Università degli Studi di Padova
Dipartimento di Ingegneria dell'Informazione
Corso di Dottorato in Ingegneria dell'Informazione
Curriculum: Scienza e Tecnologia dell'Informazione
Ciclo: XXXI

Bayesian Learning Strategies in Wireless Networks

Coordinatore:
Ch.mo. Prof. Andrea Neviani

Supervisore:
Ch.mo. Prof. Michele Rossi

Dottoranda:
Maria Scalabrin

Anno Accademico 2017/2018

Abstract

This thesis collects the research works I performed as a Ph.D. candidate, where the common thread running through all the works is Bayesian reasoning with applications in wireless networks. The pivotal role in Bayesian reasoning is inference: reasoning about what we don't know, given what we know. When we make inference about the nature of the world, then we learn new features about the environment within which the agent gains experience, as this is what allows us to benefit from the gathered information, thus adapting to new conditions. As we leverage the gathered information, our belief about the environment should change to reflect our improved knowledge.

This thesis focuses on the probabilistic aspects of information processing with applications to the following topics: Machine learning based network analysis using millimeter-wave narrow-band energy traces; Bayesian forecasting and anomaly detection in vehicular monitoring networks; Online power management strategies for energy harvesting mobile networks; Beam training and data transmission optimization in millimeter-wave vehicular networks. In these research works, we deal with pattern recognition aspects in real-world data via *supervised/unsupervised learning* methods (classification, forecasting and anomaly detection, multi-step ahead prediction via kernel methods). Finally, the mathematical framework of Markov Decision Processes (MDPs), which also serves as the basis for *reinforcement learning*, is introduced, where Partially Observable MDPs use the notion of belief to make decisions about the state of the world in millimeter-wave vehicular networks.

The goal of this thesis is to investigate the considerable potential of inference from insightful perspectives, detailing the mathematical framework and how Bayesian reasoning conveniently adapts to various research domains in wireless networks.

Sommario

Questa tesi raccoglie i lavori di ricerca svolti durante il mio percorso di dottorato, il cui filo conduttore è dato dal Bayesian reasoning con applicazioni in reti wireless. Il contributo fondamentale dato dal Bayesian reasoning sta nel fare deduzioni: ragionare riguardo a quello che non conosciamo, dato quello che conosciamo. Nel fare deduzioni riguardo alla natura delle cose, impariamo nuove caratteristiche proprie dell'ambiente in cui l'agente fa esperienza, e questo è ciò che ci permette di fare uso dell'informazione acquisita, adattandoci a nuove condizioni. Nel momento in cui facciamo uso dell'informazione acquisita, la nostra convinzione (belief) riguardo allo stato dell'ambiente cambia in modo tale da riflettere la nostra nuova conoscenza.

Questa tesi tratta degli aspetti probabilistici nel processare l'informazione con applicazioni nei seguenti ambiti di ricerca: Machine learning based network analysis using millimeter-wave narrow-band energy traces; Bayesian forecasting and anomaly detection in vehicular monitoring networks; Online power management strategies for energy harvesting mobile networks; Beam-training and data transmission optimization in millimeter-wave vehicular networks. In questi lavori di ricerca studiamo aspetti di riconoscimento di pattern in dati reali attraverso metodi di *supervised/unsupervised learning* (classification, forecasting and anomaly detection, multi-step ahead prediction via kernel methods). Infine, presentiamo il contesto matematico dei Markov Decision Processes (MDPs), il quale sta anche alla base del *reinforcement learning*, dove Partially Observable MDPs utilizzano il concetto probabilistico di convinzione (belief) al fine di prendere decisoni riguardo allo stato dell'ambiente in millimeter-wave vehicular networks.

Lo scopo di questa tesi è di investigare il considerevole potenziale nel fare deduzioni, andando a dettagliare il contesto matematico e come il modello probabilistico dato dal Bayesian reasoning si possa adattare agevolmente a vari ambiti di ricerca con applicazioni in reti wireless.

Acknowledgements

This thesis would not have been possible without the help of many people.

First, I would like to express my sincere gratitude to my Ph.D. advisor Prof. Michele Rossi for the continuous support of my Ph.D, for his patience, motivation, and true passion for research. His guidance helped me in all the hard working times of research, with constructive criticism, precious advise, and moral support. This gave me uncountable opportunities for professional and personal growth. My sincere gratitude also goes to Prof. Nicolò Michelusi, who helped me to wisely widen my research skills from various stimulating perspectives. His enthusiastic attitude towards research has been inspirational.

I would like to thank my friends and colleagues all over the world for sharing the most challenging, exciting, and intense moments of the last years together.

I would like to thank my family: my parents, Anna, Roberto, and siblings, Sara, Giacomo, Elia, and Luca, for always being part of myself, wherever I am.

Finally, last but not least, I would like to say a special thank to Emanuele, my ray of sunshine. Because I owe it all to you.

To Emanuele

"Nothing in life is to be feared, it is only to be understood.
Now is the time to understand more, so that we may fear less."
Maria Skłodowska-Curie

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation and Objectives

In the last decades we have seen a growing interest in Machine Learning. In the broadest sense, this field aims to learn new features about the environment within which the agent gains experience. How gathered information is processed leads to the development of learning algorithms, i.e., how to process the collected data and deal with the uncertainty about the nature of the world.

This thesis focuses on the probabilistic aspects of information processing with applications in wireless networks via Bayesian reasoning. The pivotal role in Bayesian reasoning is inference: reasoning about what we don't know, given what we know. When we make inference about the nature of the world, then we learn new features about the environment within which the agent gains experience, as this is what allows us to benefit from the gathered information, thus adapting to new conditions. In this respect, Bayesian probability theory provides a mathematical framework for reasoning about the nature of the world in an effective, elegant, and precise fashion. When we make inference about the nature of the world, what we need is a means of discussing statements that have different levels of uncertainty. In other words, statements that have varying degrees of belief. In addition to modeling these statements within a mathematical framework, we want to be able to process the collected data. As we leverage the gathered information, our belief about the environment should change to reflect our improved knowledge. This is what defines learning.

1.2 Contributions

This thesis collects the research works I performed as a Ph.D. candidate, where the common thread running through all the works is Bayesian reasoning with applications in wireless networks. In the last decades we have seen a growing interest in the design of adaptive models that exploit contextual information to enhance the overall network performance. In contrast to conventional distributed optimization techniques, Machine Learning inspired mechanisms are able to operate in an online fashion, learning the current states of the wireless environment and the network's users, improving the overall network performance over time. This, in turn, enables smarter network decision making, which is essential for most applications in wireless networks, particularly those that require real-time, low latency operations. The use of contextual information is also crucial to our analysis. Next, we provide an overview of the content of this thesis, which is organised in the following topics: Machine learning based network analysis using millimeter-wave narrow-band energy traces; Bayesian forecasting and anomaly detection in vehicular monitoring networks; Online power management strategies for energy harvesting mobile networks; Beam training and data transmission optimization in millimeter-wave vehicular networks. In particular, while Chapters 2,3,4 deal with pattern recognition aspects in real-world data via *supervised/unsupervised learning* methods (classification, forecasting and anomaly detection, multi-step ahead prediction via kernel methods), Chapter 5 is different from the previous ones: here, the mathematical framework of Markov Decision Processes (MDPs), which also serves as the basis for *reinforcement learning*, is introduced, where Partially Observable MDPs use the notion of belief to make decisions about the state of the world in millimeter-wave vehicular networks. Specifically, we are concerned with the problem of finding the optimal actions to take in a given situation in order to maximize a reward (or minimize a cost). Nowadays, reinforcement learning continues to be an active research area of Machine Learning. In this sense, even if Chapter 5 does not exploit state-of-the-art reinforcement solutions (where the agent gains experience while interacting with the environment, without prior knowledge of the exact mathematical model), it paves the way for pioneering research domains in millimeter-wave vehicular networks, where

deep reinforcement learning architectures can be designed to solve real-time optimization problems based on real-world data. Hereinafter, we provide an overview of the content of this thesis, in a chapter-by-chapter manner, detailing the mathematical framework and how Bayesian reasoning conveniently adapts to various research domains in wireless networks.

In Chapter 2, Hidden Markov Models (HMMs) and Explicit Duration HMMs (EDHMMs) are introduced to perform protocol frames classification in millimeter-wave wireless networks. Gaining information from spectrum usage is becoming important to provide smart adaptation capabilities to future network protocol stacks. Issues such as deafness, misaligned antennas, or blockage may severely impact network performance, and their identification is crucial. Despite the complexity of full analytical models, Machine Learning techniques are progressively being considered to improve spectrum usage at higher layers. In this chapter, we design a signal processing technique that uses narrowband physical layer energy traces, obtained from one or multiple channel sniffers. The proposed technique utilizes a combination of template matching and an EDHMM to correctly classify frames, while coping with the non-stationarity of the traces. This leads to a protocol level monitor that does not need to decode the channel at the physical layer, but just infers the type of packets that are exchanged based on sub-sampled energy traces. The performance of this framework is evaluated using off-the-shelf millimeter-wave wireless devices, quantifying its detection performance in the presence of one or multiple sniffers, and assessing the impact of physical layer parameters such as noise power and signal levels. The idea is that different viewpoints of the same channel can provide diverse information and lead to higher decoding accuracies. We remark that our tool is the first automatic classifier of IEEE 802.11ad energy traces for network diagnosis. The uniqueness of our approach prevents direct comparison with earlier work. However, the resulting knowledge is extremely valuable, as it provides useful insights for network planners and administrators.

In Chapter 3, Graphical Models are introduced to design a Bayesian forecasting and anomaly detection framework for vehicular monitoring networks. This problem is tackled through localized and small-size Bayesian Networks (BNs), which are utilized to capture the spatio-temporal relationships underpinning traffic data from nearby road links. A dedicated BN is set up, trained, and tested for each road in the monitored geographical map. The joint prob-

ability distribution between the cause nodes and the effect node in the BN is tracked through a Gaussian Mixture Model (GMM), whose parameters are estimated via Bayesian Variational Inference (BVI) operating on unlabeled data. Optimal forecasting follows from the criterion of Minimum Mean Square Error (MMSE). Moreover, we also perform anomaly detection by devising a probabilistic score associated with the marginal conditional distribution of the effect node. The so obtained GMMs are time-dependent, i.e., several GMMs can be estimated for the same target road for different days of the weeks and/or hours of the day. Also, our framework is distributed, lightweight, and capable of operating in realtime and, in turn, it appears to be a promising candidate to deal with Internet of Things (IoT) applications in large-scale networks, where new data is to be processed on-the-fly. The effectiveness of the proposed framework is tested using a large dataset from a real network deployment, comparing its prediction performance with that of selected regression algorithms from the literature, while also quantifying its anomaly detection capabilities.

In Chapter 4, Gaussian Processes (GPs) are presented within the framework of Energy Cooperation and Model Predictive Control (MPC). The design of self-sustainable Base Station (BS) deployments is addressed in this chapter. We target deployments featuring small BSs with Energy Harvesting (EH) and storage capabilities. These BSs can use ambient energy to serve the local traffic or store it for later use. A dedicated power packet grid is utilized to transfer energy across them, compensating for imbalance in the harvested energy or in the traffic load. Some BSs are *offgrid*, i.e., they can only use the locally harvested energy and that transferred from other BSs, whereas others are *ongrid*, i.e., they can additionally purchase energy from the power grid. Within this setup, an optimization problem is formulated where: harvested energy and traffic processes are estimated (at runtime) at the BSs through GPs, and a MPC framework is devised for the computation of energy allocation and transfer across base stations. The combination of prediction and optimization tools leads to an efficient and online solution that automatically adapts to EH and load dynamics. Numerical results, obtained using real EH and traffic profiles, show substantial improvements with respect to the case where the optimization is carried out without predicting future system dynamics. The main improvements are in the outage probability (zero in most cases), and in the amount of energy purchased from the power grid, that is more than halved

for the same served load.

In Chapter 5, Partially Observable MDPs (POMDPs) use the notion of belief to make decisions about the state of the world in millimeter-wave vehicular networks. Future vehicular communication networks call for new solutions to support their capacity demands, by leveraging the potential of the millimeter-wave (mm-wave) spectrum. Mobility, in particular, poses severe challenges in their design, and as such shall be accounted for. A key question in mm-wave vehicular networks is how to optimize the trade-off between directive Data Transmission (DT) and directional Beam Training (BT), which enables it. In this chapter, learning tools are investigated to optimize this trade-off. In the proposed scenario, a Base Station (BS) uses BT to establish a mm-wave directive link towards a Mobile User (MU) moving along a road. To control the BT/DT trade-off, a POMDP is formulated, where the system state corresponds to the position of the MU within the road link. The goal is to maximize the number of bits delivered by the BS to the MU over the communication session, under a power constraint. In order to address the resource constraints in our problem, we propose a Lagrangian method, and an online algorithm to optimize the Lagrangian variable based on the target cost constraint. Specifically, the Lagrangian variable is properly tuned within the main loop of the routine according to a gradient descent technique. Numerical results reveal that common-sense heuristic schemes cannot achieve the performance of the optimal policies, which take advantage of the belief update mechanism and provide adaptive BT/DT procedures according to the current belief, being able to maximize the transmission rate while showing robustness against BT errors.

Final remarks and considerations are given in Chapter 6.

Chapter 2

Machine Learning Based Network Analysis using Millimeter-Wave Narrow-Band Energy Traces

Next-generation wireless networks promise to provide extremely high data rates, especially exploiting the so-called millimeter-wave frequency range. Gaining information from spectrum usage is becoming important to provide smart adaptation capabilities to future network protocol stacks. Issues such as deafness, misaligned antennas, or blockage may severely impact network performance, and their identification is crucial. Despite the complexity of full analytical models, Machine Learning techniques are progressively being considered to improve spectrum usage at higher layers. In this chapter, we design a signal processing technique that uses narrowband physical layer energy traces, obtained from one or multiple channel sniffers. The proposed technique utilizes a combination of template matching and an Explicit Duration Hidden Markov Model (EDHMM) to correctly classify frames, while coping with the non-stationarity of the traces. This leads to a protocol level monitor that does not need to decode the channel at the physical layer, but just infers the type of packets that are exchanged based on sub-sampled energy traces. The performance of this framework is evaluated using off-the-shelf millimeter-wave wireless devices, quantifying its detection performance in the presence of one or multiple sniffers, and assessing the impact of physical layer parameters such as noise power and signal levels.

2.1 Introduction

Next-generation wireless networks are called to provide extremely high data rates, especially exploiting the so-called millimeter-wave frequency range [1]. Applications and services will benefit from these high rates and the radio spectrum will become more and more densely utilized. As wireless networks turn into increasingly complex systems, accurate and scalable analytical models to characterize their behavior are not yet available and very challenging to obtain. Instead, a promising approach is provided by machine learning tools that learn from data [2–4]. Gaining information from spectrum usage is deemed important to provide smart adaptation capabilities to future network protocol stacks. Possible applications include: (i) channel diagnosis: detect communication problems such as a link blockage, (ii) Quality of Service (QoS) tracking and adaptation, i.e., efficiently manage channel resources according to the detected energy level, (iii) information security: discovery of malicious signaling messages, etc.

As for the millimeter-wave (mm-wave) channel, its directional nature results in communication issues that strongly impact higher layers, but which are hard to identify without detailed information of the underlying physical layer effects. This includes, e.g., deafness [5], misaligned antennas, and link blockage. Commercial Off-The-Shelf (COTS) devices are typically a black box regarding this physical layer information. As a result, troubleshooting COTS-based, real-world mm-wave network deployments often translates into time-consuming "trial-and-error" analysis. While understanding performance issues in such deployments is challenging [6–8], the resulting knowledge is extremely valuable. It provides useful insights for network planners and administrators. For instance, a missing acknowledgment after a data packet hints at a deafness issue, overlapping packet frames suggest a collision, and so on. However, gaining such insights from a COTS node that forms part of the network is virtually impossible. On top of the aforementioned lack of lower-layer access, a single node would be restricted to its particular point of view—the directivity of the communication limits the insights that one could gain. To prevent this, we need to capture and compare the behavior of the network from multiple viewpoints. Given the extreme bandwidth available in mm-wave com-

munications (e.g., 2 GHz per channel in the 60 GHz band), this requires an inordinate amount of data processing, and thus would be highly challenging.

In this chapter, we start filling the above identified gaps through the design and evaluation of an automatic tool for mm-wave channel analysis based on COTS hardware. Specifically, the tool uses machine learning techniques to infer protocol-level details such as packet types, their energy level and duration, and can help detect performance bottlenecks in 60 GHz networks using *narrowband* physical layer energy traces from one or multiple (omnidirectional) sniffers. That is, we do not record and decode the full communication but only require the energy level that the sniffers receive.

Our key contribution is developing a machine learning framework that correctly classifies the transmitted frame types (data, acknowledgements and training sequences) and can help infer network issues. This is far from trivial due to (a) the non-stationarity of the traces and (b) the complexity of the IEEE 802.11ad protocol [9]. To address (a), we dynamically update the parameters of the underlying machine learning model such that it adjusts to variations in the received energy level due to, e.g., node movement. Regarding (b), we use a combination of template matching and an Explicit Duration Hidden Markov Model (EDHMM) to correctly classify frames. The core idea of our approach is also applicable to networks operating at lower frequencies such as IEEE 802.11ac. Still, in this chapter we focus on the mm-wave case, which is more challenging due to the use of directional antennas and the large bandwidth. Since we do not need to decode any of the data, our approach preserves privacy, works regardless of whether the network uses encryption, and does not require accurate time/frequency synchronization. As a result, the proposed technique is simple yet highly effective. Our contributions are summarized as follows:

- We design a machine learning framework based on template matching and an EDHMM to automatically infer protocol layer information, e.g., transmitted packet types, their energy and duration, in 60 GHz networks. The main challenge lies in the variability of the traces and in the complexity of identifying the structural elements in the traces given their noisiness, aperiodicity, and unpredictable behavior. Here, this is solved through a combination of statistical and machine learning techniques.

- We introduce a time-adaptive learning mechanism to cope with the non-stationarity of energy traces due to gain control adjustments and node movement. This run-time adaptation is barely explored in specialized work in the field of statistics but is critical for our approach. It also sets our scheme apart from existing work based on simple clustering or thresholding, which is highly sensitive to non-stationary behavior and thus often fails.

- We extend the learning framework to *jointly* process mm-wave channel traces from multiple time-synchronized sniffers. The idea is that different viewpoints of the same channel can provide diverse information and lead to higher decoding accuracies.

- We evaluate our approach in an extensive measurement campaign using COTS 60 GHz hardware to analyze its performance in a range of practical scenarios. Besides, we numerically quantify the ability of the framework to correctly identify protocol sequences (beacon pairs, data and acknowledgment frames) from single and multiple channel sniffers, looking at the impact of channel noise and its distribution across different sniffers.

This chapter is organized as follows. The related work is surveyed in Section 2.2. In Section 2.3, we discuss typical IEEE 802.11ad energy traces. Some mathematical background on the standard HMM and the extended EDHMM framework is provided in Section 2.4. The machine learning framework is introduced in Sections 2.5, 2.6, 2.7 and 2.8. In Section 2.9, this framework is generalized to perform decoding from multiple sniffers and a mm-wave channel trace generator that helps to evaluate the accuracy of our approach is presented in Section 2.10. We finally evaluate the proposed technique using experimental and simulated energy traces in Section 4.5 and provide some concluding remarks in Section 2.12.

2.2 Related work

In the following, we give an overview of performance analysis and troubleshooting in mm-wave networking. As sketched in Section 2.1, mm-wave

networks suffer from high path-loss and high absorption. To overcome this, nodes typically use directional antennas and Line-Of-Sight (LOS) paths. However, this makes links very susceptible to blockage. State-of-the-art work in this field [10–12] focuses on correctly identifying such blockage at the nodes involved in the communication, and reacting in a timely manner. For instance, BeamSpy [11] measures the set of available paths between a transmitter and a receiver. This "path skeleton" serves as a reference whenever blockage occurs—the nodes compute which of the paths in the skeleton is most likely to be unaffected by the blockage and steer their antennas accordingly. As a result, BeamSpy can avoid costly beam steering overhead. Further, earlier work by the same authors [12] looks into differentiating device movement from blockage based on Received Signal Strength (RSS) measurements. This is key to ensure that nodes react correctly when links degrade. Similarly, MOCA [10] transmits a very short control message to assess the link state. If the transmitter does not obtain a reply, it assumes that the antennas are misaligned. Otherwise, it adapts the Modulation and Coding Scheme (MCS) according to the current channel state. All of the above approaches aim at improving the performance of mm-wave networks. In contrast, our work *troubleshoots* the operation of such approaches and is thus orthogonal to them. While BeamSpy and MOCA also try to identify specific issues in the communication, they are constrained to the specific "viewpoint" of a certain node. Our framework runs on one or more external sniffers which we can place at multiple locations, thus providing richer insights. Earlier work proposes an equivalent concept based on external sniffers. However, such approaches typically consider lower frequency bands, and focus on security issues [13, 14] such as realizing an Intrusion Detection System (IDS). The key difference to our study is that such security sniffers are designed to continuously operate along with the network, thus increasing the complexity of the deployment. In contrast, our tool does not need to be part of the network, and can be used on-demand only. Hence, we do not add complexity to the network. Moreover, we focus on performance issues in directional wireless networks while the above work deals with security in the omni-directional case. However, [14] and references therein also deal with raw physical layer data, similarly to our case. Specifically, they suggest overhearing the communication and jamming unwanted packets based on, for instance, header information. Our narrow band sniffer for wide band

signals also overhears the communication but does not (and in fact cannot) decode any preambles and headers to identify the packets. Instead, we use machine learning on the timing of frames from simple energy traces to obtain the information required for network analysis.

In recent years, machine learning techniques are increasingly being used to address high-dimensional problems with multiple unpredictable factors, e.g., for traffic classification, and previous papers typically use it after demodulating and decoding frames at the physical layer [15, 16] (although these approaches are not tailored to IEEE 802.11ad). In contrast, our framework uses machine learning on the raw physical layer trace. Thus, it eliminates the need for the above operations, which are particularly complex and resource intensive in mm-wave networks and would require a sufficiently wide band channel sniffer. However, this poses additional challenges, such as identifying frames. In this chapter, we provide solutions to these challenges, which sets us apart from existing work. On top of the latest trends in wireless communications, machine learning based solutions for spectrum sensing/sharing in Cognitive Radio (CR) represent a promising approach for improving the utilization of the radio electromagnetic spectrum [17, 18]. To promote this, the Defence Advance Research Projects Agency (DARPA) [19] intends to develop technologies for extensive spectrum sensing/sharing, both in the Radio Frequency Machine Learning Systems (RFMLS) program [2] as well as in another major DARPA effort known as the Spectrum Collaboration Challenge (SC2) [3], which is regarded as the first-of-its-kind collaborative machine-learning competition to overcome spectrum scarcity. Also the National Science Foundation (NSF) [4] is promoting projects to leverage machine learning solutions in CR. In the literature, automatic network recognition offers a promising framework for the integration of cognitive concepts at the network layer, bearing similarities with the mm-wave channel analyzer proposed in Section 2.5. In [20], the authors address the problem of automatic classification of technologies, with particular focus on Wi-Fi vs. Bluetooth recognition. Previous work, as for example [21], has addressed a related problem, allowing cooperative spectrum sensing in peer-to-peer cognitive networks by using distributed detection theory [22] for identifying overlapping air interfaces based on time-frequency analysis and feature extraction. The same problem is tackled in [23], where two kinds of neural classifiers are adopted. Again, the authors focus on Wi-Fi

vs. Bluetooth recognition. While this work is related to ours, none of these studies perform protocol analysis from narrow band channel traces. We remark that our tool is the first automatic classifier of IEEE 802.11ad energy traces for network diagnosis. The uniqueness of our approach prevents direct comparison with earlier work.

2.3 A look into IEEE 802.11ad energy traces

The IEEE 802.11ad standard operates at 60 GHz. In this band, propagation conditions are worse than at lower bands, such as at 2.4 or 5 Ghz, which are used by the IEEE 802.11n/ac standards [24]. Specifically, the path loss is much higher at 60 GHz than at 2.4 or 5 GHz. To compensate for this attenuation, IEEE 802.11ad provides a mechanism for the establishment of a directional communication link between a transmitter/receiver pair using a beam training process. As a result of this process, the transmitting station focuses its energy towards the intended receiver. This compensates for the high path loss and reduces the potential interference to other stations that are located nearby.

IEEE 802.11ad divides the channel access into so called Beacon Intervals (BIs). Each BI is split into different access periods, which have different access rules and provide specific functionalities to the stations (STAs) within communication range. A typical BI is composed of a Beacon Header Interval (BHI) and a Data Transmission Interval (DTI). The BHI contains several sub-intervals and is basically used to transmit control messages, such as *beacons* that enable beam training. In the DTI period, STAs exchange data frames either exploiting a contention-based access period or a scheduled service period. The former entails a contention mechanism ("floor acquisition") to acquire the medium, which uses the enhanced distributed coordination function. Conversely, in the scheduled service period, stations access the channel in a contention-free manner.

An example IEEE 802.11ad energy trace corresponding to a data exchange is shown in Fig. 2.1. This trace depicts the start of a typical data burst. The data burst starts with a pair of beacons which contain control information. This pair of beacons is followed by a sequence of data (DATA) packets and

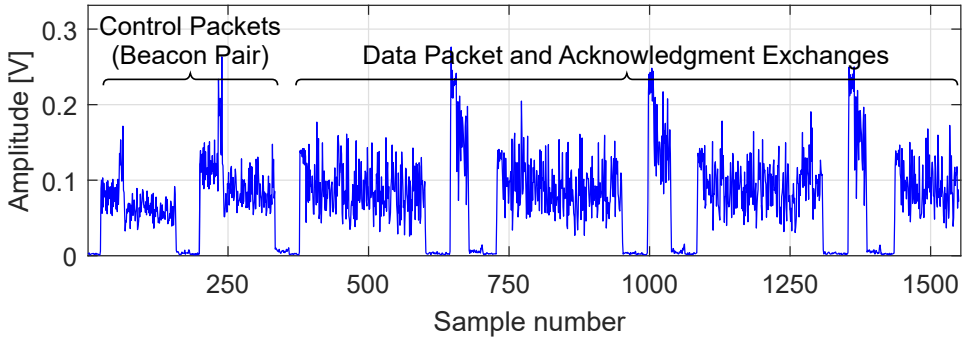


Figure 2.1: Energy trace example of a DATA/ACK burst starting with a pair of beacons.

acknowledgments (ACKs). In general, each DATA packet is followed by a corresponding ACK, which is the shorter packet in the figure and which has a higher energy level. Note that the higher amplitude of ACKs is due to the position of the sniffer, which in this case was near the receiver. The beam training process is composed of the following two phases.

- **Sector Level Sweep (SLS)**. During the SLS, a STA selects a *coarse grain* antenna sector. This phase can be implemented in two ways: 1) through a transmit sector sweep (TXSS), i.e., a STA tries to select the best transmit antenna sector towards a particular receiving STA by transmitting Sector Sweep (SSW) frames using each of its antenna sectors or 2) through a receive sector sweep (RXSS), i.e., a receiving STA trains its receive antenna sector by requesting its peer STA to transmit SSW frames using a fixed antenna pattern, while the receiving STA sweeps across its receive antenna sectors.

- **Beam Refinement (BR)**. To refine the sectors obtained in the SLS phase, multiple mechanisms are used. Basically, the two communicating STAs iteratively search for the optimal alignment starting from the coarse grain sector provided by the SLS. Occasional BR sequences retrain the antenna beams in case of, e.g., mobility, to ensure that both nodes remain in the boresight of each other.

Sequences of control packets are not difficult to identify within IEEE 802.11ad channel traces. The SLS sweeps that are used during the connection setup have 32 different energy levels. The BR sequences, which are used for re-alignement, e.g., when there is a drop in the link quality, have 35 levels [25]. The particular

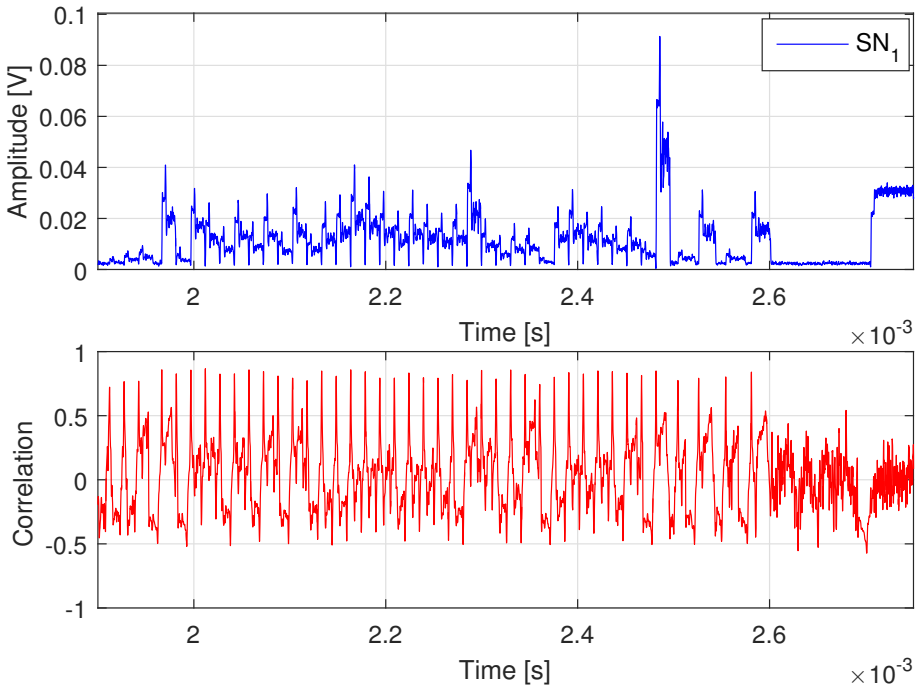


Figure 2.2: Example IEEE 802.11ad beam refinement sequence (top) and correlation coefficient (bottom) with respect to the beacon template. This BR sequence has 35 energy levels.

number of energy levels depends on the number of sectors of the antenna. Existing hardware implements the above number of sectors. An example energy trace corresponding to a BR phase is shown in Fig. 2.2.

In general, it is not difficult to recognize the individual frame types (beacons, DATA, ACKs, and BR sweeps) in the energy traces by visual inspection. This enables one to infer the dynamics of the communication. For instance, a missing acknowledgment after a data packet hints at a deafness issue, overlapping packet frames suggest a collision, and so on. However, while this is visually evident, manually inspecting the energy traces is infeasible given the number of packets when communicating at multi-gigabit-per-second rates. At the same time, recognizing frame types in an automated manner is hard. In this chapter, our goal is to devise and validate a technique for the *automatic identification and labeling of IEEE 802.11ad energy traces*. Notably, BR/SLS sweeps can be reliably identified through a standard pattern matching technique, as we briefly describe next. Each sequence is composed of beacon frames, each having a different energy level, but all of them having the same

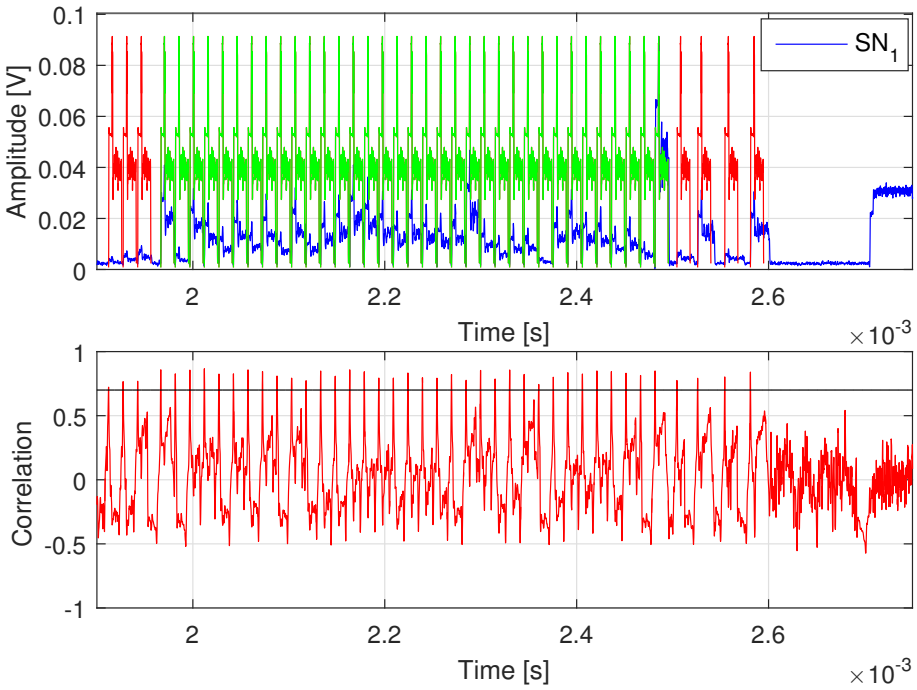


Figure 2.3: Beam Refinement (BR) sequence of Fig. 2.2 (top) and corresponding correlation coefficient (bottom) with respect to the beacon template. A correlation threshold (horizontal line in the bottom plot) is used to single out beacon messages (top graph), whereas the inter-beacon distance reveals whether a beacon is part of the BR sweep. The BR sequence has 35 energy levels and is correctly identified, see the green line in the top plot.

(although noisy) *distinctive shape*. To capture this shape, we obtained a beacon template, that is basically a smoothed out version of the beacons that were measured experimentally. Hence, a standard convolution is performed between the input energy trace and the *beacon template*; for an example see the bottom plot in Fig. 2.2. As we show in Fig. 2.3, setting a proper threshold on the correlation signal allows one to single out the start of each beacon in the original sequence. It is then not difficult to check when exactly 32 or 35 properly spaced energy levels appear in a row and, in turn, detect the SLS/BR sweeps, see again Fig. 2.3. Further details on the template matching procedure are given in Section 2.6, whilst additional results on BR sequence detection are provided in Section 2.11.1.

While the identification of SLS/BR sweeps is doable through simple processing techniques, the characterization of DATA exchange phases is much more complex. In this case, we do not know in advance the duration of DATA

frames. Similarly, we do not know the number of DATA/ACK exchanges in the data burst. There may be missing ACKs and the energy levels of ACKs and DATA frames can be arbitrary, as they depend on the location of the sniffer with respect to the transmitter and the receiver. In addition, the start of each data burst has to be reliably identified, and the start and end points of each frame transmitted therein have to be reliably assessed as well. All of this leads to a sequential estimation problem that is the subject of the work that we expound in the following sections.

2.4 EDHMM preliminaries

Next, some mathematical background on the standard HMM and the extended EDHMM framework is provided as a basis for the machine learning framework, which is introduced in Sections 2.5, 2.6, 2.7 and 2.8. Specifically, we demonstrate how the standard HMM is inadequate for our purpose. Still, we use it to calibrate the initial EDHMM.

We use uppercase and calligraphic fonts for sets, except for $\mathcal{N}(X; \mu, \sigma^2)$, which refers to a Gaussian random variable X with mean μ and variance σ^2. We denote a random sequence of length T by $X_{1:T} = (X_1, \ldots, X_T)$, where the random variable X_t at time index $t \in \{1, \ldots, T\}$ takes values in the set $\mathcal{X}$, with cardinality $|\mathcal{X}|$. Realizations are indicated by lowercase letters, i.e., x_t is the realization of X_t, and with $x_{1:T} = (x_1, \ldots, x_T)$ we denote a sequence of realizations. Vectors are indicated by bold letters, e.g., $\boldsymbol{b}$, and we refer to their elements as $\boldsymbol{b} = [b_1, \ldots, b_K]$, with $|\boldsymbol{b}| = K$. For matrices we use uppercase bold letters, e.g., $\boldsymbol{A} = \{a_{ij}\}$ is a matrix with elements a_{ij}.

Markov models, whose states correspond to observable events, are inadequate to solve our mm-wave channel estimation problem. The reason is that we measure a noisy version of the transmitted energy levels, as they are corrupted by random channel fluctuations. Instead, Hidden Markov Models (HMMs) [26] are a more appropriate tool, as their observations are probabilistic functions of the (hidden) state. Specifically, an HMM is composed of embedded stochastic processes, where an unobservable hidden random process is revealed to the observer through another set of random processes that produce the sequence of observations.

We now consider a data burst and aim to solve the following estimation problem. The observed channel samples in the data burst, $O_{1:T} = (O_1, \ldots, O_T)$, are modeled as a sequence of real-valued random variables corresponding to one of the following basic elements: "1" inter-frame space (IFS), "2" data packet (DATA) and "3" acknowledgement (ACK). Accordingly, the hidden state S_t at time t is a discrete random variable that can take values in the set $\mathcal{S} = \{1, 2, 3\}$. We define $S_{1:T} = (S_1, \ldots, S_T)$ as the sequence of random variables describing the hidden states in the data burst, i.e., $t \in \{1, \ldots, T\}$. Our objective is then to reliably estimate the sequence of hidden states $s_{1:T} = (s_1, \ldots, s_T)$ from observations $o_{1:T} = (o_1, \ldots, o_T)$. The standard HMM makes two basic assumptions regarding the embedded stochastic processes:

A1) The first assumption is that $S_{1:T}$ is a first-order Markov chain, i.e., $P(S_{t+1}|S_1, \ldots, S_t) = P(S_{t+1}|S_t)$. In particular, we have $P(S_{t+1} = j|S_t = i) = a_{ij}$, where $\boldsymbol{A} = \{a_{ij}\}$, $i, j \in \mathcal{S}$, is the single-step transition probability matrix of the HMM.[1]

A2) The second assumption is that the random variable O_t is statistically independent of $(O_1, \ldots, O_{t-1})$.[2]

Moreover, O_t is a probabilistic function of the hidden state S_t, i.e., it obeys a suitable conditional probability $P(O_t|S_t)$ and each random variable O_t can use a private distribution $P(O_t|S_t)$ over the hidden state. We use a Gaussian observation model with $P(O_t|S_t = i) = \mathcal{N}(O_t; \mu_i, \sigma_i^2)$, where μ_i and σ_i^2 specify the mean and the variance of the random variable O_t, given that the hidden state is $i \in \mathcal{S}$. This is known to well approximate the noise distribution for mm-wave channels [27]. For all hidden states $i \in \mathcal{S}$, we collect the parameter pairs $b_i = (\mu_i, \sigma_i^2)$ through vector $\boldsymbol{b} = [b_1, \ldots, b_{|\mathcal{S}|}]$. We define $\boldsymbol{\pi} = [\pi_1, \ldots, \pi_{|\mathcal{S}|}]$, where π_i is the probability that the HMM is in state $i \in \mathcal{S}$ in the first time slot of the burst.

[1] Conversely, in the extended EDHMM, the entire process is not Markovian (memoryless). Instead the process is Markovian only at specified time instants.

[2] Specifically, one observation per state is assumed in the standard HMM model while in the extended EDHMM each state emits a sequence of observations. The length of the sequence while in state $i \in \mathcal{S}$ is determined by the length of time spent in state $i \in \mathcal{S}$, i.e., the duration d. Observations are assumed to be independent of time t, while in state $i \in \mathcal{S}$. Also, in the extended EDHMM, the transition probability a_{ij} is independent of the duration d of state $i \in \mathcal{S}$ and the duration d is only conditioned on the current state $j \in \mathcal{S}$.

The HMM model is described through a further parameter vector $\boldsymbol{\Theta} = [\boldsymbol{\pi}, \boldsymbol{A}, \boldsymbol{b}]$. Its maximum likelihood estimate given a sequence of observations is obtained through the Expectation-Maximization (EM) algorithm [28], which entails two-step iterations. Briefly, initial values for $\boldsymbol{\Theta}$ are chosen, and using assumptions A1 and A2 the posterior distribution for the whole sequence $P(S_{1:T}|O_{1:T}, \boldsymbol{\Theta})$ is computed. Hence, this posterior is used to compute the expected log-likelihood (the Baum's auxiliary function), $Q(\boldsymbol{\Theta}^{\text{new}}, \boldsymbol{\Theta})$, as

$$Q(\boldsymbol{\Theta}^{\text{new}}, \boldsymbol{\Theta}) = \sum_{S_{1:T} \in \mathcal{S}^T} P(S_{1:T}|O_{1:T}, \boldsymbol{\Theta}) \log P(S_{1:T}, O_{1:T}|\boldsymbol{\Theta}^{\text{new}}) \,, \tag{2.1}$$

which is finally maximized with respect to $\boldsymbol{\Theta}^{\text{new}}$, where $\boldsymbol{\Theta}^{\text{new}}$ is the new parameter vector (HMM model) from the EM iteration. This process is repeated until convergence to a local maximum. A proper initialization of $\boldsymbol{\Theta}$ (with particular regard for $\boldsymbol{b}$) is crucial for a good convergence of the EM algorithm. For a Gaussian-observation model, applying the two-step iterations of the EM algorithm is equivalent to using Baum's re-estimation approach [29], which is as follows. Consider two new variables $\xi_t(i,j)$ and $\gamma_t(i)$, with $i, j \in \mathcal{S}$, that are defined as $\xi_t(i,j) = P(S_t = i, S_{t+1} = j|O_{1:T}, \boldsymbol{\Theta})$ and $\gamma_t(i) = \sum_{j=1}^{|\mathcal{S}|} \xi_t(i,j)$. We have:

$$\begin{aligned} \pi_i^{\text{new}} &= \gamma_1(i),\ a_{ij}^{\text{new}} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)} \\ \mu_i^{\text{new}} &= \frac{\sum_{t=1}^{T} \gamma_t(i) o_t}{\sum_{t=1}^{T} \gamma_t(i)},\ \sigma_i^{2\ \text{new}} = \frac{\sum_{t=1}^{T} \gamma_t(i)(o_t - \mu_i)^2}{\sum_{t=1}^{T} \gamma_t(i)} \end{aligned} \tag{2.2}$$

where $\xi_t(i,j)$ and $\gamma_t(i)$ are computed using the Forward-Backward algorithm, see [30, 31].

We observe that the standard HMM is inadequate for our purpose. In fact, it uses a geometric Probability Mass Function (PMF) $g(d) = (a_{ii})^{d-1}(1 - a_{ii})$ to describe the dwell time of any hidden state $S_t = i \in \mathcal{S}$ with self-transition probability a_{ii}, i.e., $g(d)$ is the probability of staying in any hidden state $S_t = i \in \mathcal{S}$ for $d - 1$ subsequent time steps and then leave the state (probability $(1 - a_{ii})$). It has been argued that this poorly models real phenomena, since most real-life applications do not obey this temporally-decaying function [32]. To tackle this, we consider the Extended Duration Hidden

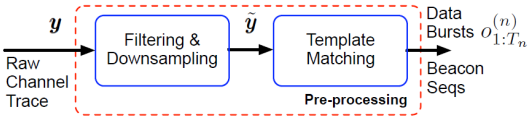


Figure 2.4: Flow diagram of the mm-wave channel pre-processing phase.

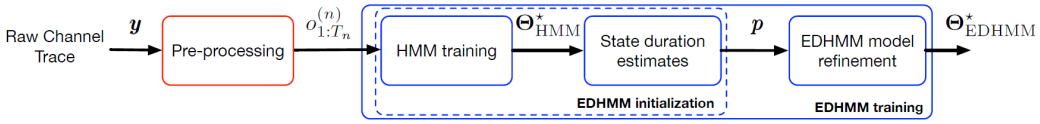


Figure 2.5: EDHMM training procedure: initial state duration estimates are obtained through HMM training and are refined using EDHMM learning tools.

Markov Model (EDHMM), where for each hidden state $i \in \mathcal{S}$ we have $a_{ii} = 0$ and a state-specific distribution $p_i(d)$ is defined over the discrete set $\mathcal{D}_i = \{d_i^{\min}, \ldots, d_i^{\max}\}$, where $d_i^{\min}$ and $d_i^{\max}$ are the minimum and maximum durations for the protocol element transmitted when the EDHMM is in state i, respectively. Hence, upon entering state $i \in \mathcal{S}$, the sequence of observations in that state is assumed to be conditionally independent (i.e., i.i.d. once the state is entered), of length $d \in \mathcal{D}_i$ (sampled from $p_i(d)$), and is emitted from $P(O_t|S_t = i) = \mathcal{N}(O_t; \mu_i, \sigma_i^2)$. For the EDHMM, the duration distributions are collected into a vector $\boldsymbol{p}$, with $\boldsymbol{p} = [p_1(\cdot), \ldots, p_{|\mathcal{S}|}(\cdot)]$ and the EDHMM is described through the parameter vector $\boldsymbol{\Theta}_{\text{EDHMM}} = [\boldsymbol{\pi}, \boldsymbol{A}, \boldsymbol{b}, \boldsymbol{p}]$. In the following analysis, we use the HMM model to initialize the state duration distribution of the EDHMM (see Section 2.7 for further details on the EDHMM training). Also, we use the forward-backward algorithm proposed by Yu and Kobayashi in [33, 34], as an alternative and efficient approach to solve Eq. (2.2).

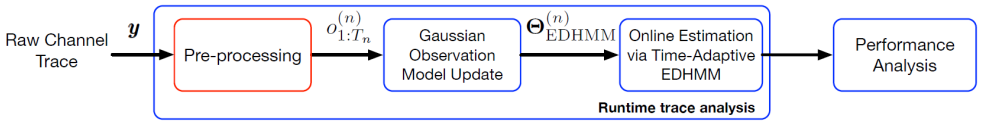


Figure 2.6: EDHMM runtime mm-wave channel analysis.

2.5 High level description of the framework

The aim of the mm-wave channel analyzer that we present in this chapter is twofold. First, we want to track when data bursts are transmitted and, for each, detect which packets are exchanged, their duration, and average energy. This allows to obtain statistics on their number, duration, whether there are channel problems (which may be detected from missing ACKs). As a second objective, we track the transmission of control packets, which are sent for link management purposes. These control packets appear in two flavors as follows:

C1) Beacon pairs that mark the beginning of a data burst.

C2) BR sequences that are utilized to maintain the radio link.

Our approach consists of three steps.

Step 1 – Pre-processing (Fig. 2.4): beacon detection and data burst extraction are implemented through the pre-processing chain of Fig. 2.4, which operates on the raw channel trace, through filtering, downsampling and template matching (see Section 2.6). We design the pre-processing chain for the case of 802.11ad but we can easily adapt it to suit other protocols. This pre-processing phase identifies all the beacons, classifies their occurrences into C1 and C2 and outputs a collection of N data bursts of the form $\{o_{1:T_n}^{(n)} | n = 1, \dots, N\}$, which are disjoint and contiguous channel subsequences.[3]

After Step 1, we delve into the semantic decoding of the protocol elements that are transmitted within each data burst, i.e., the elements in the above defined set $\mathcal{S}$. To assess which elements are transmitted, along with their average energy and timing, we utilize an EDHMM model, which is first trained (Fig. 2.5), and then used at runtime (Fig. 2.6) with non-stationary traces. Let $\boldsymbol{y} = (y_1, y_2, \dots)$ be a sequence of channel samples. In general, $\boldsymbol{y}$ can be written as $\boldsymbol{y} = \boldsymbol{x} + \boldsymbol{w}$ [35, Chapter 14], where $\boldsymbol{x} = (x_1, x_2, \dots)$ is the signal of interest *at the receiver*, that is, after transmission, and $\boldsymbol{w} = (w_1, w_2, \dots)$ is the background noise. From our experimental measurements, we know that $\boldsymbol{y}$ is highly non-stationary across data bursts, i.e., there are substantial variations in the energy associated with the signal $\boldsymbol{x}$ and the noise $\boldsymbol{w}$, which entail changes in μ_i and σ_i^2, for $i \in \mathcal{S}$. Moreover, they can also be caused by power

[3] A contiguous subsequence is made up of consecutive channel samples.

control adjustments to compensate for channel attenuation and device mobility. Nevertheless, the transmission time of the elements in set $\mathcal{S}$ are channel and protocol-specific. We proceed through the following steps.

