Synchronization and Wavelet Compression for Background Interference Classification in Wireless Environments

by

Aikaterini Vlachaki

A thesis submitted in partial fulfillment of the requirements for the degree of

Master of Science

Department of Computing Science

University of Alberta

Abstract

Dense wireless deployment environments are increasingly facing Radio Frequency (RF) spectrum congestion and increased levels of interference. Addressing the interference will require a distributed decision-making application on wireless nodes, that characterize the state of the channel with respect to the presence (or not) of interference, allowing the nodes to adopt mitigation strategies.

In this thesis, through a study of empirical data, we examine if each node was to separately characterize the state of the channel and then form consensus with other nodes, whether this consensus correctly reflects the similarity of the per-node sensed background interference. Towards this goal we develop a distributed synchronization scheme that reduces the inaccuracy inherent in distributed data sampling in local node clocks.

As an alternative to consensus of per-node characterization we also examine the effectiveness of the Discrete Wavelet Transform (DWT) to communicate to other nodes the state of the channel, as sampled by a node, in a compressed, denoised form. Our early results show that wavelet compression of the sampled time series may produce significant data volume savings, to allow the full time series to be communicated e.g. to a cloud infrastructure where large scale planning of spectrum usage can take place.

To my parents.

A man is truly ethical only when he obeys the compulsion to help all life which he is able to assist, and shrinks from injuring anything that lives.

– Albert Schweitzer

Acknowledgements

I will always be thankful to my supervisors, Dr. Ioanis Nikolaidis and Dr. Janelle Harms, for giving me the opportunity to work with them, giving me the chance to explore a whole new world and for their valuable guidance and support.

I would also like to express my gratitude and love to my family for always being there for me. A thank you to Eva who helped me see the light again.

Lastly, I would like to thank the Alberta Innovates Technology Futures for partially funding this study through a Graduate Student Scholarship.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Today's landscape of wireless networks in residential and, in many cases, in enterprise settings, is characterized by diversity and elevated expectations of efficiency and flexibility. It is now commonplace for wireless networks of various technologies to co-exist in the same space. Conceived and deployed separately from each other, the deployed networks result in a challenging operating environment. The resulting, so called cross-technology interference (CTI), gets further compounded by interference originating from non-communicating devices that exist in abundance in today's environments (microwave ovens, lighting equipment, electrical motors, etc.). The combined impact is poor performance which is only expected to get worse as two technological trends continue: (a) ever-increasing density of deployments, such as the ones heralded for supporting the Internet of Things (IoT) [11, 50], and (b) ever-decreasing operating power, meant to extend autonomy for battery-powered devices but at the same time jeopardizing the possibility for error-free reception in dense environments, because of poor Signal-to-Noise-Interference (SINR) at the receivers, which is crucially linked to the ability of receivers to receive error-free data.

Yet, all of today's wireless networks interfaces, in one way or another, connect to the Internet, i.e., to a wired infrastructure. We speculate that this common denominator, i.e., the access to a common wired backbone network, and hence to services residing there, might be sufficient to improve the performance of co-existing wireless networks via a dynamic capacity management. Residential network deployments lack any capacity planning, which is a characteristic available only to a limited degree in some high-end enterprise networking products [17]. The spectrum utilization of the shared Industrial, Scientific and Medical (ISM) bands such as the 2.4 GHz WiFi frequencies, is continuously changing with new technologies added every day, making a dynamic wireless capacity management service among heterogeneous devices a necessity. A dynamic spectrum management application aims to improve the capacity of the wireless network, to efficiently and fairly share the spectrum and to support high Quality of Service (QoS).

At a very high level of abstraction, spectrum sensing and decision making algorithms can run on the cloud, which can then inform the local user equipment of actions to take. An example is the control of multiple access points (APs) in a neighborhood, subject to the channels and power used by other APs, as well as the existence of interference in certain geographical areas. We are motivated by the fact that a modicum of adaptivity and modifiability already exists in legacy protocols, such as 802.11 (Wi-Fi), and an increasing number of access points exhibit smart software-controlled behavior. For example, the cloud-based control could inform the APs which channels, and what power, each should use.

We are inspired by the traditional cellular capacity and resource allocation schemes [32] but we speculate how, instead of being an exceptional one-time design, it can be made in a continuous control and refinement process, in particular given the dynamics of the residential wireless environment.

1.1 System Characteristics

1.1.1 Prevalence of ISM bands

Despite the considerable attention paid to cognitive wireless networks [2], in which a transceiver can intelligently detect 'spectrum holes' by sensing its environment and move into vacant channels while avoiding occupied ones, the immediately pressing needs for wireless co-existence of multiple devices is in ISM bands. The ISM bands consist of the 900, 2400 and 5000 MHz frequency bands and are used for unlicensed Industrial, Scientific and Medical (ISM) applications. Since they do not require licensing, they have been popular in the wireless industry. These bands are used for consumer and commercial WiFi and WLAN applications as well as for commercial Radio Frequency Identification (RFID) applications [34]. In ISM bands, there exists no distinction between primary ("licensed") and secondary users, thus creating more options for network control. A test case for ISM band co-existence is the co-existence in the 2.4 GHz ISM band, where one of the main concerns, given their prevalence, is the performance attained by 802.11 (Wi-Fi) devices. Nevertheless, several sub-GHz ISM devices operating in 433 MHz and 866 MHz bands (remote controls, sensor networking, alarms, movement detectors) [12] exist and are extensively used in IoT applications, especially exploring the ability to reach longer distance yet operating at very low power. Additionally, the 5 GHz ISM band, while not yet crowded, will eventually become crowded as well, as more applications are expected to migrate in the 5 GHz ISM band in search of less interference. We therefore argue that *Radio Frequency (RF) co-existence in the ISM bands is of immediate concern*.

1.1.2 Cross-technology interference

Secondly, wireless communication standards are typically conceived, designed, and implemented, independently of other wireless protocols. For example, what constitutes a "channel" for one protocol can be completely different from what is a channel in another one, or for what the power and timing relation between transmissions might be. The exception is the purposeful backward compatibility when a standard evolves, albeit still the entire family of protocols for one standard tends to ignore other protocol families. Interestingly, the ISM bands have given the opportunity for completely proprietary protocols to emerge as well. We use the term cross-technology interference (CTI), as adopted by Hithnawi et al. [28] to express the impact the operation of the protocol has, seen as "interference", on another protocol operating in the same location.

These result in wireless networks with considerable background interference that can impair a wireless network's or node's performance. Sources of wireless interference can be other wireless networks or devices that operate in the same frequency. Examples of sources of exogenous interference are: microwave ovens, Bluetooth devices, wireless cameras, baby monitors, or a neighbor's Wi-Fi device, poorly shielded cabling or power lines, railway tracks and power stations. Physical barriers, such as construction materials, can also affect the transmission of wireless signals whose power is not strong enough to overcome them.

1.1.3 Manifestations of interference

Generally, the RF front-end (the analog-to-digital or RF-to-baseband portion of a receiver [35]) and the associated signal processing chain, that is used in signal processing applications, is highly specialized to (and in this sense efficient for) the needs of the particular protocol. We will call such RF designs "monolithic" as opposed to flexible Software Defined Radio (SDR) designs, which implement components that have been typically implemented in hardware (e.g. filters, modulators/demodulators) in software on a computer or an embedded system[1]. However, some degree of agility is still possible in monolithic designs (channel selection, transmit power control, etc.). Correspondingly, the impact of another protocol's transmissions manifest themselves as interference leading to either a poor Signal to Noise-plus-Interference Ratio (SINR) figure, or simply as indication that the medium is busy. SINR is a measure of the the quality of wireless connections. It is essentially defined as the ratio of signal power to the background noise/interference power[2]. From the viewpoint of a legacy, monolithic, RF design, there exists no real difference between a source of interference due to another protocol's operation or from non-communication entities (microwave ovens, fluorescent light ballasts, internal combustion engine sparking, etc.).

[1]https://en.wikipedia.org/wiki/Software-defined_radio
[2]https://en.wikipedia.org/wiki/Signal-to-interference-plus-noise_ratio

1.2 Thesis Overview & Contributions

Our research is focused on how wireless nodes can observe wireless interference and how they can communicate the state of their channel to other nodes or central servers in order to mitigate interference.

In general, the presented work is related to the area of cognitive networking. However, it is the restricted nature of the abilities of the nodes that drive the presented research. Namely, we assume that the nodes have only a single means, a signal strength indicator, called Received Signal Strength Indicator (RSSI), for sensing the channel for purposes of analyzing any interference patterns. No special support from the physical layer is assumed or required.

Specifically, we study interference in urban environments to test whether the often–assumed strategy of deriving a distributed consensus across nodes about the nature of the channel is a strategy that reflects the reality of the channel. In consensus strategies, each node independently classifies the channel based on its own measurements and provides the result of its classification to the rest of the nodes. Subsequently, and depending on the formation of consensus, decisions about the use (or not) and the exact technique to access the channel can take place. We are not interested here in the decisions taken after the consensus is reached but on whether the channel indeed behaves the same way from the viewpoint of the nodes that determined the channel behavior to belong to the same class. For example, if a node detects a periodic spike of interference sufficient to classify it as a channel with a periodic interferer, it might agree with the class identified on the same channel by another node, but there is no guarantee that the two nodes sense the same periodic interferer.

The contributions of this thesis are:

1. We develop a scheme to facilitate a comparison of the background interference as seen by different nodes. We develop a technique to correct the lack of synchronization across the samples collected by the different distributed nodes.

2. We study how the agreement of nodes, e.g., via consensus, on the class

of a channel can be linked to the cross-correlation statistic and to what extent [57].

3. We investigate the effectiveness of the Discrete Wavelet Transform (DWT) to communicate to other nodes the state of the channel, as sampled by a node, in a compressed, denoised form. This is important for efficiently communicating information to other nodes or centralized servers so that they can make decisions regarding channel allocation [58].

1.3 Thesis Outline

Chapter 2 provides a brief survey of fields that come together under the same ambitious plan, the "Dynamic Wireless Capacity Management" and honing down eventually to Chapters 3 and 4, which specifically deal with channel classification agreement in urban WSNs and the wavelet–based representation of interference, respectively. The final, concluding, section is a compilation of open research opportunities that can be immediately pursued.

Chapter 2

Literature Review

In this Chapter, we discuss the related research by starting with an overview of the channel allocation schemes in dense urban environments. Then, we review different approaches towards obtaining information about the channel state and we conclude with an overview of the Discrete Wavelet Transform (DWT), some of its application, as well as mother wavelet and thresholding selection techniques.

2.1 Channel Allocation

The literature on wireless resource management is vast. We note, for example, the channel assignment literature for cellular networks, e.g., [32] as relevant, but one that follows the cellular network model of operation, i.e., single (or few) providers, no outside interferers, known locations (hence one-time, or infrequent, computation of allocation plans), etc. We need to abandon these assumptions in order to capture the essence of residential ISM, and in particular dense urban, environments. A starting point can thus be seen in the channel allocation schemes specific to WiFi, such as those surveyed by Chieochan et al. [15], which take into account the characteristic irregularities of AP coverage and the variety of traffic and QoS demands in different areas. They study schemes aimed for centrally managed environments where interference constitutes the metric of interest (which translates to a capacity measure) as well as schemes for uncoordinated environments.

The control of APs suggests a need for an Inter-AP Protocol (IAPP), a protocol essential for the cooperation and communication between APs. Mahonen et al. have described a scheme using the IAPP protocol [40]. The channel allocation is expressed as a classical vertex coloring approach, DSATUR, where APs are modeled as vertices of an interference graph. The ingredient of their scheme which we consider essential in a cloud-supported allocation system is the fact that their proposal for channel allocation is dynamic and cooperative. Every new station that arrives within range automatically becomes part of the interference graph. APs detect other APs by means of hearing their beacons and construct vectors of information consisting of AP MAC addresses, signal-to-noise ratios, and received signal strength [15, 40]. After the identification of their neighbours, the APs can share their knowledge with other APs in the network and the procedure can be repeated whenever the topology changes.

The lessons learned from 802.11 (Wi-Fi) can be transformed to a broader class of wireless networks. The underlying coloring problems, and any other heavily computational optimization problems, can be relegated to the cloud. Additionally, there are a number of extensions that could allow the easier transformation of 802.11 (Wi-Fi) schemes to more general ones. First, the AP-centric view has been exchanged for a client-/peer-specific view. For example, Mishra et al. [43], introduce load balancing into the channel assignment scheme where interference is examined from the clients' point of view and it is claimed that even hidden (from the APs view) interference is captured and accounted for. A client is considered to suffer from channel conflict in two cases. In the first, the interference seen by a client comes from APs located within a communication range of the client of interest and in the second from APs or wireless clients (in neighboring BSSs), located within a one-hop distance of the AP-client link of interest [15, 43]. Their client-centric algorithm is based on conflict set coloring. In this algorithm, the goal is to distribute the clients to APs in a way that the conflict is minimized while the load is balanced.

Following on similar logic, Chen at al. [14] present, among other algorithms, Local-Coord. Local-Coord coordinates the APs in a network aiming to minimize the interference as seen by both APs and wireless clients. This is work that uses in-situ, real time interference power measurements at clients and/or APs, on all the frequency channels. Local-Coord promises increased throughput and mitigation of interference by performing frequency allocation irrespective of network topology, AP activity level, number of APs, rogue interferers, or available channels. The weighted interference constitutes the cost function in this approach and it is applied for every BSS. Metrics like the average traffic volume and average RSSI of the clients within a BSS can be used as weights, thus guiding the protocol to tolerate different metrics accordingly. Correspondingly, a proposed global coordination scheme, Global-Coord, applied centrally, performs overall coordination and spectrum allocation of a network only if a new channel assignment results in lower co-channel weighted interference.

Leung and Kim propose MinMax [39], focusing on interactions among APs and aiming to minimize the maximum effective channel utilization at the network's bottleneck AP, so that its throughput is improved and the overall network escapes congestion. The effective channel utilization is defined as the time a channel assigned to an AP is used for transmission, or is sensed busy because of interference from its co-channel neighbors. More specifically, initially random channel assignment is performed to all the APs in the network. Next, the bottleneck AP's interfering neighbors channels are readjusted so that the effective channel utilization of the bottleneck AP is minimized [15, 39].

Yu et al. propose a dynamic radio resource management scheme in [59], where again the maximum effective channel utilization at the network's bottleneck AP, is to be minimized but without interactions among APs. In their work, the channel utilization is determined by a real-time algorithm that estimates the number of active stations from an AP's point of view. The real time consideration of active stations before each channel assignment, as well as a post channel assignment QoS check reinforces the dynamic nature of this approach. However, this scheme does not scale to large networks [15].

An added degree of freedom arises from power management. Power management of APs accounting for traffic load distribution and spectrum allocation is proposed in [27] by considering the notion of a "bottleneck" AP in a network and suggest its transmission power readjustment (reduction) takes place together with channel assignment in order to reduce the total interference over the network. The power reduction only applies to beacon packets and not data packets, so that clients that can no longer be supported turn to another less congested AP. In this work, the total data rate required by an AP's clients over the AP's available bandwidth forms the, so–called, congestion indicator [27, 15]. Their proposed optimization algorithm is claimed to be able to redistribute the load in a network, reduce the AP congestion and finally perform spectrum allocation so that the interference is minimized.

Concluding, we also note that whereas the coloring-based channel allocation problem is a useful abstraction, there has been evidence that a potential overlap of the allocated channels (hence abandoning a strict vertex coloring interpretation) is not as harmful as suggested by most of the literature, while it is almost unavoidable in high density network deployments anyway. Specifically, Mishra et al. [44], by explicitly modelling an interference factor (I-factor), derive capacity improvements.

2.2 Characterizing & (Re-)Acting

2.2.1 Channel Characterization

A crucial element towards performing dynamic capacity management in the face of changing conditions, such as interference, is the collection of suitable measurements. We can collectively call the various facets of this problem the Channel State Information (CSI) problem, but recognize that, in the same network, different devices can acquire completely different types of CSI metrics and for different bandwidths. Our tacit assumption is that similar measurements along with the user behavior predictability imply that similar actions need to be taken on a regular, e.g., daily, basis.

A type of detection can be performed with devices equipped with SDR capabilities using simple measurements, is energy-based signal detection [38], which involves a low computational overhead but generally performs poorly in low SINR environments. Cyclostationarity [25], a feature-based signal detection approach that takes advantage of the presence of certain periodicities in a signal, on the other hand, has been a potent technique but at a high computational cost. Clearly, if the sampled channel data can be shipped quickly and in large volumes to a cloud-based detection algorithm, cyclostationarity computation is a viable option. Nevertheless, cyclostationarity is not exhibited by non-communication (hence not modulated) interference sources. There, a form of feature detection can be used instead. An example are on-line classification mechanisms of interference such as the ones proposed by Boers et al. [7].

We anticipate that the perceived interference is location-specific. Indeed, in this thesis (Chapter 3), we measure the cross-correlation of traces of signals from different nodes and find that nearby nodes are experiencing similar interference. Seen differently, it is not necessary that all nodes in an area, e.g., not all APs, have to be devices with features of an SDR since very similar measurements would end up being collected. However, it is still not known how dense should the spatial sampling be to derive reliable channel state metrics for the various areas of the network. We note that similarities exist with the case of sampling strategies for optimal monitoring [36] as they express the situations where limited sampling resources are to handle a large network. Yet another, natural facet of the same problem is how high the sampling rate should be and how to convey the sampled data for processing in the cloud. In Chapter 4, we find that wavelet compression of the sampled signal trace may produce significant data volume savings [58].

2.2.2 Channel Classification

Researchers studying the impact of external interference in urban environments concentrate on identifying and classifying patterns of noise and interference, as well as applications of related classification techniques to cognitive networking.

Lee, Cerpa and Lewis [37] measure noise traces in many different environments in order to propose algorithms to simulate noise and interference. From these traces they observed three main patterns of interference, (i) rapid spikes, (ii) periodic spikes and (iii) noise patterns changing over time.

Boers, Nikolaidis and Gburzynski [9] measured noise and interference in a four-by-four node WSN, at high sample rates. They extracted five dominant noise and interference patterns: (i) quiet, (ii) quiet with spikes, (iii) quiet with rapid spikes, (iv) high and level and the (v) shifting mean pattern. Consequently, they classified them using a Bayesian network classifier. Later, this work was extended by classifying two of the aforementioned patterns locally at each node using single-node decision tree classifiers [7].

In cognitive-networking, known identified patterns can be exploited to coordinate cooperative sensing across the nodes of a WSN. The determined noise and interference patterns for each WSN can be utilized to build a distributed classifier. In such a scheme, the WSN nodes cooperate with each other to reach a consensus on a specific pattern, after a number of iterations, by exchanging and combining their sensing information. This aims to eliminate the impact of deficient individual pattern classifications [3]. The notion of cooperative sensing extends also to multi-hop cases whereby the sensing results of nodes are forwarded over multiple hops in order to improve the classification accuracy.

2.3 Wavelet Transform

Next, we consider the need to represent RSSI time series in a concise way, which we do by means of wavelet transforms.

Amara Graps [26] describes the use of wavelets as a whole new perspective in data processing. According to Graps, wavelet analysis enables us to see both the forest and the tree. The primary idea behind wavelet transform is the analysis of features based on different scales or resolutions, allowing big scales to portray gross features and small scales to bring up small features. In other words, long windows are used at low frequencies and short windows for high frequencies [51].

The key feature of the bases in the wavelet transform is the capability to use approximating functions that are contained in finite domains. This means that wavelets are able to be applied in the approximation of data with discontinuities and sharp spikes in contrast to the infinity width of sine and cosine bases.

2.3.1 Discrete Wavelet Transform (DWT)

Rioul et al. [51] expand on the application of the wavelet transform both in continuous and discrete signals defined as Continuous Wavelet Transform (CWT) and Discrete Wavelet Transform (DWT), respectively. The connections between wavelet transform and discrete signal processing are primarily established by Daubechies and Mallat [20], [41].

