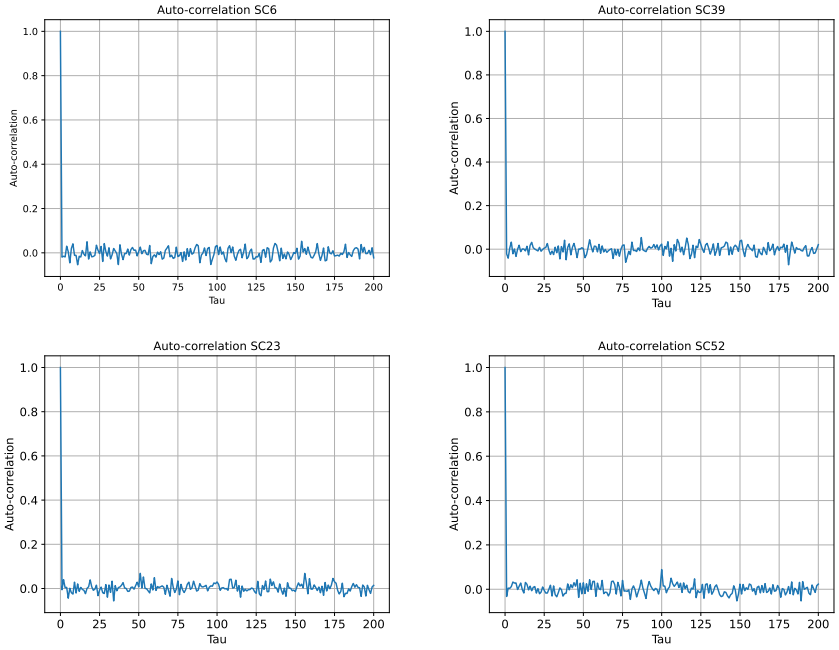

Figure 6.4: Auto-correlation of the increments of packets amplitude on an artificial trace fitting the specifics of an 80 MHz bandwidth 802.11ax channel

To provide a complete comparison between real and artificial data, we plot the graphs showing the auto-correlation of packets amplitude on each sub-carrier, as displayed in Figure 6.5.

We can see that these graphs display a much cleaner behaviour than those obtained from real data: this confirms our hypothesis that real data have some form of inter-carrier dependency which would need to be included in our model to describe the channel more accurately and comprehensively. The ripples shown in Figure 5.15 therefore derive from a complex correlation that takes into consideration frequency as well as time.

If we limit our analysis to time correlation by observing each sub-carrier

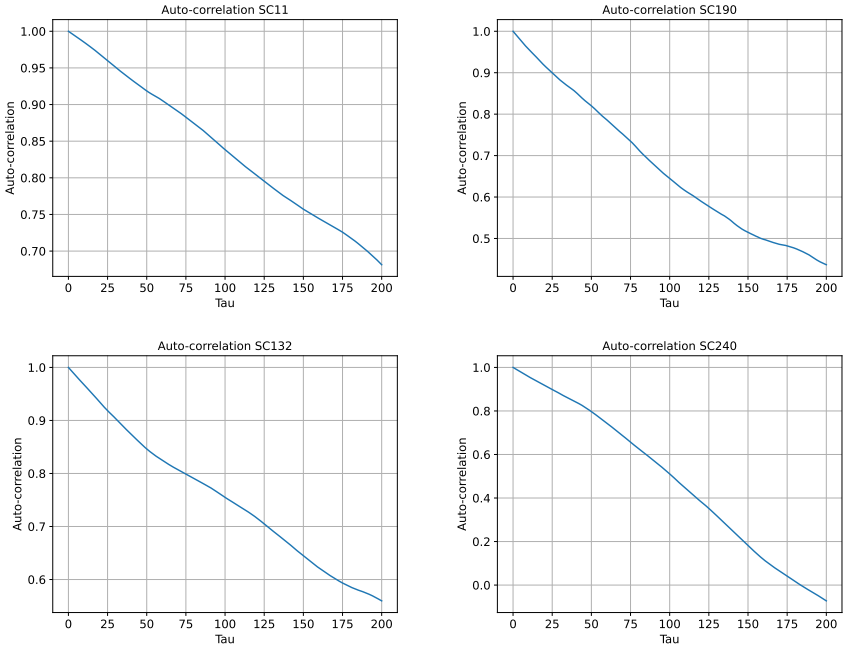

Figure 6.5: Auto-correlation of packets amplitude of an artificial trace fitting the specifics of an 80 MHz bandwidth 802.11ax channel