Step 2 – EDHMM training (Fig. 2.5): we use *stationary* channel traces[4] for a preliminary and robust training of the EDHMM parameters. Channel traces were picked so as to encompass a wide range of data rates and MCSs, which determine the different lengths of the physical layer data frames. The distance between transmitter and receiver is kept fixed and the surrounding environment (indoor for our experiments) is kept as stable as possible (i.e., no user mobility, etc.). From these stationary channels, the state-specific distributions $p_i(d)$ for $i \in \mathcal{S}$ do not undergo major changes during each trace and this allows their accurate estimation. Then, all the trace-specific distributions are combined into a global distribution considering a wide range of protocol settings, see Section 2.7. Note that training is needed only once for a given technology (e.g., IEEE 802.11ad).

Step 3 – Runtime trace analysis (Fig. 2.6): the EDHMM parameters μ_i and σ_i^2, for $i \in \mathcal{S}$ do depend on channel attenuation and noise. Thus, these parameters are estimated at runtime for each data burst using a clustering algorithm, whereas the $p_i(d)$ are known from Step 2. The so obtained EDHMM model is used to estimate the most likely sequence in $\mathcal{S}$ (called the Viterbi path) from the samples in the current data burst. This step is explained in Section 2.8.

Steps 2 and 3 rely on the further assumption that:

A3) Channel attenuation and noise are stationary within bursts.

2.6 Pre-processing

Data acquisition, filtering, and downsampling: to obtain the energy traces that we use as input for our machine learning algorithm, we overhear the communication of COTS 60 GHz devices using one or more external sniffers. Each sniffer consists of a Sivers IMA FC1005V/00 V-Band converter. The

[4]These *stationary* channel traces do not exhibit any particular trend. This means that μ_i and σ_i^2 do not significantly vary across data bursts.

Table 2.1: Average synchronization error versus template length τ.

$\tau = 1$ ms	$\tau = 2$ ms	$\tau = 3$ ms	$\tau = 4$ ms	$\tau \geqslant 5$ ms
36.36 ms	13.43 ms	2.63 ms	1.17 ms	0

converter receives signals in the 60 GHz band either via a directional (20°) or omni-directional antenna and outputs them at 2 GHz intermediate frequency (IF). We capture the IF signal using a Universal Software Radio Peripheral (USRP) X310 Software Defined Radio (SDR) at a sample rate of 30 MHz. That is, we only need to capture a *fragment* of the bandwidth of the signal to obtain an energy trace which is suitable for our machine learning technique. To obtain a second trace from a different angle, we connect a second sniffer to the same USRP to ensure perfect time synchronization among traces. Since the coverage area of a mm-wave AP is limited due to high path loss, sniffers are typically close to each other and can thus be connected to the same USRP. Moreover, if traces are recorded on different USRPs, it is possible to synchronize them in post-processing using a variant of template matching, see Fig. 2.7, where a subsequence from SN_2 is used as a template. In Tab. 2.1, we report the average synchronization error as a function of the template length τ. To obtain these results, we have run $1,000$ simulations for each value of τ picking a random subsequence from SN_1 and a subsequence from SN_2 (used as a template) with random temporal offset with respect to the subsequence from SN_1. We obtain perfect synchronization by choosing $\tau \geqslant 5$ ms. Synchronizing traces from multiple sniffers is thus doable and only requires picking a sufficiently long template length.

Fig. 2.8 shows our measurement setup. The original raw trace $\boldsymbol{y}$ is first filtered and then downsampled to a lower rate for scaling purposes, so that each sample of the new trace is computed as the mean of three subsequent samples in the original raw trace. This new trace is then smoothed using a fast and robust discretized spline filtering algorithm for data of large size [36] [37], thus obtaining the trace $\tilde{\boldsymbol{y}}$. This pre-processing phase is needed to remove part of the noise due to hardware impairments during data acquisition. It does not harm the EDHMM classification performance, but generally improves it, as the noise variance in the energy traces is reduced.

Template matching algorithm: after the data acquisition, filtering, and downsampling, a collection of N data bursts of the form $\{o_{1:T_n}^{(n)} | n = 1, \ldots, N\}$

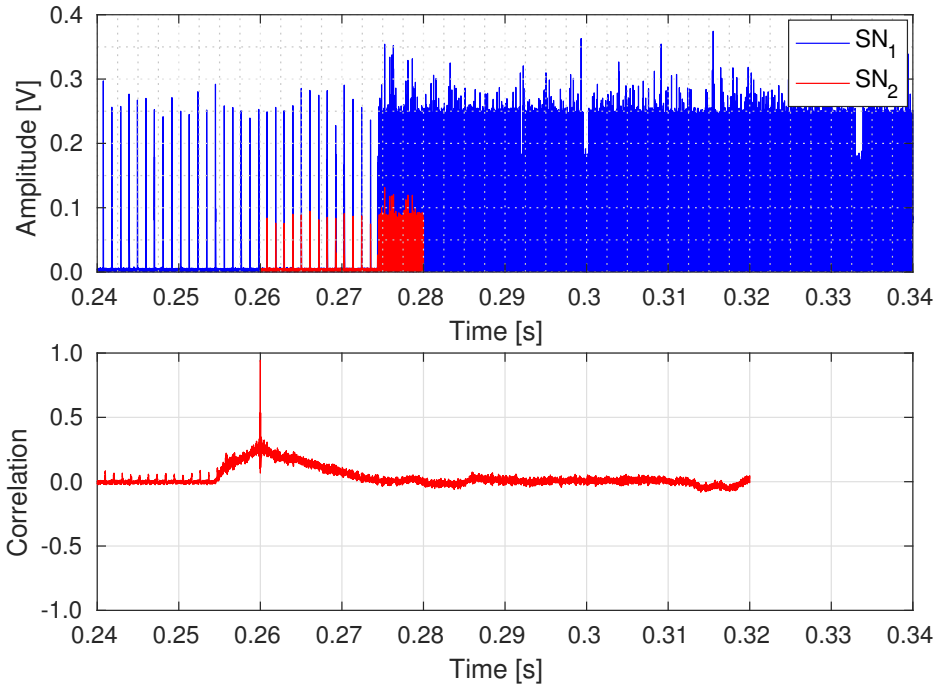


Figure 2.7: Synchronization using a variant of template matching, where a subsequence from SN_2 is used as a template.

is extracted from the mm-wave trace $\tilde{\boldsymbol{y}}$. This requires a reliable identification technique for the data bursts and, recalling that each data burst is preceded by a pair of beacons, this corresponds to reliably detecting beacon pairs. What we observe from the collected channel traces is that the beacon duration and the inter-frame spacing between them are almost constant within and across experiments. Moreover, we note that the beacon shape is quite particular, showing different energy levels at the beginning and at the end. These characteristics make it possible to exploit a template matching technique for the beacon detection. Here, we are interested in finding C1) beacon pairs, and C2) BR sequences, as these are key to understand the protocol behavior.

At the core of our template matching approach, we use Pearson's correlation coefficient $r \in [-1, 1]$ [38], which is a statistical measure of the strength of a linear relationship between two vectors $\boldsymbol{u} = [u_1, \ldots, u_K]$ and $\boldsymbol{v} = [v_1, \ldots, v_K]$ (with mean μ_u and μ_v, respectively). It is defined as the ratio of their covariance C_{uv} and the square root of the product of their variances σ_u^2 and σ_v^2, i.e., $r = C_{uv}/(\sigma_u \sigma_v)$, where C_{uv} is the sample covariance, given by:

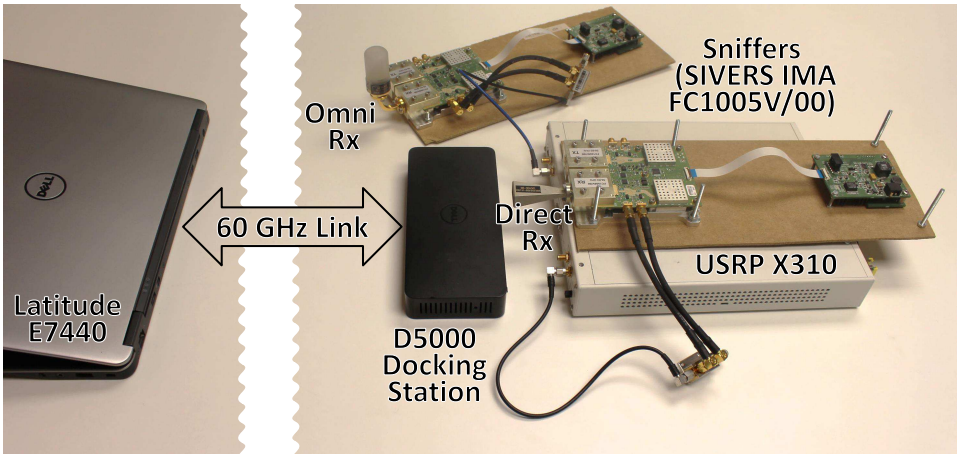


Figure 2.8: Practical sniffer setup for trace capture. The antennas at the sniffers can be both directional or omni-directional, and sniffer location can be varied.

$$C_{uv} = \frac{1}{K-1} \sum_{k=1}^{K} (u_k - \mu_u)(v_k - \mu_v) \,. \tag{2.3}$$

Pearson's correlation coefficient is suitable to deal with the non-stationarity of the traces, since it just evaluates some internal relationship between the provided vectors. Moreover, template matching is known to be the optimal detection technique in the presence of white Gaussian noise [39], which we found to be a good assumption for our mm-wave channel traces [27]. Henceforth, for our template matching technique, $\boldsymbol{u}$ corresponds to the average shape of a beacon frame (i.e., the template with a length of K samples), which the system can easily obtain from channel idle times. During those idle times, nodes only transmit periodic beacons which can be clearly identified and used as a template. Vector $\boldsymbol{v}$ contains the channel samples from the current K-dimensional sliding window, which moves over the signal trace $\tilde{\boldsymbol{y}}$, obtained after the acquisition, filtering, and downsampling of $\boldsymbol{y}$. We adopted the fast template matching scheme of [40] [41], which exploits the Fast Fourier Transform (FFT), thus obtaining dot products in the frequency domain. For a generic channel sequence $\tilde{\boldsymbol{y}}$ of $L > K$ samples, this allows the computation of the covariance in $O(L \log L)$ time. Hence, the template matching operates on $\tilde{\boldsymbol{y}} = (\tilde{y}_1, \ldots, \tilde{y}_L)$, outputting a sequence of correlation estimates $(r_1, \ldots, r_{L-K+1})$. We detect a possible beacon at sample ℓ if r_ℓ is greater than a threshold r_{th}. Then, since multiple trivial

matches (i.e., $r_\ell > r_{\text{th}}$) are likely to occur within a window of samples, we perform a further peak detection within the regions containing multiple matches, by taking the default timing parameters of the IEEE 802.11ad communication standard into account [42]. That is, two beacons can never be placed at a distance smaller than the minimum allowed by the protocol rules. As the final step, we assess which beacon pairs actually mark the start of data bursts by assessing the distance between them, as this is constant. Through this, we can reliably detect false positives, such as isolated beacons due to communication errors or to packets that are erroneously detected as beacons as their shape closely resembles that of the template. We found excellent results across all our experiments setting $r_{\text{th}} = 0.75$. Note that r_{th} is independent of the trace amplitude. Thus, we do not need to readjust it for each scenario and/or trace.

The identification of pairs of beacons (C1) allows extracting the data bursts $\{o^{(n)}_{1:T_n} | n = 1, \ldots, N\}$ from $\tilde{\boldsymbol{y}}$, which are fed as input to the following EDHMM training phase. Longer beacon sequences (C2) are likewise detected by looking at the number of energy levels of the beacons therein and at their inter-frame spacing, as dictated by the standard [42]. These events are semantically decoded as described below.

2.7 EDHMM training

For the EDHMM training we refer to Fig. 2.5. We recall that the objective of this training phase is to reliably estimate the distribution vector $\boldsymbol{p}$, modeling the duration of inter-frame spaces, packets and acknowledgements. This phase is executed once offline and is not scenario dependent. Essentially, it is a calibration step for the specific mm-wave technology used in the network, which in our case is IEEE 802.11ad. The traces used in this step should be as much as possible stationary. This means that μ_i and σ_i^2 do not significantly vary across data bursts. As a first processing stage, we use the pre-processing procedure of Section 2.6, which returns the data burst set $\{o^{(n)}_{1:T_n} | n = 1, \ldots, N\}$. Next, for illustration purposes we refer to the n-th data burst $o^{(n)}_{1:T_n} = (o_1, \ldots, o_{T_n})$, but in our implementation the HMM parameters are estimated using the entire burst set (i.e., the N bursts in the mm-wave trace). For burst n, each of the samples o_t, $t = 0, \ldots, T_n$, maps to an element $s_t \in \mathcal{S}$, where state "1" means

IFS, "2" DATA and "3" ACK. Our goal is to accurately associate each o_t in the data burst with the actual protocol element $i \in \mathcal{S}$ and, most importantly, to reliably estimate its duration PMF $p_i(\cdot)$. This estimation is performed having access to the noisy observations $(o_1, \ldots, o_{T_n})$ of the actual protocol elements.

EDHMM initialization: we consider $o_{1:T_n}^{(n)}$ as training data and our aim is to get accurate state duration estimates for the EDHMM. This is achieved by deriving initial estimates for $\boldsymbol{p}$ through a simpler HMM model. Once this vector is found, it is refined using EDHMM training tools. The HMM parameter vector is $\boldsymbol{\Theta}_{\text{HMM}}$ and the three fundamental steps involved in the HMM model estimation are:

E1) The forward-backward algorithm is used to compute metrics $\gamma_t(i)$ and $\xi_t(i,j)$ with $t = 1, \ldots, T_n$, $i, j \in \mathcal{S}$ (see Eq. (2.2)) for a given HMM transition structure and a list of observations. These weigh the probability of getting the observed sequence from the current model.

E2) The model parameter vector $\boldsymbol{\Theta}_{\text{HMM}}$ is adjusted through the EM algorithm.

E3) The Viterbi algorithm [43] is used to compute the most probable path via a Maximum Likelihood (ML) approach.

Step E2 returns the optimal parameter vector $\boldsymbol{\Theta}_{\text{HMM}}^{\star}$, whereas E3 outputs the sequence of hidden states $(s_1, \ldots, s_{T_n})$ that most likely generated the observed samples $(o_1, \ldots, o_{T_n})$.

Specifically, we assume $\pi_1 = 1$ as all the data bursts start with a silence, right after the beacon pair. Moreover, the HMM transition matrix $\boldsymbol{A}$ is constrained in the sense that the hidden state sequence evolves according to structured trajectories [44]. In particular, we have $a_{23} = a_{32} = 0$, as there must be some minimum inter-frame spacing between subsequent messages. Also, we set $a_{ii} = 1 - 1/T_s$ for $i \in \mathcal{S}$, where $T_s = 0.1$ μs is the channel sampling period after the downsampling of Section 2.6. The initialization implies geometrically distributed state dwell-time distributions. This serves as a sufficiently good initialization of the transition matrix, and increases the robustness of the HMM model against random fluctuations in the channel dynamics. Next, we use the Viterbi algorithm output (step E3) to initialize the state duration distribution of the EDHMM model.

The final parameter estimates $\boldsymbol{\Theta}^{\star}_{\text{HMM}}$ strongly depend on the initial vector $\boldsymbol{\Theta}^{0}_{\text{HMM}}$ for the EM evaluation (step E2). To obtain good initial parameter estimates, we use the K-means clustering algorithm, see [45,46], which classifies the channel samples in the data bursts around $|\mathcal{S}|$ centers. The $|\mathcal{S}|$ initial values of the centers can be randomly picked or taken as the locations of the peaks in the empirical distribution of the observed samples. The latter approach was implemented and found to perform satisfactorily across all datasets. Upon completion, the K-means algorithm returns $|\mathcal{S}|$ values for the cluster centers, which are used as initial values for μ_i for $i \in \mathcal{S}$. The $|S|$ variances σ_i^2 are derived from the distribution of the samples clustered around the centers μ_i so obtained.

At this point, we use the Viterbi algorithm output (step E3) to fit $|\mathcal{S}|$ two-parameter inverse Gaussian distributions [47] [48] for vector $\boldsymbol{p}$, where the range $\mathcal{D}_i = \{d_i^{\min}, \ldots, d_i^{\max}\}$ for state $i \in \mathcal{S}$ is such that $d_1^{\min} = \cdots = d_{|\mathcal{S}|}^{\min} = 1$ and $d_1^{\max} = \cdots = d_{|\mathcal{S}|}^{\max} = D$. In particular, we set D according to the timing parameters of the IEEE 802.11ad communication standard [42] and filter out all the state durations that are outside these boundaries. The authors in [48] show how to find maximum likelihood solutions for the parameters of any family of exponential distributions. Since the exponential family is log-concave, the global maximum can be found by setting derivatives equal to zero, yielding the maximum likelihood equations. For some distributions in the exponential family (e.g., Gaussian), these equations can be solved analytically, while most distributions must be solved numerically. In the present work, two-parameter inverse Gaussian distributions have been preferred over non-parametric state duration distributions, as parametric models require far less training data and generalize better to new data. Note also that, even if we expect a fixed set of timing parameters for the IEEE 802.11ad communication standard [42],[5] it is still possible that the duration of the protocol frames in the training data differs from that in the test data, due to MCS adjustments. Thus, a rigid setting that only allows for a few fixed values, although correct in theory, may lead to unsatisfactory results in real cases, overfitting the training data and generalizing poorly over the test cases.

EDHMM model refinement: the initial estimate $\boldsymbol{p}$ that we have found

[5]That is, only a few fixed durations for ACK, DATA and silence are possible, as DATA depend on the adopted modulation and coding scheme.

with the above HMM model is subsequently refined through EDHMM training tools. Here, we opted for the forward-backward algorithm proposed by Yu and Kobayashi in [33, 34] as it is efficient and solves practical issues such as numerical underflows occurring in the EM iterations.

2.8 Runtime trace analysis

In this section, we present a runtime analyzer that effectively deals with the non-stationarity of the traces, i.e., variations in μ_i and σ_i^2 for $i \in \mathcal{S}$ across data bursts. As a first step, we run the pre-processing block of Section 2.6, which returns the data burst sequences $\{o_{1:T_n}^{(n)} | n = 1, 2, \dots\}$ through template matching. For each sequence, the energy levels associated with the states IFS, DATA and ACK are re-estimated, as explained in the following.

Gaussian-observation model update: we rely on assumption A3, i.e., that channel statistics are stationary within each data burst. Of course, μ_i and σ_i^2 may change considerably across data bursts and we tackle this by running the K-means clustering algorithm for each burst sequence $o_{1:T_n}^{(n)}$, so as to re-initialize vector $\boldsymbol{b}$ in an online fashion. The $|\mathcal{S}|$ final values of the centers initialize μ_i, whereas the variances of the samples clustered around these centers initialize σ_i^2, for $i \in \mathcal{S}$. Upon completing the K-means algorithm, we obtain the updated parameter set $\boldsymbol{\Theta}_{\text{EDHMM}}^{(n)}$ for the current data burst, whereas vector $\boldsymbol{p}$ (which represents the "average" time-frame duration statistics) remains fixed. We remark that, for the current burst n, $\boldsymbol{\Theta}_{\text{EDHMM}}^{(n)}$ may differ from the optimal parameter set $\boldsymbol{\Theta}_{\text{EDHMM}}^{\star}$, as for the latter vectors $\boldsymbol{p}$ and $\boldsymbol{b}$ would be obtained through the ML approach of [33,34], whereas in $\boldsymbol{\Theta}_{\text{EDHMM}}^{(n)}$ the energy levels in $\boldsymbol{b}$ are estimated on-the-fly through K-means. Since the latter approach does not take into account the joint re-estimation of $\boldsymbol{p}$ and $\boldsymbol{b}$, the resulting energy levels are less accurate. However, this approach provides a substantial speedup as neither the re-estimation of the transition matrix $\boldsymbol{A}$ nor that of vector $\boldsymbol{p}$ are required and these computations account for most of the EDHMM complexity. Hence, the benefit due to the increased speed outweighs the loss in accuracy.

Online estimation via time-adaptive EDHMM: upon obtaining $\boldsymbol{\Theta}_{\text{EDHMM}}^{(n)}$ for the current data burst $o_{1:T_n}^{(n)}$, the corresponding hidden state sequence $(s_1, \dots, s_{T_n})$ is reconstructed using the Viterbi algorithm with samples $o_{1:T_n}^{(n)} =$

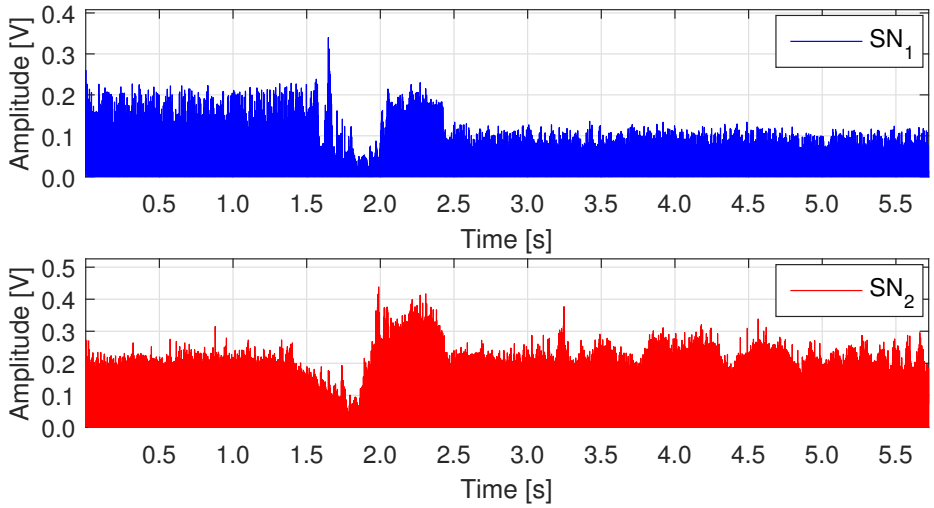


Figure 2.9: Communication recorded from two different locations SN_1 and SN_2.

$(o_1, \ldots, o_{T_n})$, i.e., each sample o_t is mapped onto one of the elements in $\mathcal{S}$. As suggested in [32], we implemented the Viterbi algorithm using logarithms to avoid numerical underflows. Also, given the sequence of observations $o_{1:T_n}^{(n)}$, the time complexity of the Viterbi algorithm for EDHMM is $O(|\mathcal{S}|ZT_nD)$, where Z is the average number of predecessors for each state $i \in \mathcal{S}$. In our case, $Z < |\mathcal{S}|$, since we set $a_{23} = a_{32} = 0$ (states 2 and 3 respectively denote DATA and ACK). Moreover, since durations are explicitly accounted through $p_i(\cdot)$, we have $a_{ii} = 0$, $\forall\, i \in \mathcal{S}$ and $Z = 4/3$. Hence, the computational cost of the Viterbi algorithm is primarily affected by the data burst length T_n and by the maximum duration D.

2.9 Generalization to multiple dimensions

Up to this point, we have considered that the observed channel samples in the data burst are modeled as a sequence of real-valued random variables corresponding to one of the following basic elements: "1" IFS, "2" DATA and "3" ACK. Accordingly, the hidden state S_t at time t is a discrete real-valued random variable that can take values in the set $\mathcal{S} = \{1, 2, 3\}$. This corresponds to capturing the channel from a single measurement point.

In this section, we are concerned with the case where multiple channel traces from the same source are concurrently monitored from different measure-

ment points. This amounts to deploying multiple time-synchronized receivers (sniffers) and have them listening to the same transmitter. The rationale is that the channel realizations that they respectively see are likely to be uncorrelated. This can provide significant improvements to the detection performance. In Fig. 2.9, we show the same transmission captured by two different sniffers, labeled SN_1 and SN_2.

In this scenario, the observed channel samples in the data burst are modeled as a sequence of multi-dimensional, real-valued random variables, $\boldsymbol{O}_{1:T} = (\boldsymbol{O}_1, \ldots, \boldsymbol{O}_T)$, where T is the data burst length. The observation vector associated with channel sample t, $\boldsymbol{O}_t$, is a probabilistic function of the hidden state S_t. According to the Gaussian-observation model, $P(\boldsymbol{O}_t|S_t = i) = \mathcal{N}(\boldsymbol{O}_t; \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$, where $\boldsymbol{\mu}_i$ and $\boldsymbol{\Sigma}_i$ respectively specify the multi-dimensional mean and the diagonal covariance matrix of the random vector $\boldsymbol{O}_t$, given that the hidden state is $i \in \mathcal{S}$. For all hidden states $i \in \mathcal{S}$, we collect the parameter pairs $b_i = (\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$ through vector $\boldsymbol{b} = [b_1, \ldots, b_{|\mathcal{S}|}]$, as for the one-dimensional case. For Baum's re-estimation approach [29], we have:

$$\begin{aligned} \boldsymbol{\mu}_i^{\text{new}} &= \frac{\sum_{t=1}^{T} \gamma_t(i) \boldsymbol{o}_t}{\sum_{t=1}^{T} \gamma_t(i)} \\ \boldsymbol{\Sigma}_i^{\text{new}} &= \frac{\sum_{t=1}^{T} \gamma_t(i) (\boldsymbol{o}_t - \boldsymbol{\mu}_i)^{\mathrm{T}} (\boldsymbol{o}_t - \boldsymbol{\mu}_i)}{\sum_{t=1}^{T} \gamma_t(i)}, \end{aligned} \tag{2.4}$$

where $\boldsymbol{o}_t$ is the realization of $\boldsymbol{O}_t$, $\xi_t(i,j)$ and $\gamma_t(i)$ are computed using the Forward-Backward algorithm, see [30, 31], and $(\cdot)^{\mathrm{T}}$ denotes vector transpose. Here, we assume that there is no correlation among the observed channel samples from multiple viewpoints, hence $\boldsymbol{\Sigma}_i$ is a diagonal covariance matrix for $i \in \mathcal{S}$. In the following, the sniffers are denoted by $\mathrm{SN}_{\mathrm{dim}}$, where $\mathrm{dim} = 1, 2, \ldots$. Moreover, when the hidden state is $i \in \mathcal{S}$, we refer to the element in position dim in $\boldsymbol{\mu}_i$ as $\mu_i^{(\mathrm{dim})}$ and to the dim-th diagonal element of $\boldsymbol{\Sigma}_i$ as $\Sigma_i^{(\mathrm{dim})}$.

2.10 Mm-wave trace generator

Need for the ground truth: a ground truth signal is necessary to precisely quantify the reconstruction performance of the one- and the multi-dimensional protocol analyzers. This would amount to acquire the actual protocol state

that is associated with each sample in $\boldsymbol{O}_{1:T}$. In practice, this could be achieved using a tool such as Wireshark on a monitor node. Unfortunately, state-of-the-art 802.11ad hardware is still unable to reliably provide such information to the higher layers. The protocol state sequence that is extracted by the radio is usually incomplete (some frames are missing), the states are shifted in time (with distorted inter-spaces) and often their order is also affected. The only metric that current devices can reliably provide are cumulative counters of packet types (see Section 2.11.1 for further discussion and experimental results).

To overcome this, we have developed a realistic mm-wave trace generator, with the goal of reproducing *narrowband* physical layer energy traces from one or multiple sniffers in a fast and accurate manner. This makes the evaluation of our diagnosis tool possible, providing quantitative results in a range of practical scenarios. The developed generator reproduces typical IEEE 802.11ad data burst sequences, mimicking random fluctuations in the channel dynamics, variability in the number and duration of DATA and ACK frames, etc. These burst sequences are separated by beacon pairs (also affected by channel noise), whereas other control messages, appearing outside the data bursts, are not modeled as they are not involved in our performance assessment.

This tool has been instrumental in the fine tuning of the EDHMM model, as it allows for a precise control of the energy levels associated with transmissions from the source and channel noise. Next, we detail its structure, which is organized into macro- and micro-states. Macro-states describe different instances of the channel transmission setup, i.e., a specific combination of coding and modulation schemes, and we assume that macro-state transitions can only occur at the end of data bursts. Instead, micro-states track the duration of DATA and ACK frames within a data burst, for any given setup (macro-state). Each data burst starts with a beacon pair and the system remains in the same macro-state for the entire duration of the burst. Once in a data burst, the micro-state model returns the sequence of DATA and ACK frames.

Notation: the superscript $^{(0)}$ is used for the macro-model parameters. The macro-state takes values in the set $\mathcal{M}^{(0)} = \{1, \ldots, |\mathcal{M}^{(0)}|\}$ and evolves according to the transition matrix $\boldsymbol{T}^{(0)}$, with steady state distribution $\boldsymbol{\pi}^{(0)}$. If the current macro-state is $M \in \mathcal{M}^{(0)}$, the micro-state m takes values in the set $\mathcal{M}^{(M)} = \{1, \ldots, |\mathcal{M}^{(M)}|\}$ and evolves according to the transition matrix $\boldsymbol{T}^{(M)}$,

with steady state distribution $\boldsymbol{\pi}^{(M)}$.

The macro-model: macro-states capture different instances of the channel transmission setup, i.e., $|\mathcal{M}^{(0)}|$ different combinations of coding and modulation schemes. For each macro-state $M \in \mathcal{M}^{(0)}$, the following statistics are specified: (i) the PMF $P(d_{\text{IDLE}}^{(M)})$ of the idle time $d_{\text{IDLE}}^{(M)}$ between subsequent packets, (ii) the joint PMF $P(d_{\text{DATA}}^{(M)}, d_{\text{ACK}}^{(M)})$ of DATA and ACK durations, respectively termed $d_{\text{DATA}}^{(M)}$ and $d_{\text{ACK}}^{(M)}$, where the ACK is the frame following the DATA one in the hidden state sequence $(s_1, \ldots, s_{T_n})$, and (iii) the PMF $P(d_{\text{burst}}^{(M)})$ of the duration of data bursts, $d_{\text{burst}}^{(M)}$.

The following remarks are in order:

- We have experimentally verified that $P(d_{\text{DATA}}^{(M)}, d_{\text{ACK}}^{(M)}) \neq P(d_{\text{DATA}}^{(M)})P(d_{\text{ACK}}^{(M)})$, which means that there exists some correlation between the marginal random variables modeling DATA and ACK durations. This is due to frame aggregation, which results in block acknowledgments. Such acknowledgments are longer than the regular acknowledgments used for shorter, non-aggregated frames.

- Stationary channel traces are utilized to obtain the duration statistics of idle times, DATA packets and ACKs. As done for the online estimation via time-adaptive EDHMM, upon obtaining $\boldsymbol{\Theta}_{\text{EDHMM}}^{(n)}$ for the current data burst $o_{1:T_n}^{(n)}$, the corresponding hidden state sequence $(s_1, \ldots, s_{T_n})$ is reconstructed using the Viterbi algorithm with samples $o_{1:T_n}^{(n)} = (o_1, \ldots, o_{T_n})$. From these estimates, duration statistics for the elements in the set $\mathcal{S} = \{1, 2, 3\}$ and for the data burst itself are obtained.

- Transitions between macro-states occur at the end of each data burst according to the transition matrix $\boldsymbol{T}^{(0)}$. This makes it possible to probabilistically model variations in the protocol behavior due to, e.g., soft link blockages (e.g., waiving a hand in the boresight of the antenna) or to modifications of the received energy due to a change in the orientation of the device.

The micro-model: consider a data burst in any macro-state M. Within this data burst, a sequence of DATA-ACK frames is exchanged, and the duration of such packets is controlled by the micro-model. Specifically, the domain of the

PMF $P(d^{(M)}_{\text{DATA}}, d^{(M)}_{\text{ACK}})$ is clustered into $|\mathcal{M}^{(M)}|$ rectangular subdomains through the Elbow method, which is a clustering technique to automatically determine the number of clusters K [49]. We run this clustering algorithm twice: for the dimension associated with $\{d^{(M)}_{\text{DATA}}\}$ and for that associated with $\{d^{(M)}_{\text{ACK}}\}$, obtaining $K^{(M)}_{\text{DATA}}$ and $K^{(M)}_{\text{ACK}}$ clusters for DATA and ACK frames, respectively. This leads to $|\mathcal{M}^{(M)}|$ rectangular subdomains with $|\mathcal{M}^{(M)}| = K^{(M)}_{\text{DATA}} \times K^{(M)}_{\text{ACK}}$. Each of such domains $m \in \mathcal{M}^{(M)}$ defines a micro-state with conditional PMF $P(d^{(M)}_{\text{DATA}}, d^{(M)}_{\text{ACK}}|m)$, representing the joint distribution of DATA and ACK durations within that region (conditioned on the model being in region m). Transitions between micro-states occur according to the transition matrix $\boldsymbol{T}^{(M)}$, which is estimated from empirical data. The steady-state probability vector $\boldsymbol{\pi}^{(M)}$ is obtained through numerical integration of $P(d^{(M)}_{\text{DATA}}, d^{(M)}_{\text{ACK}})$ within region m, for all $m \in \mathcal{M}^{(M)}$.

Outline of the algorithm: the pseudo-code of the mm-wave trace generator is given in Algorithm 4. The algorithm's output consists of sequences of data bursts, delimited by beacon pairs. The algorithm starts by picking macro- and micro-states according to the respective steady-state distributions (see function `pick`() in lines 1 and 2). $\tilde{\boldsymbol{y}}^{(\text{dim})}$ is the noisy output sequence, which is initialized as an empty vector (line 3). In line 5, the data burst duration T is sampled from the PMF $P(d^{(M)}_{\text{burst}})$ and time-synchronized burst sequences are generated for two sniffers SN_1 and SN_2 (dim = 1 and 2), see line 6. Each burst starts with a beacon pair, denoted by $\boldsymbol{tmp}^{(\text{dim})}$, which is a noisy version of the template used by the template matching algorithm of Section 2.6. The beacons are concatenated to the output sequence $\tilde{\boldsymbol{y}}^{(\text{dim})}$ in line 7, and the noisy data burst is created through the "while" cycle starting from line 9. Durations of DATA (d_2), ACK (d_3) frames and of the IDLE time between them (d_1) are respectively sampled from the PMFs $P(d^{(M)}_{\text{DATA}}, d^{(M)}_{\text{ACK}}|m)$ and $P(d^{(M)}_{\text{IDLE}})$. A noisy sequence composed of DATA (d_2 samples), IDLE (d_1 samples), ACK (d_3 samples) and IDLE (d_1 samples) is generated through the "`create_frames`()" function of line 12. The so obtained output samples $\boldsymbol{o}'$ (of length $2d_1 + d_2 + d_3$ samples) is then appended to the current sequence $\boldsymbol{o}^{(\text{dim})}$ (line 13). When the while cycle ends, $\boldsymbol{o}^{(\text{dim})}$ contains the noisy samples associated with the new data burst. The noisy samples in the sequence, which are associated with hidden state $i \in \{1, 2, 3\}$ (respectively IDLE, DATA and ACK), are computed as (additive

Algorithm 1 Pseudo-code of the mm-wave trace generator

$M = \texttt{pick}(\boldsymbol{\pi}^{(0)})$; // pick macro-state
$m = \texttt{pick}(\boldsymbol{\pi}^{(M)})$; // pick micro-state
Set $\ell = 0$; $\tilde{\boldsymbol{y}}^{(\mathrm{dim})} = \texttt{empty\_vector}()$;
while $\ell < L$ **do**
    $T = \texttt{pick}(P(d^{(M)}_{\mathrm{burst}}))$;
    for dim = 1 to 2 **do**
        $\tilde{\boldsymbol{y}}^{(\mathrm{dim})} = \texttt{concatenate}(\tilde{\boldsymbol{y}}^{(\mathrm{dim})}, \boldsymbol{tmp}^{(\mathrm{dim})})$;
        Set $t = 0$; $\boldsymbol{o}^{(\mathrm{dim})} = \texttt{empty\_vector}()$;
        while $t < T$ **do**
            $d_1 = \texttt{pick}(P(d^{(M)}_{IDLE}))$;
            $(d_2, d_3) = \texttt{pick}(P(d^{(M)}_{\mathrm{DATA}}, d^{(M)}_{\mathrm{ACK}})|m)$;
            $\boldsymbol{o'} = \texttt{create\_frames}(d_1, d_2, d_3)$;
            $\boldsymbol{o}^{(\mathrm{dim})} = \texttt{concatenate}(\boldsymbol{o}^{(\mathrm{dim})}, \boldsymbol{o'})$;
            // Change current micro-state?
            $m = \texttt{next\_state}(\boldsymbol{T}^{(M)}, m)$;
            $t = \texttt{length}(\boldsymbol{o}^{(\mathrm{dim})})$;
        end while
        $\tilde{\boldsymbol{y}}^{(\mathrm{dim})} = \texttt{concatenate}(\tilde{\boldsymbol{y}}^{(\mathrm{dim})}, \boldsymbol{o}^{(\mathrm{dim})})$;
    end for
    $M_{\mathrm{prev}} = M$;
    // Change current macro-state?
    $M = \texttt{next\_state}(\boldsymbol{T}^{(0)}, M)$;
    if $M \neq M_{\mathrm{prev}}$ **then**
        // resample from steady-state distribution
        $m = \texttt{pick}(\boldsymbol{\pi}^{(M)})$;
    end if
    $\ell = \texttt{length}(\tilde{\boldsymbol{y}}^{(\mathrm{dim})})$;
end while

Gaussian noise): $\mu_i^{(\mathrm{dim})} + \sqrt{\Sigma_i^{(\mathrm{dim})}}\texttt{randn}(1, d_i)$, where "$\texttt{randn}(1, d_i)$" denotes a random vector of d_i elements, with Gaussian distributed entries $\mathcal{N}(0, 1)$. Although not explicitly indicated, the sequence of hidden states is saved along with the noisy version $\tilde{\boldsymbol{y}}^{(\mathrm{dim})}$ and used as ground truth for the performance evaluation of Section 2.11.2.

We now discuss some example results for the case of two macro-states. $M = 1$: distance TX-RX 1.5 m, MCS 11. $M = 2$: distance TX-RX 2.5 m, MCS 10. Empirical $(d^{(M)}_{\mathrm{DATA}}, d^{(M)}_{\mathrm{ACK}})$ pairs are shown in Fig. 2.10 for $M \in \{1, 2\}$, along with the rectangular regions obtained using the Elbow clustering algorithm. We observe that the duration statistics are more spread in macro-model 1 with

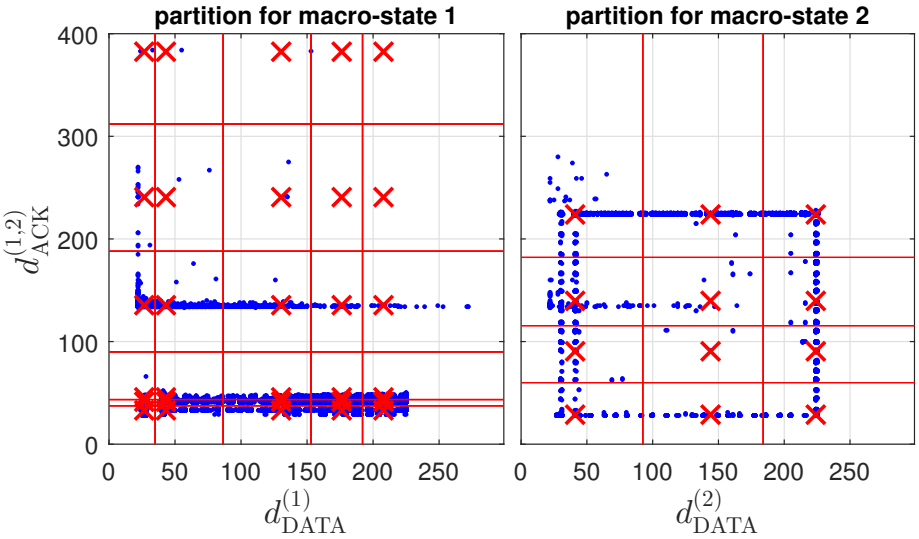


Figure 2.10: Empirical measurements $(d^{(M)}_{\text{DATA}}, d^{(M)}_{\text{ACK}})$ for two marco-states. Rectangular regions are obtained using the Elbow method. Values in the axes are expressed in number of channel samples (the sampling frequency is $T_s = 0.1$ μs).

respect to macro-model 2, meaning less data aggregation. Also, the length of the data burst is shorter in macro-model 1, see Fig. 2.11. For each macro-state M, the domain $(d^{(M)}_{\text{DATA}}, d^{(M)}_{\text{ACK}})$ is split into clusters using the Elbow method: the number of clusters is greater in macro-model 1, i.e., $K^{(1)}_{\text{DATA}} = K^{(1)}_{\text{ACK}} = 6$ and $K^{(2)}_{\text{DATA}} = K^{(2)}_{\text{ACK}} = 3$.[6] Fig. 2.12 shows empirical points $(\mu_i^{(\text{dim})}, \Sigma_i^{(\text{dim})})$ for different channel setups. In this plot, we do not distinguish between states $i \in \{1, 2, 3\}$, as our purpose is to establish a suitable relation between mean (μ) and variance (Σ) of the received energy levels, and the difference in the received energy levels captured by the sniffers depends on the relative position of the sniffers with respect to the communicating devices. These empirical points were fitted through the following curve (the red solid curve in the plot):

$$\Sigma_i = c_1 \mu_i^{c_2} \, , \; i \in \{1, 2, 3\} \, , \tag{2.5}$$

with $c_1 = 0.105$ and $c_2 = 1.905$. The coefficients c_1 and c_2 were found through a linear regression in the logarithmic domain, i.e., we fit the dataset taking into account the logarithmic counterpart of the datapoints and minimize the total

[6]The number of clusters is chosen such that the percentage of variance explained by them is greater than 90%, where this percentage represents the ratio between the group variance and the total variance.

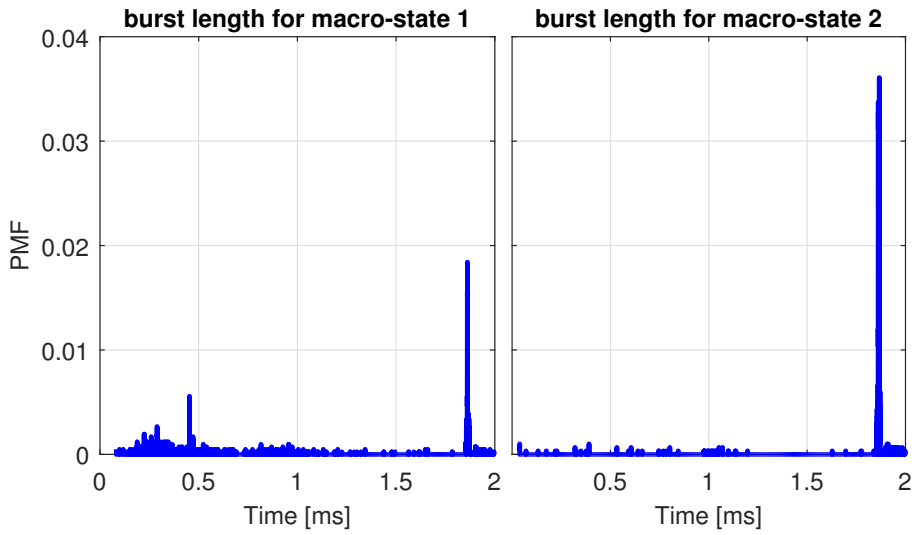


Figure 2.11: PMF of the burst length $P(d_{\text{burst}}^{(M)})$ for macro-models 1 and 2.

residual error, obtaining an excellent goodness of fit ($R^2 = 0.9595$, where R^2 is the coefficient of determination). The linear relationship in the logarithmic domain of Eq. (2.5), that we obtained empirically, is also confirmed by previous analytical work on RSS localization, see, e.g., [50].

As a final consideration, we note that the average energy levels μ_i and their variances Σ_i remain constant for the entire duration of the data bursts, which is a key assumption in the developed EDHMM algorithm (see assumption A3, in Section 2.5). In the numerical results, we assess the performance of our algorithms when assumption A3 is no longer verified, i.e., when μ_i and Σ_i do change within a DATA burst. This is achieved through an *additive* sinusoidal noise of frequency f, which is added to the generated energy traces, by tuning f and the noise amplitude.

2.11 Performance results

Next, we present some selected performance results. Experimental results are discussed in Section 2.11.1, considering single and multiple time-synchronized sniffers. The performance of our tool is further quantified in Section 2.11.2, using the mm-wave trace generator of Section 2.10.

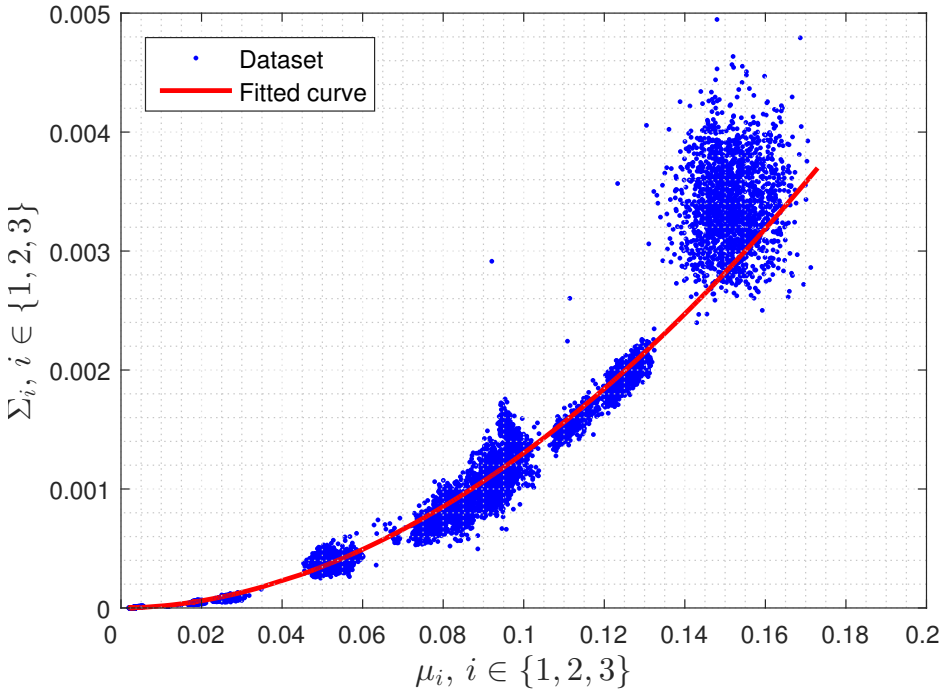


Figure 2.12: Gaussian observation model: empirical values and fitting curve for mean (μ) and variance (Σ) of the received energy levels associated with IDLE periods, DATA and ACK frames.

2.11.1 Evaluation with experimental data

We consider the setup in Section 2.6. First, we validate our machine learning framework in controlled scenarios. Next, we study the behavior of indoor links during regular operation to check how our framework can identify and characterize effects such as beam misalignment.