In the discrete wavelet analysis procedure a wavelet prototype function is chosen, called a basis wavelet or mother wavelet. Temporal analysis is performed with a contracted, high-frequency version of the prototype wavelet, while frequency analysis is performed with a dilated, low-frequency version of the same wavelet. Hence, the original signal or function can be represented in terms of the wavelet expansion using coefficients in a linear combination of the wavelet functions.

Within each family of wavelets are wavelet subclasses distinguished by the number of coefficients and by the level of iteration. Wavelets are classified within a family most often by the number of vanishing moments. This is an extra set of mathematical relationships for the coefficients that must be satisfied, and is directly related to the number of coefficients.

Combining the above, if the optimal mother wavelet is chosen and adapted to the examined data and if/when the coefficients are truncated below a threshold, the data is sparsely represented which makes wavelets an excellent tool for data compression [26].

Mallat's Multiresolution Analysis (MRA)

As Singh et al. explain [53], in order to avoid the calculation of coefficients at every scale, a subset of scales and positions based on powers of two is selected. Next, the iterative application of high pass and low pass filters to calculate the wavelet expansion of a given sequence of discrete numbers is introduced as described by Mallat and Rioul et al. [41],[51].

Vetterli in his work [56] discusses the advances in signal processing regarding wavelets. He also explains the use of filter banks as standard signal processing operators.

The DWT of a signal is calculated by passing it through two filters. First the samples are passed through a low pass filter with impulse response g and then through a high-pass filter h. The outputs give the detail and approximation coefficients from the high-pass and the low-pass filter respectively. The two filters are related to each other and they are known as a quadrature mirror filter or a filter bank. The filter outputs are then subsampled by 2. The structure is recursively implemented up to a level.

The wavelet transform computation according to Mallat [41] as descibed by Singh et al. [53] requires a pair of filters. The first would be the filter to calculate the wavelet coefficients, whereas, the other one applies the scaling function. This scaling function, implemented with filter coefficients $\{h_k\}$, provides an approximation of the signal via the following equation:

$$W_L(n,j) = \sum_m W_L(m, j-1)h(m-2n) \tag{2.1}$$

The wavelet function gives us detail signals, which are also called high, pass output as given in [41].

$$W_H(n,j) = \sum_m W_L(m, j-1)g(m-2n) \tag{2.2}$$

Where $W_L(p,q)$ is the pth scaling coefficient at the qth wavelet coefficient at the qth stage, and $h(n), g(n)$ are the filter coefficients corresponding to the scaling (low pass filter) and wavelet (high pass filter) functions, respectively [56].

2.3.2 Wavelet Selection

The mathematical procedure of wavelet analysis is fundamentally based on the selection of a mother wavelet. The results of the decomposition of a signal are basically scaled and shifted versions of the chosen mother wavelet. As a result, different mother wavelets will normally produce different outcomes.

In their work Ngui et al. [48] enlist mother wavelet selection methods based on the similarity between the analyzed signal and the candidate mother wavelet, in contrast to selecting a mother wavelet on its properties alone. They split these methods into two broad categories based on the selection of the mother wavelet, in qualitative or quantitative approaches.

In this work, we concentrate only on quantitative approaches, mentioned but not limited to Ngui's et al. [48] work, in order to select the appropriate method for choosing the mother wavelet for our analysis.

Singh et al. [53] propose a quantitative approach based on the maximum cross correlation coefficient criterion, to select the optimal wavelet basis function in order to denoise ECG signals. The analysis carried out in this work, examines the selection of a basis wavelet filter decomposition, computes the cross correlation coefficient between the electrocardiogram (ECG) signal and selected wavelet filter and finally selects the optimum wavelet filter which maximizes the cross correlation coefficient. Furthermore, in order to compare the behavior of the estimation methods performance criteria, the root mean square error (RMSE), the root means square bias (RMSB), and L1 norm were employed.

Tsui et al. [55] proposed a technique for automatic ultrasound non-destructive Foreign Body (FB)detection and classification. Close examinations of the experimental ultrasound signals indicated slight shifts with respect to each other. The amount of shifts were too small for cross-correlation to detect accurately. However, such small shifts could be translated into large variations by CWT. The authors refer to Shannon entropy [19] and relative entropy [18] as potential basis selection methods and propose to use an information measure as the criterion for their wavelet basis selection. However, from the information measures developed over the years, many required joint, computing expensive statistics. Symmetric divergence or relative entropy, did not and that was the reason it was chosen as the wavelet coefficient similarity measure between classes.

Coifman and Wickerhauser's [19] method was based on an entropy cost function measuring the flatness of the energy distribution of the signal so that minimizing this leads to an efficient representation (or coordinate system) for the signal. Because of this cost function, the best-basis algorithm was good for signal compression but not necessarily good for classification or regression problems as mentioned in Coifman and Saito's [18] research. Coifman and Saito [18] extended Coifman and Wickerhauser's [19] 'best-basis' method using the relative entropy (or regression errors) and showed its efficiency by feeding the most significant coordinates into traditional classifiers (or regression methods) such as Linear Discriminant Analysis (LDA) or a Classification and Regression Trees(CART).

Morsi and El-Hawary [45] studied steady-state power system distorted waveforms and proposed a mother wavelet selection criterion based on the energy of the wavelet coefficients at each level. In their approach, the use of DWT suffers from the problem of spectral leakage which is related to the choice of the wavelet family and the mother wavelet used in the analysis. To achieve accurate measurement of steady-state harmonic distortion while minimizing the spectral leakage, the DWT was used. Specifically, the deviations between the calculated percentage energy of the wavelet coefficients and the estimated percentage energy of the harmonic components for different wavelet families and mother wavelets were evaluated.

In a selective mapping technique for ECG compression, Chompusri et al. [16] present a mother wavelet selection based on results on the comparison of the compression ratio for different mother wavelets. However, it's important to keep in mind that the degree of of compression performance relies both on the characteristics of an ECG signal and the type of mother wavelet.

In our study, since we are concentrating on the compression effect of the DWT on our data, we are combining the steps followed in Singh's et al. [53] work, along with our version of the compression ratio inspired by the work of Chompursi et al. [16].

2.3.3 Wavelet Thresholding

By principle, the wavelet decomposition uses filters that produce detail and approximation coefficients. The idea behind compressing and denoising data using the wavelet transformation is based on thresholding. Using thresholding techniques, coefficients that correspond to signal details and approach zero can be actually set to zero and therefore be omitted in the application of the inverse wavelet transform in the recovery of the signal.

Wavelet Thresholding Selection Methods

Several thresholding techniques have been studied and many of them are standard in numerical computing environments such as MATLAB. David Donoho has been a leader in the field by firstly suggesting a universal threshold proportional to the sample size M $t_h = \sqrt{2 * log(M)}$ together with Johnstone in their work [23],[21]. They are also behind the Minimax thresholding technique [24] which uses a fixed threshold chosen to yield minimax performance for mean square error against an ideal procedure. The minimax principle is used in statistics in order to design estimators. Since the denoised signal can be assimilated to the estimator of the unknown regression function, the minimax estimator is the one that realizes the minimum of the maximum mean square error obtained for the worst function in a given set' [1]. Stein's Unbiased Risk Estimate or SURE [54] is a third adaptive threshold selection method. In Stein's work an unbiased estimate of risk is calculated for an arbitrary estimate,using mean squared error as loss.

In an attempt to combine the Minimax and SURE thresholding methods, a heuristic SURE thresholding method [13], a fixed form threshold is used when the signal to noise ratio is very small resulting in a very noisy SURE estimate[2].

In our work, all the above thresholding methods were implemented in MATLAB, using both 'hard' and 'soft' thresholding [22]. 'Hard' thresholding is harsher as for an input signal X it returns the thresholded signal $Y = X$, for $|X| > T$ and a threshold value T[3] . In the second case, the thresholded signal will be $Y = sign(X) * (|X| - T)_+$, for $(x)_+ = 0$ if $x < 0$ and $(x)_+ = x$, if $x \geq 0$ and a threshold value T.

[1]http://www.mathworks.com/help/wavelet/ref/thselect.html

[2]http://www.mathworks.com/help/wavelet/ref/thselect.html

[3]http://www.mathworks.com/help/wavelet/ref/wthresh.html?refresh=true#zmw57dd0e51987

Chapter 3

Channel Classification Agreement in Urban WSNs

3.1 Introduction

In this chapter, we present the extension of a previous effort of characterizing the background interference in a deployed WSN [9]. Characterizing the interference allows the operation of the WSN to adopt strategies to circumvent the interference and its impact. For example, a Medium Access Control (MAC) protocol that operates around interference patterns was developed in [5].

The work of this chapter is an updated and extended version of our paper in [57]. The model assumed throughout this study is that each node independently decides on what is the nature of the interference via a classification technique (placing it in one of five classes) as outlined in [9] and shown in Figure 3.3. To facilitate a comparison of the background interference as seen by different nodes, we develop a technique to correct the lack of synchronization across the samples collected by the different nodes. The lack of synchronization is caused by the absence of a global clock and the individual node clock drift. The purpose of this section is to study, from collected empirical evidence whether, if, when consensus is reached, it is indeed valid, i.e., it concerns the same interference seen by all the nodes at the same points in time. To this end, we examine whether, consensus exists (via cross-correlation as in our paper [57]) and if the levels of interference are compatible across the nodes, i.e., the small time scale behavior is the same. For example two nodes with a

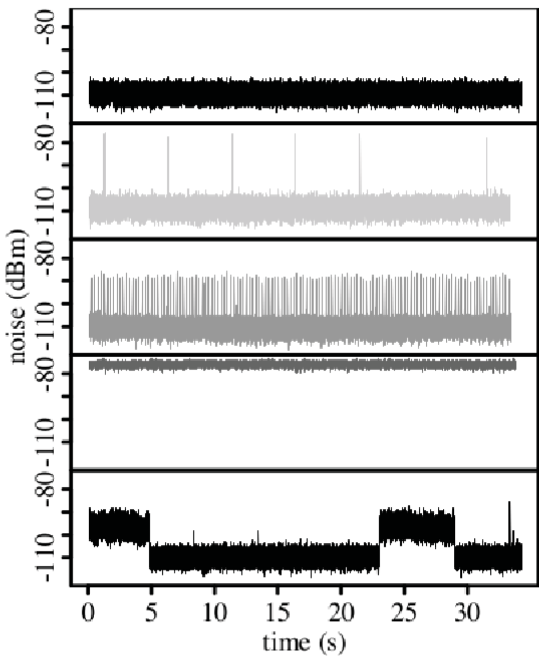


Figure 3.1: Example RSSI traces of interference, as adopted from [9], shown for (a) quiet, (b) quiet-with-spikes, (c) quiet-with-rapid-spikes, (d) high-end-level and (e) shifting-mean channel.

valid consensus characterizing the channel as having periodic spikes may still perceive different noise floors and variance of noise between spikes, making the potential Signal to Noise Ratio (if a transmission were to be attempted) drastically different from the perspective of the two nodes. In short, we are studying whether a simple class-based consensus can be relied upon to represent the common reality across the nodes of the same WSN, confined in the same geographic area.

3.2 WSNs and Interference

It is often asserted that Wireless Sensor Networks (WSNs) will be increasingly deployed in hostile environments. While a hostile environment usually means an environment inhospitable to human presence, in another sense a hostile environment can be one of continuous human presence albeit with adverse impact on the operation of the WSN nodes. Such is the case of urban environments with the multitude of sources of interference, some of which are rather wellunderstood, e.g., other co-located wireless data communication networks, and some that are less so, i.e., electromagnetic interference from nearby operating appliances (lamps, microwaves, etc.), elevators, car engines, etc. It is tempting to lump all such interference into a category that, on the average and across a large number of interferers, would be conveniently modelled as a Gaussian noise source. Unfortunately, empirical evidence collected so far [37, 9] suggests that interference in urban environments does not fit the simplifying Gaussian assumption.

3.3 The Data

We use the RSSI traces collected by Boers et al. [9], across 256 channels spanning ISM and non-ISM bands in an indoor urban environment. We concentrate on the sample-cross correlation for each channel and for every pair of nodes, aiming to quantify and justify the similarity between nodes that have classified the channel as exhibiting the same pattern, as well as to identify disagreements at a microscopic level.

3.3.1 The Data Collection System

The RSSI traces were collected within the first implementation of the Smart Condo at the University of Alberta within the Telus Centre (a medium sized office building) located across from a large residential apartment building [8]. Within the 80 m^2 space of the Smart Condo, WSN nodes were placed in a four-by-four grid with 1.84 m spacing as presented in Figure 3.2. Each node stood 28 cm above floor level. While running the data collection experiments, the room's doors were closed and there was no movement within the room. Additionally, all the measurements were noise measurements, meaning that the sensor nodes did not introduce any transmissions on their own.

The WSN nodes were model EMSPCC11 provided by Olsonet Communications [49] consisting of a TI MSP430F1611 microcontroller and a TI CC1100 transceiver. The transceiver was configured for 38.4 kbit/s using 2-FSK modulation. The nodes ran an operating system named PicOS [1] and a PicOS

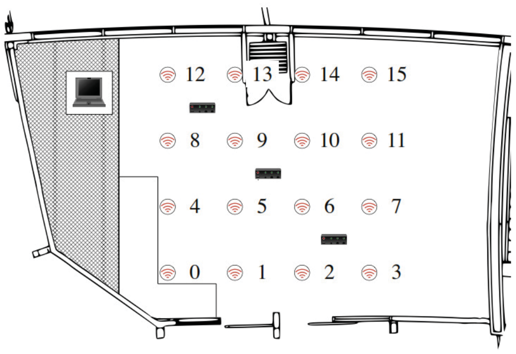


Figure 3.2: The experimental setup within the Smart Condo. The circles represent the WSN nodes. The computer collecting the data was placed outside the grid of WSN nodes (top-left). The figure is adopted from [9].

application collected noise measurements by reading the RSSI value from the CC1100's RSSI register.

The RSSI was measured by each one of the 16 nodes of the WSN for every channel. In total, 256 channels were examined to produce a total of 4096 traces. The configuration of the WSN nodes was at a base frequency of 904 MHz. The channels were spaced 199.9512 kHz apart. Each channel occupies a bandwidth of 101.5625 kHz. Using these settings the nodes were listening on frequencies within and outside the ISM band, from 904 MHz to 928 MHz and 929 MHz to 954 MHz, respectively. For each channel and node combination 175000 successive RSSI samples were collected, representing a duration of 35 seconds. The entire data collection process was completed in approximately 2.5 hours.

3.3.2 Channel Classes

Five dominant noise and interference patterns were encountered from a closer examination and the hand classification of the collected RSSI traces. We repeat here the characteristics of each class:

1. The *quiet* channel, which is characterized by a low maximum.
2. The *quiet-with-spikes* channel is similar to the quiet channel, but it has short-duration impulse-like "spikes" that give it a higher maximum.

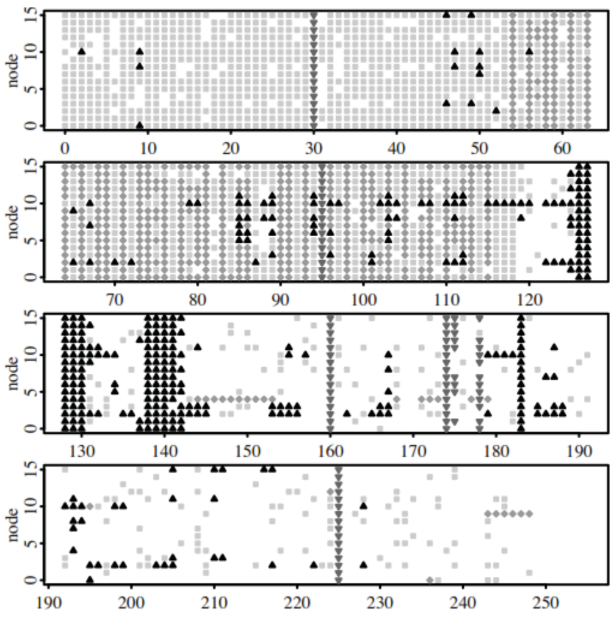


Figure 3.3: Classification of noise and interference traces from 256 channels with 16 nodes per channel, as adopted from [9]. The correspondence between symbol and classification is: (a) no symbol: quiet, (b)■: quiet with spikes, (c)♦: quiet with rapid spikes (d)▼: high- and- level, (e)▲: shifting mean [9].

3. The *quiet-with-rapid-spikes* channel has a higher frequency of spikes than the quiet-with-spikes channel.

4. The *high-and-level* channel exhibits a continuous high and has a high minimum.

5. The *shifting-mean* channel has its RSSI samples distributed bimodally.

A visual classification of the noise traces for each node per channel are presented in Figure 3.3.

3.4 The methodology

3.4.1 Pre–processing

Two significant parameters taken into consideration are the node clock drifts and timestamping of the sampled data, as well as the noisy nature of the RSSI traces themselves.

Data Collection Timestamping

In the work of Boers' et al. [9], the node clock drift during the collection of the RSSI traces was surprisingly high, even over short intervals. Since WSN node clocks cannot be relied upon to provide the correspondence to the natural time, the timestamping was performed with respect to the clock of the personal computer to which the data collection was being streamed (via serial USB connections) from the individual WSN nodes. In other words, the clock of the collecting host was trusted as authoritative. Naturally, this collection at the host was performed for the purposes of the data analysis presented here and is, in principle, absent in a real network. Between two readings from the same node/port several RSSI samples could have been buffered in the meantime. In the case of multiple samples found in the incoming buffer, the reading application would assign those samples equi-spaced timestamps between the current time and the time the buffer was read last. The result is that two readings performed at the same point in natural time from two different nodes may appear with different timestamps. Hence, some means of synchronizing the time series is necessary. Furthermore, the synchronization scheme ought to be able to be used in a completely distributed fashion.

Synchronization Scheme

The synchronization process has the RSSI time series samples of the nodes in a channel as input, and converts it to synchronized RSSI time series.

Specifically, the purpose of synchronization is to produce a sequence of samples $s'(i, j)$ for the data of node i at time j where $j \in \{1, ..., J'\}$. The input is a sequence of samples $s(i, j)$ where $j \in \{1, ..., J\}$, that were acquired by each node i separately from the rest. When example results are presented, they are for the data set described earlier and $J = 175000$. The specific sampling at the J instants is assumed to be at a constant rate, i.e., the (natural) time difference between any two successive sample instants is assumed to be constant and equal to Δt; that is sample i is taken at time $(i-1)\Delta t$ with the first sample assumed to be taken at time 0. However, the sampling is taking place separately by each node, and their clocks do not agree, neither does their notion of Δt. Hence, assuming that the natural time at which the j–th sample of node i was acquired is denoted by $t(i, j)$ then, due to the distributed clocks of nodes i and i', $t(i, j) \neq t(i', j)$, i.e., the natural time when the j–th sample was sampled is different at each node.

The synchronization is expressed as a step of post-processing taking place after each node sampled a set of samples J separately from each other. Clearly, one of the nodes (fastest running clock) will end sampling earlier than the rest. Let us define that "earliest termination" node as f (for "fastest"). That is:

$$f = \arg\min_{i} t(i, J) \tag{3.1}$$

and the corresponding termination time as $e = t(f, J)$.

In the dataset used in this study, the values of $t(i, j)$ come from the timestamps when the (central) data collecting host received the measurement from the corresponding node. Clearly, this is not a strategy that is applicable in a distributed system. In a distributed system, the equivalent function is achieved by having the node that terminates the sampling of J samples earlier than the rest, broadcast to the rest of the nodes a "sampling done" message. The rest of the process is applicable equally to a distributed or a centralized version of the algorithm.

