UNIVERSITÀ DEGLI STUDI DELL'AQUILA

DIPARTIMENTO DI INGEGNERIA E SCIENZE DELL'INFORMAZIONE E MATEMATICA

Dottorato di Ricerca in Ingegneria e Scienze dell'Informazione

Curriculum Systems Engineering, telecommunications and HW/SW platforms

XXXV ciclo

Titolo della tesi

An Investigation of data-driven Methods Applied to Telecommunications

SSD ING-INF/04

Dottorando

Luis Felipe Florenzan Reyes

Coordinatore del corso

Prof. Vittorio Cortellessa

Tutor

Prof. Alessandro D'Innocenzo

Co-tutor

Dott. Francesco Smarra

a.a. 2021/2022

Abstract

The main objective of this thesis is to investigate the impact of Data-Driven techniques in ICT Engineering, with a specific focus on telecommunication systems. Firstly, a brief background on Data-Driven methods is provided, containing a recap on Machine Learning and Data reduction methods. Then, a literature review illustrates the impact of Data-driven methodologies on applications in the field of communication systems. The contributions of this PhD work are related to 4 different engineering applications to telecommunication systems, and are described as follows: (1) the use of data-driven methodologies to improve Multiple-Input Multiple-Output (MIMO) performance for crosstalk cancelation in optical systems supporting multiple spatial modes is addressed. In this respect, we proposed a reduction algorithm based on the Principal Component Analysis (PCA) and cross-correlation analysis to improve traditional equalizers' performance. (2) A novel regression trees-based methodology able to learn a Markov model of a fading channel via historical data of the signal-to-interference-plus-noise-ratio (SINR) is proposed. Such methodology is used to derive a Markov jump model of a wireless control network and thus to design a stochastic optimal controller that considers the inter-dependence between the plant and the wireless channel dynamics. Our methodology is validated using a WirelessHART [1] point-to-point communication based on the IEEE-802.15.4 standard. (3) A novel complexity reduction methodology is proposed for a data-driven control algorithm based on regression trees. In particular, the refinement procedure aims to reduce the dimension of the dynamical model without compromising the model accuracy and mitigating the overfitting problem. (4) the application of edge computing for real-time analysis to support autonomous operations of unmanned aerial vehicles (UAV) is addressed. Indeed, AUV autonomous operations necessitate the real-time analysis of information-rich signals, such as camera and LiDAR feeds, where the analysis algorithms often take the form of extremely complex deep neural networks (DNN). The continuous execution of such models onboard the UAV imposes a considerable resource consumption (*e.g.*, energy), while offloading the execution of the models to edge servers requires the transmission of the input signals over capacity-constrained, time-varying, wireless channels. We propose an innovative approach to control the computing pipeline of signal processing.

Contents

List of Figures

LIST OF ABBREVIATIONS

ACK Acknowledgment

AWGN Additive White Gaussian Noise

ACF Auto-Correlation Function

AR Auto-Regressive

ANN Artificial Neural Network

BER Bit Error Rate

CART Classification And Regression Trees

CMA Constant Modulus Algorithm

DA Data-Aided

DDS Direct Digital Synthesizer

DSSS Direct-Sequence Spread Spectrum

DRL Deep Reinforcement Learning

DAC Digital-Analog Converter

DSP Digital Singal Processing

ECL External Cavity Laser

EC Edge Computing

FIR Finite Impulse Response

GMM Gaussian Mixture Model

GRV Gaussian Random Variable

HMM Hidden Markov Model

LMS Least-Mean Square

LC Local Computing

MC Markov Chain

MDP Markov Decision Process

MJSLS Markov Jump Switched Linear System

mAP Mean Average Precision

MIMO Multiple-Input Multiple-Output

MD Mode Dispersion

ML Machine Learning

NRMSE Normalized Root-Mean-Square Error

OQPSK Offset Quadrature Phase-Shift Keying

RF Random Forest

RL Reinforcement Learning

RT Regression Tree

PDP Packet Delivery Probability

PEP Packet Error Probability

PWARX Piece-Wise Auto-Regressive eXogenous

PWA Piece-Wise Affine

PCE Power Control Error

PDP Power Delay Profile

PC Principal Component

PCA Principal Components Analysis

PDF Probability Density Function

PLL Phase-Locked Loop

QP Quadratic Programming

QPSK Quaternary Phase Shift Keying

SEM Standard Error of the Sample Mean

SDM Space-Division Multiplexed

SNR Signal-to-Noise Ratio

SINR Signal-to-Interference-plus-Noise Ratio

SMA Simple Moving Average

SVM Support Vector Machine

S-MPC Stochastic-Model Predictive Control

SARX Switching Auto-Regressive eXogenous

TDE Time-Domain Equalizer

TPM Transition Probability Matrix

RL Reinforcement Learning

UAV Unmanned Aerial Vehicles

WDM Wavelength Division Multiplexed

WNCS Wireless networked control system

1 Introduction

Contents

Almost all past and present generations of communication networks are based on mathematical models derived from theoretical considerations. Indeed, all phases of network design employ physical models describing in quantitative terms the effect each system component has on the overall performance. Models are used for initial network planning and deployment, network resource management, and network maintenance and control. Based on theoretical considerations, infrastructure nodes are statically deployed to cover and manage fixed geographical areas. In addition, traditional optimization theory is used to optimize network performance through the

centralized allocation of the available system resources. However, this traditional approach to network design has at least two drawbacks [2]:

- every model is inherently an approximation and a trade-off exists between the accuracy of the model and its complexity. Accurate models can be too complex to handle, whereas simple models cannot be accurate enough.
- Static infrastructure deployment might need to be more flexible to adapt to heterogeneous service requirements and randomly evolving environments with unpredictable on-demand connectivity requests.

The issues above can be ignored if the scenario allows the derivation of a physical model that is accurate and tractable enough to meet the desired performance requirements. For instance, physical modeling was efficiently used from 1G to 4G wireless systems; nevertheless, the past and present generations of wireless networks need a new design paradigm. As shown in [3], present and future networks are anticipated to witness an exponential increase due to the dramatic growth of connected devices and the rise of innovative vertical services with heterogeneous and stringent performance requirements. Indeed, according to the International Data Corporation, there will be 80 billion devices connected to the Internet by 2025, the global data will reach 163 zettabytes (which is 10 times the data generated in 2016) and it is forecasted that by 2020 over one billion people and more than 26 billion devices will be connected to the Internet, raising the number of connected devices by more than 10 billion compared to 2015 [4].

Deploying more performing communication technologies could not ensure the flexibility to accommodate diverse users with extremely heterogeneous service requirements. Instead, new architectural and management solutions are required. Our vision to overcome this situation is to resort to a data-driven paradigm for network design. The best policy comes from analyzing a mathematical model and studying and processing previous communication data. In other words, based on the performance obtained by given policies during previous communication sessions,

one should be able to decide the best policy to use. A framework that goes in this direction is Machine Learning (ML).

Although the main reason for using ML tools is to reduce the reliance on network design and operation on mathematical models, our vision is not for data-driven approaches to completely replace mathematical modeling and analysis. On the contrary, to overcome the complexity crunch, a cross-fertilization between model-based and data-driven approaches is necessary. Our vision, which will be supported throughout this work, is for ML and mathematical modeling to complement each other.

The rest of this chapter contains an introduction to the main motivation for the introduction of data-driven methodologies in communication systems, then, a background on the data-driven methodologies used in this work is provided. Some concrete examples of data-driven methodologies applied to communication systems are provided. Finally, we summarize the novelty and contributions of this thesis.

1.1 Motivation for using data-driven techniques in communication systems

In the last few years, the application of mathematical approaches derived from the ML discipline has attracted the attention of many researchers and practitioners in optical and wireless communications and networking fields [5]. In a general sense, the underlying motivations for this trend can be identified as follows:

- increased system complexity: the adoption of advanced transmission techniques, such as those enabled by coherent technology, and the introduction of highly flexible networking principles, such as, e.g., the Enhanced Optical Networking paradigm, have made the design and operation of optical networks extremely complex, due to the large number of tunable parameters to be considered (e.g., modulation formats, symbol rates, adaptive coding rates, adaptive channel bandwidth, etc.); in such a scenario, accurately modeling the system through closed-form formulas is often very hard and often causes

resource underutilization and, consequently, increasing of the system cost; on the contrary, ML methods can capture complex nonlinear system behavior with relatively simple training of supervised or unsupervised algorithms, which exploit knowledge of historical network data and therefore solve complex cross-layer problems, typical of the optical networking field;

- increased data availability: modern networks are equipped with a large number of monitors, able to provide several types of information on the entire system, e.g., traffic traces, signal quality indicators (such as Bit Error Rate (BER)), equipment failure alarms, users' behavior, etc.; here, the enhancement brought by ML consists of leveraging the plethora of collected data and discovering hidden relations between various types of information.

We analyze in the following subsection several use cases wherein data-driven methodologies have been applied to enhance the performance of optical and wireless systems.

Optical communications

The application of ML to physical layer use cases is mainly motivated by nonlinear effects in optical fibers, which makes analytical models inaccurate or even too complex. As main applications, we consider the impact on the optical communication performance in terms of BER.

Regarding the networking layer, the same motivation holds for applying ML techniques. In particular, the design and management of optical networks are continuously evolving, driven by the enormous increase in transported traffic and drastic changes in traffic requirements, e.g., capacity, latency, user experience, and Quality of Service (QoS). Therefore, current optical networks are expected to be run at much higher utilization than in the past while providing strict guarantees on service quality. While aggressive optimization and traffic-engineering methodologies are required to achieve these objectives, such complex methodologies may suffer scalability issues and involve unacceptable computational complexity.

Wireless communications

Problems that arise in wireless communication systems are frequently formulated as classification, detection, estimation, and optimization problems; for which ML techniques can provide elegant and practical solutions. In this context, the application of Machine Learning (ML) to wireless communications seems almost natural and presents a clear motivation [6].

More in detail, we discuss the application of machine learning techniques to identify fading channels with the goal of introducing the methodologies presented in chapter 3 regarding wireless communication standards specifically designed for automation. We refer the readers to [7] and [2] for a complete discussion.

Since the mid-1990s, Markov models have been widely used for modeling wireless flat-fading channels in various applications, ranging from modeling channel error bursts to decoding at the receiver. Markov models are versatile, and with suitable choices of model parameters, can capture the essence of time-varying fading channels [8].

Considering a case study of industrial communications, several challenges such as interference, power, and bandwidth constraints come into play. Significant performance gain can be achieved by learning and estimating the varying channel dynamics and nullifying the channel's effect from the received signal samples to estimate the transmitted bits, commonly known as adaptive channel equalization.

ML methodologies provided efficient solutions for Markov model identification.

1.2 Background on data-driven methodologies

This section summarizes the Machine Learning and Data Analysis fundamental concepts needed for the argumentation of the topics in this thesis.

1.2.1 Machine Learning

This section overviews some of the most popular algorithms commonly classified as machine learning. The literature on ML is so extensive that even a superficial overview of all the main ML approaches goes far beyond the possibilities of this section. The readers can refer to several books on the subjects [9], [10], and [11].

ML is a branch of Artificial Intelligence that pushes forward the idea that, by giving access to previous experience, machines can learn by themselves how to solve a specific problem. By leveraging complex mathematical and statistical tools, ML renders machines capable of performing independent tasks previously solved by human beings. This idea of automating complex tasks has generated high interest in the networking field, with the expectation that several activities involved in designing and operating communication networks can be offloaded to machines.

We divide the algorithms into three main categories, described in the following sections: supervised learning, unsupervised learning, and reinforcement learning.

Supervised learning

Supervised learning is used in various applications, such as speech recognition, spam detection, and object recognition. The goal is to predict the value of a vector variable ρ given the value of a vector of input variables λ. The output variable is a continuous variable in regression problems, i.e $\rho \in \mathbb{R}^{n_\rho}$ or a discrete variable in classification problems, i.e. $\rho \in \mathbb{N}^{n_\rho}$. A training data set comprises D samples of the input variables and the corresponding output values. Different learning methods construct a function $\hat{\rho}(\lambda)$ that allows predicting the value of the output variables in correspondence to a new value of the inputs.

Supervised learning can be broken down into two main classes, described below: Parametric models, where the number of parameters to use in the model is fixed, and non-parametric models, where their number is dependent on the training set.

Parametric models: the function $\hat{\rho}$ is a combination of a fixed number of parametric basis functions. These models use training data to estimate a fixed set of parameters w. After the learning stage, the training data can be discarded since the prediction in correspondence with new inputs is computed using only the learned parameters w. Linear models for regression and classification, which consist of a linear combination of fixed nonlinear basis functions, are the simplest parametric models in terms of analytical and computational properties. Many choices are available for the basis functions: from polynomial to Gaussian, to sigmoidal, to

Fourier basis, etc. In the case of multidimensional output variables, it is possible to use separate basis functions for each output component or, more commonly, apply the same set of basis functions for all the components. Note that these models are linear in the parameters w, which results in several advantageous properties, e.g., closed-form solutions to the least-squares problem.

ANNs apply a series of functional transformations to the inputs. An ANN is a network of units or neurons. Each unit's basis or activation function is a nonlinear function of a linear combination of the unit's inputs. Each neuron has a bias parameter that allows for any fixed offset in the data. The bias is incorporated into the set of parameters by adding a dummy input of unitary value to each unit. The linear combination coefficients are the parameters w estimated during the training phase. The logistic sigmoid and hyperbolic tangent are the most commonly used nonlinear functions. The activation function of the output units of the ANN is the identity function, the logistic sigmoid function, and the softmax function for regression, binary classification, and multiclass classification problems, respectively.

Decision trees consider a predictor dataset and a response dataset of D samples each. In the case of the CART algorithm [12], the dataset is partitioned into a set of hyper-rectangles $R_1, \ldots, R_\ell$, corresponding to the ℓ leaves of the tree. Then, $\hat{\rho}(\lambda)$ is estimated in each leaf τ_i using a constant c_{τ_i} predictors. Without any loss of generality, we restrict our attention to recursive binary partition. The CART algorithm creates the partition using a greedy algorithm to optimize the split variables and split points: starting with the whole dataset, consider a split variable j over the n available and a split point s, and define the pair of optimal half-planes. For each splitting variable, the determination of the split point s can be determined very quickly; hence, by scanning through all of the inputs, the best pair (j, s) is feasible. Once the best split is found, the dataset is partitioned into the two resulting regions, then the splitting procedure is repeated on each of the two regions. The process is repeated on all the resulting regions until a stopping criterion is applied, e.g. tree size is a tuning parameter chosen to avoid overfitting and variance phenomena. The methodology will be further discussed in chapter 3 and chapter 4.

Random Forest (RF) are a combination of tree predictors such that each tree depends on the values of a random vector sampled independently and with the same distribution for all trees in the forest [13]. The generalization error for forests converges to a limit as the number of trees in the forest becomes large. The generalization error of a forest depends on the strength of the individual trees in the forest and the correlation between them.

Non-parametric models: In non-parametric methods, the number of parameters depends on the training set. These methods keep a subset or the entirety of the training data and use them during prediction. The most used approaches are k-nearest neighbor models and Support Vector Machine (SVM)s. Both can be used for regression and classification problems.

In the case of k-nearest neighbor methods, all training data samples are stored (training phase). During prediction, the k-nearest samples to the new input value are retrieved. For classification problems, a voting mechanism is used; for regression problems, the mean or median of the k-nearest samples provides the prediction.

In SVMs [14], basis functions are centered on training samples; the training procedure selects a subset of the basis functions. The number of selected basis functions, and the number of training samples that have to be stored, is typically much smaller than the cardinality of the training dataset. SVMs build a linear decision boundary with the largest possible distance from the training samples. Only the closest points to the separators, the support vectors, are stored. A nonlinear optimization problem with a convex objective function has to be solved to determine the parameters of SVMs, for which efficient algorithms exist. An essential feature of SVMs is that by applying a kernel function, they can embed data into a higher dimensional space, in which data points can be linearly separated. The kernel function measures the similarity between two points in the input space; it is expressed as the inner product of the input points mapped into a higher dimensional feature space in which data become linearly separable. The simplest example is the linear kernel, in which the mapping function is the identity function. However, provided that we can express everything in terms of kernel evaluations, it

is not necessary to explicitly compute the mapping in the feature space. Indeed, in the case of one of the most commonly used kernel functions, the Gaussian kernel, the feature space has infinite dimensions.

Unsupervised learning

Social network analysis, gene clustering, and market research are among the most successful applications of unsupervised learning methods. In the case of unsupervised learning, the training dataset consists only of a set of input vectors λ. While unsupervised learning can address different tasks, clustering or cluster analysis is the most common. Clustering is the process of grouping data so that the intra-cluster similarity is high while the inter-cluster similarity is low. The similarity is typically expressed as a distance function, which depends on the data type. There exists a variety of clustering approaches. Here, we focus on k-means and Gaussian Mixture Model (GMM)s as examples of partitioning approaches, we refer to [9] and [10] for further details.

K-means is an iterative algorithm starting with an initial partition of the data into K clusters. The cluster's centers are optimally computed in the first step. Then, data points are assigned to the cluster with the closest center. The procedure - center computation and data assignment - is repeated until the assignment does not change or a predefined maximum number of iterations is exceeded. As a result, the algorithm may terminate at a local optimum partition.

While k-means assigns each point uniquely to one cluster, probabilistic approaches perform a soft assignment and provide a measure of the uncertainty associated with the assignment. GMMs linearly combine Gaussian distributions and are one of the most widely used probabilistic approaches to clustering. The model's parameters are the mixing coefficient of each Gaussian component, the mean, and the covariance of each Gaussian distribution. The expectation-maximization algorithm is used to maximize the log-likelihood function for the parameters given a dataset since no closed-form solution exists in this case. After initializing the parameters and evaluating the initial value of the log-likelihood, the algorithm

alternates between two steps. In the expectation step, the current values of the parameters are used to determine the "responsibility" of each component for the observed data, i.e., the conditional probability of latent variables given the dataset. The maximization step uses these responsibilities to compute a maximum likelihood estimate of the model's parameters.

So far, we have assumed that the data is unstructured, i.e., the observations are assumed to be independent and identically distributed. This assumption is unreasonable for many data sets where the observations arrive in a sequence and subsequent observations are correlated. Sequential data can occur in time series modeling, as in financial data or the weather, and also in situations where the sequential nature of the data is not necessarily tied to time, as in protein data which consists of sequences of amino acids. As the most basic level, time series modeling consists of building a probabilistic model of the present observation given all past observations $p(\lambda_{t_k}|\lambda_{t_{k-1}}, \lambda_{t_{k-2}}...)$. Because the history of observations grows arbitrarily large, it is necessary to limit the complexity of such a model. The first approach is to limit the window of past observations by considering a first-order Markov model. Thus one can simply model $p(\lambda_{t_k}|\lambda_{t_{k-1}}, \lambda_{t_{k-2}}...)$ and assume that this relation holds for all t_k.

The critical aspect of Hidden Markov Model (HMM), see [15], is the switching law associated with the discrete hidden state indicating the active mode. Let s_{t_k} denote the hidden state of an HMM at time t_k. We assume that s_{t_k} can take discrete values in $\{1, ..., K\}$. The state-transition probabilities $P(s_{t_k}|s_{t_{k-1}}, w)$ are captured by a $K \times K$ transition matrix A, with elements $A_{i,j} = p(s_{t_k} = i|s_{t_{k-1}} = j, w)$. The observations in an HMM can be either continuous or discrete. For continuous observations λ_{t_k} one can, for example, choose a Gaussian density; thus $p(\lambda_{t_k}|s_{t_k} = i, w)$ would be a different Gaussian for each choice of $i \in \{1, ..., K\}$. This model is the dynamical generalization of Gaussian mixtures.

Reinforcement Learning

Reinforcement Learning (RL) is used, in general, to address applications such as robotics, financial investment decisions, and inventory management, where the goal is to learn a policy, i.e., a mapping between states of the environment into actions to be performed while directly interacting with the environment.

The RL paradigm is characterized by agents aiming to learn by exploring the available actions and refining their behavior using only evaluative feedback, often called reward. The agent's goal is to maximize its long-term performance. Hence, the agent does not just consider the immediate reward but evaluates the consequences of its actions in the future. Delayed reward and trial-and-error constitute the two most significant features of RL.

In this subsection, we summarize the most basic RL algorithm, for further discussion we refer the readers to [16] and [17].

The most basic RL algorithm works in the Markov Decision Process (MDP) framework. The agent's perception at time t_k is represented as a state $s(t_k) \in S$, where S is the finite set of environment states. The agent interacts with the environment by performing actions. At the time t_k, the agent selects an action $a(t_k) \in A$, where A is the finite set of actions of the agent, which could trigger a transition to a new state. According to a reward function, the agent will receive a reward as a result of the transition. The agent's goal is to find the sequence of state-action pairs that maximizes the expected discounted reward, i.e., the optimal policy. In the context of MDP, it has been proved that an optimal deterministic and stationary policy exists. Several algorithms exist that learn the optimal policy in case the state transition and reward functions are known (model-based learning) and in case they are not (model-free learning). The most used RL algorithm is Q-learning, a model-free algorithm that estimates the optimal action-value function.

1.2.2 Data reduction

Complex data analysis and mining vast amounts of data can take a long time, making such analysis impractical or infeasible. Data reduction techniques can be applied to

obtain a reduced representation of the data set that is much smaller in volume yet closely maintains the integrity of the original data. Data reduction strategies include dimensionality reduction, numerosity reduction, and data compression. Feature reduction reduces the number of random variables or attributes under consideration. Feature selection is a method of dimensionality reduction in which irrelevant, weakly relevant, or redundant attributes or dimensions are detected and removed. Numerosity reduction techniques replace the original data volume with alternative, more minor forms of data representation. In data compression, transformations are applied to obtain a reduced or "compressed" representation of the original data. If the original data can be reconstructed from the compressed data without any information loss, the data reduction is called lossless. If we can reconstruct only an approximation of the original data, then the data reduction is called lossy.

In the rest of this section, we provide a brief background on two techniques of feature reduction: feature extraction and feature reduction that will useful in chapter 2. We refer readers to [18] for further discussion.

Feature Selection

Feature selection reduces the data set size by removing irrelevant or redundant attributes (or dimensions). The goal of attribute subset selection is to find a minimum set of attributes such that the resulting probability distribution of the data classes are as close as possible to the original distribution obtained using all attributes. Basic heuristic methods of attribute subset selection include the techniques that follow:

- Stepwise forward selection: starting from an empty set of attributes as the reduced set, the best of the original attributes is determined and added to the reduced set at each subsequent iteration.

- Stepwise backward elimination: The procedure starts with the complete set of attributes. At each step, it removes the worst attribute remaining in the set.

- Combination of forward selection and backward elimination: The stepwise forward selection and backward elimination methods can be combined so that, at each step, the procedure selects the best attribute and removes the worst from among the remaining attributes.

- Decision tree induction: Decision tree induction constructs a flowchart-like structure where each internal (non-leaf) node denotes a test on an attribute, each branch corresponds to an outcome of the test, and each external (leaf) node denotes a class prediction, as mentioned in section 1.2.1. The algorithm chooses the best attribute to partition the data into individual classes at each node. When decision tree induction is used for attribute subset selection, a tree is constructed from the given data. All attributes that do not appear in the tree are assumed to be irrelevant. The set of attributes appearing in the tree forms the reduced subset of attributes.

The stopping criteria for the methods may vary depending on the application. In addition, the measure employed during the reduction procedure also depends on the case study.

Feature extraction

We provide an intuitive introduction to principal components analysis as a dimensionality reduction method. Suppose that the data to be reduced consists of tuples or data vectors described by n attributes or dimensions. Principal components analysis searches for k n-dimensional orthogonal vectors that best represent the data, where $k \leq n$. The original data are thus projected onto a much smaller space, reducing dimensionality. Unlike attribute subset selection, which reduces the attribute set size by retaining a subset of the initial set of attributes, PCA "combines" the essence of attributes by creating an alternative, smaller set of variables. The initial data can then be projected onto this smaller set.

1.3 Application of data-driven methodologies to communication systems

After presenting the main concepts and tools of the data-driven framework, this section describes practical applications to the design of optical and wireless communication systems.

1.3.1 Optical communications

We focus this section on the use cases in the physical layer for conciseness. We refer the readers to [5] and [19] for a complete discussion including the other layers.

In the following, a description of the applications of ML at the physical layer is presented.

Quality of Service estimation

Quality of Transmission generally refers to different physical layer parameters, such as received Optical Signal-to-Noise Ratio (OSNR), BER, Q-factor, etc., which impact the "readability" of the optical signal at the receiver. Such parameters give a quantitative measure to check if a predetermined level of QoT would be guaranteed and are affected by several tunable design parameters, such as e.g. modulation format, baud rate, coding rate, physical path in the network, etc.

Therefore, optimizing this choice is not trivial, and often this wide variety of possible parameters cannot be solved manually.

A data-driven solution consists of designing a binary classification capable of predicting whether a candidate lightpath will meet the required quality based on the system characteristics.

In [20], the authors reduce uncertainties on network parameters and design margins by exploiting the data collected by a software-defined network controller. A regression model estimates the SNR based on guesses of unknown network parameters. Then, a gradient descent algorithm iteratively updates such guesses until the difference between the estimated and the SNR falls below a predefined threshold.

Similarly, in [21] and [22], ANN-based methods are proposed for accurately deciding the QoT of the newly arriving multicast requests in metro optical networks.

The data-driven QoT technique analyzes data of previous connection requests and, through a training procedure performed on a neural network, returns a data-driven QoT model that near-accurately decides the QoT of the newly arriving requests.

A random forest binary classifier is adopted in [23] to predict whether the BER of unestablished lightpaths meet the required system threshold based on traffic volume, desired route, and modulation format. The classifier inputs features, including the candidate lightpath's length, the number of traversed links, the amount of traffic to be transmitted, and the modulation format adopted for transmission. As output, the algorithm finds the lightpath that will most likely satisfy the BER requirements.

Nonlinearity mitigation

Due to optical fiber nonlinearities, such as the Kerr effect, self-phase modulation (SPM), and cross-phase modulation (XPM), the behavior of several performance parameters, including BER, Q-factor, Chromatic Dispersion (CD), Polarization Mode Dispersion (PMD), is highly unpredictable, which may cause signal distortion at the receiver (e.g., I/Q imbalance and phase noise). Therefore, complex analytical models are often adopted to react to signal degradation and/or compensate for undesired nonlinear effects. One of the performance metrics commonly used for optical communication systems is the data-rate×distance product. Due to the fiber loss, optical amplification needs to be employed, and for increasing transmission distance, an increasing number of optical amplifiers must be employed accordingly. Machine learning has been efficiently used to overcome the nonlinearities effects [24], as a case study, we present the optimal symbol detection.