separately, we can consider the proposed Gauss-Markov model descriptive of the process.

6.4 Reliability of Artificial Data Analysis

To formally ensure that the artificial traces behave similarly to the collected ones, we would need to perform an acceptance test by comparing the outcomes of the processing performed on the artificial traces to those obtained from processing the data collected on the Wi-Fi channels.

As stated earlier in this thesis, the available traces collected on the wireless channels are too short to allow a perfect description of the phenomenon; this also means that we cannot consider our traces representative of the whole population. To perform an accurate acceptance test, we would need to compare the results obtained from processing the artificial traces with results obtained

from a whole population. Without a wide enough sample, we cannot perform an acceptance test that would give us back comprehensive results.

Nevertheless, we can already see that most empirical results and interpretations of processed data are confirmed by artificially generated traces that behave as we expect them to.

7 Conclusions and Future Works

The question we posed at the beginning of this thesis was whether the Channel State Information relative to a wireless channel had any consistent behaviour that we could outline using a defined mathematical model.

We started by processing CSI traces collected on two different Wi-Fi channels. Because every CSI corresponds to a complex number, the initial simplification we introduced was to restrict the study to the sole amplitude, omitting the phase. As said in Chapter 5, it would be best to take into consideration both the amplitude and the phase of the CSI traces we analyze. Working in the complex field from the start would not have allowed us to work up our solution step by step, therefore we chose to temporarily simplify the problem by working with real numbers only, leaving studies on CSI phase values to future research.

Our working method has given us the chance to observe the phenomena we were trying to describe from a simpler, yet effective, point of view, which made it possible to split our work into more basic steps that we eventually combined to obtain the final result.

The first step in our analysis consisted in plotting the amplitude time evolution for each sub-carrier of the considered channel. We have seen that the trends tend to stabilize with time, but it was evident that the available traces were too short to grant universally valid results.

It was also noticeable that adjacent sub-carriers influence each other's packet amplitude, suggesting that time correlation alone would be too simplistic a model. The inter-sub-carrier correlation phenomenon (which we pre-

viously called frequency correlation) is too complex to describe for the time being, which is the reason why we also left this branch of the analysis to upcoming research.

We decided to go in this direction because it made it possible to obtain significant results that could work as a basis to future studies. Surely, it would be best to perform this research in a more structured way, possibly by expanding the size of the available datasets and comparing more outcomes we would get from applying the algorithm we created for this case study.

To facilitate the comparison of the outputs we obtained, we plotted the distribution of the amplitude increments for each sub-carrier; we then realized that the increments tend to distribute symmetrically to the y axis. This led us to think that the statistical distribution we were looking for had to be symmetrical and have a mean value close to zero.

Subsequently, we analyzed the values of the increments auto-correlation coefficient, concluding that they behave according to a Markovian process (i.e., a *memoryless* process).

Finally, we tried to fit some selected distributions on the increments and found that the Normal distribution was accurately descriptive of the studied process. We concluded by asserting that the process displays a Gauss-Markovian behaviour: such model may seem quite simple, yet it fits the data well enough to be considered comprehensive of the main features we included in our study.

The work we have led up to this point opens up to future expansion and further investigations. Disposing of longer traces (about a million packets per trace would be the minimum needed quantity) would surely grant more reliability and wider validity to the results, confirming whether there is any form of dependency of the channel behaviour from the used channel itself. These data would also allow us to analyze the frequency correlation that is

entwined with the time correlation, to provide a more accurate model of the process.