For each node i, separately from the rest, compute the number of samples that were collected up to the early termination.

$$\forall i \neq f : \qquad r_i = \arg\max_{j} t(i, j) < e \tag{3.2}$$

One of the nodes has the least number of samples collected trough that period, i.e., it is the "slowest", and is indicated by z, i.e.,

$$z = \arg\min_{i \neq f} r_i \tag{3.3}$$

and the corresponding number of samples as $J' = r_z$. The essence of the synchronization is that we produce as output $s'(i, j)$ a sequence of fewer than J samples – specifically J' samples that are as many samples as the slowest node, z, sampled within the time that starts at 0 and ends when the fastest node, f, finished sampling. This guarantees that all nodes have at least J' samples for this interval.

It is now possible to construct the new samples under the approximating assumption that the rate of sampling for the synchronized trace in the interval from 0 to e is constant and equal to $\Delta t' = e/J'$. Each node's clock is assumed to have its own rate determined by the number of samples it completed within time 0 to e. That is, the local rate at node i is $\Delta t^{(i)} = e/r_i$. To generate the j-th synchronized sample $s'(i,j)$ where $j \in \{1,...,J'\}$, we note that the time for the synchronized sample ($j\Delta t'$) falls between two samples at node i's clock rate, i.e.,

$$(m-1)\Delta t^{(i)} \leq j\Delta t' \leq m\Delta t^{(i)} \tag{3.4}$$

Of the two samples, one at $(m-1)\Delta t^{(i)}$ and one at $m\Delta t^{(i)}$, we select the one whose timestamp is closer to $m\Delta t^{(i)}$. In other words, if $j\Delta t' - (m-1)\Delta t^{(i)} < m\Delta t^{(i)} - j\Delta t'$ then $s'(i,j) = s(i,m-1)$; otherwise $s'(i,j) = s(i,m)$.

As an illustration, we also provide a very simple example with only four nodes and eight samples. We can see in Figure 5.1 that node C collects the eight samples in 5 seconds, so this will initially constitute the fastest node. We observe that in those five seconds node A collects five samples, node B collects 4 samples and node D collects 4 samples as well. The slowest node, according to the indices can be either node B or D. Assume the tie is broken arbitrarily, then Node B is the slowest node and the rest of the nodes will synchronize according to its number of samples (4). Finally, we end up with four synchronized sequences, with four samples each. Specifically, for node A the 1st, 3rd, 4th and 6th of the samples will be used.For nodes B and D the first four samples will be used and for node C the 2nd, 4th, 6th and 8th samples will be used.

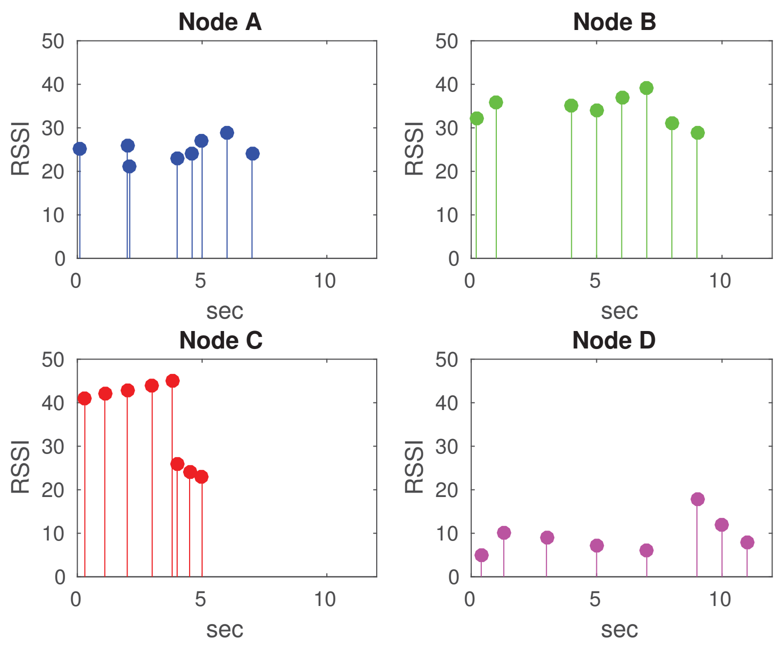


Figure 3.4: Synchronization example, RSSI traces and their timestamps before synchronization.

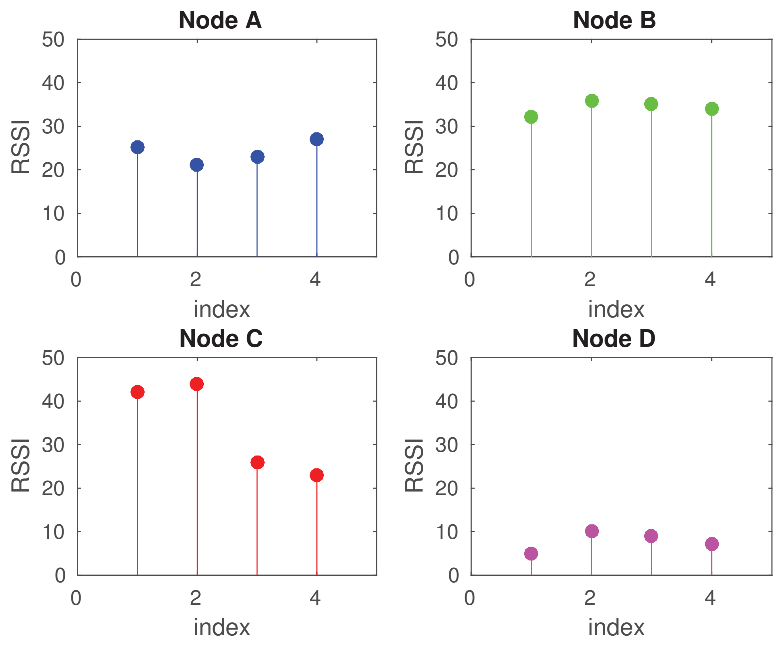


Figure 3.5: Synchronization example, RSSI traces and their after synchronization.

Filtering

The second step before the sample cross-correlation estimation is to apply a low pass filter to the time series. This is done in order to enhance the calculation of the cross-correlation by removing high–frequency noise from the RSSI time series, leaving the characteristic low–frequency shape of each sequence intact. A low pass filter emphasizes the behaviour and the characteristics of the observed patterns by producing a time series where the amplitude of variations at high frequencies is reduced.

Namely, we apply, to each time series, a Butterworth low-pass filter of 4th order with a cut-off frequency of 375 Hz (for a sample rate of 5000 Hz). Its frequency response is presented in Figure 3.6 and the spectrum of a time series before and after the application of this filter is presented in Figure 3.7.

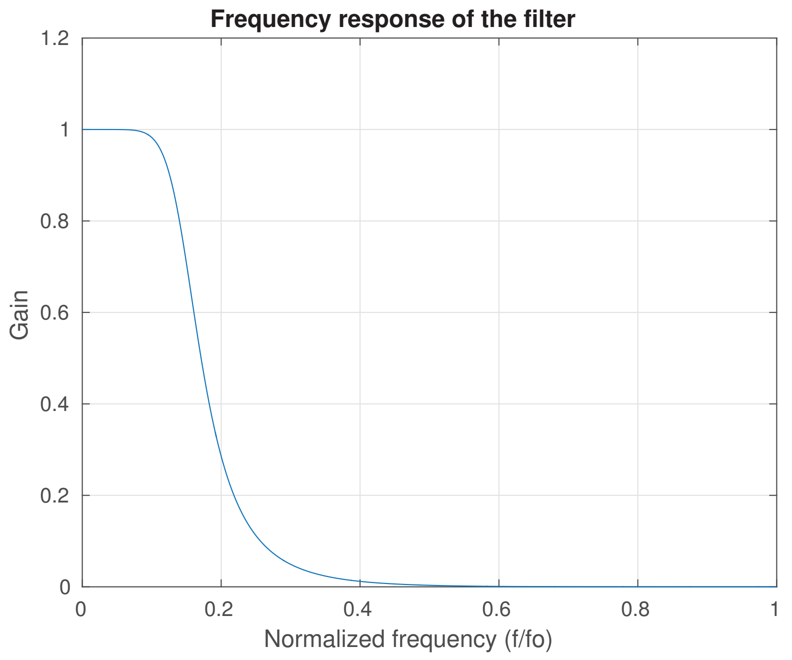


Figure 3.6: The frequency response of the Butterworth filter, of 4th order, with cutoff frequency 375 Hz.

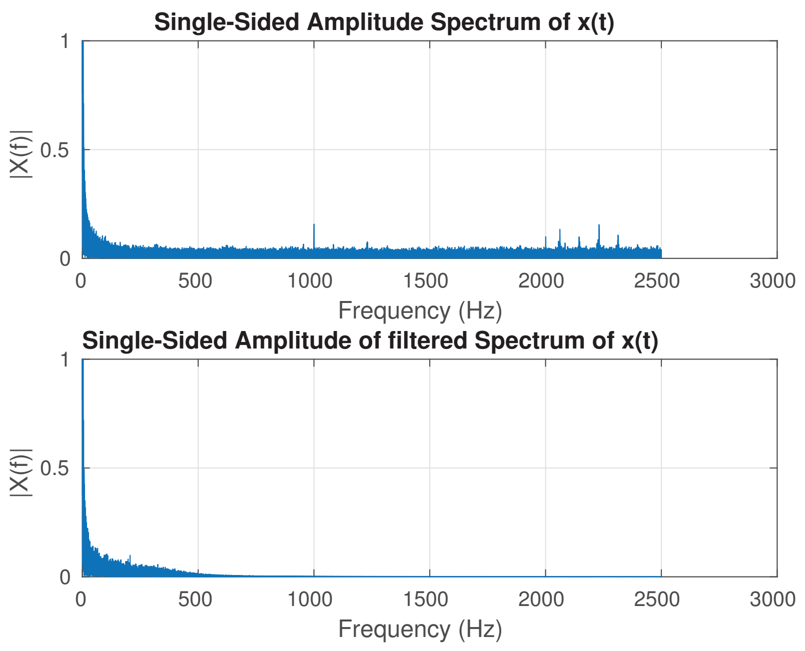


Figure 3.7: RSSI time series from channel 32 and node 6, before and after the application of a Butterworth filter of 4th order with cutoff frequency 375 Hz.

We present examples of the synchronized RSSI time series before and after the filtering process. We include one example for each of the five different classes of interference to demonstrate that their key features are preserved. Specifically, Figure 3.8 shows the time series from the quiet channel 241 at node 4 before and after the filtering process, Figure 3.9 shows the quiet-with-spikes time series from channel 33 at node 6 before and after the filtering process and Figure 3.10 shows the quiet-with-rapid-spikes time series from channel 91 at node 16 before and after the filtering process. Figure 3.11 shows the shifting-mean time series from channel 127 at node 2 before and after the filtering process and lastly Figure 3.12 shows the high-and-level time series from channel 161 at node 13 before and after the filtering process. It is evident that the low-frequency variations (spikes in the cases of Figures 3.9-3.10) that are the features characterizing the time series, are preserved albeit with a somewhat smaller amplitude. On the other hand, the high frequency variations are smoothed out as expected.

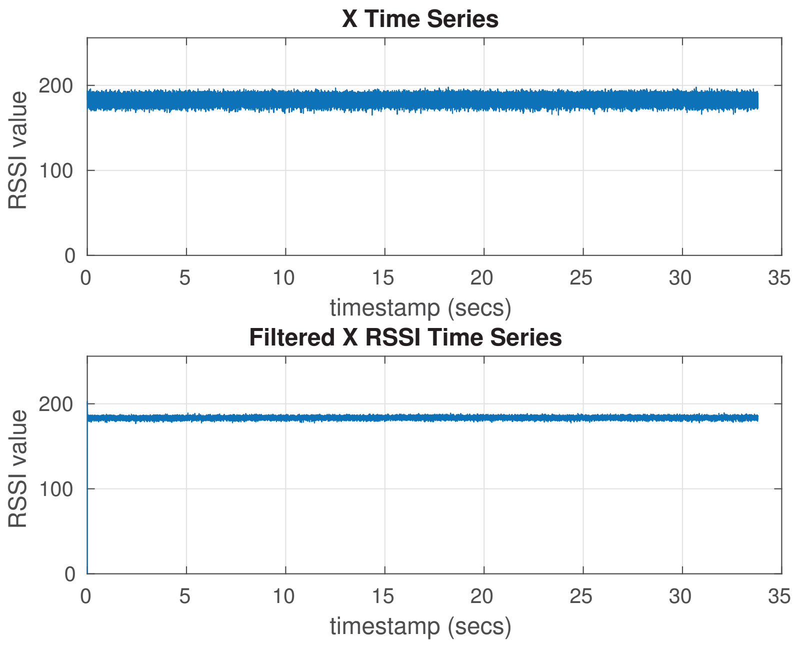


Figure 3.8: Example of the application of the Butterworth filter on the sampled RSSI time series from channel 241 (quiet) and node 4.

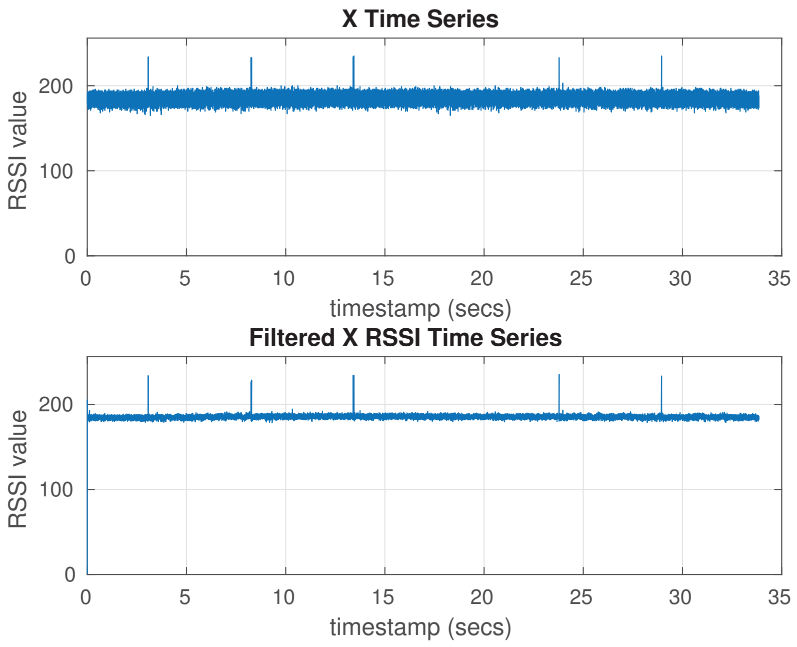


Figure 3.9: Example of the application of the Butterworth filter on the synchronized RSSI time series from channel 33 (quiet-with-spikes) and node 6.

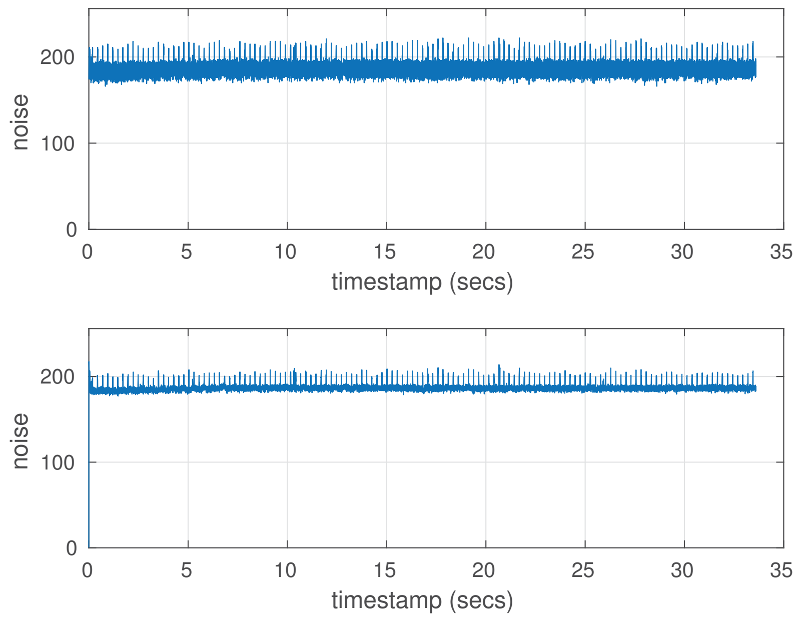


Figure 3.10: Example of the application of the Butterworth filter on the synchronized RSSI time series from channel 91 (quiet-with-rapid-spikes) and node 16.

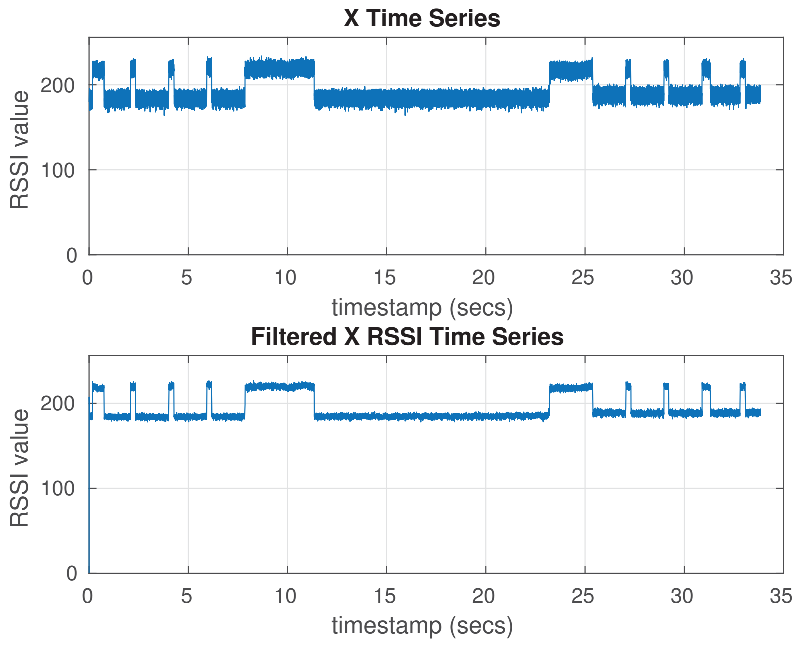


Figure 3.11: Example of the application of the Butterworth filter on the synchronized RSSI time series from channel 127 (shifting-mean) and node 2.

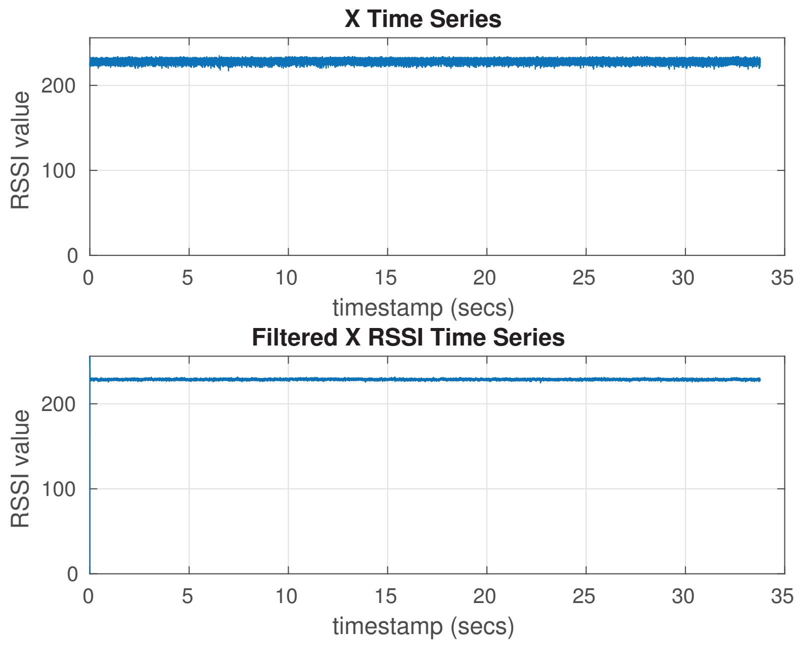


Figure 3.12: Example of the application of the Butterworth filter on the synchronized RSSI time series from channel 161 (high-and-level) and node 13.