In general, the receiver's task is to perform optimum symbol detection. When the noise has a circularly symmetric Gaussian distribution, the optimum symbol detection is performed by minimizing the Euclidean distance between the received symbol and all the possible symbols of the constellation alphabet. The nonlinearities distort the constellation diagram, and as an effect, cluster shapes become elliptical instead of circular and symmetric. In those cases, optimum symbol detection is no longer based on the Euclidean distance matrix, and the knowledge and full parametrization of the likelihood function is necessary.

In [25], the authors show the impact of Gaussian Mixture Models (GMMs) and the expectation maximization (EM) algorithm for combating nonlinear phase noise. Indeed, they show the EM's ability to track nonlinear distortion in the constellation. Similarly, in [26], SVM classifiers are used to identify decision boundaries separating the points of an M-PSK constellation.

In the context of signal equalization, chapter 2 provides a reduced complexity algorithm to efficiently implement a Digital Signal Process while preserving the quality of the information. In particular, we apply data-driven methodologies to improve MIMO DSP performance for crosstalk cancelation in optical systems supporting multiple spatial modes. Our idea is to focus on techniques that can reduce the amount of information needed to perform the equalization process.

1.3.2 Wireless communications and Wireless networked control systems

As the first topic, we focused on the use of data-driven learning methodologies able to identify time-varying channels. We refer the readers to [2] and [4] for a complete discussion including the other applications. As the second topic, we focused on enhancing Wireless networked control system closed-loop performance.

In [27], an adaptive rate control strategy based on Reinforcement Learning is proposed to learn the dynamically varying channel conditions. The time-varying fading channel is modeled as a finite state Markov chain, whose channel state transition probabilities are unknown but the instantaneous channel gains can be estimated. In this work, the authors propose to use Q-learning to track the varying environmental changes in pursuit of the optimal control policy online.

In [28], the authors combine Hidden Markov Models and the EM algorithm to estimate the channel parameters and inference the status of the channel. They also propose a procedure for choosing the correct number of states to obtain a model as faithful and as simple as possible.

Instead, in chapter 3, we estimate the channel of a Wireless Network Control System, as defined in [29], based on the historical data of the transmissions. In

particular, the approach focuses on the interdependence between the system and the channel with the goal of analyzing the impact of accurate channel models in closed-loop performance.

The last topic of our research is the exploitation of Edge Networking to enhance Wireless networked control system performances. Edge computing is a central component of modern infrastructures and a key enabler of many applications. Recent contributions proposed a wide array of solutions to improve the performance of edge offloading and optimize resource usage. In [30], the authors propose a framework to solve a mixed-integer linear program that jointly optimizes service cashing, cloud usage, and energy consumption. [31] presents a controller to minimize the overall energy consumption under hard per-task delay constraints in systems with energy harvesting. In [32], the authors propose a multi-scale control logic to dynamically reconfigure distributed cloudlets. The framework in [33] jointly considers devices' topology, available resources, and wireless channel state to assign computing tasks to edge servers. In [34], the authors propose a solution to dynamically control DNN-based video analytics on edge servers.

In contrast with these contributions, in chapter 5, we proposed a methodology that controls task offloading based on the control needs of an autonomous unmanned aerial vehicle to reduce resource usage while preserving mission performance.

1.4 Thesis contribution and organization

The remainder of this dissertation will contain the novelty contribution of this work, and will be organized as follows:

- *chapter 2* discusses the use of data-driven methodologies to improve MIMO DSP performance for crosstalk cancelation in optical systems supporting multiple spatial modes. In this respect, the chapter proposed a reduction algorithm based on the PCA and cross-correlation analysis to improve traditional equalizers' performance. We test our reduction method on the data of the world-first Space-Division Multiplexed (SDM) field trial conducted

in the INCIPICT testbed [35] (http://incipict.univaq.it/), where successful transmission in a coupled-core four-cores fiber deployed in the underground tunnel network in the historical downtown of the city of L'Aquila, Italy, was demonstrated [36]. The methodology has been presented in [37].

- *chapter 3* proposes a Regression trees-based methodology able to learn a Markov model of a fading channel via historical data of the signal-to-interference-plus-noise-ratio (SINR). Such methodology can be used to derive a Markov jump model of a wireless control network, and thus to design a stochastic optimal controller that considers the interdependence between the plant and the wireless channel dynamics. The methodology is validated using a WirelessHART [1] point-to-point communication based on the IEEE-802.15.4 standard. The studies have been presented in [38].

- in *chapter 4* a complexity reduction methodology is proposed for a data-driven control algorithm proposed in chapter 3. In particular, the refinement procedure aims to reduce the dimension of the dynamical model without compromising (and indeed improving) the model accuracy and mitigating the overfitting problem. The innovation related to the refinement procedure proposed in this work was presented in [39] and is currently patent pending.

- *chapter 5* discusses the application of edge computing for real-time analysis to support autonomous operations of unmanned aerial vehicles (UAV). Indeed, AUV autonomous operations necessitate the real-time analysis of information-rich signals, such as camera and LiDAR feeds, where the analysis algorithms often take the form of extremely complex deep neural networks (DNN). The continuous execution of such models onboard the UAV imposes a considerable resource consumption (*e.g.*, energy) while offloading the execution of the models to edge servers requires the transmission of the input signals over capacity-constrained, time-varying, wireless channels. In this chapter, we propose an innovative approach to control the computing pipeline of signal processing. The methodology has been presented in [40].

- *chapter 6* draws some concluding remarks and outlining future research directions.

1.5 List of publications and patent

Publications

- L.F. Florenzan Reyes, F. Smarra, Y. Zacchia Lun, A. D'Innocenzo. Learning Markov models of fading channels in wireless control networks: a regression trees based approach. In Proc. 29th IEEE Mediterranean Conference on Control and Automation, Bari, Italy, June 2021 (Invited session).

- L.F. Florenzan Reyes, F. Smarra, A. D'Innocenzo. Reduced SARX modeling and control via Regression Trees. In Proc. American Control Conference 2022, Atlanta, USA, June 2022.

- L. F. Florenzan Reyes, F. Smarra, R. Ryf, T. Hayashi, A. Marotta, C. Antonelli, A. D'Innocenzo. Data-driven efficient digital signal processing over a field trial space-division multiplexed fiber-optic transmission. In Proc. International Conference on Computer Communication and Networks 2022, Hawaii, USA, July 2022.

- L. F. Florenzan Reyes, F. Smarra, A. D'Innocenzo, M. Levorato. CADET: Control-Aware Dynamic Edge Computing for Real-Time Target Tracking in UAV Systems. In Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing, Rhodes islands, Greece, June 2023 (Invited session)

Patent

- A. D'Innocenzo, F. Smarra, F. Graziosi, L. F. Florenzan Reyes. A computer-implemented method for optimizing energy consumption. Provisionally accepted by Ministero dello Sviluppo Economico, Italy.

2

Data-driven efficient digital signal processing over a field trial space-division multiplexed fiber-optic transmission

Contents

This chapter discusses the use of data-driven methodologies to improve Multiple-Input Multiple-Output (MIMO) Digital Singal Processing (DSP) performance for crosstalk cancelation in optical systems supporting multiple spatial modes.

A crucial problem in time-domain equalization is the large quantity of data needed to compensate for the effect of the crosstalk when the signals cover a large number of kilometers. In this respect, the study of techniques that can

reduce the amount of information needed to perform equalization is addressed, and a reduction algorithm based on the Principal Components Analysis (PCA) and cross-correlation analysis to improve traditional equalizers' performance is proposed. Experimental validation is performed over the world-first Space-Division Multiplexed (SDM) field trial.

The discussion in this chapter has already been presented in [37].

Chapter organization. The chapter is organized as follows. Section 2.1 provides a brief background about space-division multiplexed fiber-optic transmission. We introduce the structure of a MIMO DSP based on time-domain equalization in Section 2.2. We describe the proposed reduction methodology in Section 2.3. We describe the coupled-core four-core optical fiber experimental setup in Section 2.4. Finally, we experimentally validate the performance of the proposed reduction methodology in Section 2.5.

Chapter notation. Coherently with the literature, we denote in this chapter with $y_l(t)$ the received signal at channel l, with $z_i(kT)$ the k-th sample of the signal at the i-th MIMO equalizer output, and with $a_{i,k}$ the symbol associated to the i-th channel at the k-th time-step.

2.1 Introduction

The demand for data traffic has been growing exponentially over the past decades, fueled by an increasing number of data-hungry applications relying on the global fiber-optic network [41]. Growth rates peaking at above 100% per year were recorded in the late nineties, and following the telecom bubble they stabilized to long-term growth of about 60% per year, implying a doubling of the demand every year and a half. The most obvious approach to accommodating the increasing demand would be the deployment of new fibers, however, this approach is in contrast with the operators' pressure to reduce the cost per bit of transmitted information. In fact, since the beginning, the solution of choice has been the search for efficient transmission techniques with the goal of achieving the full exploitation of the fiber-optic channel capacity [42]. In the nineties, the transition

from single-channel transmission to wavelength-division multiplexing enabled the efficient use of the fiber bandwidth, while the advent of coherent technology in early 2000 allowed the transition from simple on-off-keying signaling to the use of advanced modulation formats in conjunction with polarization multiplexing, thereby yielding an increase of the achieved transmission rates by a factor of four (the two quadratures and orthogonal polarizations of the electric field). Correspondingly, the spectral efficiencies of commercial systems increased from a fraction of a bit to several bits. Nonetheless, despite these impressive advances, the growth rate of spectral efficiency has recently reduced, owing to the fact that modern coherent systems have almost reached the fundamental transmission capacity limits of single-mode fiber systems dictated by Shannon theory [43]. In this situation, it became clear that spatial parallelism is the only sustainable approach to further scaling the capacity of fiber-optic systems, with space being the only unexploited physical dimension of the optical fiber.

Spatial multiplexing can be implemented in diverse forms and requires some degree of integration to scale the capacity while allowing at the same time some cost-per-bit reduction. A particularly interesting implementation of SDM transmission, intensely researched in the past decade, is the one based on the use of multi-mode and multi-core fibers [44], where the transmitted information is encoded in multiple modes of the same fiber. Extracting the transmitted information at the receiver requires MIMO DSP, with a complexity that depends on the propagation regime. Indeed, owing to manufacturing imperfections and deployment-related issues, ideally uncoupled modes undergo some coupling during propagation from transmitter to receiver. If coupling occurs to a negligible extent, as is the case in uncoupled-core multi-core fibers, then only the two polarization components of each mode need to be disentangled, just like in coherent single-mode systems, and 2×2 MIMO DSP is required for each spatial channel. If, conversely, the degree of coupling is considerable, then $2M \times 2M$ MIMO DSP is necessary, where M is the number of spatial modes and the factor of two accounts for the two-fold polarization degeneracy of each spatial mode. In this case, the MIMO DSP complexity is aggravated by

the fact that spatial modes are characterized by different propagation velocities, which disperses the inter-modal cross-talk in time - a phenomenon known as Mode Dispersion (MD) [45]. In this chapter, we focus on the case of coupled-core multi-core fibers, where strong mixing between all of the transmitted signals occurs on a very short length-scale, of the order of a fraction of a meter. In this propagation regime MD accumulates proportionally to the square root of propagation distance, rather than to the propagation distance itself (as is the case in the regime of weak coupling between modes), but it remains the major factor determining the MIMO DSP receiver complexity.

In this respect, the goal of the chapter is to provide a reduced complexity algorithm to efficiently implement DSP while preserving the quality of the information.

In particular, we consider using data-driven methodologies to improve MIMO DSP performance for crosstalk cancelation in optical systems supporting multiple spatial modes.

Our idea is to focus our attention on techniques that can reduce the amount of information needed to perform the equalization process. To this aim, different options can be exploited.

One consists of reducing the number of taps by analyzing the input-output cross-correlation of the MIMO system. In this case, the basic idea is to exploit the most correlated samples of the measurements collected from the transmission assuming that less correlated ones provide less information with respect to the DSP accuracy: this could provide a complexity reduction of the DSP algorithm for each specific channel. Another well-known technique for data set reduction is PCA, a well-known method that analyzes high-dimensional data sets and identifies auto-correlations among the data entries. PCA projects the data onto a lower-dimensional space where the most important relations between features and other relevant information of the data set are preserved, while the rest is discarded. In a similar fashion, Canonical Correlation Analysis finds a linear combination of two random variables that maximize their cross-correlation [46].

In this chapter, we provide a taps reduction methodology that merges in a novel fashion the cross-correlation and the PCA and discusses its impact in terms of the performance of the DSP algorithm. We test our reduction method on the data of the world-first SDM field trial conducted in the INCIPICT testbed [35] (http://incipict.univaq.it/), where successful transmission in a coupled-core four-cores fiber deployed in the underground tunnel network in the historical downtown of the city of L'Aquila, Italy, was demonstrated [36].

Results show that with the proposed approach it is possible to reduce the number of taps, without compromising the decoding accuracy, of 60% and 50% when considering a transmission over 692 and 3450 km respectively.

We believe this preliminary result is very interesting if thought in perspective of its combined use with methodologies such as digital subcarriers. As it is well known, the complexity of the equalization process in the time domain varies linearly with the number of taps, while the one in the frequency domain goes with the log_2 [47]. In particular, for transmissions of approximately 100km, the complexity of the 2 processes is comparable when the number of taps is around 13. In this sense, the proposed methodology could be integrated with the digital subcarriers to further reduce the number of taps while guaranteeing decoding accuracy. In this way, the gap between the time and the frequency equalization processes could be significantly reduced, but this is the venue for future work.

2.2 Digital Signal processing

In this section, we summarize the structure of the DSP used in the $M \times M$ MIMO receiver to produce one of the M output sequences, with M denoting the number of channels. The structure is obtained leveraging the method proposed in [48], which consists of an equalization stage and a carrier synchronization stage.

We denote with $y_l(t)$ the received signal at channel l, for $l = 1, \ldots, M$. Each signal $y_l(t)$ is sampled with a sampling frequency equal to $\frac{2}{T}$, where T is the symbol interval of the transmission. The sampled signals, $y_l(kT/2)$, are applied to the input

of a linear fractionally spaced MIMO equalizer consisting of a $M \times M$ matrix of Finite Impulse Response (FIR) filters, each of the same length $\bar{N} = 2N + 1$.

The k-th sample of the signal at the i-th MIMO equalizer output, taken at time kT, is given by the multidimensional convolution

$$z_i(kT) = \sum_{l=1}^{M} \sum_{n=-N}^{N} c_{il,n}^{(k)} y_l(kT - nT/2), \tag{2.1}$$

where $c_{il,n}^{(k)}$ denotes the n-th tap of the equalizer between input l and output i at the same time. Let $\boldsymbol{c}_{il}^{(k)} = \left[c_{il,-N}^{(k)}, \ldots, c_{il,N}^{(k)}\right]^\top$ and $\boldsymbol{y}_l\left(kT\right) = \left[y_l\left(kT - NT/2\right), \ldots y_l\left(kT + NT/2\right)\right]^\top$ denote respectively the vector of the $\bar{N}$ equalizer taps between the $l-$th input and $i-$th output and the vector of samples at the l-th input. The equalizer taps are recursively updated according to the classical stochastic gradient algorithm as

$$\boldsymbol{c}_{il}^{(k+1)} = \boldsymbol{c}_{il}^{(k)} + \gamma \boldsymbol{y}_l^*\left(kT\right) e_i(kT), \tag{2.2}$$

where γ is the step size, $e_i(kT)$ is the error at the $i-$th equalizer's output, and $*$ denotes complex conjugation. Equalizer pre-convergence is achieved by implementing Data-Aided (DA) Least-Mean Square (LMS) adaptation of the equalizer taps.

After pre-convergence, the equalizer adaptation is based on the Constant Modulus Algorithm (CMA): at the $k-$th time instant, the error signal used to update the equalizer taps is given by

$$e_i(kT) = \begin{cases} a_{i,k} - z_i(kT), & \text{for DA-LMS,} \\ z_i(kT)(R_2 - z_i(kT)z_i^*(kT)), & \text{for CMA,} \end{cases} \tag{2.3}$$

where $a_{i,k}$ is the Quaternary Phase Shift Keying (QPSK) symbol associated to the $i-$th transmitted sequence and R_2 is the Godard constant of order 2. The cost function used in the CMA is phase-blind. This means that the CMA output is a constellation affected by a phase error. For this reason, after the equalization stage, the signal experiences a carrier synchronization process. The carrier synchronizer used in this work is a closed-loop compensator consisting of a Phase-Locked Loop

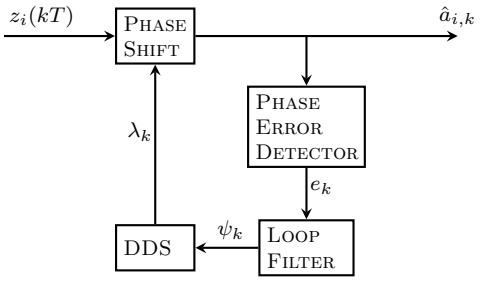

Figure 2.1: Carrier Synchronizer.

(PLL)-based algorithm described in [49]. The block scheme that represents the carrier synchronizer is shown in Fig. 2.1.

The output of the synchronizer, $\hat{a}_k$, is a frequency-shifted version of the signal $z(kT)$ for the k-th sample. The synchronizer output is

$$\hat{a}_k = z(kT)e^{j\lambda_k}, \tag{2.4}$$

where λ_k is the output of the Direct Digital Synthesizer (DDS). The DSS is the discrete-time version of a voltage-controlled oscillator. To correct for the frequency offset, the algorithm first determines the phase error e_k with the following scheme:

$$\begin{aligned} e_k = &\operatorname{sgn}\left(\operatorname{Re}\left\{z(kT)\right\}\right)\operatorname{Im}\left\{z(kT)\right\} \\ &- \operatorname{sgn}\left(\operatorname{Im}\left\{z(kT)\right\}\right)\operatorname{Re}\left\{z(kT)\right\}. \end{aligned} \tag{2.5}$$

Then, the phase error passes through the following biquadratic loop filter to ensure system stability:

$$\psi_k = g_1 e_k + \psi_{k-1}, \tag{2.6}$$

where ψ_k is the output of the loop filter at sample k, and g_1 is the integrator gain. The DSS is another biquadratic loop filter with the following expression:

$$\lambda_k = (g_p e_{k-1} + \psi_{k-1}) + \lambda_{k-1}. \tag{2.7}$$

2.3 Reduction approaches

Complex multivariate data structures are often better understood by studying low-dimensional projections because some of the inputs can be correlated with others, thus providing less useful information to the equalizer. For this reason, we propose in this section reduction methodologies that aim at selecting a reduced number of inputs that provide the needed information. In particular, we propose the following two approaches consisting on the use of the cross-correlation, and its combination with the PCA.

2.3.1 Cross-Correlation Analysis

The basic idea behind the cross-correlation is to select the inputs of the equalizer based on their correlation with the equalized signal. For each channel, the Time-Domain Equalizer (TDE) can be seen as a linear regression where the predictor variable lies in $\mathbb{C}^K$, where $K = \bar{N}M$, and the response variable is a_i. To perform the reduction we analyze the cross-correlation between the input of the equalizer, i.e. $\boldsymbol{y}$, and the transmitted QPSK symbol, i.e. a_i. Then, we define the vector $R \in \mathbb{R}^K$ as follows

$$R = corr(\boldsymbol{y}, a_i) \otimes corr(\boldsymbol{y}, a_i)^*, \tag{2.8}$$

with $\otimes$ we denote the element-by-element multiplication. Finally, we select the $\bar{K} < K$ samples with the largest correlation index.

2.3.2 Principal Components Analysis

Principal Components Analysis (PCA) [50] is a method that analyzes high-dimensional data and identifies correlations among the data entries (features). PCA then projects the data down to a lower-dimensional representation in which important relations between features and other relevant information of the data set are preserved, while the rest is discarded. Our goal is to provide a reduction methodology that combines the PCA with the cross-correlation analysis.

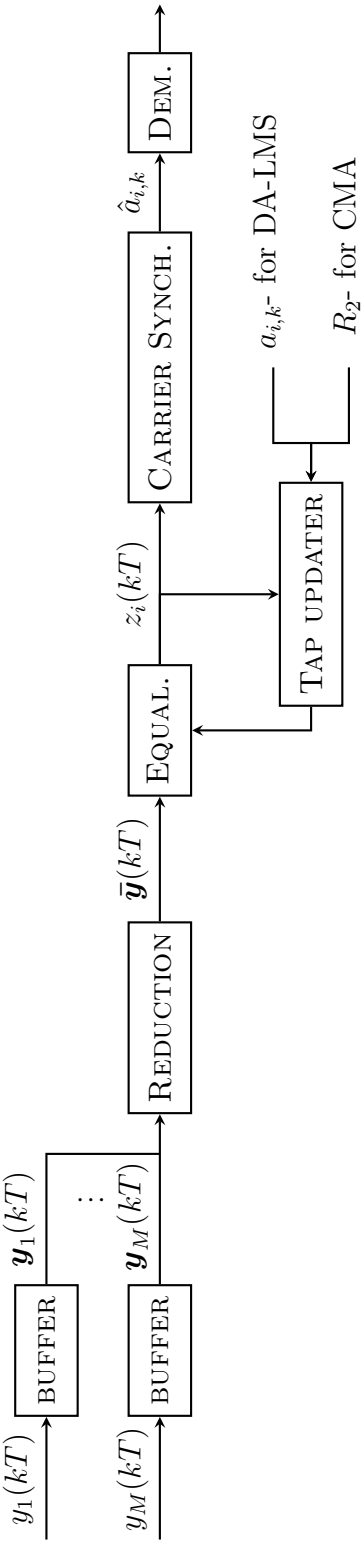

Figure 2.2: Reduced TDE - channel i.

Given a data set described by an $n \times p$ data matrix X, whose j^{th} column vector is the x_j observation of the j^{th} variable. PCA seeks a linear combination of matrix columns X with maximum variance. Such a linear combination is given by $\sum_{j=1}^{p} t_j x_j = Xt$ with $t = [t_1, \dots t_p]$. The variance of the linear combination is given by $var(Xt) = t^\top Xt$, where S is the sample covariance matrix associated with the data set X. An additional restriction must be imposed to obtain a well-defined problem: the most common is the solution normalization, i.e. $t^\top t = 1$.

The problem can be rewritten by exploiting the method of Lagrange multiplier and differentiating with respect to t in the following

$$St - \lambda t = 0 \Leftrightarrow St = \lambda t, \tag{2.9}$$

where λ is the Lagrange multiplier. Thus, λ must be the largest eigenvalue of the covariance matrix S, and t the corresponding eigenvector.

A Lagrange multipliers approach, with the added restrictions of orthogonality of different coefficient vectors, can also be used to show that the full set of eigenvectors of S is the solution to the problem of obtaining up to p new linear combinations $Xt_k = \sum_{j=1}^{p} t_{j,k} x_j$, which successively maximize variance subject to uncorrelatedness with previous linear combinations. Such linear combinations Xt_k are called the Principal Component (PC)s of the data set.

The quality of any q-dimensional approximation can be measured by the variability associated with the set of retained PCs. The sum of variances of the p original variables is the trace of the covariance matrix S. Hence, the standard measure of the quality of a given PC is the proportion of the total variance that it accounts for:

$$\pi_j = \frac{\lambda_j}{\sum_{i=1}^{p} \lambda_p}. \tag{2.10}$$

The index is often used to estimate the number of components needed to represent the data set.

The basis of this new representation, called principal components, is orthogonal by construction, as it is the span of eigenvectors of the auto-covariance matrix of

the feature variables. PCA's main advantage in this particular application is that it removes correlated features that do not provide any contribution.

For the time-domain -DSP, we exploit PCA to project the K-dimensional samples in input to the equalizer, i.e. $\boldsymbol{y} = [\boldsymbol{y}_1, \ldots, \boldsymbol{y}_M]$, onto the $\bar{K}$-dimensional version, with $K > \bar{K}$.

2.3.3 Combined cross-correlation/PCA reduction methodology

The basic idea behind the proposed approach is to iteratively apply the cross-correlation and PCA.

The first step is exploiting the cross-correlation analysis to select the signals carrying useful information for the DSP. This step focuses on a reduction technique that prioritizes the transmitted-received signal relationship. Indeed, we select the inputs of the equalizer based on their cross-correlation with the transmitted signal. Then, we obtain a reduced version of $\boldsymbol{y}$ containing only $K_1 < K$ signals of the sequence, that we denote by $\bar{\boldsymbol{y}}^{(1)}$.

This step is lacking because signals are chosen based only on the input-output correlation, without considering their auto-correlation. For instance, a replica of the most input-output cross-correlated signal would be included in the reduced variable even if that signal does not carry any further useful information for the DSP.

For this reason, we design a second stage based on the PCA to project $\bar{\boldsymbol{y}}^{(1)}$ onto a lower dimensional space of uncorrelated features. We denoted the signal after the second reduction stage by $\bar{\boldsymbol{y}}$. In contrast to the cross-correlation stage, PCA focuses on the auto-covariance of the input ignoring the transmitted-received relationship. In this way, we mitigate the disadvantage of each methodology by applying both iteratively.

Finally, we design a new TDE based on the reduced version $\bar{\boldsymbol{y}}$ of the signal leveraging the reasoning in Section 2.2. Fig. 2.2 shows the reduced DSP.

The strategy we propose is summarized in Algorithms 1 and 2.