The analysis we illustrated in this thesis consists only of the first few stages of a much wider study. Nonetheless, we have come up with a partial solution that satisfies our initial goals and paves the road to future more in-depth analyses.

Bibliography

[1] H. Tataria, M. Shafi, A. F. Molisch, M. Dohler, H. Sjöland, and F. Tufvesson, "6G Wireless Systems: Vision, Requirements, Challenges, Insights, and Opportunities," *Proceedings of the IEEE*, vol. 109, no. 7, pp. 1166–1199, Mar. 2021.

[2] A. Ashleibta, A. Taha, M. Khan, et al., "5G-enabled contactless multi-user presence and activity detection for independent assisted living," *Nature Scientific Reports*, vol. 11, no. 17590, Sep. 2021.

[3] W. Saad, M. Bennis, and M. Chen, "A Vision of 6G Wireless Systems: Applications, Trends, Technologies, and Open Research Problems," *IEEE Network*, vol. 34, no. 3, pp. 134–142, 2020.

[4] C. B. Barneto, S. D. Liyanaarachchi, T. Riihonen, L. Anttila, and M. Valkama, "Multi-beam Design for Joint Communication and Sensing in 5G New Radio Networks," in *ICC 2020 - 2020 IEEE International Conference on Communications (ICC)*, 2020, pp. 1–6.

[5] A. Liu, Z. Huang, M. Li, et al., "A Survey on Fundamental Limits of Integrated Sensing and Communication," *IEEE Communications Surveys & Tutorials*, vol. 24, no. 2, pp. 994–1034, 2022.

[6] M. Cominelli, F. Gringoli, and R. Lo Cigno, "AntiSense: Standard-compliant CSI obfuscation against unauthorized Wi-Fi sensing," *Elsevier Computer Communications*, vol. 185, pp. 92–103, Mar. 2022.

[7] *Wi-Fi Overview of the 802.11 - Physical Layer and Transmitter Measurements*, `https://www.cnrood.com/en/media/solutions/Wi-Fi_Overview_of_the_802.11_Physical_Layer.pdf`, Copyright © 2013 Tektronix, 2013.

[8] R. Prasad, *OFDM for Wireless Communications Systems*. London, UK: Artech House, 2004.

[9] "Introduction to 802.11ax High-Efficiency Wireless," [Online]. Available: https://www.ni.com/it-it/innovations/white-papers/16/introduction-to-802-11ax-high-efficiency-wireless.html.

[10] E. Khorov, A. Kiryanov, A. Lyakhov, and G. Bianchi, "A Tutorial on IEEE 802.11ax High Efficiency WLANs," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 1, pp. 197–216, 2019. [Online]. Available: https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=8468986.

[11] L. Guo, L. Wang, C. Lin, et al., "Wiar: A Public Dataset for Wifi-Based Activity Recognition," *IEEE Access*, vol. 7, pp. 154 935–154 945, Oct. 2019.

[12] F. Gringoli, M. Schulz, J. Link, and M. Hollick, "Free Your CSI: A Channel State Information Extraction Platform For Modern Wi-Fi Chipsets," in *13th ACM Int. Workshop on Wireless Network Testbeds, Experimental Evaluation and Characterization (WiNTECH '19)*, Los Cabos, Mexico, Oct. 2019, pp. 21–28.

[13] M. Schulz, D. Wegemer, and M. Hollick. "Nexmon: The C-based Firmware Patching Framework." (May 2017), [Online]. Available: https://nexmon.org.

[14] L. Ghiro, M. Cominelli, F. Gringoli, and R. Lo Cigno, "On the Implementation of Location Obfuscation in openwifi and Its Performance," 2022.

[15] M. Cominelli, F. Gringoli, and R. Lo Cigno, "Non Intrusive Wi-Fi CSI Obfuscation Against Active Localization Attacks," in *16th IFIP/IEEE Conf. on Wireless On demand Network Systems and Services (WONS)*, Klosters, Switzerland, Mar. 2021, pp. 87–94.