Validation in controlled scenarios: Fig. 2.13 shows a trace decoding example for our diagnosis tool. In the upper part of the figure, we show the raw trace as captured by the Sivers IMA converter. The two initial frames are beacons that indicate the start of a data burst. After that, we observe a sequence of data and acknowledgment frames (c.f. Fig. 2.1). The lower part of the figure shows that our framework can correctly identify all frames in the trace. We observe that the framework successfully classifies data packets, acknowledgments, beacons, and inter-frame spacing. Moreover, Fig. 2.13 also demonstrates the need for our EDHMM approach. The HMM method wrongly classifies many of the samples—within a data or acknowledgment frame, it often fluctuates between states. In contrast, the EDHMM classifies all samples

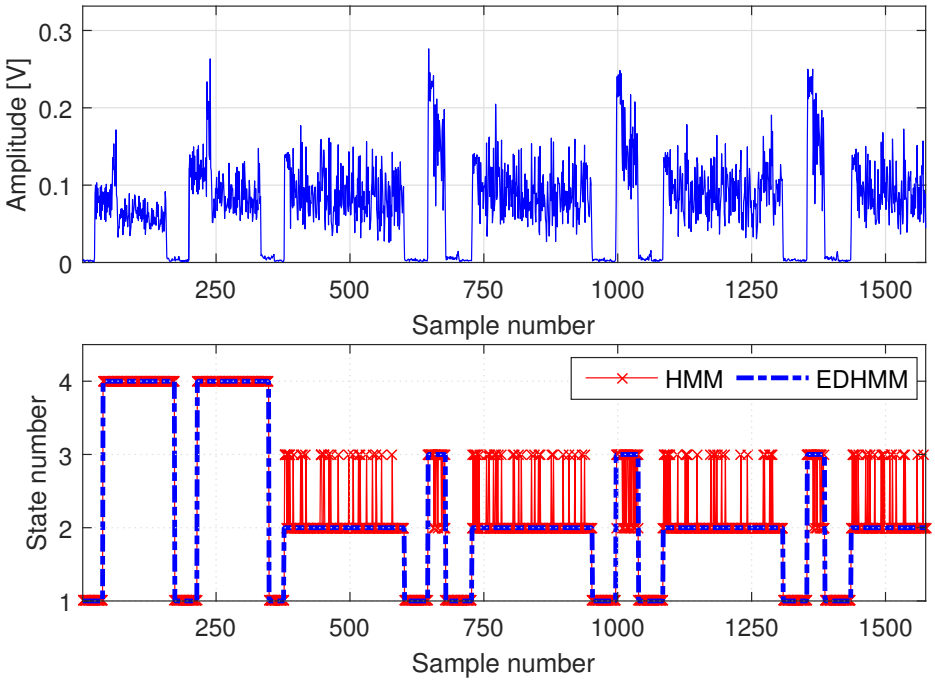


Figure 2.13: Trace decoding example for our machine learning framework.

correctly, even in case of varying data packet lengths.

In addition to the visual inspection in Fig. 2.13, we validate our framework using two approaches. First, we compare the number of data packets that our tool identifies with the number of packets that the driver of our 60 GHz device reports. For the case without blockage in Fig. 2.14, the driver reports 31960 sent packets at the end of the trace. This matches the data packet counter in our results. Second, we record the same data exchange using two independent sniffers SN_1 and SN_2, and process the resulting traces using our framework. For no blockage, Fig. 2.14 shows that both sniffers count the same number of both data and control packets. This again validates that our framework is correctly decoding the trace. For data packets, the counter stabilizes at one second at which point we stop the data transmission. Still, the control packet counter increases steadily because the devices continue to exchange control packets even if no data transmission is taking place.

Fig. 2.14 also depicts similar measurements for two blockage cases. The first is a "hard blockage", i.e., crossing the link and thus interrupting it completely for a few milliseconds. The second is a "soft blockage", which refers to partial blockage such as waiving a hand in the boresight of the antenna. These

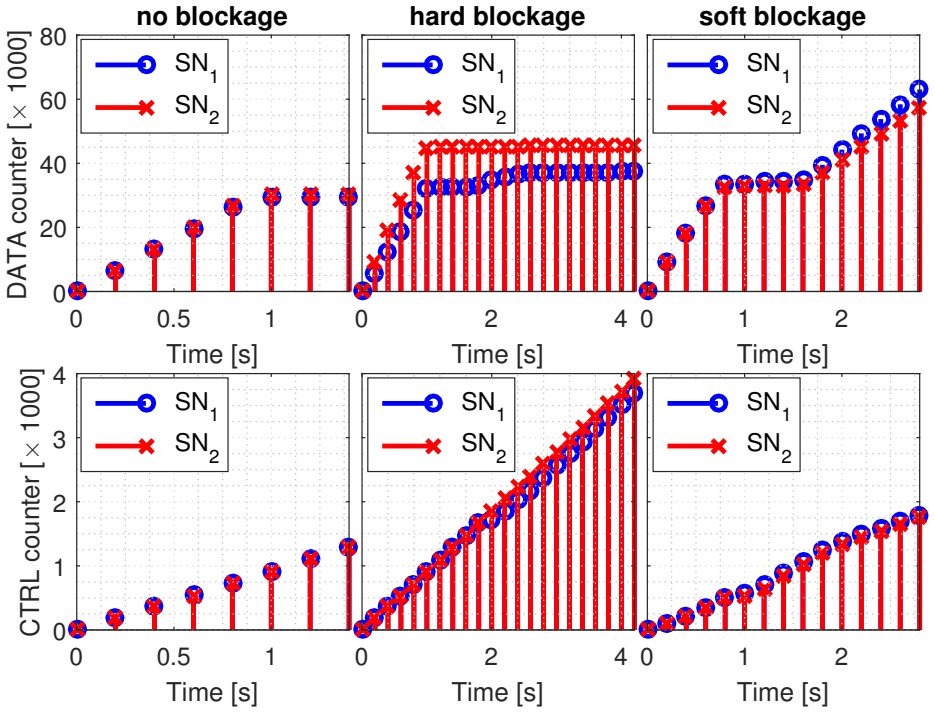


Figure 2.14: Number of data and control packets identified by our tool. We show the results for two sniffers SN_1 and SN_2 placed at different locations.

blockages cause a drop in the energy levels captured by the two independent sniffers SN_1 and SN_2, which actually perceive a different number of both data and control packets due to the different relative positions of the sniffers with respect to the mm-wave link.

Regular operation: in the following, we show some selected diagnosis capabilities of our tool for regular link operation. Fig. 2.15 depicts the Empirical Cumulative Distribution Function (ECDF) of the packet and burst lengths that our tool computes for different links. All links have the same length and are deployed in the same location. However, we change their orientation to induce different antenna beam patterns which result in suboptimal performance, and which our framework can identify. The protocol used by our 60 GHz test devices defines that the maximum burst length is two milliseconds and the maximum aggregated packet length is 20 microseconds [6]. Since we perform this experiment with full transmission buffer at the nodes, the burst and packet lengths should match the maximum values.

For the different measurements, the link distance is maintained to be equal to 3 meters, while the rotation of the nodes varies, resulting in changes in the

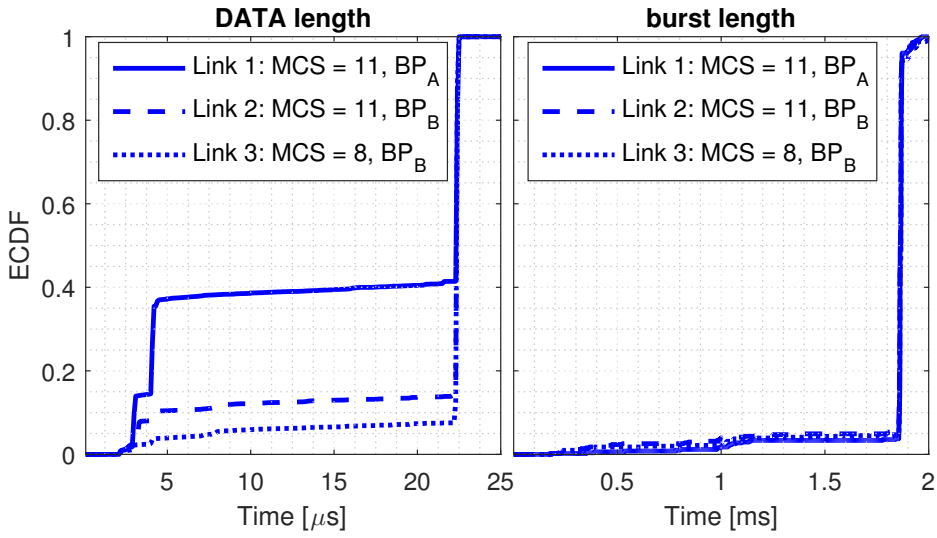


Figure 2.15: CDF of packet and burst lengths for three links deployed in the same environment but with varying performance. "BP" stands for beampattern.

MCS and frame duration. Specifically, for Link 3 the antennas of the devices are facing one another, whereas for Link 1 and 2 they are not. While the MCS of Link 1 and 2 are the same, for Link 1 in Fig. 2.15 we observe smaller packet durations. To reduce the packet error rate when the link quality is worse, the MAC reduces the level of aggregation, i.e., the MAC layer aggregates fewer data packets than the maximum into a single MAC packet. Indeed, our framework also reveals that the trace energy level differs compared to Link 2, which suggests antenna misalignment. We omit the energy trace level in the interest of space but the device driver reveals that both Links 1 and 2 operate otherwise identically in terms of MCS and traffic load. In other words, our framework successfully identifies the suboptimal device orientation for Link 1. Fig. 2.15 shows that Link 3 performs even better in terms of packet length. Again, the device driver confirms this insight since Link 3 uses a more robust MCS than Link 2. Thus, Link 3 is more likely to succeed when transmitting longer packets.

External disturbance: regarding external disturbances, we focus on the case of link blockage. Our tool is able to identify and classify such blockage. This provides means for network operators to determine how often blockage actually occurs for a certain mm-wave link during a certain time-frame, for instance, a day.

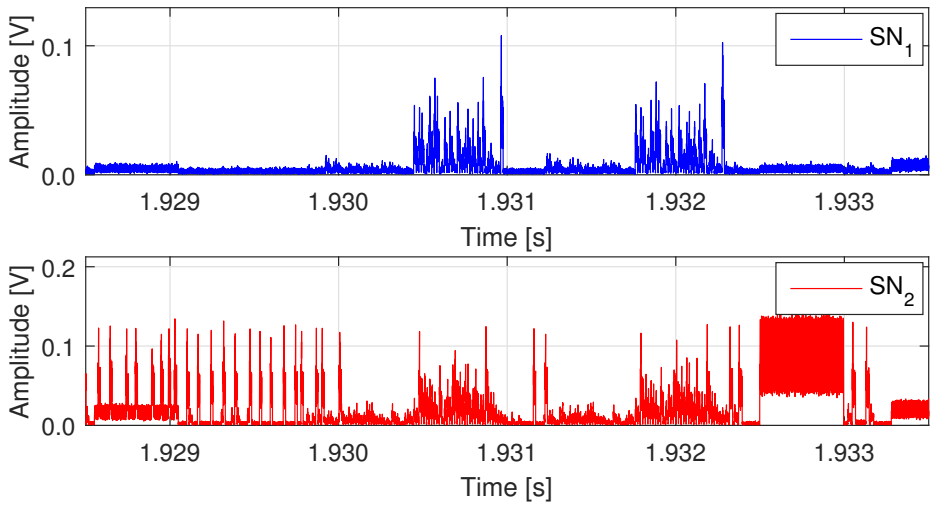


Figure 2.16: Blockage recorded from two different locations SN_1 and SN_2. The figure shows a fraction of the blockage, i.e., the blockage affects all samples.

Identifying blockage is challenging because it may block the LOS path to the sniffer, too. To prevent this, our framework can record and compare the channel activity from two or more sniffers at different locations, as shown in Fig. 2.16. We observe that while sniffer SN_1 barely receives any of the activity prior to second 1.93, SN_2 is able to receive all frames during the blockage. This allows our framework to obtain a much more complete view of the activity on the channel. Based on this information, we automatically identify beam refinement (BR) sequences. Such sequences are rare in static scenarios but are likely to occur if the link is impaired. Fig. 2.17 depicts a segment of the trace in Fig. 2.16, overlapped with the locations at which our framework identifies BR sequences. We observe that the BRs identified by both sniffers match but that not all sniffers capture all sequences due to the blockage. This highlights again the benefit of being able to analyze the network behavior from multiple viewpoints. Moreover, Fig. 2.16 depicts a soft blockage. Thus, the connection does not break and the device continuously adapts its beampattern, resulting in a large number of BRs. In contrast, hard blockage results in less BRs since the transmitter and the receiver cannot communicate during the blockage. As per our measurements, the average number of BRs per trace for no blockage, hard blockage, and soft blockage is 0 BRs/trace, 0.42 BRs/trace, and 4.02 BRs/trace, respectively. The difference in terms of BR frequency allows our diagnosis tool to classify blockage. This is highly valuable to determine why a

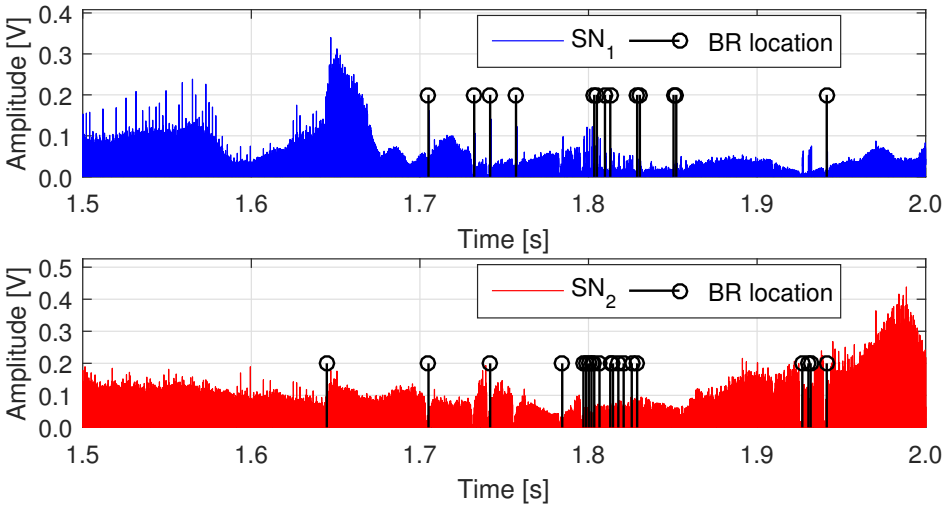


Figure 2.17: Beam refinement sequences during soft link blockage.

mm-wave link is performing poorly.

In Fig. 2.14, we also show packet counters for the case of blockage. The data packet counter for hard blockage stabilizes at roughly $40,000$ packets because we stop transmission at that point. For soft blockage, we transmit continuously and thus the packet counter increases throughout the trace. We observe that the data packet counters for each sniffer disagree as soon as blockage occurs. The underlying reason is that one of the sniffers does not receive the full channel activity while the other one does. While not as unambiguous as the detection of BRs, this also hints at potential blockage scenarios. We observe that hard blockage causes a stronger disagreement than soft blockage, providing means to differentiate them. The mismatch among sniffers is less explicit for control packets since the shape of such patterns is easier to identify than the shape of data packets. Hence, both sniffers are more likely to correctly classify such control packets even in case of blockage.

Combining different viewpoints: next, we exploit the learning framework to jointly process mm-wave channel traces from multiple time-synchronized sniffers, since different viewpoints of the same channel will provide complementary information and thus can lead to higher decoding accuracies. To this end, we perform several experiments deploying the transmitter, the receiver and the sniffers as shown in Fig. 2.18. While multiple sniffer combinations were tested, for the sake of brevity we only report the results for the sniffer pairs SN_1 and SN_2, as we found it very instructive and sufficient to reveal the

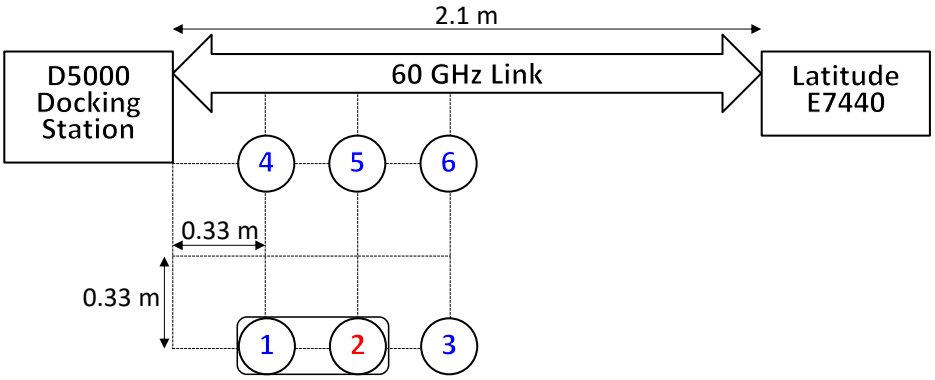


Figure 2.18: Indoor measurement setup. Specifically, sniffers SN_1 and SN_2 are time-synchronized.

dynamics behind the decoding process. In Fig. 2.19, we compare the number of data packets that our tool identifies using two independent sniffers SN_1 and SN_2 in a controlled scenario (no link blockage). In this case, while both sniffers count the same number of control packets (as beacon pairs are robustly detected via template matching), the number of data packets differ. In this case, SN_2 provides better decoding accuracy compared to SN_1, as can be also observed from the ECDF in Fig. 2.20 (left plot). The multi-dimensional (joint) processing of the two traces allows us to estimate the packet duration statistics in a much more reliable fashion, see the left plot of Fig. 2.20 (dashed line), and to correct the bias in the DATA count for SN_1, see Fig. 2.19 (right plot). The reason why SN_2 provides higher decoding performance compared to SN_1 is that SN_2 records more distinctive values for the amplitude mean and standard deviation (std) of the EDHMM states $i \in \{1, 2, 3\}$, whereas for SN_1 it is difficult to discriminate among different energy levels at the receiver. This is shown in Fig. 2.21, where we plot the estimated amplitude mean and standard deviation (std) of the EDHMM states $i \in \{1, 2, 3\}$ for the sniffers SN_1 and SN_2, together with the other (omnidirectional) sniffers from the measurement setup of Fig. 2.18. In the barplot, we note that DATA and ACK packets for SN_1 are more likely to be erroneously classified by the EDHMM, as they present almost indistinguishable energy levels. Further, we remark that the estimated amplitude mean and standard deviation (std) of the EDHMM states $i \in \{1, 2, 3\}$ do not necessarily directly reflect the distances of the sniffers with respect to the mm-wave link, as the sniffers are deployed close to each other in

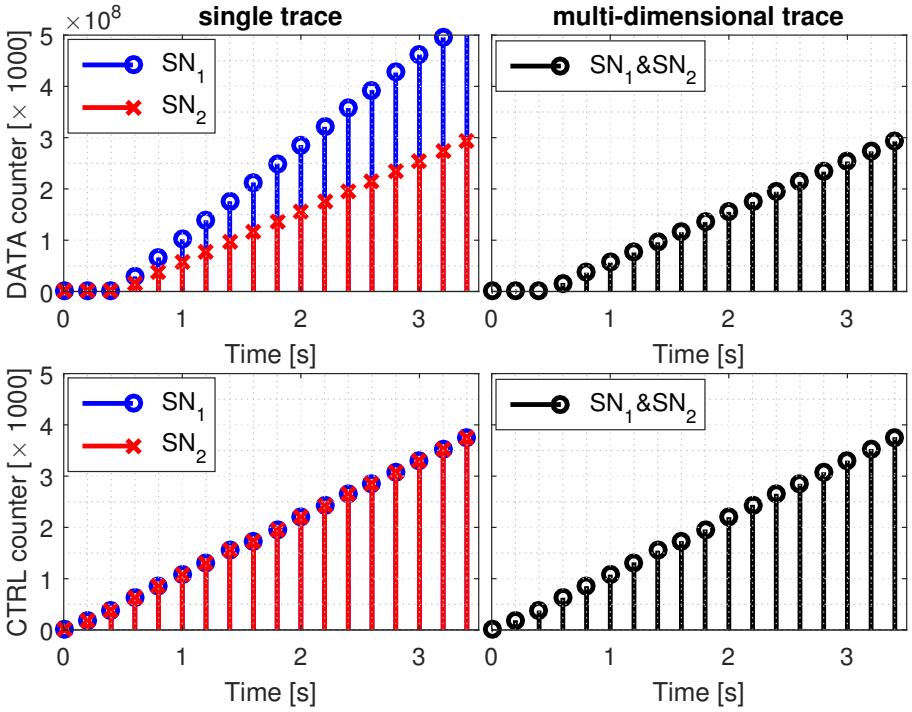


Figure 2.19: Number of data and control packets identified by our tool. The results are for two sniffers SN_1 and SN_2 placed at different locations. The left plot shows the results from decoding the two traces independently; the results for the multi-dimensional trace (joint) processing are shown in the right plot.

an indoor measurement setup.[7] However, the difference in the received energy levels captured by the sniffers depends on the relative position of the latter with respect to the communicating devices.

2.11.2 Reconstruction from generated traces

For the performance evaluation in this section, we define the reconstruction factor $\rho \in [0, 1]$ as the fraction of samples in the received sequence that are correctly reconstructed by the algorithm. To this end, we compare the reconstructed sample sequence obtained by the Viterbi algorithm against the ground truth, see Section 2.10. $\rho = 1$ means perfect reconstruction.

We consider two time-synchronized sniffers, dim $= 1, 2$. Moreover, with $\Delta_{ij}^{(\text{dim})}$ we indicate the difference in the received energy levels associated with IDLE periods, DATA and ACKs frames for the channel acquired by sniffer

[7] As sniffers are omnidirectional, we do not experience errors in the estimated amplitude mean and standard deviation (std) of the EDHMM states $i \in \{1, 2, 3\}$ due to antenna misalignment.

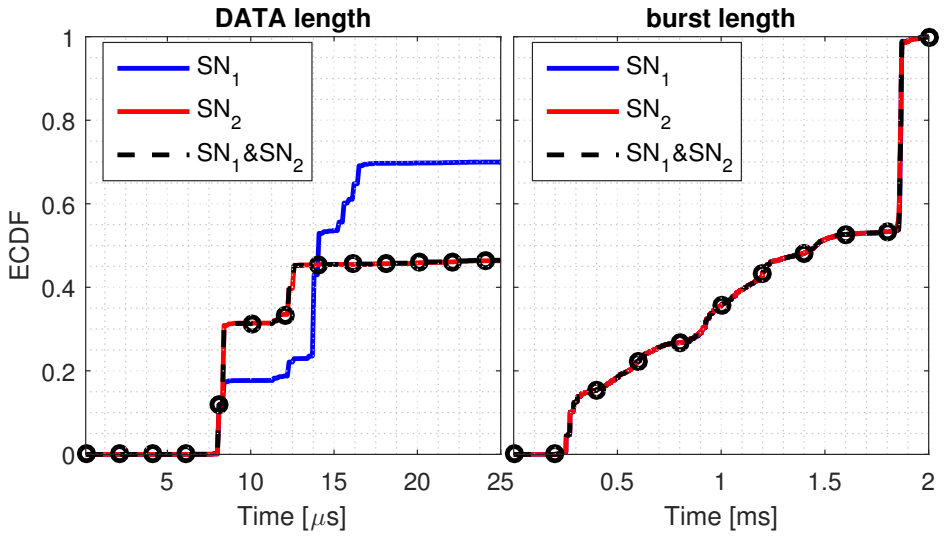


Figure 2.20: ECDF of packet and burst lengths for the single trace and the multi-dimensional (joint) trace processing (SN_1 & SN_2).

dim, i.e., $\Delta_{ij}^{(\text{dim})} = \mu_j^{(\text{dim})} - \mu_i^{(\text{dim})}$, with $i, j \in \{1, 2, 3\}$ and $i < j$. Specifically, we aim to evaluate the reconstruction factor $\rho \in [0, 1]$ as a function of $\Delta_{12}^{(\text{dim})}$ and $\Delta_{23}^{(\text{dim})}$, as $\Delta_{13}^{(\text{dim})}$ can be obtained from $\Delta_{13}^{(\text{dim})} = \Delta_{12}^{(\text{dim})} + \Delta_{23}^{(\text{dim})}$. In principle, the difference in the received energy levels captured by the sniffers depends on the relative position of the latter with respect to the communicating devices. For the performance evaluation in this section, without loss of generality, we consider $\Delta^{(\text{dim})} = \Delta_{12}^{(\text{dim})} = \Delta_{23}^{(\text{dim})}$ to conveniently evaluate the reconstruction factor $\rho \in [0, 1]$ through a single free parameter $\Delta^{(\text{dim})}$, simplifying the parameter space, and allowing a compact graphical representation of the results.

In Fig. 2.22, we plot ρ when the decoding is jointly performed over the traces from two sniffers as a function of $\Delta^{(1)}$ (energy gaps for sniffer 1, on the abscissa), keeping $\Delta^{(2)}$ fixed for each curve. For this plot, $\mu_1^{(\text{dim})} = 0.001$ for $\text{dim} = 1, 2$, the remaining energy levels follow from $\Delta^{(\text{dim})}$, whereas the variances $\Sigma_i^{(\text{dim})}$ are obtained through Eq. (2.5). For comparison, the case of a single sniffer ($\text{dim} = 1$) is also shown through the solid blue curve. As expected, the reconstruction factor ρ increases with an increasing $\Delta^{(1)}$, since the runtime mm-wave analyzer in this case better initializes the EDHMM parameters for each data burst. A single trace ($\text{dim} = 1$) leads to excellent results even for $\Delta^{(1)} = 0.001$, and using two traces allows for a further improvement.

However, we remark that this good performance is not always possible and this very much depends on the chosen initial levels $\mu_1^{(\text{dim})}$, $\text{dim} = 1, 2$. To see

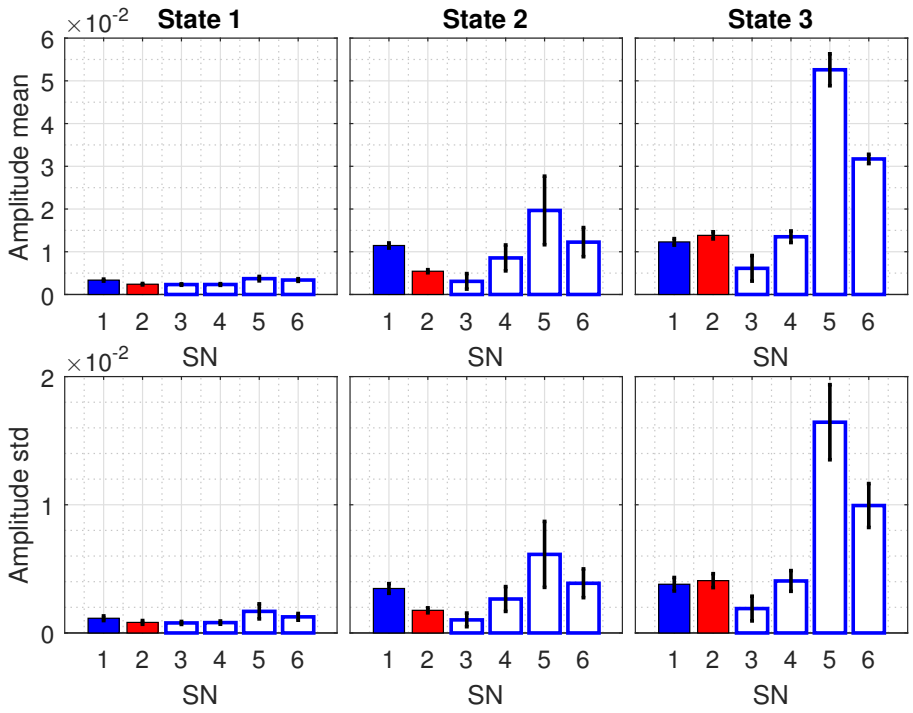


Figure 2.21: Estimated amplitude mean and standard deviation (std) of the EDHMM states $i \in \{1, 2, 3\}$ for the sniffers SN_1 and SN_2, together with other (omnidirectional) sniffers, as shown in the measurement setup of Fig. 2.18.

this, we have considered a scenario where the distance between energy levels is rather small, i.e., $\Delta^{(1)} = \Delta^{(2)} = 0.001$, whereas $\mu_1^{(1)}$ and $\mu_1^{(2)}$ are allowed to change. These results are plotted in Fig. 2.23, where $\mu_1^{(1)}$ appears on the x-axis and $\mu_1^{(2)}$ is kept fixed for each curve. The single trace performance is shown for comparison through a solid blue line. In this case, we observe that the reconstruction performance is heavily affected by an increasing background noise level μ_1. In fact, as μ_1 gets larger, according to Eq. (2.5), the noise variance also increases (almost quadratically) and since the energy gap (Δ) between levels remains constant, it becomes increasingly difficult to discriminate among different energy levels at the receiver. (ii) Moreover, there is a fundamental tradeoff in the number of sniffers to use. For example, for $\mu_1^{(1)} = 0.003$, the addition of a second sniffer helps if $\mu_1^{(2)}$ is smaller than or equal to 0.004. However, if the second trace has a very poor quality (e.g., $\mu_1^{(2)} = 0.005$) a single trace provides a better reconstruction performance. Interestingly, when $\mu_1^{(1)} \approx \mu_1^{(2)}$, it always pays off to use two sniffers, no matter the value of μ_1.

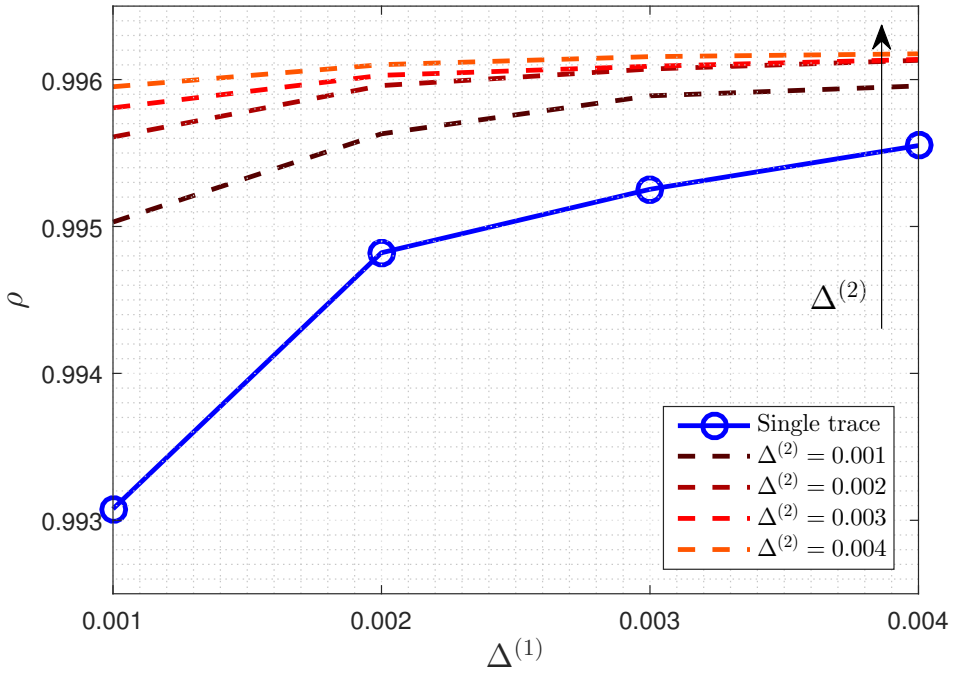


Figure 2.22: ρ as a function of $\Delta^{(1)}$, keeping $\Delta^{(2)}$ fixed for each curve. For this plot, $\mu_1^{(\mathrm{dim})} = 0.001$ for $\mathrm{dim} = 1, 2$, the remaining energy levels follow from $\Delta^{(\mathrm{dim})}$, whereas the variances $\Sigma_i^{(\mathrm{dim})}$ are obtained through Eq. (2.5).

In the above paragraph (referring to Fig. 2.23), we have considered a scenario where the distance between energy levels is rather small, i.e., $\Delta^{(1)} = \Delta^{(2)} = 0.001$, and the background noise dominates the overall performance. It turns out that this choice of parameters is critical for proper EDHMM functionality. Note that, even though these are extreme cases, such small values of Δ have been found in our measurements. However, if we allow for a slightly greater distance between energy levels, i.e., $\Delta^{(1)} = \Delta^{(2)} = 0.002$, then the overall performance increases significantly (see Fig. 2.24). Using two traces can lead to much better results, especially when the noise affecting the first one is significant, e.g., $\mu_1^{(1)} \geqslant 0.005$.

Now, we test the performance of our diagnosis tool by also inducing random fluctuations in the channel dynamics. That is, we account for an *additive* sinusoidal noise signal with frequency f, which is added to the generated traces. Then, we compare the performance of our diagnosis tool as a function of the frequency f, which is related to the average length of the data burst T_B. To do this, we proceed as follows: 1) given the channel transmission

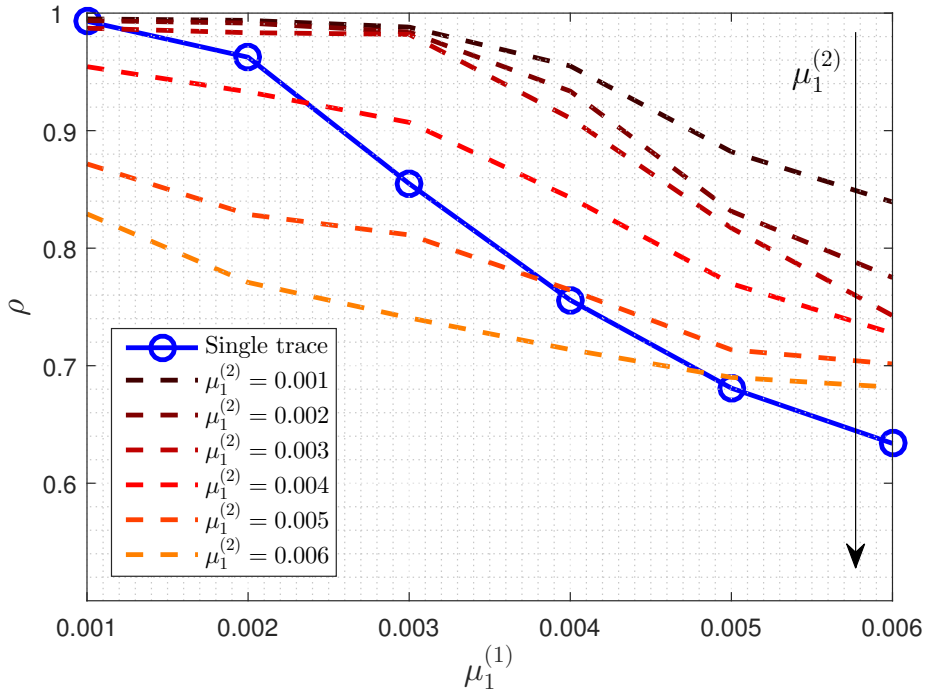


Figure 2.23: ρ as a function of $\mu^{(1)}$ with $\Delta^{(1)} = \Delta^{(2)} = 0.001$, whereas $\mu_1^{(1)}$ and $\mu_1^{(2)}$ are allowed to change.

setup $M \in \mathcal{M}^{(0)}$, we compute the average length of the data burst T_B (in seconds); 2) we design an additive sinusoidal noise signal $\sin(2\pi ft)$ as a function of the frequency f, which is related to the average length of the data burst T_B, i.e., $\sin(2\pi ft) = \sin(2\pi k/T_B \ell T_s)$, where $\ell = 1, \ldots, L$ counts the samples in vector $\tilde{\boldsymbol{y}}^{(\mathrm{dim})}$ and T_s is the sampling time of the generated traces (we use $T_s = 0.1\ \mu s$); 3) we test the runtime mm-wave analyzer for varying k, where $f = k/T_B$, and T_B depends on $P(d_{\mathrm{burst}}^{(M)})$. Specifically, the sinusoidal (noise) signal $A \cdot [\sin(2\pi k/T_B \ell T_s) + 1]/2$ is added to the generated traces, where A is the maximum amplitude of the sine wave. In principle, superimposed sinusoidal noise can occur for several reasons, such as interference from another transmission, and mobility. Also, hardware impairments during data acquisition are a major cause of superimposed sinusoidal noise. These distortion effects are very hardware specific and, as such, the value of f can vary significantly, depending on the experimental setting. An example of sinusoidal noise which was found in the measured channel traces is shown in Fig. 2.25, together with a reconstructed signal which is representative of the superimposed sinusoidal noise.

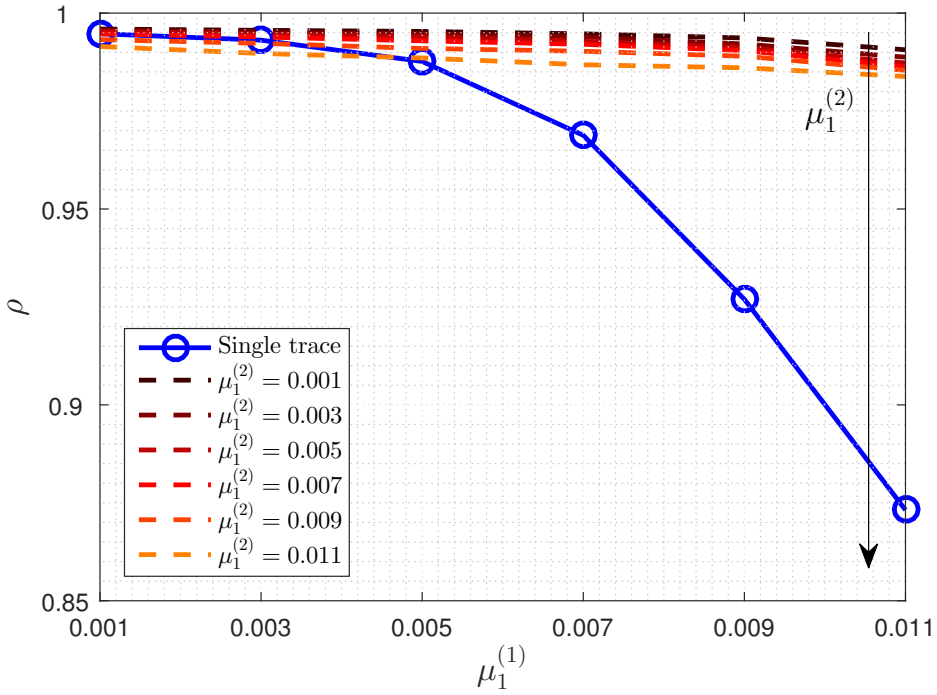


Figure 2.24: ρ as a function of $\mu^{(1)}$ with $\Delta^{(1)} = \Delta^{(2)} = 0.002$, whereas $\mu_1^{(1)}$ and $\mu_1^{(2)}$ are allowed to change.

In Fig. 2.26, we plot ρ when the decoding is jointly performed over the traces from two sniffers as a function of $k^{(1)}$ (the value of k associated with sniffer 1), thus $f = k^{(1)}/T_B$, whereas $k^{(2)}$ (sniffer 2) is kept fixed for each curve. Specifically, we considered $T_B = 1$ ms, which is close to the average length of the data burst, measured from real channel traces (see Fig. 2.11). For this plot, $\mu_1^{(\text{dim})} = 0.001$ for dim $= 1, 2$, the remaining energy levels follow from $\Delta^{(\text{dim})}$, whereas the variances $\Sigma_i^{(\text{dim})}$ are obtained through Eq. (2.5). Also, $\Delta = 0.001$ for both traces. If we do not consider any additive sinusoidal noise signal, this choice of parameters leads to $\rho \simeq 1$. On the contrary, the performance of the runtime mm-wave analyzer decreases as a function of frequency f. This is reasonable, since channel attenuation and noise are no longer stationary within each data burst (see assumption A3, in Section 2.5). Then, taking advantage of the joint information from two different viewpoints can lead to better results (with respect to a single trace) as long as at least one of the traces is not affected by heavy channel fluctuations, see, for example the curves with $k^{(2)} = 0.001$ ($f = 1$ Hz). Finally, the overall performance drastically drops for an increasing A, where A is the maximum amplitude of the sine wave. For this

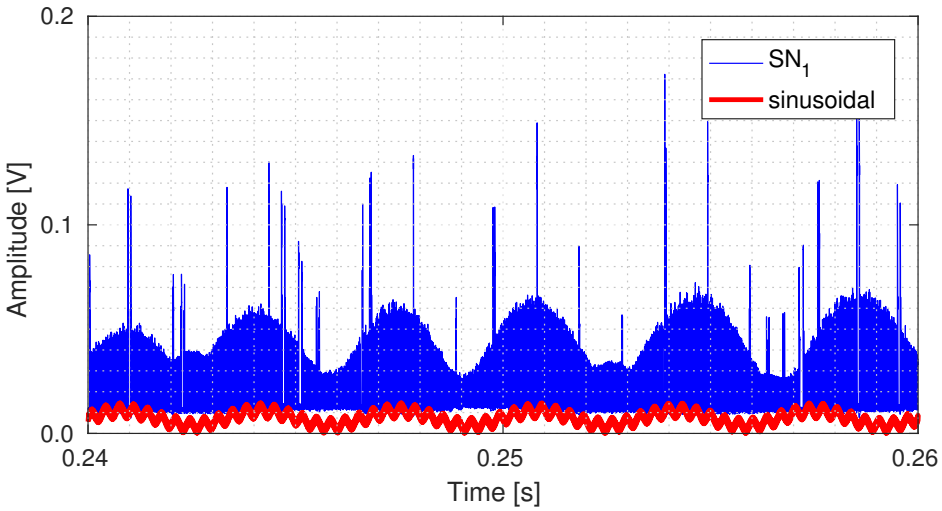


Figure 2.25: An example of sinusoidal noise which was found in the channel dynamics. The reconstructed signal is given as a sum of $N = 3$ sine waves, by looking at the $N = 3$ highest peaks in the Fourier transform of the trace. Specifically, the main component has frequency 300 Hz, amplitude 0.0035 V.

plot, we set A equal to 1, 10, and 100 times the gap between energy levels Δ. It results that $A = \Delta \cdot 100$, combined with values of f above 100 Hz, always results in $\rho < 0.5$.

2.12 Conclusions

In this chapter we have designed and evaluated a machine learning framework to automatically perform protocol layer analysis and diagnose physical layer issues in 60 GHz networks. The main challenge lies in the variability of the channel traces and in the complexity of identifying their structural elements given their noisiness, aperiodicity, and unpredictable behavior. Standard machine learning approaches fail in such a scenario. Our tool uses narrowband physical layer energy traces from one or multiple sniffers to identify lower-layer performance issues. Network planners, administrators, as well as researchers can use it to improve the performance of mm-wave networks even though COTS networking devices provide very limited access to lower-layer information. The developed approach provides a convenient trade-off between the efficient but limited monitoring capabilities of IEEE 802.11ad devices in monitor mode, and the highly detailed but costly decoding of full bandwidth

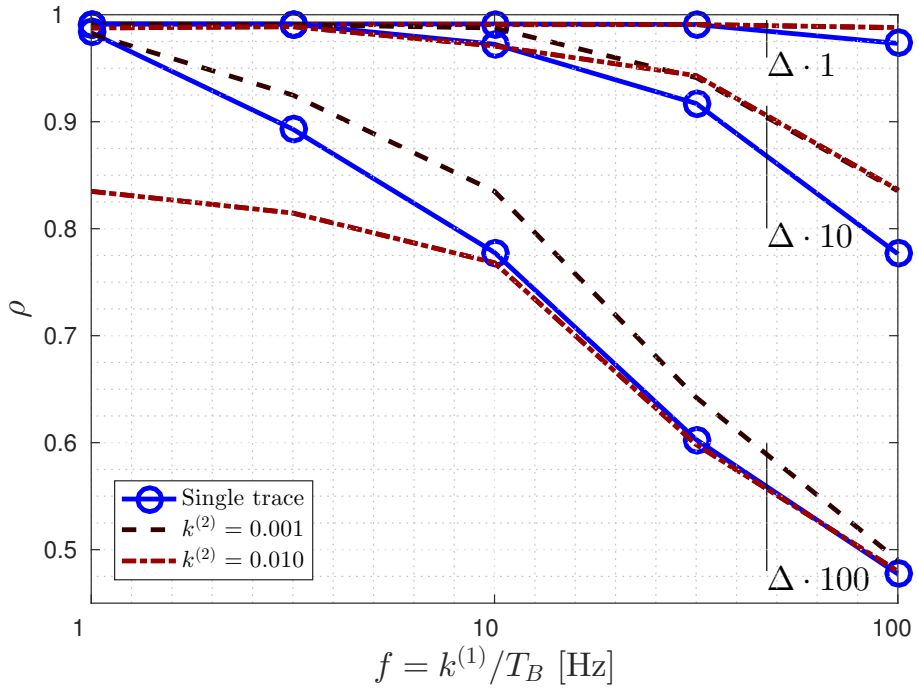


Figure 2.26: ρ in the presence of an *additive* sinusoidal noise with frequency f.

signals through software-defined radios. Our algorithms are tested through an extensive measurement campaign using COTS 60 GHz hardware, considering a range of practical scenarios. Besides, the ability of the framework to correctly identify protocol sequences (beacon pairs, data and acknowledgment frames) from single and multiple channel sniffers is quantified, looking at the impact of channel noise, energy levels and their distribution across different sniffers.

Chapter 3

A Bayesian Forecasting and Anomaly Detection Framework for Vehicular Monitoring Networks

In this chapter, we are concerned with the automated and runtime analysis of vehicular data from large scale traffic monitoring networks. This problem is tackled through localized and small-size Bayesian Networks (BNs), which are utilized to capture the spatio-temporal relationships underpinning traffic data from nearby road links. A dedicated BN is set up, trained, and tested for each road in the monitored geographical map. The joint probability distribution between the cause nodes and the effect node in the BN is tracked through a Gaussian Mixture Model (GMM), whose parameters are estimated via Bayesian Variational Inference (BVI) operating on unlabeled data. Forecasting and anomaly detection are performed on statistical measures derived at runtime by the trained GMMs. Our design choices lead to several advantages: the approach is scalable as a small-size BN is associated with and independently trained for each road and the localized nature of the framework allows flagging atypical behaviors at their point of origin in the monitored geographical map. The effectiveness of the proposed framework is tested using a large dataset from a real network deployment, comparing its prediction performance with that of selected regression algorithms from the literature, while also quantifying its anomaly detection capabilities.

3.1 Introduction

Smart cities are witnessing a digital revolution involving a constellation of sensing technologies, which are being employed to gather a range of environmental, parking and traffic data. Application examples for such data abound: park monitoring networks are being installed in major cities such as Los Angeles [51], San Francisco [52], and Barcelona [53], among others. Parking sensor data can be utilized to identify free parking lots and, moreover, to pinpoint atypical parking patterns [54, 55]. Solutions to manage traffic flows are also being developed. For example, in [56, 57] video processing techniques are used to estimate the traffic density based on video frames from surveillance cameras deployed on traffic light poles, devising smart control strategies for traffic lights. Other works address traffic forecasting, see, e.g., [58]. A common trait of these systems is that data is generated in large amounts and, in turn, its manual inspection is impractical. Moreover, the patterns of interest are often hidden and difficult to observe through a mere visual inspection, even by skilled personnel. Machine learning tools are deemed a natural means to efficiently and effectively process these data.