Algorithm 1 Cross-correlation PCA learning (off-line)

INPUT:
$\boldsymbol{Y} = \{\boldsymbol{y}(kT)\}_{k=N/2}^{D+N/2}$, $\boldsymbol{A} = \{a_{1,k}, \ldots, a_{M,k}\}_{k=N/2}^{D+N/2}, K_1, \bar{K}$
OUTPUT: $\{\boldsymbol{I_l}, \boldsymbol{W_l}\}_{l=1}^{M}$
function LEARNING
    for each channel $l \in [1, \ldots, M]$ **do**
        COMPUTE $R = corr(\boldsymbol{y}, a_i) \otimes corr(\boldsymbol{y}, a_i)^*$,
        $\boldsymbol{I_l} = \text{MAXK}(R, K_1)$
        SET $\bar{\boldsymbol{Y}}^{(1)} = \{\boldsymbol{Y}_i \in [\boldsymbol{Y}_1 \ldots, \boldsymbol{Y}_K] : i \in \boldsymbol{I_l}\}$, i.e. Select the K_1 most cross-correlated inputs
        Compute $\boldsymbol{W_l} = \text{PCA}(\bar{\boldsymbol{Y}}^{(1)}, \bar{K})$, i.e. Learn a $\bar{K}$-dimensional projection of $\bar{Y}$ using PCA
    end for
end function

In particular, based on historical transmission data, Algorithm 1 exploits the cross-correlation analysis to learn a first input projection for each channel, denoted $\boldsymbol{I_l}$. After this process, $\bar{\boldsymbol{Y}}^{(1)}$ consists of the most K_1 cross-correlated signals of the dataset. Then, we design a new projection for each channel based on PCA, called $\boldsymbol{W_l}$

Algorithm 2 Cross-correlation PCA reduction (run-time)

INPUT: $\{\boldsymbol{I_l}, \boldsymbol{W_l}\}_{l=1}^{M}, \boldsymbol{y}(kT)$
OUTPUT: $\{\bar{\boldsymbol{y}}_l(kT)\}_{l=1}^{M}$
function REDUCTION
    Initialize reduced variable: $\bar{\boldsymbol{y}}(kT) = \{\emptyset\}$
    for each channel $l \in [1, \ldots, M]$ **do**
        SET
        $\bar{\boldsymbol{y}}_{\boldsymbol{l}}{}^{(1)}(kT) = \{y_i(kT) \in [y_1(kT), \ldots, y_K(kT)] : i \in \boldsymbol{I_l}\}$,
        i.e. Select the K_1 most cross-correlated inputs
        Compute $\bar{\boldsymbol{y}}_{\boldsymbol{l}}(kT) = \boldsymbol{W_l}\bar{\boldsymbol{y}}_{\boldsymbol{l}}{}^{(1)}(kT)$
        SET $\bar{\boldsymbol{y}}(kT) = [\bar{\boldsymbol{y}}(kT), \bar{\boldsymbol{y}}_{\boldsymbol{l}}(kT)]$
    end for
end function

Algorithm 2 projects the current signal $\boldsymbol{y}(kT)$ to the lower-dimensional space during the run-time.

2.4 Experimental setup description

Fig. 2.3 shows the experimental setup where 2 transmitters were utilized to produce 5 Wavelength Division Multiplexed (WDM) channels spaced at 33.33 GHz and modulated at 30 Gbaud. Four independent Digital-Analog Converter (DAC)s operating at 60 GS/s produced the signals to drive in-phase and quadrature arms of Mach-Zehnder modulators (DN-MZMs). We utilized an External Cavity Laser (ECL) to generate the channel of interest and 4 Distributed Feedback Lasers (DFLs) spaced at 33.33 GHz to generate the surrounding dummy channels. The channel under test and the dummy channels were modulated with two independent signals using 2 different DN-MZMs. A second ECL was used as a local oscillator at the receiver.

A polarization multiplexing emulation stage was set up by introducing a 50 ns delay between the polarizations. To decorrelate the 4 signals to be transmitted into the 4 cores, we split the signal into 4 paths by introducing a 0-300ns delay on each path. 4 solid-state switches were utilized to inject the signals into the recirculating loop and a load switch was introduced to enforce the extinction of the injected signal.

Each loop was composed of 11 concatenated RC4CFs, resulting in a total length of 69.2 km, connected to four two-stage single-mode amplifiers and four Wavelength Selective Switches configured as dynamic gain equalizing filters. Fan-In and Fan-Out for the interface to the 4-core fiber were realized using laser-inscribed 3D waveguides. To precisely control the launch power, 4 Variable Optical Attenuators were introduced before the Fan-In. All 4 single-mode path lengths were matched to $<$1 cm (corresponding to a delay $<$50 ps) to optimize the recirculating loop and the relative launch power was optimized to generate a minimal mode-dependent loss at a distance of 1000 km. Transmitted signals were subsequently extracted from the loops by using the 10% arm of 10:90 couplers to be fed to 4 Polarization-Diverse Coherent Receivers. Finally, a Digital Storage Oscilloscope operating at an 80 GS/s sampling rate, was used to capture the resulting 16 electrical signals.

At the offline processing stage signals were first down-sampled to 2 samples per symbol, then chromatic dispersion and frequency-offset compensation followed by timing identification and an 8×8 MIMO processing, based on a Time-Domain

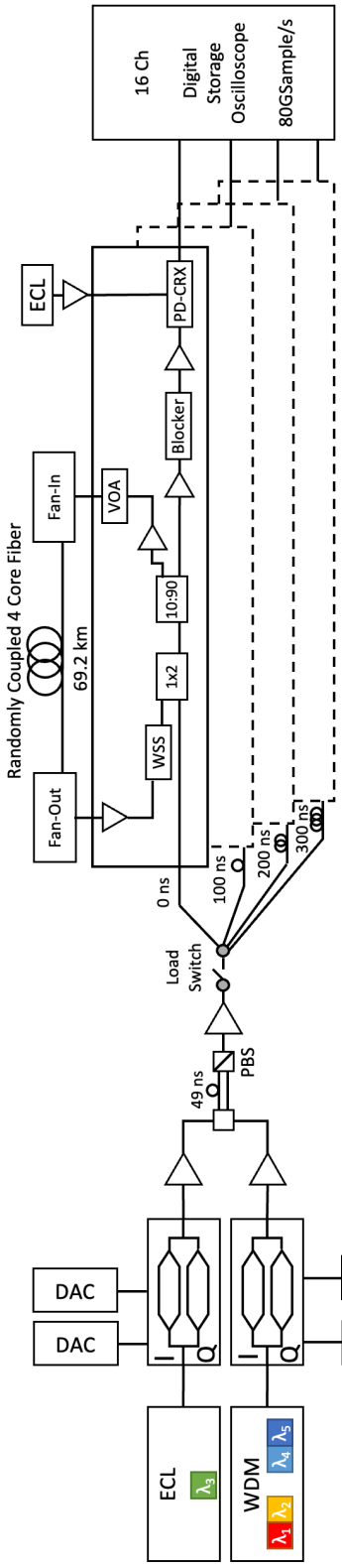

Figure 2.3: Setup for WDM/SDM transmission over randomly coupled 4-core fiber.

Equalizer (TDE). Finally, carrier-phase recovery and Bit Error Rate (BER) counting were performed across all 8 spatial tributaries.

2.5 Results

In this section, we provide a comparison between the results obtained with the classical DSP procedure described in Section 2.2 and the ones obtained with the reduction procedure derived in Section 2.3. Such comparison is performed over the world-first SDM field trial conducted in the INCIPICT testbed [35], where successful transmission in a coupled-core four-cores optical fiber deployed in the underground tunnel network in the historical downtown of the city of L'Aquila, Italy, was demonstrated [36].

In particular, in the following, we first analyze in Section 2.5.1 the Time-Domain DSP performance to justify the need for a reduction approach, and then in Section 2.5.2 we show the improvements produced by the proposed reduction procedure.

2.5.1 Time-Domain Digital Signal Processing

We analyze 8 data sets of a Dual-Polarization QPSK transmission through the four-cores INCIPICT fiber. In a transmission time, 8 symbols are sent, one for each channel. Each data set contains data related to a transmission distance of 0, 69, 138, 692, 1384, 2076, 2760, and 3450 km, and consists of 2961988 symbols sent for each channel. The receiver sampling frequency is twice the transmitter: so, we have two received samples for each symbol sent.

Up to 400×10^3, symbols are used for the initial pre-converge with the DA-LMS algorithm. After pre-convergence, the equalizer adaptation is based on the CMA. The equalized signals experienced a Carrier phase recovery stage based on a phase-locked loop (PLL). Finally, the symbols are decoded. The last 10^6 bits are used for BER estimation.

In this work, we only analyzed the DSP performance on the first channel. Table 2.1 shows the TDE performance of the first channel when the signal covered a distance included in the interval $[0, 3450]$km.

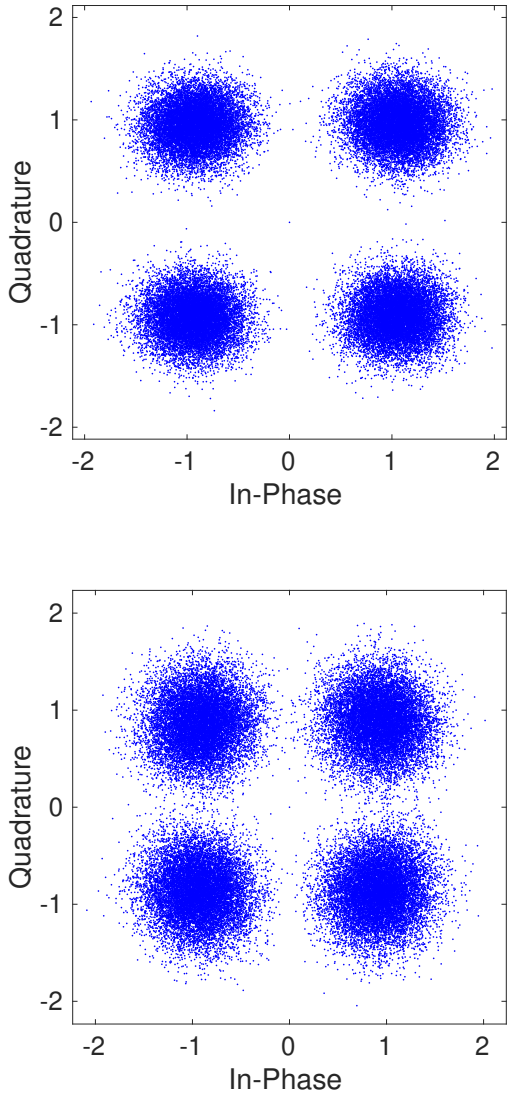

Figure 2.4: Signal after the time-domain digital signal processing. The upper figure shows the signal after covering 692km, the lower shows the signal after 3450km.

Fig. 2.4 shows the synchronized signal after covering 692km and 3450km.

As shown in Table 2.1, the problem complexity (taps number) increases with the number of kilometers covered by the signal. The equalizer mitigates the distortion increasing the window length of the input signals. The idea behind the proposed reduction procedure is to reduce the number of inputs needed by the Time-Domain equalizer.

km	0	69	138	692	1384	2076	2760	3450
Loops	0	1	2	10	20	30	40	50
N	8	16	24	170	300	300	600	1000
K	144	323	400	2576	4816	4816	9616	16016
BER	0	0	0	0	1e-5	1.1e-3	2.8e-4	2.2e-3

Table 2.1: Bit Error Rate of the first channel performed by Time-Domain DSP. In the first column, km indicates the distance in kilometers covered by the signals, Loops counts the complete turns of the INCIPICT ring covered by the signals, N is the window length considered by the equalizer, and K is the tap equalizer number. We achieve a BER of order equal to, or lower, than 10^{-3} in each case. This demonstrates that the MIMO processing is capable of mitigating the crosstalk distortion.

2.5.2 Reduction approaches

For conciseness, we only report the cases of signals that have covered 692km and 3450km into the optical ring.

To obtain a less complex DSP able to achieve a BER of 2×10^{-3}, we reduced for the 629 km case the equalizer window length to $N = 120$. Starting from this point, we applied the reduction technique aiming at further reducing the equalizer complexity without compromising the accuracy.

Differently, for the 3450 km case, we increased the equalizer window length to $N = 1800$. In this way, through the correlation technique, we are already able to obtain a better BER, with respect to the TDE, with the same number of taps (see Table 2.3). Then, we apply the reduction technique to decrease equalizer complexity.

We start our experiments considering the cross-correlation between the transmitted signal and the input of the equalizer. We perform both the principal component and the cross-correlation analysis based on the first 500×10^3 samples sequence of each data set.

Fig. 2.5 shows the modulus of the cross-correlation between the transmitted signal and the samples analyzed by the equalizer for the 692km and 3450km cases. It is possible to appreciate not-flat cross-correlation distribution. More precisely, the upper figure shows that signals from 0 to approximately 700 have fewer cross-correlations than the signals in the range $[700, 1936]$. In a similar way, signals

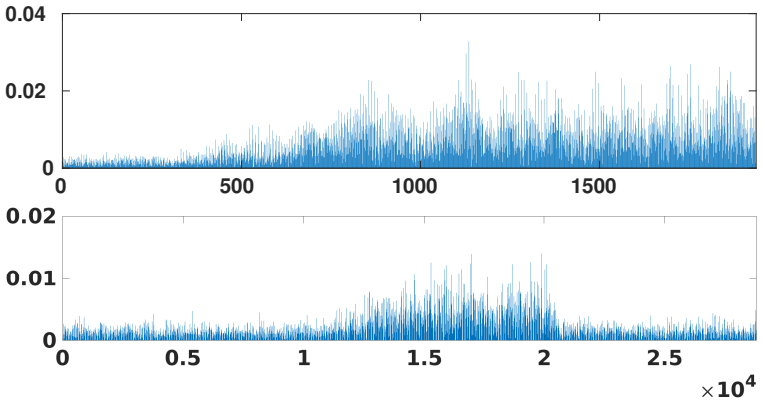

Figure 2.5: Cross-correlation between the transmitted signal and the samples analyzed by the equalizer. The upper figure shows the signal after covering 692km, the lower shows the signal after 3450km.

around 1200 and 2000 are more correlated with the output. Based on this reasoning, we aim to design a DSP based only on the most correlated signals.

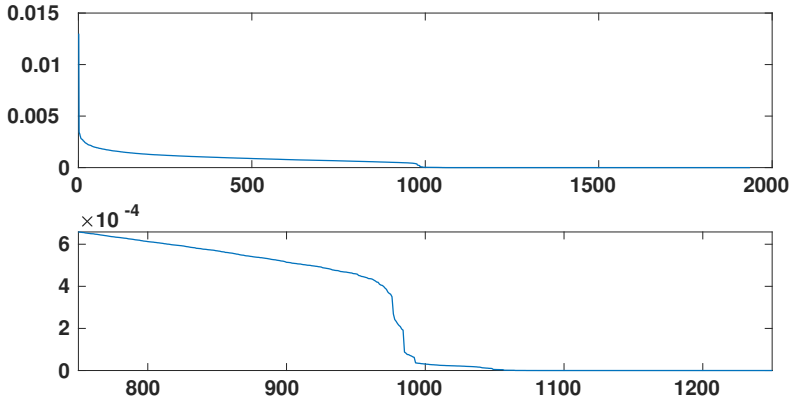

Figure 2.6: PCA portion of the total variance of the input equalizer for the case 692km. The upper figure shows the portion of total variance for each component, the lower shows the components in the range $[750, 1250]$.

Fig. 2.6 shows the PCA portion of the total variance of the equalizer's input for the case 692km. As exposed in Section 2.3.2, the metric is often used to identify the number of components needed to explain the data set. The lower plot in Fig. 2.6 emphasizes that components after the 1000^{th} contribute significantly less to the total variance. The very high proportion of variability explained by the first 1000^{th} principal components provided a solid ground for our thesis about the importance

of the signals in the DSP. Indeed, rapid declines in the portion of total variance are associated with high-correlated inputs carrying useless information for the DSP.

Based on the result exposed in Fig. 2.5 and 2.6, we analyze the importance of each input in the equalization process to prove the existence of samples that carry more information than others. From such results, we design a methodology to pick the samples useful to decrease the equalizer complexity.

N	$\bar{\mathbf{K}}$	**% of** K	**TD-DSP**	**Corr**	**PCA**	**Corr + PCA**
120	1936	100%	0.002			
108	1744	90%	0.015	0.002	0.002	
120	1552	80%	-	0.002	0.002	
120	1360	70%	-	0.0022	0.002	
120	1168	60%	-	0.0027	0.002	
120	976	50%	-	0.0045	0.0027	
120	784	40%	-	-	-	0.0027

Table 2.2: The bit error rate of the first channel performed by the TD-DSP with the reduced input case 692km. From the 3^{th} row, each row shows a reduced DSP with a different number of inputs. The table provides the considered window length in the first column, the input number to the equalizer in the second column, the percent reduction with respect to the classic approach (in the 2^{th} row) in the third column, the BER performed for a classic equalizer with the input number in column 1 in the fourth column, and respectively the reduced-DSP BER exploiting the cross-correlation, principal component and both the analysis performed iteratively in the last three columns.

For the 692km case, our goal was to choose the lowest number of taps $\bar{K}$ of the reduced TDE to guarantee a BER of order 10^{-3}. To this aim, we first reduced N of the original TDE until the BER increased to 10^{-3}, and then we applied the 2 reduction methodologies, i.e. correlation and correlation with the PCA. This step is to better appreciate the impact of our methodology; indeed, in this critical point, it is not possible to decrease the classical equalizer window length without compromising the DSP performance. As shown in Table 2.2, both methodologies aim at reducing the number of inputs of the equalizer. The best performance in terms of BER is obtained when the TD-DSP process applies both methodologies iteratively: 7^{th} column, "Corr+PCA". We designed the "Corr+PCA" DSP selecting 976 signals based on the cross-correlation analysis first, and then exploiting the PCA, thus performing an additional reduction. Note that in the second row of

N	$\bar{K}$	% of K	TD-DSP	Corr	Corr+PCA
1000	16016	100%	0.002		
1800	16016	100%		0.001	
1800	14414	90%		0.001	
1800	11211	70%		0.001	
1800	9609	60%		0.001	
1800	8008	50%		0.0017	0.0016

Table 2.3: The bit error rate of the first channel performed by the TD-DSP with the reduced input case 3450km. From the 3^{th} row, each row shows a reduced DSP with a different number of inputs. The table provides information equivalent to Table 2.2.

Table 2.2 we decreased the classical equalizer window-length by 10% obtaining an unacceptable BER of 1×10^{-2}. This confirms the necessity of advanced techniques to perform the reduction. In this case, we obtained a result in terms of BER equal to 2.7×10^{-3}, and a reduction in the number of inputs by the 60%.

For the 3450km case, we aim at reducing the taps number of the equalizer guaranteeing a BER of the order of 10^{-3}. In such experiments, we increased the window length of the equalizer N to 1800, and as a consequence the number of input signals. Then, we applied the correlation analysis to select the most correlated signals to be used in the DSP. Finally, we performed both the correlation and PCA iteratively, as illustrated above. As shown in Table 2.3, the best performance is obtained when the TD-DSP process applies our combined methodology: 6^{th} column, "Corr+PCA". We designed the "Corr+PCA" by selecting 11211 signals based on the cross-correlation and then exploiting the PCA.

Finally, for the sake of completeness, we show in Fig. 2.7 the synchronized signal after covering 692km and 3450km exploiting the "Corr+PCA" TD-DSP.

2.6 Conclusions

In this chapter, we presented a taps reduction methodology that merges the cross-correlation and PCA, and discussed its impact in terms of BER of the DSP algorithm. We validated the proposed approach on the data of the world-first SDM multi-core fiber field trial conducted within the INCIPICT project in the city of L'Aquila, Italy. Results showed a reduction in the number of taps of the 60% and 50%

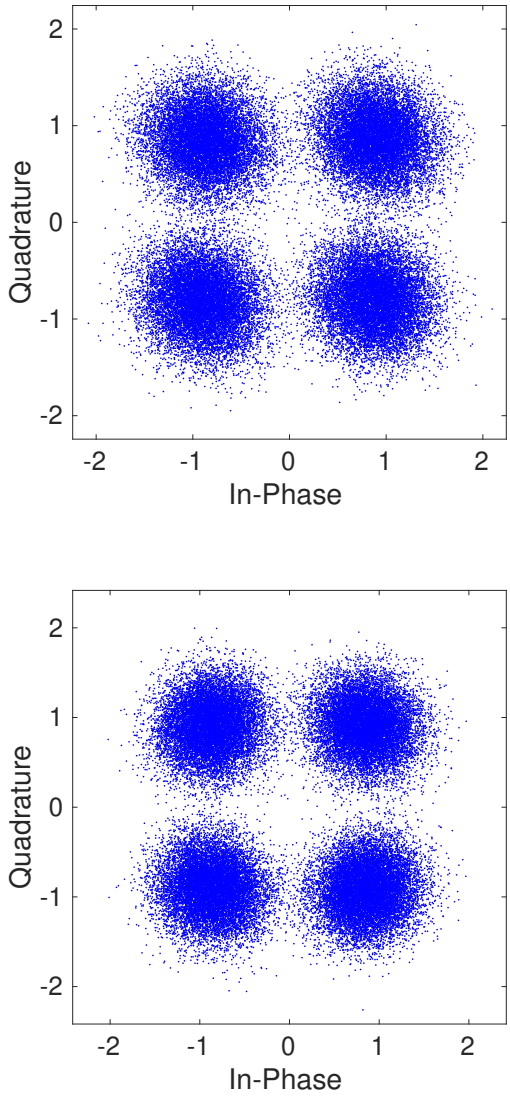

Figure 2.7: Signal after the reduced time-domain digital signal processing. The upper figure shows the signal after covering 692km, the lower shows the signal after 3450km.

concerning the classical approach proposed in [48] considering a transmission over 650 and 3450 km respectively.

In future work, we plan to further reduce the complexity of the DSP algorithm extending the proposed approach through the combination with other methodologies, such as digital subcarriers and Frequency-domain DSP. Furthermore, we plan to investigate the impact of adaptive reduction methodologies aiming to guarantee a minimum BER over time.

3

Learning Markov models of fading channels in wireless control networks: a regression trees based approach

Contents

Finite-state Markov models are widely used for modeling wireless channels affected by various non-idealities, ranging from shadowing to interference. In an industrial environment, the derivation of a Markov model based on wireless communication physics can be prohibitive as it requires complete knowledge of both the communication dynamics parameters and the disturbances/interferers. This work proposes a novel methodology to learn a Markov model of a fading channel via historical data of the Signal-to-Interference-plus-Noise Ratio (SINR). Such methodology can be used to derive a Markov jump model of a wireless control network, and thus to design a stochastic optimal controller that considers

the interdependence between the plant and the wireless channel dynamics. The proposed method is validated by comparing its prediction accuracy and control performance with those of a stationary finite-state Markov chain derived assuming perfect knowledge of the physical channel model and parameters of a WirelessHART point-to-point communication based on the IEEE-802.15.4 standard.

The discussion in this chapter has already been presented in [38].

3.1 Introduction

Wireless networked control system (WNCS)s are composed of spatially distributed sensors, actuators, and controllers communicating through wireless networks [29]. Despite their success in industrial monitoring applications, existing wireless sensor-actuator network technologies face significant challenges in supporting control systems due to their lack of real-time performance and dynamic wireless conditions in industrial plants [51]. A key challenge in WNCS design is channel modeling in an industrial environment because of its inherent complexity [51] [52]. These communication channels are frequently subject to time-varying fading and interference, which may lead to packet losses.

From the automatic control perspective, an example of WNCS consists of a nonlinear process with intermittent control packets due to the lossy communication channel described by the following equations:

$$\begin{cases} y\left(k+1\right) = f\left(y(k), u_a(k)\right), k \in \mathbb{N} \\ u_a(k) = \nu(k)u(k) \\ y(0) = y_0 \in \mathbb{R}^{n_y} \end{cases} \tag{3.1}$$

where $y(k) \in \mathbb{R}^{n_y}$ is the output of the system and $u(k) \in \mathbb{R}^{n_u}$ is the input signal. $\{\nu(k)\}_{k\in\mathbb{N}}$ is a discrete-time Boolean process modeling the packet delivery of the control signals: if the packet is correctly delivered then $\nu(k) = 1$, otherwise if it is lost then the actuator does nothing $\nu(k) = 0$. The controlPacket Error Probability (PEP) $\mathbb{P}(\nu(k) = 0)$, depends on the SINR of the communication $\Gamma(k)$, i.e.,

$$\mathbb{P}(\nu(k) = 0) = g\left(\Gamma(k)\right), \tag{3.2}$$

where $g : \mathbb{R} \to [0, 1]$ is a deterministic function defined by the communication standard. We assume full-state observation with no measurement noise and no observation packet loss. In this work, we consider a channel model based on WirelessHART [1], an on-the-market wireless communication standard specifically designed for process automation. We assume that the value of the SINR $\Gamma(k)$ is measurable and is sent to the controller via an Acknowledgment (ACK). However, this ACK is available only after the current decision on the control input to apply has been made and sent through the link since the actual success of the transmission is not known in advance. Hence, at each time k the controller receives measurements of $y(k)$ and $\Gamma(k-1)$, as depicted in Fig. 3.1.

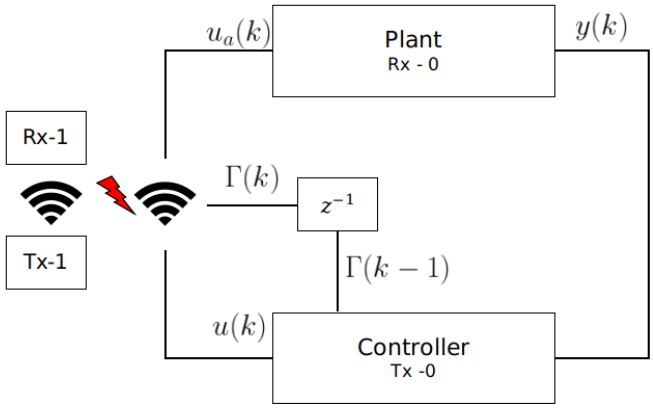

Figure 3.1: Wireless networked control system

Since $\Gamma(k)$ is represented by a generic stochastic process, the obtained model may not be computationally tractable when it is derived for wireless communications in an industrial environment (see Sec. 3.2), especially if the objective is to apply optimal control algorithms (e.g. model predictive control – MPC). For this reason, a preliminary investigation of channel model abstraction is fundamental for controller design. In the WNCS literature, the packet dropouts have been modeled as stochastic or deterministic phenomena. For what concerns stochastic models, a vast amount of research assumes memoryless packet drops so that dropouts are realizations of a Bernoulli process. In [53], the packet delivery process is modeled as a Bernoulli random process, then an optimal controller is derived. In [54] a

Markov chain model is used to derive an accurate abstraction of the WirelessHART channel, and it is proven that such a model allows for characterizing the stability of a WNCS for the scenarios where the simple Bernoulli-like channel models (which cannot model packet bursts) fail.