[16] M. Cominelli, F. Kosterhon, F. Gringoli, R. Lo Cigno, and A. Asadi, "An Experimental Study of CSI Management to Preserve Location Privacy," in *14th ACM Workshop on Wireless Network Testbeds, Experimental evaluation & CHaracterization (WiNTECH)*, London, UK, Sep. 2020, pp. 1–8.

[17] ——, "IEEE 802.11 CSI randomization to preserve location privacy: An empirical evaluation in different scenarios," *Elsevier Computer Networks*, vol. 191, no. 22, p. 107 970, May 2021.

[18] X. Wang, L. Gao, S. Mao, and S. Pandey, "CSI-based Fingerprinting for Indoor Localization: A Deep Learning Approach," *Trans. Veh. Technol.*, vol. 66, no. 1, pp. 763–776, Jan. 2017.

[19] T. Cokelaer, *fitter*, `https://github.com/cokelaer/fitter`, Copyright © 2019-2022, Thomas Cokelaer, 2014.

[20] E. Tonini, *Thesis Source Code - GitHub*, `https://github.com/eletoo/tesi`, 2022.

A Python Documentation

The current chapter contains the documentation describing the code used to elaborate the sampled datasets.

As shown in Listing A.1 the script retrieves the desired file from the current working directory where it is supposed to be located and reads its content.

The file contains multiple lines made of comma-separated values (`csv` format), each of them representing a complex number as a string. Essentially, the file contains an $m \times n$ table having as many columns as the studied sub-carriers (n equals 256 if we refer to an 80 MHz bandwidth, 128 if we work on a 40 MHz bandwidth, or 64 if there is a 20 MHz bandwidth) and as many rows (n) as the packets that have been sampled on each sub-carrier.

The code acts on two versions of the data: on the one hand, the data is processed as it is, after having undergone some formatting (needed for Python to parse the string values contained in the `csv` file correctly as complex numbers). On the other hand, some more processing is done on the increments calculated on the original dataset: considering that each column of the above-mentioned table corresponds to a sub-carrier, increments are calculated between the magnitudes of consecutive packets sent on the same sub-carrier by subtracting the values of the previous row to those of the current row.

The code shown in Listing A.1 references some functions that are defined in the following Python files (in alphabetical order):

- `artificial_trace_processor.py`: generates and processes artificial traces
- `autocorrelation_plotter.py`: plots the graphs showing the auto-correlation

process for each sub-carrier

- `best_fits_param_calculator.py`: the file contains functions that are used to fit the five distributions that best fit the merged increments onto each sub-carrier and to calculate and plot parameters of such distributions

- `fitting_by_sc.py`: the file contains functions that are used to find the best-fitting distribution for each sub-carrier and their increments and save both outputs in dedicated files

- `histograms_plotter.py`: plots the histograms showing the relative frequency of the magnitude for each sub-carrier (see Section A.1 for further details)

- `increments_plotter.py`: plots an increment/frequency histogram for each sub-carrier. The increments are calculated between the amplitudes of consecutive packets sent on the same sub-carrier. This implies that, as for the other plots, a histogram is generated for each used sub-carrier

- `merged_plotter.py`: elaborates the merged increments (i.e. by considering all the increments as if they belonged to a single sub-carrier)

- `parameters_calculator.py`: the file contains the functions that are used to calculate variance, skewness, and kurtosis for each sub-carrier and the distribution of the increments of each sub-carrier

- `std_deviation_and_kurtosis_plotter.py`: plots standard deviation and kurtosis calculated on the increments of each sub-carrier

- `time_evolution_plotter.py`: the file contains a function that is used to plot graphs representing the evolution of the signal magnitude over time for each sub-carrier