In this chapter, a Bayesian framework for vehicular traffic monitoring networks is proposed. Its core idea is that information from road links that are in close proximity is likely to be highly correlated with that in the current (target) road link, at any time interval. Moreover, temporal correlation is also relevant, i.e., past observations from nearby links also tend to be correlated with the current reading at any target road. Owing to these facts, we model the spatio-temporal evolution of vehicular streams among multiple road links in large-scale scenarios through *localized* and *small-size* Bayesian Networks (BNs). These, are implemented as Directed Acyclic Graphs (DAG), representing conditional independence relations among random variables. Specifically, a dedicated BN is configured, trained, and tested *for each target road* in the monitored (urban) geographical map. The joint probability distribution between the cause nodes (data utilized for forecasting) and the effect node (data to be predicted at any "current" time, belonging to the target road link) is described through a Gaussian Mixture Model (GMM) whose parameters are estimated via Bayesian Variational Inference (BVI) operating on unlabeled data. Op-

timal forecasting follows from the criterion of Minimum Mean Square Error (MMSE). Moreover, we also perform *anomaly detection* by devising a probabilistic score associated with the marginal conditional distribution of the effect node. The so obtained GMMs are time-dependent, i.e., several GMMs can be estimated for the same target road for different days of the weeks and/or hours of the day. Also, our framework is *distributed*, lightweight, and capable of operating in realtime and, in turn, it appears to be a promising candidate to deal with Internet of Things (IoT) applications in large-scale networks, where new data is to be processed on-the-fly. The key features of the model are: i) the approach is scalable as a BN is associated with and independently trained for each road, ii) spatio-temporal information is considered (for increased robustness and accuracy of the statistical model so obtained) and iii) the localized nature of the framework allows flagging atypical behaviors at their point of origin in the monitored geographical map. In addition, the approach is here validated against a number of popular regression schemes from the literature, to quantify its predictive power, testing it on data from a large real-world deployment, featuring readings from 686 measurement points for a full year. Finally, we quantify the framework's capability of detecting anomalies in the presence of injected noise, which we precisely control in terms of power, location (road link) and position in time. Our numerical results reveal that a localized and small size DAG, implemented for each road in the monitored area suffices to obtain very good prediction and anomaly detection accuracies. This means that large monitoring networks can be tackled by training independent and small-size DAGs, a process that can be efficiently parallelized across disjoint processors, ensuring scalability as the size of the network increases.

This chapter is organized as follows. In Section 3.2, the state of the art on anomaly detection and prediction in vehicular data is reviewed. In Section 5.2.1, the Bayesian framework is formulated, detailing the dataset (Section 3.3.1), the BN/GMM models (Section 3.3.2), the use of real data for training/validation (Section 3.3.3). Numerical results are provided in Section 3.4, assessing the prediction (Section 3.4.1) and anomaly detection (Section 3.4.2) performance of the new scheme. Future research directions are discussed in Section 5.5.

3.2 State of the Art Analysis

A substantial amount of research has been carried out to provide anomaly detection techniques in a wide range of application domains such as cyber-intrusion detection, fraud detection, medical anomaly detection, industrial damage detection, image processing, textual anomaly detection, sensor networks, etc., and has been reviewed in several surveys [59–61]. We direct the reader to these sources for a detailed and comprehensive treatment of the anomaly detection problem. Next, we solely focus on works dealing with traffic analysis.

In [62], the authors introduce a *road segment based* anomaly detection problem, observing the road segment whose traffic condition deviates the most from the expected behavior. This work departs from other scientific papers [63, 64], which are region/grid specific, and not road based. First, a deviation-based method is put forward to quantify the anomaly by means of a score in the range $[0, 1]$. Second, a diffusion-based algorithm exploiting a *heat diffusion model* is proposed to infer the major anomaly causes on the transportation network, given that an abnormal traffic trend in a road segment can trigger another abnormal traffic trend in a road segment located nearby. This model does not include historical information streams from adjacent road links, but represents the expected behavior of the road segment as a normal random variable. In [65], a more sophisticated scheme is presented. There, a Temporal Outlier Discovery (TOD) framework is proposed to quantify the anomaly based on drastic changes in the agglomerated temporal information of the *entire dataset*. Specifically, at each time step, every road segment checks its similarity with respect to every other segment, and historical similarity values are stored in a temporal neighborhood vector. Anomaly detection for each road is accomplished by jointly considering mobility data from all the streets in the data set. While this should be quite robust in terms of detection capability, as it is expected to be reliable even when all the neighborhood of the current road is experiencing a traffic anomaly, it is *hardly scalable* and *difficult to train and use at runtime* as the number of roads to monitor increases. It is therefore deemed impractical for the large-scale network that we consider in this chapter. [66] adopts a Bayesian framework, as we do here. The

authors focus on the short-term traffic flow *forecasting* task, which amounts to determine the traffic condition of a *target road* in the near future, usually within a time range of $5 - 30$ minutes. Historical information streams from the adjacent road links and the target link are taken into account by means of Bayesian networks, where the joint probability distribution between the *cause* nodes (directional information streams from the adjacent road links) and the *effect* node (traffic condition of the target link in the next time interval, to be predicted) is represented through a Gaussian Mixture Model (GMM), and forecasting is performed by computing the expectation of the Probability Density Function (PDF) associated with the predicted traffic condition. Also, the authors improve the analysis carried out in [67, 68] by including historical information streams from the adjacent road links and the target link. However, besides forecasting, the authors do not provide any reasonable anomaly detection criterion to state whether the distribution associated with the predicted traffic condition deviates from the expected behavior. In a sense, the authors do not exploit Bayesian networks to their full extent. Furthermore, the scheme is tested on a limited dataset (only $2,400$ sample points from real world traffic data), which make their numerical evaluation quite preliminary.

In this chapter, we propose a Bayesian framework for vehicular traffic monitoring networks. Our approach bears similarities with [66], as both use a BN and an associated GMM. Nevertheless, we demonstrate the effectiveness of localized Bayesian networks in large and real datasets, extending their capabilities to anomaly detection, validating the framework with a large real-world deployment and comparing it against a number of regression approaches from the literature, to test its ability to capture the spatio-temporal structure underpinning real data.

3.3 Bayesian Framework

As briefly discussed above, in vehicular traffic monitoring networks it makes sense to exploit information from multiple road links at past time intervals to forecast the traffic flow in any target road link at any (current) time interval. In the following, we consider a (any) *target* road in the topology, interchangeably referring to it as the *current* road. The objective of the BN is to jointly track

current and past speed values for the current road, along with past speed values of *upstream* roads. To cut down the number of connections among multiple road links in the associated DAG, we have chosen to only consider the target road link and its upstream road links in the DAG construction, while neglecting downstream links. This choice allows reducing the number of possible connections in the DAG and, in turn, its complexity. This can be justified from the way BNs represent conditional independence relations among random variables: *a node is independent of its ancestors given its parents, where the ancestor/parent relationship is with respect to some fixed topological ordering of the nodes* [69].

3.3.1 Traffic Readings

Mobility data was acquired from 686 Bitcarrier sensor nodes scanning Bluetooth/Wi-Fi signals of mobile devices traveling on road links [70] for a full year. This sensing system has been deployed by Worldsensing in a major city (details are not provided to protect the company's industrial plans). Readings were taken with a time granularity of 5 minutes (the time slot duration), and each reading from any of the sensors corresponds to the average traveling speed (in [km/h]) gathered during the corresponding time slot (with timestamps in UTC format). For 686 Bitcarrier sensor nodes, this amounts to a total of $54,591,660$ data points in the considered time period from January 2016 to December 2016. To account for missing/unavailable data points, the entire dataset is pre-processed via linear interpolation, jitter removal, and low-pass filtering. After this pre-processing phase, we obtain time series (one per sensor) with one point every 5 minutes. Our Bayesian learning routines operate on these time slotted signals.

3.3.2 Probabilistic inference via GMM

As per the above discussion, we assume that the traffic flow in the current road link at current time interval is independent of other road links, given the traffic flow in the current link and in its upstream road links at past time intervals. Taking advantage of conditional independence relations, we can analyze the trends of the current link statistically, computing the marginal conditional distribution of the effect (i.e., current) node, as we now describe.

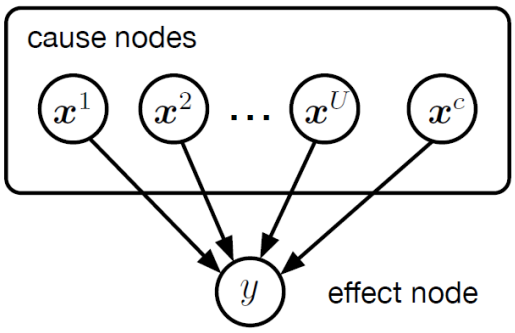


Figure 3.1: Bayesian network associated with any target node in the physical network topology.

At any current time t, let $\boldsymbol{z} = (\boldsymbol{x}, y)$ denote a multidimensional random vector: $\boldsymbol{x}$ is a random vector containing the random variables (r.v.s) associated with the cause nodes in the DAG and y denotes a (scalar) r.v. representing the speed value of the current node (to be estimated). In the BN, we keep track of past measures: if t is the current time slot, the memory spans those samples in $\{t-1, \ldots, t-W\}$, where W is the memory size. Let U be the number of upstream roads in the physical network topology. For each upstream road we define a cause node u in the DAG with r.v.s. $\boldsymbol{x}^u = (x^u_{t-1}, x^u_{t-2}, \ldots, x^u_{t-W})$, where $u = 1, \ldots, U$ and x^u_{t-i} represents the speed reading for road u at time $t-i$, with $i = 1, \ldots, W$. For the current road, we define a further cause node in the DAG, with associated vector $\boldsymbol{x}^c = (x^c_{t-1}, x^c_{t-2}, \ldots, x^c_{t-W})$. Hence, $\boldsymbol{x}$ is obtained as the concatenation of $\boldsymbol{x}^u$ $(u = 1, \ldots, U)$ and $\boldsymbol{x}^c$, i.e., $\boldsymbol{x} = (\boldsymbol{x}^1, \boldsymbol{x}^2, \ldots, \boldsymbol{x}^U, \boldsymbol{x}^c)$. The r.v. y contains the speed at the current time t for the current road c. A diagram of the just described Bayesian network is shown in Fig. 3.1. For the numerical results in this chapter we considered a memory of $W = 5$ time slots, i.e., 25 minutes of historical data.

To approximate the joint probability distribution of $\boldsymbol{z}$ we adopt a GMM. Besides forecasting, we also perform *anomaly detection* by taking into account a probabilistic score associated with the marginal Cumulative Distribution Function (CDF) of the effect node, as we detail shortly in Section 3.4.2. We remark that, to the best of our knowledge, the use of a probabilistic score associated with the marginal CDF of the effect node sets us apart from existing works on anomaly detection via probabilistic inference.

The GMM is defined as:

$$P(\boldsymbol{z}|\Theta) = \sum_{m=1}^{M} \alpha_m P_m(\boldsymbol{z}|\boldsymbol{\theta}_m)\,, \tag{3.1}$$

where M is the number of Gaussians in the mixture, whose parameters are $\Theta = \{\alpha_1, \ldots, \alpha_M, \boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_M\}$. α_m are scalars such that $\sum_{m=1}^{M} \alpha_m = 1$. Each $P_m(\cdot|\boldsymbol{\theta}_m)$ is a Probability Density Function (PDF) characterized by $\boldsymbol{\theta}_m = (\boldsymbol{\mu}_m, \boldsymbol{\Sigma}_m)$, $m = 1, \ldots, M$, i.e., $P_m(\boldsymbol{z}|\boldsymbol{\theta}_m) = G(\boldsymbol{z}; \boldsymbol{\mu}_m, \boldsymbol{\Sigma}_m)$. In this chapter, the parameters are estimated via Bayesian Variational Inference (BVI) with unlabeled data [69]. BVI can be seen as an extension of the Expectation-Maximization (EM) algorithm from a maximum a posteriori estimation of the single most probable value of each parameter to a complete Bayesian estimation, which computes (an approximation to) the entire posterior distribution of the parameters and latent variables. The marginal conditional distribution of the effect node is computed as:

$$P(y|\boldsymbol{x}) = \frac{P(\boldsymbol{x}, y)}{P(\boldsymbol{x})} = \frac{P(\boldsymbol{x}, y)}{\sum_y P(\boldsymbol{x}, y)}\,, \tag{3.2}$$

where the sum is over the domain of r.v. y and $P(\boldsymbol{x}, y) = P(\boldsymbol{z}|\Theta)$. Exploiting the properties of Gaussian PDFs [71], we can derive a concise optimal forecasting relationship, which allows new data to be processed on-the-fly:

$$\begin{aligned}
\widehat{y} &= \int y P(y|\boldsymbol{x}) dy \\
&= \int y \left[\sum_{m=1}^{M} \beta_m G(y|\boldsymbol{x}; \mu_{m,y|\boldsymbol{x}}, \Sigma_{m,y|\boldsymbol{x}}) \right] dy = \\
&= \sum_{m=1}^{M} \beta_m \left[\int y G(y|\boldsymbol{x}; \mu_{m,y|\boldsymbol{x}}, \Sigma_{m,y|\boldsymbol{x}}) dy \right] = \\
&= \sum_{m=1}^{M} \beta_m \mu_{m,y|\boldsymbol{x}}\ .
\end{aligned} \tag{3.3}$$

Specifically, for $m = 1 \ldots, M$,

$$\begin{aligned}
\mu_{m,y|\boldsymbol{x}} &= \mu_{m,y} - \boldsymbol{\Sigma}_{m,y\boldsymbol{x}} \boldsymbol{\Sigma}_{m,\boldsymbol{x}\boldsymbol{x}}^{-1} (\boldsymbol{\mu}_{m,\boldsymbol{x}} - \boldsymbol{x}) \\
\Sigma_{m,y|\boldsymbol{x}} &= \Sigma_{m,yy} - \boldsymbol{\Sigma}_{m,y\boldsymbol{x}} \boldsymbol{\Sigma}_{m,\boldsymbol{x}\boldsymbol{x}}^{-1} \boldsymbol{\Sigma}_{m,\boldsymbol{x}y} \\
\boldsymbol{\mu}_m &= (\boldsymbol{\mu}_{m,\boldsymbol{x}}, \mu_{m,y}) \\
\boldsymbol{\Sigma}_m &= \begin{pmatrix} \boldsymbol{\Sigma}_{m,\boldsymbol{x}\boldsymbol{x}} & \boldsymbol{\Sigma}_{m,\boldsymbol{x}y} \\ \boldsymbol{\Sigma}_{m,y\boldsymbol{x}} & \Sigma_{m,yy} \end{pmatrix} \\
\beta_m &= \frac{\alpha_m G(\boldsymbol{x}; \boldsymbol{\mu}_{m,\boldsymbol{x}}, \boldsymbol{\Sigma}_{m,\boldsymbol{x}\boldsymbol{x}})}{\sum_{n=1}^{M} \alpha_n G(\boldsymbol{x}; \boldsymbol{\mu}_{n,\boldsymbol{x}}, \boldsymbol{\Sigma}_{n,\boldsymbol{x}\boldsymbol{x}})} \ .
\end{aligned} \tag{3.4}$$

For the numerical results in this chapter, M is set to 20 and the GMM parameters Θ are initialized via K-means clustering [69].

3.3.3 Data Matrices and Typical Weekly Profiles

Upon collecting and pre-processing the raw data from the sensor nodes, we define two additional data objects: 1) the *data matrix* and 2) the *typical weekly profile*. These are defined for *each* target road as follows. 1) The data matrix is the collection of readings gathered from the target road (effect node in the DAG) for all times t and from the cause nodes in the previous W time slots (i.e., variable $\boldsymbol{z}$, see Section 3.3.2). 2) The typical weekly profile is a polished time series, which is *anomaly-free* and will be used in Section 3.4.2 as a ground truth signal to quantify the Bayesian framework's ability to detect anomalies. This profile is obtained as follows: for each time t in a certain day of the week d (e.g., Monday), we take a window of data points before and after it (the window size is equal to 15 minutes, for a total of 30 minutes with the current time t being in the center of it). We then collect all the data points belonging to this time window for this *same* day of the week for an entire year (e.g., all Mondays in a year for a window of 30 minutes centered on t) and compute the PDF associated with the readings in this time window. For each time t, we finally take the median of this PDF, which becomes the new datapoint for the considered target road at time t and day d. The typical weekly profile may be seen as a sort of expected behavior for the traffic on each road across the entire year.

3.4 Numerical Results

In this section, we assess the performance of the Bayesian framework of Section 5.2.1 (referred to in the following as `Bayesian net`) from two points of view: 1) its **forecasting capability** (Section 3.4.1) and 2) its **anomaly detection accuracy** (Section 3.4.2).

1) Forecasting: for the forecasting capability, `Bayesian net` is evaluated in terms of the error (*residual*) between the original and the predicted data points, i.e., $\text{Res}(y_t, \widehat{y}_t) = y_t - \widehat{y}_t$, where y_t is the speed reading at time t for any of the deployed sensors with $t = 1, \ldots, T$, with T being the number of readings for that sensor in a full year. The following state-of-the-art algorithms are also considered in the performance assessment: Last Sample ("Last sample"), Bayesian Regression ("Bayes regr"), Lasso Regression ("Lasso regr"), Linear Regression ("Linear regr"), Ridge Regression ('Ridge regr"), and Regression Tree ("Tree regr"). Numerical results are obtained using 75% of the data matrix (*training set*) to estimate the GMM parameters for each road in the physical network topology, whereas the remaining 25% (*validation set*) is employed to test the obtained GMM. The forecasting capability of the above schemes is obtained aggregating the results of the data points in the validation set for each sensor node, computing various statistics for the residuals. The prediction performance is shown and discussed in Section 3.4.1.

2) Anomaly detection: the typical weekly profile of Section 3.3.3 is utilized as a ground-truth signal to evaluate the anomaly detection accuracy, as follows. For each sensor node, we inject artificial anomalies consisting of random noise sequences whose duration is $D = 5$ time slots (5 minutes each), in random non-overlapping positions of the sensor's weekly profile. This artificial signal is a D-sequence of additive, zero-mean white Gaussian noise samples with standard deviation σ_n. For the anomaly detection, a probabilistic *score* is obtained from the marginal distribution of the effect node applied to the noisy weekly profile. Hence, we compare the anomaly rating against a sensor-specific threshold ζ. This threshold is set according to a network requirement quantifying the percentage of anomalies that a sensor is expect to flag in a day in the considered vehicular network. To evaluate the anomaly detection accuracy, we aggregate the results of all sensor nodes, each evaluated considering a noisy

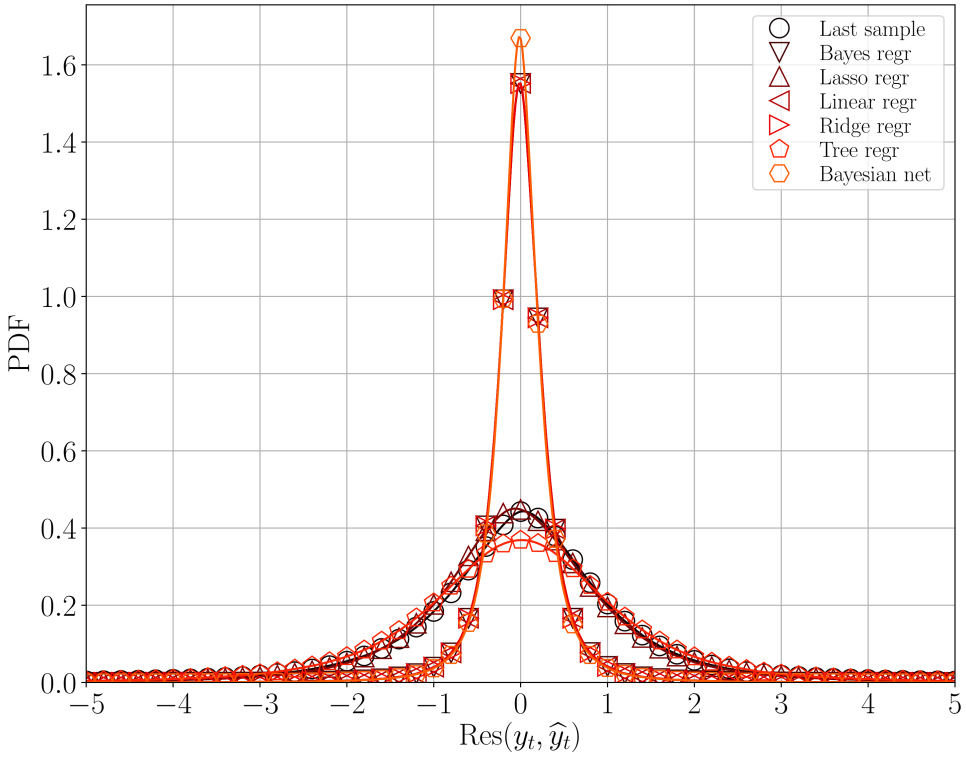


Figure 3.2: PDF of the residual Res$(y_t, \widehat{y}_t)$.

version of its weekly profile. Numerical results are discussed in Section 3.4.2.

3.4.1 Forecasting Capability

In Fig. 3.2 we show the PDF of the residual Res$(y_t, \widehat{y}_t)$. From this plot we see that `Bayesian net` has a greater forecasting capability that the other schemes. In fact, its PDF is more sharply peaked around zero, meaning a smaller difference between the actual and the predicted samples, $|y_t - \widehat{y}_t|$. As a consequence, the Root Mean Squared Error (RMSE) is also smaller, where we define RMSE $= (\sum_{t=1}^{T}(y_t - \widehat{y}_t)^2/T)^{1/2}$.

Fig. 3.3 shows the (empirically measured) complementary CDF of the residual Res$(y_t, \widehat{y}_t)$, i.e., $P[\text{Res}(y_t, \widehat{y}_t) \geqslant R])$, where R is kept fixed for all sensor nodes. `Bayesian net` reports a lower number of events for which Res$(y_i, \widehat{y}_i) \geqslant R$ and, in turn, its curve in Fig. 3.3 decreases the fastest, reaching a minimum of 10^{-6} for $R = 11$ km/h. These results indicate that a Bayesian approach is an appropriate tool to perform forecasting, achieving competitive performance with the best algorithms from the literature.

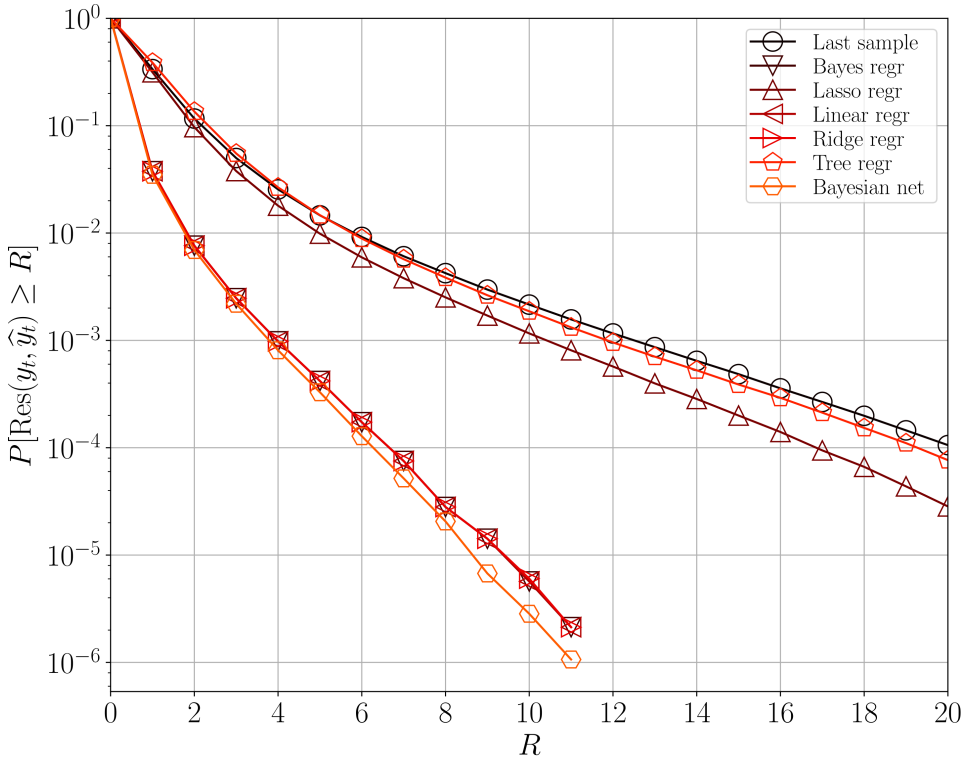


Figure 3.3: Complementary CDF of the residual $\mathrm{Res}(y_t, \widehat{y}_t)$.

3.4.2 Anomaly Detection Accuracy

Next, the typical weekly profiles are used as ground-truth signals to evaluate the anomaly detection accuracy. This makes it possible to work with *labeled* time series and, in turn, to precisely quantify the classification performance of `Bayesian net` in terms of: i) `Precision`, ii) True Positive Rate (`TPR`) and iii) `F measure`.

In Fig. 3.4, we plot a segment from a typical weekly profile for sensor (target) node with ID 1000003 and cause nodes 1000003, 1000090, 1000197 (the speed traces in this plot are synchronized in time). According to the DAG construction method of Section 3.3.2, the target node 1000003 is also among the cause nodes in the DAG, and it contains the past readings $\{x^c_{t-1}, \ldots, x^c_{t-W}\}$. Instead, the effect node contains y_t, i.e., the speed measured at the target road at time t.

Let us now consider Fig. 3.4 to illustrate how anomalies are injected and detected. For each sensor node, we inject random artificial anomalies of length D time slots in random non-overlapping positions, as explained above. Hence, we compute a probabilistic *score* from the marginal CDF of the effect node (that is computed taking the noisy profile as the input sequence). Whenever the anomaly rating exceeds the (sensor-specific) threshold ζ, the corresponding time slot is flagged as containing an anomaly (see the circular markers in the top subplot of Fig. 3.4). In the bottom subplot, we show the score for the trace in the top subplot, which is defined as:

$$\text{SCORE}_t = \begin{cases} \log_{10}(C_t) - \log_{10}(0.5) & C_t < 0.5 \\ -\log_{10}(1 - C_t) + \log_{10}(0.5) & C_t \geqslant 0.5\,, \end{cases} \tag{3.5}$$

where $C_t = \text{CDF}(y_t|\boldsymbol{x}_t)$ is the Cumulative Distribution Function computed for y_t and conditioned on the past readings ($\boldsymbol{x}_t$ in the DAG). Using Eq. (3.5), the further the current speed y_t is from the median of the PDF $P(y_t|\boldsymbol{x}_t)$, the greater is the score. A high score means that the speed value y_t is atypical with respect to what would be predicted by the (marginalized) PDF. In this example, we use $\sigma_n = 5$ km/h, which is the maximum noise level that was considered in our experiments. As for the threshold ζ, for Fig. 3.4 we have set $\log_{10}$(no. of anomalies/no. of samples) $= -3$ (application *requirement*), where

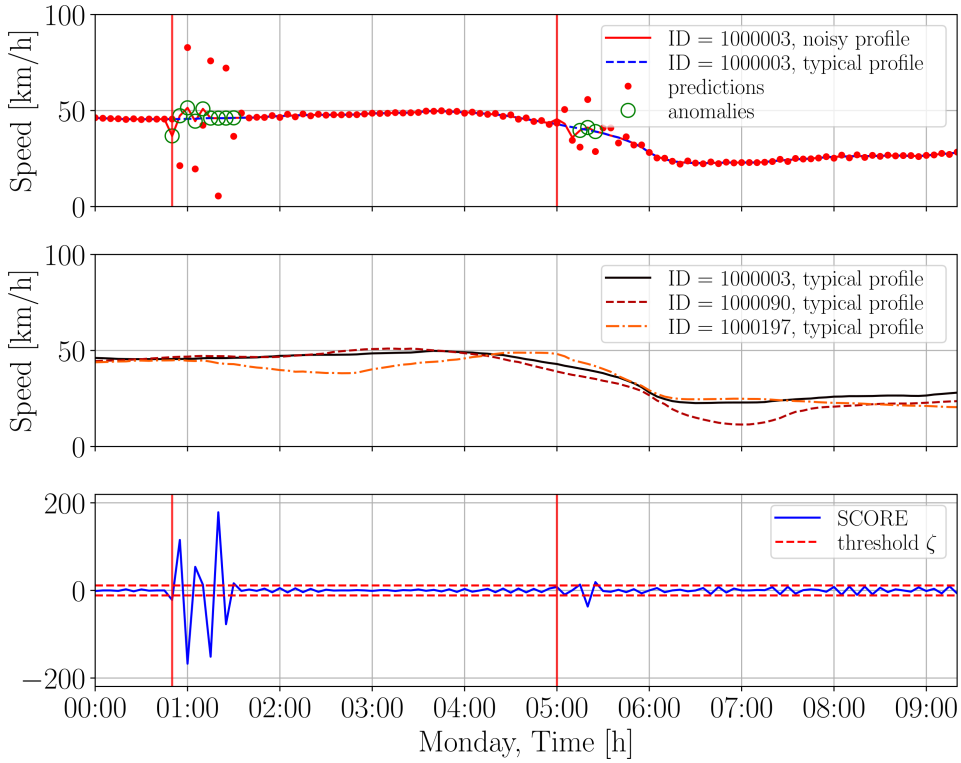


Figure 3.4: Anomaly detection example for sensor node with ID 1000003 and cause nodes 1000003, 1000090, 1000197.

"no. of samples" is the total number of data points in the validation set for each sensor node; ζ is numerically found to meet this. For a given threshold ζ, anomalies are detected (circular markers in the top subplot of Fig. 3.4) by assessing whether $|\text{SCORE}_t| \geqslant \zeta$. Note that in Fig. 3.4 we consider a single target road and, as such, the given application requirement is used to compute a single threshold ζ for that road. However, for an entire network, this same requirement is employed to derive one threshold per DAG (i.e., one for each target road in the physical topology).

Referring to α as the total number of (artificial) anomalies that were injected, we have that the number of time slots that may be possibly affected is $S = \alpha(D + W)$. This is because anomalies are non-overlapping, each anomaly lasts D time slots and its effect could propagate for W further time slots due to the memory in the DAG, i.e., $D + W$ is the support of a single anomaly. Given this, we define $\mathcal{S}$ (with $|\mathcal{S}| = S$) as the set of time slots that could possibly contain an anomalous reading, as per the previous reasoning. From this definition, it follows that the maximum number of True Positives (`TP`) is

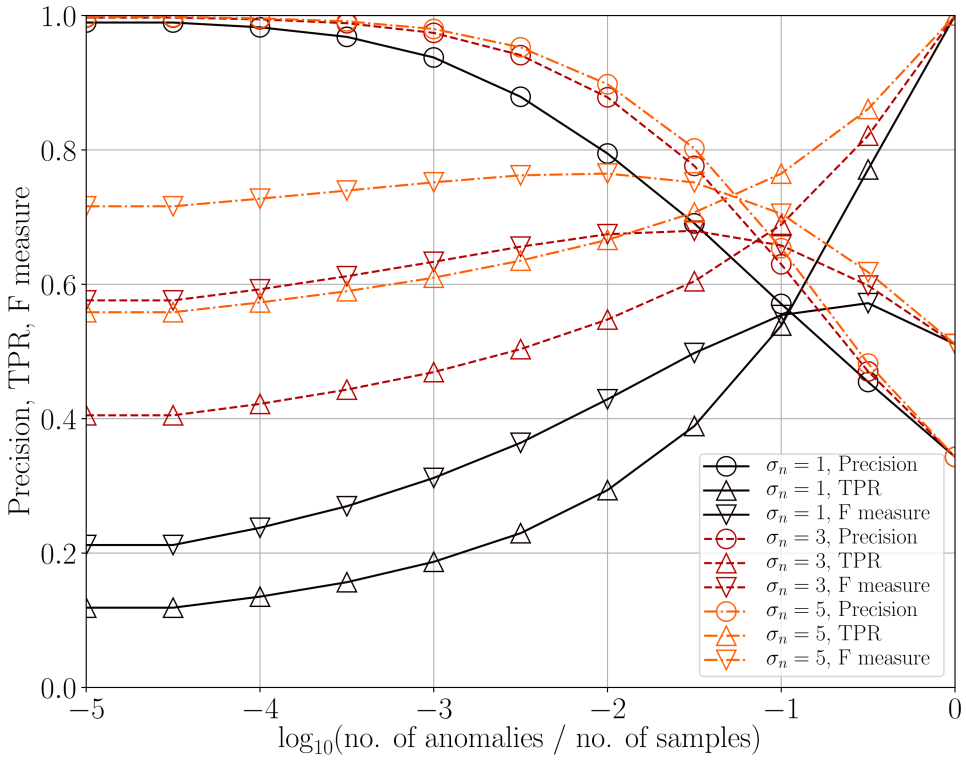


Figure 3.5: BN performance, when used as an anomaly classifier.

$|\mathcal{S}| = S$. We also track the number of False Negatives (FN), False Positives (FP), and True Negatives (TN) and with $\sum X$ we mean the total number of time slots that are flagged as being of type X, with $X \in \{\texttt{TP}, \texttt{FP}, \texttt{TN}, \texttt{FN}\}$. For instance, in the example of Fig. 3.4, we have $\alpha = 2$, $D = 5$, $W = 5$, $S = 20$, $\sum \texttt{TP} = 13$, $\sum \texttt{FN} = 7$, $\sum \texttt{FP} = 0$, and $\sum \texttt{TN} = T' - S$, where T' is the number of time slots in the graphs. For the following results, we used $\alpha = 70$, which corresponds to an average of 10 artificial anomalies that are added per day.

Fig. 3.5 shows the classification performance of the proposed score-based anomaly detector in terms of: i) `Precision`, ii) `TPR` and iii) `F measure` (F). These metrics are defined as follows: $\texttt{Precision} = \sum \texttt{TP} / \sum(\texttt{TP} + \texttt{FP})$, $\texttt{TPR} = \sum \texttt{TP} / \sum(\texttt{TP} + \texttt{FN})$, and F, which is a weighted average of `Precision` and `TPR`, i.e., $F = 2\sum \texttt{TP} / (2\sum \texttt{TP} + \sum(\texttt{FP} + \texttt{FN}))$. In Fig. 3.5, these metrics are plotted as a function of the application requirement (i.e., $\log_{10}$(no. of anomalies / no. of samples)), which is reported in the abscissa. As an example, a requirement equal to zero means that *all* the time samples are flagged as containing an anomaly and, as such, the true positive rate is $\texttt{TPR} = 1$. However, in this case, the `Precision` is heavily impacted by the number of false positives (FP), which is at least $T - S$, where S is the maximum number of time slots affected by real anomalies (true positives) and T corresponds to the number of time slots in the time series. As expected, the anomaly detection accuracy increases with an increasing noise level, approaching $F = 0.8$ for $\sigma_n = 5$ km/h (when the requirement on the x-axis is -2). Also, a higher `Precision` entails a smaller `TPR` and vice-versa.

In Fig. 3.6, we show the Receiver Operating Characteristic (ROC) space, obtained plotting the `TPR` (i.e., `Sensitivity`) against the False Positive Rate, $\texttt{FPR} = \sum \texttt{FP} / \sum(\texttt{FP} + \texttt{TN})$ (i.e., $1 - \texttt{Specificity}$) varying the application requirement as a free parameter. This space shows the discrimination capability of the score-based anomaly classifier as we vary the requirement. Ideally, we would like to get $\texttt{TPR} \to 1$ and $\texttt{FPR} \to 0$, which means that desirable working points lie in the upper-left corner of the ROC space. As expected, the anomaly detection accuracy increases with an increasing noise level (increasing σ_n). Moreover, we can further improve the performance of the proposed Bayesian framework through the following `TP` aggregation criterion ("TP aggr."). As discussed above, each anomaly has an associated support of $D + W$ time slots. Hence, whenever the score exceeds the threshold at any given time instant,

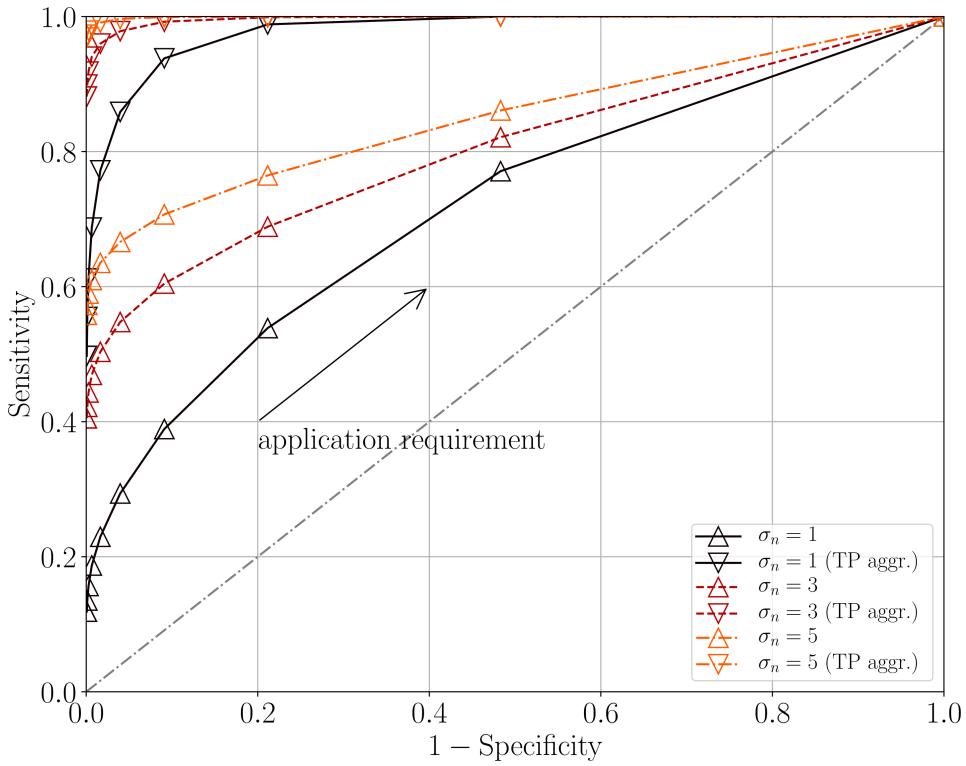


Figure 3.6: The ROC space.

$D + W$ data points per anomaly instance are counted as true positives if at least one alarm is raised within the real (and known) support of the injected anomaly. With aggregation, the ROC curves effectively move towards the upper-left corner of the space, leading to some major improvement. For the example in Fig. 3.4, this strategy leads to $\sum \texttt{TP} = 20$, $\sum \texttt{FN} = 0$, $\sum \texttt{FP} = 0$, and the total number of true negatives is $\sum \texttt{TN} = T' - S$, where T' is the number of time slots in the plot. The rationale about this approach is that, if there is at least one alarm within the support of an anomaly instance, in practice, this may be sufficient to declare the entire anomaly instance as detected.

3.5 Open Research Questions

With this analysis we have demonstrated that localized Bayesian networks are an efficient and lightweight means to tackle prediction and anomaly detection problems in large vehicular networks. Nevertheless, a few research avenues remain open. An automated procedure could be set up to adapt the memory

size W and the considered upstream nodes in the DAG in a sensor-specific fashion. Suitable dimensionality reduction tools could be used to reduce the complexity associated with the BN training task. A more refined Bayesian model could be defined, associating a state to each sensor according to the locally experienced traffic condition (e.g., normal, congested) and specializing the GMMs to it.

Chapter 4

Online Power Management Strategies for Energy Harvesting Mobile Networks

The design of self-sustainable Base Station (BS) deployments is addressed in this chapter. We target deployments featuring small BSs with Energy Harvesting (EH) and storage capabilities. These BSs can use ambient energy to serve the local traffic or store it for later use. A dedicated power packet grid is utilized to transfer energy across them, compensating for imbalance in the harvested energy or in the traffic load. Some BSs are offgrid, i.e., they can only use the locally harvested energy and that transferred from other BSs, whereas others are ongrid, i.e., they can additionally purchase energy from the power grid. Within this setup, an optimization problem is formulated where: harvested energy and traffic processes are estimated (at runtime) at the BSs through Gaussian Processes (GPs), and a Model Predictive Control (MPC) framework is devised for the computation of energy allocation and transfer across base stations. The combination of prediction and optimization tools leads to an efficient and online solution that automatically adapts to energy harvesting and load dynamics. Numerical results, obtained using real energy harvesting and traffic profiles, show substantial improvements with respect to the case where the optimization is carried out without predicting future system dynamics. The main improvements are in the outage probability and in the amount of energy purchased from the power grid, that is more than halved for the same load.

4.1 Introduction

The massive use of Information and Communications Technologies (ICT) is increasing the amount of energy drained by the telecommunication infrastructure and its footprint on the environment. Forecast values for 2030 are that 51% of the global electricity consumption and 23% of the carbon footprint by human activity will be due to ICT [72]. As such, energy efficiency and self-sufficiency are becoming key considerations for any development in the ICT sector.

In this chapter, we advocate future networks where small Base Stations (BSs) are densely deployed to offer coverage and high data rates, and Energy Harvesting (EH) hardware (e.g., solar panels and energy storage units) is installed to power them [73]. BSs collect energy from the environment, and have a local energy storage, which they can use to accumulate energy when the harvested inflow is abundant. This local energy reserve can be utilized to serve the local traffic and can be transferred to other BSs (energy routing) to compensate for imbalance in the harvested energy or in the traffic load, see Fig. 4.1. Some of the Base Stations (BSs), referred to as *ongrid*, are connected to the power grid, whereas the others are *offgrid* and, as such, rely on either the locally harvested energy or on the energy transferred from other BSs. Within this setup, intelligent policies are to be devised to transfer the surplus energy to *offgrid* BSs, to ensure the self-sustainability of the mobile system. Energy transfer is an important feature of these networks and can be accomplished in two ways: i) through Wireless Power Transfer (WPT) or ii) using a Power Packet Grid (PPG) [74]. For i), previous studies [75] have shown that its transfer efficiency is too low for it to be a viable solution when distances exceed a few meters, but ii) looks promising. In analogy with communications networks, in a PPG a number of power *sources* and power *consumers* exchange power (Direct Current, DC) in the form of "packets", which flow from sources to consumers thanks to power lines and electronic switches. The energy routing process is controlled by a special entity called the *energy router* [76]. Following this architecture, a local area packetized power network consisting of a group of energy subscribers and a core energy router is presented in [77], where a strategy to match energy suppliers and consumers is devised.

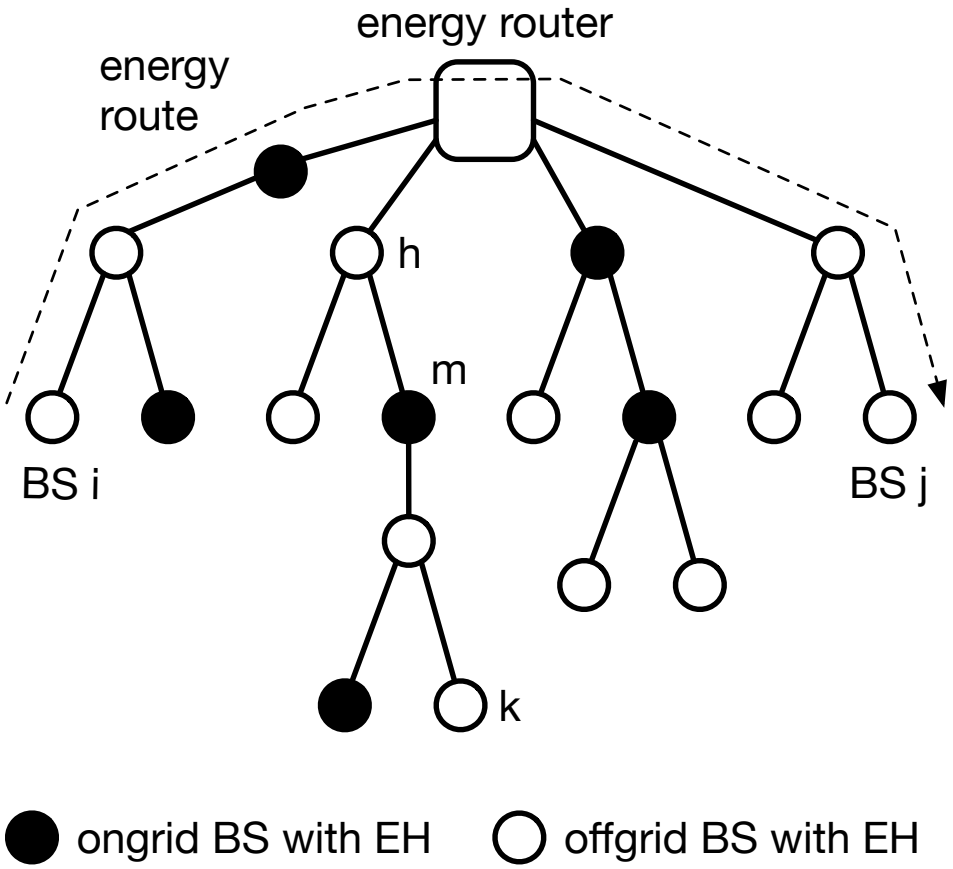


Figure 4.1: Power packet grid topology.

Within this setting, in the present chapter the allocation and transfer of energy among the BSs is performed through the PPG infrastructure, where a centralized energy router is responsible for deciding the power allocation and transfer among the BSs over time (Fig. 4.1). This energy allocation and transfer problem is solved devising an online framework combining: 1) pattern learning (time series forecasting), 2) Model Predictive Control (MPC), 3) energy allocation and 4) energy routing, see Fig. 4.2. **Pattern learning** is performed via Gaussian Processes (Gaussian Processes (GPs)), to learn the BS energy harvesting and consumption (load) patterns over time. This knowledge is utilized within the multi-step ahead **MPC** block, that is in charge of determining the role of each BS, i.e., whether it should act as an energy *source* or *consumer*, and the maximum energy amount that it can either supply (if acting as a source) or demand (consumer), in order to keep the BS system as much as possible energetically self-sufficient over time. The **energy allocation** block

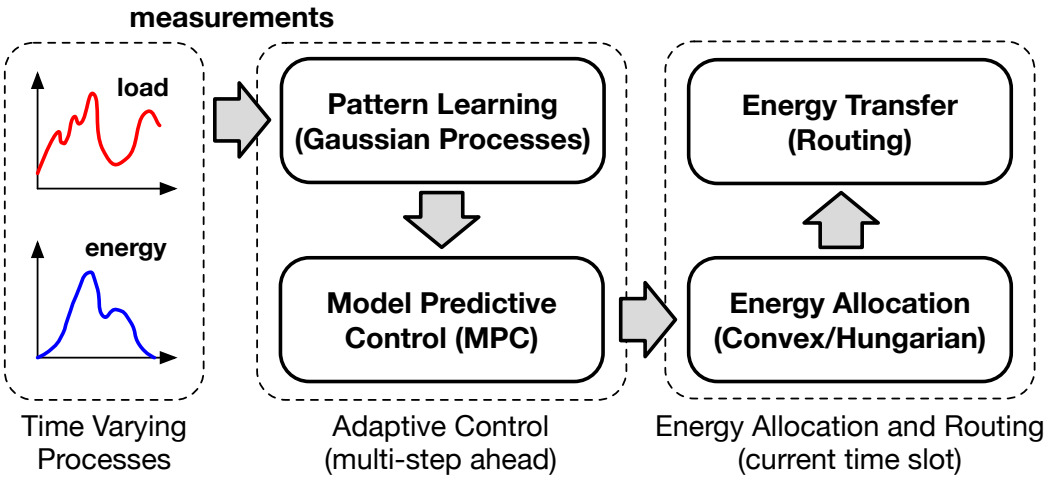


Figure 4.2: Overview of the optimization framework.

computes the actual amount of energy to transfer from energy sources to consumers: two schemes are proposed, one based on convex optimization and one, used as a benchmark, based on an optimal assignment problem, that is solved via the Hungarian method. Finally, once the energy allocations are computed, the fourth block accomplishes the **energy routing** within the network. An algorithm is presented to find a feasible schedule that implements the required energy transfers from sources to consumers. Since the PPG is operated in a Time Division Multiplexing (TDM) fashion, each power link can only be used for a single energy transfer operation at a time. Thus, the proposed routing strategy seeks to find disjoint routes between energy sources and consumers, while minimizing the time needed to perform the energy transfer. Further details are provided in Section 4.4.