3.4.2 Pair-wise Time Series Characterization

In this section, we present the definition of sample cross-correlation followed by the definition of another tool, the Fano factor, that is helpful in examining the relation of pairs of time series. We determine the sample cross-correlation on all pairs of nodes and for each channel. What is described in this section does not constitute a distributed step; rather a means to allow us to determine the similarity (or not) of simultaneously collected time series.

Sample Cross-Correlation

The sample cross-correlation is a measure of similarity of two time series as a function of a time-lag, or time offset, between them. Consider N pairs of observations on two time series x_t and y_t where N is the series length, $\overline{x}$ and $\overline{y}$ are the sample means, and k is the lag. The sample cross-covariance function (ccvf) is given by (3.5) and (3.6). The sample variances of the two series, c_{xx} and c_{yy} are described by (3.7) and the sample cross-correlation is given by (3.8).

$$c_{xy}(k) = \frac{1}{N}\sum_{t=1}^{N-k}(x_t - \overline{x})(y_{t+k} - \overline{y}),\ \ k = 0, 1, ..., (N-1) \tag{3.5}$$

$$c_{xy}(k) = \frac{1}{N}\sum_{t=1-k}^{N}(x_t - \overline{x})(y_{t+k} - \overline{y}),\ \ k = -1, ..., -(N-1) \tag{3.6}$$

$$c_{xx} = \frac{1}{N}\sum_{t=1}^{N}(x_t - \overline{x})^2 \qquad c_{yy} = \frac{1}{N}\sum_{t=1}^{N}(y_t - \overline{y})^2 \tag{3.7}$$

$$r_{xy}(k) = \frac{c_{xy}(k)}{c_{xx(0)}c_{yy(0)}} \tag{3.8}$$

The sample cross correlation can take values within the following bounds, $-1 \leq r_{xy}(k) \leq 1$, with the bounds indicating maximum correlation, and 0 indicating no correlation. Note that a high negative correlation shows a high correlation of the inverse of one of the series [10].

We calculate the sample cross-correlation across all lags between every pair of nodes for every channel using the pre-processed synchronized and filtered time series. Our interest concentrates on the maximum absolute value of the sample cross-correlation and the lag at which it is maximized.

In an ideal globally synchronized distributed clock experiment, we would only care about the presence of strong cross–correlation at lag zero, as it expresses whether the nodes observe or not the same channel behavior at the exact time instant. Given our understanding of the node clock drift and the possible impact of buffering and imperfect synchronization at the nodes, we conjecture that, as long as the cross-correlation is maximized at a lag to within a small range around lag zero, it is very likely that the nodes indeed observe the same channel behavior at the same point in natural time, and it is only the reporting of their data that is skewed with respect to timestamp values. We rather arbitrarily set the "acceptable" lag range to within $+/-10$ samples. Cross-correlation maximized outside this short range of lags is suspicious and a strong indication that, either our technique to synchronizing the traces has failed, or the nodes do not observe the same channel behavior at the same point in time. In light of the second option consensus of per-node channel characterization would likely be erroneous.

As an illustration, the sample cross-correlation function calculated for six cases is presented below. We include five examples where the classes match for the nodes we calculate the sample cross-correlation for, and one example for classes that do not match. For each function presented, the nodes belong to the same channel.

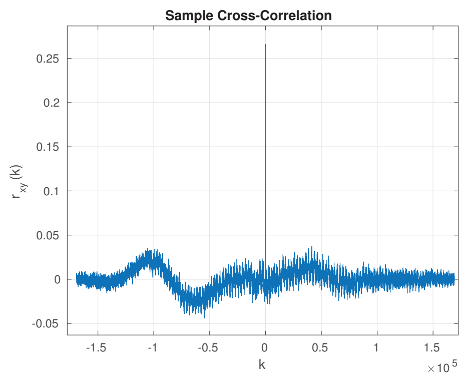
Figure 3.13a shows the sample cross-correlation for channel 242 and the quiet nodes 1 and 11. Figure 3.13b shows the sample cross-correlation for channel 33 and the quiet-with-spikes nodes 1 and 11. The sample cross-correlation for two quiet-with-rapid-spikes nodes is shown in Figure 3.13c for channel 70 and nodes 1 and 11. Figure 3.18b shows the sample cross-correlation for channel 127 and the shifting-mean nodes 1 and 11. Finally, Figure 3.13e presents the sample cross-correlation for channel 161 and the high-and-level nodes 1 and 11, while Figure 3.13f shows the sample cross-correlation for channel 121 and the quiet-with-spikes and shifting-mean nodes 3 and 11 respectively.

Fano Factor

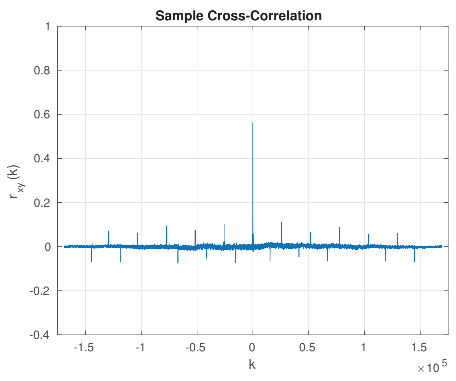
A weak cross-correlation could show that, even if a pair of nodes is observing the same behavior on the channel, the impact of noise and interference on them is different. A means to visually inspect cases where there are discrepancies despite the in-principle agreement of two nodes on the channel class is to plot the index of dispersion or variance-to-mean ratio (VMR). VMR is a normalized measure of the dispersion of a probability distribution. The VMR is defined as the ratio of the variance σ^2 over the mean μ, a statistic also known as the Fano factor, that is:

$$D = \frac{\sigma_W^2}{\mu_W} \tag{3.9}$$

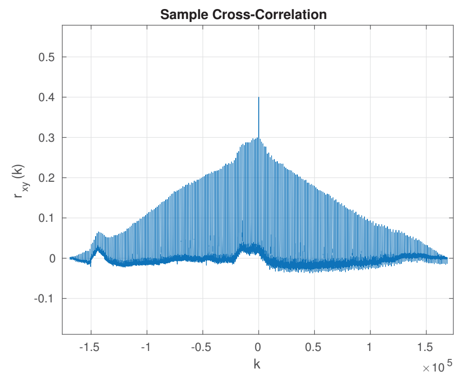
In our work, we compute the Fano factor over 500 jumping windows. A large Fano factor statistic in an interval denotes that there exist significant departures from the average behavior over that interval. Moreover, if two series do not have the same Fano factor value in an interval, the difference between the two time series cannot be compensated for by means of a simple scaling factor. That is, the nodes see a potentially different behavior with respect



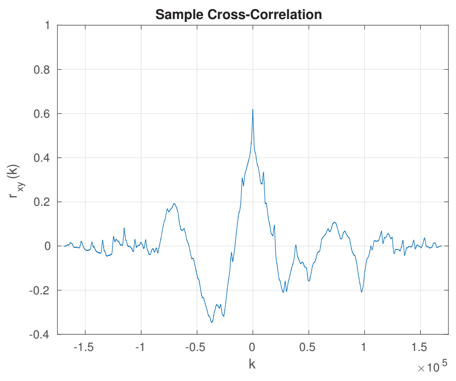
(a) Channel 242, $r_{xy}(k)$ between nodes 1 and 11 (both quiet).



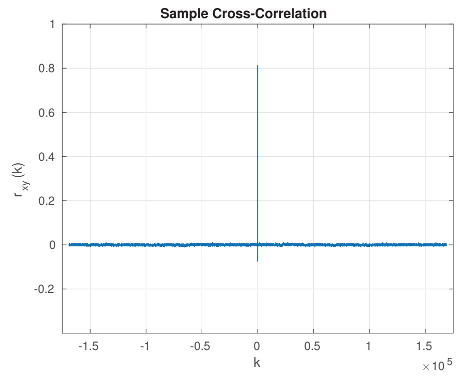
(b) Channel 33, $r_{xy}(k)$ between nodes 1 and 11 (both quiet-with-spikes).



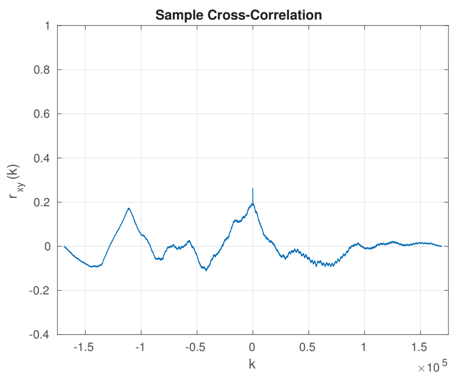
(c) Channel 70, $r_{xy}(k)$ between nodes 1 and 11 (both quiet-with-rapid-spikes).



(d) Channel 127, $r_{xy}(k)$ between nodes 1 and 11 (both shifting-mean).



(e) Channel 161, $r_{xy}(k)$ between nodes 1 and 11 (high-and-level).



(f) Channel 121, $r_{xy}(k)$ between nodes 3 and 11 (quiet-with-spikes and shifting-mean).

Figure 3.13: Sample cross-correlation function.

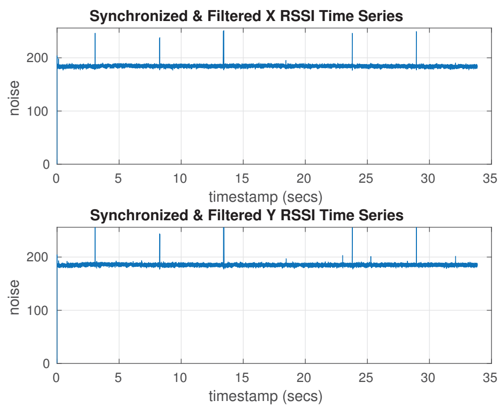
to the noise process and that, in turn, might indicate completely different SNR if communication between the nodes was attempted. As we will see, there are numerous cases where, even though the nodes agreed on the class, in reality the channel conditions seen by different nodes differ, expressed as low cross-correlation results. In such cases, the Fano factor helps clarify those differences.

3.5 Results

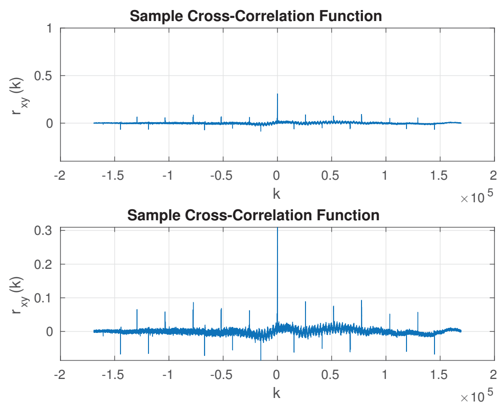
In this section, we first present sample cross-correlation results for a few selected channels in which all 16 nodes have been separately characterized but agree on the channel. We will examine whether such an agreement can be linked to the sample cross-correlation. Subsequently, we look into the sample cross-correlation results for the combination of all pairs of nodes over all channels to determine what are typical maximum sample cross-correlations depending on classes [57].

3.5.1 Quiet with Spikes (qs) Channel

Channel 33 is representative of the quiet with spikes pattern. Figure 3.14a presents the RSSI time series for nodes 1 and 2. The characteristic of this pattern, namely the spikes, are the primary contributors to the sample cross-correlation value. Specifically, aligned spikes across the two time series will produce the maximum sample cross-correlation value. As an illustration, in Figure 3.14b it is clear that the maximum sample cross-correlation for the two nodes was found at lag 0, indicating complete synchronization between the examined RSSI time series. In Figure 3.22a the Fano factor for nodes 1 and 2 in channel 33 is presented. It is evident that the levels of dispersion in the two signals are similar with the higher dispersion values occurring at the spikes. Consequently, we can safely characterize the observed channel behavior as being similar between nodes 1 and 2.



(a) Channel 33 (quiet-with-spikes), RSSI time series for node 1 (X) and 2 (Y).

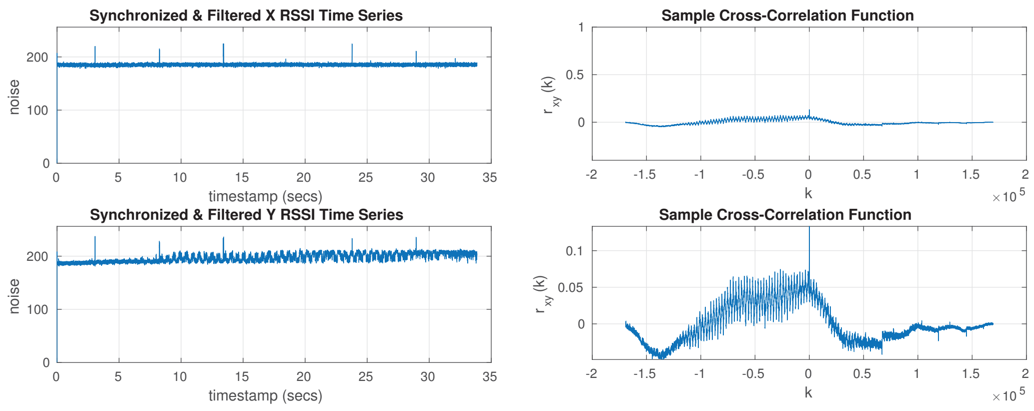


(b) Channel 33 (quiet-with-spikes), $r_{xy}(k)$ between nodes 1 and 2.

Figure 3.14: RSSI time series and sample cross-correlation function.

Nevertheless, there exist numerous cases where the maximum sample cross-correlation is low, even though the nodes agree on the classification of the channel. Such cases include aligned spikes that have different amplitudes, or cases where the spikes are preserved but the mean and variance of the segments between spikes vary significantly. There are also instances where there are mixed class characteristics within the same signal, so the signal was characterized depending on its most prominent characteristic. Figure 3.15a illustrates the latter case. In channel 33 all nodes were classified as quiet-with-spikes, however when we look closer at node's 16 RSSI time series, although it contains the characteristic spikes its amplitude doesn't remain on the same level for almost 3/4 of the signal duration. In such cases, if we rely on the maximization of the sample cross-correlation to determine if the time series lag is within acceptable synchronization error, it is possible to find the maximum cross-correlation at a lag outside the acceptable lags. or if it is within the acceptable lags it will be significantly low. In channel 33 such behavior is encountered in pairs composed of the nodes 9, 13 and 16. Specifically, Figure 3.15b presents the sample cross-correlation function for channel 32, between nodes 10 and 16. Although, it seems that we have correct synchronization since the maximum sample cross-correlation value is occurring at lag 0, it is also notable that the absolute maximum sample cross-correlation value is 0.1332, significantly lower than the one in the synchronized time series of the same channel presented in Figure 3.14b, which reached the value 0.3095. Such disagreement is an indicator that, at a microscopic level, the nodes observe the channel as being drastically different despite their consensus characterization as being of the same class.

Additionally, observing the Fano factor for nodes 10 and 16 in Figure 3.22b, we notice that the dispersion of the received signal in node 16 is significantly different and higher than the one in node 10, due to fluctuations of the mean value. It is interesting to observe that the large dispersion values for node 10, correspond to times where spikes occur and totally overlap with the dispersion values of node 16. As a result, even with the presence of the spikes in both series and despite the spikes being aligned/synchronized, the intervening quiet segments of the channel observed by node 16 exhibit severe fluctuations compared to node 10.



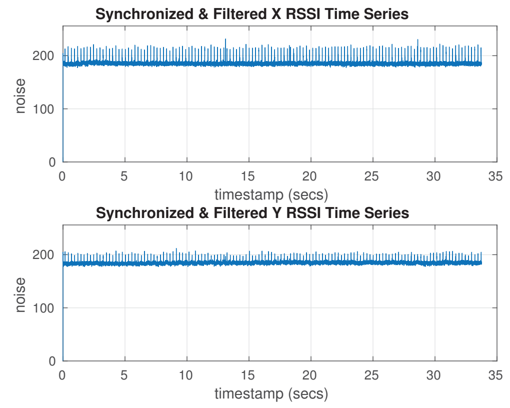
(a) Channel 33 (quiet-with-spikes), RSSI time series for node 10 (X) and 16 (Y).

(b) Channel 33, $r_{xy}(k)$ between nodes 10 and 16.

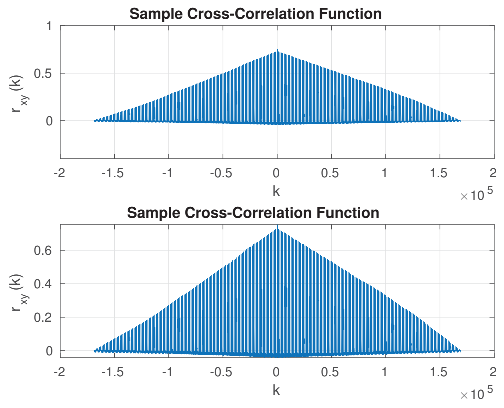
Figure 3.15: RSSI time series and sample cross-correlation function.

3.5.2 Quiet with Rapid Spikes (qrs) Channel

Channel 61 represents the quiet with rapid spikes pattern. High sample cross-correlation values were encountered, as there are more synchronized spikes to contribute to the sample cross-correlation statistic. The interesting observation for this class is that the (usually) periodic nature of the rapid spikes results in a sample cross-correlation function which captures this periodicity. This behavior is also captured in Figure 3.16b, which represents the sample cross-correlation between nodes 1 and 7. Observe the max sample cross-correlation of 0.4472 for synchronization at lag 0. Note that this behaviour is also observed in other quiet-with-rapid spikes channels, namely 59, 60, 62, and 82.



(a) Channel 62 (quiet-with-rapid-spikes), RSSI time series for node 1 (X) and 7 (Y).



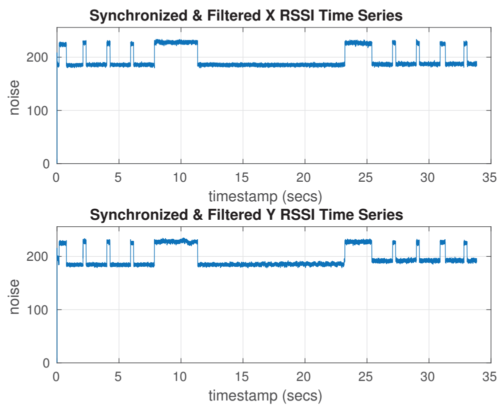
(b) Channel 62, $r_{xy}(k)$ between nodes 1 and 7.

Figure 3.16: RSSI time series and sample cross-correlation function.

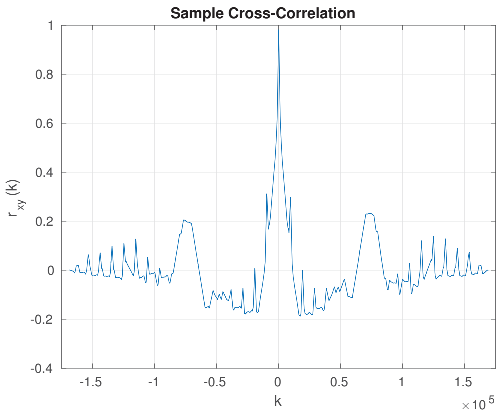
Furthermore, the Fano factor shown in Figure 3.22c (zoomed into a range to clearly show the periodic nature), reveals that for nodes 1 and 7 on channel 62, the high dispersion values are present whenever a spike occurs. Note that the Fano factor values are close, suggesting similar mean and variance, confirming a very strong similarity on how the channel is observed by the two nodes across the length of the trace, but at the same time capturing the higher amplitude of spikes in node 1.