Markov models are a powerful tool for modeling stochastic random processes. They are general enough to model with high accuracy a large variety of processes and are relatively simple, allowing us to compute analytically many important parameters of the process which are very difficult to calculate for other models [55]. Hidden Markov models ([56] [57]) have been exploited to learn channels models in [52] and [58]. In [59], the authors expose the benefit of exploiting finite-state Markov chains to model the behavior of wireless fading channels. In [60], the authors derive a Markov chain that estimates the PEP of an industrial wireless protocol, and then propose an optimal stochastic controller for linear systems.

Inspired by the challenges in [60] related to the derivation of a physics-based Markov chain abstraction of the wireless channel, the main contribution of this chapter is a novel data-driven methodology, based on Regression Tree (RT)s [12], to identify such Markov chain abstraction. More precisely, we propose a novel methodology to model the PEP $\nu(k)$ based on a Markov chain $\theta(k)$: each state of $\theta(k)$ is associated with a partition of $\mathbb{R}^{n_y+1}$ consisting of rectangular sets $\{R_i\}_{i=1}^{\ell}$, each representing the range of possible values of $\Gamma(k) \in \mathbb{R}$ and $y(k) \in \mathbb{R}^{n_y}$ at time k. We construct the Transition Probability Matrix (TPM) of $\theta(k)$ as follows:

$$\begin{aligned} p(j \mid i) &\doteq \mathbb{P}(\theta(k+1) = j \mid \theta(k) = i) \\ &= \mathbb{P}(\theta(k+1) = j \mid (\Gamma(k), y(k)) \in R_i) \\ &= h(\Gamma(k), y(k)), \end{aligned} \tag{3.3}$$

where identifying from historical data the function $h : \mathbb{R}^{n_y+1} \to [0, 1]$, which depends on the current measurements $\Gamma(k)$ and $y(k)$, is the objective of this chapter. Note that, given any two states i, j of the Markov chain, $p(j \mid i)$ also depends on the plant output: indeed, $y(k)$ may for example determine the distances between the transmitter, the receiver, and the interferer, and, as illustrated in Sec. 3.2, this

strongly affects the dynamics of $\Gamma(k)$. Finally, the PEP can be easily computed using (3.2), which in this chapter is based on a point-to-point WirelessHART communication based on the IEEE 802.15.4 standard.

Nevertheless, we remark that our data-driven methodology is independent on the transmission technology and can be replicated for any communication protocol if the corresponding wireless channel can be effectively modeled by a Markov chain.

In summary, in an industrial environment, the derivation of a Markov model based on wireless communication physics can be prohibitive as it requires complete knowledge of both the communication dynamics parameters and the disturbances/interferers. Our methodology has 3 main advantages: (1) it only exploits historical data and hence does not require prior knowledge of the system and channel parameters. Moreover, most physics-based approaches cannot handle time-varying parameters, as the computational complexity of the obtained abstraction would be intractable, while with our methodology we consider a dependency between the parameters of the communication system and the dynamics of the plant which is still tractable in terms of computational complexity; (2) as a byproduct of leveraging our techniques in [61], and beyond the main contribution of identifying the Markov chain abstraction of $\Gamma(k)$, we also construct a Switching Auto-Regressive eXogenous (SARX) model for the nonlinear plant f in (3.1), i.e. our methodology does not require any linearity assumption on the plant's dynamics; (3) the obtained identified models, both for $\Gamma(k)$ and f, can be combined obtaining a Markov jump system, which can be directly used to setup a classical MPC problem that can be solved very efficiently, i.e. using Quadratic Programming (QP).

3.2 Channel modeling

In this section we first describe the channel model under consideration, then we present the Markovian model for the wireless link developed in [60], and finally we introduce an analytical model that will be useful to experiment the data-driven approach we propose in this chapter.

WirelessHART physical model

We analyze the industrial environment described in Fig. 3.1, wherein wireless communications are affected by interference. We study a point-to-point transmission based on the WirelessHART protocol, i.e. the IEEE 802.15.4-2006 defined in [62], interfered by another WirelessHART transmission. The proposed model considers the effect of path loss, shadow fading, and the residual power fluctuations left by the power control. The effect of multipath fading is supposed to be compensated by the aforementioned power control. We denote with $i=0$ the reference Transmitter-Receiver (Tx-Rx) link, and with $i=1$ the link between the interferer Tx and the reference Rx.

The effect of the path loss model is defined in [62]. For a system with bandwidth $W=2.4$GHz the path loss coefficient of the link between transmitter i and the reference receiver (i.e., a mobile plant in our case) is $\alpha_i(k)=10^{-\frac{\varsigma(d_i(k))}{10}}$, where

$$\varsigma(d_i(k)) = \begin{cases} 40.2 + 20\log_{10}(d_i(k)), \text{ if } d_i(k) \leq 8, \\ 58.5 + 33\log_{10}(\frac{d_i(k)}{8}), \text{ otherwise;} \end{cases} \tag{3.4}$$

and $d_i(k)$ is the length of the link i at time instant k, i.e. a distance in meters that may e.g. depend on the position of the plant (see the inverted pendulum on a cart in Sec. 3.6).

The shadow fading is modeled following [63] by assuming a log-normal model for each link i, which introduces a multiplicative factor $e^{\beta_i(k)}$, where $\beta_i(k)$ is a zero-mean Gaussian process with variance $\sigma^2_{\beta_i}$ and auto-covariance function $c_{\beta_i}(\tau)$, with τ being a time lapse between two consecutive (time-driven) control packets. We remark that $c_{\beta_i}(\tau)$ may also depend on the state of the plant (e.g. the speed of a cart) thus exhibiting a time-varying behavior.

For each link i the residual Power Control Error (PCE) is also modeled as a log-normal process, $e^{\xi_i(k)}$, where $\xi_i(k)$ is a zero-mean Gaussian process with variance $\sigma^2_{e_i}$ and auto-covariance $c_{\xi_i}(\tau)$.

By considering the characteristics of the Offset Quadrature Phase-Shift Keying (OQPSK) with Direct-Sequence Spread Spectrum (DSSS) modulation, as specified

in [62], we can derive the power value of the SINR, $\gamma(k)$, at time k as in [60]:

$$\gamma(k) = \sqrt{\frac{P_0(k)\alpha_0^2(k)e^{\beta_0(k)+\xi_0(k)}}{\frac{N_0}{4} + \frac{8}{3G}P_1(k)\alpha_1^2(k)e^{\beta_1(k)+\xi_1(k)}}}, \tag{3.5}$$

where P_0 and P_1 are respectively the reference user transmission power and the interferer transmission power, N_0 is the noise spectral density, and $G = WT_s$ is the processing gain, with T_s being the symbol time. In the rest of this chapter, we denote as $\Gamma(k) = 10\log_{10}(\gamma(k))$ the SINR in decibels.

In WirelessHART the forward error correction is not implemented, thus even one erroneous bit leads to a corrupted WirelessHART data packet. For this reason, the packet error probability R_p is related to the SINR as follows

$$\begin{aligned} R_b(\gamma(k)) &= \frac{1}{30}\sum_{\iota=2}^{16}(-1)^{\iota}\binom{16}{\iota}\exp\left(20\cdot\gamma(k)\frac{1-\iota}{\iota}\right) \\ R_p(\gamma(k)) &= 1-\left(1-R_b\left(\gamma\left(k\right)\right)\right)^{l_f}, \end{aligned} \tag{3.6}$$

where l_f is the number of bits in a frame, and R_b is the bit error rate computed according to [62]. It is worth remarking that the distance Tx-Rx $d_i(k)$ influences the SINR $\gamma(k)$ that, in turn, influences the packet error rate. This will be useful for the discussion of the simulation results in Sec. 3.6.

The main issue with the above channel model is that, despite the ability to describe the SINR, in the majority of the control applications equation (3.5) is intractable, both in the continuous-time and discrete-time form. For this reason, in [60], a Markov chain model to derive a discrete-time abstraction of (3.5) has been proposed. We recall such a technique in the following subsection as a comparison for the proposed method in this chapter.

Finite-state Markov chain

It is straightforward to see that (3.5) can be expressed as a weighted linear combination of correlated log-normal processes: hence there is no exact explicit closed-form expression of its distribution. We can use the moment matching technique [64] to approximate the probability distribution of (3.5) by a log-normal process, thus presenting $\Gamma(k)$ as a Gaussian process with mean $\mu_\Gamma(k)$, variance $\sigma_\Gamma^2(k)$

and auto-covariance $c_\Gamma(\tau)$, as detailed in [60]. Clearly, due to the dependence of the SINR on the state of a mobile plant through, e.g. the parameters $d_i(k)$ (which define the path loss coefficients $\alpha_i(k)$ via (3.4)), the moment-matching approximation should be made for each relevant value of the aforementioned parameters. In the rest of this subsection we will focus on a Gaussian process $\hat{\Gamma}(k)$, which is a moment-matching approximation of the SINR with arbitrary values of the parameters in (3.5) corresponding to any given state of the plant, i.e. for $\alpha_i(k)=\hat{\alpha}_i$, $\beta_i(k)=\hat{\beta}_i(k)$, and $P_i(k)=\hat{P}_i$ in (3.5), with $i=0,1$, we have that $\hat{\Gamma}(k) \approx 10\log_{10}(\gamma(k))$.

At this point, to obtain a finite-state Markov channel abstraction, the first step is to divide the range of $\hat{\Gamma}(k)$ into several consecutive regions, each associated with a certain representative PEP. Specifically, a region r of the values of the SINR is mapped into a state s_r of the related Markov chain, and it is delimited by two thresholds ζ_r and ζ_{r+1} belonging to the set of extended reals. These SINR thresholds are determined by the chosen partitioning method [59]. In this chapter, we rely on a well-known equiprobable partitioning, where the thresholds are selected in such a way that the steady-state probabilities of being in any state are equal.

Then, the steady-state probability $\boldsymbol{p_r}$ of a state s_r is defined as the probability that the value of $\hat{\Gamma}(k)$ is between the two thresholds of the region, and it is given by

$$\boldsymbol{p_r} = \int_{\zeta_r}^{\zeta_{r+1}} \phi_{\mathcal{N}}(\zeta; \mu_{\hat{\Gamma}}, \sigma^2_{\hat{\Gamma}}) d\zeta, \tag{3.7}$$

where $\mu_{\hat{\Gamma}}$ and $\sigma^2_{\hat{\Gamma}}$ are respectively the mean and variance of $\hat{\Gamma}(k)$, and $\phi_{\mathcal{N}}(\zeta; \mu, \sigma^2)$ is the Probability Density Function (PDF) of a Gaussian Random Variable (GRV) $\mathcal{N}(\mu, \sigma^2)$.

The Power Delay Profile (PDP) associated with the same state within the respective region is given by

$$1 - \nu_M^{(r)} = \frac{1}{\boldsymbol{p_r}} \int_{\zeta_r}^{\zeta_{r+1}} R_p(10^{\frac{\zeta}{10}}) \phi_{\mathcal{N}}(\zeta; \mu_{\hat{\Gamma}}, \sigma^2_{\hat{\Gamma}}) d\zeta. \tag{3.8}$$

Finally, the channel state transition probabilities are derived from integrating the joint PDF of the SINR $\hat{\Gamma}(k)$ over two consecutive packet transmissions and

over the desired regions, r, and q, as

$$p(q \mid r) = \frac{1}{\boldsymbol{p_r}} \int_{\zeta_r}^{\zeta_{r+1}} \int_{\zeta_q}^{\zeta_{q+1}} \varphi_{\mathcal{N}}\Big(\varsigma_{k-1}, \varsigma_k; \mu_{\hat{\Gamma}}, \sigma^2_{\hat{\Gamma}}, c_{\hat{\Gamma}}(\tau)\Big) d\varsigma_{k-1} d\varsigma_k,$$

where $\varphi_{\mathcal{N}}(\varsigma_{k-1}, \varsigma_k; \mu_{\hat{\Gamma}}, \sigma^2_{\hat{\Gamma}}, c_{\hat{\Gamma}}(\tau))$ is the two-dimensional PDF of the Gaussian process $\hat{\Gamma}(k)$, as detailed in [60].

However, one issue related to the above technique is that in real (industrial) cases some of the channel parameters required for the above modeling are time-varying (due to their dependence on the state of the plant) and often only partially known. For these reasons, we propose a new approach to model the SINR, and thus the PEP. In this respect, a promising direction is the exploitation of historical data of the communication channel to identify a model via system identification and machine learning techniques.

Auto-regressive model for channel simulation

The application of data-driven methodologies requires the existence of a dataset containing trajectories of the SINR: in this chapter, we run Monte Carlo simulations of the wireless transmission model (3.5) and then apply our techniques to such trajectories. To this aim, in this section we illustrate the approach presented in [65], where the study of auto-regressive stochastic models for computer simulation of fading channels is addressed, to derive discrete-time trajectories of the process described in equation (3.5).

In particular, let us consider a discrete-time Gaussian process $\{z(k)\}_{k\in\mathbb{N}}$ with Auto-Correlation Function (ACF) $R_{zz}(n)$. We can derive an Auto-Regressive (AR) model of the following form that is able to generate trajectories of the process:

$$z(k) = -\sum_{n=1}^{p} a_n z(k-n) + w(k), \tag{3.9}$$

where $w(k)$ is a zero-mean white Gaussian noise process. The AR model parameters consist of coefficients $\{a_1, \ldots, a_p\}$ and variance σ_p^2 of the driving noise $w(k)$. To estimate the coefficients a_j, $j = 1, \ldots, p$, once the ACF R_{zz} is fixed from β_i and ξ_i, $i = 0, 1$, the relationship between R_{zz} and a_j is given as follows [66]:

$$R_{zz}(n) = \begin{cases} -\sum_{m=1}^{p} a_m R_{zz}(n-m), & n \geq 1 \\ -\sum_{m=1}^{p} a_m R_{zz}(-m) + \sigma_p^2, & k = 0 \end{cases} \tag{3.10}$$

Finally, coefficients a_j can be determined by solving the set of Yule-Walker equations that can be easily derived from (3.10) (we refer the reader to [66] for further details).

The generated AR process has the following auto-correlation function:

$$\hat{R}_{zz}(n) = \begin{cases} R_{zz}(n), & 0 \leq n \leq p \\ -\sum_{m=1}^{p} a_m \hat{R}_{zz}[n-m], & n > p \end{cases} \tag{3.11}$$

The simulated process has the attractive property that its sampled ACF perfectly matches the desired sequence of ACF up to lag p. Therefore, since we know the ACF of both the residual power control error $\xi_i(k)$ and the shadowing correlation $\beta_i(k)$ we can exploit the AR model to generate sequences of $\xi_i(k)$ and $\beta_i(k)$ and then apply (3.5) to obtain the discrete-time trajectories of $\Gamma(k)$.

3.3 The CART algorithm

The aim of this section is to provide a short description of the Classification And Regression Trees (CART) algorithm [12], in order to provide the basic notions to present our method in Sec. 3.5 that identifies a Markov chain wireless channel abstraction.

In a supervised framework, we consider a predictor dataset $\mathcal{P} = \{\lambda(k)\}_{k=1}^{D}$ and a response dataset $\mathcal{R} = \{\rho(k)\}_{k=1}^{D}$ of D samples each, where $\rho(k) \in \mathbb{R}$ is called response variable and $\lambda(k) \in \mathbb{R}^n$ is called predictor variable. The final goal of CART is to identify a function $\mathcal{T}$ to estimate $\hat{\rho}(k) = \mathcal{T}(\lambda(k))$.

In the specific case of the CART algorithm [12], the dataset is partitioned into a set of hyper-rectangles $R_1, \ldots, R_\ell$, corresponding to the ℓ leaves of the tree. Then, $\hat{\rho}(k)$ is estimated in each leaf τ_i using a constant c_{τ_i} given by the average of the samples in the partition. Without any loss of generality, we restrict our attention to recursive binary partition. Due to space limitations, we only briefly recall the partitioning algorithm of CART and refer the reader to [12] for more details.

The CART algorithm creates the partition using a greedy algorithm to optimize the split variables and split points: starting with the whole dataset, consider a split variable j over the n available and a split point s, and define the pair of half-planes as $R_L(j,s) = \{\lambda(k) \mid \lambda_j(k) < s\}$ and $R_R(j,s) = \{\lambda(k) \mid \lambda_j(k) \geq s\}$. Then, CART solves the following optimization problem to find the optimal j and s

$$\min_{j,s} \left[\min_{c_L} \sum_{\lambda(k) \in R_L(j,s)} (\rho(k) - c_L)^2 + \min_{c_R} \sum_{\lambda(k) \in R_R(j,s)} (\rho(k) - c_R)^2 \right], \tag{3.12}$$

and for any choice of j and s the inner minimization is solved by $c_L = \text{ave}(\rho(k) \mid \lambda(k) \in R_L(j,s))$ and $c_R = \text{ave}(\rho(k) \mid \lambda(k) \in R_R(j,s))$, where ave$(\cdot)$ is the arithmetic mean of the output samples. In other words, the optimal j^* and s^* minimize the sum of the quadratic prediction errors of the left and right partitions induced by the split variable and split point.

For each splitting variable, the determination of the split point s can be done very quickly and hence, by scanning through all of the inputs, the determination of the best pair (j,s) is feasible. Once the best split is found the dataset is partitioned into the two resulting regions, then the splitting procedure is repeated on each of the two regions. The process is repeated on all of the resulting regions until a stopping criterion is applied, e.g. tree size is a tuning parameter chosen to avoid overfitting and variance phenomena.

In the rest of this work we denote with $\mathcal{T}$ the regression tree, with τ_i the i^{th} leaf of $\mathcal{T}$, with $|\mathcal{T}|$ the number of leaves of $\mathcal{T}$, with $|\tau_i|$ the number of samples in τ_i, with $c_{\tau_i} = \text{ave}(\rho(k) | \lambda(k) \in \tau_i)$ the prediction of leaf τ_i and, with a slight abuse of notation, with $\mathcal{T}(\lambda)$ the prediction of the regression tree, i.e.

$$\mathcal{T}(\lambda) = \sum_{\tau_i \in \mathcal{T}} \sum_{\lambda(k) \in \tau_i} \frac{\rho(k)}{|\tau_i|} I\{\lambda \in \tau_i\} = \sum_{\tau_i \in \mathcal{T}} c_{\tau_i} I\{\lambda \in \tau_i\}, \tag{3.13}$$

where $I\{\lambda \in \tau_i\}$ is the indicator function, which is equal to 1 if $\lambda \in \tau_i$ and 0 otherwise.

3.4 Switching ARX Identification

As discussed above, in this chapter we leverage the techniques in [61] to construct a SARX model for the nonlinear plant f in (3.1) that can be directly used to set up an MPC problem, as will be done in Sec. 3.6. In particular, starting from a dataset $\mathcal{D} = \{(y(k), u(k))\}_{k=1}^{D}$ of D samples collected from the measurements of a physical system, respectively consisting of outputs $y(k) \in \mathbb{R}^{n_y}$ and inputs $u(k) \in \mathbb{R}^{n_u}$, we will derive for each $j = 0, \ldots, N-1$ a model as follows:

$$x(k+j+1) = A_{\sigma_j(x(k))} x(k+j) + B_{\sigma_j(x(k))} u(k+j) + F_{\sigma_j(x(k))},$$

with $\sigma_j : \mathbb{R}^{n_x} \to \mathcal{M} \subset \mathbb{N}$ the switching signal,

$$x(k) \doteq \left[y^\top(k) \ \cdots \ y^\top(k-\delta_y) \ u^\top(k-1) \ \cdots \ u^\top(k-\delta_u)\right]^\top \in \mathbb{R}^{n_x}$$

the state consisting of the regressive terms of the inputs and the outputs, $\delta_y, \delta_u \geq 0$, and $n_x = (\delta_y + 1)n_y + n_u\delta_u$.

3.5 Markov model based on regression trees

In industrial environments, the derivation of Markov models based on the physics of wireless communication can be prohibitive as it requires complete knowledge of both the communication dynamics parameters and the disturbances/interferers. Furthermore, the presence of time-varying parameters can increase the complexity of the obtained model. We propose a novel methodology to derive a Markov model of the PDP based on the WNCS measurements. The approach exploits the transmissions' historical data to deal with the above-mentioned circumstances. In particular, we handle the case wherein there is a dependency between the communication system and the plant's measurable outputs.

The main idea is to derive a Markovian model $\{\theta(k) \in \Theta\}_{k\in\mathbb{N}}$ abstracting the SINR stochastic process $\Gamma(k)$, with TPM P as in (3.3), and associate to each state $i \in \Theta$ a PDP. The learning procedure consists of three steps: (1) we grow a regression tree $\mathcal{T}$ with predictor variables the current SINR and the plant measurements, i.e. $(\Gamma(k), x(k))$, and with response variable the SINR at the next time step, i.e. $\Gamma(k+1)$. Since the leaves of $\mathcal{T}$ form a partition of the predictor space, we define the state-space of $\theta(k)$ associating to each element of the partition $\{R_{\tau_i}\}_{i=1}^{\ell_\mathcal{T}}$ a state of the Markov chain, i.e. $\Theta \doteq \{i \in \mathbb{N} : \tau_i \in \mathcal{T}\}$; (2) we grow a regression tree Π with predictor variables the current SINR, i.e. $\Gamma(k)$, and with response variable the prediction of the SINR at the next time step obtained using the regression tree $\mathcal{T}$, i.e. $\mathcal{T}(\Gamma(k), x(k))$. We will combine Π and $\mathcal{T}$ to identify the TPM of $\{\theta(k)\}_{k\in\mathbb{N}}$; (3) we associate to each state of $\{\theta(k)\}_{k\in\mathbb{N}}$ a PDP.

Step 1

We grow a regression tree $\mathcal{T}$ using as predictor dataset $\{\Gamma(k), x(k)\}_{k=1}^{D} = \mathcal{P}_\mathcal{T} \subset \mathbb{R}^{n_x+1}$, consisting of the current SINR and the plant measurable outputs, and as response dataset $\{\Gamma(k+1)\}_{k=1}^{D} = \mathcal{R}_\mathcal{T} \subset \mathbb{R}$, consisting of the SINR at the next time step. Let $\mathcal{T}((\Gamma(k), x(k)))$, with $\mathcal{T} : \mathcal{P}_\mathcal{T} \to \mathcal{R}_\mathcal{T}$, be the prediction of $\mathcal{T}$ given the current measurements. As mentioned above, we associate to each element of the partition $\{R_{\tau_i}\}_{i=1}^{\ell_\mathcal{T}}$ a state of the Markov chain $\{\theta(k) \in \Theta\}_{k\in\mathbb{N}}$, i.e. $\Theta \doteq \{i \in \mathbb{N} : \tau_i \in \mathcal{T}\}$. Note that $\forall\,(\Gamma(k), x(k)) \in \mathbb{R}^{n_x+1}, \exists!\ \tau_i \in \mathcal{T} : (\Gamma(k), x(k)) \in R_{\tau_i}$, i.e. the current measurement $(\Gamma(k), x(k))$ is deterministically associated to one, and only one, discrete state of $\theta(k)$.

As each leaf τ_i contains a random subset of the SINR sequence, we assume independent and identically distributed samples belonging to the same partition element. We fit a GRV $G_{\tau_i} \sim \mathcal{N}(\mu_i, \sigma_i^2)$ to model the current level of SINR in

each leaf $\tau_i \in \mathcal{T}$ of the tree:

$$\begin{aligned}\mu_i &= \frac{1}{|\tau_i|} \sum_{(\Gamma(k),x(k))\in R_{\tau_i}} \Gamma(k),\\ \sigma_i^2 &= \frac{1}{|\tau_i|-1} \sum_{(\Gamma(k),x(k))\in R_{\tau_i}} (\Gamma(k)-\mu_i)^2.\end{aligned} \tag{3.14}$$

In Sec. 3.5 we will exploit $\{G_{\tau_i}\}_{\tau_i\in\mathcal{T}}$ to estimate the PEP in each state of $\{\theta(k)\}_{k\in\mathbb{N}}$.

Following the same reasoning, we fit a GRV $G^+_{\tau_i} \sim \mathcal{N}(\mu_{i^+}, \sigma^2_{i^+})$ to model the next-step level of SINR for each leaf $\tau_i \in \mathcal{T}$ of the tree:

$$\begin{aligned}\mu_{i^+} &= \frac{1}{|\tau_i|} \sum_{(\Gamma(k),x(k))\in R_{\tau_i}} \Gamma(k+1),\\ \sigma^2_{i^+} &= \frac{1}{|\tau_i|-1} \sum_{(\Gamma(k),x(k))\in R_{\tau_i}} (\Gamma(k+1)-\mu_{i^+})^2.\end{aligned} \tag{3.15}$$

In Sec. 3.5 we will exploit $\left\{G^+_{\tau_i}\right\}_{\tau_i\in\mathcal{T}}$ to identify the TPM of $\{\theta(k)\}_{k\in\mathbb{N}}$.

We remark that the TPM of $\{\theta(k)\}_{k\in\mathbb{N}}$ could be identified using only $\mathcal{T}$, based on the number of samples that at time k stay in a leaf $\tau_i \in \mathcal{T}$, and at time $k+1$ jump to a leaf $\tau_j \in \mathcal{T}$, i.e. $p(j \mid i) = \tilde{n}(\tau_i, \tau_j) \cdot |\tau_i|^{-1}$ where

$$\begin{aligned}\tilde{n}(i,j) \doteq| \{(\Gamma(k_\epsilon), x(k_\epsilon)) \in \mathcal{P}_{\mathcal{T}} :\\ (\Gamma(k_\epsilon), x(k_\epsilon)) \in R_{\tau_i}, (\Gamma(k_\epsilon+1), x(k_\epsilon+1)) \in R_{\tau_j}\Big\}|.\end{aligned} \tag{3.16}$$

The lack of this approach is that the tree $\mathcal{T}$ partitions the dataset to minimize the prediction error of a deterministic estimate of $\Gamma(k+1)$, instead of minimizing the prediction error with respect to the estimate $\mathbb{E}\left[\mathcal{T}(\Gamma(k), x(k)) \mid \Gamma(k)\right]$. We overcome this issue in the next section by growing an additional regression tree Π, and combining $\mathcal{T}$ and Π to identify the TPM to minimize the error with respect to $\mathbb{E}\left[\mathcal{T}(\Gamma(k), x(k)) \mid \Gamma(k)\right]$.