The solution extends a previous work in [78], adding *online* control and *foresighted* optimization, and obtaining remarkable improvements, see Section 4.5. Although the considered online optimization problem can be solved with other tools, such as a monolithic formulation or dynamic programming, the presented decomposition into four sub-problems and the use of MPC make it possible to deal with low-complexity convex problems, without introducing significant approximations and/or quantization to the involved variables. The resulting approach is practical and appealing for real-world applications.

Numerical results, obtained with real-world harvested energy traces and traffic load patterns, show that the proposed approach effectively keeps the

outage probability[1] to nearly zero for a wide range of traffic load and system configurations. Also, the amount of energy purchased from the power grid to operate the mobile network is reduced by more than 50% with respect to computing energy schedules solely based on the present network status [78], i.e., disregarding future energy arrivals and load conditions. As we elaborate in Section 4.2, we have not identified previous works coping with distributed BS deployments with energy harvesting, storage and transfer capabilities (via PPGs), and proposing an energy management solution based on statistical learning and predictive control. The main contributions of the present chapter are:

- We present an online statistical learning framework based on Gaussian Processes, which is customized to learn the Energy Harvesting (EH) and consumption (load) patterns over time. Specifically, a specific composite kernel is designed and tuned with optimal hyperparameters to cope with local quasi-periodic structures in the data, with noise operating at different scales. GPs are then utilized to predict these processes in future time slots, in an online and adaptive fashion (based on the most recent samples).

- We formulate an online and predictive optimization problem for the energy transfer across EH BSs with the goal of making the mobile network energetically self-sustainable.

- We provide numerical results, quantifying the effectiveness of the proposed solution with real-world harvested energy and load traces. An important finding is that the combination of forecasting and predictive control can substantially reduce the total amount of energy that the BS system drains from the power grid, halving it in most cases. This descends from a more intelligent redistribution of the harvested energy.

The chapter is organized as follows. In Section 4.2, we present the literature on energy cooperation, the mathematical tools used in this chapter and highlight the novel aspects of our design. The network scenario is described in Section 4.3. Our optimization framework is detailed in Section 4.4. The

[1]Computed as the ratio between the number of BSs that are unable to serve the users within range due to energy scarcity, and the total number of BSs.

numerical results are presented in section 4.5, and final remarks are provided in section 4.6.

4.2 Related Work

Energy transfer in mobile cellular networks: the concept of *energy transfer*, also referred to as *energy cooperation* [79–81] or *energy exchange* [82], is motivated by the fact that the distributed renewable energy generated at the BSs can be leveraged upon through a microgrid connecting them [83], with the aim of improving the network self-sustainability, while reducing the cost entailed in purchasing the energy from the power grid. Energy sharing among BSs is investigated in [80] through the analysis of several multiuser network structures. A two-dimensional and directional water-filling-based offline algorithm is proposed to control the harvested energy flows in time and space, with the objective of maximizing the system sum-rate throughput. In [81], the authors introduce a new entity called the *aggregator*, which mediates between the grid operator and a group of BSs to redistribute the energy flows, reusing the existing power grid infrastructure: one BS injects power into the aggregator and, simultaneously, another one draws power from it. This scheme does not consider the use of *energy storage devices*, and for this reason some of the harvested energy can be lost if none of the base stations drains it when it is injected. The authors of [84] consider BSs with energy harvesting capabilities connected to the power grid as a means to carry out the energy trading. A joint optimization tackling BS operation and power distribution is performed to minimize the on-grid power consumption of the BSs. Wired energy transfer to/from the power grid, and a user-BS association scheme based on cell zooming are investigated. The problem is solved using heuristics. A similar approach is considered in [85], where two problems are addressed: the first one consists of optimizing the energy allocation at individual BSs to accommodate for the temporal dynamics of harvested energy and mobile traffic. Considering the spatial diversity of mobile traffic patterns, the second problem amounts to balancing the energy consumption among BSs, by adapting the cell size (radio coverage) to reduce the on-grid energy consumption of the cellular network. Again, the solutions are obtained through heuristic algorithms. Also, in these

works, differently to what we propose here, *BSs do not perform any actual energy transfer among them.*

A two-cell renewable-energy-powered system is studied in [86], where the sum-rate over all users is maximized while determining the direction and amount of energy to be transferred between the two BSs. Energy can be transferred across the network either through power lines or wireless transfer and the energy transfer efficiency is taken into account. This resource allocation problem is formulated under a Frequency Division Multiple Access (FDMA) setup and is solved numerically. A low-complexity heuristic approach is also proposed as a practical near-optimal strategy when the transfer efficiency is sufficiently high and the channel gains are similar for all users. A similar two-BS scenario is considered in [79], where BSs gather energy from the power grid and from renewable energy sources, have a limited energy storage capability, and are connected through power lines. The authors study the case where renewable energy and energy demand profiles are deterministically known ahead of time, and find the optimal energy cooperation policy by solving a linear program. They then consider a more realistic case where the profiles are stochastic and propose an online greedy algorithm. Finally, an intermediate scenario is addressed, where the energy profiles are obtained from a deterministic pattern, adding a small amount of random noise at each time step.

The authors of [82] consider a setup similar to ours, i.e., multiple BSs, energy harvesting with local storage devices and energy exchange among BSs through the power grid. The main differences are that *perfect knowledge* of hourly varying energy demand profile (BS load) and hourly harvested energy is assumed, and energy routing is not studied. An optimal constrained problem is formulated, assessing its performance via simulation. Other relevant papers are [83,87]. There, energy sharing takes place either via physical power lines or through the power grid (virtual energy exchange). Interestingly, the authors investigate the impact of the power line infrastructure topology: agglomerative and divisive hierarchical clustering algorithms are utilized to determine it. Upon establishing the physical connections among BSs, an optimization framework for day-to-day cost optimization is developed for the cases of 1) zero knowledge, 2) perfect knowledge, and 3) partial future knowledge of the harvested energy process. The main differences with respect to our analysis are: for the partial future knowledge case, a static model is adopted, where

the amount of energy harvested through the day is given by an average value for each time slot, plus a random displacement. Average values are obtained from historical data, but *are kept fixed* during the day. In contrast to this, we devise an *online* estimation algorithm through which future harvested energy incomes are estimated based on those measured in the most recent time slots, providing *online adaptation and tracking* capabilities. Also, in [83] perfect knowledge of the BS consumption pattern across a whole day is assumed, whereas we estimate and track it at runtime. Online predictive control takes these estimates into account for the computation of optimal energy transfers, making our solution applicable to real settings.

Techniques exploiting energy trading / sharing through, e.g., spectrum sharing or Coordinated MultiPoint (CoMP), are combined for energy cost reduction in EH BS networks and a power grid in [88]. While the authors discuss interesting future research directions, their system model does not consider time dynamics. Following the idea of energy trading and CoMP, a framework focusing on beamforming designs for CoMP downlink communication systems is introduced in [89]. Formulated as a convex problem, the proposed scheme provides an offline ahead-of-time beamformer and energy schedule over a finite time horizon. Online solutions are left for future research. The joint energy purchase and wireless load sharing among mobile network operators is exploited in [90] to reduce energy costs. The authors of this chapter propose a scheme named *energy group buying with load sharing*, where the two operators are aggregated into a single group to implement a day-ahead and real-time energy purchase, and their BSs share the wireless traffic to maximally put lightly-loaded BSs into sleep mode. This scenario is tackled using two-stage stochastic programming. The scenario that we consider here is different, as we focus on actual energy exchange among BSs belonging to the same operator. Finally, the authors of [91] consider a delay minimization problem in an energy harvesting communication network with energy cooperation. Their study considers fixed data and energy routing topologies, determining optimum data rates, transmit powers, and energy transfers through an iterative algorithm, subject to flow and energy conservation constraints, to minimize the network delay.

On combining pattern learning with multi-step optimization tech-

niques: next, we briefly review the mathematical tools that we use in the present chapter, namely, MPC and GPs.

MPC has its roots in optimal control theory. The main idea is to use a dynamic model to forecast the system behavior, and exploit the forecast state sequence to obtain the *control* at the current time. The system usually evolves in slotted time, the control action is obtained by solving, at each time slot, a finite horizon optimal control problem where the initial state is the current state of the system. The optimization yields a finite control sequence, and the first control action in this sequence is applied [92]. MPC has the ability to anticipate future events and makes control decisions accordingly. It has been widely used in industrial processes, including chemical plants [93–95] and oil refineries [96,97] and, recently, to balance energy consumption in smart energy grids [98–100]. Moreover, it has been applied to supply chain management problems, with promising results [101–104].

It is known that using time-series forecasting within an MPC framework can improve the quality of the control actions by providing insight into the future [105]. Over the last decades, numerous forecasting approaches have been developed, including Autoregressive Integrated Moving Average (ARIMA) processes and Artificial Neural Networks (ANNs). ARIMA models (introduced by Box and Jenkins in [106]) are known for their prediction accuracy, but their main limitation lies in the assumption that the data follows a linear model. Conversely, ANNs capture non-linear models and, in turn, can be a good alternative to ARIMA [107]. Nonetheless, ANNs give rise to mixed results for purely linear correlation structures. In [108,109], hybrid schemes that combine them are put forward to take advantage of their unique strengths. Experimental results with real-world data indicate that their combined use can improve the prediction accuracy achieved by either of the techniques when used in isolation.

Several authors have proposed the use of non-linear models to build non-linear adaptive controllers. In most applications, however, these non-linearities are unknown, and non-linear parameterization must be used instead. In time-series analysis, where the underlying structure is largely unknown, one of the main challenges is to define an appropriate form of non-linear parameterization for the forecasting model. Some implementations claim to be non-parametric, such as GPs, which can be considered (in some sense) as equivalent to models

based on an infinite set of non-linear basis functions [110]. The basic idea of GPs is to place a *prior distribution* directly on the space of functions, without finding an appropriate form of non-linear parameterization for the forecasting model. This can be thought of as a generalization of a Gaussian distribution over functions. Moreover, a GP is completely specified by the mean function and the *covariance function* or *kernel*, which has a particular (but simple) parametric structure, defined through a small number of *hyperparameters*. The term non-parametric does not mean that there are no parameters, but that the parameters can be conveniently adapted from data. While GPs have been used in time-series forecasting [111], to the best of the authors' knowledge, [112] is the first application of GPs to electrical load forecasting [113–116].

The electricity supply is mainly influenced by meteorological conditions and daily seasonality. Nevertheless, forecasting for short-term horizons of about a day is often performed using univariate prediction models, which are considered to be sufficient because the weather tends to change in a smooth fashion, which is reflected in the electricity demand itself. Also, in a real-world online forecasting scenario, multivariate modeling is usually considered impractical [117]. Due to daily seasonality, we can say that the electrical load data bears some similarities with the time series that we consider in this chapter, i.e., the harvested energy profile of Section 4.3.2 and the traffic load of Section 4.3.3.

The idea of combining MPC and GPs was first proposed in [118], where the framework is evaluated by means of a simple (simulated) first order non-linear process. Other practical implementations can be found in application domains such as greenhouse temperature control systems [119], gas-liquid separation plant control systems [120], combustion power plants control systems [121] and in a number of other cases [122–125]. To the best of our knowledge, the present analysis is the first where MPC and GPs are combined to control an energy harvesting mobile network. Our purpose is thus to demonstrate the feasibility of application and realization of a GP based control algorithm for online power management, highlighting its potentials for the development of greener technologies, with the aim of improving the network self-sustainability.

Novelty of the present analysis: despite the existence of previous works on energy cooperation, here we consider a more complete setup, where: (i) EH BSs are equipped with storage capabilities, (ii) the harvesting process and

traffic load in the system are unknown and *fully stochastic*, (iii) the harvested energy and traffic load in BSs that we use for GP training and for our numerical results come from real-world traces, and (iv) the physical power grid is based on the novel concept of PPG. The combination of MPC and GPs has already been considered in the literature. However, to the best of our knowledge, this is the first time where this tool chain is used in an energy-aware mobile network scenario. Also, in the proposed optimization architecture, the overall problem is split into sub-blocks, where optimization problems are convex, can be solved at runtime and have low-complexity. This makes it possible to implement the presented solution in real systems.

4.3 System Model

We consider a mobile network comprising a set $\mathcal{S}$ of $n_s = |\mathcal{S}|$ BSs, each with energy harvesting capabilities, i.e., a solar panel, an energy conversion module and an energy storage device. Some of the BSs are ongrid (termed *ongrid* BSs, being part of set $\mathcal{S}_{\text{on}}$) and, in turn, can obtain energy from the power grid. The remaining BSs are *offgrid* (set $\mathcal{S}_{\text{off}}$). The proposed optimization process evolves in slotted time $t = 1, 2, \ldots$, where the slot duration corresponds to the time granularity of the control and can be changed without requiring any modifications to the following algorithms.

4.3.1 Power Packet Grids

A PPG is utilized to distribute energy among the BSs. The grid architecture is similar to that of a multi-hop network, see Fig. 4.1, where circles are BSs and the square is the energy router, which is in charge of energy routing decisions and power allocation. As assumed in [77], BSs are connected through Direct Current (DC) power links (electric wires) and the transmission of energy over them is operated in a TDM fashion. Energy transfer occurs by first establishing an *energy route*, which corresponds to a sequence of power links between the energy source and the destination. Each power link can only be used for a single transfer operation at a time. Power distribution losses along the power links follow a linear function of the distance between the source and the destination [77]. They depend on the resistance of the considered trans-

mission medium and are defined by [126]: $R = \rho\ell/A$, where ρ is the resistivity of the wire in $\Omega\text{mm}^2/\text{m}$, ℓ is the length of the power link in meters, and A is the cross-sectional area of the cable in mm^2. In this chapter, we consider a PPG with a single energy router in the center of the topology. A number of sub-trees originates from the router and, without loss of generality, each hop is assumed to have the same length ℓ, i.e., the same power loss.

4.3.2 Harvested Energy Profiles

Solar energy generation traces have been obtained using the SolarStat tool [127]. For the solar modules, the commercially available Panasonic N235B photovoltaic technology is considered. Each solar panel has 25 solar cells, leading to a panel area of $0.44\,\text{m}^2$, which is deemed practical for installation in a urban environment, e.g., on light-poles. As discussed in [73,127], the EH inflow is generally bell-shaped with a peak around mid-day, whereas the energy harvested during the night is negligible. Here, the framework in [127] is utilized to obtain the amount of energy harvested for each BS $n \in \{1, \ldots, n_s\}$ in time slot t, which is denoted by $H_n(t)$.

4.3.3 Traffic Load and Power Consumption

Traffic load traces have been obtained using real mobile data from the Big Data Challenge organized by Telecom Italia Mobile (TIM) [128]. The dataset is the result of a computation over Call Detail Records (CDRs), logging the user activity within the TIM cellular network for the city of Milan during the months of November and December 2013. For the traffic load traces we use the CDRs related to SMS, calls and Internet activities, performing spatial and temporal aggregation. In this way, we obtain a daily traffic load profile for each BS.

Clustering techniques have been applied to the dataset to understand the behavior of the mobile data. To this end, we use DBSCAN unsupervised clustering [129] to classify the load profiles into several categories. In Fig. 4.3, we show the typical traffic behavior of two clusters, corresponding to the heaviest (cluster 1) and lightest (cluster 2) daily load. As noted in previous works, the traffic is time-correlated (and daily periodic) [73, 130]. In our numerical results, each BS has an associated load profile, which is picked at random as

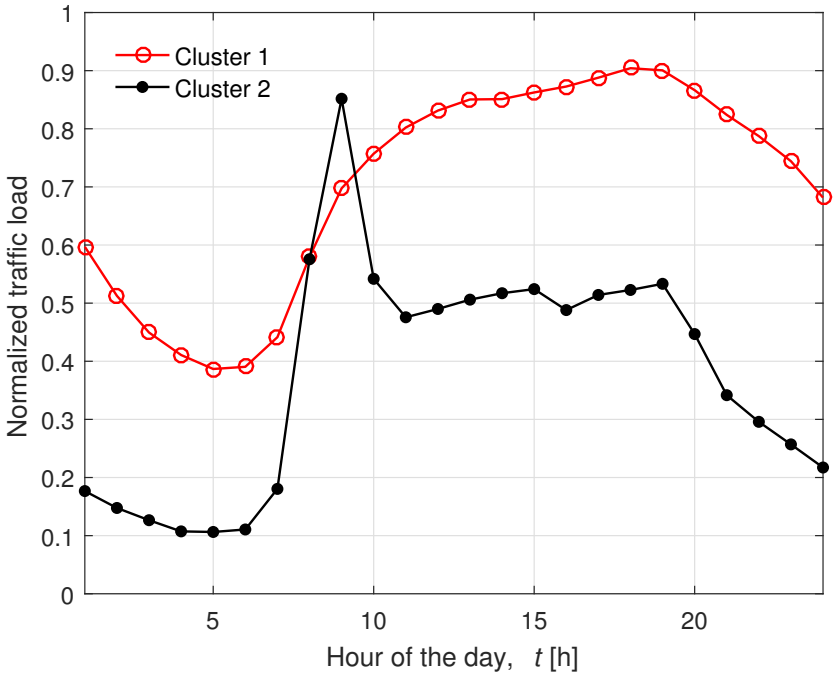


Figure 4.3: Load pattern profiles (two classes).

one of the two clusters in Fig. 4.3. Depending on the cluster association probabilities, there is some imbalance in the load distribution across BSs that, as we shall see, plays a key role in the performance of energy transfer algorithms. Given the traffic load profile $L_n(t)$, intended as the percentage of the total bandwidth that a BS n allocates to serve the users in its radio cell, the BS energy consumption (energy outflow), referred to in the following as $O_n(t)$, is computed through the linear model in [73] (see Eq. (1) in that paper).

4.3.4 Energy Storage Units

Energy storage units are interchangeably referred to as Energy Buffers (EBs). The EB level for BS $n \in \{1, \ldots, n_s\}$ is denoted by $B_n(t)$ and three thresholds are defined: B_{up}, B_{ref} and B_{low}, respectively termed the *upper*, *reference* and *lower* energy threshold, with $0 < B_{\text{low}} < B_{\text{ref}} < B_{\text{up}} < B_{\text{max}}$. B_{max} is the EB capacity, B_{ref} is the desired (reference) EB level and B_{low} is the lowest energy level that any BS should ever reach. B_{up} is used in the energy purchase process from the power grid, as detailed shortly below. For an *offgrid* BS, i.e., $n \in \mathcal{S}_{\text{off}}$, if t is the current time slot, $B_n(t)$ is the EB level at the beginning of

time slot t, which is updated at the beginning of the next time slot $t+1$ as:

$$B_n(t+1) = \xi(t)(B_n(t) + H_n(t) - O_n(t) + T_n(t))\,, \tag{4.1}$$

where $T_n(t)$ is the amount of energy transferred to/from BS n in time slot t, which is positive if BS n is a consumer or negative if BS n acts as an energy source. In fact, for a source we have $T_n(t) < 0$, as this models the outflow energy, i.e., the energy that the BS transfers, which is drained from its energy buffer, while for a consumer we use $T_n(t) > 0$, as this models inflow energy, i.e., the new energy that is injected into the buffer. $H_n(t)$, $O_n(t)$ are the amount of energy harvested and the energy that is locally drained (to support the local data traffic), respectively. Finally, $\xi(t)$ represents the losses in the EB due to charging and discharging. It depends upon the current state of charge of the EB, which is a realistic assumption. For example, using the model in [131], we have:

$$\xi(t) = 1 - \frac{(B_n(t) - B_{\max}/2)^2}{\beta_{\text{loss}}(B_{\max}/2)^2}\,, \tag{4.2}$$

where $\beta_{\text{loss}} > 1$ is a constant depending upon the technology in use. Note that, as β_{loss} increases, the storage losses decrease, whereas $\beta_{\text{loss}} \to \infty$ models an ideal battery.

The EB level of an *ongrid* BS $n \in \mathcal{S}_{\text{on}}$ is updated as:

$$B_n(t+1) = \xi(t)(B_n(t) + H_n(t) - O_n(t) + T_n(t) + \theta_n(t))\,, \tag{4.3}$$

where $\theta_n(t) \geqslant 0$ represents the energy purchased by BS n from the power grid during time slot t. The behavior of a BS (i.e., $T_n(t)$ and $\theta_n(t)$) depends on its EB level. If the BS behaves as an *energy source*, it is eligible for transferring a certain amount of energy $T_n(t)$ to other BSs. In this chapter, we assume that if the total energy in the buffer at the beginning of the current time slot t is $B_n(t) < B_{\text{up}}$ and the BS n is ongrid, then the difference $\theta_n(t) = B_{\text{up}} - B_n(t)$ is purchased from the power grid in slot t, as an ongrid BS should always be a source, i.e., in the position of transferring energy to other BSs. If instead the BS behaves as an *energy consumer*, it demands energy from the sources. For example, the energy demand in time slot t may be set to $B_{\text{ref}} - B_n(t)$, so that the EB level would ideally become no smaller than the

reference threshold B_{ref} by the end of the current time slot t. Note that, this can only be strictly guaranteed if $H_n(t) - O_n(t) \geqslant 0$. However, $B_n(t)$ is updated at the beginning of time slot t, whereas $H_n(t)$ and $O_n(t)$ are only known by the end of it. To cope with this, the theory of Sections 4.4.2 and 4.4.3 computes $T_n(t)$ accounting for the expected behavior $\mathbb{E}[H_n(t) - O_n(t)]$, where $\mathbb{E}[\cdot]$ is the expectation operator.

4.4 Optimization for online energy management

Next, we present an optimal online algorithm for power allocation and transfer, whose objective is to make the offgrid BSs as energy self-sustainable as possible.

4.4.1 Overview of the optimization framework

A diagram of the optimization process is shown in Fig. 4.2, involving 1) pattern learning (forecasting), 2) model predictive control, 3) energy allocation and 4) energy routing. These algorithms are executed at runtime.

1) Pattern learning (Section 4.4.2): the harvested energy and traffic load processes are statistically modeled through Bayesian non-parametric tools ("pattern learning" in Fig. 4.2). This allows each BS n to *independently* track its own energy ($H_n(t)$) and load ($L_n(t)$) processes, capturing their statistical behavior and obtaining multi-step ahead forecasts for the corresponding time series. Our forecasting method is agnostic to the type of signal, and for this reason can be extended to other processes, if need be. These forecasts are then fed into the MPC optimization approach of Section 4.4.3. Their use allows taking the future system evolution into account.

2) Model predictive control (Section 4.4.3): the goal of the MPC block is to determine the BS role (source or consumer) and obtain $T_n(t)$, for all BSs n and t. At any time t, if BS n gets $T_n(t) > 0$, then it behaves as a consumer and its energy demand is $d_n \stackrel{\Delta}{=} T_n(t)$, if instead $T_n(t) < 0$, then BS n behaves as a source and $|T_n(t)|$ represents the energy it offers to the other BSs. The MPC block considers the current system state, i.e., traffic load, harvested energy and EB levels, but also future ones (based on the forecasts from the pattern learning block). This is a main difference with respect to the

work in [78], where BS energy roles are solely determined based on the current system state at time t. Further, in our solution the forecasts of harvested energy and traffic load, and the control actions, are adapted in an online fashion during the day, according to the most recent samples. This is in contrast to [87] and [83], where the harvested energy forecasts are estimated one day ahead and the load profile is known.

3) Energy allocation (Section 4.4.4): once $T_n(t)$ is obtained for each BS n, in a third optimization step we compute how the energy $T_n(t)$ from the sources is to be split among the consumers in time slot t. To understand this, let us indicate the set of sources and consumers by $\mathcal{Y}_s$ and $\mathcal{Y}_c$, respectively. We have that $i \in \mathcal{Y}_s$ if $T_i(t) < 0$ and $j \in \mathcal{Y}_c$ if $T_j(t) > 0$. If BS i is a source, $|T_i(t)|$ can be split as $|T_i(t)| = \sum_{j \in \mathcal{Y}_c} T_i^j(t)$, where $T_i^j(t)$ is the share of $|T_i(t)|$ that source i sends to consumer j. The objective of this optimization block is to find these shares (energy allocations) for each source. To this end, two approaches are proposed, one based on convex optimization and another one formulated as an assignment problem. Their objective is to maximize the number of consumers that receive energy, while maximizing the energy transfer efficiency (accounting for the number of links that separate sources and consumers).

4) Energy routing (Section 4.4.5): for every time t, once the energy allocations $T_i^j(t)$ are assessed, the last step is to find a feasible schedule that implements the required energy transfers (energy routing) from sources to consumers. Since the PPG is operated in a TDM fashion, each power link can only be used for a single energy transfer operation at a time. Thus, the proposed routing strategy seeks to find disjoint routes between energy sources and consumers, while minimizing the time needed to perform the energy transfer.

The list of symbols that we use in the mathematical framework is provided in Table 4.1, Table 4.2, and Table 4.3.

4.4.2 Pattern learning

In this section, we present statistical models to automatically capture the hidden structure in harvested energy and load processes. GPs have become popular for regression and classification, often showing impressive performance [132]. Hereinafter, we will focus on GPs for regression, according to the function-space view applied to the Bayesian linear model [132]. The

Table 4.1: List of symbols used in the chapter.

Definition	**Variable name**
Base station set	$\mathcal{S}$
Ongrid base station set	$\mathcal{S}_{\text{on}}$
Offgrid base station set	$\mathcal{S}_{\text{off}}$
Number of base stations	$n_s = \lvert\mathcal{S}\rvert$
Traffic load profile in slot t	$L(t)$
Harvested energy profile in slot t	$H(t)$
BS energy consumption in slot t	$O(t)$
Effective energy income $W(t)$	$H(t) - O(t)$
Energy buffer level in slot t	$B(t)$
Maximum energy buffer capacity	$B_{\max}$
Upper, lower and reference buffer thresholds	B_{up}, B_{low}, B_{ref}
Transferred energy in slot t	$T(t)$
Purchased grid energy in slot t	$\theta(t)$

Bayesian linear model for regression is defined as:

$$f(\boldsymbol{x}) = \phi(\boldsymbol{x})^\top \boldsymbol{w}, \quad r = f(\boldsymbol{x}) + \epsilon, \tag{4.4}$$

where $\boldsymbol{w}$ is a vector of weights, also known as model parameters, $f(\boldsymbol{x})$ is the function value, which is linear in the weights $\boldsymbol{w}$, r is the observed real value, and $\phi(\cdot) : \mathbb{R}^D \to \mathbb{R}^F$ maps the D-dimensional input column vector $\boldsymbol{x}$ into an F-dimensional feature vector $\phi(\boldsymbol{x}) = \boldsymbol{\phi}$. Assume we are given with a training dataset with N observations, $\mathcal{D} = \{(\boldsymbol{x}_i, r_i)\}_{i=1}^N$, where each pair $(\boldsymbol{x}_i, r_i)$ consists of the D-dimensional input column vector $\boldsymbol{x}_i$ and the scalar target r_i. We can aggregate inputs and targets in a $D \times N$ matrix $\boldsymbol{X}$ and an N-dimensional column vector $\boldsymbol{r}$, so that $\mathcal{D} = (\boldsymbol{X}, \boldsymbol{r})$, and $\phi(\boldsymbol{X}) = \boldsymbol{\Phi}$ becomes an $F \times N$ matrix in the feature space. We are interested in the conditional distribution of the targets, given the inputs in the feature space and the model parameters. We further assume that r differs from $f(\boldsymbol{x})$ by additive noise, which follows an independent identically distributed (i.i.d.) Gaussian distribution with zero mean and variance σ_n^2, i.e., $\epsilon \sim \mathcal{N}(0, \sigma_n^2)$. From the i.i.d. assumption, it follows that the *likelihood* (i.e., the conditional distribution of the targets given the

Table 4.2: List of symbols used by the pattern analysis block.

Definition	**Variable name**
Number of observations (training dataset)	N
Number of observations (test set)	N_*
The transpose of vector $\boldsymbol{x}$	$\boldsymbol{x}^\top$
The weights of the Bayesian linear model	$\boldsymbol{w}$
The function value $f(\boldsymbol{x}) = \phi(\boldsymbol{x})^\top \boldsymbol{w}$	$f(\boldsymbol{x})$
The observed real value $r = f(\boldsymbol{x}) + \epsilon$	r
N-dimensional column vector of targets	$\boldsymbol{r}$
Map in the feature space	$\phi(\cdot) : \mathbb{R}^D \rightarrow \mathbb{R}^F$
Training dataset $\mathcal{D} = \{(\boldsymbol{x}_i, r_i)\}_{i=1}^N$	$\mathcal{D}$
D-dimensional input column vector	$\boldsymbol{x}$
D-dimensional input column vector (test set)	$\boldsymbol{x}_*$
F-dimensional feature vector	$\phi(\boldsymbol{x}) = \boldsymbol{\phi}$
$D \times N$ matrix of inputs	$\boldsymbol{X}$
$D \times N_*$ matrix of inputs in the test set	$\boldsymbol{X}_*$
$F \times N$ matrix in the feature space	$\phi(\boldsymbol{X}) = \boldsymbol{\Phi}$
Gaussian dist. with zero mean and variance σ_n^2	$\epsilon \sim \mathcal{N}(0, \sigma_n^2)$
Covariance matrix of the model parameters $\boldsymbol{w}$	$\boldsymbol{\Sigma}_w$
Gaussian process	$\mathcal{GP}(m(\boldsymbol{x}), k(\boldsymbol{x}, \widehat{\boldsymbol{x}}))$
Gaussian process: mean function	$m(\boldsymbol{x})$
Gaussian process: *covariance function (kernel)*	$k(\boldsymbol{x}, \widehat{\boldsymbol{x}})$
Gaussian process: predictive mean vector	$\boldsymbol{\mu}$
Gaussian process: predictive covariance matrix	$\boldsymbol{\Sigma}$
$N \times N$ covariance matrix (training dataset)	$\boldsymbol{K}$
$N \times N$ identity matrix	$\boldsymbol{I}_N$
Function values (training dataset)	$\boldsymbol{f}$
Function values (test set)	$\boldsymbol{f}_*$

inputs in the feature space and the model parameters) is factorized over cases for the N observations, i.e., $\boldsymbol{r}|\boldsymbol{X}, \boldsymbol{w} \sim \mathcal{N}(\boldsymbol{\Phi}^\top \boldsymbol{w}, \sigma_n^2 \boldsymbol{I}_N)$.

We can perform regression in the function-space view by using a GP, modeling a distribution over functions. Formally: a GP is a collection of random variables, any finite number of which have a joint Gaussian distribution.

Table 4.3: List of symbols used by the optimization block.

Definition	**Variable name**
MPC optimization horizon (time steps)	M
Weight parameter for MPC	α
Set of energy sources in slot t	$\mathcal{Y}_s$
Set of energy consumers in slot t	$\mathcal{Y}_c$
Energy availability matrix in slot t	$\boldsymbol{E}$
Energy allocation matrix in slot t	$\boldsymbol{Y}$
Energy demand vector in slot t	$\boldsymbol{d}$
Maximum transmission energy capacity	$e_{\max}$
Matrix of number of hops in the electrical grid	$\boldsymbol{G}$
Weight parameter for energy allocation	β
Cost matrix for Hungarian method	$\boldsymbol{C}$

Moreover, it is completely specified by the mean function and the *covariance function* (or *kernel*). We define the mean function and the *covariance function* of process $f(\cdot) \sim \mathcal{GP}(m(\boldsymbol{x}), k(\boldsymbol{x}, \widehat{\boldsymbol{x}}))$ as

$$\begin{aligned} m(\boldsymbol{x}) &= \mathbb{E}[f(\boldsymbol{x})] \\ k(\boldsymbol{x}, \widehat{\boldsymbol{x}}) &= \mathbb{E}[(f(\boldsymbol{x}) - m(\boldsymbol{x}))(f(\widehat{\boldsymbol{x}}) - m(\widehat{\boldsymbol{x}}))^\top]. \end{aligned} \tag{4.5}$$

Next, we consider the zero mean function, i.e., $m(\boldsymbol{x}) = 0$, which is a very typical choice [132]. In the Bayesian linear model of Eq. (4.4), the *prior* distribution is set to be Gaussian with zero mean and covariance matrix $\boldsymbol{\Sigma}_w$, i.e., $\boldsymbol{w} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{\Sigma}_w)$. Thus, we can derive an example GP as:

$$\begin{aligned} m(\boldsymbol{x}) &= \phi(\boldsymbol{x})^\top \mathbb{E}[\boldsymbol{w}] = \boldsymbol{0} \\ k(\boldsymbol{x}, \widehat{\boldsymbol{x}}) &= \phi(\boldsymbol{x})^\top \mathbb{E}[\boldsymbol{w}\boldsymbol{w}^\top] \phi(\widehat{\boldsymbol{x}}) = \phi(\boldsymbol{x})^\top \boldsymbol{\Sigma}_w \phi(\widehat{\boldsymbol{x}}). \end{aligned} \tag{4.6}$$

Assume the training dataset has N observations, then vector $\boldsymbol{f} = [f(\boldsymbol{x}_1), \ldots, f(\boldsymbol{x}_N)]^\top$ has a joint Gaussian distribution, i.e., $\boldsymbol{f}|\boldsymbol{X} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{K})$, where the $N \times N$ covariance matrix $\boldsymbol{K}$ can be computed evaluating the *covariance function* or *kernel* for the N observations, i.e., $\boldsymbol{K}_{ij} = \phi(\boldsymbol{x}_i)^\top \boldsymbol{\Sigma}_w \phi(\boldsymbol{x}_j)$ for $i, j = 1, \ldots, N$. Given the noise $\epsilon \sim \mathcal{N}(0, \sigma_n^2)$, it follows from the i.i.d. assumption that a diagonal matrix $\sigma_n^2 \boldsymbol{I}_N$ must be added to $\boldsymbol{K}$, as compared to the noise-free

model in the GP literature [132]. To make prediction for the test case $f(\boldsymbol{x}_*) = f_*$ given $\phi(\boldsymbol{x}_*) = \boldsymbol{\phi}_*$, we consider the joint Gaussian *prior* distribution over functions

$$\begin{bmatrix} \boldsymbol{r} \\ f_* \end{bmatrix} = \mathcal{N}\left(\boldsymbol{0}, \begin{bmatrix} \boldsymbol{K} + \sigma_n^2 \boldsymbol{I}_N & \boldsymbol{k}_* \\ \boldsymbol{k}_*^\top & k(\boldsymbol{x}_*, \boldsymbol{x}_*) \end{bmatrix}\right), \tag{4.7}$$

where we define the N-dimensional column vector $\boldsymbol{k}_*$ such that the i-th element equals $\phi(\boldsymbol{x}_i)^\top \boldsymbol{\Sigma}_w \phi(\boldsymbol{x}_*)$. To derive the *posterior* distribution over functions we need to condition the joint Gaussian *prior* distribution over functions on the data, so that we get the key predictive equations of GPs for regression:

$$\begin{aligned} &f_* | \boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{r} \sim \mathcal{N}(\mu, \Sigma) \\ &\mu = \boldsymbol{k}_*^\top [\boldsymbol{K} + \sigma_n^2 \boldsymbol{I}_N]^{-1} \boldsymbol{r} \\ &\Sigma = k(\boldsymbol{x}_*, \boldsymbol{x}_*) - \boldsymbol{k}_*^\top [\boldsymbol{K} + \sigma_n^2 \boldsymbol{I}_N]^{-1} \boldsymbol{k}_*. \end{aligned} \tag{4.8}$$

In practice, the predictive mean μ is used as a point estimate for the function output, while the variance Σ can be translated into uncertainty bounds (predictive error-bars) on this point estimate, thus making Gaussian Process (GP) for regression very appealing for MPC applications (see [118, 133–135]).

For any set of basis functions in the feature space, we can compute the corresponding *covariance function* or *kernel*; conversely, for every (positive definite) *covariance function* or *kernel*, there exists a (possibly infinite) expansion in terms of basis functions in the feature space. As we show shortly, the choice of the kernel deeply affects the performance of a GP for a given task, as much as the choice of the parameters (architecture, activation functions, learning rate, etc.) does for a neural network. Specifically, the *hyperparameters* of the kernel must be set in order to optimize the *marginal likelihood*,

$$p(\boldsymbol{r}|\boldsymbol{X}) = \int p(\boldsymbol{r}|\boldsymbol{f}, \boldsymbol{X}) p(\boldsymbol{f}|\boldsymbol{X}) d\boldsymbol{f}. \tag{4.9}$$

Under the Gaussian assumption, the *prior* distribution is Gaussian, $\boldsymbol{f}|\boldsymbol{X} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{K})$, and the *likelihood* is a factorized Gaussian, $\boldsymbol{r}|\boldsymbol{f}, \boldsymbol{X} \sim \mathcal{N}(\boldsymbol{f}, \sigma_n^2 \boldsymbol{I}_N)$, thus $\boldsymbol{r}|\boldsymbol{X} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{K} + \sigma_n^2 \boldsymbol{I}_N)$. Extensive derivation for the formulation of $f_*|\boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{r}$ and generalization to more that one test case can be found in [132].

Suppose we have N_* observations in the test set, i.e., $(\boldsymbol{X}_*, \boldsymbol{r}_*)$, to make

prediction for the test cases $f(\boldsymbol{X}_*) = \boldsymbol{f}_*$ given $\phi(\boldsymbol{X}_*) = \boldsymbol{\Phi}_*$, we consider the joint Gaussian *prior* distribution over functions

$$\begin{bmatrix} \boldsymbol{r} \\ \boldsymbol{f}_* \end{bmatrix} = \mathcal{N}\Bigg(\boldsymbol{0}, \begin{bmatrix} \boldsymbol{K} + \sigma_n^2 \boldsymbol{I}_N & \boldsymbol{K}_* \\ \boldsymbol{K}_*^\top & \boldsymbol{K}_{**} \end{bmatrix}\Bigg), \tag{4.10}$$

where we define the $N \times N_*$ matrix $\boldsymbol{K}_*$ similarly to $\boldsymbol{k}_*$, such that $\boldsymbol{K}_{*,ij} = \phi(\boldsymbol{x}_i)^\top \boldsymbol{\Sigma}_w \phi(\boldsymbol{x}_{*,j})$ for $i = 1, \ldots, N$, $j = 1, \ldots, N_*$, and $\boldsymbol{x}_{*,j}$ is a column vector in $\boldsymbol{X}_*$. Finally, we define the $N_* \times N_*$ matrix $\boldsymbol{K}_{**}$ similarly to $k(\boldsymbol{x}_*, \boldsymbol{x}_*)$, such that $\boldsymbol{K}_{**,ij} = \phi(\boldsymbol{x}_{*,i})^\top \boldsymbol{\Sigma}_w \phi(\boldsymbol{x}_{*,j})$ for $i, j = 1, \ldots, N_*$, thus we get the key predictive equations of GPs for regression:

$$\begin{aligned} &\boldsymbol{f}_* | \boldsymbol{X}_*, \boldsymbol{X}, \boldsymbol{r} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}) \\ &\boldsymbol{\mu} = \boldsymbol{K}_*^\top [\boldsymbol{K} + \sigma_n^2 \boldsymbol{I}_I]^{-1} \boldsymbol{r} \\ &\boldsymbol{\Sigma} = \boldsymbol{K}_{**} - \boldsymbol{K}_*^\top [\boldsymbol{K} + \sigma_n^2 \boldsymbol{I}_I]^{-1} \boldsymbol{K}_*. \end{aligned} \tag{4.11}$$

The choice of the kernel: this choice deeply affects the performance of a GP for a given task, as it encodes the similarity between pairs of outputs in the function domain. There has been significant work on constructing base and composite kernels [136]. Common base kernels include the Squared Exponential (SE) kernel, the Rational Quadratic (RQ) kernel, and the Standard Periodic (SP) kernel, defined as:

$$\begin{aligned} k_{\text{SE}}(\boldsymbol{x}, \widehat{\boldsymbol{x}}) &= \sigma_{\text{SE}}^2 \exp(-||\boldsymbol{x} - \widehat{\boldsymbol{x}}||^2 / (2\ell_{\text{SE}}^2)) \\ k_{\text{RQ}}(\boldsymbol{x}, \widehat{\boldsymbol{x}}) &= \sigma_{\text{RQ}}^2 (1 + ||\boldsymbol{x} - \widehat{\boldsymbol{x}}||^2 / (2\alpha_{\text{RQ}} \ell_{\text{RQ}}^2))^{-\alpha_{\text{RQ}}} \\ k_{\text{SP}}(\boldsymbol{x}, \widehat{\boldsymbol{x}}) &= \sigma_{\text{SP}}^2 \exp(-2 \sin^2(\pi ||\boldsymbol{x} - \widehat{\boldsymbol{x}}|| \mathrm{p}_{\text{SP}}) / \ell_{\text{SP}}^2). \end{aligned} \tag{4.12}$$

The properties of the functions under a GP with a SE kernel can display long range trends, where the length-scale ℓ_{SE} determines how quickly a process varies with the inputs. The RQ kernel is derived as a scale mixture of SE kernels with different length-scales. The SP kernel is derived by mapping the two dimensional variable $(\cos(\boldsymbol{x}); \sin(\boldsymbol{x}))$ through the SE kernel. Derivations for the RQ and SP kernels are in [132].

Note that valid kernels (i.e., those having a positive-definite covariance function) are closed under the operators + and ×. This allows one to create more representative (and composite) kernels from well-understood basic

components, according to the following key rules [136]:

- Any subexpression[2] $\mathcal{P}$ can be replaced with $\mathcal{P} + \mathcal{B}$, where $\mathcal{B}$ is any base kernel family.
- Any subexpression $\mathcal{P}$ can be replaced with $\mathcal{P} \times \mathcal{B}$, where $\mathcal{B}$ is any base kernel family.
- Any base kernel $\mathcal{B}$ can be replaced with any other base kernel family $\mathcal{B}'$.

In time series, summing kernels can express superpositions of different processes, operating at different scales, whereas multiplying kernels may be a way of converting global data properties onto local data properties. From here on, we will use one-dimensional kernels in the form RQ $\times$ SP with period $\mathrm{p_{SP}}$, which correspond to a local quasi-periodic structure in the data, with noise operating at different scales. Note that kernels over multidimensional inputs can be constructed via the operators + and $\times$ over individual dimensions. Next, we consider models based on zero-mean GPs for the runtime multi-step ahead forecasting of time series, with application to a) Harvested Energy Profile $H_n(t)$ (defined in Section 4.3.2) and b) Traffic Load $L_n(t)$ (Section 4.3.3).

The basic routine for prediction: we use models based on zero-mean GPs for the runtime forecasting of time series, with application to $H_n(t)$ and $L_n(t)$, $n \in \{1, \ldots, n_s\}$, $t = 1, \ldots, T$. The strong daily seasonality of the data is evident for both time series, as well as the presence of noise at different scales. Therefore, we define composite kernels for $H_n(t)$ and $L_n(t)$ in the form RQ$\times$SP with period $\mathrm{p_{SP}}$, i.e.,

$$k(x, \widehat{x}) = \sigma^2 \exp(-2\sin^2(\pi d \mathrm{p_{SP}})/\ell_{\mathrm{SP}}^2)(1 + d^2/(2\alpha_{\mathrm{RQ}}\ell_{\mathrm{RQ}}^2))^{-\alpha_{\mathrm{RQ}}} \tag{4.13}$$

where $\sigma = \sigma_{\mathrm{RQ}}\sigma_{\mathrm{SP}}$ and $d = |x - \widehat{x}|$ is the Euclidean distance between inputs. At this point, the *hyperparameters* of the kernel must be set in order to optimize the *marginal likelihood*, which is defined in Eq. (4.9), and here implemented using the toolbox of [137]. For compactness, we aggregate the *hyperparameters* of the kernel in the initialization set $\boldsymbol{\theta}^{(s)} = \{\sigma, \mathrm{p_{SP}}, \ell_{\mathrm{SP}}, \alpha_{\mathrm{RQ}}, \ell_{\mathrm{RQ}}\}$. Here, we opt for $\sigma = 1$, $\mathrm{p_{SP}} = 24$, and select the free parameters (ℓ_{SP}, α_{RQ}, ℓ_{RQ}) via

[2]Subexpression refers to any valid kernel family, either basic or composite.

a grid search, scanning combinations in the range $[10^{-2}, 10^{2}]$. To model the strong daily seasonality in the data, we also opt for a prior distribution on the period $\mathrm{p_{SP}}$, which is a delta function, i.e., $\delta(\mathrm{p_{SP}} - 24) = 1$ if and only if $\mathrm{p_{SP}} = 24$, so that we treat the period $\mathrm{p_{SP}}$ as a constant, excluding it from the optimization (see [137]).[3]

Algorithm 2 Pseudo-code for the basic routine

1: Pre-training phase: find the optimal *hyperparameters* $\boldsymbol{\theta}^{(0)}$ for the kernel $k(\cdot,\cdot)$, starting from $\boldsymbol{\theta}^{(s)}$ and minimizing the *marginal likelihood* on the training dataset $\{(\boldsymbol{x}_i, r_i)\}_{i=1}^{N}$
2: Set $t = 1$
3: **while** $t \leqslant T - (N + N_*)$ **do**
4: Set $\mathcal{D}^{(t)} = (\boldsymbol{X}^{(t)}, \boldsymbol{r}^{(t)}) = \{(\boldsymbol{x}_i, r_i)\}_{i=t-1+1}^{t-1+N}$
5: Set $\mathcal{D}_*^{(t)} = (\boldsymbol{X}_*^{(t)}, \boldsymbol{r}_*^{(t)}) = \{(\boldsymbol{x}_i, r_i)\}_{i=t-1+N+1}^{t-1+N+N_*}$
6: **if** $(t-1 \mod S) = 0$ **then** % $t-1$ is a multiple of S
7: Training phase: find the optimal *hyperparameters* $\boldsymbol{\theta}^{(t)}$ for the kernel $k(\cdot,\cdot)$, starting from $\boldsymbol{\theta}^{(0)}$ and minimizing the *marginal likelihood* on the training dataset $(\boldsymbol{X}^{(t)}, \boldsymbol{r}^{(t)})$
8: **end if**
9: Forecasting phase: get $(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ with test set $(\boldsymbol{X}_*^{(t)}, \boldsymbol{r}_*^{(t)})$ and using the key predictive equations of GPs in Eq. (4.11)
10: Compute $\mathrm{RMSE}_*^{(t)} = \sqrt{(\sum_{i=1}^{N_*} e_i^2)/N_*},\ \boldsymbol{e} = \boldsymbol{r}_*^{(t)} - \boldsymbol{\mu}$
11: Set $t = t + 1$
12: **end while**

Algorithm 2 describes the basic routine for the pre-training phase (line 1), training phase (line 7), and forecasting phase (line 9) for both zero-mean GPs, i.e., the same basic reasoning holds for $H_n(t)$ and $L_n(t)$, where $\boldsymbol{x}_t$ contains the time indices (in either the training or test dataset) and r_t refers to either process $H_n(t)$ or $L_n(t)$, at time t and BS n. Also, we assume that we can access the N values in the training dataset, and we wish to predict the N_* values in the test set, where $\mathcal{D}^{(t)} = (\boldsymbol{X}^{(t)}, \boldsymbol{r}^{(t)})$ refers to the training dataset and $\mathcal{D}_*^{(t)} = (\boldsymbol{X}_*^{(t)}, \boldsymbol{r}_*^{(t)})$ refers to the test set, at time t, respectively. According to the pre-training phase, we first have to find the optimal *hyperparameters* $\boldsymbol{\theta}^{(0)}$ for the kernel $k(\cdot,\cdot)$, starting from $\boldsymbol{\theta}^{(s)}$ and minimizing the *marginal likelihood* on the training dataset $\{(\boldsymbol{x}_i, r_i)\}_{i=1}^{N}$. Note that $\boldsymbol{\theta}^{(0)}$ will serve as initialization

[3] In our numerical results, we have considered a time step duration of one hour, so setting $\mathrm{p_{SP}} = 24$ entails a periodicity of one day.

for the optimal *hyperparameters* $\boldsymbol{\theta}^{(t)}$ at each step of the online forecasting routine, as the optimal *hyperparameters* $\boldsymbol{\theta}^{(t)}$ are found over the training dataset $(\boldsymbol{X}^{(t)}, \boldsymbol{r}^{(t)})$, which changes at each step of the online forecasting routine. Assuming Gaussian noise with variance σ_n^2, thus Gaussian likelihood, it follows that we can perform exact inference. To do it, we use the Conjugate Gradients (CG) optimization tool implemented in toolbox [137]. We get $(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ via Eq. (4.11) given the test set $(\boldsymbol{X}_*^{(t)}, \boldsymbol{r}_*^{(t)})$ with N_* test cases, at time t. Finally, we derive the Root Mean Square Error (RMSE) $\text{RMSE}_*^{(t)}$ over the N_* test cases, starting from residuals $\boldsymbol{e}$, at time t, and iterating the procedure (except for the pre-training phase) up to time $T-(N+N_*)$. For the numerical results, the training phase (line 7) is performed once every S steps: in Algorithm 2, we write $(t-1 \mod S) = 0$, i.e., $t-1$ is a multiple of S. Thus, the training phase (line 7) is performed when $t = 1$.