3.5.3 Shifting Mean (sm) Channel

Channel 127 is an example of a shifting mean channel. Shifting mean channels were characterized by overall higher maximum sample cross-correlation values, frequently exceeding 0.9 and approaching 1.0. For channel 127, whose instances for nodes 13 and 15 are depicted in Figure 3.17a the lag values were always acceptable (within ±10 samples) with all the pairs indicating maximum sample cross-correlation value at lag 0. As an example, the maximum sample cross-correlation value for the node pair 13 and 14 reaches the value 0.9812 at lag 0, as shown in Figure 3.17b.



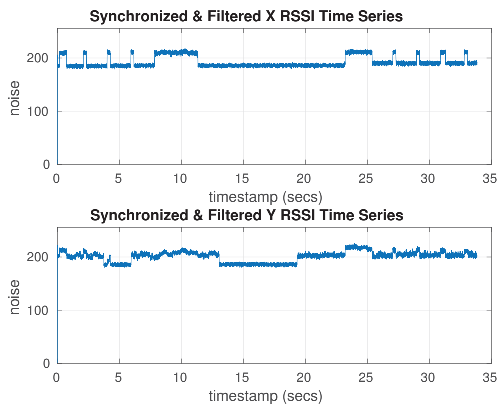
(a) Channel 127 (shifting-mean), RSSI time series for node 12 (X) and 14 (Y).



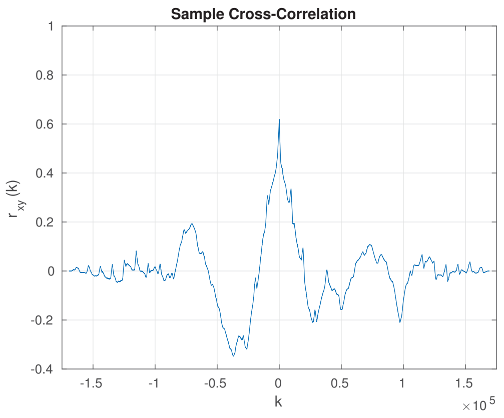
(b) Channel 127, $r_{xy}(k)$ between nodes 13 and 15.

Figure 3.17: RSSI time series and sample cross-correlation function.

Nevertheless, there exists a notable exception for channel 127, that of pairs involving node 11, whose maximum sample cross-correlation values are somewhat lower in the 0.50-0.60 region. We select the node pair consisting of nodes 1 and 11, shown in Figure 3.18a, that falls in this category and its sample cross-correlation function is pictured in Figure 3.18b. In the examined case the maximum sample cross-correlation value is 0.6195 for lag 0, indicating absolute synchronization. But after a closer observation, we conclude that even if the time series are near perfectly synchronized (i.e., the level shifts occur at lag 0 for the two signals), the sample cross-correlation value strongly depends on the levels of the mean and variance. Since the mean and variance are not necessarily the same, lower sample cross-correlation could be calculated as a result. The Fano factor captures this behavior in both node pairs we select to present for channel 127, in Figures 3.22d and 3.22e.



(a) Channel 127 (shifting-mean), RSSI time series for node 1 (X) and 11 (Y).



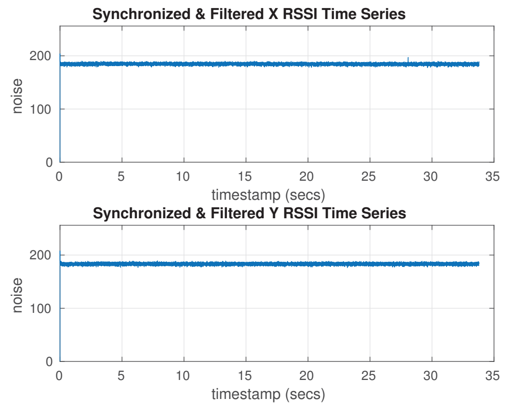
(b) Channel 127, $r_{xy}(k)$ between nodes 1 and 1.

Figure 3.18: RSSI time series and sample cross-correlation function.

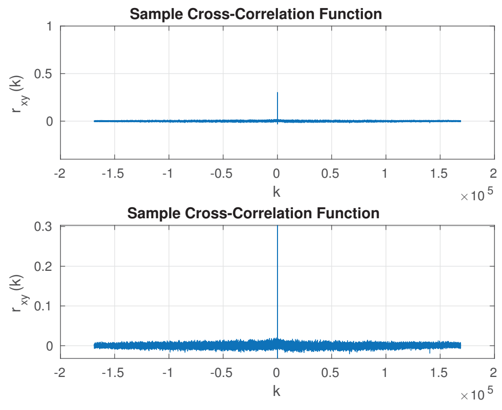
3.5.4 Quiet (q) Channel

Channel 251 is an example of a quiet channel. Quiet channels do not present any distinguishing characteristics, like spikes. However, even if the variance between a pair of quiet signals tends to stay at the same levels, their mean values are not necessarily similar. As a result, for quiet channels the sample cross-correlation produces the lowest maximum values. These small maximum values can be as low as in the 0.0 to 0.4 range. They rarely exceed 0.5. The sample cross-correlation function of channel 251 between nodes 7 and 14 shown in Figure 3.19b, exhibits the 'difficulty' of the two signals to be synchronized. The sample cross-correlation stays at extremely low levels throughout the lags, while the maximum value 0.3031 is calculated for lag 0.

In Figure 3.22f (zoomed into a range to clearly show the lack of agreement), the cause becomes clearer as the Fano factor values are totally disparate. This is an indication of different mean values and variance. As a result the synchronization of the time series becomes harder and, consequently, the sample cross-correlation values remain in low levels. In addition to channel 251, channels 213, 214, 215 and 216 exhibit the same behavior.



(a) Channel 251 (quiet), RSSI time series for node 7 (X) and 14 (Y).

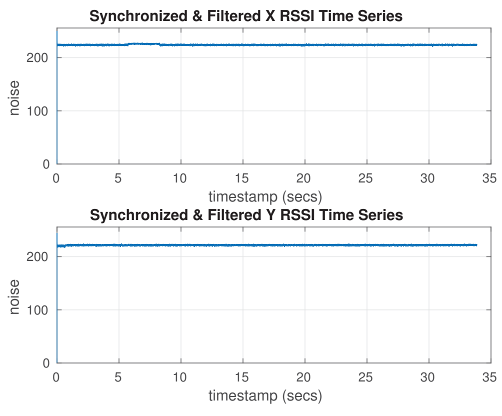


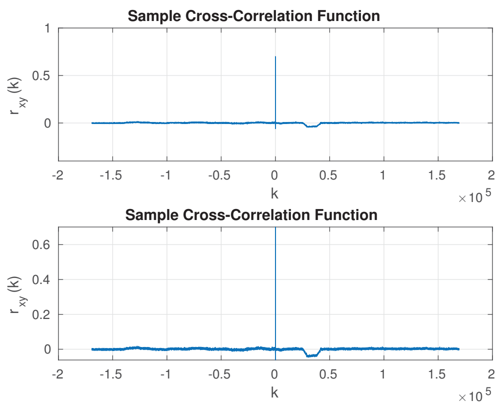
(b) Channel 251, $r_{xy}(k)$ between nodes 7 and 14.

Figure 3.19: RSSI time series and sample cross-correlation function.

3.5.5 High and Level (hl) Channel

Channels characterized as high-and-level like 96, 161 and 226 present a behaviour similar to quiet channels. High and level channels can also be characterized as quiet but with higher amplitudes. However, looking closer into a pair of traces from channel 96 and nodes 8 and 14 in Figure 3.20a and their sample cross-correlation function in Figure 3.20b, we observe a max value of sample cross-correlation found at lag 0. It's value is a surprisingly high 0.7010, and generally high-and-level channels present high maximums within 0.5-0.8 , even though they don't have characteristic attributes (like spikes) to contribute to the higher value. The similarities are also validated from the Fano factor results in Figure 3.21 which although looks a lot like the Fano factor in the quiet case in Figure 3.22f, we csn observe how much smaller the differences for the statistic are for nodes 8 and 25 in Figure 3.21 in the y axis.





(a) Channel 96 (high-and-level), RSSI time series for node 8 (X) and 14 (Y).

(b) Channel 96, $r_{xy}(k)$ between nodes 8 and 14.

Figure 3.20: RSSI time series and sample cross-correlation function.

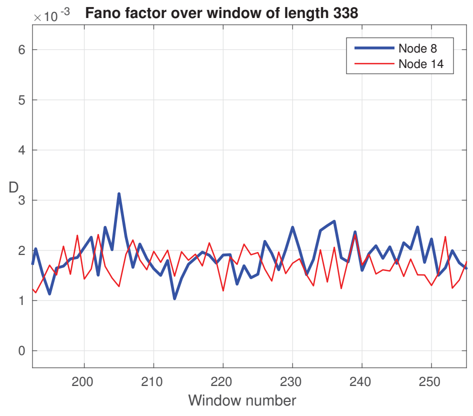
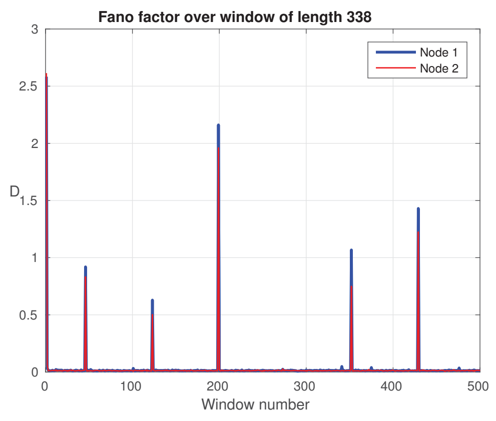
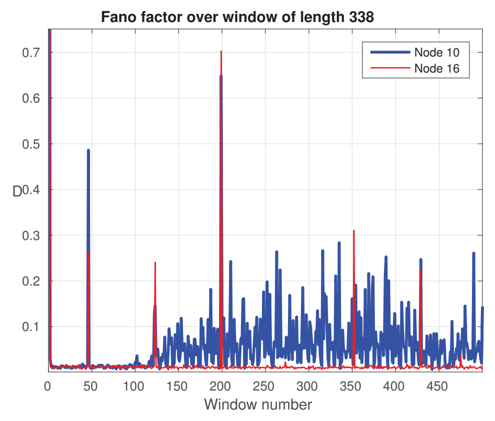


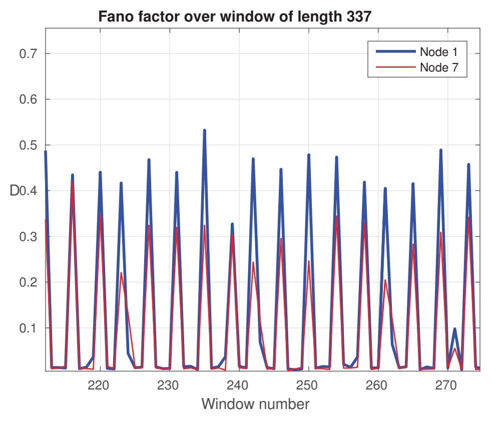
Figure 3.21: Channel 96, D for nodes 8 and 14.



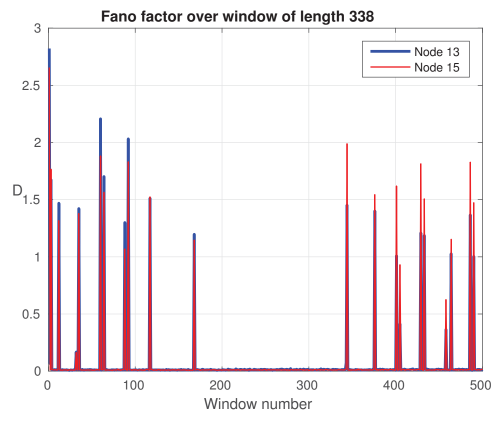
(a) Channel 33, D for nodes 1 and 2.



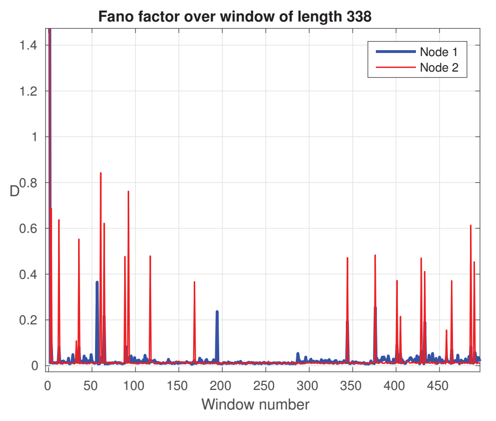
(b) Channel 33, D for nodes 10 and 16.



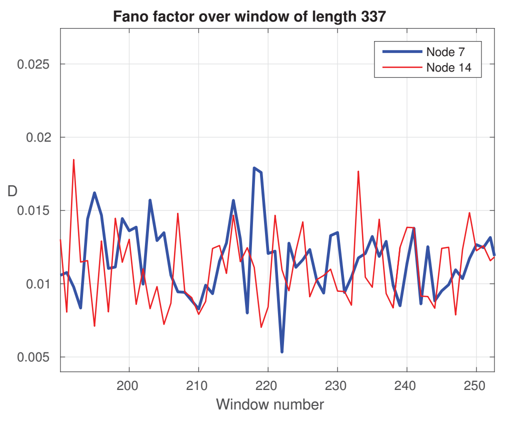
(c) Channel 62, D for nodes 1 and 7.



(d) Channel 127, D for nodes 13 and 15.



(e) Channel 127, D for nodes 1 and 11.



(f) Channel 251, D for nodes 7 and 14.

Figure 3.22: Fano factor.

3.5.6 Analysis of Node Pairs

We first analyze all pairs of nodes across all channels. 120 unique node pairs can be defined, which multiplied by 256 channels give us 30720 pairs under examination. Of those, 23438 node pairs agree on the class to which they have classified the channel and the remaining 7282 disagree on the class. Of the 23438 that agree on the class, 21299 exhibit maximum sample cross-correlation at small lags $\in [-10, 10]$ *samples* that indicate synchronization *and* agreement on the correct classification of the signals.

We first use this group of 21299 pairs for our conclusions on the linkage between maximum cross-correlation and classification, as shown in Table 3.1. It can be seen that the quiet with rapid spikes, shifting mean and high-and-level channels class characterizations can be trusted as depicting accurately the same channel state. The quiet-with-rapid-spikes and high-and-level classification is debatable as a non-trivial percentage (15.4% and 13.0% respectively) corresponds to low maximum cross-correlation, which could indicate lack of actual correlation, but still more than 50% exhibit significant correlation. The quiet and the quiet with spikes classifications are the most problematic because of the very low cross-correlation, however a 28.8% of the quiet-with-spikes class present significant correlation.

Table 3.1: Percentages of the maximum $r_{xy}(k)$ occurring at k $\in [-10, 10]$ and with value falling within specific bounds, for same-class node pairs.

max $r_{xy}(k)$	q	qs	qrs	sm	hl
$[0, 0.2)$	9.9%	6.9%	5.1%	0.3%	0.3%
$[0.2, 0.4)$	83.5%	62.4%	15.4%	2.3%	13.0%
$[0.4, 0.6)$	5.5%	28.8%	39.5%	8.7%	31.8%
$[0.6, 0.8)$	1.0%	1.9%	36.3%	13.6%	46.5%
$[0.8, 1)$	0.1%	0.0%	3.6%	75.1%	8.3%

For the sake of comparison, we considered pairs of nodes that disagreed on the channel class. This is shown in Table 3.2 and confirms that the results for quiet and quiet with spikes are readily comparable to the case where the nodes observe what they classify as completely different channel behaviors.

Finally, we consider the results for pairs (2139 of them) that were found to

Table 3.2: Percentages of the maximum $r_{xy}(k)$ falling within specific bounds for pairs of nodes that do not belong to the same class.

max $r_{xy}(k)$	Percentage
$[0, 0.2)$	19.1%
$[0.2, 0.4)$	64%
$[0.4, 0.6)$	12.9%
$[0.6, 0.8)$	3.1%
$[0.8, 1)$	0.9%

Table 3.3: Percentages of the maximum $r_{xy}(k)$ occurring at k $\notin [-10, +10]$ and with value falling within specific bounds, for same-class node pairs.

max $r_{xy}(k)$	q	qs	qrs	sm	hl
$[0, 0.2)$	48.5%	59.6%	17.8%	3.3%	50.0%
$[0.2, 0.4)$	43.6%	32.0%	35.4%	18.5%	43.1%
$[0.4, 0.6)$	6.8%	7.9%	39.3%	21.0%	6.9%
$[0.6, 0.8)$	1.1%	0.5%	7.5%	14.4%	0.0%
$[0.8, 1)$	0.0%	0.0%	0.0%	42.8%	0.0%

be “out–of–sync” with respect to the lags. This can be used to point out how a simple cross-correlation metric is limiting the study of similarity between time series. As shown in Table 3.3 in the case of quiet with rapid spikes, using the maximum cross–correlation may result in favouring a large lag, primarily because the amplitude of the periodic peaks further away in time could be larger and add up to a numerically higher cross-correlation at unnaturally large lags. We conjecture that the same happens with the shifting mean class because a pattern of shifts could be repeated at higher amplitude further away in the time series than lag zero.

Final Remarks

We have studied whether the cross-correlation between RSSI measurements carried out by WSN nodes in the same network reflects accurately the consensus about the channel state, had the nodes independently decided on the channel state based on a classification scheme. The results paint a mixed picture whereby a consensus towards a shifting mean, a quiet with rapid spikes or high-and-level classification can be trusted, effectively if we were to draw a line at cross-correlation of 0.4 lower values would suggest the nodes are likely seeing different time series behavior while larger values would mean agreement. However, patterns that do not exhibit much dynamic behavior, i.e., quiet, or even quiet with occasional spikes, are not characterized by a cross-correlation much higher than what would have been if there was no agreement on the class of the traces at all. The recommendation therefore is that, if consensus algorithms are to be utilized, a very reliable per-node classifier for quiet, or quiet with spikes channel would be necessary.

Our study is far from perfect. For example, the use of cross-correlation as the means of studying similarity between time series over their entire length does not reveal possible short-term similarities that do not necessarily persist or are, numerically speaking, diluted over a long time period. Additionally, information could be used to annotate the classification, i.e., the period for periodic spikes. Nevertheless, we point out that a (summarized) description based on temporal characteristics that further "parameterize" the class would still need some common synchronization adjustment. Equipped with these observations we will maintain the synchronization described here but add a mechanism for each node to compress and send its synchronized time series to the rest of the nodes as described in the next chapter.

Chapter 4

Wavelet-Based Analysis of Interference

4.1 Introduction

In this study, we explore whether wavelet compression is an effective means to convey sampled background noise information collected by wireless nodes. The work of this chapter is an updated and extended version of our paper in [58]. The motivating application is that of distributed decision-making by wireless sensor nodes regarding the state of the channel with respect to the presence (or not) of interference on the channel. The nodes could, subsequently, adopt mitigation strategies particular to the interference at hand. Here, we are only concerned with developing a low overhead means to communicate the sampled background noise information, among the nodes in a local wireless network. The assumption is that the nodes are battery–powered and any additional processing and transmission, beyond what is needed by the applications, should be minimized.

Our previous work in this area in Chapter 3, as well as that of a number of other authors, suggested that there are a few particular classes of interference [9, 37]. As discussed previously in 3.3.2, five classes were identified in [9]: quiet (q), quiet-with-spike (qs), quiet-with-rapid-spikes (qrs), high-and-level (hl), and shifting-mean (sm). Note that the main difference between qs and qrs is that qs has seemingly random impulses whereas qrs has strong periodic characteristics. A receiver at the mercy of qrs, sm, and hl is likely to be, on occasion, unable to properly decode a packet received during high level incursions of the interference. Note that, as in our previous work, we assume an inexpensive approach to sampling the channel, i.e., by collecting RSSI (Received Signal Strength Indicator) measurements, of the background noise.