It is worth mentioning that the functions used for plotting only plot the graphs relative to sub-carriers that are used for modulation. The unused sub-carriers are listed either in the file named `unnecessaryPlots` or in an array.

```
import [...]

if __name__ == '__main__':

    ########## INFORMATION SETUP ##########
    csi_name = 'csi.csv'  # file containing the data to be processed
    specific_path = "csi"  # folder path where to save the output of the code, can be an empty string
    bandwidth = 80  # channel bandwidth: 20, 40, 80 MHz
    ########################################

    path = os.path.join(os.getcwd(), csi_name)

    if bandwidth == 80:
        colnames = ["SC" + str(i) for i in range(0, 256)]
        df = pd.read_csv(path, names=colnames, header=None)
    elif bandwidth == 40:
        colnames = ["SC" + str(i) for i in range(0, 128)]
        df = pd.read_csv(path, names=colnames, header=None)
    elif bandwidth == 20:
        colnames = ["SC" + str(i) for i in range(0, 64)]
        df = pd.read_csv(path, header=None)
        df = df.transpose()
        df.columns = colnames

    if bandwidth == 80:
        with open(os.path.join(os.getcwd(), "unnecessaryPlots")) as f:
            unnecessary_plots = f.read().splitlines()
    elif bandwidth == 40:
        unnecessary_plots = []
    elif bandwidth == 20:
        unnecessary_plots = ['SC0', ...]

    for title in df:
        if title in unnecessary_plots:
            del df[title]
        else:
            df[title] = pd.DataFrame(abs(complex(value.replace(" ", "").replace("i", "j"))) for value in df[title])

    response = input("Plot magnitude/relative frequency histogram for each sub-carrier? [Y/n]")
    if response.lower() == "y" or response == '':
        batch_size = len(df)
        for x in reversed(range(1, len(df))):
            if len(df) % x == 0:
```

```
                batch_size = x
                break
        for title in df:
            plot_histogram_for_sc(title, df, batch_size, path=specific_path)
    if response.lower() == "n":
        pass

    response = input("Plot evolution in time for each sub-carrier? [Y/n]")
    if response.lower() == "y" or response == '':
        plot_time_evolution_for_sc(df, path=specific_path)
    if response.lower() == "n":
        pass

    response = input("Plot increment/frequency histogram for each sub-carrier?
         [Y/n]")
    if response.lower() == "y" or response == '':
        plot_increments_for_sc(df, path=specific_path)
    if response.lower() == "n":
        pass

    response = input("Plot auto-correlation function for each sub-carrier? [Y/
        n]")
    if response.lower() == "y" or response == '':
        plot_autocorrelation(df, path=specific_path)
    if response.lower() == "n":
        pass

    distributions = {"norm": s.norm, ...}

    response = input("Fit distributions on data and increments? [Y/n]")
    if response.lower() == "y" or response == '':
        fit_data_by_sc(df, distributions, path=specific_path)
    if response.lower() == "n":
        pass

    response = input("Plot and fit merged data and their increments? [Y/n]")
    if response.lower() == "y" or response == '':
        merged_plotter.plot_merged_data(df, distributions, path=specific_path)
    if response.lower() == "n":
        pass

    response = input("Calculate variance, skewness and kurtosis for each sub-
        carrier and for their increments? [Y/n]")
    if response.lower() == "y" or response == '':
        parameters_calculator.calculate_params(df, path=specific_path)
    if response.lower() == "n":
        pass

    response = input("Plot standard deviation and kurtosis for the increments?
         [Y/n]")
    if response.lower() == "y" or response == '':
```

```
91          std_deviation_and_kurtosis_plotter.plot_std_dev_and_kurtosis(df, path=
                specific_path)
92      if response.lower() == "n":
93          pass
94
95      if not os.path.exists(os.path.join(specific_path, 'merged_plot', 'Best
            five distributions fitting Increments of Merged Data.csv')):
96          merged_plotter.plot_merged_data(df, distributions, path=specific_path)
97
98      file = open(os.path.join(os.getcwd(), specific_path, 'merged_plot', 'Best
            five distributions fitting Increments of Merged Data.csv'), "r")
99      f = pd.read_csv(file, sep='\t', header=None)
100     best_dists = f.iloc[:, 0].drop(0)
101     best_distributions = {}
102
103     print("\nThe distributions that best fit the merged increments are: ")
104     for dist in best_dists:
105         print("-> " + str(dist))
106         best_distributions[dist] = distributions[dist]
107
108     response = input("Calculate and plot their parameters for each sub-carrier
            ? [Y/n]")
109     if response.lower() == "y" or response == '':
110         best_fits_param_calculator.calculate_best_params(df,
                best_distributions, path=specific_path)
111     if response.lower() == "n":
112         pass
113
114     response = input("Find distribution that best fits the increments of each
            sub-carrier? [Y/n]")
115     if response.lower() == "y" or response == '':
116         fitting_by_sc.find_best_dist(df.diff().drop(labels=0, axis=0),
                distributions, os.path.join(os.getcwd(), specific_path))
117     if response.lower() == "n":
118         pass
119
120     response = input("Process artificial_increments trace? [Y/n]")
121     if response.lower() == "y" or response == '':
122         artificial_path = os.path.join(os.getcwd(), specific_path, '
                artificial_increments')
123         if not os.path.exists(artificial_path):
124             os.mkdir(os.path.join(os.getcwd(), specific_path, artificial_path)
                    )
125
126         if not os.path.exists(os.path.join(specific_path, '
                normal_distribution_info.csv')):
127             fitting_by_sc.find_best_dist(df.diff().drop(labels=0, axis=0),
                    distributions, os.path.join(os.getcwd(), specific_path))
128         file_name = "normal_distribution_info.csv"
129         data = pd.read_csv(os.path.join(specific_path, file_name), header=None
                )
```

```
130         std_dev = pd.DataFrame(data.iloc[:, 2].map(lambda x: x.rstrip(')')).
                astype(float))
131         artificial_trace_processor.process_artificial_increments(path=
                artificial_path, sub_carriers=df.columns, std_dev=std_dev,
                num_samples=10000)
132     if response.lower() == "n":
133         pass
```