4.4.3 Model predictive control

System dynamics: the system to be controlled is described by means of a discrete-time model:

$$\boldsymbol{B}(t+1) = \boldsymbol{B}(t) + \boldsymbol{T}(t) + \boldsymbol{W}(t), \tag{4.14}$$

where t is the current time slot. The $M \times n_s$ matrix $\boldsymbol{B}(t)$ with elements $[\boldsymbol{B}(t)]_{k,n} = B_n(k)$ denotes the *system state*, representing for each BS $n \in \mathcal{S}$ the energy buffer level for time slot k, with $k = t, t+1, \ldots, t+M-1$, were M is the *optimization horizon*. Note that the system state in the first time slot t is known, whereas those in the following $M-1$ time slots have to be estimated. Referring to Section 4.4.2, we thus have $M = N_* + 1$. The $M \times n_s$ matrix $\boldsymbol{T}(t)$ with elements $[\boldsymbol{T}(t)]_{k,n} = T_n(k)$ denotes the *control* matrix, representing the amount of energy that each BS n shall either transfer (if $T_n(k) < 0$) or receive ($T_n(k) > 0$) in time slot $k = t, \ldots, t+M-1$. The $M \times n_s$ matrix $\boldsymbol{W}(t)$ with elements $[\boldsymbol{W}(t)]_{k,n} = H_n(k) - O_n(k)$ models the effective energy income, i.e., the stochastic behavior of the forecast profiles (harvested and consumed energy), with:

$$\boldsymbol{W}(t) \sim \mathcal{N}(\overline{\boldsymbol{W}}(t), \Sigma_{\boldsymbol{W}(t)}), \tag{4.15}$$

where $\overline{\boldsymbol{W}}(t)$ and $\Sigma_{\boldsymbol{W}(t)}$ contain the mean and variance of the forecast estimates, respectively. Note that processes $H_n(k)$ and $O_n(k)$ are statistically characterized through the prediction framework of Section 4.4.2, and their difference is still a Gaussian r.v. (in fact, $O_n(k)$ is derived from $L_n(k)$ through a linear model, and as such is still Gaussian distributed). Following [138], due to the stochastic nature of Eq. (4.15), the system state $\boldsymbol{B}(t)$ can also be written in a probabilistic way:

$$\boldsymbol{B}(t) \sim \mathcal{N}(\bar{\boldsymbol{B}}(t), \Sigma_{\boldsymbol{B}(t)}), \tag{4.16}$$

where $\bar{\boldsymbol{B}}(t)$ and $\Sigma_{\boldsymbol{B}(t)}$ are the mean and the variance of $\boldsymbol{B}(t)$, respectively.

Objective functions: the goal of the MPC controller is to determine the amount $T_n(k)$ that each BS n should either transfer or receive in time slots $k = t, \ldots, t + M - 1$, so that all the energy buffers remain as close as possible to the reference value B_{ref}. A first quadratic cost function tracks the total amount of energy that is to be exchanged among BSs in the optimization horizon $k = t, \ldots, t + M - 1$:

$$f_1^{\text{MPC}}(\boldsymbol{T}(t)) = \sum_{k=t}^{t+M-1} \sum_{n=1}^{n_s} T_n(k)^2. \tag{4.17}$$

$f_1^{\text{MPC}}(\cdot)$ is used to minimize the total amount of energy exchanged, so as to keep the energy losses low during the subsequent energy transfer phase. Through a second objective function, the MPC controller seeks to equalize the BS energy buffer levels as close as possible to the reference threshold B_{ref} (defined in Section 4.3.4). To achieve this, a second cost function is defined as follows:

$$f_2^{\text{MPC}}(\boldsymbol{B}(t), B_{\text{ref}}) = \sum_{k=t}^{t+M-1} \sum_{n=1}^{n_s} (B_n(k) - B_{\text{ref}})^2. \tag{4.18}$$

Control problem: the following finite-horizon multi-objective optimization problem is formulated:

$$\min_{\boldsymbol{T}(t)} \quad \mathbb{E}\left[\alpha f_1^{\mathrm{MPC}}(\boldsymbol{T}(t)) + (1-\alpha) f_2^{\mathrm{MPC}}(\boldsymbol{B}(t), B_{\mathrm{ref}})\right] \tag{4.19a}$$

$$\text{subject to:} \quad \boldsymbol{B}(t) \sim \mathcal{N}(\bar{\boldsymbol{B}}(t), \Sigma_{\boldsymbol{B}(t)}), \tag{4.19b}$$

$$\boldsymbol{W}(t) \sim \mathcal{N}(\overline{\boldsymbol{W}}(t), \Sigma_{\boldsymbol{W}(t)}), \tag{4.19c}$$

$$B_{\mathrm{low}} \leqslant B_n(k) \leqslant B_{\mathrm{max}}, \tag{4.19d}$$

$$T_n(k)_{\mathrm{min}} \leqslant T_n(k) \leqslant T_n(k)_{\mathrm{max}}, \tag{4.19e}$$

$$\text{with: } k = t, t+1, \ldots, t+M-1$$

where $\alpha \in [0,1]$ is a weight to balance the relative importance of the two cost functions. B_{low} and B_{max} are the energy buffer limitations defined in Section 4.3.4. Finally, constraint Eq. (4.19e) defines the amount of energy that each BS $n \in \mathcal{S}$ can exchange in slot k and depends on the system state, i.e., the energy buffer level $B_n(k)$, the expected harvested energy and expected traffic load: the system state defines the limits of the control action for each k.

For any fixed value of α, and since the optimization problem must be solved at runtime, it is strongly preferable to choose a convex optimization formulation such as Eq. (4.19), which can be solved through standard techniques. Here, we have used the `CVX` tool [139] to obtain the optimal solution $\boldsymbol{T}(t)^* = [T_n(k)^*]$, which represents the amount of energy that BS $i \in \mathcal{S}$ shall either offer or demand in time slot $k = t, \ldots, t+M-1$.

Optimization algorithm: the MPC controller performs as follows [140]:

1. **Step 1:** at the beginning of time slot k, the system state is obtained, that is energy buffer levels for all BSs, the harvested energy and traffic load forecasts for the next M time slots (the optimization horizon).

2. **Step 2:** the control problem in Eq. (4.19) is solved yielding a sequence of control actions over the horizon M.

3. **Step 3:** only the first control action is performed and the system state is updated upon implementing the required energy transfers.

4. **Step 4:** Forecasts are updated and the optimization cycle is repeated

from **Step 1**.

4.4.4 Energy allocation

Solving Eq. (4.19), we obtain $T_n(t)$ for each BS n in any given slot t. In this section, we solve the energy allocation problem, i.e., we compute for each source n, how to split $T_n(t)$ among the consumers. Note that this also depends on the distribution losses between sources and consumers and, in turn, on the electrical PPG topology.

Notation: at time t, we use indices i and j to respectively denote an arbitrary BS source (set $\mathcal{Y}_s$) and an arbitrary BS consumer (set $\mathcal{Y}_c$). g_{ij} is the number of hops in the PPG topology between source $i \in \mathcal{Y}_s$ and consumer $j \in \mathcal{Y}_c$, in matrix notation $\boldsymbol{G} = [g_{ij}]$. We assume that all hops have the same physical length and $a(g_{ij}) \in [0, 1]$ is the attenuation coefficient between i and j, due to the power loss (depending on the number of hops, i.e., on the physical distance that the energy has to travel, see Section 4.3.1). Let i be a source, the maximum amount of energy that a consumer j can receive from i is defined as $e_{ij} \triangleq |T_i(t)|a(g_{ij})$, $i \in \mathcal{Y}_s, j \in \mathcal{Y}_c$. In matrix notation $\boldsymbol{E} = [e_{ij}]$. For notation compactness, we collect the energy demands from all consumers j into a demand vector $\boldsymbol{d} = [d_1, d_2, \ldots, d_{|\mathcal{Y}_c|}]$, where element $d_j \triangleq T_j(t)$ is the energy demand from consumer j.

Objective functions: as a first objective, we seek to minimize the difference between the amount of energy that the BS sources $i \in \mathcal{Y}_s$ deliver to the BS consumers $j \in \mathcal{Y}_c$ and the consumers' energy demand. As stated above, the maximum energy that i can send to j is e_{ij}. However, this energy amount can possibly be distributed among multiple consumers and we denote by $y_{ij} \in [0, 1]$ the fraction of e_{ij} that is actually allocated from source i to consumer j, in matrix notation $\boldsymbol{Y} = [y_{ij}]$. We thus write a first cost function as:

$$f_1(\boldsymbol{Y}, \boldsymbol{E}, \boldsymbol{d}) = \sum_{i \in \mathcal{Y}_s} \left(\sum_{j \in \mathcal{Y}_c} y_{ij} e_{ij} - d_j \right)^2 . \tag{4.20}$$

Due to the existence of a single path between any source and consumer pair and due to the fact that each power link can only be used for a single transfer

operation at a time, a desirable solution shall: i) pick source and consumer pairs (i, j) in such a way that the physical distance (g_{ij}) between them is minimized and ii) achieve the best possible match between sources and consumers, i.e., use source i, whose available energy is the closest to that required by consumer j, for all (i, j) pairs. Ideally, for each i we would like y_{ij} to be equal to 1 for a single value of j and zero for any other consumer (i.e., 1-of-$|\mathcal{Y}_c|$ coding scheme, where $|\mathcal{Y}_c|$ gives the number of consumers). If this is infeasible, multiple sources will supply the consumer, leading to $y_{ij} > 0$ for multiple values of j and $\sum_j y_{ij} = 1$. Minimizing the following cost function, amounts to minimizing the number of hops g_{ij} between sources and consumers and favoring solutions with 1-of-$|\mathcal{Y}_c|$ coding for y:

$$f_2(\boldsymbol{Y}, \boldsymbol{G}) = \sum_{i \in \mathcal{Y}_s} \left(\sum_{j \in \mathcal{Y}_c} - \exp\left(\frac{y_{ij}}{g_{ij}}\right) \right) . \tag{4.21}$$

With this cost function we are looking for a sparse solution (i.e., a small number of sources with y_{ij} close to 1). Note that when $y_{ij} \to 1$ and g_{ij} is minimized, the argument y_{ij}/g_{ij} is maximized and the negative exponential is minimized. Also, the exponential function was picked as it is convex, but any increasing and convex function would do.

Solution through convex optimization: at each time slot t, each BS n updates its buffer level $B_n(t)$, using either Eq. (4.1) or Eq. (4.3) (note that $B_n(t-1)$, $H_n(t-1)$, $O_n(t-1)$, $T_n(t-1)$ and $\theta_n(t-1)$ are all known in slot t, see Section 4.3). The MPC problem of Section 4.4.3 is solved. Each source i evaluates e_{ij} for all $j \in \mathcal{Y}_c$ through $e_{ij} = |T_i(t)|a(g_{ij})$, and each consumer j sets its energy demand as $d_j = T_j(t)$. Hence, using Eq. (4.20) and Eq. (4.21), the following *convex* optimization problem is formulated:

$$\min_{\boldsymbol{Y}} \quad \beta f_1(\boldsymbol{Y}, \boldsymbol{E}, \boldsymbol{d}) + (1-\beta) f_2(\boldsymbol{Y}, \boldsymbol{G}) \tag{4.22a}$$

$$\text{subject to:} \quad 0 \leqslant y_{ij} \leqslant 1, \qquad \forall i \in \mathcal{Y}_s, \forall j \in \mathcal{Y}_c, \tag{4.22b}$$

$$\sum_{j \in \mathcal{Y}_c} y_{ij} \leqslant 1, \qquad \forall i \in \mathcal{X}_s, \tag{4.22c}$$

where $\beta \in [0, 1]$ is a weight used to balance the relative importance of the two cost functions. The first constraint represents the fact that y_{ij} is a fraction of

the available energy e_{ij} from source i, and the second constraint encodes the fact that the total energy i that each source transfers cannot exceed its offer $|T_i(t)|$. For any fixed value of β, Eq. (4.22) is a convex minimization problem which can be solved through standard techniques. The optimal solution $\boldsymbol{Y}^* = [y^*_{ij}]$ dictates the energy fraction tat any source i must send to consumer j, this energy share is precisely given by $T^j_i(t) = y_{ij}|T_i(t)|$.

Solution as an assignment problem: at any time t, the energy distribution problem from sources to consumers can alternatively be modeled as an assignment problem, where each source $i \in \mathcal{Y}_s$ has to be *matched* with a consumer $j \in \mathcal{Y}_c$. This approach can be solved through the *Hungarian method* [141], an algorithm capable of finding an optimal assignment for a given square $m \times m$ cost matrix, where $m = \max(|\mathcal{Y}_s|, |\mathcal{Y}_c|)$. An assignment is a set of m entry positions in the cost matrix, no two of which lie in the same row or column. The sum of the m entries of an assignment is its cost. An assignment with the smallest possible cost is referred to as *optimal*. Let $\boldsymbol{C} = [c_{ij}]$ be the cost matrix, where rows and columns respectively correspond to sources i and consumers j. Hence, c_{ij} is the cost of assigning the i-th source to the j-th consumer and is obtained as follows:

$$c_{ij} = \beta(e_{ij} - d_j)^2 + (1 - \beta)\left(-\exp\left(\frac{1}{g_{ij}}\right)\right), \tag{4.23}$$

where $\beta \in [0, 1]$, the first term weighs the quality of the match (d_j should be as close as possible to e_{ij}) and the second the quality of the route. To ensure the cost matrix is a square matrix, additional rows or columns are to be added when the number of sources and consumers differs. As typically assumed, each element in the added row or column is set equal to the largest number in the matrix.

The main difference between the optimal solution found by solving the convex optimization problem (Eq. (4.22)) and that found by the Hungarian method is that the latter *always* returns a one-to-one match between sources and consumers, i.e., each consumer can only be served by a single source (1-of-$|\mathcal{Y}_c|$ coding). While this is desirable to diminish losses, it is not always optimal and can lead to inefficient allocations in some cases, as we shall see shortly.

4.4.5 Energy routing

Next, we describe how the energy transfer from source i to consumer j, in time slot t, is implemented. The following algorithm is executed at the beginning of each time slot t, upon obtaining 1) $T_i(t)$ for each BS source (Section 4.4.3) and 2) the allocation matrix $\boldsymbol{Y}^*$ (Section 4.4.4).

Each time slot is further split into a number of mini slots. Given a certain maximum transmission energy capacity $e_{\max}$ for a power link in a mini slot, the required number of mini slots to transfer a certain amount of energy $y_{ij}e_{ij}$ from source i to consumer j is obtained as $n_{ij} = \lceil y_{ij}e_{ij}/e_{\max} \rceil$. We consider tree topologies, see Fig. 4.1. Since each power link can only be used for a single energy transfer operation at a time, we seek to minimize the number of mini slots that are required, while using disjoint routes. An energy route for the source-consumer pair (i, j) is defined as the collection of intermediate nodes to visit when transferring energy from i to j. If the nodes are direct neighbors in the PPG topology (BSs k and m in Fig. 4.1), or are within the same forward path toward the energy router (k and h), energy is transmitted directly, without passing through the energy router. If instead the nodes belong to different branches, e.g., nodes i and j in Fig. 4.1, the energy router is involved in the transfer. Also, we assume that the energy router is the only node capable of concurrently handling multiple flows that cross at the root of the tree.

The algorithm proceeds as follows (Algorithm 3): 1) a route r_{ij} is identified for each source i and consumer j by using the Dijkstra algorithm. Note that for the given tree topology (see Fig. 4.1) this route is *unique*, 2) the disjoint routes, with no power links in common, are found and are allocated to as many (i, j) pairs as possible, 3) for each of these pairs (i, j), the energy transfer $y_{ij}e_{ij}$ is accomplished using route r_{ij} for a number of mini slots n_{ij}, 4) when the transfer for a pair (i, j) is complete, we check whether a new route is released (i.e., no longer used and available for subsequent transfers). If that is the case, and if this route can be used to transfer energy for any of the remaining pairs (i', j') (not yet considered), this route is allocated to any of the eligible pairs (i', j') for $n_{i',j'}$ further mini slots. This process is repeated until all source-consumer pairs have completed their transfer. For the system configuration of Section 4.5, energy routing is always completed within the

time granularity of the control.

Algorithm 3 Pseudo-code for energy routing

Inputs: $\mathcal{Y}_s$, $\mathcal{Y}_c$ and $\boldsymbol{Y}^*$
for each source $i \in \mathcal{Y}_s$ **do**
    for each consumer $j \in \mathcal{Y}_s$ **do**
        if $y_{ij} > 0$ **then**
            Find route between i and j (Dijkstra)
        end if
    end for
end for
Sort all routes in descending order of n_{ij}.
Create a route table r_{table}.
while r_{table} is not empty **do**
    for each mini slot n_{ij} **do**
        Check status of power links: used, available.
        Find the first implementable route r'_{ij} in r_{table}.
        if r'_{ij} is found **then**
            Find all possible disjoint routes wrt r'_{ij}.
            Remove route(s) from r_{table}.
            Set route(s) and perform the routing.
        end if
        Release power link(s) if any route is completed.
        Update status of power links.
    end for
end while

4.5 Numerical Results

The forecasting approach based on GPs is evaluated in Section 4.5.1, whereas results of the proposed optimization framework are provided in Section 4.5.2, using the algorithm of [78] as a benchmark.

4.5.1 Performance evaluation of the Pattern Learning scheme

The proposed GP-based forecasting method proposed in Section 4.4.2 is here assessed for the runtime multi-step ahead forecasting of time series $H(t)$ and $L(t)$. The time slot duration is set to one hour, $N = 24 \times 14 = 336$ hours

Table 4.4: Average $\text{RMSE}_*^{(t)}$.

	$N_* = 1$	$N_* = 2$	$N_* = 12$	$N_* = 24$
$S = 1$	0.0119	0.0170	0.0385	0.0512
$S = T$	0.0116	0.0166	0.0383	0.0511

(a) Average $\text{RMSE}_*^{(t)}$ for $H(t)$.

	$N_* = 1$	$N_* = 2$	$N_* = 12$	$N_* = 24$
$S = 1$	0.0389	0.0464	0.0670	0.0740
$S = T$	0.0415	0.0483	0.0671	0.0743

(b) Average $\text{RMSE}_*^{(t)}$ for $L(t)$.

(i.e., two weeks of data), $T = 24 \times 60 = 1,440$ hours (i.e., two months of data), and $\sigma_n = 10^{-5}$. This choice of parameters is valid for both time series, as well as the use of the kernel $k(\cdot, \cdot)$ in Eq. (4.13), whereas the *hyperparameters* differ, depending on the nature of data.

In Table 4.4a and Table 4.4b we show the average RMSE for $H(t)$ and $L(t)$, computed evaluating the mean of the RMSE measures up to time $T-(N+N_*)$, where we track $\text{RMSE}_*^{(t)}$ over the N_* test cases, given $N_* = 1, 2, 12, 24$. Also, as we perform the training phase once every S steps, comparing the numerical results when $S = 1$ and $S = T$, i.e., when we update the free GP parameters at each step of the online forecasting routine ($S = 1$), or just once every T steps ($S = T$), at time $t = 1$. In general, the average $\text{RMSE}_*^{(t)}$ increases as we increase the N_* test cases up to 24, which corresponds to predicting the time series one day into the future. However, the worst performance is 0.0743, which is still rather small if we consider that both time series are normalized in $[0, 1]$ prior to processing. Also, predictions for $H(t)$ (Fig. 4.4a) are more precise than predictions for $L(t)$ (Fig. 4.4b), and this is due to the nature of the data, given that we use the same kernel for both time series. In fact, values in $H(t)$ (Fig. 4.4a) follow a more regular behavior than those in $L(t)$ (Fig. 4.4b), with quasi-periodic streams of zero values corresponding to zero solar energy income during the night. These quasi-periodic streams of zero values help reinforcing prediction, while allowing for a higher confidence at nighttime (see Fig. 4.5a). Finally, tuning parameter S explains the impact of re-optimizing the *hyperparameters* according to the most recent history (i.e., two weeks of data), but with a longer execution time. Numerical results suggest that tuning parameter S could be reasonable when data exhibit multiple strong

local behaviors rather than just a strong daily seasonality, and the kernel has to adapt to these. However, $S = 1$ could not be the obvious optimal choice (see Table 4.4a).

In Fig. 4.4a and Fig. 4.4b we show real values and predictions for two weeks of data, where we track the one-step (i.e., $N_* = 1$) predictive mean value at each time slot of the online forecasting routine. The strong daily seasonality is evident, as well as the quasi-periodic structure in data with noise operating at different scales. Note that predictions for $H(t)$ (Fig. 4.4a) are more accurate than those for $L(t)$ (Fig. 4.4b), and this result can be confirmed by comparing the average $\text{RMSE}_*^{(t)}$ in Tables 4.4a and 4.4b for $N_* = 1$. As expected, predictions may be far from real values when some unusual events occur, see, for example, the low solar energy income within hours 456 and 480 (sixth peak from the left), in Fig. 4.4a, or the sudden peaks in the traffic load profile of Fig. 4.4b, which are very day-specific.

In Figs. 4.5a and 4.5b we show real and predicted values for three days of data, i.e., the last two days of the training dataset, and 24 hours for the test set, plotting the multi-step predictive mean value with $N_* = 24$. Here, we compare the use of the kernel $k(\cdot, \cdot)$ in Eq. (4.13) with common base kernels from the literature, such as the popular Squared Exponential (SE) kernel, the Rational Quadratic (RQ) kernel, and the Standard Periodic (SP) kernel, see Eq. (4.13). Also, we compare the use of the kernel $k(\cdot, \cdot)$ in Eq. (4.13) in terms of generalization capabilities over the training dataset and the test set, i.e., we perform forecasting over the training dataset and the test set, after the optimization of the *hyperparameters* given the observations. Note that the proposed kernel (solid line) shows the best performance in terms of forecasting, since composite kernels are more representative than base ones. Specifically, the RMSE is close to zero over the training dataset (due to the fact that we set $\sigma_n = 10^{-5}$, i.e., $\sigma_n \neq 0$), and this result also holds for both the SE and RQ cases. However, the generalization capabilities over the test set are quite limited for SE and RQ. In fact, these base kernels have limited expressive power, and simply act like smoothers. Finally, the SP kernel succeeds in recovering the strong daily seasonality in the data, but it fails to model noise at different scales. Again, its expressive power is quite limited, with respect to our proposed kernel in Eq. (4.13).

4.5.2 Performance evaluation of the Optimization Framework

In this section, the following schemes are compared: i) no energy exchange **(NOEE)**, i.e., the offline BSs only have to rely on the locally harvested energy, ii) convex solution **(CONV)**: this is the scheme of [78], which computes energy allocations solely based on the system configuration in the current time slot. This approach is *myopic*, as no knowledge into the future behavior of the system is exploited. iii) Hungarian solution **(HUNG)**: the energy allocation is found through the Hungarian method of Section 4.4.4; this is also a myopic approach. iv) Convex solution with model predictive control **(GPs+MPC+CONV)**: this is the combined optimization approach of Sections 4.4.2, 4.4.3 and 4.4.4, and v) Hungarian solution with model predictive control **(GPs+MPC+HUNG)**. ii) and iii) carry out energy allocation and routing only considering the current time slot, while iv) and v) also take into account the future system evolution, exploiting pattern learning and multi-step ahead adaptive control.

Before discussing the numerical results, some considerations are in order. All the algorithms purchase some energy from the power grid, although the way in which they use it differs. With NOEE, the energy purchased is exclusively used to power the base stations that are ongrid, whereas those being offgrid have to uniquely rely on the harvested energy. Convex and Hungarian solutions allow some energy redistribution among the base stations. With these schemes, an energy rich BS can transfer energy to other BSs whose energy buffer is depleted. Note that an energy rich base station may belong to either the ongrid set or to the offgrid one. The latter case occurs when, for instance, a BS experiences no traffic during the day and all the energy it harvests is stored locally. In this case, this BS is likely to be "energy rich", and energy transfer schemes consider it as an energy source for other BSs. Looking at the whole BS network, it can gather energy in two ways: i) harvesting it from the environment and ii) purchasing it from the power grid. The harvested energy is basically free of charge and shall be utilized to the best extent: energy transfer among BSs makes this possible. The energy bought by the ongrid BSs is costly and shall also be utilized as efficiently as possible. Below, we shall evaluate both aspects.

Table 4.5: System parameters used in the numerical results.

Parameter	**Value**
Number of BSs, n_s (set $\mathcal{S}$)	18
Number of ongrid BSs (set $\mathcal{S}_{\text{on}}$)	6
Cable resistivity, ρ	$0.023\,\Omega\text{mm}^2/\text{m}$
Cable cross-section, A	$10\,\text{mm}^2$
Length of a power link, ℓ	100 m
Energy buffer capacity, $B_{\max}$	360 kJ
Upper energy threshold, B_{up}	$0.7B_{\max}$ (70%)
Reference energy threshold, B_{ref}	$0.5B_{\max}$ (50%)
Lower energy threshold, B_{low}	$0.1B_{\max}$ (10%)
Time slot duration	1 h
Mini slot duration	60 s
Maximum transmission energy capacity, $e_{\max}$	90 kJ/mini-slot
MPC optimization horizon M	24 h
MPC weight parameter α	0.5
Energy allocation weight parameter β	0.5

For the numerical results, we consider the scenario of Section 4.3. For the EBs, we set $B_{\max} = 360\,\text{kJ}$, which corresponds to a battery capacity of 100 W h (e.g., a small size Li-Ion battery). The slot time is set to one hour, solar EH traces are obtained using SolarStat [127] for the city of Chicago, and the BS topology is that of Fig. 4.1, with 6 ongrid BSs and a total of $n_s = 18$ BSs. The other simulation parameters are listed in Table 4.5. The curves plotted in Figs. 4.6, 4.7, 4.9 and 4.10 are obtained averaging over $1,000$ simulation instances. Each simulation instance accounts for 168 hours, i.e., one week. The harvested energy profile for each BS is set at the beginning of each simulation instance starting from a specific date, which is picked at random from the real-trace dataset. For the traffic load, each BS picks one of the two available load clusters at random, with probability p (in the abscissa). Moreover, Figs. 4.6, 4.7 and 4.8 are obtained with ideal EBs, i.e., $\beta_{\text{loss}} \to \infty$. This changes in Figs. 4.9 and 4.10 where both cases, ideal and non-ideal ($\beta_{\text{loss}} = 3$), are shown. Finally, the location of the ongrid BSs within the topology of Fig. 4.1 changes randomly at every simulation instance.

In Fig. 4.6, we show the average BS energy buffer level over different traffic load configurations. For the load assignment, each BS independently picks one of the two traffic clusters in Section 4.3.3: cluster 2 (low traffic load) is picked with probability p and cluster 1 (high load) is picked with probability $1-p$. p is then varied as a free parameter along the abscissa. As expected, the average energy buffer level when $p = 0$ is lower than that with $p = 1$, as the traffic load in cluster 1 is higher. Regarding the approaches, the highest difference in the energy buffer levels is found between NOEE and GPs+MPC+HUNG, with an increment of around 60% (on average) when MPC is adopted. Moreover, the Hungarian methods outperform convex solutions because, with their assignment policy, any consumer is matched to a single source and this reduces the amount of energy that is distributed, leaving more energy in the energy rich buffers. As we show shortly, this behavior is not really desirable as, e.g., it leads to higher outage probabilities.

As a proxy to the network Quality of Service (QoS), the outage probability at time t, $\gamma(t)$, is here defined as the ratio between the number of BSs whose energy buffer level gets completely depleted, and the total number of BSs in the system n_s. The outage probability $\gamma(t)$ as a function of the traffic load is plotted in Fig. 4.7. For all schemes, $\gamma(t)$ is an increasing function of the load. The probability that a BS runs out of service due to energy scarcity is higher when energy cannot be transferred among BSs (NOEE) and is in general very high across the whole day for HUNG-based solutions. However, applying MPC to the Hungarian method leads to a reduction in the average outage probability of about 54%. Moreover, from Fig. 4.8 we see that with the Hungarian method, $\gamma(t)$ increases when the amount of energy harvested is very little (i.e., nighttime). The problem is that the Hungarian allocation technique returns a matching of source-consumer pairs, where each source is allocated to a single consumer and, in turn, some of the BSs may not be allocated in some time slots (due to the imbalance between number of sources and number of consumers). This leads to high outage probabilities for the considered scenario. CONV-based techniques are more flexible in this respect, as they allow energy transfer from multiple BSs and in different amounts. This translates into a zero outage probability in both cases, with and without MPC.

From the previous graphs, one may conclude that CONV and GPs+MPC+CONV (foresighted optimization) provide the same benefits, being both capa-

ble of lowering the outage probability down to zero. However, looking at additional metrics reveals that the two approaches show important differences. For example, in Fig. 4.9 we compare these solutions in terms of amount of energy that ongrid BSs purchase from the power grid. A big gap can be observed between the two schemes, proving that the application of pattern learning and MPC is indeed highly beneficial, leading to a reduction of more that 55% in the amount of energy purchased from the power grid.

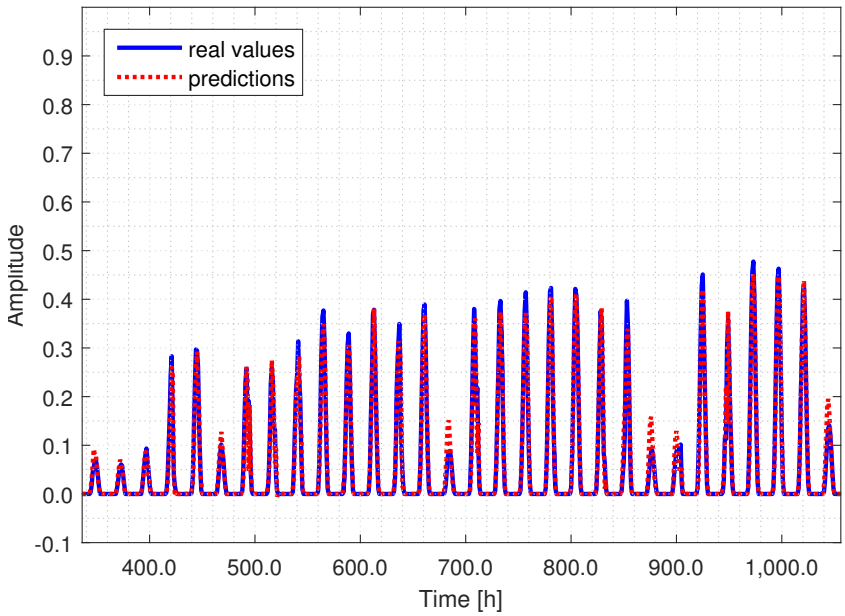
Along these lines, we perform another set of simulations by putting a cap on the maximum amount of energy that can be bought during a full day by the ongrid BSs. Specifically, we define a *purchased energy threshold* η as the ratio between the amount of energy that each ongrid BS is allowed to purchase and the total amount of energy it would require to serve a fully loaded scenario across an entire day, i.e., the BS purchases energy up to $B_{\max}$ every time slot. A plot of $\gamma(t)$ against threshold η is shown in Fig. 4.10. From this graph, we see that predictive control (GPs+MPC+CONV) leads to a much smaller outage probability than CONV. Moreover, as η increases beyond 0.5 the outage probability drops to zero, which is a big improvement with respect to CONV, for which γ is about 10%. Similar results are obtained for GPs+MPC+HUNG when compared with HUNG, although in this case the gain is slightly smaller.

The use of non-ideal energy buffers is evaluated in Figs. 4.9 and 4.10, using $\beta_{\text{loss}} = 3$. In this case, the energy losses incurred in the charging and discharging processes lead to an increase in the energy purchased from the power grid ($\approx 10\%$) and in the outage probability ($\approx 15\%$) over time, due to the smaller EB levels with respect to the ideal EB scenario.

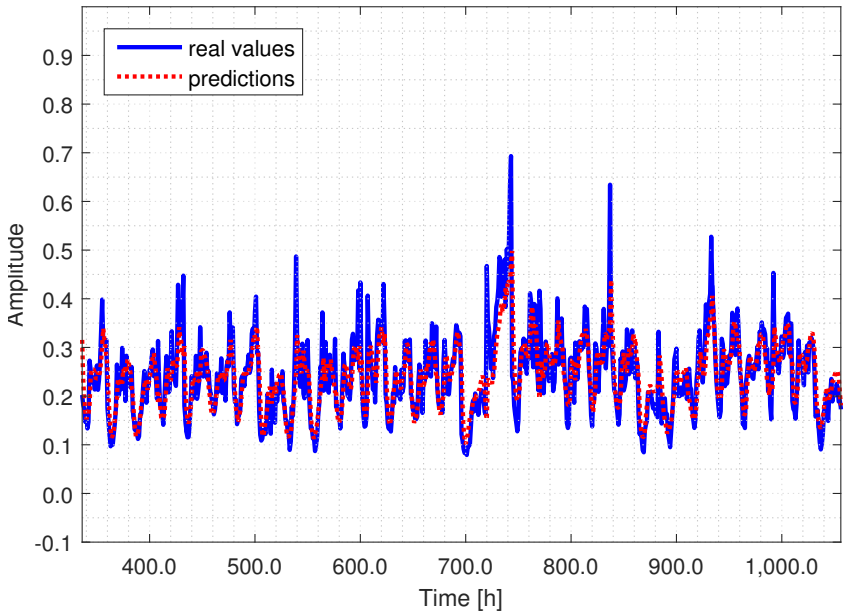
4.6 Conclusions

In this chapter, we have considered small cell deployments where energy harvesting and packet power networks are combined to provide energy self-sustainability through the use of own-generated energy and carefully planned power transfers among network elements. This amounts to a combined learning and optimization problem (resource scheduling), where learning is carried out on energy arrival (harvested ambient energy) and traffic load traces and this knowledge is exploited, at runtime, for the computation of optimal en-

ergy transfer policies among the distributed energy buffers. This foresighted optimization is performed combining model predictive control and convex optimization techniques. Numerical results reveal great advantages over the case where energy transfer schedules are optimized disregarding future energy and load forecasts: the amount of energy purchased from the power grid is reduced by more than 50% and the outage probability is lowered to zero in nearly all scenarios.

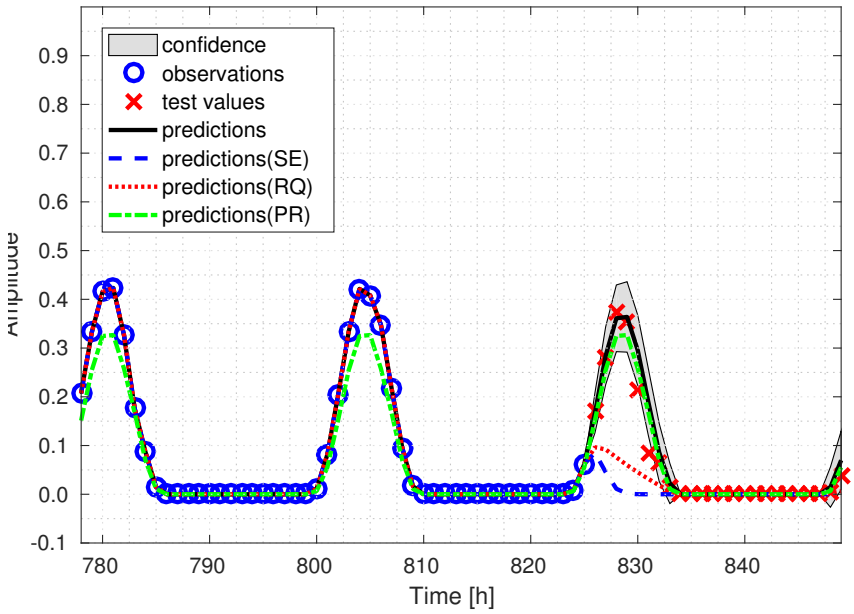


(a) One-step predictive mean value for $H(t)$.

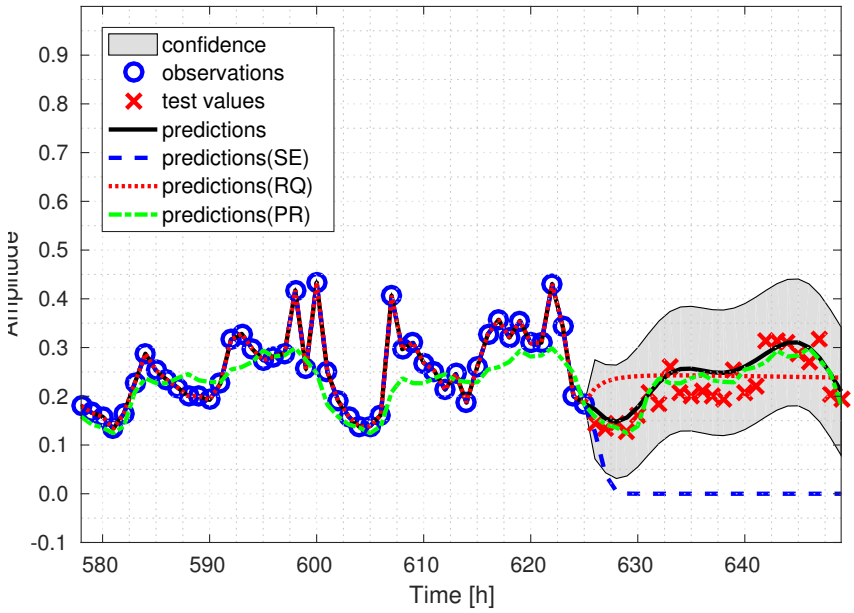


(b) One-step predictive mean value for $L(t)$.

Figure 4.4: One-step online forecasting for two weeks of data.



(a) Multi-step predictive mean value for $H(t)$.



(b) Multi-step predictive mean value for $L(t)$.

Figure 4.5: Multi-step prediction with different kernels.

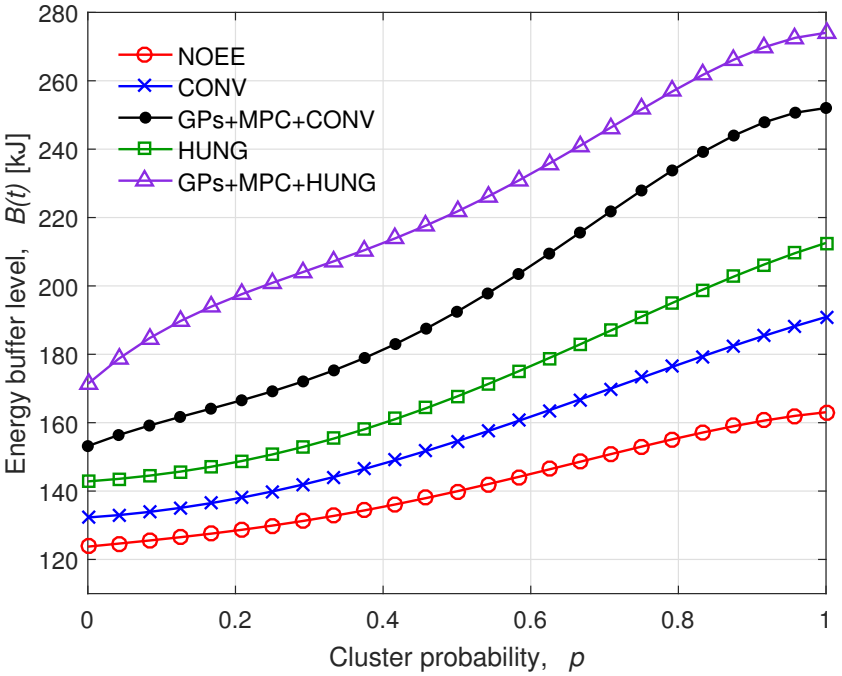


Figure 4.6: Energy buffer level *vs* cluster probability p.

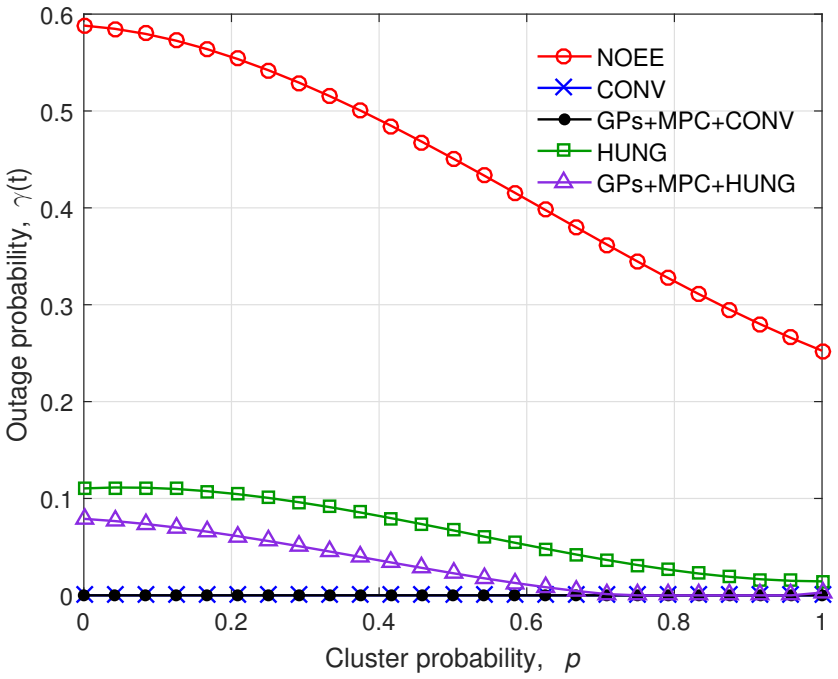


Figure 4.7: Outage probability $\gamma(t)$ *vs* cluster probability p.

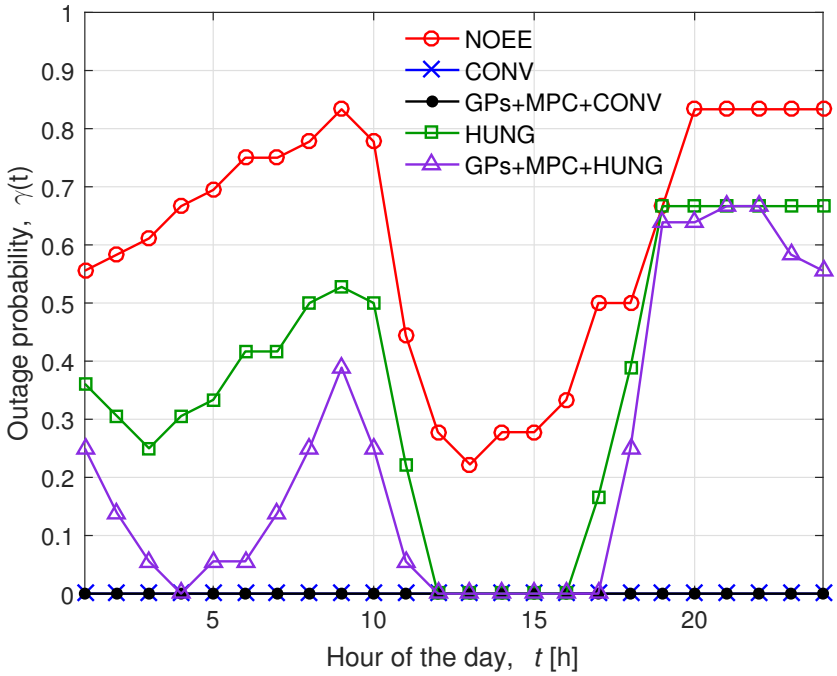


Figure 4.8: Outage probability $\gamma(t)$ over a day.

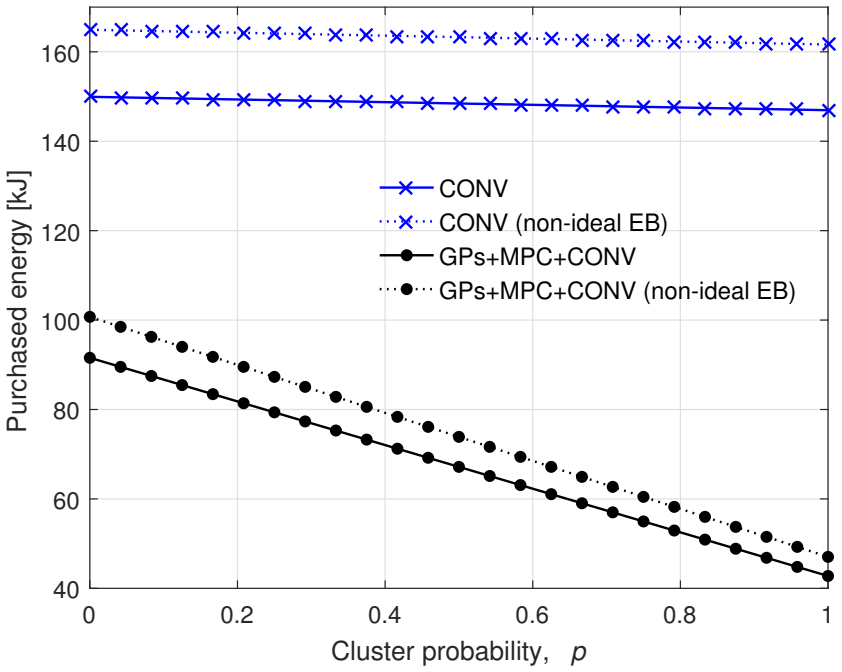


Figure 4.9: Purchased energy *vs* cluster probability p.