In order to reach an agreement (or not) among nodes in a network regarding the class of interference that is present, there exist two main strategies: (a) *local classification:* where each node samples the background noise and classifies it independently of the rest, sending its classification result to the rest, or (b) *global classification:* where each node sends the entire time series of sampled background noise to the remaining nodes, allowing each node to perform a comparison, and if deemed similar, run a classifier. Option (a) entails the risk that, even if the class inferred is the same, two or more nodes may be seeing a completely different temporal pattern e.g. a different phase of a qrs-like time series. Option (b) allows a thorough sample-by-sample analysis, e.g., via cross-correlation calculation of the time series from different nodes, but the transmission and reception energy cost involved to collect the samples by all nodes is prohibitive for battery-powered wireless nodes.

The characterization of the agreement between nodes with respect to the interference seen on the channel (Chapter 3) was performed assuming access to the time series as sampled by each node. In this chapter we follow the logic of Chapter 3 by introducing (instead of consensus of per-node classification) an explicit transmission step of the sampled time series by all nodes to all nodes (broadcast). An intermediate (compressed) representation of the sampled data is used via a Discrete Wavelet Transform (DWT). DWT allows the representation of the signal as a set of coefficients. Note that the application of DWT is a delicate matter when we wish to have the noise patterns in some way "survive" the transformation but at the same time we wish to eliminate some of the less helpful (from the point of deciding the class) facets of noise. Hence, in our work we examined various DWT alternatives and their effects. In essence, we use a lossy compression, and we evaluate its impact.

Note that it is implied that a node has to store its own time series as well as the time series samples sent from other nodes (in DWT form). Given the general scarcity of memory in sensor nodes the time series segments are not very long (or the sampling rate to acquire them is not very high). Operationally, we would expect that the nodes take, possibly periodically, some time off their regular operation to simultaneously sample the background noise of the channel in order to collectively determine the state/class of the channel. In Section 4.3 we explain how the comparison of the time series is performed based on the cross-correlation across different nodes' time series.

4.2 Wavelet Experimentation

Choosing the most suitable wavelet filter allows the characterization of the frequency content of the time series by a small set of coefficients, which after an operation called "thresholding" and using the produced coefficients via an inverse DWT (IDWT) transform to reconstruct the signal, will lead to a recovered and denoised version of the time series while preserving the characteristics of every class.

4.2.1 The Data

We use the Received Signal Strength Indicator (RSSI) traces collected by Boers et al. [9], across 256 channels in an indoor urban environment as they are presented in Section 3.3. The pre-processing as explained in 3.4.1 is also maintained. The time series to be compressed are the synchronized and filtered versions of the raw RSSI traces. However, it is important to note that filtering adds a fractional part to the RSSI values (which before the filtering was represented with 8 bits per RSSI value). In order to keep the 8 bit representation, we round the filtered RSSI values so that additional overhead is not added before the compression due to the fractional part after the decimal point.

4.2.2 Metrics

We used three metrics to evaluate the wavelet and thresholding choice:

1. Root Mean Square Error (RMSE): Given the synchronized and filtered RSSI time series X (8 bit RSSI values) of length n, and the recovered signal after compression X_R via the IDWT, the RMSE is:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(X_i - X_{Ri})^2} \tag{4.1}$$

   The RMSE is used as a general error measure and it has a tendency to amplify the impact of large errors.

2. Compression Percentage (C): Inspired by Chompusri's et al. [16] definition of compression ratio, we define our metric to measure the fraction of the original data volume by representing it as DWT coefficients. Namely, we calculate the number of bits needed to represent the compressed signal (output), as well as the number of bits to represent the synchronized and filtered (rounded) RSSI values we give as an input. We define the wavelet decomposition vector of coefficients as $DWTCX$ and the bookkeeping vector as L. L contains the population of thresholded coefficients vector up to level N. Its length is $N + 2$. We also define the length of the examined RSSI series as n.

   The bits needed to represent the synchronized and filtered (rounded) RSSI values are:

$$InputBits = n * 8 \tag{4.2}$$

   Note that before the thresholding the set of coefficients $DWTCX$ produced by the DWT also contain a fractional part after the decimal point that we discard. To do so, we round the set of coefficients $DWTCX$ and this is the new set, CX, of coefficients that is subsequently thresholded. After the thresholding with different techniques, certain coefficients are zero (the ones that were close to zero have already been set to zero after the rounding). Only the non-zero coefficients need to be explicitly transmitted. However, we also use a bitmap vector to convey the location of the zero coefficients. This bitmap vector consists of a number of bits equal to the length of CX.

The number of bits needed per thresholded coefficient is set to be just enough to represent the range of magnitude of the integers in array CX:

$$CoefBits = \lceil (log2((max(CX) - min(CX)) + 1) \rceil \tag{4.3}$$

The number of bits needed for the bookkeeping vector L is:

$$LBits = \lceil (log_2(n)) \rceil \tag{4.4}$$

The total number of bits needed for compressed signal representation (output) is therefore:

$$OutputBits = (nnz(CX) * CoefBits) + (LBits * (N + 2)) + length(CX) \tag{4.5}$$

Where $nnz(CX)$ is the number of non-zero coefficients of CX. Finally, to calculate the percentage decrease after the compression (output bits) in comparison with the input bits we use the next formula:

$$C = \frac{InputBits - OutputBits}{InputBits} * 100 \tag{4.6}$$

3. Retained Energy (RE): It is defined by the following equation, as the energy of the recovered signal over the energy of the uncompressed signal, described by their squared $L - 2$ norms, and it is expressed as a percentage.

$$RE = \frac{(\|X_R\|_2)^2}{(\|X\|_2)^2} * 100 = \frac{((\sum_{i=1}^{n} |X_{Ri}|)^{\frac{1}{2}})^2}{((\sum_{i=1}^{n} |X_i|)^{\frac{1}{2}})^2} * 100 \tag{4.7}$$

In addition, we visually inspected the recovered signals to verify whether the patterns of each class remain untampered.

4.2.3 Mother Wavelet and Thresholding Selection

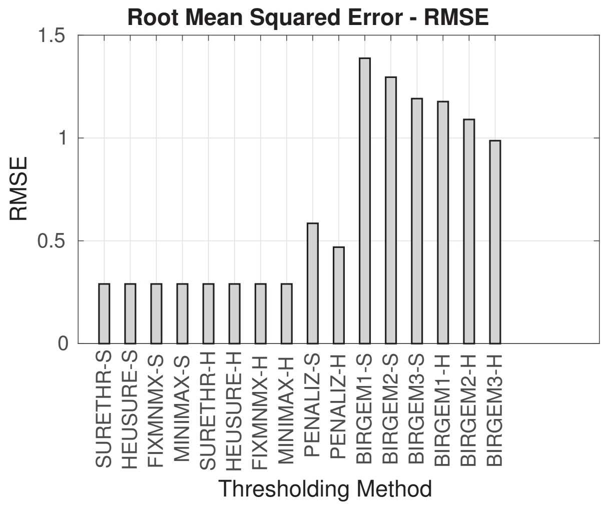
In this section, we evaluate choosing the mother wavelet and the thresholding strategy. We consider a quiet-with-rapid-spikes channel first because it has the most visual features. In Figure 4.1 the performance criteria are applied to a qrs channel (channel 54) at node 6, for different thresholding techniques. The wavelet filter is kept constant at DB2, and 5 levels (N=5) are used. In Figure 4.2 shows the performance criteria for channel 54 and node 6 is shown, for different mother wavelets, while keeping the thresholding method to heuristic SURE and 5 levels (N=5).

Initially, we keep the wavelet filter constant and implement all the thresholding methods, using MATLAB's Wavelet Toolbox [42]. After the best thresholding method is selected, it is kept constant and the various types of wavelet basis functions are evaluated. Specifically we consider the following:

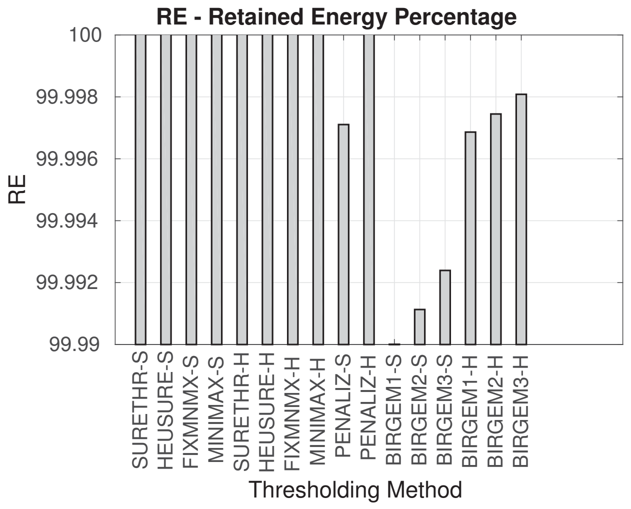
1. Thresholding methods (both soft and hard): SURE (SURETHR), Heuristic SURE (HEUSURE), fixed minimax multiplied by logarithm of length (FIXMNMX), Minimax (MINIMAX), Penalized Threshold 1-D de-noising (PENALIZ), Birge-Massart Thresholding (for three M values, where M defines the number of coefficients to be kept at level i)(BIRGEM1,BIRGEM 2,BIRGEM3).

2. Type of wavelet basis function [1]: Daubechies filter (DB) of order 2,4,6,8,10, Symmlet filter (SYM) of order 2,4,6,8, Coiflet filter (COIF) of order 1,2,3,4.

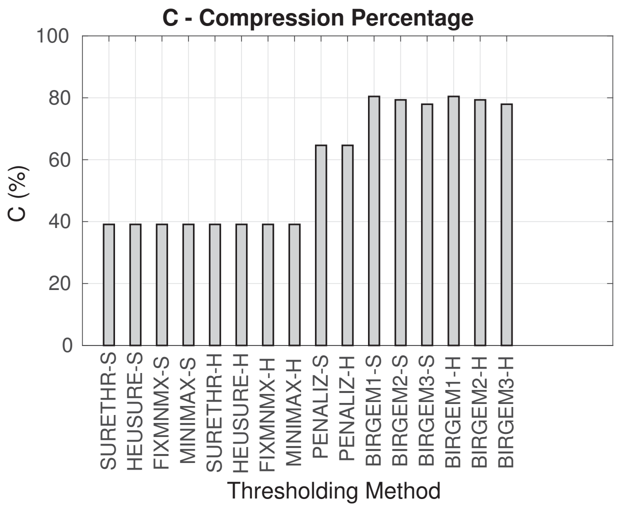
[1] `http://www.mathworks.com/help/wavelet/ref/waveletfamilies.html#zmw57dd0e33578`



(a) RMSE

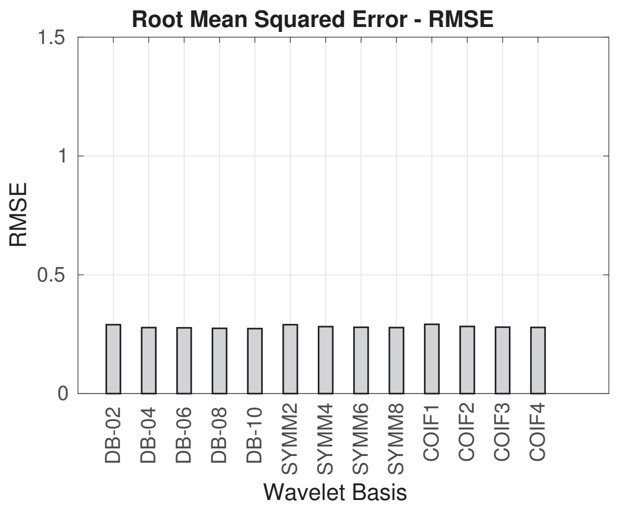


(b) RE

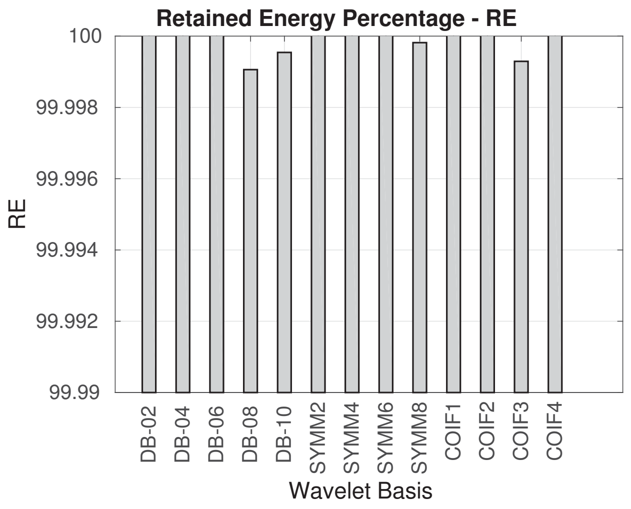


(c) C

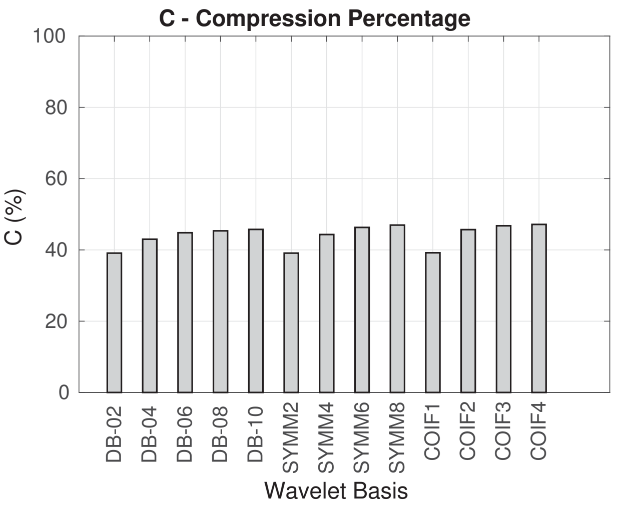
Figure 4.1: Channel 55 (quiet-with-rapid-spikes), for node 7 and levels=5, wavelet filter=db2.



(a) RMSE



(b) RE



(c) C

Figure 4.2: Channel 55 (quiet-with-rapid-spikes), for node 7 and levels=5, thresholding method=Soft Heuristic SURE.

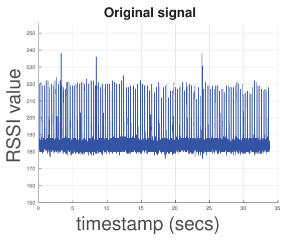
From the results in Figure 4.1, it is evident that the different thresholding schemes are very similar in terms of absolute values, for the RMSE and RE metrics. The cost in terms of bits seems to be reduced in the same percentage, about 40%, for the first eight thresholding methods and it doubles for soft and hard Birge-Massart thresholding. The latter doesn't come without a cost, although not significant in terms of RMSE, as shown in Figure 4.1 (a).

From Figure 4.2, we observe that we cannot favor a specific mother wavelet depending on the RMSE and RE results, when using heuristic SURE as our thresholding method, however mother wavelet of families in their lowest order (DB2, SYMM2, COIF1) give slightly better compression results.

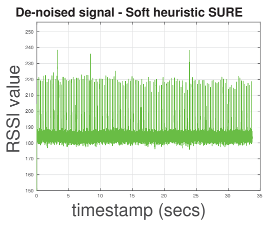
These metrics did not provide the full story as to whether the interference pattern in the time series were preserved. After visual inspection, this turned out to be strongly dependent on the thresholding method chosen. Visually speaking, we are interested in the thresholding method that guarantees consistent preservation of spikes for the qs and qrs classes, as well as a uniform behavior of the noise present in the signals for all levels (i.e. noise preserved or suppressed throughout the duration of each signal and on the same scale across different classes). Also, the more levels that were used in the transform, the more intense the denoising and compression effects became. Moreover, as the levels increase the possibility of losing a pattern increases depending on the principles applied by the thresholding method.

Some examples for acceptable and unacceptable results of thresholding in qrs and sm classes are presented in Figures 4.3 - 4.4 . Note that the qs exhibits a similar behaviour to qrs, while the q class is similar to the hl.

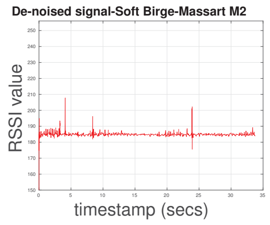
A qrs class series (channel 55, node 7) shown in Figures 4.3a - 4.3e shows the following: the recovered signal after soft heuristic SURE thresholding and denoising over 10 levels using a db2 wavelet filter limits the amplitude of the noise in the base of the signal, while preserving the periodic spikes. The soft heuristic SURE method is preferred over the hard heuristic SURE method, because hard seems to suppress periodic spikes. Also, not choosing the right thresholding method is evident in Figures 4.3d and 4.3e, where the soft Birge-Massart thresholding will completely suppress the periodic spikes (leaving be-



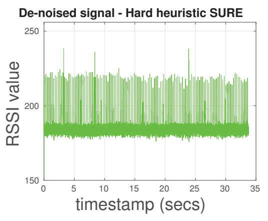
(a) Channel 55, node 7.



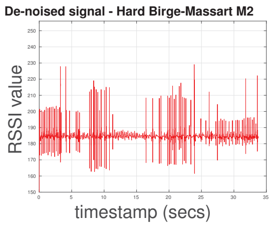
(b)



(d)

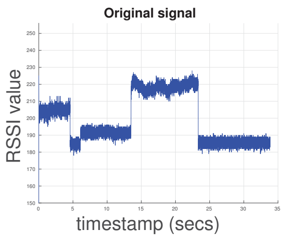


(c)

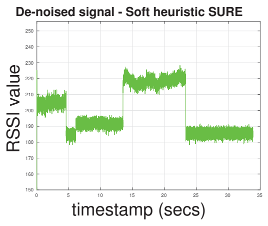


(e)

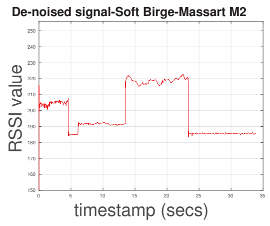
Figure 4.3: Channel 55 (quiet-with-rapid-spikes), for node 7 and levels=10.



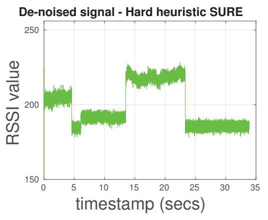
(a) Channel 139 node 9.



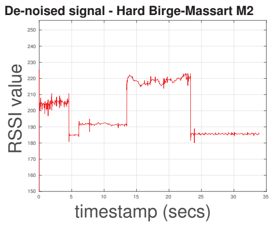
(b)



(d)



(c)



(e)

Figure 4.4: Channel 139 (shifting-mean), for node 9 and levels=10.

hind only two in this example), while the hard version of the same thresholding method will preserve more spikes but not of the same amplitude, completely deforming the initial qrs pattern. Note that the Birge-Massart thresholding showed the best compression results.

A sm class series (channel 139, node 9) shown in Figures 4.4a - 4.4e shows the following: good results are obtained from soft heuristic SURE method and hard heuristic SURE (Figure 4.4c) and even the soft version of the Birge-Massart thresholding (Figure 4.4d). However, in the soft and hard Birge-Massart thresholding implementation only the shape of the RSSI class pattern is preserved and the noise level doesn't reflect the one of the original signal.

We have also studied the case of hl class time series, and a distinct observation there is that the hard Birge-Massart thresholding results in undesired effects, as it does not uniformly suppress the noise, leaving spikes of noise randomly interfering with the dominant "clean" pattern.