Listing A.1: Main

A.1 histograms_plotter.py

As shown in Listing A.2, the `plot_histogram_for_sc` function uses the other functions defined in its file to plot a histogram representing the frequency of the magnitude of the packets sent on each used sub-carrier.

The function mentioned above also fulfils the purpose of calculating and printing the mean value for each dataset column Every mean value is calculated by taking into consideration only the magnitude of the complex numbers, leaving out their phase.

The third and last purpose of the `plot_histograms_for_sc` function is to divide each column in *batches* of a predefined size — using the `create_batches` function — , have them processed by the `process_batch` function, and finally plot a histogram for each processed column using the `plot` function.

```
1  import [...]
2
3
4  def is_stationary(batch_mean, column_mean):
5      return abs(batch_mean - column_mean) < 0.1 * column_mean
6
7
8  def process_batch(column_mean, batch, size):
9      sum = 0
10     for value in batch:
11         sum += abs(value)
12
13     batch_mean = sum / size
14     print(batch_mean)
15     if not is_stationary(batch_mean, column_mean):
16         print("Non stationary process")
```

```
17          return False
18      else:
19          return True
20
21
22  def create_batches(data, length):
23      return [data[i:i + length] for i in range(0, len(data), length)]
24
25
26  def plot_histogram_for_sc(title, df, size, path: str= ""):
27      if path != "" and not os.path.exists(path):
28          os.mkdir(path)
29
30      col = df[title]
31
32      print(title)
33      column_mean = float(col.mean())
34      print("Column mean:", column_mean)
35      for batch in create_batches(df, size):
36          process_batch(column_mean, batch[title], size)
37      plot(col, column_mean, title, 'histograms', path)
38
39
40  def plot(c: pandas.DataFrame, column_mean: float, title: str, dir_name: str,
        path: str):
41      c = c - column_mean
42      c.hist(bins=100, density=True)
43      pl.xlabel('Magnitude')
44      pl.ylabel('Relative frequency')
45      pl.title(title)
46      pl.xlim(-150, 150)
47      pl.rcParams.update(
48          {'axes.titlesize': 'large', 'axes.labelsize': 'large', 'xtick.
                labelsize': 'large', 'ytick.labelsize': 'large'})
49      if not os.path.exists(os.path.join(path, "histograms")):
50          os.mkdir(os.path.join(path, "histograms"))
51      pl.savefig(os.path.join(os.getcwd(), path, dir_name, 'figure' + str(title)
             + '.pdf'))
52      pl.close()
```