Step 2

The idea is to define a partition $\{R_{\pi_r}\}_r^{|\Pi|}$ of $\mathbb{R}$ minimizing the prediction error with respect to the estimate $\mathbb{E}\left[\mathcal{T}(\Gamma(k), x(k)) \mid \Gamma(k)\right]$, and exploit the Markov property to split the identification process of the TPM of $\{\theta(k)\}_{k\in\mathbb{N}}$ in two steps:

$$\begin{aligned}&\mathbb{P}(\theta(k+1)=j \mid \theta(k)=i)\\ &\doteq \mathbb{P}((\Gamma(k+1), x(k+1)) \in R_{\tau_j} \mid (\Gamma(k), x(k)) \in R_{\tau_i})\\ &= \sum_{\pi_r\in\Pi} \{\mathbb{P}((\Gamma(k+1), x(k+1)) \in R_{\tau_j} \mid \Gamma(k+1) \in R_{\pi_r})\\ &\mathbb{P}(\Gamma(k+1) \in R_{\pi_r} \mid (\Gamma(k), x(k)) \in R_{\tau_i})\}.\end{aligned} \tag{3.17}$$

To estimate $\mathbb{P}((\Gamma(k+1), x(k+1)) \in R_{\tau_j} \mid \Gamma(k+1) \in R_{\pi_r})$ we define a new predictor dataset $\{\Gamma(k)\}_{k=1}^D = \mathcal{P}_\Pi \subset \mathbb{R}$ and a new response dataset $\{\mathcal{T}(\Gamma(k), x(k))\}_{k=1}^D =$

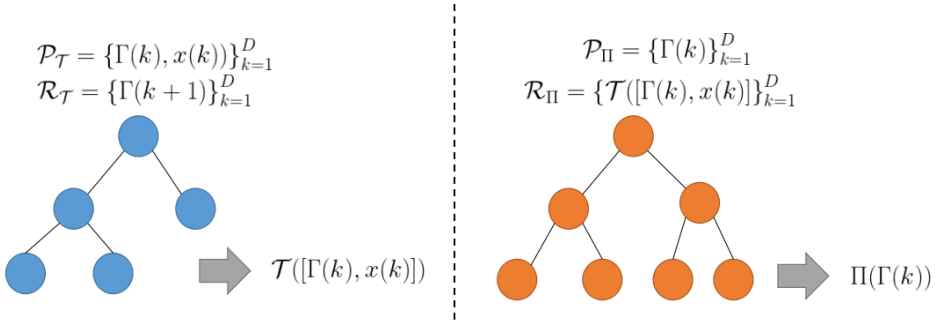

Figure 3.2: TPM identification via regression trees

$\mathcal{R}_{\Pi} \subset \mathbb{R}$. Then, we derive a new regression tree Π applying the CART algorithm to identify the function $\Pi : \mathcal{P}_{\Pi} \to \mathcal{R}_{\Pi}$. Let $\Pi(\Gamma(k))$ be the optimal estimation given by the tree Π. Define

$$\mathbb{P}((\Gamma(k+1), x(k+1)) \in R_{\tau_j} \mid \Gamma(k+1) \in R_{\pi_r}) \doteq p(\tau_j \mid \pi_r) = \frac{n(\pi_r, \tau_j)}{|\pi_r|} \tag{3.18}$$

where $n(\pi_r, \tau_j)$ is the number of samples that belong both the leaf $\pi_r \in \Pi$ and the leaf $\tau_j \in \mathcal{T}$, i.e

$$n(\pi_r, \tau_j) \doteq |\{(\Gamma(k_\epsilon), x(k_\epsilon)) \in \mathcal{P}_{\mathcal{T}} : \Gamma(k_\epsilon) \in R_{\pi_r}, (\Gamma(k_\epsilon), x(k_\epsilon)) \in R_{\tau_j}\}|. \tag{3.19}$$

Notice that in contrast to the definition in equation (3.16), equation (3.19) does not involve time evolution. For this reason, eq. (3.19) is useful to estimate the probabilities $\mathbb{P}((\Gamma(k+1), x(k+1)) \in R_{\tau_j} \mid \Gamma(k+1) \in R_{\pi_r})$.

Proposition 1. *Let us define the transition probabilities $p(\tau_j \mid \pi_r)$ as in Equation (3.18), then the CART algorithm creates the partition induced by $\{R_{\pi_r}\}_{\pi_r \in \Pi}$ optimally estimating the conditional expectation of the prediction in each leaf $\pi_r \in \Pi$, i.e. $c_{\pi_r} = \mathbb{E}\left[\mathcal{T}(\Gamma(k), x(k)) \mid \Gamma(k) \in R_{\pi_r}\right], \forall \pi_r \in \Pi$.*

Proof.

$$c_{\pi_r} = \sum_{\Gamma(k_\epsilon)\in R_{\pi_r}} \frac{\mathcal{T}(\Gamma(k_\epsilon), x(k_\epsilon))}{|\pi_r|} \tag{3.20}$$

$$= \sum_{\Gamma(k_\epsilon)\in R_{\pi_r}} \frac{1}{|\pi_r|} \sum_{\substack{(\Gamma(k'_\epsilon),x(k'_\epsilon))\in R_{\tau_j} \\ (\Gamma(k_\epsilon),x(k_\epsilon))\in R_{\tau_j}}} \frac{\Gamma(k'_\epsilon+1)}{|\tau_j|} \tag{3.21}$$

$$= \sum_{\substack{\Gamma(k_\epsilon)\in R_{\pi_r} \\ (\Gamma(k_\epsilon),x(k_\epsilon))\in R_{\tau_j}}} \frac{1}{|\pi_r|} \sum_{(\Gamma(k'_\epsilon),x(k'_\epsilon))\in R_{\tau_j}} \frac{\Gamma(k'_\epsilon+1)}{|\tau_j|} \tag{3.22}$$

$$= \sum_{\substack{\Gamma(k_\epsilon)\in R_{\pi_r} \\ (\Gamma(k_\epsilon),x(k_\epsilon))\in R_{\tau_j}}} \frac{c_{\tau_j}}{|\pi_r|} = \sum_{\tau_j\in\mathcal{T}} \frac{c_{\tau_j}\cdot n(\pi_r,\tau_j)}{|\pi_r|} = \sum_{\tau_j\in\mathcal{T}} p(\tau_j \mid \pi_r) c_{\tau_j} \tag{3.23}$$

$$\simeq \sum_{\tau_j\in\mathcal{T}} p(\tau_j \mid \pi_r)\mathbb{E}\left[\mathcal{T}(\Gamma(k),x(k)) \mid (\Gamma(k),x(k))\in R_{\tau_j}\right] \tag{3.24}$$

$$= \mathbb{E}\left[\mathcal{T}(\Gamma(k),x(k)) \mid \Gamma(k)\in R_{\pi_r}\right] \tag{3.25}$$

□

In (3.24) we assume that the dataset consists of independently drawn observations and that the number of samples in each region of the tree is large enough to neglect the Standard Error of the Sample Mean (SEM): as a consequence, the expectation can be assumed approximately equal to the sample mean. In conclusion, running the CART algorithm on our extended dataset derives transition probabilities that minimize the square of the error between the samples of the dataset $\{\mathcal{P}_\mathcal{T}, \mathcal{R}_\mathcal{T}\}$ and the corresponding conditional expectation of the predictive model of $\mathcal{T}$.

To estimate $\mathbb{P}(\Gamma(k+1)\in R_{\pi_r} \mid (\Gamma(k),x(k))\in R_{\tau_i}) = p(\pi_r \mid \tau_i)$ we exploit the set of GRVs defined in Equation (3.15) and the partition induced by the tree Π:

$$p(\pi_r \mid \tau_i) = \mathbb{P}\left(G^+_{\tau_i}\in R_{\pi_r}\right) = \int_{R_{\pi_r}} \phi_\mathcal{N}(\zeta;\mu_{i^+},\sigma^2_{i^+})d\zeta \tag{3.26}$$

where $\phi_\mathcal{N}(\zeta;\mu,\sigma^2)$ is the PDF of the GRV $\mathcal{N}(\mu,\sigma^2)$. The TPM $P = [p(j \mid i)]_{i,j} \in \mathbb{R}^{|\mathcal{T}|\times|\mathcal{T}|}$ is defined by $p(j \mid i) = \mathbb{P}(\theta(k+1) = j \mid \theta(k) = i)\ \forall\tau_i,\tau_j \in \mathcal{T}$, as in Equation (3.17).

Step 3

The variable of interest in control applications is the PDP. The IEEE-802.15.4 provides the estimation of the PER given the SINR, see Equation (3.6). Let $\boldsymbol{\nu} = (\nu_1,\ldots,\nu_{|\mathcal{T}|})$ associate a PDP for each channel operating mode as follows:

$$1-\nu_i = E[R_p(10^{G_{\tau_i}/10}) \mid \theta(k) = i] = \int_\mathbb{R} R_p(10^{\zeta/10})\phi_\mathcal{N}(\zeta;\mu_{c_i},\sigma^2_{c_i})d\zeta. \tag{3.27}$$

Algorithms 3 and 4 summarize respectively the learning methodology and the procedure to estimate the PDP over a time horizon of length N.

Algorithm 3 Learning algorithm (off-line)

1: INPUT: $\{\Gamma(k), x(k)\}_{k=1}^{D}$, OUTPUT: $\mathcal{T}$, P, $\boldsymbol{\nu}$
2: **function** LEARN PEP
3: DEFINE $\mathcal{P}_{\mathcal{T}} = \{\Gamma(k), x(k)\}_{k=1}^{D}, \mathcal{R}_{\mathcal{T}} = \{\Gamma(k+1)\}_{k=1}^{D}$;
4: BUILD $\mathcal{T}$ USING $\mathcal{P}_{\mathcal{T}}$ TO PREDICT $\mathcal{R}_{\mathcal{T}}$;
5: DEFINE $\mathcal{P}_{\Pi} = \{\Gamma(k)\}_{k=1}^{D}, \mathcal{R}_{\Pi} = \{\Pi((\Gamma(k), x(k)))\}_{k=1}^{D}$;
6: BUILD Π USING $\mathcal{P}_{\Pi}$ TO PREDICT $\mathcal{R}_{\Pi}$;
7: BUILD P BASED ON (3.17);
8: COMPUTE $\boldsymbol{\nu}$ USING (3.27);
9: **return** $\mathcal{T}$, P, $\boldsymbol{\nu}$
10: **end function**

Algorithm 4 PDP estimation (run-time)

1: INPUT: $\mathcal{T}$, P, $\boldsymbol{\nu}$, $(\Gamma(k), x(k))$, N, OUTPUT: $\hat{\boldsymbol{\nu}}$
2: **function** PREDICT PDP
3: Let $i \in \mathbb{N}$ be the integer such that $(\Gamma(k), x(k)) \in \tau_i$;
4: **for all** $j = 1, \ldots, N$ **do**
5: $\hat{\boldsymbol{\nu}}(j) = \boldsymbol{\nu} P^j(i, :)$, where $P^j(i, :)$ is the i-th row of P^j.
6: **end for**
7: **return** $\hat{\boldsymbol{\nu}}$
8: **end function**

3.6 Case study

We consider a WNCS consisting of an inverted pendulum on a cart remotely controlled over a WirelessHART link as illustrated in Sec. 3.2. In the numerical simulations, we model the plant using the nonlinear discrete-time model of the inverted pendulum and we model the packet delivery process as in equation (3.6) through the SINR trajectories obtained from the AR model in Sec. 3.2. We compare two implementations of Stochastic-Model Predictive Control (S-MPC) [67]: one based on the physics-based channel model of Sec. 3.2 as in [60], and one based on the data-driven channel model of Sec. 3.5.

Plant model

We consider the following nonlinear discrete-time model $y(k+1) = y(k) + f(x(k), u_a(k))T_u$, where $y \in \mathbb{R}^4$, $u \in \mathbb{R}$ and

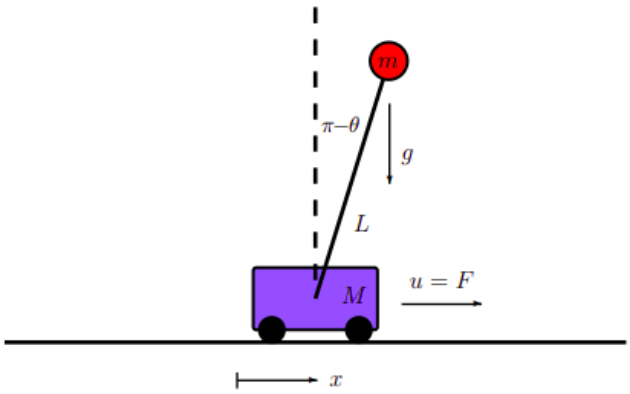

Figure 3.3: Inverted pendulum on cart.

$$
\begin{aligned}
f_1(y,u) &= y_2 \\
f_2(y,u) &= \frac{mL^2}{D}\left[-mg\cos(y_3)\sin(y_3) + mLy_4^2\sin(y_3)\right] \\
&\quad + \frac{mL^2}{D}(u - \delta y_2) \\
f_3(y,u) &= y_4 \\
f_4(y,u) &= \frac{mL}{D}\left[(m+M)g\sin(y_3) - \cos(y_3)mLy_4^2\sin(y_3)\right] \\
&\quad - \frac{mL\cos(y_3)}{D}(u - \delta y_2) \\
D &= mL^2\left[M + m\left(1 - \cos(y_3)^2\right)\right],
\end{aligned}
\tag{3.28}
$$

where y_1 is the cart position, y_2 is the velocity, y_3 is the pendulum angle, y_4 is the angular velocity, m is the pendulum mass, M is the cart mass, L is the length of the pendulum arm, g is the gravitational acceleration, δ is a friction damping on the cart, u is a control force applied to the cart and T_u is the sampling time.

We analyze a case wherein the model's dynamics influence channel behavior. More in detail, we consider that a time-varying distance between the plant and the controller, d_0 in equation (3.4), depends on the cart's position, y_1, i.e. $d_0(k) = y_1(k) + \bar{d}_0$, $\bar{d}_0 \in \mathbb{R}^+$. The other channel parameters are assumed constants. As a consequence, the intermittent control packets can be modeled as follows:

$$u_a(k) = \nu(k)u(k), \quad \nu(k) \sim B(1, 1 - R_b(\gamma(k))) \tag{3.29}$$

where $u(k)$ is the input computed by the controller, $B(n,p)$ is the Bernoullian distribution with a time-varying parameter. Notice that $y_1(k)$ influences $\gamma(k)$ via equation (3.4).

Numerical results

The data-driven model derived in Sec. 3.4 can be used to formalize the following:

Parameter	Value	Unit
Cart mass M	0.5	kg
Pendulum mass m	0.2	kg
Distance from the pivot to the mass center L	0.3	m
Friction coefficient of the cart d	0.1	N·s/m
Sampling time T_u	0.001	s

Table 3.1: Parameter values inverted pendulum

Problem 1. *Stochastic Model Predictive Control*

$$\begin{aligned}
\min_{u_k} \quad & \mathbb{E}\left[e_{k+N}^\top Q e_{k+N} + \sum_{j=0}^{N-1} e_{k+j+1}^\top Q e_{k+j+1} + u_{k+j}^\top R u_{k+j} \right] \\
\text{s.t.} \quad & x_{k+j+1} = A'_{i_j} x_k + \sum_{\alpha=0}^{j} B'_{i_j,\alpha} \hat{\nu}_{k+\alpha} u_{k+\alpha} + F'_{i_j}, \\
& u_{k+j} \in \mathcal{U}, \mathbb{E}\left[x_{k+j+1}\right] \in \mathcal{O}, \\
& \mathbb{E}\left[x_{k+N}\right] \in \mathcal{O}_N, x_k = x(k), j = 0 \ldots, N-1,
\end{aligned}$$

where $e_k = x_k - x_k^*$ is the difference between the current state of the plant and the target, and $\mathcal{O}$,$\mathcal{U}$ are polyhedra that specify the variables constraints. At each time step the optimal inputs $u_k^*, \ldots, u_{k+N-1}^*$ are computed using QP, and only the first one is applied to the system, i.e. $u(k) = u_k^*$.

The plant dynamics used in the MPC solution are identified using the methodology summarized in Sec. 3.4 and using a dataset consisting of 100 simulations, each one with time duration of 6 seconds, of the input and output of the plant with no packet losses. At any time k we can use such model and the measurement of $x_k = x(k)$ to determine the switching sequence $i_0, \ldots, i_{N-1}$ and hence the matrices $A_{i_{j-1},i_j}, B_{i_{j-1},i_j}, F_{i_{j-1},i_j}$. The solution of Problem 4 also requires knowledge of the initial state of $\hat{\nu}_k$ and of its TPM P [67]: we derive two models of $\hat{\nu}_k$, respectively using the methodology in [60] and in of Sec. 3.5: for both the data-driven and physics-based channel Markov models, we set the number of channel operating modes equal to 9.

The cost function in Problem 4 models a tradeoff between penalizing deviations from the desired trajectory x_k^* and minimizing the control effort. We define as control performance metric the cumulative cost of Problem 4 to compare the performance using the physics-based channel model of Sec. 3.2 as in [60] and the data-driven channel model of Sec. 3.5.

Tables 3.1 and 3.2 show the plant and channel parameters, respectively. The minimum update period $T_u = 0.1s$ of the WirelessHART standard is too slow for several control applications, and makes the wireless link uncorrelated at the packet level. Thus, in view of showing the impact of our algorithms as a methodological enabler for the development of mobile network technologies that support much

Parameter	Value	Unit
symbol rate $1/T_s$	62.5	ksymb/s
channel bandwidth W	2	MHz
users speed v_0, v_1	5.37	m/s
shadowing decay distance dc_0, dc_1	9	m
shadowing standard dev. σ_{β_0}, σ_{β_1}	2	dB
power control error standard dev. σ_{e_0}, σ_{e_1}	1.5	dB
power control error decorr. time τ_{ξ_0} and τ_{ξ_1}	1.5	dBm
reference user tx. power P_0	0	dBm
interferer tx. power P_1	10	dBm
distance reference tx-rx pair d_1	10	m

Table 3.2: Parameter values channel

higher update rates, we consider $T_u = 0.001$s. Moreover, to emphasize the impact and improvements of stochastic vs deterministic MPC, we set $\bar{d}_0 = 14$ to consider a scenario based on significant packet loss rates, where it is evident that deterministic MPC cannot guarantee acceptable performance while stochastic MPC does.

To statistically validate the control performance we ran Monte Carlo simulations generating 500 admissible trajectories, each with 6000 samples (corresponding to 6s). For all the simulations, the initial state is $x(0) = [2, 0, \pi, 0]^\top$ and the target state is $x^* = [5, 0, \pi, 0]^\top$. Figures 3.4, 3.5 and 3.6 illustrate simulation results. For each figure, the averaged performance is displayed in a solid line, and the 95.4% confidence interval is represented with a shaded area. The performance of the controllers based on physics-based and data-driven models is very close, with the advantage of the data-driven approach that no a-priori knowledge of channel model and parameters is required.

3.7 Conclusions

This chapter provides a novel technique to learn Markov models representing fading wireless channels. We consider a validation scenario consisting of a WNCS that exploits a WirelessHART radio link to send the optimal control inputs generated by a Stochastic MPC, and show that the control performances of our data-driven approach and a physics-based approach based on a stationary finite-state Markov chain are extremely close: this implies that in practical applications when assuming perfect knowledge of the channel model and parameters is not possible, the methodology presented in this chapter is a valid and very effective alternative. In future work, we plan to validate our techniques in an experimental setup and consider more general communication scenarios.

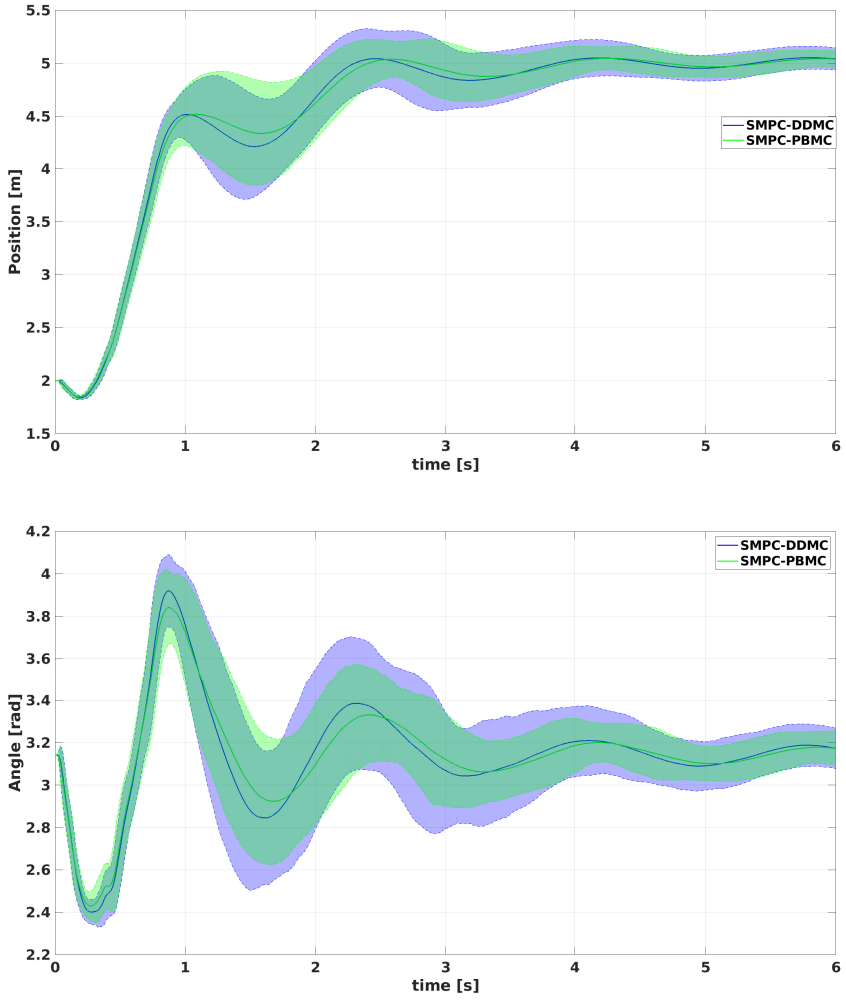

Figure 3.4: Controlled states in the closed-loop simulation.

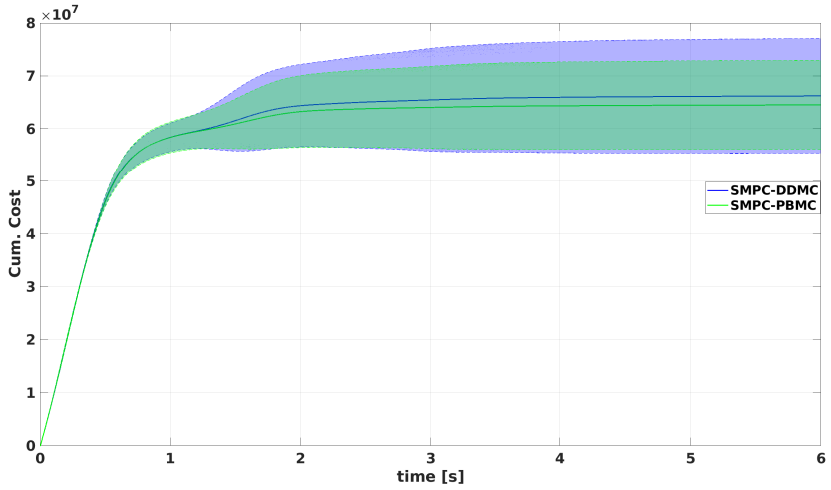

Figure 3.5: Cumulative cost of the closed-loop simulation.

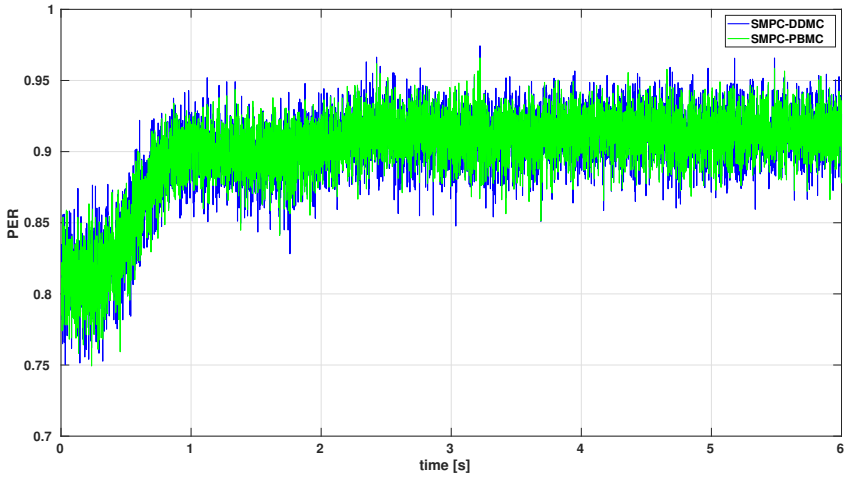

Figure 3.6: PER of the closed-loop simulation.

4

Reduced SARX modeling and control via Regression Trees

Contents

In this chapter, a complexity reduction methodology is proposed for a data-driven Switching Auto-Regressive eXogenous (SARX) model identification algorithm based on Regression Trees. In particular, we aim at reducing the number of submodels of a SARX dynamical model without compromising (and indeed improving) the model accuracy and mitigating the overfitting problem. A validation procedure is addressed to compare the performance of the reduced model with respect to the original one. Results show an important reduction in the number of modes of the identified model that ranges between 96% and 99.74%. The accuracy of the reduced model is also tested in terms of closed-loop control performance in a Model Predictive Control (MPC) setup, on a benchmark consisting of a non-linear inverted pendulum on a cart: the comparison is provided with respect to an *oracle*, i.e. an MPC setup with perfect knowledge of the plant dynamics.

The discussion in this chapter has already been presented in [39] and is currently patent pending.

4.1 Introduction

SARX systems are defined as collections of Auto-regressive exogenous systems indexed by a discrete-valued additional variable, called discrete state. Piece-Wise Auto-Regressive eXogenous (PWARX) systems are a particular class of SARX systems obtained by partitioning the state-input domain into a finite number of polyhedral regions and by considering linear/affine dynamics in each region [68], [69]. SARX and PWARX systems have been successfully used in the past years to model different kinds of phenomena, for example, a vast amount of literature focuses on the approximation of nonlinear systems with SARX and PWARX models, see e.g. [70], [71].

In this context, this work focuses on the data-driven approximation via SARX modeling of unknown nonlinear system dynamics given a collection of input-output pairs generated by such a system.

Literature review. The SARX and PWARX identification processes, in general, require the estimation of the submodels number, the ARX parameters, the model orders, and the discrete state estimation.