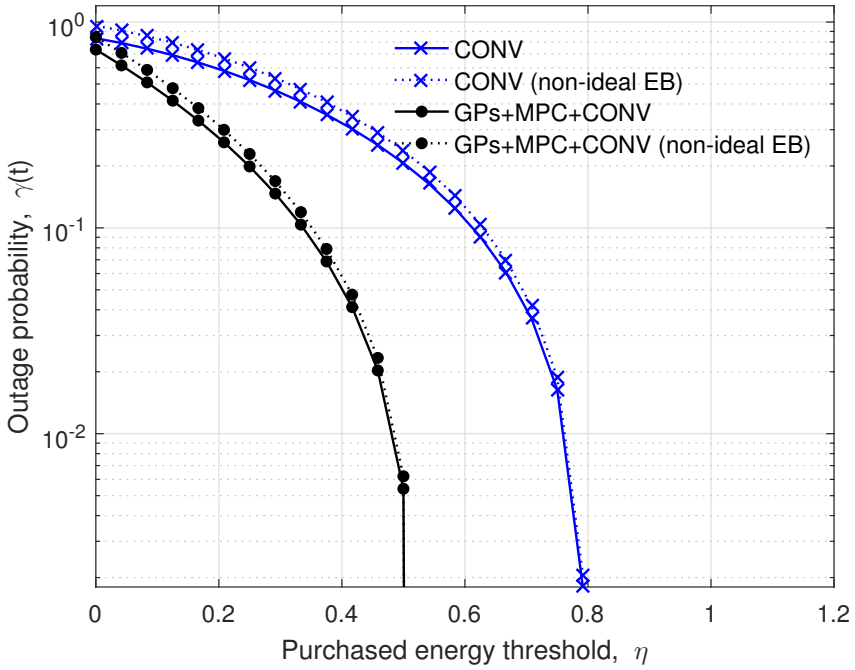


Figure 4.10: Outage probability $\gamma(t)$ *vs* purchased energy threshold.

Chapter 5

Beam Training and Data Transmission Optimization in Millimeter-Wave Vehicular Networks

Future vehicular communication networks call for new solutions to support their capacity demands, by leveraging the potential of the millimeter-wave (mm-wave) spectrum. Mobility, in particular, poses severe challenges in their design, and as such shall be accounted for. A key question in mm-wave vehicular networks is how to optimize the trade-off between directive Data Transmission (DT) and directional Beam Training (BT), which enables it. In this chapter, learning tools are investigated to optimize this trade-off. In the proposed scenario, a Base Station (BS) uses BT to establish a mm-wave directive link towards a Mobile User (MU) moving along a road. To control the BT/DT trade-off, a Partially Observable (PO) Markov Decision Process (MDP) is formulated, where the system state corresponds to the position of the MU within the road link. The goal is to maximize the number of bits delivered by the BS to the MU over the communication session, under a power constraint. The resulting optimal policies reveal that adaptive BT/DT procedures significantly outperform common-sense heuristic schemes, and that specific mobility features, such as user position estimates, can be effectively used to enhance the overall system performance and optimize the available system resources.

5.1 Introduction

The state-of-the-art protocols for vehicular communication address vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I) communication systems, generally termed V2X. Currently, these communication systems enable a maximum data rate of 100 Mbps for high mobility (using 4G) [142, 143], which are not deemed sufficient to support applications such as autonomous driving, augmented reality and infotainment, which will populate next-generation vehicular networks. Therefore, future vehicular communication networks call for new solutions to support their capacity demands, by leveraging the huge amount of bandwidth in the $30-300$ GHz band, the so called millimeter-wave (mm-wave) spectrum. While communication at these frequencies is ideal to support high capacity demands, it relies on highly directional transmissions, which are extremely susceptible to the vehicle mobility. Therefore, a key question is: *How do we leverage mobility information to optimize the trade-off between directive Data Transmission (DT) and directional Beam Training (BT), which enables it, to optimize the communication performance? How much do we gain by doing so?* To address these questions and optimize this trade-off, in this chapter we envision the use of learning tools. We demonstrate significant gains compared to common-sense beam alignment schemes.

Compared to conventional lower frequencies, propagation at mm-waves poses several challenges, such as high propagation loss and sensitivity to blockage. To counteract these effects, mm-wave systems are expected to use large antenna arrays to achieve a large beamforming gain via directional transmissions. However, these techniques demand extensive beam training, such as beam sweeping, estimation of angles of arrival and of departure, and data-assisted schemes [144], as well as beam tracking [145]. Despite their simplicity, the overhead incurred by these algorithms may ultimately offset the benefits of beamforming in highly mobile environments [142, 143]. While wider beams require less beam training, they result in a lower beamforming gain, hence smaller achievable capacity [146]. While contextual information, such as GPS readings of vehicles [144], may alleviate this overhead, it does not eliminate the need for beam training due to noise and GPS acquisition inaccuracy. Thus, the design of schemes that alleviate this overhead is of great importance.

In all of the aforementioned works, a priori information on the vehicle's mobility is not leveraged in the design of BT/DT protocols. In contrast, *we contend that leveraging such information via adaptive beam design techniques can greatly improve the performance of automotive networks* [147,148]. In this chapter, we bridge this gap by designing adaptive strategies for BT/DT that leverage a priori mobility information via Partially Observable (PO) Markov Decision Processes (MDPs). Our numerical evaluations demonstrate that these optimized policies significantly outperform common-sense heuristic schemes, which are not tailored to the vehicle's observed mobility pattern. Compared to [149], which develops an analytical framework to optimize the BT/DT trade-off and the BT parameters based on the "worst-case" mobility pattern, in this chapter, we assume a statistical mobility model.

In the proposed scenario, a Base Station (BS) attempts to establish a mm-wave directive link towards a Mobile User (MU) moving along a road. To this end, it alternates between BT and DT. The goal is to maximize the number of bits delivered by the BS to the MU over the communication session, under a power constraint. To manage the BT/DT trade-off, we exploit a POMDP formulation, where the system state corresponds to the position of the MU within the road link. Specifically, we implement a POMDP with temporally extended actions (i.e., actions with different durations) to model the different temporal scales of BT and DT, and a constraint on the available resources of the system. POMDPs model an agent decision process in which the system dynamics are determined by the underlying MDP (in this case, the MU dynamics), but the agent cannot directly observe the system state. Instead, it maintains a probability distribution (called *belief*) over the world states, based on observations and their distribution, and the underlying MDP. An exact solution to a POMDP yields the optimal action for each possible belief over the world states. POMDPs have been successfully implemented in a variety of real-world sequential decision processes, including robot navigation problems, machine maintenance, and planning under uncertainty [150,151]. To address the complexity of POMDPs, we use PERSEUS [152], an approximate solution technique which uses a sub-set of belief points as representative of the belief state. However, in contrast to the original formulation using random belief point selection, we tailor it by selecting a deterministic set of belief points representing uncertainty in MU position, and demonstrate significant perfor-

mance gains. A unified approach for constrained MDP is given by [153, 154]. Notably, there has been relatively little development in the literature for incorporating constraints into the POMDP [155–158]. In order to address the resource constraints in our problem, we propose a Lagrangian method, and an online algorithm to optimize the Lagrangian variable based on the target cost constraint.

This chapter is organized as follows. In Section 5.2, we introduce the system model, followed by the optimization in Section 5.3. We present numerical results in Section 5.4, followed by concluding remarks in Section 5.5.

5.2 System Model

We consider a scenario where a BS aims at establishing a mm-wave directive link with a MU moving along a road. To this end, it alternates between BT and DT: with BT, the BS refines its knowledge on the position of the MU within the road link, to perform more directive DT. Our goal is to maximize the number of bits that the BS delivers to the MU during a transmission episode, defined as the time interval between the two instants when the MU enters and exits the coverage range of the BS, under a power constraint. The mm-wave link is in Line-Of-Sight (LOS), thus knowledge on the position of the MU is sufficient for aligning the beams [148].

5.2.1 Problem formulation

We consider a *dense* cell deployment, as shown in Fig. 5.1. The MU is associated with its closest BS, at a distance d_0 from the road link. The road link served by the reference BS is divided into S road sub-links of equal length $\Delta_{\mathrm{s}} = 2d_0 \tan(\Theta/2)/S$, where Θ is the maximum coverage range of the BS. We let $\mathcal{S} \equiv \{1, \ldots, S\}$ be the set of indices of the S road sub-links. The BS associates a beam with each one of the S road sub-links, with angular support, for the s-th beam,

$$\Phi_s = \left[\tan^{-1} \frac{-d_0 \tan(\Theta/2) + (s-1)\Delta_{\mathrm{s}}}{d_0}, \tan^{-1} \frac{-d_0 \tan(\Theta/2) + s\Delta_{\mathrm{s}}}{d_0} \right] \quad (5.1)$$

and beamwidth $\theta_s = |\Phi_s|$, so that $\cup_{s\in\mathcal{S}} \Phi_s = [-\Theta/2, \Theta/2]$ and $\sum_{s\in\mathcal{S}} \theta_s = \Theta$.

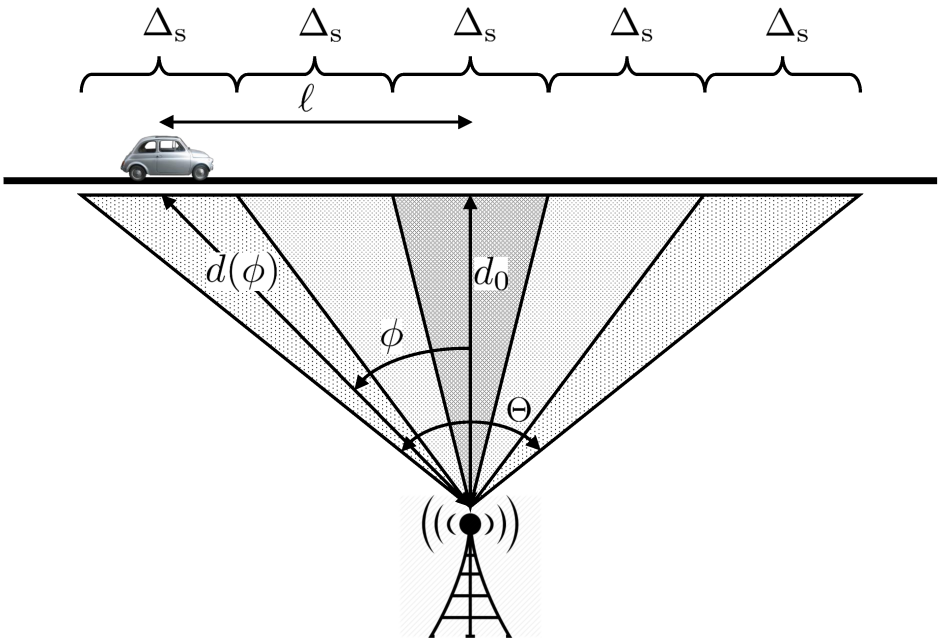


Figure 5.1: A *dense* cell deployment.

The time is discretized into micro time-slots of duration Δ_t, with Δ_t being the time for a Primary Synchronization Signal (PSS), which allows a proper channel estimation at the receiver [146]. At time t, the MU is located in one of the S road sub-links, until it exits the coverage area of the BS, denoted by the absorbing state $\bar{s} = S + 1$. We denote the sub-link occupied by the MU at time t as $X_t \in \mathcal{S}$. We assume that the position of the MU within the road link evolves among the S road sub-links following a random walk with probabilities $0 \leqslant q < p < 1$, with $1 - p - q > 0$, where $p = \mathbb{P}[X_{t+1} = s + 1 \mid X_t = s]$ and $q = \mathbb{P}[X_{t+1} = s - 1 \mid X_t = s]$. Under this model, the MU will exit the BS coverage area at some point. We can view such random walk as an abstraction of the following *physical* mobility model, where the MU moves with average speed $\mathbb{E}[v]$ and speed variance $\mathrm{Var}[v]$: assume that the MU moves at speed v_t at time t, with $v_t \in \{0, v_{\max}, -v_{\max}\}$. Also, let $\mathbb{P}[v_t = v_{\max}] = p$, $\mathbb{P}[v_t = -v_{\max}] = q$, and $\mathbb{P}[v_t = 0] = 1 - p - q$. Note that the maximum speed supported by this model is $v_{\max} \leqslant \Delta_s/\Delta_t$ (otherwise, the MU may move more than one sub-link within a single micro-slot). It follows that $\mathbb{E}[v] = v_{\max}(p - q) > 0$ and $\mathrm{Var}[v] = v_{\max}^2(p + q) - (\mathbb{E}[v])^2 > 0$. Thus, given average $\mathbb{E}[v]$ and $\mathrm{Var}[v]$, we

obtain p and q as

$$\begin{aligned} p &= \frac{\mathrm{Var}[v] + (\mathbb{E}[v])^2}{2v_{\max}^2} + \frac{\mathbb{E}[v]}{2v_{\max}}, \\ q &= \frac{\mathrm{Var}[v] + (\mathbb{E}[v])^2}{2v_{\max}^2} - \frac{\mathbb{E}[v]}{2v_{\max}}. \end{aligned} \tag{5.2}$$

To meet the conditions for the probabilities $0 \leqslant q < p < 1$, with $1-p-q > 0$, the following inequalities must hold:

$$\begin{aligned} &p < 1 \rightarrow \mathrm{Var}[v] < -(\mathbb{E}[v])^2 - v_{\max}\mathbb{E}[v] + 2v_{\max}^2, \\ &q \geqslant 0 \rightarrow \mathrm{Var}[v] \geqslant -(\mathbb{E}[v])^2 + v_{\max}\mathbb{E}[v], \\ &1-p-q > 0 \rightarrow \mathrm{Var}[v] + (\mathbb{E}[v])^2 < v_{\max}^2, \end{aligned} \tag{5.3}$$

which defines a region of feasible pairs $(\mathbb{E}[v], \mathrm{Var}[v])$. This model can be extended, e.g., to account for memory in the velocity process, although we leave it for future work.

During BT or DT, at time t, the BS transmits using a beam that covers a sub-set of sub-links, $\hat{\mathcal{S}}_t \subseteq \mathcal{S}$, part of our design. Assuming a large antenna array, which allows for arbitrarily sharp beam patterns (i.e., the beams are much larger than the minimum resolution of the antenna array), the beam is designed in order to support a target SNR SNR_t on the beam support, $\mathcal{B}_t \equiv \cup_{s\in\hat{\mathcal{S}}_t}\Phi_s$. To this end, we let $P_t(\phi)$ be the power per radian projected in the angular direction $\phi \in \mathcal{B}_t$, and $P_t(\phi) = 0, \phi \notin \mathcal{B}_t$. To attain the target SNR constraint, we must have that

$$\frac{\Gamma P_t(\phi)}{d(\phi)^2} = \mathrm{SNR}_t, \tag{5.4}$$

where $\Gamma \triangleq \lambda^2\xi/(8\pi N_0 W_{\mathrm{tot}})$ is the SNR scaling factor, $\lambda = f_c/c$ is the wavelength, N_0 is the noise power spectral density, ξ is the antenna efficiency, W_{tot} is the bandwidth, and $d(\phi) = d_0\sqrt{1+\tan(\phi)^2}$ is the distance of the point in the road link at angular direction ϕ, so that $d(\phi)^{-2}$ models distance dependent path loss. It follows that the total transmit power is given by

$$P_t = \int_{\mathcal{B}_t} P_t(\phi)\mathrm{d}\phi = \mathrm{SNR}_t \sum_{s\in\hat{\mathcal{S}}_t} \int_{\Phi_s} \frac{d_0^2}{\Gamma}[1+\tan(\phi)^2]\mathrm{d}\phi. \tag{5.5}$$

Using the change of variables $\phi \to \ell = d_0 \tan(\phi)$, we obtain

$$P_t = \mathrm{SNR}_t \frac{1}{\Gamma} \sum_{s \in \hat{\mathcal{S}}_t} \int_{-d_0 \tan(\Theta/2)+(s-1)\Delta_\mathrm{s}}^{-d_0 \tan(\Theta/2)+s\Delta_\mathrm{s}} d_0 \mathrm{d}\ell = \mathrm{SNR}_t \frac{\Delta_\mathrm{s} d_0}{\Gamma} |\hat{\mathcal{S}}_t|. \tag{5.6}$$

In other words, the total transmit power is independent of the sub-link indices, but depends solely on the number of sub-links $|\hat{\mathcal{S}}_t|$ and on the target SNR. This result is in line with the intuition that larger distances are achievable via smaller beamwidths, and vice versa [148].

During DT, assuming isotropic reception at the MU, such target SNR implies an achievable rate given by

$$R_t = W_\mathrm{tot} \log_2 \left(1 + \frac{\Gamma}{\Delta_\mathrm{s} d_0} \frac{P_t}{|\hat{\mathcal{S}}_t|} \right). \tag{5.7}$$

During BT, the SNR is set so as to achieve target mis-detection and false-alarm probabilities. To design this parameter, the generic signal detection problem corresponds to receiving a signal $y[l]$, $l = 1, \ldots, L$, over a noisy channel. The two hypotheses are

$$\begin{aligned} &\mathcal{H}_0 : y[l] = w[l] \qquad \text{(no signal at the RX)} \\ &\mathcal{H}_1 : y[l] = x[l] + w[l] \quad \text{(signal at the RX)} \end{aligned} \tag{5.8}$$

where $w[l]$, $l = 1, \ldots, L$, are independent random variables, $w[l] \sim \mathcal{N}(0, \sigma_w^2)$, with $\sigma_w^2 = N_0$. Our task is to decide in favor of $\mathcal{H}_0$ or $\mathcal{H}_1$ on the basis of the measurements $y[l]$, $l = 1, \ldots, L$, i.e.,

$$\mathbb{P}(y[1], \ldots, y[L] | \mathcal{H}_1) \mathbb{P}(\mathcal{H}_1) \gtrless \mathbb{P}(y[1], \ldots, y[L] | \mathcal{H}_0) \mathbb{P}(\mathcal{H}_0),$$

or equivalently,

$$\sum_{l=1}^{L} y[l] x[l] \gtrless \sigma_w^2 \ln \left(\frac{\mathbb{P}(\mathcal{H}_0)}{\mathbb{P}(\mathcal{H}_1)} \right) + \frac{1}{2} E_x \,, \tag{5.9}$$

where $E_x = \sum_{l=1}^{L} x[l]^2$ is the energy of the pilot signal $x[l]$. If the Neyman-Pearson formulation is used, then the right hand side of Eq. (5.9) is replaced by a decision threshold $\bar{\rho}$, function of the target error probability. According to

the Neyman-Pearson Lemma [159], for a given target error probability, we can derive a decision rule as follows. The false-alarm probability, $\mathbb{P}_{\text{FA}}$ (accept $\mathcal{H}_1$ when $\mathcal{H}_0$ is true), is given as $\mathbb{P}_{\text{FA}} = \int_{\bar{\rho}}^{\infty} \mathbb{P}(y \mid \mathcal{H}_0)dy = Q(\frac{\bar{\rho}}{\sigma_w\sqrt{E_x}})$, where $Q(\cdot)$ is the Q-function. The mis-detection probability, $\mathbb{P}_{\text{MD}}$ (accept $\mathcal{H}_0$ when $\mathcal{H}_1$ is true), is given as $\mathbb{P}_{\text{MD}} = 1 - \mathbb{P}_{\text{D}}$, where the probability of correct detection is given by $\mathbb{P}_{\text{D}} = \int_{\bar{\rho}}^{\infty} \mathbb{P}(y \mid \mathcal{H}_1)dy = Q(\frac{\bar{\rho}-E_x}{\sigma_w\sqrt{E_x}}) = Q(Q^{-1}(\mathbb{P}_{\text{FA}}) - \frac{\sqrt{E_x}}{\sigma_w})$, which shows that $\mathbb{P}_{\text{D}}$ is a function of $\mathbb{P}_{\text{FA}}$. Applying the inverse $Q^{-1}(\cdot)$ to both sides of the last equation, leads to a measure of the SNR required to attain the target error performance:

$$\text{SNR}_t = \frac{E_x}{\sigma_w^2} = (Q^{-1}(\mathbb{P}_{\text{FA}}) - Q^{-1}(\mathbb{P}_{\text{D}}))^2, \tag{5.10}$$

which is plugged into Eq. (5.6) to find the transmit power as a function of the number of sub-links covered, $|\hat{\mathcal{S}}_t|$.

5.2.2 Partially Observable Markov Decision Process

Next, we define a constrained Partially Observable (PO) Markov Decision Process (MDP).

States: $\bar{\mathcal{S}}$ is a finite set of states describing the position of the MU within the road link, along with the absorbing state $\bar{s}$ when the MU exits the coverage area of the BS. Therefore, $\bar{\mathcal{S}} \equiv \mathcal{S} \cup \{\bar{s}\}$, and $\mathcal{S} \equiv \{1, \ldots, S\}$ is the set of road sub-links.

Actions: $\mathcal{A}$ is a finite set of actions that the BS can perform. Specifically, the BS can perform actions for Beam Training (BT) and actions for Data Transmission (DT), which involve selection of the transmission beam, power, and duration. In general, $a \in \mathcal{A}$ is in the form $a = (\hat{\mathcal{S}}, P_c, T_c)$, where: $c = \{\text{BT}, \text{DT}\}$ refers to the action class; $\hat{\mathcal{S}} \subseteq \mathcal{S}$ is a sub-set of sub-links, defining the support of the transmission beam; P_c is the transmission power per beam, such that $P_t = P_c|\hat{\mathcal{S}}|$ in Eq. (5.6) is the total transmit power, T_c is the transmission duration of action $a \in \mathcal{A}$ (number of micro time-slots of duration Δ_{t}). If $c = \text{BT}$, then $a = (\hat{\mathcal{S}}, P_{\text{BT}}, T_{\text{BT}})$, where P_{BT} and T_{BT} are fixed parameters of the model. Specifically, we assume that BT actions perform simultaneous beamforming over $\hat{\mathcal{S}}$ in one interval of Δ_{t} seconds (i.e., $T_{\text{BT}} = 1$). Also, $P_{\text{BT}} = P_{\min}$, where $P_{\min}$ is the power per beam required to attain the target SNR constraint, i.e., $P_{\min} = \text{SNR}\frac{\Delta_{\text{s}} d_0}{\Gamma}$, and SNR is a function of false-

alarm and mis-detection probabilities $\mathbb{P}_{\text{FA}}$ and $\mathbb{P}_{\text{MD}}$, which are also fixed parameters of the model, via Eq. (5.10). If $c = \text{DT}$, then $a = (\hat{\mathcal{S}}, P_{\text{DT}}, T_{\text{DT}})$, where P_{DT} and T_{DT} are part of the optimization. Specifically, we assume that DT actions perform simultaneous data communication over $\hat{\mathcal{S}}$ for $T_{\text{DT}} - 1$ micro time-slots, where the last interval of Δ_{t} seconds is dedicated to the ACK/NACK feedback transmission from the MU to the BS. During DT, the transmission rate follows from Eq. (5.7). Note that the action space grows as $|\hat{\mathcal{S}}| = \sum_{s=1}^{S} S!/s!/(S-s)! = 2^S - 1$. To reduce its cardinality, we restrict $\mathcal{A}$ such that $\hat{\mathcal{S}}$ is a sub-set of *consecutive* indices in $\mathcal{S}$, i.e., the beam directions specified by $\hat{\mathcal{S}}$ define a compact range of transmission for the BS. Thus, $|\hat{\mathcal{S}}| = S(S+1)/2$.

Observations: $\mathcal{O}$ is a finite set of observations, defined as $\mathcal{O} \equiv \{\text{ACK}, \text{NACK}, \bar{s}\}$. Specifically, $o = \bar{s}$ means that the MU exited the coverage area of the BS; for simplicity, in this chapter we assume that such event is observable, i.e., the BS knows when the MU exited its coverage area.

Transition probabilities: $\mathbb{P}(s'|s,a)$ is the transition probability from $s \in \bar{\mathcal{S}}$ to $s' \in \bar{\mathcal{S}}$ given $a \in \mathcal{A}$. Note that these probabilities are a function of the duration T_c of action a. If the transmission duration of $a \in \mathcal{A}$ is $T_c = 1$, then we store the 1-step transition probabilities into matrix $\mathbf{M} = [\mathbf{M}_{ss'}]$, with elements $\mathbf{M}_{ss'} = \mathbb{P}(s'|s,a)$ given by the 1-step mobility model, as:

$$\begin{aligned}
&\mathbb{P}(s'|s,a) = p && s' = s+1,\ s = 1,\dots,S \\
&\mathbb{P}(s'|s,a) = q && s' = s-1,\ s = 2,\dots,S \\
&\mathbb{P}(s'|s,a) = 1-p-q && s' = s,\ s = 2,\dots,S \\
&\mathbb{P}(s'|s,a) = 1-p && s' = s,\ s = 1 \\
&\mathbb{P}(s'|s,a) = 1 && s' = s,\ s = \bar{s}.
\end{aligned} \tag{5.11}$$

If the transmission duration of $a \in \mathcal{A}$ is $T_c = N$, then we compute the N-step transition probabilities into matrix $\mathbf{M}^N$, i.e., we take the N-th power of matrix $\mathbf{M}$ so as to account for the N-step evolution of the system state under $a \in \mathcal{A}$ with transmission duration T_c.

Observation model: $\mathbb{P}(o|s,a,s')$ is the probability of observing $o \in \mathcal{O}$ given $s \in \bar{\mathcal{S}}$ and $a \in \mathcal{A}$ with transmission duration T_c, ending in $s' \in \bar{\mathcal{S}}$. We assume that the BS can successfully perform $a = (\hat{\mathcal{S}}, P_c, T_c)$ if the MU remains within $\hat{\mathcal{S}}$ for T_c subsequent micro time-slots, i.e., the MU does not exit from the beam support, so that all signal is received. In this case, the MU feeds back an

ACK to the BS, $o = \text{ACK}$. Therefore, we define $X_0^{T_c} = \{X_0, \ldots, X_{T_c}\}$, as the system state path from time 0 to time T_c, and the event $E_{s,s'}^N = \{X_0^{T_c} \in \hat{\mathcal{S}}^{T_c+1} \mid X_0 = s, X_{T_c} = s', T_c = N\}$, meaning that the system state path $X_0^{T_c}$ remains within $\hat{\mathcal{S}}$ for T_c subsequent micro time-slots, given that $X_0 = s$, $X_{T_c} = s'$, $T_c = N$. In order to compute it, we also define matrix $\tilde{\mathbf{M}}$ as the transition probability matrix restricted to the beam support $\hat{\mathcal{S}}$, i.e., $\tilde{\mathbf{M}} = [\tilde{\mathbf{M}}_{ss'}]$, with elements $\tilde{\mathbf{M}}_{ss'} = \mathbf{M}_{ss'}$ if $\{s, s'\} \in \hat{\mathcal{S}}$, otherwise $\tilde{\mathbf{M}}_{ss'} = 0$. We derive $\mathbb{P}(E_{s,s'}^N)$ as:

$$\begin{aligned}
&\mathbb{P}(E_{s,s'}^N) = \mathbb{P}(X_0^{T_c} \in \hat{\mathcal{S}}^{T_c+1} \mid X_0 = s, X_{T_c} = s', T_c = N) \\
=&\frac{1}{\mathbf{M}_{ss'}^N}\mathbb{P}(X_0^{T_c} \in \hat{\mathcal{S}}^{T_c+1}, X_{T_c} = s' \mid X_0 = s, T_c = N) \\
=&\frac{1}{\mathbf{M}_{ss'}^N}\sum_{s_0^N \in \hat{\mathcal{S}}^{N+1}} \prod_{z=0}^{N-1} \mathbb{P}(X_{z+1}{=}s_{z+1} \mid X_z{=}s_z){=}\frac{\tilde{\mathbf{M}}_{ss'}^N}{\mathbf{M}_{ss'}^N}.
\end{aligned} \tag{5.12}$$

Given $a \in \mathcal{A}$ with transmission duration T_c, $\mathbb{P}(o|s, a, s')$ is defined as follows. If $c = \text{BT}$, we account for false-alarm and mis-detection errors in the beam detection process. In particular, if $\{s, s'\} \in \hat{\mathcal{S}}$ (i.e., the MU is within the beam support during the duration of BT) then $\mathbb{P}(\text{ACK}|s, a, s') = \mathbb{P}_{\text{D}}$ (correct detection) and $\mathbb{P}(\text{NACK}|s, a, s') = \mathbb{P}_{\text{MD}}$ (mis-detection); on the other hand, if $\{s, s'\} \notin \hat{\mathcal{S}}$ (i.e., the MU is outside of the beam support during the duration of BT), then $\mathbb{P}(\text{ACK}|s, a, s') = \mathbb{P}_{\text{FA}}$ (false-alarm) and $\mathbb{P}(\text{NACK}|s, a, s') = 1 - \mathbb{P}_{\text{FA}}$. If $c = \text{DT}$, then the transmission is successful if the event $E_{s,s'}^N$ occurs, so that $\mathbb{P}(\text{ACK}|s, a, s') = \mathbb{P}(E_{s,s'}^N)$ for $\{s, s'\} \in \mathcal{S}$, and $\mathbb{P}(\text{NACK}|s, a, s') = 1 - \mathbb{P}(\text{ACK}|s, a, s')$. Finally, $\mathbb{P}(\bar{s}|s, a, s') = 1$ whenever either $s = \bar{s}$ or $s' = \bar{s}$, i.e., the BS knows when the MU exited its coverage area.

Rewards: $r(s, a)$ is the expected reward given $s \in \bar{\mathcal{S}}$ and $a \in \mathcal{A}$, defined as the transmission rate (number of bits transmitted from the BS to the MU) during DT if the MU remains within $\hat{\mathcal{S}}$ for T_c subsequent micro time-slots. Formally, $r(s, a) = \sum_{s' \in \bar{\mathcal{S}}} \mathbb{P}(s'|s, a) r(s, a, s') = \sum_{s' \in \bar{\mathcal{S}}} \mathbf{M}_{ss'}^N \mathbb{E}[R(T_{\text{DT}} - 1)\mathcal{X}(E_{s,s'}^N)]$, where $\mathcal{X}(E_{s,s'}^N) = 1$ iff the event $E_{s,s'}^N$ is true (thus $r(s, a) = 0$ if the MU exits from the beam support). Note that $\mathbb{E}[\mathcal{X}(E_{s,s'}^N)] = \mathbb{P}(E_{s,s'}^N)$, which is computed in Eq. (5.12). The transmission rate R follows from Eq. (5.7) when $P_t = P_{\text{DT}}|\hat{\mathcal{S}}|$. Finally, $r(s, a) = R(T_{\text{DT}} - 1) \sum_{s' \in \bar{\mathcal{S}}} \tilde{\mathbf{M}}_{ss'}^N$, where $T_{DT} - 1$ refers to the fact that we reserve one micro time-slot over the total DT duration for the feedback

transmission. If $c = \text{BT}$, then $r(s, a) = 0$, as no bits of data are transmitted.

Costs: $c(s, a)$ is the expected energy cost given $s \in \bar{\mathcal{S}}$ and $a \in \mathcal{A}$. The total expected cost during a transmission episode is subject to the constraint C. If $c = \text{DT}$, then $c(s, a) = P_{\text{DT}}|\hat{\mathcal{S}}|(T_{\text{DT}} - 1)$, $\forall s \in \mathcal{S}$ (we reserve one micro time-slot for the feedback transmission). If $c = \text{BT}$, then $c(s, a) = P_{\text{BT}}|\hat{\mathcal{S}}|$. In this POMDP formulation, the overhead cost given by BT with respect to DT is implicitly modelled with the total fraction of time spent by BT with respect to DT.

5.3 Optimization Problem

Since the agent cannot directly observe the system state, we introduce the notion of *belief*. A belief $b \in \mathcal{B}$ is a probability distribution over $\bar{\mathcal{S}}$. The state estimator must compute a new belief, $b' \in \mathcal{B}$, given an old belief $b \in \mathcal{B}$, an action $a \in \mathcal{A}$, and an observation $o \in \mathcal{O}$, i.e., $b' = f(o, a, b)$. It can be obtained via Bayes' rule as:

$$\begin{aligned} b'(s') &= \mathbb{P}(s' \mid o, a, b) \\ &= \frac{\sum_{s \in \bar{\mathcal{S}}} \mathbb{P}(o|s, a, s')\mathbb{P}(s'|s, a)b(s)}{\mathbb{P}(o|a, b)}. \end{aligned} \tag{5.13}$$

where $\mathbb{P}(o \mid a, b)$ is a normalizing factor, $\sum_{s' \in \bar{\mathcal{S}}} b'(s') = 1$.

Our goal is to determine a policy π (i.e., a map from beliefs to actions) that maximizes the total expected reward the agent can gather, under a constraint on the total expected cost during a transmission episode, following π and starting from $b_0 = b$:

$$\begin{aligned} &\max_{\pi} \mathbb{E}_{\pi}\left[\sum_{t=0}^{\infty} r_t(b_t, \pi(b_t)) \mid b_0 = b\right], \\ &\text{s.t. } \mathbb{E}_{\pi}\left[\sum_{t=0}^{\infty} c_t(b_t, \pi(b_t)) \mid b_0 = b\right] \leqslant C \end{aligned} \tag{5.14}$$

where we have defined expected rate and cost metrics under belief b_t as

$$\begin{aligned} r_t(b_t, \pi(b_t)) &= \sum_{s \in \bar{\mathcal{S}}} r_t(s, \pi(b_t)) b_t(s) \\ c_t(b_t, \pi(b_t)) &= \sum_{s \in \bar{\mathcal{S}}} c_t(s, \pi(b_t)) b_t(s) \,. \end{aligned} \tag{5.15}$$

At this point, we opt for a Lagrangian relaxation approach such that $\mathcal{L}(s,a) = r(s,a) - \lambda c(s,a)$ is the metric to be maximized, for some Lagrangian multiplier $\lambda \geqslant 0$, and the total expected cost during a transmission episode is subject to the constraint C. Hereinafter, according to the notation, $\mathcal{L}_t(b_t, \pi(b_t)) = \sum_{s \in \bar{\mathcal{S}}} \mathcal{L}_t(s, \pi(b_t)) b_t(s)$. At the end of Section 5.3.2, we will consider an online algorithm to optimize parameter λ so as to solve the original problem in Eq. (5.14).

5.3.1 Value Iteration for POMDPs

In POMDPs, a policy π is a function over a continuous set $\mathcal{B}$ of probability distributions over $\bar{\mathcal{S}}$. A policy π is characterized by a value function $V^\pi : \mathcal{B} \to \mathbb{R}$, which is defined as:

$$V^\pi(b) = \mathbb{E}_\pi \left[\sum_{t=0}^{\infty} \mathcal{L}_t(b_t, \pi(b_t)) \mid b_0 = b \right]. \tag{5.16}$$

A policy π that maximizes V^π is called an optimal policy π^*. The value of an optimal policy π^* is the optimal value function V^*, that satisfies the Bellman optimality equation $V^* = HV^*$ (with Bellman backup operator H):

$$V^*(b) = \max_{a \in \mathcal{A}} \left[\sum_{s \in \bar{\mathcal{S}}} b(s) \mathcal{L}(s,a) + \sum_{o \in \mathcal{O}} \mathbb{P}(o|a,b) V^*(b') \right], \tag{5.17}$$

where $b' = f(o,a,b)$ (see Eq. (5.13)). When Eq. (5.17) holds for every belief $b \in \mathcal{B}$ we are ensured the solution is optimal. V^* can be arbitrarily well approximated by iterating over a number of stages, at each stage considering a step further into the future. Also, for problems with an infinite planning horizon, V^* can be approximated, to any degree of accuracy, by a PieceWise Linear and Convex (PWLC) value function [152]. Thus, $V_{n+1} = HV_n$ and we parameterize a value function V_n at stage n by a finite set of vectors (hyperplanes)

$\{\alpha_n^i\}_{i=1}^{|V_n|}$, such that $V_n(b) = \max_{\{\alpha_n^i\}_{i=1}^{|V_n|}} b \cdot \alpha_n^i$, where $(\cdot)$ denotes inner product. Each vector in $\{\alpha_n^i\}_{i=1}^{|V_n|}$, is associated with an action $a(\alpha_n^i) \in \mathcal{A}$, which is the optimal one to take at stage n, and defines a region in the belief space for which this vector is the maximizing element of V_n (thus $\pi(b) = a(\alpha_n^i)$). The key idea is that for a given value function V_n at stage n and a belief $b \in \mathcal{B}$, we can compute the vector α_{n+1}^b in $\{\alpha_{n+1}^i\}_{i=1}^{|HV_n|}$ such that:

$$\alpha_{n+1}^b = \underset{\{\alpha_{n+1}^i\}_i^{|HV_n|}}{\arg\max}\ b \cdot \alpha_{n+1}^i \,, \tag{5.18}$$

where $\{\alpha_{n+1}^i\}_{i=1}^{|HV_n|}$ is the (unknown) set of vectors for HV_n. We will denote this operation $\alpha_{n+1}^b = \text{backup}(b)$. It computes the optimal vector for a given belief $b \in \mathcal{B}$ by back-projecting all vectors in the current horizon value function one step from the future and returning the vector that maximizes the value of $b \in \mathcal{B}$. Defining vectors $\mathcal{L}_a$ such that $\mathcal{L}_a(s) = \mathcal{L}(s, a)$ and $g_{a,o}^i$ such that $g_{a,o}^i(s) = \sum_{s' \in \bar{\mathcal{S}}} \mathbb{P}(o|s, a, s')\mathbb{P}(s'|s, a)\alpha_n^i(s')$ ($g_{a,o}^i$ is a projected vector given action a, observation o, and current horizon vector α_n^i), we have [152]:

$$\begin{aligned} \text{backup}(b) &= \underset{\{g_a^b\}_{a \in \mathcal{A}}}{\arg\max}\, b \cdot g_a^b \\ &= \underset{\{g_a^b\}_{a \in \mathcal{A}}}{\arg\max}\, b \cdot [\mathcal{L}_a + \sum_{o \in \mathcal{O}} \underset{\{g_{a,o}^i\}_i}{\arg\max}\, b \cdot g_{a,o}^i] \,. \end{aligned} \tag{5.19}$$

In general, computing optimal planning solutions for POMDPs is an intractable problem for any reasonably sized task. This calls for approximate solution techniques, e.g., PERSEUS [152], which we introduce next.

5.3.2 Randomized Point-based Value Iteration for POMDPs

PERSEUS is an approximate Point-Based Value Iteration (PBVI) algorithm for POMDPs. It implements a randomized approximate backup operator $\tilde{H}$ that increases (or at least does not decrease) the value of all beliefs $b \in \tilde{\mathcal{B}} \subset \mathcal{B}$. The key idea is that for a given value function V_n at stage n, we can build a value function $V_{n+1} = \tilde{H}V_n$ that improves the value of all beliefs $b \in \tilde{\mathcal{B}} \subset \mathcal{B}$ by only updating the value of a (randomly selected) subset of beliefs $\tilde{\mathcal{B}} \subset \mathcal{B}$, i.e., we can build a value function $V_{n+1} = \tilde{H}V_n$ that upper bounds V_n over $\tilde{\mathcal{B}} \subset \mathcal{B}$ (but not necessarily over $\mathcal{B}$): $V_n(b) \leqslant V_{n+1}(b)$, $\forall b \in \tilde{\mathcal{B}} \subset \mathcal{B}$.

Starting with V_0, PERSEUS performs a number of backup stages until some convergence criterion is met. Each backup stage is defined as in Algorithm 4 (where $\tilde{\mathcal{B}}_{\text{temp}}$ is an auxiliary set containing the non-improved beliefs).

Algorithm 4 function PERSEUS

function PERSEUS($\tilde{\mathcal{B}}$, V_n)
  Set $V_{n+1} = 0$. Initialize $\tilde{\mathcal{B}}_{\text{temp}}$ to $\tilde{\mathcal{B}}$.
  while $\tilde{\mathcal{B}}_{\text{temp}} \neq \varnothing$ **do**
    Sample b from $\tilde{\mathcal{B}}_{\text{temp}}$
    Compute $\alpha = \text{backup}(b)$ (see Eq. (5.19))
    if $b \cdot \alpha \geqslant V_n(b)$ **then**
      Add α to V_{n+1}
    else
      Add $\alpha' = \arg\max_{\{\alpha_n^i\}_i^{|V_n|}} b \cdot \alpha_n^i$ to V_{n+1}
    end if
    Compute set $\tilde{\mathcal{B}}_{\text{temp}} = \{b \in \tilde{\mathcal{B}} : V_{n+1}(b) < V_n(b)\}$
  end while
  return V_{n+1}
end function

Algorithm 5 function BELIEFS

function BELIEFS
  $\tilde{\mathcal{B}} = \varnothing$
  for $w = 1 : W$ **do**
    for $i = 1 : S + 1 - w$ **do**
      Build b such that $b_{i:i+w-1} = 1/w$ and $\tilde{\mathcal{B}} \leftarrow b$
    end for
  end for
  return $\tilde{\mathcal{B}}$
end function

Key to the performance of PERSEUS is the design of $\tilde{\mathcal{B}}$. Several standard schemes to select beliefs have been proposed for PBVI, mainly based on grids of points in the belief space. A different option to select beliefs is to simulate the model, i.e., sampling random actions and observations, and generating trajectories through the belief space, as suggested in [152]. Although this approach may seem reasonable, one may argue that the probability distributions collected in $\tilde{\mathcal{B}}$ are not very representative of the system dynamics history, where actions and observations must also depend on beliefs. Hereinafter, we

leverage the structure of the POMDP presented in Section 5.2 and provide an algorithm (Algorithm 5) to collect beliefs in $\tilde{\mathcal{B}}$ in a smarter fashion. Our approach is simple but effective, and does not require any prior knowledge of the system dynamics: according to Algorithm 5, $\tilde{\mathcal{B}}$ is made of uniform probability distributions over $\bar{\mathcal{S}}$, which are uniformly distributed over at most W *consecutive* road sub-links. Then, this design of $\tilde{\mathcal{B}}$ reflects the compact range of transmission for the BS, where the BS degree of uncertainty on the MU state scales with W.

The basic routine for PBVI is given in Algorithm 6, where $V_{n+1} = V^r_{n+1} - \lambda V^c_{n+1}$ approximates the optimal value function for a given value of λ. Note that we are interested in the optimal policy π^* when b_0 is such that $b_0(s) = \delta(s-1)$, i.e., the agent knows when the MU enters the maximum coverage range of the BS.

Algorithm 6 Point-Based Value Iteration (PBVI)

$\tilde{\mathcal{B}}$ = BELIEFS
Set $n = 0$, $V_0 = 0$, $V^c_0 = 0$, $\lambda_0 = \lambda$, $V_{\text{opt}} = -\infty$.
Define $\mathcal{L}(s,a) = r(s,a) - \lambda_0 c(s,a)$
for $n = 0, \dots$ **do**
    $V_{n+1} = \text{PERSEUS}(\tilde{\mathcal{B}}, V_n)$
    if $|\sum_{b\in\tilde{\mathcal{B}}} V_{n+1}(b) / \sum_{b\in\tilde{\mathcal{B}}} V_n(b) - 1| < \epsilon_V$ **then**
        Break
    end if
end for
$V^*(b_0) = V_{n+1}(b_0)$
$V^*_c(b_0) = V^c_{n+1}(b_0)$
if $V^*_c(b_0) < C$ and $V^*(b_0) > V_{\text{opt}}$ **then**
    $\lambda_{\text{opt}} = \lambda_0$
    $V_{\text{opt}} = V^*(b_0)$
end if

To find the optimal multiplier λ_{opt}, we have to run the routine for different values of λ. PERSEUS performs a number of backup stages until some convergence criterion is met. At this point, we check if the constraint $V^*_c(b_0) < C$ is satisfied, update λ_{opt} if $V^*(b_0) > V_{\text{opt}}$, and repeat the routine for different values of λ. These values of λ can be sequentially selected from a sorted sequence or properly tuned at the end of the routine in a smarter fashion. However, in both cases we have to wait until convergence. To speed up the

Algorithm 7 Point-Based Value Iteration (PBVI) - ONLINE

$\tilde{\mathcal{B}}$ = BELIEFS
Set $n = 0$, $V_0 = 0$, $V_0^c = 0$, $\lambda_0 = \lambda$, $\alpha_0 = \alpha$.
Define $\mathcal{L}(s, a) = r(s, a) - \lambda_0 c(s, a)$
for $n = 0, \ldots$ **do**
    $V_{n+1} = \text{PERSEUS}(\tilde{\mathcal{B}}, V_n)$
    if $|\sum_{b \in \tilde{\mathcal{B}}} V_{n+1}(b) / \sum_{b \in \tilde{\mathcal{B}}} V_n(b) - 1| < \epsilon_V$ **then**
        if $(V_{n+1}^c(b_0) - C)/C < \epsilon_c$ **then**
            Break
        end if
    end if
    $\lambda_{n+1} = \max(0, \lambda_n + \alpha_n (V_{n+1}^c(b_0) - C))$
    Define $\mathcal{L}(s, a) = r(s, a) - \lambda_{n+1} c(s, a)$
end for
$V^*(b_0) = V_{n+1}(b_0)$
$V_c^*(b_0) = V_{n+1}^c(b_0)$
$\lambda_{\text{opt}} = \lambda_n$
$V_{\text{opt}} = V^*(b_0)$

search for the optimal multiplier λ_{opt}, we formulate an online version of Algorithm 6, which is presented in Algorithm 7. Here, λ is properly tuned within the main loop of the routine according to a gradient descent technique [160] [1]: $\lambda_{n+1} = \max(0, \lambda_n + \alpha_n (V_{n+1}^c(b_0) - C))$, where the discount factor is $\alpha_n = \alpha_0/(n+1)$. Finally, given λ_{n+1}, we update the Lagrangian relaxation as $\mathcal{L}(s, a) = r(s, a) - \lambda_{n+1} c(s, a)$. In addition to the convergence criterion of the standard PBVI, we also consider the requirement $(V_{n+1}^c(b_0) - C)/C < \epsilon_c$.

5.4 Numerical results

We set $\Delta_{\text{t}} = 10$ μs. We consider the following parameters: $\Theta = 120^\circ$, $d_0 = 10$ m, $W_{\text{tot}} = 400$ MHz, $f_c = 60$ GHz, $\xi = 1$, $N_0 = -174$ dBm, $S = 10$, $\mathbb{E}[v] = 20$ m/s, $P_{\text{DT}} \in \{10, 20, 30\}$ dBm, $T_{\text{DT}} \in \{1000, 2000, 3000\}$ (number of micro time-slots, i.e., $\{10, 20, 30\}$ ms), $T_{\text{BT}} = 1$, $P_{\text{BT}} = P_{\min}$, where $P_{\min}$ follows from $\mathbb{P}_{\text{FA}} = \mathbb{P}_{\text{MD}} = \epsilon$ (specified below). Finally, we compare different sets $\tilde{\mathcal{B}}$. Let D be the average duration of a transmission episode. The average

[1] Note that a gradient descent technique adjusts the parameter λ after each iteration in the direction that would reduce the error on that iteration the most. The target here depends on the parameter λ, but if we ignore that dependence when we take the derivative, then what we get is a semi-gradient update [161].

rate (bit/s) and power (dBm) are computed as V^r_{n+1}/D and V^c_{n+1}/D.