Listing A.2: `histograms_plotter.py` file

The `process_batch` function is used to calculate and print the mean value for the single batch and to verify whether the behaviour of the sub-carrier can be considered stationary by calling the `is_stationary` function.

The `plot` function is used to set the desired title, labels, limits, and the number of bins to obtain the required histogram and save it in the specified

directory.

The analysed data are shown using a histogram because this representation allows a better and easier interpretation as well as the immediate display of the distribution derived from the analysis.

A.2 Artificial Traces Manipulation

To generate and process artificial traces, we created dedicated functions that perform the same manipulations as those performed on real data on artificial data as well.

The used script is shown in Listing A.3.

```
1  import os
2  import [...]
3
4  def process_artificial_increments(real_increments, path, sub_carriers, std_dev
       , num_samples=1000):
5      new_data = generate_artificial_increments(sub_carriers=sub_carriers,
           std_dev=std_dev, num_samples=num_samples)
6
7      if not os.path.exists(os.path.join(path, "increments_hist")):
8          os.mkdir(os.path.join(path, "increments_hist"))
9
10     for title in sub_carriers:
11         histograms_plotter.plot_histogram_for_sc(title, new_data, num_samples,
               path)
12     time_evolution_plotter.plot_time_evolution_for_sc(new_data, path=path)
13     increments_plotter.plot_superimposed_increments(real_increments, new_data,
           path=path)
14     autocorrelation_plotter.plot_autocorrelation(new_data, path=path)
15
16
17 def generate_artificial_increments(sub_carriers, std_dev, num_samples=1000):
18     new_data = pd.DataFrame()
19     i = 0
20     for title in sub_carriers:
21         new_data[title] = np.random.normal(0, std_dev[2][i], num_samples)
22         new_data[title] = new_data[title].cumsum()
23         i += 1
24     return new_data
```

Listing A.3: `artificial_trace_processor.py` file

A.3 Full Source Code

The full version of the code that was written to fulfil the goals of this report can be found on GitHub [20] together with its complete documentation and the output produced by the different scripts stored in separate folders.

Ringraziamenti

Un sentito ringraziamento al Professor Lo Cigno che in questi mesi di lavoro ha saputo guidarmi e motivarmi nella ricerca e nella stesura dell'elaborato, consigliandomi e supportandomi sempre in modo tempestivo.

Un ringraziamento al Dottor Ghiro, i cui suggerimenti mi hanno permesso di portare avanti l'aspetto "algoritmico" di questa ricerca in modo efficiente.

Sono loro grata per avermi mostrato come mantenere sempre vivo l'interesse e la curiosità nei confronti di una disciplina, tramite la passione che sempre mostrano per il campo di ricerca in cui lavorano.

Un grazie va anche alla mia famiglia per essere stata al mio fianco durante questo percorso triennale, spronandomi nell'impegno, gioendo con me dei successi e mai dubitando delle mie capacità.

Ringrazio, infine, chi mi ha insegnato a credere davvero in me stessa, a raccontarmi, a essere curiosa, a mettere in pratica quanto apprendo e soprattutto chi, con pazienza, mi ha insegnato che gli obiettivi apparentemente irraggiungibili si possono invece perseguire più facilmente se non si procede da soli.

Di nuovo, grazie
Elena