The difficulty of the problem depends on which quantities are assumed to be known or fixed a priori: in the simplest case, the identification boils down to a standard least-squares estimation when the discrete states for SARX, or the partition for PWARX, are either known or fixed a priori. On the other hand, the SARX and PWARX identification problems are NP-hard when the submodel number is known a priori, and thus the exponential complexity in the dimension is a natural expectation for any exact algorithm, see e.g. [72] and [73]. In this respect, the identification problem can be rewritten as a mixed-integer optimization, which is computationally tractable only for small instances.

In [74] the authors proposed a two-stage approach for Piece-Wise Affine (PWA) regression based on the combined use of recursive multimodel least-squares techniques and linear multicategory discrimination, while in [75] they introduced a methodology to fit jump models, a general class of models that encompasses PWA. The key idea is to alternate between minimizing a loss function to fit multiple model parameters and minimizing a discrete loss function to determine which set of model parameters is active at each data point.

The identification problem is ill-posed when the submodel number is unknown as the solution is only defined up to a trade-off between the submodel number and the model accuracy. A trivial solution is to assign a submodel to each data point, fitting the data perfectly. Obviously, this overfits the data and compromises the model generalization ability. A non-trivial solution is to fix the submodel number and then adjust it to improve the fitting. In [76], a recursive procedure is proposed to identify switched linear and PWA models from input-output data. Starting from an initial guess of the parameter vectors, representing the different submodels, the proposed algorithm alternates between data assignment to submodels and parameter update. At each iteration, the discrete state is determined as the index of the submodel that appears to have most likely generated the regressor vector observed at that instant. The approach requires the knowledge of an upper bound for the number of submodels and then discards the unassigned ones.

In [77], the authors proposed a three-stage procedure for the parametric identification of PWARX models. The first stage simultaneously classifies the data points, estimates the number of submodels and the corresponding parameters by solving a relaxed version of the identification problem. In the second stage, a refinement procedure reduces misclassifications and improves parameter estimates and the number of submodels. Finally, the third stage determines a polyhedral partition of the regressor set via two-class or multiclass linear separation techniques.

Motivation. In [61], the authors proposed a methodology to exploit a supervised learning technique called Regression Trees (RTs, [78]) to build a SARX dynamical model of a large-scale system using historical data so that MPC can be directly applied using a such dynamical model. More precisely, the RTs algorithm partitions the dataset into hyper-rectangular regions, minimizing the error performed by a constant predictor in each region. The authors in [61] exploited the obtained partitioning, and assigned an ARX model to each region (instead of the constant predictor), thus obtaining a SARX dynamical model. The learning algorithm has shown impressive experimental performance, especially when applied to real-life systems whose model is unknown and quite complex to be derived using physics-based approaches (see e.g. [79, 80, 81, 82, 83]).

However, since the partitioning is optimized for identifying constant predictors, and not ARX models, the approach tends to overestimate the number of submodels and, consequently, requires more computational and memory resources to implement control laws. Furthermore, it cannot even recognize data that are generated by linear plants or by systems wherein a single or few ARX models are enough to represent the system dynamics. As a consequence overfitting issues often arise, as we show in Section 4.4.

This chapter is based on the intuition that the number of regions needed by the SARX identification in [61] is significantly lower than the one obtained by constant predictors. For this reason, it is necessary to investigate methodologies able to reduce the complexity of the identified models. In this respect, some work has been done in the literature.

In [84] the authors focus on the problem of finding a minimal representation of polyhedral piecewise systems. More specifically, for a given polyhedral piecewise system, the authors solve the problem of deriving a model equivalent to the former and minimal in the number of polyhedra. This is done by merging the polyhedra that exhibit exactly the same model. However, this approach is not suitable for the case of the SARX model derived in [61] exploiting the RTs algorithm, where the models in the partitions are in general all different, although some can be *similar* in some sense. Moreover, the partitions obtained via RTs are hyper-rectangles and are already in minimal form.

In [85] the authors present an algorithm to a posteriori reduce the storage demand and the complexity of the closed-form controller for PWARX systems. Thus, the effort is focused on the control-law complexity reduction neglecting the PWARX model refinement.

In [77], the authors consider first an identification procedure to generate a SARX modeling framework starting from a dataset of collected measurements, and

then a reduction procedure on the identified model. The reduction procedure first uses a metric based on Euclidean norm to find discrete states that are *similar*, i.e. where the model parameters are close in the sense of a Euclidean metric and then clusterizes one by one the samples associated to each mode. Finally, the authors link the samples in each cluster to the new (reduced) discrete modes solving a classification problem. This procedure, as the authors in [74] remark, works well with small datasets, as the samples are handled separately. The approach we propose, instead, is based on the binary tree structure (generated by the Regression Trees) in [61] to formalize the polyhedra characterizing the piecewise system: this binary tree structure both removes the need to handle samples separately and of solving a further classification problem.

Contribution. We propose in this work a methodology to refine the model obtained via [61] with the goal of reducing the number of submodels and recognizing linear or almost-linear patterns, without modifying the RT, and hence the partitioning, structure. To this aim, we define a procedure that maintains the SARX model structure based on RTs, but drastically reduces the number of discrete modes of the SARX model, while improving modeling accuracy and computational cost in run-time, and mitigating overfitting problems.

More precisely, we use a standard metric based on the Euclidean norm to select the submodels that are *similar* to each other and merge them together. A discard procedure is also applied to eliminate submodels associated to an insufficient portion of the dataset.

We validate the procedure on a benchmark consisting of an inverted pendulum model on a cart. We compare it in terms of model accuracy and discrete modes reduction with respect to the original model provided by [61]. Results show an important reduction in terms of mode number and an improvement in terms of model accuracy and mitigation of peaks due to overfitting problems.

Finally, we also employ the reduced model to build an MPC problem to control the nonlinear benchmark. We compare the closed-loop performance of the MPC with the reduced model against the *oracle*, i.e. the nonlinear MPC with perfect knowledge of the system dynamics. Results show that our model reaches satisfactory performance also in terms of control, with the advantage of drastically decreasing the computational cost without requiring any preliminary knowledge of the system other than the collected data.

Chapter organization. In Section 4.2 we address the problem we want to solve in this chapter. The main contribution of the chapter, i.e. the reduction algorithm, is presented in Section 4.3. Finally, the proposed algorithm is tested in terms of model validation and control performance in Section 4.4.

4.2 Problem formulation

The goal of the chapter is to provide a methodology to reduce the number of submodels of a SARX model identified via the recently published technique in [61] without compromising the model accuracy and mitigating the overfitting problem. To this aim, we first recall the technique introduced in [61] to identify a SARX

model starting from historical data and using Regression Trees, and then we state our goal, which consists of a complexity reduction methodology.

SARX modeling via RTs. Let a dataset $\mathcal{D} = \{(y(k), u(k))\}_{k=1}^{D}$ of D samples collected from the measurements of a physical system be given, where $y(k) \in \mathbb{R}^{n_y}$ and $u(k) \in \mathbb{R}^{n_u}$ correspond to the output and input measurements respectively.

The goal is to learn a SARX model as

$$y(k) = \theta_{\sigma(x(k))}^{\top} x(k), \tag{4.1}$$

where

$$\begin{aligned} x(k) \doteq [&y(k-1)^{\top}, \dots, y(k-\delta_y)^{\top}, \\ &u(k-1)^{\top}, \dots, u(k-\delta_u)^{\top}, 1]^{\top}, \end{aligned} \tag{4.2}$$

$\sigma : \mathbb{R}^{n_x} \to \{1, \dots, s\} \subset \mathbb{N}$ is a function that, on the basis of a partition $\{\mathcal{R}_i\}_{i=1}^{s}$ of $\mathbb{R}^{n_x}$, with $n_x = n_y\delta_y + n_u\delta_u + 1$, associates to each region $\mathcal{R}_i$ a discrete mode i and the corresponding parameters $\theta_i \in \mathbb{R}^{n_y \times n_x}$.

According to [68], the general SARX model identification problem reads as follows.

Problem 2. *Given a collection of D input-output pairs $(y(k), u(k))$, $k = 1, \dots, D$, estimate the model orders δ_y, δ_u, the number of modes s, and the parameters θ_i, $i = 1, \dots, s$, and the regions $\{\mathcal{R}_i\}_{i=1}^{s}$.*

In [61] the authors address Problem 2 and, assuming that the model orders δ_y, δ_u are design parameters, proposed a methodology to exploit machine learning techniques, more precisely Regression Trees, to identify a SARX dynamical model as in (4.1) of a large-scale system using historical data, and used it to apply a finite-horizon optimal control strategy.

Without any loss of generality, and only for the simplicity of discussion, we consider from now on the case with $n_y = 1$ and a one-step finite horizon. The extension to the case with $n_y > 1$ and a finite horizon of arbitrary length is trivial and is discussed in [61] and Section 4.4.

More precisely, the authors in [61] used the CART algorithm (see the Appendix for details) to partition the dataset $\mathcal{D}$ to generate a partition of $\mathbb{R}^{n_x}$ consisting of hyper-rectangles $\{\mathcal{R}_i\}_{i=1}^{s}$ to be associated with the discrete modes of (4.1).

The output of the CART algorithm is a binary tree structure $\mathcal{T}$ where a mode i and a subset $\mathcal{D}_i$ of the dataset $\mathcal{D}$ is associated with each leaf so that $\{\mathcal{D}_i\}_{i=1}^{s}$ (where s is the number of leaves) is a partition of D. In each leaf i, and on the basis of the subset $\mathcal{D}_i$ of samples, they identified the parameter matrices θ_i, $i = 1, \dots, s$, via the Least Squares algorithm. For more details about the identification procedure, we refer the reader to [61].

Main contribution: complexity reduction in SARX modeling via RTs. As mentioned in the introduction, a lack of the methodology above is that the CART algorithm partitions the feature space by $\{\mathcal{R}_i\}_{i=1}^{s}$ with the aim of minimizing the response variable variance in each region. This corresponds to minimizing the prediction error performed by a constant estimator in each leaf, and thus it is in

general not optimal anymore when an ARX modeling is used instead of constant estimators. To address this issue, starting from the nominal SARX model in (4.1), the aim of this chapter is to derive a procedure to learn a SARX model

$$y(k) = \bar{\theta}^{\top}_{\sigma(x(k))} x(k), \tag{4.3}$$

where $\sigma : \mathbb{R}^{n_x} \to \{1, \dots, \bar{s}\} \subset \mathbb{N}$, with $\bar{s} << s$ and without compromising the model accuracy.

4.3 SARX refinement procedure

In this section, we address the refinement procedure to reduce the number of submodels of the SARX modeling framework identified via Regression Trees as discussed in Section 4.2.

Indeed, the main issue of the approach in [61] is that the identification procedure tends to overestimate the number s of submodels needed to fit the data, thus increasing the complexity and generating overfitting problems. In this respect, the idea is to apply a merge and discard procedure to all the ARX submodels. The *similarity* of the parameters of all submodels (not necessarily corresponding to adjacent regions $\mathcal{R}_i$) is quantified according to a metric, as defined later on: the samples associated to *similar* submodels are merged together, while if a submodel does not represent a relevant portion of the dataset it is discarded. Finally, the parameter vectors of the reduced model are computed based on the new partitioning.

Let a model as in (4.1) with $n_y = 1$ be given, generated from a dataset $\mathcal{D} = \{(y(k), u(k))\}_{k=1}^{D}$ via the RTs procedure in [61]. Let $\{\mathcal{R}_i\}_{i=1}^{s}$ be the hyper-rectangular regions associated to the leaf of the binary tree $\mathcal{T}$ generated by the CART algorithm, each associated with a discrete mode of (4.1), and θ_i, $i = 1, \dots, s$, the parameter matrices identified in each leaf i via the Least Squares algorithm to the corresponding dataset $\mathcal{D}_i$.

We define in the following a refinement procedure consisting of three steps: (1) submodel merging, (2) submodel discarding, (3) parameter estimation.

Submodels merging. The first step of the refinement procedure consists of a submodel *similarity* analysis, where the goal is to merge submodels that are described by *similar* parameters. To this aim, we consider the following standard submodels *similarity* index based on the Euclidean norm

$$\mu\left(\theta_i, \theta_j\right) = \frac{||\theta_i - \theta_j||_2}{\min\left\{||\theta_i||_2, ||\theta_j||_2\right\}}, \tag{4.4}$$

where θ_i and θ_j are respectively the parameter vectors describing two submodels of the form of (4.1). We use Equation 4.4 to provide the definition of *similarity cluster* of submodels.

Definition 1 (similar cluster)**.** *Let a model as in* (4.1) *be given, and let* $\alpha \in \mathbb{R}$, $\alpha > 0$. *A* similar cluster *w.r.t.* α *for the region* $\mathcal{R}_i$ *is defined as*

$$\mathcal{C}_i \doteq \left\{ \cup_{j=1}^{s} \mathcal{D}_j | \mu(\theta_i, \theta_j) \leq \alpha \right\}. \tag{4.5}$$

Clearly, Definition 1 allows *similar clusters* to be overlapping. The goal of the submodels merging procedure is to create a cluster set $\mathcal{C}$ of non-overlapping *similar clusters* $\mathcal{C} = \{\mathcal{C}_i\}_{i=1}^{\hat{s}}$, with $\hat{s} \leq s$, such that $\uplus_{i=1}^{\hat{s}} \mathcal{C}_i = \mathcal{D}$, where $\uplus$ represents the disjoint union. There are several ways to achieve this goal. The strategy we propose is summarized in Algorithm 5.

Algorithm 5 Submodel clustering

```
INPUT: {θ_i}_{i=1}^s, α
OUTPUT: {C_ν}_{ν=1}^ŝ
function SUBMODEL CLUSTERING
    Initialize cluster index: ν = 1
    Initialize mode index set: M = {1, ..., s}
    for i ∈ M do
        Initialize cluster: C_ν = D_i
        for j ∈ M, j ≠ i do
            Compute μ(θ_i, θ_j)
            if μ(θ_i, θ_j) ≤ α then
                Update cluster: C_ν = C_ν ∪ D_j
                M = M \ {j}
            end if
        end for
        ν = ν + 1
    end for
end function
```

Submodels discarding. Numerical simulations show that, in some cases, the merging stage creates a gap in the amount of data samples associated with each cluster. For this reason, we introduce a discarded stage where the basic idea is to eliminate the clusters associated with an insufficient portion of the dataset.

Let $|\mathcal{C}_i|$ be the number of samples in the cluster $\mathcal{C}_i$.

Definition 2 (discarded cluster)**.** *Let a cluster set $\mathcal{C} = \{\mathcal{C}_i\}_{i=1}^{\hat{s}}$ be given, and let $\beta \in \mathbb{R}$, $\beta > 0$. A discarded cluster w.r.t. β is defined as the set*

$$\mathcal{C}_d = \{\cup_{i=1}^{\hat{s}} \mathcal{C}_i \in \mathcal{C} | \frac{|\mathcal{C}_i|}{|\mathcal{D}|} \leq \beta\}.$$

The submodels discard procedure consists in creating a discarded cluster $\mathcal{C}_d$, and a non-discarded cluster set $\mathcal{C}_{nd} = \mathcal{C} \setminus \mathcal{C}_d = \{\mathcal{C}_{nd,i}\}_{i=1}^{\bar{s}-1}$, with $\bar{s} \leq \hat{s}$. Note that $\bar{s} = \hat{s}$ if $\mathcal{C}_d = \{\emptyset\}$.

Then, the final cluster set obtained after the merge and discard procedure is defined as

$$\mathcal{C}_{fin} = \begin{cases} \{\mathcal{C}_{nd}, \mathcal{C}_d\} & \text{if } \mathcal{C}_d \neq \{\emptyset\} \\ \mathcal{C} & \text{if } \mathcal{C}_d = \{\emptyset\} \end{cases}.$$

Parameters estimation.

Once the final cluster is obtained, thanks to the merge and discard procedures, we associate to each cluster $\mathcal{C}_{fin,i}$, $\forall i = 1, \dots, \bar{s}$, with $\bar{s} < s$, a parameter vector $\bar{\theta}_i$ that is estimated via the Least Squares method using the samples in the cluster, and update the model of all leaves whose samples are contained in the cluster $\mathcal{C}_{fin,i}$.

Note that while the number of discrete modes is reduced, the tree structure is not modified, hence the number of leaves of the tree remains the same. This implies that the same submodel described by $\bar{\theta}_i$ can be associated with more than one leaf of the tree. This is an advantage with respect to classical tree pruning, as we can merge predictions associated with regions that are not adjacent. The whole refinement procedure is summarized in Algorithm 6.

Algorithm 6 Refinement Procedure

INPUT: $\{\theta_i\}_{i=1}^{s}$, α, β
OUTPUT: $\{\bar{\theta}_i\}_{i=1}^{\bar{s}}$
function REFINEMENT
    Compute $\mathcal{C} = \{\mathcal{C}_\nu\}_{\nu=1}^{\hat{s}}$ via Algorithm 5
    Initialize the discarded cluster: $\mathcal{C}_d = \{\emptyset\}$
    Initialize the non-discarded cluster set: $\mathcal{C}_{nd} = \mathcal{C}$
    Initialize new cluster set dimension: $\bar{s} = \hat{s}$
    for $\nu = 1 : \hat{s}$ **do**
        if $\frac{|\mathcal{C}_\nu|}{|\mathcal{D}|} \leq \beta$ **then**
            $\mathcal{C}_d = \mathcal{C}_d \cup \mathcal{C}_\nu$
            $\mathcal{C}_{nd} = \mathcal{C}_{nd} \setminus \mathcal{C}_\nu$
            $\bar{s} = \bar{s} - 1$
        end if
    end for
    $\bar{s} = \bar{s} + 1$, i.e. the number of remaining clusters + the discarded cluster
    $\mathcal{C}_{fin} = \{\mathcal{C}_{nd}, \mathcal{C}_d\} = \{\mathcal{C}_{fin,i}\}_{i=1}^{\bar{s}}$
    for $i = 1 : \bar{s}$ **do**
        Compute $\bar{\theta}_i$ using the samples in $\mathcal{C}_{fin,i}$
    end for
end function

4.4 Case study

In this section, we provide simulation results concerning the proposed methodology on a simulated inverted pendulum on a cart. In particular, in what follows we first introduce the plant model, then we describe the procedure we use to generate the data to test our algorithm on, and finally, we provide the results on the validation of the reduction procedure w.r.t. the original one and on the control performance of the reduced model.

We model the plant using the nonlinear discrete-time model of the inverted pendulum proposed in [86]:

$$y(k+1) = y(k) + f(y(k), u(k))T_u, \; y \in \mathbb{R}^4, \tag{4.6}$$

where y_1 is the cart position, y_2 is the velocity, y_3 is the pendulum angle, y_4 is the angular velocity, $m = 0.2$kg is the pendulum mass, $M = 0.5$kg is the cart mass, $L = 0.3$m is the length of the pendulum arm, $\delta = 0.1$Ns/m is a friction damping on the cart, u is a control force applied to the cart and $T_u = 0.05$s is the sampling time.

4.4.1 Dataset generation

To generate the data used in the modeling process we ran several closed-loop control experiments of 8 seconds each.

We ran 1000 simulations implementing, for the first 6 seconds of each run, the nonlinear MPC in Problem 3 exploiting the model in eq. 4.6.

Problem 3. *Nonlinear Model Predictive Control*

$$\begin{aligned} \min_{u_k,\dots,u_{k+N-1}} \quad & \sum_{j=0}^{N-1} e_{k+j+1}^\top Q e_{k+j+1} + u_{k+j}^\top R u_{k+j} \\ \text{s.t.} \quad & y_{k+j+1} = y_{k+j} + f(y_{k+j}, u_{k+j}) T_u \\ & y_k = y(k), \end{aligned}$$

where $e_k = y_k - y^*$ is the difference between the current state of the plant and the target. We did not use any constraint on the input and the state since the goal was to simply run closed-loop simulations to collect a wide variety of data. As done in the MPC, only the first input of the optimal control sequence is applied at each time step, i.e. $u(k) = u_k$. For the first simulation, we set the initial state to have the cart position in zero and the inverted pendulum in the down configuration, i.e. $y_0 = [0, 0, -\pi, 0]^\top$. The subsequent runs had as the initial state the final state of the previous simulation. For each experiment, we draw the target cart position from a uniform distribution $p^* \sim \mathcal{U}[1, 7]$, and the up configuration for the inverted pendulum as the target angle, i.e. $y^* = [p^*, 0, 0, 0]^\top$.

In the last 2 seconds of each run, we draw the input from a normal distribution, i.e. $u(k) \sim \mathcal{N}(0, 0.7)$. In this way the inverted pendulum loses the stability of the up configuration, thus we can solve the swing-up problem in the subsequent simulation at each run, other than generating input-output data with random inputs. Notice that this step is fundamental to appreciating the nonlinear behavior of the plant; otherwise, if we restricted our research only to cases wherein the inverted pendulum starts in up-configuration, a linear model could have been enough to describe the plant dynamics.

4.4.2 Identification and validation

In this stage, we consider the data generated from the first 900 simulations as the training dataset for the learning algorithm, and the data from the remaining 100 simulations as the testing dataset for the model validation.

As a first step, we learn a model based on the approach proposed in [61], i.e. the model recalled in (4.1), using the training dataset with a predictive horizon of 1 second, i.e. $N = 20$, and call it *SARX* from now on. Then, we exploit the

methodology exposed in Sec. 4.3 to learn, using the training dataset, a second model of the plant with the same predictive horizon, i.e. $N = 20$, and call it *R-SARX* (Reduced SARX).

Remark 1. *The procedure in Section 4.3 has been defined considering $n_y = 1$ and $N = 1$. To extend the procedure to the case of $n_y > 1$ and $N > 1$, as in this case, it is sufficient to create an RT per each output and an RT per horizon step, for a total of $n_y N$ RTs. Then, for each step of the horizon, we can put the parameter vectors, obtained from the n_y RTs, as the rows of the θ_i parameter matrices in* (4.1) *(see [61]). The reduction procedure of Section 4.3 can then be applied to every single tree.*

As a second step, we proceed with the validation and compare the performance of the 2 models in terms of prediction accuracy using the testing dataset and the number of switching modes obtained for the 2 models over the horizon. The accuracy is compared using the Normalized Root-Mean-Square Error (NRMSE) with the normalization operated with respect to the difference between the maximum and the minimum value of the sequence, i.e.

$$\text{NRMSE}[\%] = \frac{\sqrt{\frac{1}{D}\sum_{k=1}^{D}\left(y(k) - \hat{y}(k)\right)^2}}{\max\left(y(k)\right) - \min\left(y(k)\right)} \cdot 100, \tag{4.7}$$

where D is the number of samples in the testing dataset and $\hat{y}$ is the model estimate. We consider the NRMSE and the number of submodels for each output of the plant, i.e. $i = 1, \ldots, n_y$, and for each step of the predictive horizon, i.e. $j = 0, \ldots, N-1$.

Figure 4.1 shows the NRSME for each state of the pendulum and over the predictive horizon $j = 0, \ldots, N-1$.

We emphasize that in some isolate cases, the model predictions provided by the SARX modeling can provide spikes due to numerical errors and to the overfitting problem, thus resulting in peak errors in the NRMSE computation as we can see in Figure 4.1. They appear in general farther in the predictive horizon. However, this fact does not affect much the closed-loop performance since it happens in isolated cases, as we can see for example in Figure 4.2, where we show the trajectory of the prediction of the state y_3 at the horizon step $k + 18$. In particular, in Figure 4.2 top we can see how large these spikes can be. A zoom of the trajectories can be seen in Figure 4.2 bottom, where we can see how smaller spikes can be more frequent.

The number of discrete modes obtained from the learning of the SARX and R-SARX models for each state y_i, $i = 1, \ldots, 4$, over the predictive horizon, is reported in Table 4.1. We can appreciate how the proposed reduction methodology is able to drastically reduce the number of modes needed to describe the system dynamics without affecting the prediction capability of the SARX model proposed in [61]. Actually, the R-SARX learning improves the prediction accuracy and reduces the peaks generation problem of the SARX approach in [61], as we can see in Figure 4.1, since by increasing the number of samples in each cluster it reduces the effects of the overfitting problem.

The quantitative performance obtained with the proposed methodology performs a reduction within 96% and 99.74% for each case.

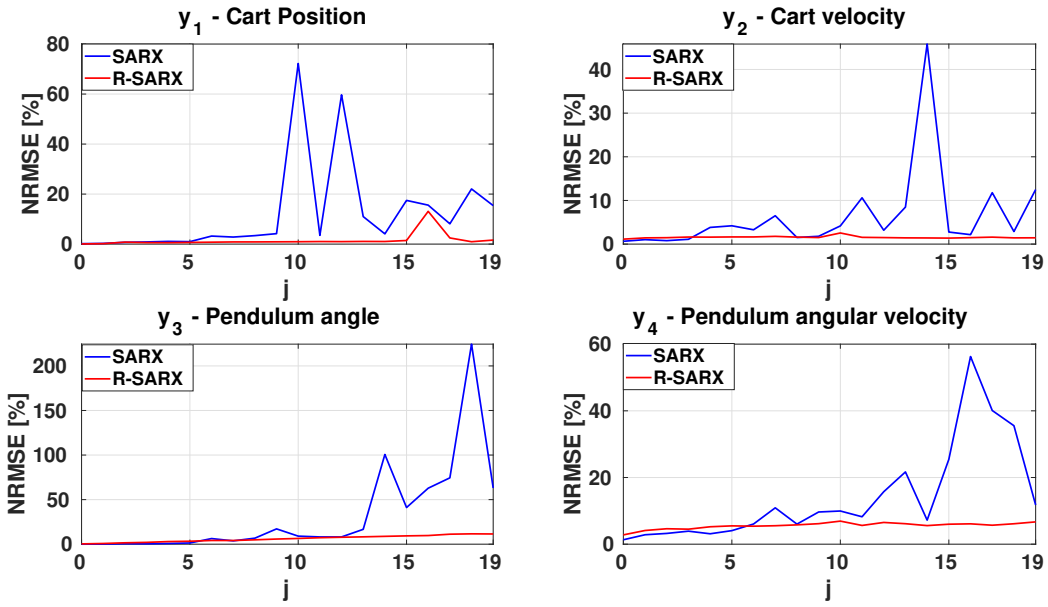

Figure 4.1: Comparison of the NRMSE [%] from the validation procedure over the predictive horizon for the nominal SARX model and the reduced one.