In summary, based on the results we collected, we can claim that the soft heuristic SURE thresholding method has the acceptable and desired effects of denoising and compression for all five different classes, while we cannot favor a specific mother wavelet filter, see e.g. Figure 4.2(c), more than others since they all manage to give good results. Interestingly, the thresholding methods we rejected, while not capable of preserving the characteristic patterns of the classes well, exhibit a good RE of the signal, maintain a low C and a low RMSE. In short, we have indications that the quantitative facets are unfortunately insufficient. Exclusively trusting them without visual inspection does not provide a sense of whether patterns are preserved.

4.3 Time Series Comparison

With the wavelet basis function and thresholding decided (and assumed known to all nodes), the next task is to find the similarity across the time series collected from the various nodes, once decompressed via IDWT, via pairwise cross–correlation computation. Time series of different nodes, that we know are very similar, should exhibit a very high cross-correlation, as established in

Pairwise Analysis of Time Series

Following the same strategy as in 3.4.2 we set the “acceptable” lag range to correspond to timestamp discrepancies of $+/-10$ samples, so that these results are comparable to the ones from our previous work in Chapter 3. In this section we compare DWT-based results for 2, 5, and 10 levels of compression, against the cross–correlation results reported in Chapter 3 since we want to focus on the comparison between the two approaches. We are interested in the aggregate analysis of node pairs concentrated on the maximum cross–correlation occurring at small lags. We investigate how the different levels affect this aggregate analysis by presenting results for 2, 5 and 10 levels of compression, side by side with the results in Chapter 3. We investigate how the different levels affect this aggregate analysis.

Our findings are summarized in Tables 4.1-4.3. We begin with Table 4.1, which considers node pairs that are known to agree on the channel classification to see how well cross correlation can identify the similarity and presents the percentages of the maximum cross–correlation occurring at small lags, as defined previously. Table 4.2 also presents the statistics for node pairs that agree on the class, but in this case we take into consideration cases of the maximum cross–correlation occurring outside the ’acceptable’ lags, as defined previously.

Lastly, Table 4.3, considers node pairs that are known to not agree on the channel classification to see how well cross correlation depicts the percentages of the maximum cross–correlation falling within specific bounds (bins) for pairs of nodes that do not belong to the same class in order to observe their behavior.

In this work, having 30720 pairs in total, from the 23438 (based on the ground truth found [9]) that agree on the class, we found that 21300 and 21299 pairs exhibit maximum cross–correlation at lags corresponding to timestamp discrepancies of $+/-10$ samples that indicate synchronization *and* correct classification of the signals, at 2 and 5 levels of wavelet compression, respectively. For 10 levels we have 21296 such pairs. As a result, we observe that

Table 4.1: Percentages of the maximum $r_{xy}(k)$ occurring at k $\in [-10, 10]$ and with value falling within specific bounds, for same-class node pairs from previous work [57] (Orig.) and for soft heuristic SURE thresholding, db2 filter and 2, 5 and 10 levels.

max $r_{xy}(k)$	q				qs				qrs				sm				hl			
	Orig.	2	5	10	Orig.	2	5	10	Orig.	2	5	10	Orig.	2	5	10	Orig.	2	5	10
$[0, 0.2)$	9.9	10.6	10.8	10.7	6.9	7.2	7.3	7.3	5.1	5.2	5.3	5.2	0.3	0.4	0.4	0.4	0.3	0.3	0.3	0.3
$[0.2, 0.4)$	83.5	83.3	83.2	83.2	62.4	63.7	64.1	64.1	15.4	15.8	16.0	16.0	2.3	2.1	2.2	2.2	13.0	14.3	14.8	14.8
$[0.4, 0.6)$	5.5	5.1	5.0	5.0	28.8	27.5	27.0	27.0	39.5	40.0	40.1	40.1	8.7	8.8	8.8	8.8	31.8	35.0	35.7	35.7
$[0.6, 0.8)$	1.0	1.0	1.0	1.0	1.9	1.6	1.6	1.6	36.3	35.7	35.4	35.5	13.6	13.8	13.8	13.8	46.5	49.0	48.0	48.0
$[0.8, 1)$	0.1	0.1	0.1	0.1	0.0	0.0	0.0	0.0	3.6	3.3	3.2	3.2	75.1	74.9	74.8	74.8	8.3	1.3	1.2	1.2

Table 4.2: Percentages of the maximum $r_{xy}(k)$ occurring at k $\notin [-10, 10]$ and with value falling within specific bounds, for same-class node pairs from previous work, [57] (Orig.) and for soft heuristic SURE thresholding, db2 filter and 2, 5 and 10 levels.

max $r_{xy}(k)$	q				qs				qrs				sm				hl			
	Orig.	2	5	10	Orig.	2	5	10	Orig.	2	5	10	Orig.	2	5	10	Orig.	2	5	10
$[0, 0.2)$	48.5	50.8	51.2	51.0	59.6	60.4	60.7	60.8	17.8	17.7	18.0	17.9	3.3	3.3	3.3	3.3	50.0	50.0	50.0	50.0
$[0.2, 0.4)$	43.6	41.9	41.4	41.6	32.0	31.6	31.6	31.3	35.4	35.7	35.6	35.6	18.5	18.9	18.4	18.5	43.1	43.1	43.1	43.1
$[0.4, 0.6)$	6.8	6.4	6.4	6.4	7.9	7.5	7.2	7.4	39.3	39.6	39.8	39.9	21.0	20.9	20.9	21.0	6.9	6.9	6.9	6.9
$[0.6, 0.8)$	1.1	0.9	0.9	0.9	0.5	0.5	0.5	0.5	7.5	6.9	6.6	6.6	14.4	14.3	14.3	14.4	0.0	0.0	0.0	0.0
$[0.8, 1)$	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	42.8	42.6	43.0	42.8	0.0	0.0	0.0	0.0

the number of node pairs that are considered to belong to the same class is almost identical in comparison to the ones in the previous work that were 21299 as shown in Chapter 3. This is very encouraging, since the cross–correlation proves that not only the DWT is a good fit for conveying the signal and its characteristics in a compressed and denoised form, but also that the applied 'Soft' Heuristic SURE thresholding technique as well as the deployed mother wavelet 'DB2' were a good fit for our data.

Aggregate Analysis of Node Pairs

To summarize our findings in Chapter 3, (as the 'Orig.' columns) of Table 4.1 that the quiet-with-rapid-spikes (qrs), shifting-mean (sm) and high-end-level (hl) channels class characterizations can be trusted as depicting accurately the same channel state. The quiet-with-rapid-spikes and high-and-level classification is debatable as a non-trivial percentage (15.4% and 13.0% respectively) corresponds to low maximum cross-correlation, which could indicate lack of actual correlation, but still more than 50% exhibit significant correlation. The quiet (q) and the quiet-with-spikes (qs) classifications are the most problematic

Table 4.3: Percentages of the maximum $r_{xy}(k)$ value falling within specific bounds, for node pairs that do not belong in the same class from previous work [57] (Orig.) and for soft heuristic SURE thresholding, db2 filter and 2, 5 and 10 levels.

max $r_{xy}(k)$	Orig.	2	5	10
$[0, 0.2)$	19.1	19.7	19.6	19.7
$[0.2, 0.4)$	64.0	64.1	64.1	64.1
$[0.4, 0.6)$	12.9	12.3	12.4	12.3
$[0.6, 0.8)$	3.1	3.0	3.0	3.0
$[0.8, 1)$	0.9	0.9	0.9	0.9

because of the very low cross-correlation.

Additionally, for the wavelet decomposition levels 2, 5 and 10 that were examined, as shown in Tables 4.1-4.3, we have results almost identical to those mentioned above and presented in Chapter 3. It is evident that for the examined levels the statistics remain the same with negligible differences, so the easy choice to make is to choose the decomposition level that will compress the signal to the least number of bits.

A cost incurred by the particular coefficient encoding we use are the non-zero coefficients that needed to represent the time series as the levels increase. While we found that for larger levels fewer non-zero coefficients are produced, the magnitude of those coefficients is larger and hence require more bits per coefficient to encode them.

As an illustration, for an uncompressed version of the RSSI time series in channel 57 and node 5, we would need 1,350,352 bits for the original data, but level-2 decomposition requires 749.172 (44.52 % decrease) bits, its level-5 requires 815.434 bits (39.61% less bits), and the level 10 requires 928.194 (31.26% decrease) bits. For comparison of the wavelet compression with off-the-shelf compression schemes, we also provide the compression results for a popular data compression program, GZIP[2]. GZIP requires 776.520 bits (42.49 % decrease), for the same RSSI time series in channel 57 and node 5, a size reduction directly comparable to the level-2 wavelet compression output of our study.

[2]`https://www.gnu.org/software/gzip/`

Consequently, the obvious choice with the least overhead in terms of levels is the least levels of decomposition.

4.4 Final Remarks

We have studied whether DWT is a suitable method to compress and denoise RSSI time series in a wireless network. We are aiming to communicate the state of a channel from the perspective of a single node among the WSN nodes, in a new compressed and denoised version. We have examined the suitability of different wavelet filters and thresholding methods aiming to compress while preserving the noise patterns and find an appropriate compression level.

Since the new cross–correlation results match the ones produced in Chapter 3 we demonstrated that not only the DWT is a good fit for conveying the signal and its characteristics in a compressed and denoised form, but also that the applied 'Soft' Heuristic SURE thresholding technique as well as the deployed mother wavelet 'DB2' were a good fit for our data.

Chapter 5

Future Directions

5.1 Future Directions

Given the results of this thesis, we note the emergence of several technologies whose synergy could accelerate the deployment of dynamic wireless capacity management schemes. These technologies can make use of approaches such as the time series compression of interference patterns outlined in this thesis.

5.1.1 Spectrum sensing

The emergence of Software Defined Radios (SDRs) has allowed the development of distributed spectrum analyzers [47], providing a comprehensive view of the spectrum use (including non-communication interference sources). We speculate that the features of SDRs will be integrated in wireless access points in the near future. Access points endowed with such capabilities, allow them to switch between serving traffic, to, during idle periods, sample the background noise. A minimal form of spectrum sensing is also possible by legacy devices if access to Received Signal Strength Indicator (RSSI) values is supported, on which the work presented in this thesis is based. For example, almost all inexpensive low-frequency (sub-GHz) RF transceivers allow RSSI values to be sampled. Yet another, coarser and indirect indicator of spectrum condition, is the frame/packet error statistics collected by even the least flexible legacy devices (as interface statistics).

5.1.2 Cloud processing

The connectivity to the wired Internet by access points (as well as, indirectly, by C, and P) devices allows significant amounts of data collected by spectrum sensing to be shipped over for processing in the cloud. Trends and location-specific behavior of the interference and load demands can then be analyzed by computationally-intensive algorithms. In other words, the limited in-situ processing is overcome by off-site cloud-based processing. The idea of using the same cloud-based infrastructure to control the network has been an open research direction in 5G networks [52]. We augment this by (a) including sensing of the spectrum to ascertain the presence of multiple protocol device and sources of interference and (b) assume that legacy devices cannot form part of the set of controllable devices, and hence it is up to other, more capable) devices to infer the behavior of co-located legacy devices.

5.1.3 Web services

While usually deployed as cloud-based application themselves, additional web-based services can assist in augmenting the decision making process. For example, WiFi Service Set Identifiers (SSIDs) mapped to geographic locations (such as `wigle.net`) or live traffic updates, provide, correspondingly, approximate information about the spatial separation of APs and a basis for anticipating residential data traffic load fluctuations. Moreover, persistent non-communication interference sources can be localized and their locations related to map coordinates [30]. While this does not imply a mitigation strategy, it can enable actions outside the automated network control. Another related example is a database service for area-specific white spaces demonstrated in SenseLess [46], geared towards non-ISM cognitive networks.

5.1.4 Interference Reaction/Mitigation

Current strategies to handle interference include *interference alignment* which is a transmission strategy relying on the coordination of multiple transmitters so that their mutual interference aligns at the receivers, and promises to improve the network's throughput [4]. Such techniques set as their objective to approximate the maximum degrees of freedom, also known as the channel's maximum multiplexing gain. Nevertheless, there are still open issues that need to be considered, such as realistic propagation environments, and the role of channel state information at the transmitter, and most importantly for our approach, the practicality of interference alignment in large networks [4].

Krishnamachari and Varanasi [33] study systems characterized by multi-user interference channels with single or multiple antennas at each node and develop an interference alignment scheme where there is no global channel knowledge in the network, but where each receiver knows its channels from all the transmitters and broadcasts a quantized version of it to all the other terminals, at a rate that scales sufficiently fast with the power constraint on the nodes. The quantized channel estimates are treated as being perfect and it is shown that they are indeed sufficient to attain the same sum degrees of freedom as the interference alignment implementation utilizing perfect channel state information at all the nodes. Significant is also the observation by Jafar [29], that statistical knowledge of channel autocorrelation structure alone is sufficient for interference alignment and, to this end, an alternative to CSI.

Another philosophy is that of embracing and exploiting interference. For example, Katti et al. [31] show that by combining physical-layer and network-layer information, network capacity can be increased. Their analog network coding scheme actually encourages strategically picked senders to interfere. Instead of forwarding packets, it suggests that routers forward the interfering signals, so that the destination leverages network-level information to cancel the interference and recover the signal destined to it. On a different tangent of embracing interference is the work by Boers et al. [6] which essentially proposes a MAC coordination scheme that takes into consideration the temporal behavior of interference patterns and aims to steer transmissions around them. Their approach simulates a pattern-aware MAC (PA-MAC) and their results include improved packet reception rates in both single and multi-hop environments at the cost of increased latency.

5.2 Conclusions

Given the reviewed literature regarding the future directions outlined, we identify a relative lack of work and, hence, a need to address the following technical issues:

1. Developing techniques to determine when, and for how long, nodes can take time away from their regular business of handling traffic and instead sensing the spectrum, i.e., a form of scheduling the spectrum analysis tasks with minimal impact on the regular operation of the networks,

2. Developing techniques that allow spectrum sensing data acquired from different nodes to be temporally aligned correctly and referenced back to natural time, despite the lack of strongly synchronized clocks,

3. Developing tools that will allow us to quickly identify correlations in interference patterns, both for determining the origin of the interference, and for guiding nodes to follow similar mitigation strategies,

4. Developing simple metrics and models for quantifying the CTI over a broad set of protocols, such that they can be used to capture cost metrics useful to resource management optimization decisions.

What we have contributed towards in this thesis is a means to answer question 2 (with a new synchronization scheme), and, to a certain extent question 4 via the cross-correlation metrics over time series we constructed from their wavelet coefficients.

Bibliography

[1] E. Akhmetshina, P. Gburzynski, and F. Vizeacoumar. Picos: A tiny operating system for extremely small embedded platforms. In *Las Vegas*, pages 116–122, 2003.

[2] I. F. Akyildiz, W. Lee, M. C. Vuran, and S. Mohanty. Next generation/-dynamic spectrum access/cognitive radio wireless networks: A survey. *Computer Networks*, 50(13):2127 – 2159, 2006.

[3] I. F. Akyildiz, B. F. Lo, and R. Balakrishnan. Cooperative spectrum sensing in cognitive radio networks: A survey. *Phys. Commun.*, 4(1):40–62, March 2011.

[4] O. El Ayach, S. W. Peters, and R. W. Heath. The practical challenges of interference alignment. *IEEE Wireless Communications*, 20(1):35–42, February 2013.

[5] N. M. Boers, I. Nikolaidis, and P. Gburzynski. Impulsive interference avoidance in dense wireless sensor networks. In *Proceedings of the 11th international conference on Ad-hoc, Mobile, and Wireless Networks*, ADHOC-NOW'12, pages 167–180, Berlin, Heidelberg, 2012. Springer-Verlag.

[6] N. M. Boers, I. Nikolaidis, and P. Gburzynski. *Impulsive Interference Avoidance in Dense Wireless Sensor Networks*, pages 167–180. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012.

[7] N. M. Boers, I. Nikolaidis, and P. Gburzynski. Sampling and classifying interference patterns in a wireless sensor network. *ACM Trans. Sen. Netw.*, 9(1):2:1–2:19, November 2012.

[8] N.M. Boers, D. Chodos, J. Huang, P. Gburzynski, I. Nikolaidis, and E. Stroulia. The smart condo: visualizing independent living environments in a virtual world. In *Pervasive Computing Technologies for Healthcare, 2009. PervasiveHealth 2009. 3rd International Conference on*, pages 1–8, 2009.

[9] N.M. Boers, I. Nikolaidis, and P. Gburzynski. Patterns in the RSSI traces from an indoor urban environment. In *Computer Aided Modeling, Analysis and Design of Communication Links and Networks (CAMAD), 2010 15th IEEE International Workshop on*, pages 61–65, 2010.

[10] P. Bourke. Cross correlation,autocorrelation , 2d pattern identification. `http://paulbourke.net/miscellaneous/correlate/`, 1996. [Online; accessed 2015].

[11] D. Burrus. The internet of things is far bigger than anyone realizes. `https://www.wired.com/insights/2014/11/the-internet-of-things-bigger/`, 2017. [Online; accessed 2017-04-10].

[12] M. Centenaro, L. Vangelista, A. Zanella, and M. Zorzi. Long-range communications in unlicensed bands: the rising stars in the iot and smart city scenarios. *CoRR*, abs/1510.00620, 2015.

[13] S. G. Chang, B. Yu, and M. Vetterli. Adaptive wavelet thresholding for image denoising and compression. *IEEE Transactions on Image Processing*, 9(9):1532–1546, September 2000.

[14] J. K. Chen, G. de Veciana, and T. S. Rappaport. Improved measurement-based frequency allocation algorithms for wireless networks. In *IEEE GLOBECOM 2007*, pages 4790–4795, Nov 2007.

[15] S. Chieochan, E. Hossain, and J. Diamond. Channel assignment schemes for infrastructure-based 802.11 WLANs: A survey. *IEEE Communications Surveys Tutorials*, 12(1):124–136, First 2010.

[16] Y. Chompusri, K. Dejhan, and S. Yimman. Mother wavelet selecting method for selective mapping technique ECG compression. In *Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON), 2012 9th International Conference on*, pages 1–4, May 2012.

[17] Cisco. Radio resource management under unified wireless networks. Technical Report 71113, Cisco Systems, May 2010.

[18] R. R. Coifman and N. Saito. Selection of best bases for classification and regression. In *Information Theory and Statistics, 1994. Proceedings., 1994 IEEE-IMS Workshop on*, pages 51–, Oct 1994.

[19] R. R. Coifman and M. V. Wickerhauser. Entropy-based algorithms for best basis selection. *IEEE Transactions on Information Theory*, 38(2):713–718, March 1992.

[20] I. Daubechies. Orthonormal bases of compactly supported wavelets. *Communications on Pure and Applied Mathematics*, 41(7):909–996, 1988.

[21] D. L. Donoho. De-noising by Soft-thresholding. *IEEE Trans. Inf. Theor.*, 41(3):613–627, May 1995.

[22] D. L. Donoho, M. Vetterli, R. A. DeVore, and I. Daubechies. Data compression and harmonic analysis. *IEEE Transactions on Information Theory*, 44(6):2435–2476, October 1998.

[23] David L. Donoho and Iain M. Johnstone. Ideal denoising in an orthonormal basis chosen from a library of bases. *Comptes Rendus Acad. Sci., Ser. I*, 319:1317–1322, 1994.

[24] David L. Donoho and Iain M. Johnstone. Minimax estimation via wavelet shrinkage. *The Annals of Statistics*, 26(3):879–921, 1998.

[25] W. A. Gardner and C. M. Spooner. Signal interception: performance advantages of cyclic-feature detectors. *IEEE Trans. Communications*, 40(1):149–159, 1992.

[26] A. Graps. An Introduction to Wavelets. *IEEE Comput. Sci. Eng.*, 2(2):50–61, June 1995.