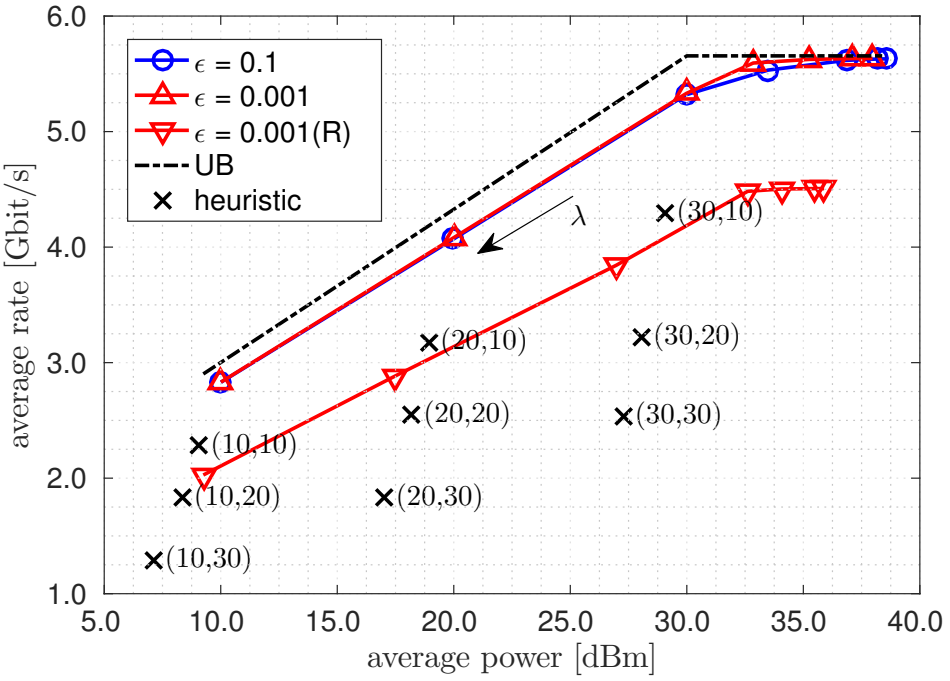


Figure 5.2: Rate and power as a function of λ and comparison with the heuristic π_{H} for different pairs (P_{DT}, T_{DT}) (black crosses) and PBVI with points in $\tilde{\mathcal{B}}$ obtained by random sampling (R).

The average rate and power as a function of $\lambda \in [10^5, 10^{11}]$ are plotted in Fig. 5.2, with $\epsilon = \{0.1, 0.001\}$. As λ increases, both the average rate and power decrease, and the optimal policies are more conservative: given the current belief, the BS performs more directive DT over a smaller set $\hat{\mathcal{S}}$, using smaller values of P_{DT} and T_{DT}. Conversely, as λ decreases, both the average rate and power increse, and the optimal policies are more energy-demanding. Also, as λ decreases, the impact of ϵ on the performance of the optimal policies is slightly more evident: given the average power, we can achieve a larger average rate with a smaller ϵ, i.e., a larger $P_{\min}$, meaning that the optimal policies are more sensitive to the observation errors when performing BT than to the actual value of $P_{\min}$. Overall, the plot suggests that the performance of the optimal policies is quite robust to the observation errors when performing BT, whereas this is not true for any heuristic policy. The heuristic π_{H} works as follows: the BS performs BT with (P_{BT}, T_{BT}) over $\hat{\mathcal{S}} = \{s\}$ (starting from $s = 1$). If the MU replies with ACK, then the BS performs DT with (P_{DT},

T_{DT}) over $\hat{\mathcal{S}}$, until the first NACK. At this point, the BS scans the two states $s-1$ and $s+1$ in two different micro time-slots. If the MU replies with ACK, then the BS repeats the scheme (starting from the new state). The heuristic π_{H} for different pairs (P_{DT}, T_{DT}) (black crosses) cannot achieve the performance of the optimal policies, which take advantage of the belief update mechanism and provide adaptive BT/DT procedures according to the current belief. The average rate and power for π_{H} are plotted in Fig. 5.2 for $\epsilon = 0.001$. The achievable rate drops significantly when considering $\epsilon = 0.1$ (not shown in Fig. 5.2), since π_{H} does not provide any countermeasure to the observation errors when performing BT. An upper bound on the average rate is given when performing DT over the system state (i.e., assuming that the BS knows the position of the MU), thus achieving the maximum transmission rate without wasting power. When the probability distributions collected in $\tilde{\mathcal{B}}$ are not very representative of the system dynamics history, then the performance of the optimal policies can greatly degrade, see Fig. 5.2, where the probability distributions are obtained by sampling random actions and observations, as suggested in [152], and compared to the ones obtained by Algorithm 5 with $W = S$. Here, the total number of beliefs collected in $\tilde{\mathcal{B}}$ is the same in the two cases. However, the performance of the optimal policies with beliefs obtained by Algorithm 5 can provide an improvement in the achievable rate of ~ 1 Gbit/s (1.15 Gbit/s gap using an average power of 35 dBm).

An example of the outcome of Algorithm 7 is given in Fig. 5.3, starting from parameters $\lambda = 0$ and $\alpha = 100$. Here, we set the constraint C (which corresponds to $V_c^*(b_0)$ for $\lambda = 10^5$ in Algorithm 6), and we achieve convergence of the rate to $\bar{R}$, with $\lambda_{\mathrm{opt}} \simeq 10^5$. The average rate and power as a function of n are plotted in Fig. 5.3, with $\bar{R}/D$ and C/D.

5.5 Conclusions

In this chapter, learning tools are investigated to optimize the trade-off between directive Data Transmission (DT) and directional Beam Training (BT) in a mm-wave vehicular network. In the proposed scenario, a Base Station (BS) has to face the issue of beam alignment/realignment towards a Mobile User (MU) moving along a road and, furthermore, has to decide the trans-

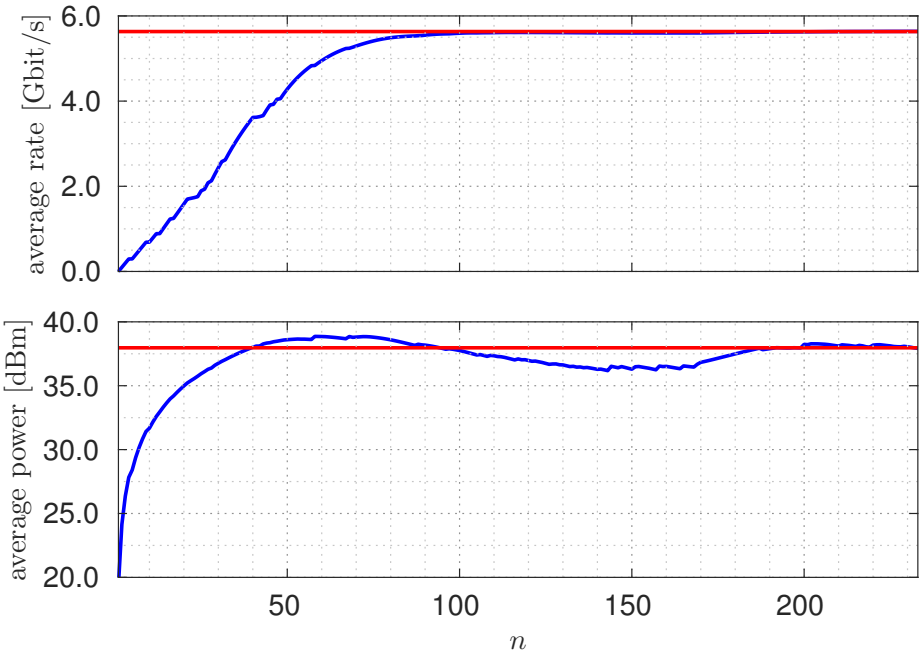


Figure 5.3: Rate and power as a function of λ, which is properly tuned according to $\lambda_{n+1} = \max(0, \lambda_n + \alpha_n(V^c_{n+1}(b_0) - C))$.

mission parameters to use in the DT phase. This leads to a complex problem involving the estimation of the position of the MU within the road link, which must be leveraged in the design of BT/DT protocols. Numerical results reveal that adaptive BT/DT procedures do provide a major advantage over common-sense heuristic schemes, being able to maximize the transmission rate while showing robustness against BT errors, under a power constraint.

Chapter 6

Conclusions

The goal of this thesis is to investigate the considerable potential of inference from insightful perspectives, detailing the mathematical framework and how Bayesian reasoning conveniently adapts to various research domains in wireless networks.

In Chapter 2, we have designed and evaluated a Machine Learning framework to automatically perform protocol layer analysis and diagnose physical layer issues in 60 GHz networks. The main challenge lies in the variability of the channel traces and in the complexity of identifying their structural elements given their noisiness, aperiodicity, and unpredictable behavior. Standard Machine Learning approaches fail in such a scenario. Our tool uses narrowband physical layer energy traces from one or multiple sniffers to identify lower-layer performance issues. Network planners, administrators, as well as researchers can use it to improve the performance of millimeter-wave networks even though off-the-shelf networking devices provide very limited access to lower-layer information. The developed approach provides a convenient trade-off between the efficient but limited monitoring capabilities of IEEE 802.11ad devices in monitor mode, and the highly detailed but costly decoding of full bandwidth signals through software-defined radios. Our algorithms are tested through an extensive measurement campaign using off-the-shelf 60 GHz hardware, considering a range of practical scenarios. Besides, the ability of the framework to correctly identify protocol sequences (beacon pairs, data and acknowledgment frames) from single and multiple channel sniffers is quantified, looking at the impact of channel noise, energy levels and their distribution across different

sniffers.

In Chapter 3, we have demonstrated that localized Bayesian Networks (BNs) are an efficient and lightweight means to tackle prediction and anomaly detection problems in large vehicular networks. The joint probability distribution between the cause nodes (data utilized for forecasting) and the effect node (data to be predicted at any "current" time, belonging to the target road link) is described through a Gaussian Mixture Model (GMM) whose parameters are estimated via Bayesian Variational Inference (BVI) operating on unlabeled data. Optimal forecasting follows from the criterion of Minimum Mean Square Error (MMSE). Moreover, we also perform anomaly detection by devising a probabilistic score associated with the marginal conditional distribution of the effect node. The key features of the model are: i) the approach is scalable as a BN is associated with and independently trained for each road, ii) spatio-temporal information is considered (for increased robustness and accuracy of the statistical model so obtained) and iii) the localized nature of the framework allows flagging atypical behaviors at their point of origin in the monitored geographical map. Also, our framework is capable of operating in realtime and, in turn, it appears to be a promising candidate to deal with Internet of Things (IoT) applications in large-scale networks, where new data is to be processed on-the-fly.

In Chapter 4, we have considered small cell deployments where Energy Harvesting (EH) and packet power networks are combined to provide energy self-sustainability through the use of own-generated energy and carefully planned power transfers among network elements. This amounts to a combined learning and optimization problem (resource scheduling), where learning is carried out on energy arrival (harvested ambient energy) and traffic load traces and this knowledge is exploited, at runtime, for the computation of optimal energy transfer policies among the distributed energy buffers. This foresighted optimization is performed combining Model Predictive Control (MPC) and convex optimization techniques. Numerical results reveal great advantages over the case where energy transfer schedules are optimized disregarding future energy and load forecasts: the amount of energy purchased from the power grid is reduced by more than 50% and the outage probability is lowered to zero in nearly all scenarios.

In Chapter 5, learning tools are investigated to optimize the trade-off be-

tween directive Data Transmission (DT) and directional Beam Training (BT) in a mm-wave vehicular network. In the proposed scenario, a Base Station (BS) has to face the issue of beam alignment/realignment towards a Mobile User (MU) moving along a road and, furthermore, has to decide the transmission parameters to use in the DT phase. This leads to a complex problem involving the estimation of the position of the MU within the road link, which must be leveraged in the design of BT/DT protocols. To control the BT/DT trade-off, a Partially Observable (PO) Markov Decision Process (MDP) is formulated, where the system state corresponds to the position of the MU within the road link. Numerical results reveal that adaptive BT/DT procedures do provide a major advantage over common-sense heuristic schemes, being able to maximize the transmission rate while showing robustness against BT errors, under a power constraint. At this point, some future research directions need to be addressed. An automated procedure could be set up to learn the parameters of the POMDP via state-of-the-art reinforcement solutions (where the agent gains experience while interacting with the environment, without prior knowledge of the exact mathematical model). Moreover, deep reinforcement learning architectures can be designed to solve real-time optimization problems based on real-world data, where the position of the MU evolves in a more realistic fashion. Smarter network decision making can be evaluated while taking into account the presence of channel blockage and the highly dynamic behavior of the mm-wave link, which can move in and out of the outage condition on a very short time scale. In this respect, cell selection represents a fundamental functionality to preserve communication, as it involves choosing which BS the MU should be connected to in the event of link obstruction: the MU will track the quality of the mm-wave link and rapidly switch to another BS in response to the fast-varying link state. Finally, a proper mm-wave antenna design could be investigated to precisely quantify the beam steering performance with respect to the idealized antenna model.

List of publications

List of publications on international journals

J1. M. Scalabrin, G. Bielsa, A. Loch, M. Rossi and J. Widmer,"Machine Learning Based Network Analysis using Millimeter-Wave Narrow-Band Energy Traces," submitted to IEEE Transactions on Mobile Computing.

J2. A. Gambin, M. Scalabrin, M. Rossi, "Online Power Management Strategies for Energy Harvesting Mobile Networks," submitted to IEEE Transactions on Green Communications and Networking.

List of publications on conference proceedings

C1. M. Scalabrin, N. Michelusi, M. Rossi, "Beam Training and Data Transmission Optimization in Millimeter-Wave Vehicular Networks," accepted at 2018 IEEE Global Communications Conference (GLOBECOM).

C2. M. Scalabrin, M. Gadaleta, R. Bonetto and M. Rossi, "A Bayesian Forecasting and Anomaly Detection Framework for Vehicular Monitoring Networks," 2017 IEEE International Workshop on Machine Learning for Signal Processing (MLSP), September 25-28, 2017, Roppongi, Tokyo, Japan, pp. 1-6.

C3. M. Scalabrin, M. Rossi, G. Bielsa, A. Loch and J. Widmer, "Millimetric Diagnosis: Machine Learning Based Network Analysis for mm-Wave Communication," 2017 IEEE International Symposium on A World of Wireless, Mobile and Multimedia Networks (WoWMoM), Macau, China, June 12-15, 2017, pp. 1-9.

C4. M. Scalabrin, V. Vadori, A. V. Guglielmi and L. Badia, "A Zero-Sum Jamming Game with Incomplete Position Information in Wireless Scenarios," 2015 IEEE European Wireless Conference, Budapest, Hungary, May 20-22, 2015, pp. 1-6.

C5. V. Vadori, M. Scalabrin, A. V. Guglielmi and L. Badia, "Jamming in Underwater Sensor Networks as a Bayesian Zero-Sum Game with Position

Uncertainty," 2015 IEEE Global Communications Conference (GLOBECOM), San Diego, CA, USA, December 6-10, 2015, pp. 1-6.

C6. L. Badia and M. Scalabrin, "Stochastic Analysis of Delay Statistics for Intermittently Connected Vehicular Networks," 2014 IEEE European Wireless Conference, Barcelona, Spain, May 14-16, 2014, pp. 1-6.

Bibliography

[1] S. Rangan, T. S. Rappaport, and E. Erkip, "Millimeter-Wave Cellular Wireless Networks: Potentials and Challenges," *Proc. of the IEEE*, vol. 102, no. 3, pp. 366–385, 2014.

[2] DARPA, "News." `https://www.darpa.mil/news-events/2017-08-11a`. Accessed: 2018-08-30.

[3] DARPA, "News." `https://www.darpa.mil/news-events/2016-07-19a`. Accessed: 2018-08-30.

[4] NSF, "Homepage." `http://www.nsf.org/`. Accessed: 2018-08-30.

[5] R. R. Choudhury and N. H. Vaidya, "Deafness: a MAC Problem in Ad Hoc Networks when using Directional Antennas," in *Proc. of IEEE ICNP*, (Berlin, Germany), Oct 2004.

[6] T. Nitsche, G. Bielsa, I. Tejado, A. Loch, and J. Widmer, "Boon and Bane of 60 GHz Networks: Practical Insights into Beamforming, Interference, and Frame Level Operation," in *Proc. of ACM CoNEXT*, (Heidelberg, Germany), Dec 2015.

[7] Y. Zhu, Z. Zhang, Z. Marzi, C. Nelson, U. Madhow, B. Y. Zhao, and H. Zheng, "Demystifying 60GHz Outdoor Picocells," in *Proc. of ACM MobiCom*, (Maui, Hawaii, USA), Sep 2014.

[8] S. K. Saha and D. Koutsonikolas, "Towards Multi-gigabit 60 GHz Indoor WLANs," in *Proc. of IEEE ICNP*, (San Francisco, CA, USA), Nov 2015.

[9] IEEE, "Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment 3: Enhancements for Very High Throughput in the 60 GHz Band," *IEEE Std 802.11ad-2012*, 2012.

[10] M. K. Haider and E. W. Knightly, "Mobility Resilience and Overhead Constrained Adaptation in Directional 60 GHz WLANs: Protocol Design and System Implementation," in *Proc. of ACM MobiHoc*, (Paderborn, Germany), Jul 2016.

[11] S. Sur, X. Zhang, P. Ramanathan, and R. Chandra, "BeamSpy: Enabling Robust 60 GHz Links Under Blockage," in *Proc. of USENIX NSDI*, 2016.

[12] S. Sur, V. Venkateswaran, X. Zhang, and P. Ramanathan, "60 GHz Indoor Networking Through Flexible Beams: A Link-Level Profiling," in *Proc. of ACM SIGMETRICS*, (Portland, Oregon, USA), Jun 2015.

[13] R. do Carmo and M. Hollick, "DogoIDS: A Mobile and Active Intrusion Detection System for IEEE 802.11s Wireless Mesh Networks," in *Proc. of ACM HotWiSec*, (Budapest, Hungary), Apr 2013.

[14] D. S. Berger, F. Gringoli, N. Facchi, I. Martinovic, and J. Schmitt, "Gaining Insight on Friendly Jamming in a Real-world IEEE 802.11 Network," in *Proc. of ACM WiSec*, (Oxford, UK), Jul 2014.

[15] T. T. T. Nguyen and G. Armitage, "A Survey of Techniques for Internet Traffic Classification Using Machine Learning," *IEEE Communications Surveys Tutorials*, vol. 10, no. 4, pp. 56–76, 2008.

[16] A. McGregor, M. Hall, P. Lorier, and J. Brunskill, "Flow Clustering Using Machine Learning Techniques," in *Passive and Active Network Measurement (PAM)*, 2004.

[17] J. Mitola and G. Q. Maguire, "Cognitive Radio: Making Software Radios More Personal," *IEEE Personal Communications*, vol. 6, no. 4, pp. 13–18, 1999.

[18] S. Haykin, "Cognitive Radio: Brain-Empowered Wireless Communications," *IEEE Journal on Selected Areas in Communications*, vol. 23, no. 2, pp. 201–220, 2005.

[19] DARPA, "Homepage." `https://www.darpa.mil/`. Accessed: 2018-08-30.

[20] M. G. D. Benedetto, S. Boldrini, C. J. M. Martin, and J. R. Diaz, "Automatic Network Recognition by Feature Extraction: A Case Study in the ISM Band," in *Proc. of IEEE CROWNCOM*, (Cannes, France), Jun 2010.

[21] M. Gandetto and C. Regazzoni, "Spectrum Sensing: A Distributed Approach for Cognitive Terminals," *IEEE JSAC*, vol. 25, no. 3, pp. 546–557, 2007.

[22] P. Varshney, *Distributed Detection and Data Fusion*. Springer New York, 2012.

[23] M. Gandetto, M. Guainazzo, and C. S. Regazzoni, "Use of Time-frequency Analysis and Neural Networks for Mode Identification in a Wire-less Software-defined Radio Approach," *EURASIP JASP*, pp. 1778–1790, 2004.

[24] H. Assasa and J. Widmer, "Implementation and Evaluation of a WLAN IEEE 802.11ad Model in ns-3," in *Proc. of the Workshop on ns-3*, (Seattle, WA, USA), Jun 2016.

[25] D. Steinmetzer, D. Wegemer, M. Schulz, J. Widmer, and M. Hollick, "Compressive Millimeter-Wave Sector Selection in Off-the-Shelf IEEE 802.11ad Devices," in *Proc of. ACM CoNEXT*, (Incheon, Republic of Korea), Dec 2017.

[26] L. R. Rabiner, "A tutorial on hidden markov models and selected applications in speech recognition," *Proc. of the IEEE*, vol. 77, no. 2, pp. 257–286, 1989.

[27] Y. Niu, Y. Li, D. Jin, L. Su, and A. V. Vasilakos, "A Survey of Millimeter Wave Communications (mmWave) for 5G: Opportunities and Challenges," *Wireless Networks*, vol. 21, no. 8, pp. 2657–2676, 2015.

[28] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum Likelihood From Incomplete Data via the EM Algorithm," *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 39, no. 1, pp. 1–38, 1977.

[29] L. E. Baum, T. Petrie, G. Soules, and N. Weiss, "A Maximization Technique Occurring in the Statistical Analysis of Probabilistic Functions of Markov Chains," *The Annals of Mathematical Statistics*, vol. 41, no. 1, pp. 164–171, 1970.

[30] L. E. Baum and J. A. Eagon, "An Inequality With Applications to Statistical Estimation for Probabilistic Functions of Markov Processes and to a Model for Ecology," *Bulletin of the American Mathematical Society*, vol. 73, no. 3, pp. 360–363, 1967.

[31] L. E. Baum and G. R. Sell, "Growth Transformations for Functions on Manifolds," *Pacific Journal of Mathematics*, vol. 27, no. 2, pp. 211–227, 1968.

[32] S.-Z. Yu, "Hidden Semi-Markov Models," *Artificial Intelligence*, vol. 174, no. 2, pp. 215–243, 2010.

[33] S.-Z. Yu and H. Kobayashi, "An Efficient Forward-Backward Algorithm for an Explicit-Duration Hidden Markov Model," *IEEE Signal Processing Letters*, vol. 10, no. 1, pp. 11–14, 2003.

[34] S.-Z. Yu and H. Kobayashi, "Practical Implementation of an Efficient Forward-Backward Algorithm for an Explicit-Duration Hidden Markov Model," *IEEE Trans. on Signal Processing*, vol. 54, no. 5, pp. 1947–1951, 2006.

[35] A. V. Oppenheim and G. C. Verghese, *Signals, Systems and Interference*. Pearson, 2015.

[36] D. Garcia, "Robust Smoothing of Gridded Data in One and Higher Dimensions with Missing Values," *Computational Statistics & Data Analysis*, vol. 54, no. 4, pp. 1167–1178, 2010.

[37] D. Garcia, "A Fast All-in-One Method for Automated Post-Processing of PIV Data," *Experiments in Fluids*, vol. 50, no. 5, pp. 1247–1259, 2011.

[38] J. Lee Rodgers and W. A. Nicewander, "Thirteen Ways to Look at the Correlation Coefficient," *The American Statistician*, vol. 42, no. 1, pp. 59–66, 1988.

[39] T. Kailath and H. V. Poor, "Detection of Stochastic Processes," *IEEE Trans. on Information Theory*, vol. 44, Oct 1988.

[40] A. Mueen, H. Hamooni, and T. Estrada, "Time Series Join on Subsequence Correlation," in *Proc. of IEEE ICDM*, (Shenzhen, China), Dec 2014.

[41] A. Mueen, E. Keogh, and N. Young, "Logical-shapelets: an expressive primitive for time series classification," in *Proc. of ACM SIGKDD*, (San Diego, CA, USA), Aug 2011.

[42] IEEE, "Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment 3: Enhancements for Very High Throughput in the 60 GHz Band," *IEEE Std 802.11ad-2012*, Dec 2012.

[43] C. Bishop, *Pattern Recognition and Machine Learning*. Springer, 2007.

[44] S. T. Roweis, "Constrained Hidden Markov Models," in *Proc. of NIPS*, (Denver, Colorado, USA), Nov 1999.

[45] J. MacQueen, "Some Methods for Classification and Analysis of Multivariate Observations," in *Proc. of BSMSP*, 1967.

[46] D. Arthur and S. Vassilvitskii, "K-means++: The Advantages of Careful Seeding," in *Proc. of ACM-SIAM SODA*, (New Orleans, Louisiana, USA), Jan 2007.

[47] S. E. Levinson, "Continuously Variable Duration Hidden Markov Models for Automatic Speech Recognition," *Computer Speech & Language*, vol. 1, no. 1, pp. 29–45, 1986.

[48] C. Mitchell and L. Jamieson, "Modeling Duration in a Hidden Markov Model with the Exponential Family," in *Proc. of IEEE ICASSP*, (Minneapolis, MN, USA), Apr 1993.

[49] R. Tibshirani, G. Walther, and T. Hastie, "Estimating the Number of Clusters in a Data Set via the Gap Statistic," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 63, no. 2, pp. 411–423, 2001.

[50] A. Zanella, "Best Practice in RSS Measurements and Ranging," *IEEE Communications Surveys and Tutorials*, vol. 18, no. 4, pp. 2662–2686, 2016.

[51] C. Dance, "Lean Smart Parking," *The Parking Professional*, vol. 30, no. 6, pp. 26–29, 2014.

[52] G. Pierce and D. Shoup, "Getting the Prices Right," *Journal of the American Planning Association*, vol. 79, no. 1, pp. 67–81, 2013.

[53] Worldsensing, "Smartprk – Making Smart Cities Happen." `http://www.fastprk.com/`.

[54] N. Piovesan, L. Turi, E. Toigo, B. Martinez, and M. Rossi, "Data Analytics for Smart Parking Applications," *Sensors*, vol. 16, no. 10, 2016.

[55] E. I. Vlahogiannia, K. Kepaptsogloua, V. Tsetsosa, and M. G. Karlaftisa, "A Real-Time Parking Prediction System for Smart Cities," *Journal of Intelligent Transportation Systems: Technology, Planning, and Operations*, vol. 20, no. 2, pp. 192–204, 2016.

[56] A. Kanungo, A. Sharma, and C. Singla, "Smart traffic lights switching and traffic density calculation using video processing," in *IEEE Recent Advances in Engineering and Computational Sciences (RAECS)*, 2014.

[57] B. Ghazal, K. ElKhatib, K. Chahine, and M. Kherfan, "Smart traffic light control system," in *International Conference on Electrical, Electronics, Computer Engineering and their Applications (EECEA)*, 2016.

[58] J. Rzeszótko and S. H. Nguyen, "Machine Learning for Traffic Prediction," *Fundamenta Informaticae - Concurrency Specification and Programming*, vol. 119, no. 3-4, pp. 407–420, 2012.

[59] Z. A. Bakar, R. Mohemad, A. Ahmad, and M. M. Deris, "A Comparative Study for Outlier Detection Techniques in Data Mining," in *Cybernetics and Intelligent Systems, 2006 IEEE Conference on*, 2006.

[60] V. J. Hodge and J. Austin, "A Survey of Outlier Detection Methodologies," *Artificial intelligence review*, vol. 22, no. 2, pp. 85–126, 2004.

[61] M. Agyemang, K. Barker, and R. Alhajj, "A Comprehensive Survey of Numeric and Symbolic Outlier Mining Techniques," *Intelligent Data Analysis*, vol. 10, no. 6, pp. 521–538, 2006.

[62] J. Lan, C. Long, R. C.-W. Wong, Y. Chen, Y. Fu, D. Guo, S. Liu, Y. Ge, Y. Zhou, and J. Li, "A New Framework for Traffic Anomaly Detection," in *Proceedings of the 2014 SIAM International Conference on Data Mining*, 2014.

[63] W. Liu, Y. Zheng, S. Chawla, J. Yuan, and X. Xing, "Discovering Spatio-temporal Causal Interactions in Traffic Data Streams," in *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2011.

[64] L. X. Pang, S. Chawla, W. Liu, and Y. Zheng, "On Mining Anomalous Patterns in Road Traffic Streams," in *International Conference on Advanced Data Mining and Applications*, 2011.

[65] X. Li, Z. Li, J. Han, and J. G. Lee, "Temporal Outlier Detection in Vehicle Traffic Data," in *2009 IEEE 25th International Conference on Data Engineering*, 2009.

[66] S. Sun, C. Zhang, and G. Yu, "A Bayesian Network Approach to Traffic Flow Forecasting," *IEEE Transactions on Intelligent Transportation Systems*, vol. 7, no. 1, pp. 124–132, 2006.

[67] H. Yin, S. Wong, J. Xu, and C. Wong, "Urban Traffic Flow Prediction Using a Fuzzy-Neural Approach," *Transportation Research Part C: Emerging Technologies*, vol. 10, no. 2, pp. 85–98, 2002.

[68] G. Yu, J. Hu, C. Zhang, L. Zhuang, and J. Song, "Short-Term Traffic Flow Forecasting Based on Markov Chain Model," in *IEEE IV2003 Intelligent Vehicles Symposium. Proceedings (Cat. No.03TH8683)*, 2003.

[69] C. M. Bishop, *Pattern recognition and machine learning*. Springer, New York, 2006.

[70] *Bitcarrier technology.*

[71] C. R. Rao, *Linear statistical inference and its applications*. John Wiley & Sons, 2009.

[72] A. S. Andrae and T. Edler, "On global electricity usage of communication technology: trends to 2030," *Challenges*, vol. 6, pp. 117–157, April 2015.

[73] D. Zordan, M. Miozzo, P. Dini, and M. Rossi, "When telecommunications networks meet energy grids: cellular networks with energy harvesting and trading capabilities," *IEEE Communications Magazine*, vol. 53, pp. 117–123, June 2015.

[74] H. Sugiyama, "Packet switched power network with decentralized control based on synchronized QoS routing," *ICT&Applications and Collocated Events*, pp. 147–152, Nov 2012.

[75] L. Bonati, A. F. Gambin, and M. Rossi, "Wireless Power Transfer under the Spotlight: Charging Terminals amid Dense Cellular Networks," in *IEEE International Symposium on a World of Wireless Mobile and Multimedia Networks (WoWMoM)*, (Macao, China), IEEE, Jun 2017.

[76] V. Krylov, D. Ponomarev, and A. Loskutov, "Toward the power InterGrid," in *IEEE International Energy Conference and Exhibition (ENERGYCON)*, (Manama, Bahrain), pp. 351–356, IEEE, May 2010.

[77] J. Ma, L. Song, and Y. Li, "Optimal power dispatching for local area packetized power network," *IEEE Transactions on Smart Grid*, 2017.

[78] A. F. Gambin and M. Rossi, "Energy Cooperation for Sustainable Base Station Deployments: Principles and Algorithms," in *IEEE Global Communications Conference (GLOBECOM)*, (Singapore, Singapore), IEEE, Dec 2017.

[79] Y.-K. Chia, S. Sun, and R. Zhang, "Energy cooperation in cellular networks with renewable powered base stations," *IEEE Transactions on Wireless Communications*, vol. 13, no. 12, pp. 6996–7010, 2014.

[80] B. Gurakan, O. Ozel, J. Yang, and S. Ulukus, "Energy cooperation in energy harvesting communications," *IEEE Transactions on Communications*, vol. 61, pp. 4884–4898, Nov 2013.

[81] J. Xu and R. Zhang, "CoMP meets smart grid: A new communication and energy cooperation paradigm," *IEEE Transactions on Vehicular Technology*, vol. 64, pp. 2476–2488, Aug 2015.

[82] J. Leithon, T. J. Lim, and S. Sun, "Energy exchange among base stations in a cellular network through the smart grid," in *IEEE International Conference on Communications (ICC)*, pp. 4036–4041, IEEE, 2014.

[83] M. J. Farooq, H. Ghazzai, A. Kadri, H. ElSawy, and M.-S. Alouini, "A hybrid energy sharing framework for green cellular networks," *IEEE Transactions on Communications*, vol. 65, no. 2, pp. 918–934, 2017.

[84] X. Huang, T. Han, and N. Ansari, "Smart grid enabled mobile networks: Jointly optimizing BS operation and power distribution," *IEEE/ACM Transactions on Networking*, 2017.

[85] T. Han and N. Ansari, "On optimizing green energy utilization for cellular networks with hybrid energy supplies," *IEEE Transactions on Wireless Communications*, vol. 12, no. 8, pp. 3872–3882, 2013.

[86] Z. Guo, T. J. Lim, and M. Motani, "Base station energy cooperation in green cellular networks," in *Global Conference on Signal and Information Processing (GlobalSIP), 2013 IEEE*, pp. 349–352, IEEE, 2013.

[87] M. J. Farooq, H. Ghazzai, A. Kadri, H. ElSawy, and M.-S. Alouini, "Energy sharing framework for microgrid-powered cellular base stations," in *Global Communications Conference (GLOBECOM), 2016 IEEE*, pp. 1–7, IEEE, 2016.

[88] J. Xu, L. Duan, and R. Zhang, "Cost-aware green cellular networks with energy and communication cooperation," *IEEE Communications Magazine*, vol. 53, no. 5, pp. 257–263, 2015.

[89] X. Wang, Y. Zhang, G. B. Giannakis, and S. Hu, "Robust smart-grid-powered cooperative multipoint systems," *IEEE Transactions on Wireless Communications*, vol. 14, no. 11, pp. 6188–6199, 2015.

[90] J. Xu, L. Duan, and R. Zhang, "Energy group buying with loading sharing for green cellular networks," *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 4, pp. 786–799, 2016.

[91] B. Gurakan, O. Ozel, and S. Ulukus, "Optimal energy and data routing in networks with energy cooperation," *IEEE Transactions on Wireless Communications*, vol. 15, no. 2, pp. 857–870, 2016.

[92] J. B. Rawlings and D. Q. Mayne, *Model predictive control: Theory and design*. Nob Hill Pub., 2009.

[93] Y. Fang and A. Armaou, "Nonlinear model predictive control using a bilinear Carleman linearization-based formulation for chemical processes," in *American Control Conference (ACC), 2015*, pp. 5629–5634, IEEE, 2015.

[94] J. W. Eaton and J. B. Rawlings, "Model predictive control of chemical processes," in *American Control Conference, 1991*, pp. 1790–1795, IEEE, 1991.

[95] M. Arefi, A. Montazeri, J. Poshtan, and M. Jahed-Motlagh, "Nonlinear model predictive control of chemical processes with a Wiener identification approach," in *IEEE International Conference on Industrial Technology*, pp. 1735–1740, IEEE, 2006.

[96] K. Nejadkazemi and A. Fakharian, "Pressure control in gas oil pipeline: A supervisory model predictive control approach," in *4th International Conference on Control, Instrumentation, and Automation (ICCIA)*, pp. 396–400, IEEE, 2016.

[97] A. Pavlov, D. Krishnamoorthy, K. Fjalestad, E. Aske, and M. Fredriksen, "Modelling and model predictive control of oil wells with electric submersible pumps," in *IEEE Conference on Control Applications (CCA)*, pp. 586–592, IEEE, 2014.

[98] R. Halvgaard, L. Vandenberghe, N. K. Poulsen, H. Madsen, and J. B. Jørgensen, "Distributed model predictive control for smart energy systems," *IEEE Transactions on Smart Grid*, vol. 7, no. 3, pp. 1675–1682, 2016.

[99] P. Stadler, A. Ashouri, and F. Maréchal, "Distributed model predictive control of energy systems in microgrids," in *Annual IEEE Systems Conference (SysCon)*, pp. 1–6, IEEE, 2016.

[100] K. Meng, Z. Y. Dong, Z. Xu, and S. R. Weller, "Cooperation-driven distributed model predictive control for energy storage systems," *IEEE Transactions on Smart Grid*, vol. 6, no. 6, pp. 2583–2585, 2015.

[101] S. G. Tzafestas, P. Borne, and L. Grandinetti, *Parallel and Distributed Computing in Engineering Systems: Proceedings of the IMACS/IFAC International Symposium on Parallel and Distributed Computing in Engineering Systems, Corfu, Greece, 23-28 June 1991.* North Holland, 1992.

[102] E. Perea-Lopez, B. E. Ydstie, and I. E. Grossmann, "A model predictive control strategy for supply chain optimization," *Computers & Chemical Engineering*, vol. 27, no. 8, pp. 1201–1218, 2003.

[103] M. W. Braun, D. E. Rivera, M. Flores, W. M. Carlyle, and K. G. Kempf, "A model predictive control framework for robust management of multi-product, multi-echelon demand networks," *Annual Reviews in Control*, vol. 27, no. 2, pp. 229–245, 2003.

[104] P.-H. Lin, S.-S. Jang, and D. S.-H. Wong, "Predictive control of a decentralized supply chain unit," *Industrial & engineering chemistry research*, vol. 44, no. 24, pp. 9120–9128, 2005.

[105] P. Doganis, E. Aggelogiannaki, and H. Sarimveis, "A combined model predictive control and time series forecasting framework for production-inventory systems," *International Journal of Production Research*, vol. 46, no. 24, pp. 6841–6853, 2008.

[106] G. Box and G. Jenkins, "(1970). time series analysis; forecasting and control. holden-day, san francisco(ca),"

[107] G. P. Zhang and M. Qi, "Neural network forecasting for seasonal and trend time series," *European Journal of Operational Research*, vol. 160, no. 2, pp. 501–514, 2005.

[108] G. P. Zhang, "Time series forecasting using a hybrid arima and neural network model," *Neurocomputing*, vol. 50, pp. 159–175, 2003.

[109] I. Khandelwal, R. Adhikari, and G. Verma, "Time series forecasting using hybrid arima and ann models based on dwt decomposition," *Procedia Computer Science*, vol. 48, pp. 173–179, 2015.

[110] D. J. MacKay, "Gaussian processes-a replacement for supervised neural networks?," 1997.

[111] C. K. Williams, "Prediction with Gaussian processes: From linear regression to linear prediction and beyond," *NATO ASI. Series D, Behavioural and Social Sciences*, vol. 89, pp. 599–621, 1998.

[112] D. J. Leith, M. Heidl, and J. V. Ringwood, "Gaussian process prior models for electrical load forecasting," in *Probabilistic Methods Applied to Power Systems, 2004 International Conference on*, pp. 112–117, IEEE, 2004.

[113] M. Blum and M. Riedmiller, "Electricity demand forecasting using gaussian processes," *Power*, vol. 10, p. 104, 2013.

[114] J. W. Taylor, "Short-term electricity demand forecasting using double seasonal exponential smoothing," *Journal of the Operational Research Society*, vol. 54, no. 8, pp. 799–805, 2003.

[115] J. W. Taylor, L. M. De Menezes, and P. E. McSharry, "A comparison of univariate methods for forecasting electricity demand up to a day ahead," *International Journal of Forecasting*, vol. 22, no. 1, pp. 1–16, 2006.

[116] W. Ketter, J. Collins, and P. Reddy, "Power tac: A competitive economic simulation of the smart grid," *Energy Economics*, vol. 39, pp. 262–270, 2013.

[117] J. Taylor, "Short-Term Electricity Demand Forecasting Using Double Seasonal Exponential Smoothing," *The Journal of the Operational Research Society (JSTOR)*, vol. 54, pp. 799–805, Aug 2003.

[118] J. Kocijan, R. Murray-Smith, C. E. Rasmussen, and B. Likar, "Predictive control with Gaussian process models," in *EUROCON 2003. Computer as a Tool. The IEEE Region 8*, vol. 1, pp. 352–356, IEEE, 2003.

[119] A. Pawlowski, J. L. Guzmán, F. Rodríguez, M. Berenguel, and J. E. Normey-Rico, "Predictive control with disturbance forecasting for greenhouse diurnal temperature control," *IFAC Proceedings Volumes*, vol. 44, no. 1, pp. 1779–1784, 2011.

[120] B. Likar and J. Kocijan, "Predictive control of a gas–liquid separation plant based on a Gaussian process model," *Computers & Chemical Engineering*, vol. 31, no. 3, pp. 142–152, 2007.

[121] A. Grancharova, J. Kocijan, and T. A. Johansen, "Explicit stochastic predictive control of combustion plants based on Gaussian process models," *Automatica*, vol. 44, no. 6, pp. 1621–1631, 2008.

[122] R. Palm, "Multiple-step-ahead prediction in control systems with Gaussian process models and TS-fuzzy models," *Engineering Applications of Artificial Intelligence*, vol. 20, no. 8, pp. 1023–1035, 2007.

[123] J. Ko, D. J. Klein, D. Fox, and D. Haehnel, "Gaussian processes and reinforcement learning for identification and control of an autonomous blimp," in *Robotics and Automation, 2007 IEEE International Conference on*, pp. 742–747, IEEE, 2007.

[124] J. M. Maciejowski and X. Yang, "Fault tolerant control using Gaussian processes and model predictive control," in *Control and Fault-Tolerant Systems (SysTol), 2013 Conference on*, pp. 1–12, IEEE, 2013.

[125] Y. Wang, C. Ocampo-Martínez, V. Puig, and J. Quevedo, "Gaussian-process-based demand forecasting for predictive control of drinking water networks," in *International Conference on Critical Information Infrastructures Security*, pp. 69–80, Springer, 2014.

[126] A. Von Meier, *Electric power systems: a conceptual introduction*. John Wiley & Sons, 2006.

[127] M. Miozzo, D. Zordan, P. Dini, and M. Rossi, "SolarStat: Modeling photovoltaic sources through stochastic markov processes," in *IEEE International Energy Conference (ENERGYCON)*, (Cavtat, Croatia), pp. 688–695, IEEE, May 2014.

[128] T. I. (TIM), "Open Big Data Challenge," 2015.

[129] M. Ester, H.-P. Kriegel, J. Sander, X. Xu, *et al.*, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *KDD*, vol. 96, pp. 226–231, 1996.

[130] G. Auer, O. Blume, V. Giannini, I. Godor, M. Imran, Y. Jading, E. Katranaras, M. Olsson, D. Sabella, P. Skillermark, *et al.*, "D2. 3: Energy efficiency analysis of the reference systems, areas of improvements and target breakdown," *EARTH*, pp. 1–69, Dec 2010.

[131] A. Biason and M. Zorzi, "On the Effects of Battery Imperfections in an Energy Harvesting Device," in *2016 International Conference on Computing, Networking and Communications*, (Hawaii, USA), IEEE, Feb 2016.

[132] C. E. Rasmussen and C. K. Williams, *Gaussian processes for machine learning*, vol. 1. MIT press Cambridge, 2006.

[133] R. Murray-Smith, T. A. Johansen, and R. Shorten, "On transient dynamics, off-equilibrium behaviour and identification in blended multiple model structures," in *Control Conference (ECC), 1999 European*, pp. 3569–3574, IEEE, 1999.

[134] D. Leith, R. Murray-Smith, and W. Leithead, "Nonlinear structure identification: A Gaussian Process prior/Velocity-based approach," 2000.

[135] E. Solak, R. Murray-Smith, W. E. Leithead, D. J. Leith, and C. E. Rasmussen, "Derivative observations in Gaussian process models of dynamic systems," in *Advances in neural information processing systems*, pp. 1057–1064, 2003.

[136] D. Duvenaud, J. R. Lloyd, R. Grosse, J. B. Tenenbaum, and Z. Ghahramani, "Structure discovery in nonparametric regression through compositional kernel search," *arXiv preprint arXiv:1302.4922*, 2013.

[137] C. E. R. . H. Nickisch, "The GPML Toolbox version 4.0." `http://http://www.gaussianprocess.org/gpml/code/matlab/doc/manual.pdf`.

[138] Y. Wang, C. Ocampo-Martinez, and V. Puig, "Stochastic model predictive control based on Gaussian processes applied to drinking water networks," *IET Control Theory & Applications*, vol. 10, no. 8, pp. 947–955, 2016.

[139] M. Grant, S. Boyd, and Y. Ye, "CVX: Matlab software for disciplined convex programming," 2008.

[140] T. Zhang, Y. Zhang, H. Lei, B. Guo, and Y. Zha, "Optimal microgrid operation based on model predictive control framework," in *3rd International Conference on Control, Automation and Robotics (ICCAR)*, (Nagoya, Japan), pp. 575–581, IEEE, Apr 2017.

[141] H. W. Kuhn, "The Hungarian method for the assignment problem," *Naval research logistics quarterly*, vol. 2, pp. 83–97, Mar 1955.

[142] J. Choi, V. Va, N. Gonzalez-Prelcic, R. Daniels, C. R. Bhat, and R. W. Heath, "Millimeter-wave vehicular communication to support massive automotive sensing," *IEEE Communications Magazine*, vol. 54, no. 12, pp. 160–167, 2016.

[143] M. Giordani, A. Zanella, and M. Zorzi, "Millimeter wave communication in vehicular networks: Challenges and opportunities," in *Modern Circuits and Systems Technologies (MOCAST), 2017 6th International Conference on*, pp. 1–6, IEEE, 2017.

[144] V. Va, J. Choi, T. Shimizu, G. Bansal, and R. W. Heath, "Inverse multipath fingerprinting for millimeter wave v2i beam alignment," *IEEE Transactions on Vehicular Technology*, 2017.

[145] J. Palacios, D. De Donno, and J. Widmer, "Tracking mm-wave channel dynamics: Fast beam training strategies under mobility," in *INFOCOM 2017-IEEE Conference on Computer Communications, IEEE*, pp. 1–9, IEEE, 2017.

[146] M. Giordani, M. Mezzavilla, and M. Zorzi, "Initial access in 5g mmwave cellular networks," *IEEE Communications Magazine*, vol. 54, pp. 40–47, November 2016.

[147] V. Va, X. Zhang, and R. W. Heath, "Beam switching for millimeter wave communication to support high speed trains," in *Vehicular Technology Conference (VTC Fall), 2015 IEEE 82nd*, pp. 1–5, IEEE, 2015.

[148] V. Va, T. Shimizu, G. Bansal, and R. W. Heath, "Beam design for beam switching based millimeter wave vehicle-to-infrastructure communications," in *Communications (ICC), 2016 IEEE International Conference on*, pp. 1–6, IEEE, 2016.

[149] N. Michelusi and M. Hussain, "Optimal beam-sweeping and communication in mobile millimeter-wave networks," in *2018 IEEE International Conference on Communications (ICC)*, pp. 1–6, May 2018.

[150] L. P. Kaelbling, M. L. Littman, and A. R. Cassandra, "Planning and acting in partially observable stochastic domains," *Artificial Intelligence*, vol. 101, no. 1, pp. 99 – 134, 1998.

[151] L. P. Kaelbling, M. L. Littman, and A. W. Moore, "Reinforcement learning: A survey," *Journal of artificial intelligence research*, vol. 4, pp. 237–285, 1996.

[152] M. T. J. Spaan and N. A. Vlassis, "Perseus: Randomized point-based value iteration for pomdps," *CoRR*, vol. abs/1109.2145, 2011.

[153] E. Altman, *Constrained Markov decision processes*, vol. 7. CRC Press, 1999.

[154] E. Altman and F. Spieksma, "The linear program approach in multi-chain markov decision processes revisited," *Zeitschrift für Operations Research*, vol. 42, no. 2, pp. 169–188, 1995.

[155] J. D. Isom, S. P. Meyn, and R. D. Braatz, "Piecewise linear dynamic programming for constrained pomdps.," in *AAAI*, pp. 291–296, 2008.

[156] D. Kim, J. Lee, K.-E. Kim, and P. Poupart, "Point-based value iteration for constrained pomdps," in *IJCAI*, pp. 1968–1974, 2011.

[157] P. Poupart, A. Malhotra, P. Pei, K.-E. Kim, B. Goh, and M. Bowling, "Approximate linear programming for constrained partially observable markov decision processes.," in *AAAI*, pp. 3342–3348, 2015.

[158] A. Undurti and J. P. How, "An online algorithm for constrained pomdps," in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pp. 3966–3973, IEEE, 2010.

[159] S. M. Kay, *Fundamentals of statistical signal processing: Practical algorithm development*, vol. 3. Pearson Education, 2013.

[160] S. Shalev-Shwartz and S. Ben-David, *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014.

[161] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*, vol. 1. MIT press Cambridge, 1998.