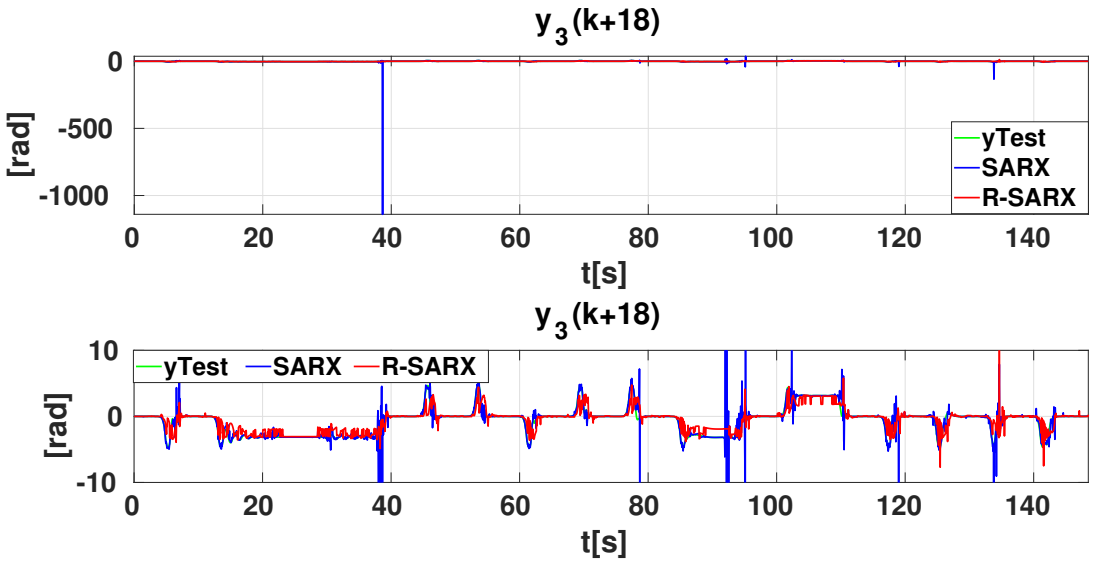

Figure 4.2: Trajectory of $y_3(k+18)$ compared with the ground truth. The top figure shows the overall trajectories, showing how large the spikes can be. The bottom figure shows a zoom of the trajectories, showing that small spikes are more frequent.

In particular, we emphasize that the performance reached for the linear dynamics, i.e. the prediction for y_1 and y_3 at $k+1$, provided a reduction of respectively 99.5% and 99.6%, obtaining for both cases only 2 modes, thus approaching the optimality consisting on a single mode needed to represent a linear behavior.

	SARX	**R-SARX**	**SARX**	**R-SARX**	**SARX**	**R-SARX**	**SARX**	**R-SARX**
k+	y_1	y_1	y_2	y_2	y_3	y_3	y_4	y_4
1	410	2	996	9	500	2	1036	25
2	651	2	1092	22	707	3	1129	29
3	858	3	1154	26	775	4	1169	35
4	1029	3	1154	27	849	5	1191	32
5	1098	3	1165	27	874	9	1228	23
6	1119	3	1179	34	896	13	1237	20
7	1161	3	1195	35	910	16	1239	25
8	1164	3	1204	37	917	20	1250	18
9	1160	4	1226	23	950	23	1281	26
10	1165	5	1220	30	999	20	1275	33
11	1177	6	1204	35	1001	24	1318	26
12	1186	8	1230	20	1025	22	1351	28
13	1188	7	1245	26	1035	25	1353	35
14	1159	8	1236	21	1063	26	1378	30
15	1174	8	1265	26	1094	28	1385	28
16	1176	7	1264	28	1126	23	1376	23
17	1172	12	1277	29	1147	30	1382	26
18	1170	13	1312	28	1181	23	1379	27
19	1155	15	1313	28	1177	29	1389	32
20	1168	19	1327	34	1189	27	1363	28

Table 4.1: Number of modes obtained from the SARX and R-SARX learning.

4.4.3 Control performance

In this subsection, we analyze the performance of the reduced modeling procedure in the context of the closed-loop control and compare it to an oracle, i.e. closed-loop control with perfect knowledge of the system dynamics addressed in Problem 3. The closed-loop MPC problem that we implement for the R-SARX model is the following

Problem 4. *(R-SARX Model Predictive Control)*

$$\begin{aligned} \underset{u_k,\dots,u_{k+N-1}}{\textit{minimize}} \quad & \sum_{j=0}^{N-1} \left(e_{k+j+1}^\top Q e_{k+j+1} + u_{k+j}^\top R u_{k+j} \right) \\ \textit{subject to} \quad & y(k+j+1) = \theta_{\sigma(x(k))}^\top x(k+j+1) \\ & y_k = y(k),\ j = 0,\dots,N-1, \end{aligned} \tag{4.8}$$

where $e_k = y_k - y^*$ is the difference between the current state of the plant and the target, and the θ_i are the parameter matrices obtained through the reduction procedure.

The goal is to compare the control performance using the reduced model with respect to the oracle. To this aim, we compute the closed-loop control by running numerical simulations of both Problem 4 with the R-SARX model and Problem 3 to control the nonlinear plant.

We run the closed-loop control for 10 seconds. We set the initial state $y(0) = [2, 0, -\pi, 0]^\top$. The targets consist in to perform a swing-up with the inverted pendulum and reach a target position with the cart, i.e. $y^* = [4, 0, 0, 0]^\top$. Matrices Q and R have been set as $Q = \text{diag}([200, 1, 980, 1])$ and $R = 2$.

At each time step, the optimal control sequence $u_k^*, \ldots, u_{k+N-1}^*$ is computed, and only the first input is applied to the system, i.e. $u(k) = u_k^*$.

Figure 4.3 and Figure 4.4 show the simulative results of the states and input trajectories respectively. We can appreciate how the dynamics obtained with the R-SARX are close to the oracle ones, keeping in mind that the oracle is an ideal non-realistic case. We can also see how the input trajectory gets close to the ideal one very quickly. It is also worthy of notice that Problem 4 is a QP, and can be efficiently solved, while Problem 3 is a Nonlinear Program that requires more time to be solved.

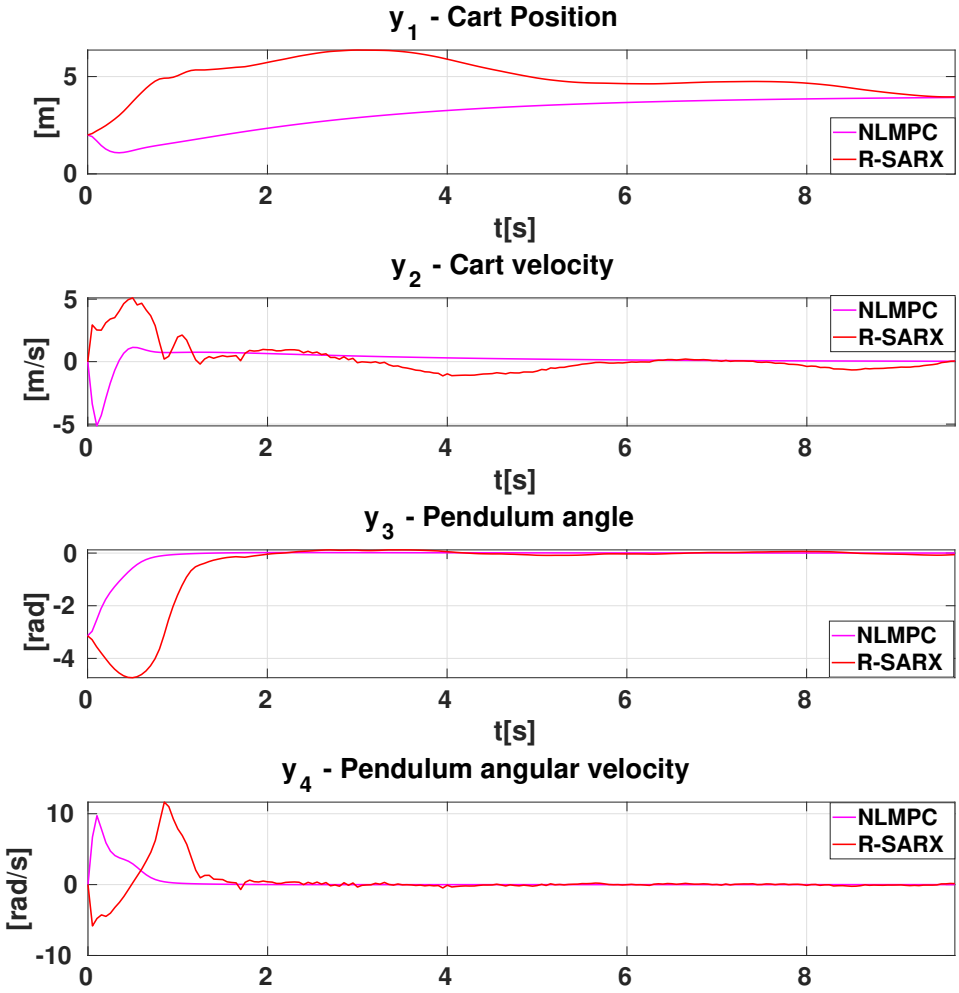

Figure 4.3: Controlled states in the closed-loop simulation.

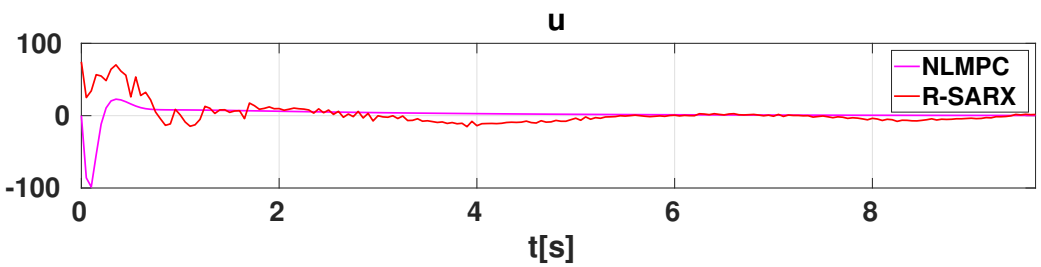

Figure 4.4: Optimal input applied in the closed-loop simulation.

4.5 Conclusions

In this chapter, we provided a technique to reduce the number of discrete modes in a SARX model identified via Regression Trees without compromising (indeed improving substantially, in our case of study) the prediction accuracy and mitigating the overfitting problems due to the identification procedure.

We validated the proposed methodology on a benchmark consisting on a nonlinear inverted pendulum over a cart, by comparing its prediction accuracy and the number of discrete modes with respect to the model identified using the approach in [61]. Results show an important reduction in the number of modes of the identified model that ranges between 96% and 99.74%. As our complexity reduction algorithm also alleviates overfitting issues, the overall prediction accuracy improves as well. The reduced model has been also tested in terms of closed-loop control performance: we compared two MPC algorithms, one using the reduced model and the other using an *oracle*, i.e. perfect knowledge of the plant dynamics. Results show that the control performance of the reduced model and the *oracle* are very close. Thus in practical applications, when assuming a perfect knowledge of the model is not possible, the methodology presented in this chapter is a valid and computationally efficient alternative.

In future work, we plan to further improve the reduction methodology and provide theoretical results about its convergence. We also aim at validating our technique on a real-life experimental setup.

5

Control-Aware Dynamic Edge Computing for Real-Time Target Tracking in UAV Systems

Contents

The autonomous operations of Unmanned Aerial Vehicles (UAV) necessitate the real-time analysis of information-rich signals, such as camera and LiDAR feeds, where the analysis algorithms often take the form of extremely complex Artificial Neural Network (ANN)s. The continuous execution of such models onboard the UAV imposes a considerable resource consumption (*e.g.*, energy), while offloading the execution of the models to edge servers requires the transmission of the input signals over capacity-constrained, time-varying, wireless channels. In this chapter, we propose an innovative approach – CADET – to control where sensor signals are processed in the system. In addition to traditional features and measures, such as channel state, energy consumption and channel usage, CADET makes dynamic task routing decisions – local computing vs edge computing – based on the state of the flight controller. The proposed methodology is based on Markov jump-switched linear systems, where an embedded filter predicts and controls the state of the joint motion/computing dynamics. To overcome technical challenges in terms of state observability, our control logic is based on an innovation measure that detects target

motion characteristics. computing decision problem as a Markov decision process involving the control system metrics and computational requirements. Then, we design a novel filter for Markov jump systems able to estimate the target state based on the current measurement; Finally, we control the computing pipeline by defining a quality indicator of the filtered state able to detect changes in the target motion.

The methodology has been accepted in [40].

5.1 Introduction

UAV are rapidly emerging as a key technology in a broad range of applications, including delivery, reconnaissance, surveillance, and emergency response. In order to operate autonomously, the UAVs need to analyze real-time streams of information-rich signals, such as images and LiDAR. The analysis algorithms typically take the form of large Artificial Neural Network (ANN) for object detection, classification, and decision-making. The need to continuously execute these complex models clashes with the severe constraints of these airborne platforms regarding computing power and energy availability. To make an example, EfficientDet 7 [87], a recent 2D object detector that achieves a Mean Average Precision (mAP) of 55.1% on the COCO dataset [88], has 52M of parameters: a complexity that makes its execution unfeasible even on relatively powerful embedded platforms such as the NVIDIA Jetson Nano due to memory constraints. Smaller models, such as EfficientDet 1 (3.9M parameters) and SSD MobileNet v2 (2.1M parameters) achieve an mAP of 33% and 20%, respectively - a considerable performance drop compared to the best-performing models. However, even the execution of such a more compact models model is challenging: in our experiments, we measured a maximum frame rate of 6frames per second using SSD MobileNet v2 on the NVIDIA Jetson Nano, while power consumption increased from 2.8W to 5W - a non-negligible amount considering that a medium size quadcopter consumes 24W when in motion. Recent techniques such as pruning, quantization, and knowledge distillation can lead to a further reduction in the computational load but also result in a non-negligible performance loss [89]. For instance, Yolo-Lite [90] achieves a frame rate of 22 frames per second on embedded devices but has a mAP of 12.36% on the COCO dataset.

An alternative solution is to connect the UAV to the infrastructure, where compute-capable devices at the network edge - the edge servers - take over the execution of heavyweight tasks [91]. Clearly, this strategy requires transferring the data to the edge server over the wireless channel connecting the UAV to the infrastructure. Intuitively, the time needed to transport the data depends on the capacity of the wireless channel, which is subject to well-known impairments and uncertainty [92]. On the other hand, the edge server can use more complex algorithms due to its stronger capabilities, while possibly reducing the bare execution time compared to the execution of smaller models onboard the UAV. Edge computing considerably reduces power consumption, as the UAV does not need to execute models on its onboard resources.

Based on the above discussion, both options have advantages and disadvantages:

Local Computing (LC): *Pros:* (i) the task completion delay is (almost) deter-

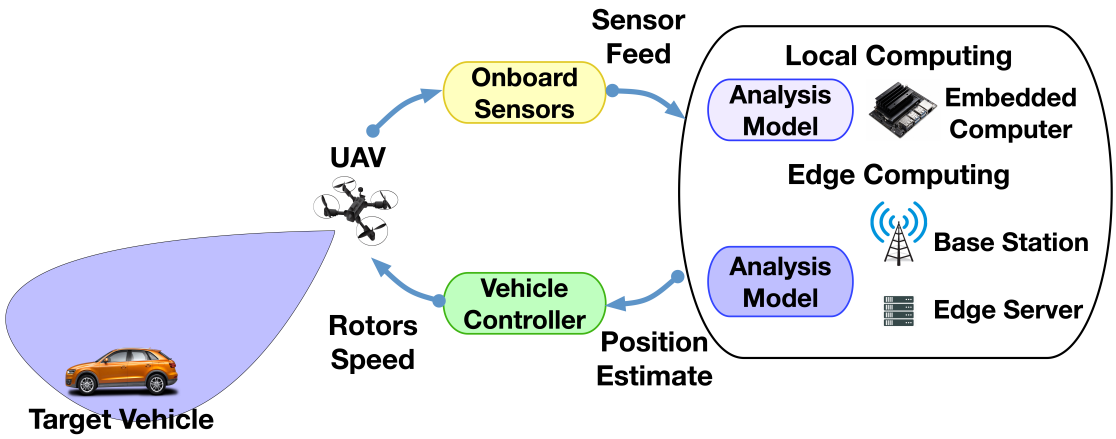

Figure 5.1: High-level overview of the system considered in the chapter. A UAV follows a target using a position estimated by analyzing real-time data acquired by onboard sensors. The UAV can use its on onboard resources or infrastructure-level resources to extract the position. The resulting estimation error and delay are a function of the computing strategy.

ministic; (*ii*) it does not require any use of infrastructure-level resources such as channel and server time; *cons:* (*i*) only small low-performance ANN models can be executed; (*ii*) onboard energy consumption increases significantly.

Edge Computing (EC): *Pros:* (*i*) the edge server can execute large ANN models with good performance; (*ii*) reduced energy consumption onboard the UAV; *cons:* (*i*) the wireless channel may induce large delays and delay variations depending on the propagation environment and network state; (*ii*) intense use of infrastructure-level resources to transport and process the data - which may constrain the number of supported vehicles in practical deployments.

Most current solutions choose one or the other approach depending on expected performance. More recent frameworks [92, 93, 30, 31, 32, 33, 34] dynamically select whether to process information onboard or send it to the edge server to optimize of the tradeoff between resource usage and key performance metrics such as delay and accuracy based on a perceived state of the system - such as channel gain. This chapter makes an important conceptual innovation: we connect the edge computing (EC) vs local computing (LC) dynamic selection process to the control of the flight dynamics of the UAV given a specific task. Although our considerations can be general, we focus on a target tracking application [94], where the task of the UAV is to follow a target, whose position is acquired by onboard sensors, such as cameras and LiDAR. The acquired data are processed using algorithms that extract the position of the target - *e.g.*, an ANN for object detection. Due to this process, the target state observation - the position in the considered case - is delayed and inaccurate, where both these factors are affected by the computing pipeline (LC or EC), the used ANN model, and the state of the overall system in general.

One of the observations at the core of the proposed framework is that the impact of the two non-idealities of state acquisition on mission performance depends on the characteristics of the target's motion. As LC and EC correspond to different distributions of estimation delay and accuracy, the decision process that dynamically

determines the computing pipeline (computing control task) based on the state of the UAV motion controller (flight control task).

CADET: We propose CADET, a framework for the dynamic control of the computing pipeline used by the UAV in response to: (*i*) the current target dynamics - and more specifically the state of the flight controller; (*ii*) onboard energy usage constraints; and (*iii*) channel and server usage constraints. CADET stems from discrete-time Markov jump linear systems (MJLS) [95], a mathematical framework that allows us to jointly consider the dynamics of the motion controller and of the computing pipeline. MJLSs are switched linear systems, where the switching signal is governed by a Markov Chain (MC). We associate states in the MC with the state of the system – in the considered setting the wireless channel state – and identify a model for the system dynamics. The system's state is then linked to the estimation characteristics and connected to the flight controller. In CADET, we embed a MDP in the MJLS instead of the MC to make the process controlled – a Markov Jump Switched Linear System (MJSLS) [96]. This approach enables the control of the system behavior - in our case to determine the computing pipeline based on the output of a Kalman filter we designed for this purpose.

5.2 Related Work

Edge computing is a central component of modern infrastructures and a key enabler of many applications. Recent contributions proposed a wide array of solutions to improve the performance of edge offloading and optimize resource usage. In [30], the authors propose a framework to solve a mixed-integer linear program that jointly optimizes service cashing, cloud usage, and energy consumption. [31] presents a controller to minimize the overall energy consumption under hard per-task delay constraints in systems with energy harvesting. In [32], the authors propose a multi-scale control logic to dynamically reconfigure distributed cloudlets. The framework in [33] jointly considers devices' topology, available resources, and wireless channel state to assign computing tasks to edge servers. In [34], the authors propose a solution to dynamically control ANN-based video analytics on edge servers.

In contrast with these contributions, CADET controls task offloading based on the control needs of an autonomous UAV to reduce resource usage while preserving mission performance. Our framework paves the way to a different perspective on infrastructure support for lightweight autonomous vehicles.

Edge computing in the context of UAV systems has been attracting considerable attention from the research community in recent years. One of the most popular research directions is that of using UAVs as "mobile" edge servers, whose position and configuration can be optimized to reflect the characteristics of the users (*e.g.*, spatial distribution). An overview of this area of investigation can be found in [97]. Our study, instead, focuses on the opposite scenario: where resource-constrained UAVs use infrastructure-level resources to support their real-time computing processes. In this latter area of investigation, recent work has proposed predictive edge server selection methodologies [93] to counteract the extreme mobility of these platforms. In [92], the authors devise an adaptive task replication

strategy to mitigate connectivity uncertainty. Both contributions are based on Deep Reinforcement Learning (DRL).

Closely related to the application scenario considered in this chapter, [94] develops an energy-efficient UAV-aided target tracking system, where the UAV offloads video processing tasks to the edge servers in its flight trajectory. We claim that our approach, which considers the inner state of the flight controller and target motion to determine where to execute computing tasks, introduces significant conceptual and technical innovations in the context of dynamic offloading for UAV systems.

5.3 System Overview and Sensing Model

System and Mission Objective: We consider the system depicted in Fig. 5.1, where an interconnected UAV follows a target with a possibly complex motion. For instance, the UAV may follow a specific car in a city or another drone. Formally, we describe the dynamics of the UAV using the nonlinear differential equation $\dot{x}(t) = f(x(t), u(t))$,

where $x \in \mathbb{R}^{n_x}$ is the state vector composed of the position in the 3D space, angular positions, 3D velocity, and angular velocities, and $u \in \mathbb{R}^{n_u}$ is the control vector. The target state is $x^0 : \mathbb{R}_{\geq 0} \to \mathbb{R}^n$ composed of 3D position and velocity and has piece-wise linear dynamics.

Sensing model: The UAV needs to acquire the target's position to control its motion.

We assume a non-cooperative setting, where the UAV acquires data from onboard sensors at fixed intervals that are processed using an algorithm to extract a position estimate $y_a^0(t_k)$, where t_k is a temporal index. For instance, the UAV may use onboard sensors such as a camera, a stereo camera, or a LiDAR, whose output is processed using a Deep Neural Network for 3D object detection. As a consequence of the scenario described above, the position acquisition is: (i) non-instantaneous, and (ii) imperfect. To reflect these two properties, we express the estimate as $y_a^0(t_k) = x^0(t_k - \tau_k^a) + \epsilon_k^a$,

where $\{\tau_k^a\}_{k\in\mathbb{N}}$ and $\{\epsilon_k^a\}_{k\in\mathbb{N}}$ are discrete-time random processes modeling respectively the estimation delay and error, and a is a discrete, controlled, variable denoting specific sensing-computing pipelines. In our scenario, we consider two main options – LC and EC – which correspond to different delay and noise distributions. For instance, the UAV can use its onboard resources to execute a lightweight estimation model, which leads to a low-delay-low-accuracy estimate or to offload analysis to an edge server. The latter option allows the execution of high-accuracy models; however, it may suffer a larger delay due to the need to transfer information-rich signals over a capacity-constrained wireless channel. We set $a = l$ and $a = s$ to indicate the use of onboard "local" resources and remote "server" resources to process the acquired sensor data, respectively.

We assume a PDP approximable by a one-sided exponential function for both the EC and LC pipelines. We characterize the estimation error identifying a hidden Markov model based on the UAV 123 benchmark [98], which consists of

123 annotated HD video sequences captured from a low-altitude aerial perspective. Specifically, we leverage the methodology in [38] to define the space of the hidden states as $\mathcal{S} = \mathcal{S}_l \cup \mathcal{S}_s$, where with $\mathcal{S}_l$ and $\mathcal{S}_s$ are the state space of the LC and EC computing pipelines. We model the estimation error $\{\epsilon_k\}_{k\in\mathbb{N}}$ as a time-varying Additive White Gaussian Noise (AWGN) whose distribution depends on the state s of the Markov Model, that is, $\epsilon_k \sim \mathcal{N}(0, \Sigma_i)$, $i{=}s(k)$.

5.4 CADET

The core objective of our framework is to seamlessly connect flight control and computing control planes. More in detail, an integral part of our framework is a Markov jump-switching linear process that we use to design a filter to estimate the target position from noisy and delayed observations. Then, we design a methodology to dynamically determine the computing pipeline using the closed-loop measurements of the control system depicted in Fig. 5.2.

Control Objectives: The core observation behind CADET is that different motion parameters and the current target state lead to different computing pipelines - characterized by different delay and error distribution - to be locally optimal from a tracking perspective. This observation is empirically validated in Section 5.5. Moreover, the UAV may have constraints on the use of computing resources. Herein, we consider a scenario where channel usage limitations determine a constraint in the usage of the EC pipeline, even if the approach is extendable to limited onboard energy reservoir scenarios.

Our framework connects two traditionally separate control planes: the process controlling the UAV's 3D flight parameters, given the target position's estimate, is coupled with a process dynamically controlling how the sensor data feed is transformed into the position estimate. Formally, we seek to design the motion control rule $u(t_k) \in \mathbb{R}^m$ to minimize the difference between $x(t_{k+1})$ and $x^0(t_k)$, and the computing control rule $a(k)$ – that determines the estimation delay and noise distributions.

To measure the mission performance, we adopt a metric based on the root-mean-square closed-loop error: $\bar{J}_e = \sum_{k=0}^{D} J_e\left(x(k), x^0(k), a(k)\right)$. We remark that the closed-loop tracking error depends on many factors such as the target dynamics $x^0(t_k)$, the closed-loop UAV flight control and dynamics, *i.e.*, the dynamics $x(t_k)$, control inputs $u(t_k)$, and the computing policy $a(k)$. Importantly, CADET uses information from the closed-loop controller that can be independently solved based on the inputs. In this section, we slightly abuse notation by using $J_e(a(k))$ instead of $J_e\left(x(k), x^0(k), a(k)\right)$, to emphasize the dependence between the tracking error and the computing policy.