[27] M. Haidar, R. Akl, H. Al-Rizzo, and Y. Chan. Channel assignment and load distribution in a power-managed WLAN. In *2007 IEEE 18th International Symposium on Personal, Indoor and Mobile Radio Communications*, pages 1–5, September 2007.

[28] A. Hithnawi, S. Li, H. Shafagh, J. Gross, and S. Duquennoy. Crosszig: Combating cross-technology interference in low-power wireless networks. In *2016 15th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*, pages 1–12, April 2016.

[29] S. A. Jafar. Exploiting channel correlations - simple interference alignment schemes with no CSIT. In *GLOBECOM*, 2010.

[30] K. Joshi, S. Hong, and S. Katti. Pinpoint: Localizing interfering radios. In *10th USENIX Symposium on Networked Systems Design and Implementation (NSDI 13)*, pages 241–253, Lombard, IL, 2013. USENIX.

[31] S. Katti, S. Gollakota, and D. Katabi. Embracing wireless interference: Analog network coding. *Comput. Commun. Rev.*, 37(4):397–408, August 2007.

[32] I. Katzela and M. Naghshineh. Channel assignment schemes for cellular mobile telecommunication systems: a comprehensive survey. *IEEE Personal Communications*, 3(3):10–31, Jun 1996.

[33] R. T. Krishnamachari and M. K. Varanasi. Interference alignment under limited feedback for MIMO interference channels. In *2010 IEEE International Symposium on Information Theory*, pages 619–623, June 2010.

[34] L-Com. Advantages and disadvantages of ism band frequencies. `http://www.l-com.com/content/Article.aspx?Type=N&ID=10421`, 2013. [Online; accessed 2017-04-10].

[35] L-Com. What's in an RF front end? `http://www.eetimes.com/document.asp?doc_id=1276331`, 2017. [Online; accessed 2017-04-10].

[36] T. Le, C. Szepesvári, and R. Zheng. Sequential learning for multi-channel wireless network monitoring with channel switching costs. *IEEE Trans. Signal Processing*, 62(22):5919–5929, 2014.

[37] H. Lee, A. Cerpa, and P. Levis. Improving wireless simulation through noise modeling. In *Proceedings of the 6th international conference on Information processing in sensor networks*, IPSN '07, pages 21–30, New York, NY, USA, 2007. ACM.

[38] J. Lehtomäki. *Analysis of Energy Based Signal Detection*. PhD thesis, University of Oulu, Faculty of Technology, Department of Electrical and Information Engineering, Oulu, Finland, November 2005.

[39] K. K. Leung and B. J. Kim. Frequency assignment for IEEE 802.11 wireless networks. In *Vehicular Technology Conference, 2003. VTC 2003-Fall. 2003 IEEE 58th*, volume 3, pages 1422–1426 Vol.3, Oct 2003.

[40] P. Mahonen, J. Riihijarvi, and M. Petrova. Automatic channel allocation for small wireless local area networks using graph colouring algorithm approach. In *PIMRC 2004*, volume 1, pages 536–539 Vol.1, Sept 2004.

[41] S. G. Mallat. A theory for multiresolution signal decomposition: the wavelet representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(7):674–693, July 1989.

[42] Mathworks MATLAB. Wavelet toolbox. `https://www.mathworks.com/products/wavelet.html`, 2017. [Online; accessed 2017-04-10].

[43] A. Mishra, V. Brik, S. Banerjee, A. Srinivasan, and W. Arbaugh. A client-driven approach for channel management in wireless LANs. In *Proceedings IEEE INFOCOM 2006. 25TH IEEE International Conference on Computer Communications*, pages 1–12, April 2006.

[44] A. Mishra, V. Shrivastava, S. Banerjee, and W. Arbaugh. Partially overlapped channels not considered harmful. *SIGMETRICS Perform. Eval. Rev.*, 34(1):63–74, June 2006.

[45] W. G. Morsi and M. E. El-Hawary. The most suitable mother wavelet for steady-state power system distorted waveforms. In *Electrical and Computer Engineering, 2008. CCECE 2008. Canadian Conference on*, pages 000017–000022, May 2008.

[46] R. Murty, R. Chandra, T. Moscibroda, and P. Bahl. SenseLess: A database-driven white spaces network. *IEEE Transactions on Mobile Computing*, 11(2):189–203, February 2012.

[47] J. Naganawa, H. Kim, S. Saruwatari, H. Onaga, and H. Morikawa. Distributed spectrum sensing utilizing heterogeneous wireless devices and measurement equipment. In *New Frontiers in Dynamic Spectrum Access Networks (DySPAN), 2011 IEEE Symposium on*, pages 173–184, May 2011.

[48] W. K. Ngui, M. S. Leong, L. M. Hee, and A. M. Abdelrhman. Wavelet Analysis: Mother Wavelet Selection Methods. *Applied Mechanics and Materials*, 393:953–958, 2013.

[49] Olsonet. Platform for rd in sensor networking. `http://www.olsonet.com/Documents/emspcc11.pdf`, 2008. [Online; accessed 2013].

[50] M. Rahul. Iot applications with examples. `http://internetofthingswiki.com/iot-applications-examples/541/`, 2015. [Online; accessed 2017-04-10].

[51] O. Rioul and M. Vetterli. Wavelets and signal processing. *IEEE Signal Processing Magazine*, 8(4):14–38, October 1991.

[52] P. Rost, C. J. Bernardos, A. D. Domenico, M. D. Girolamo, M. Lalam, A. Maeder, D. Sabella, and D. Wbben. Cloud technologies for flexible 5G radio access networks. *IEEE Comm. Magazine*, 52(5):68–76, May 2014.

[53] B. N. Singh and A. K. Tiwari. Optimal Selection of Wavelet Basis Function Applied to ECG Signal Denoising. *Digit. Signal Process.*, 16(3):275–287, May 2006.

[54] C. M. Stein. Estimation of the Mean of a Multivariate Normal Distribution. *The Annals of Statistics*, 9(6):1135–1151, 1981.

[55] P. P. C. Tsui and O. A. Basir. Wavelet basis selection and feature extraction for shift invariant ultrasound foreign body classification. *Ultrasonics*, 45(14):1 – 14, 2006.

[56] M. Vetterli. Wavelets, approximation, and compression. *IEEE Signal Processing Magazine*, 18(5):59–73, September 2001.

[57] A. Vlachaki, I. Nikolaidis, and J. J. Harms. A study of channel classification agreement in urban wireless sensor network environments. In O. Postolache, M.Sinderen, F.H. Ali, and C. Benavente-Peces, editors, *SENSORNETS 2014*, pages 249–259. SciTePress, 2014.

[58] A. Vlachaki, I. Nikolaidis, and J. J. Harms. Wavelet-based analysis of interference in WSNs. In *41st IEEE Conference on Local Computer Networks (LCN)*, November 2016 (to appear).

[59] M. Yu, H. Luo, and K. K. Leung. A dynamic radio resource management technique for multiple APs in WLANs. *IEEE Transactions on Wireless Communications*, 5(7):1910–1919, July 2006.

Appendix A

Algorithm 1 RSSI Trace Synchronization

```
for every channel do
  for every node do
    $RSSI$ {Load RSSI trace values}
    $Timestamps$ {Load timestamp values}
    $EndTimes = Timestamps(175000)$ {Last timestamp values}
  end for
  $[minTm, idxMin] = min(EndTimes)$ {Find the fastest node (idxMin)}
  $referenceNode$ {Node with the min ending timestamp }
  $referenceindices = [1...175000]${Reference node indices initialization}
  $referenceNodeTimestamps$ {Fetch raw values of timestamps of the
  fastest node }
  $referenceNodeValues${Fetch raw values of RSSI of the fastest node }
  for every node do
    $searchValue = minTm$ {Initialization of the global clock end time,
    as the of the fastest node}
    $RSSI$ {Load RSSI trace values of the current node}
    $f = Timestamps$ {Load timestamp values of the current node}
    $tmp = abs(f - searchVal)$ {Absolute value of the difference between
    all the timestamps and the search value}
    $[closest, closestIdx] = min(tmp)$ {Find the end index (closestIdx) for
    the current node.}
    $realEndIdx(node) = closestIdx$ {Number of samples to be kept for
    the current node.}
    $allTimestamps(node, 1...closestIdx) = timestamps(1...closestIdx)$
    $allValues(node, 1...closestIdx) = RSSINodeValues(1...closestIdx)$
    $idxVector = [1..closestIdx]$
    for i=1...idxVectorLength do
      $finalNewIdxVec = round(\frac{length(referenceindices)*i}{length(idxVector)})$
      {'Map' the indices of the current node to the ones of the reference
      node indices.}
    end for
    $allindices(node, 1...closestIdx) = finalNewIdxVec$ {The correct in-
    dices for all nodes are saved.}
end for
{ Build the new synced RSSI values and indices table.}
$[minIndex, realIdxMin] = min(realEndIdx)$ {min value of realEndIdx
vector holds the minimum value an ending index takes, realIdxMin will
give us the slowest node, the node with the minimum value of an ending
index}
$referenceindicesVector = allindices(realIdxMin, 1...minIndex)$ {referenceindicesVector
contains the indices of the slowest node to which we are going to 'map'
the indices of the rest of the nodes}
for every node do
    $examinedTrace = allindices(node, 1...realEndIdx(node))$ {Retrieve
    the unsynchronized indices vector.}
    $examinedValues = allValues(node, 1...realEndIdx(node))$ {Retrieve
    the unsynchronized RSSI values.}
    for c=1...referenceindicesVectorLength do
      $idxVal = referenceindicesVector(c)$ {For each one of the referen-
      ceindicesVector items that contains the indices of the slowest node}
      $fTrace = examinedTrace$
      $tmp2 = abs(fTrace - idxVal)$ {We take the unsynchronized vector
      and subtract the idxVal of the slowest node. Then we take the
      absolute value of the differences of those index values and find its
      minimum.}
```

```
            $[examIdxVal, examIdx] = min(tmp2)$ {The examIdxVal will hold the closest index to the reference index value we are currently on.}
            $fclosestExam = fTrace(examIdx)$ {According to the index of the vector this minimum value was found we synchronize indices and RSSI values.}
            $allSyncedValues(node, c) = examinedValues(examIdx)$
            $allSyncedindices(node, c) = fclosestExam$
        end for
    end for
end for
```

```
RSSI_A = [25 26 21 23 24 27 29 24] //Node 1
RSSI_B = [32 36 35 34 37 39 31 29] //Node 2
RSSI_C = [41 42 43 44 45 26 24 23] //Node 3
RSSI_D = [ 5 10  9  7  6 18 12  8] //Node 4


A_timestamps=[0.1  2 2.1 4 4.6 5  6  7] //Node 1
B_timestamps=[0.2  1  4  5  6  7  8  9] //Node 2
C_timestamps=[0.3 1.1 2  3 3.8 4 4.5 5]//Node 3
D_timestamps=[0.4 1.3 3  5  7  9 10 11]//Node 4

End_Times = [7 9 5 11]
minTm = 5
idxMin = 3
referenceNode = 3
referenceindices = [1 2 3 4 5 6 7 8]
referenceNodeTimestamps = [0.3 1.1 2 3 3.8 4 4.5 5]
referenceNodeValues = [41 42 43 44 45 26 24 23]



searchValue = 5

tmp =  [4.9  3  2.9 1 0.4 0  1  2]  //Node 1
       [4.8  4   1  0  1  2  3  4]  //Node 2
       [4.7 3.9  3  2 1.2 1 0.5 0]  //Node 3
       [4.6 3.7  2  0  2  4  5  6]  //Node 4

[closest, closestIdx] = [0 6] //Node 1
                        [0 4] //Node 2
                        [0 8] //Node 3
                        [0 4] //Node 4
```

```

realEndIdx =[6  //Node 1
             4  //Node 2
             8  //Node 3
             4] //Node 4

allTimestamps =  [4.9  3  2.9  1  0.4  0   0  0 //Node 1
                  4.8  4   1   0   0   0   0  0 //Node 2
                  4.7 3.9  3   2  1.2  1  0.5 0 //Node 3
                  4.6 3.7  2   0   0   0   0  0]//Node 4


allValues = [25 26 21 23 24 27  0  0  //Node 1
             32 36 35 34  0  0  0  0  //Node 2
             41 42 43 44 45 26 24 23  //Node 3
              5 10  9  7  6  0  0  0] //Node 4

idxVector = [1 2 3 4 5 6 0 0] //Node 1
            [1 2 3 4 0 0 0 0] //Node 2
            [1 2 3 4 5 6 7 8] //Node 3
            [1 2 3 4 0 0 0 0] //Node 4

allindices = [1 3 4 5 7 8 0 0  //Node 1
              2 4 6 8 0 0 0 0  //Node 2
              1 2 3 4 5 6 7 8  //Node 3
              2 4 6 8 0 0 0 0] //Node 4

[minIndex, realIdxMin] = [4 2]

referenceindicesVector = [2 4 6 8]

examinedTrace = [1 3 4 5 7 8 0 0]   //Node 1
                [2 4 6 8 0 0 0 0]   //Node 2
                [1 2 3 4 5 6 7 8]   //Node 3
                [2 4 6 8 0 0 0 0]   //Node 4

examinedValues= [25 26 21 23 24 27  0  0]   //Node 1
                [32 36 35 34 0   0  0  0]   //Node 2
                [41 42 43 44 45 26 24 23]   //Node 3
                [ 5 10  9  7  6  0  0  0]   //Node 4


//Node 1

c=1...4
idxVal = 2 //c=1
         4 //c=2
         6 //c=3
         8 //c=4

tmp2 = [1 1 2 3 5 6] //c=1
       [3 1 0 1 3 4] //c=2
       [5 3 2 1 1 2] //c=3
       [7 5 4 3 1 0] //c=4
```

```

[examIdxVal, examIdx] = [0 1] //c=1
                        [0 3] //c=2
                        [1 4] //c=3
                        [0 6] //c=4

fClosestExam = 1    //c=1
               4    //c=2
               5    //c=3
               8    //c=4

allSyncedValues = 25        //c=1
                  21        //c=2
                  23        //c=3
                  27        //c=4

allSyncedindices = 1        //c=1
                   4        //c=2
                   5        //c=3
                   8        //c=4

//Node 2

c=1...4
idxVal = 2 //c=1
         4 //c=2
         6 //c=3
         8 //c=4

tmp2 = [0 2 4 6] //c=1
       [2 0 2 4] //c=2
       [4 2 0 2] //c=3
       [6 4 2 0] //c=4

[examIdxVal, examIdx] = [0 1] //c=1
                        [0 2] //c=2
                        [0 3] //c=3
                        [0 4] //c=4

fClosestExam =   2  //c=1
                 4  //c=2
                 6  //c=3
                 8  //c=4

allSyncedValues = 32        //c=1
                  36        //c=2
                  35        //c=3
                  34        //c=4

allSyncedindices = 1        //c=1
                   4        //c=2
                   5        //c=3
                   8        //c=4

```

```
//Node 3

c=1...4
idxVal = 2 //c=1
         4 //c=2
         6 //c=3
         8 //c=4

tmp2 = [1 0 1 2 3 4 5 6] //c=1
       [3 2 1 0 1 2 3 4] //c=2
       [5 4 3 2 1 0 1 2] //c=3
       [7 6 5 4 3 2 1 0] //c=4

[examIdxVal, examIdx] = [0 2] //c=1
                        [0 4] //c=2
                        [0 6] //c=3
                        [0 8] //c=4

fClosestExam =   2  //c=1
                 4  //c=2
                 6  //c=3
                 8  //c=4

allSyncedValues = 42        //c=1
                  44        //c=2
                  26        //c=3
                  23        //c=4

allSyncedindices = 2        //c=1
                   4        //c=2
                   6        //c=3
                   8        //c=4


//Node 4

c=1...4
idxVal = 2 //c=1
         4 //c=2
         6 //c=3
         8 //c=4

tmp2 = [0 2 4 6] //c=1
       [2 0 2 4] //c=2
       [4 2 0 2] //c=3
       [6 4 2 0] //c=4

[examIdxVal, examIdx] = [0 1] //c=1
                        [0 2] //c=2
                        [0 3] //c=3
                        [0 4] //c=4

fClosestExam =   2  //c=1
                 4  //c=2
```

```
                  6   //c=3
                  8   //c=4

allSyncedValues =   5         //c=1
                   10         //c=2
                    9         //c=3
                    7         //c=4

allSyncedindices = 2          //c=1
                   4          //c=2
                   6          //c=3
                   8     //c=4
```

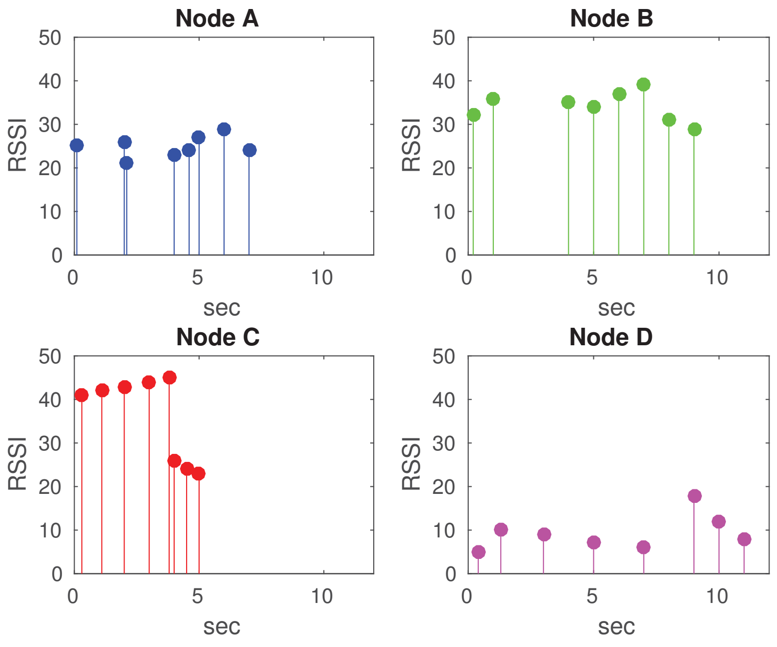


Figure 5.1: Synchronization example, RSSI traces and their timestamps before synchronization.

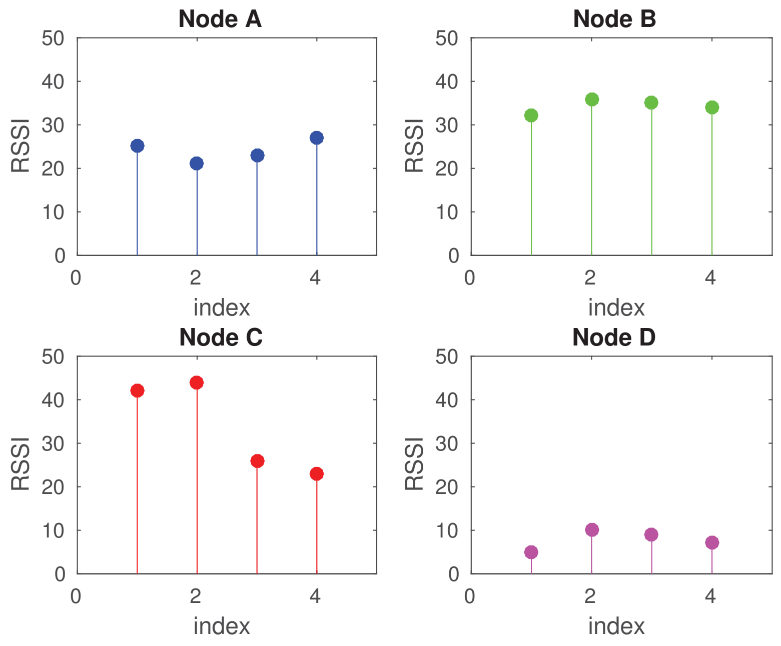


Figure 5.2: Synchronization example, RSSI traces and their indices after synchronization.