At each time step, the computing control policy attempts to solve the following multi-objective problem based on the current measurement $y_a^0(t_k)$

$$\min_{a_{k+1},\ldots,a_D} \quad \sum_{i=1}^{D-k} [J_e(a_{k+i}) + J_c(a_{k+i})] \tag{5.1}$$

$$\text{s.t.} \quad \sum_{i=1}^{D-k} J_e(a_{k+i}) < \bar{J}_e^l, \quad \sum_{i=1}^{D-k} J_e(a_{k+i}) < \bar{J}_e^s, \tag{5.2}$$

$$a_{k+i} \in \{s, l\} \ \forall i \leq D-k, \tag{5.3}$$

where D is the temporal horizon of the mission. The cost-function in Eq. (5.1) considers both the tracking error $J_e(a_{k+i})$, and the computing cost $J_c(a_{k+i})$ in the whole mission. The computing cost $J_c(a_{k+i})$ can be expressed in channel usage. The constraints in Eq. (5.2) emphasize the accuracy improvement requirement concerning the "pure" LC and EC strategies. The parameters $\bar{J}_e^l, \bar{J}_e^s \in \mathbb{R}$ are the total tracking error of the LC and EC strategies, respectively.

The resolution of the optimization problem in Eq. (5.1)-(5.3) is technically challenging: (*i*) the tracking error depends on the state of the target $x^0(t_k)$ that is not directly observable, (*ii*) the cost function depends on the future target behavior that is not known a priori, and (*iii*) the interdependence between the dynamic and computing controllers makes predicting the cost function difficult.

We address the challenges above by proposing a novel methodology that: (*i*) we model the computing decision problem as a Markov decision process that jointly considers the control system metrics and computational requirements; (*ii*) we design a new Kalman filter based on the underlying MDP that can estimate the target state $x^0(k)$ based on the current measurement $y_a^0(t_k)$; (*iii*) we define a quality indicator of the filtered state and design a finite-state machine that controls the MDP based on the quality indicator.

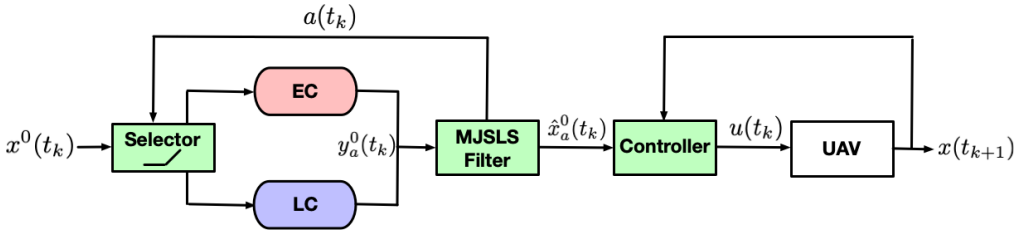

Figure 5.2: Block diagram of CADET.

Jump Switched Linear Filter: The first component of CADET is a *filter* for Markov Jump Switched Linear systems.

The equations for the filter fall into two groups: time update equations and measurement update equations. The time update equations are responsible for projecting forward the current estimation and error covariance estimates. The measurement update equations incorporate the new measurement into the a priori estimate to obtain an a posteriori estimate. One of the technical challenges is that in our problem the estimation process is controlled.

We take the Kalman-like filter design proposed in [95] for MJLS processes as a starting point. Leveraging the definition proposed in [96], we embed an MDP

in the MJLS instead of the MC to control the computational pipeline based on the output of a Kalman filter. We extend existing methodologies for MJSLSs that assume that both the measurement $y^0(t_k)$ and the jump variable $s(k)$ are available. Let the tuple $(\mathcal{S}, \mathcal{A}, \mathbb{P}, s_0)$ be a Markov decision process defined on the probability space $\mathcal{S} = \mathcal{S}_l \cup \mathcal{S}_s$, where $\mathcal{S}$ is a finite set of discrete states, $\mathcal{S}_l$ and $\mathcal{S}_s$ contain the onboard and server states, respectively, and $\mathcal{A} = \{l, s\}$ is the finite set of actions. As defined earlier, the actions l and s indicate the use of onboard "local" resources and remote "server" resources to process the acquired sensor data, respectively. $\mathbb{P} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ is the transition probability function such that for every decision epoch $k \in \mathbb{N}$, $i, j \in \mathcal{S}, a \in \mathcal{A}$,

$s_0 \in \mathcal{S}$ is the initial state.

Notice that, Unlike [96], we do not define the cost function since it is useless for the following discussion.

Let us consider the following continuous-time MJSLS Σ_c,

$$\Sigma_c : \begin{cases} \dot{x}^0(t) = A_c x^0(t), \\ y^0(t_k) = x^0(t_k - \tau_{s(k)}(t_k)) + H_{s(k)} w(t_k), \\ x(0) = x_0,\ s(0) = s_0, \end{cases} \tag{5.4}$$

where $w(t_k)$ is a n-dimensional white noise, and $\mathbb{E}\left[\epsilon_k^{s(k)} \epsilon_k^{s(k)\top}\right] = H_{s(k)}$; $\{y^0(t_k), k \in \mathbb{N}\}$ is the n-dimensional sequence of measurable variables affected by delay $\tau_{s(k)}$ and noise $\epsilon_k^{s(k)} = H_{s(k)} w(t_k)$. The state dynamics in Eq. (5.4) are described in ordinary differential equations as the UAV trajectory evolves in continuous time. However, the output vector is expressed in terms of discrete-time measurements. The hybrid modeling approach allows us to consider real-valued stochastic processes generating delay and accuracy. Since the Markov Jump Switched Linear framework is compatible with discrete-time dynamics, the first step is to discretize the continuous-time dynamic and output in Eq. (5.4), as follows,

$\bar{y}^0(t_k) = L_{s(k)} x^0(t_k) + H_{s(k)} w(t_k)$, where $L_{s(k)} = \exp\left(-A_c \mathbb{E}[\tau_{s(k)}]\right)$, $A = \exp(A_c T)$, $T = t_{k+1} - t_k,\ \forall k \in \mathbb{N}$ is the sampling time.

Then, we estimate the target state using the following dynamic Markov Jump filter,

$$\mathcal{G} : \begin{cases} \hat{x}^0(t_{k+1}) = \hat{A}_{s(k)}(k) \hat{x}^0(t_k) + \hat{B}_{s(k)}(k) \bar{y}(t_k), \\ x(0) = \hat{x}_0, \end{cases} \tag{5.5}$$

where the matrices $\hat{A}_i(k)$ and $\hat{B}_i(k)$ are found via the coupled Riccati differential equations:

$$\begin{aligned} Y_j(k+1) = \sum_{i \in \mathbb{J}(k)} p_{i,j}^{a(k)}(k) \Big[& A Y_i(k) A^\top - A Y_i(k) L_i^\top \times \\ & \left(H_i H_i^\top \pi_i(k) + L_i Y_i(k) L_i^\top\right)^{-1} L_i Y_i(k) A^\top \times \\ & \pi_i(k) G G^\top \Big] \\ Y_j(0) = \pi_j(0) \mathbb{E}\left[(x_0 - \mathbb{E}[x_0])(x_0 - \mathbb{E}[x_0])^\top\right] & \end{aligned} \tag{5.6}$$

and

$$M_i(k) = \begin{cases} -A_i Y_i(k) L_i^\top \left(H_i H_i^\top \pi_i(k) + L_i Y_i(k) L_i^\top \right)^{-1}, & i \in \mathbb{J}(k) \\ 0, & i \notin \mathbb{J}(k) \end{cases} \tag{5.7}$$

where $\pi_j(k) = \mathbb{P}[s(k) = j]$ and $\mathbb{J}(k) = \{i \in \mathcal{S} : \pi_i(k) > 0\}$. Finally,

$$\hat{A}_i(k) = A_i + M_i(k) L_i(k),\ \hat{B}_i(k) = -M_i(k). \tag{5.8}$$

Remark 2. *It is known that for the case where $(y^0(k), \theta(k))$ are available, the best linear estimator of $x^0(k)$ is derived from the Kalman filter for time-varying systems since all the values of the mode of operation are known at time k. Nonetheless, offline computation of the Kalman filter for time-varying systems is practically unfeasible since the gain matrix is sample path dependent, and the number of sample paths grows exponentially with time. On the other hand, the optimal filter in the form of Eq. (5.5) requires a smaller number of pre-computed gains, which means highly reduced memory usage. Specifically, our design has linear memory usage instead of exponential [95].*

Policy "Control-Aware Dynamic Edge Computing":

To measure the filter performance, we use the *Kalman innovation* $\nu_o(k)$, which is defined as the difference between the observation measurement $y^0(t_k)$ and its prediction derived using the information available at time t_{k-1}, that is,

$$\nu_o(k) = y^0(t_k) - L_{s(k)} \hat{x}^0(t_k). \tag{5.9}$$

The Kalman innovation measures the new information obtained by adding another measurement in the estimation process. Indeed, rapid increases in the Kalman innovation indicate that the quality of the filtered state is compromised, for instance, due to discontinuities in the target dynamics. Based on this reasoning, we define the indicator $q(k)$ to drive the control policy based on a Simple Moving Average (SMA) of the previous M innovation data points.

Then, we use the two parameters $\alpha, \beta \in \mathbb{R}_{>0}$ to design a switching rule to commute between local and edge pipelines. In particular, `CADET`'s policy commutes from local to edge computing when the event $r_{l,s} = \{q(k) \geq \alpha\}$ is verified, and switch from edge to local depending on $r_{s,l} = \{q(k) \leq \beta\}$.

At each time step k, we choose the action $a(k)$ based on the current level of innovation. The remark that events we aim to detect are discontinuities in the target motion that determine changes in the optimal delay/accuracy point. Furthermore, α and β determine the channel usage trade-off. Indeed, by varying the parameters, we obtain computing control policies that are more conservative in channel or onboard energy usage. For example, a policy that privileges EC rather than LC (*e.g.*, due to limited energy reservoir available to the UAV) is obtained by decreasing α and increasing β. Conversely, in scenarios where the infrastructure needs to support many UAVs, the policy can privilege LC over EC by increasing α and decreasing β. In both scenarios, `CADET` optimizes the activation of EC and LC based on the Kalman innovation metric to maximize tracking performance.

5.5 CADET Evaluation

This section describes the setting and parameters we use in our evaluation.

(*i*) *System parameters:* We consider a lightweight quadcopter with four rotors directed upwards. From the center of mass of the quadcopter, rotors are placed in a square formation at equal distances. We use the non-linear continuous-time motion model derived following the approach proposed in [99]. The control strategy implements the Nonlinear Model Predictive Control proposed in [100]. The target state is $x^0(t) : \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}^n$ composed of 3D position and velocity, and piece-wise linear dynamics.

(*ii*) *Sensing parameters*: Based on the dasatet in [98], we leverage the methodology in [38] to the state space of the hidden-Markov model $\mathcal{S} = \mathcal{S}_l \cup \mathcal{S}_s$ and, the transition probabilities defined in Sec. 5.4. When EC is used, we assume that the position and velocity estimate are affected by AWGN with standard deviation identified based on the current hidden state of the Markov model. Instead, when LC is active, the standard deviations of the AWGN are amplified by a factor of 6.5.

We assume a one-sided exponential density function to generate the delays of the LC and EC pipelines, with expected values equal to 0.15s and 0.3, respectively.

(*iii*) *Numerical results*: We consider a target with the initial position on the origin of the 3-dimensional reference system and the initial velocities are $v_x = 3$ m/s, $v_y = 4$ m/s, $v_y = 2$ m/s. The target instantaneously changes its velocity after 10 seconds. To statistically validate the control performance, we ran Monte Carlo simulations generating 50 admissible trajectories, each with 200 samples (corresponding to 20s). We aim to demonstrate the ability of CADET to decrease channel usage by smartly changing computing policy while assuring system accuracy. Fig. 5.3 shows the MRMSE along the trajectory for LC (blue), EC (red), and

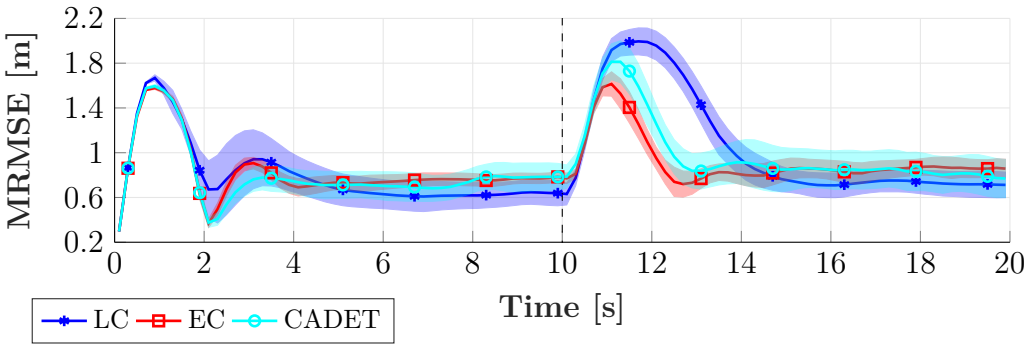

Figure 5.3: MRMSE. The black dashed corresponds to the target velocity change point.

CADET (cyan) computing strategy. We set the parameters to make CADET prioritize LC over EC, ($\alpha = 7, \beta = 1.5$), that is, we consider a case where the objective of control is to reduce infrastructure-resource usage by activating EC only when needed to avoid degradation of tracking error. It can be seen how EC achieves faster convergence and a lower overshoot around the discontinuity compared to

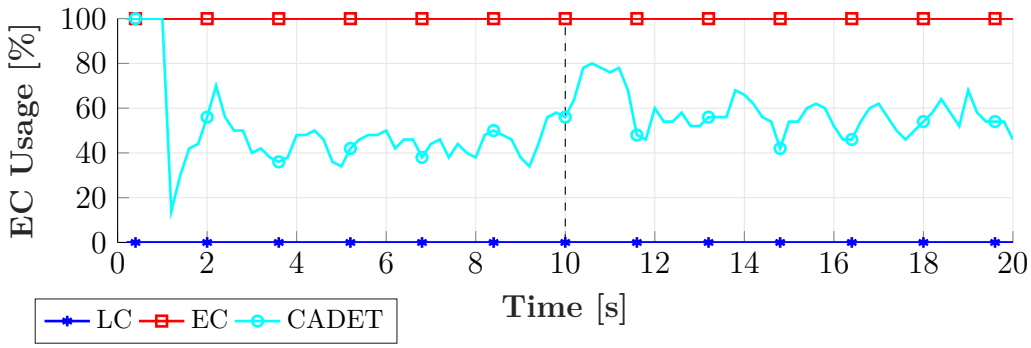

Figure 5.4: EC usage.

LC but has a higher permanent error. `CADET` dynamically changes the policy by preferring EC after the discontinuity to mitigate the overshoot and rapidly decrease the error, then the EC usage can be reduced. We show in Fig. 5.4 the average EC usage of each strategy. The average is taken at each time step over the Monte Carlo simulations. The trend demonstrates the strong dependency between EC usage and the characteristics of the target motion.

To characterize the dependency between the MRMSE and the target motion, we show in figure 5.5 the cumulative average of the MRMSE performed by the three strategies, where for each time-step, we compute the average MRMSE from the initial time-step. Significantly, by efficiently shaping the computing strategy, `CADET` mitigates the non-idealities of both LC and EC state acquisition and outperforms both LC and EC: the accuracy gain reaches 15.2% concerning LC and 6.2% to EC. The results are achieved by using 54% of channel resources.

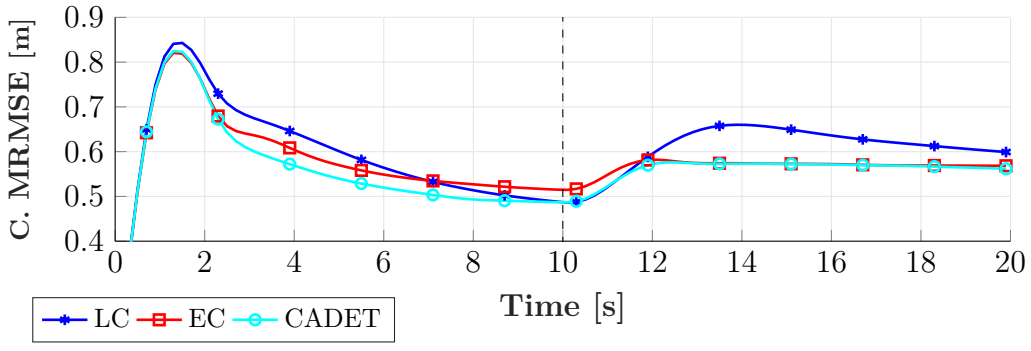

Figure 5.5: Cumulative MRMSE.

5.6 Conclusions

This chapter proposed an innovative approach – `CADET` – to dynamically select the computing pipeline – LC or EC – used by a UAV to estimate the position of a target from sensor feeds such as 2D and 3D imaging. The core conceptual

innovation of `CADET` is to make the selection process aware of the position estimation process and, thus, of the target motion, in addition to traditional energy and channel/server usage metrics.

The core of the proposed approach is a filter whose output controls a Markov linear jump process tracking the system state. Computing decisions are based on metric measuring innovation in the filter's output signal, whose pattern reflects the effectiveness of different options to track the target.

Our results demonstrate the ability of `CADET` to adapt the computing strategy at a fine temporal granularity to motion discontinuities to maximize estimation convergence speed. With respect to LC and EC, the accuracy of our methodology increases by up to 15% depending on the target motion characteristics, while onboard and server resources are parsimoniously used based on preset parameters.

6
Conclusions

The main objective of this thesis is to investigate the importance of Data-Driven techniques in ICT Engineering. The introduction shows that the scientific community has introduced Data-driven methodologies in several wired and wireless communication systems. All past and present generations of communication networks are based on mathematical models derived from theoretical considerations. Indeed, all phases of network design employ physical models describing in quantitative terms the effect each system component has on the overall performance. However, as shown in this thesis, innovative mixed data-driven-model based approaches outperform traditional pure model-based methodologies:

(1) In chapter 2, we presented a taps reduction methodology that merges the cross-correlation and PCA and discussed its impact in terms of BER of the DSP algorithm. We validated the proposed approach on the data of the world-first SDM multi-core fiber field trial conducted within the INCIPICT project in the city of L'Aquila, Italy. Results showed a reduction in the number of taps of the 60% and 50% with respect to the classical approach proposed in [48] considering a transmission over 650 and 3450 km respectively.

(2) chapter 3 provides a novel technique to learn Markov models representing fading wireless channels. We consider a validation scenario consisting of a WNCS that exploits a WirelessHART radio link to send the optimal control inputs generated by a Stochastic MPC, and show that the control performances of our data-driven approach and of a physics-based approach based on a stationary finite-state Markov chain are extremely close: this implies that in practical applications when assuming perfect knowledge of the channel model and parameters is not possible, the methodology presented in chapter 3 is a valid and very effective alternative.

(3) In chapter 4, we provided a technique to reduce the number of discrete modes in a SARX model identified via Regression Trees without compromising (indeed improving substantially, in our case of study) the prediction accuracy and mitigating the overfitting problems due to the identification procedure. We validated the

proposed methodology on a benchmark consisting on a nonlinear inverted pendulum over a cart, by comparing its prediction accuracy and the number of discrete modes with respect to the model identified using the approach in [61]. Results show an important reduction in the number of modes of the identified model that ranges between 96% and 99.74%. As our complexity reduction algorithm also alleviates overfitting issues, the overall prediction accuracy improves as well. The reduced model has also been tested in terms of closed-loop control performance: we compared two MPC algorithms, one using the reduced model and the other using an *oracle*, i.e. perfect knowledge of the plant dynamics. Results show that the control performance of the reduced model and the *oracle* are very close. Thus in practical applications, when assuming a perfect knowledge of the model is not possible, the methodology presented in this paper is a valid and computationally efficient alternative.

(4) chapter 5 proposed an innovative approach – `CADET` – to dynamically select the computing pipeline – LC or EC – used by a UAV to estimate the position of a target from sensor feeds such as 2D and 3D imaging. The core conceptual innovation of `CADET` is to make the selection process aware of the position estimation process and, thus, of the target motion, in addition to traditional energy and channel/server usage metrics. The core of the proposed approach is a filter whose output controls a Markov linear jump process tracking the system state. Computing decisions are based on metric measuring innovation in the filter's output signal, whose pattern reflects the effectiveness of different options to track the target. Our results demonstrate the ability of `CADET` to adapt the computing strategy at a fine temporal granularity to motion discontinuities to maximize estimation convergence speed. With respect to LC and EC, the accuracy of our methodology increases by up to 15% depending on the target motion characteristics, while onboard and server resources are parsimoniously used based on preset parameters.

References

[1] Aloysius Mok Deji Chen Mark Nixon. *WirelessHART*. Springer, 2010.

[2] Alessio Zappone, Marco Di Renzo, and Mérouane Debbah. "Wireless networks design in the era of deep learning: Model-based, AI-based, or both?" In: *IEEE Transactions on Communications* 67.10 (2019), pp. 7331–7376.

[3] NGMN Alliance. "5G white paper". In: *Next generation mobile networks, white paper* 1 (2015).

[4] Guangxu Zhu et al. "Toward an intelligent edge: Wireless communication meets machine learning". In: *IEEE communications magazine* 58.1 (2020), pp. 19–25.

[5] Francesco Musumeci et al. "An overview on application of machine learning techniques in optical networks". In: *IEEE Communications Surveys & Tutorials* 21.2 (2018), pp. 1383–1408.

[6] Mario Bkassiny, Yang Li, and Sudharman K Jayaweera. "A survey on machine-learning techniques in cognitive radios". In: *IEEE Communications Surveys & Tutorials* 15.3 (2012), pp. 1136–1159.

[7] Jithin Jagannath et al. "Machine learning for wireless communications in the Internet of Things: A comprehensive survey". In: *Ad Hoc Networks* 93 (2019), p. 101913.

[8] Parastoo Sadeghi et al. "Finite-state Markov modeling of fading channels-a survey of principles and applications". In: *IEEE Signal Processing Magazine* 25.5 (2008), pp. 57–80.

[9] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. *The elements of statistical learning*. Vol. 1. 10. Springer series in statistics New York, 2001.

[10] Christopher M Bishop and Nasser M Nasrabadi. *Pattern recognition and machine learning*. Vol. 4. 4. Springer, 2006.

[11] Trevor Hastie et al. *The elements of statistical learning: data mining, inference, and prediction*. Vol. 2. Springer, 2009.

[12] Leo Breiman. *Classification and regression trees*. Routledge, 2017.

[13] Leo Breiman. "Random forests". In: *Machine learning* 45.1 (2001), pp. 5–32.

[14] Alex J Smola and Bernhard Schölkopf. "A tutorial on support vector regression". In: *Statistics and computing* 14.3 (2004), pp. 199–222.

[15] Olivier Bousquet, Ulrike von Luxburg, and Gunnar Rätsch. *Advanced Lectures on Machine Learning: ML Summer Schools 2003, Canberra, Australia, February 2-14, 2003, Tübingen, Germany, August 4-16, 2003, Revised Lectures*. Vol. 3176. Springer, 2011.

[16] Lucian Busoniu et al. *Reinforcement learning and dynamic programming using function approximators*. CRC press, 2017.

[17] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[18] Jiawei Han, Jian Pei, and Hanghang Tong. *Data mining: concepts and techniques*. Morgan kaufmann, 2022.

[19] Javier Mata et al. "Artificial intelligence (AI) methods in optical networks: A comprehensive survey". In: *Optical switching and networking* 28 (2018), pp. 43–57.

[20] Emmanuel Seve et al. "Learning process for reducing uncertainties on network parameters and design margins". In: *Journal of Optical Communications and Networking* 10.2 (2018), A298–A306.

[21] Tania Panayiotou, Georgios Ellinas, and Sotirios P. Chatzis. "A data-driven QoT decision approach for multicast connections in metro optical networks". In: *2016 International Conference on Optical Network Design and Modeling (ONDM)*. 2016, pp. 1–6.

[22] Tania Panayiotou, Sotirios P Chatzis, and Georgios Ellinas. "Performance analysis of a data-driven quality-of-transmission decision approach on a dynamic multicast-capable metro optical network". In: *Journal of Optical Communications and Networking* 9.1 (2017), pp. 98–108.

[23] Cristina Rottondi et al. "Machine-learning method for quality of transmission prediction of unestablished lightpaths". In: *Journal of Optical Communications and Networking* 10.2 (2018), A286–A297.

[24] Darko Zibar et al. "Machine learning techniques in optical communication". In: *Journal of Lightwave Technology* 34.6 (2015), pp. 1442–1452.

[25] Darko Zibar et al. "Nonlinear impairment compensation using expectation maximization for dispersion managed and unmanaged PDM 16-QAM transmission". In: *Optics express* 20.26 (2012), B181–B196.

[26] Danshi Wang et al. "Nonlinear decision boundary created by a machine learning-based classifier to mitigate nonlinear phase noise". In: *2015 European Conference on Optical Communication (ECOC)*. IEEE. 2015, pp. 1–3.

[27] Xiaochen Li, Haibo He, and Yu-Dong Yao. "Reinforcement learning based adaptive rate control for delay-constrained communications over fading channels". In: *The 2010 International Joint Conference on Neural Networks (IJCNN)*. IEEE. 2010, pp. 1–7.

[28] Kavé Salamatian and Sandrine Vaton. "Hidden markov modeling for network communication channels". In: *ACM SIGMETRICS Performance Evaluation Review* 29.1 (2001), pp. 92–101.

[29] P. Park et al. "Wireless Network Design for Control Systems: A Survey". In: *IEEE Communications Surveys Tutorials* 20.2 (2018), pp. 978–1013.

[30] Tuyen X Tran, Kevin Chan, and Dario Pompili. "Costa: Cost-aware service caching and task offloading assignment in mobile-edge computing". In: *2019 16th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON)*. IEEE. 2019, pp. 1–9.

[31] Thembelihle Dlamini and Ángel Fernández Gambın. "Adaptive resource management for a virtualized computing platform within edge computing". In: *2019 16th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON)*. IEEE. 2019, pp. 1–9.

[32] Lei Jiao et al. "Multiple granularity online control of cloudlet networks for edge computing". In: *2018 15th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON)*. IEEE. 2018, pp. 1–9.

[33] Chang Shu et al. "Dependency-aware and latency-optimal computation offloading for multi-user edge computing networks". In: *2019 16th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON)*. IEEE. 2019, pp. 1–9.

[34] Tianxiang Tan and Guohong Cao. "Deep learning video analytics on edge computing devices". In: *2021 18th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON)*. IEEE. 2021, pp. 1–9.

[35] Tetsuya Hayashi et al. "Field-Deployed Multi-Core Fiber Testbed". In: *2019 24th OptoElectronics and Communications Conference (OECC) and 2019 International Conference on Photonics in Switching and Computing (PSC)*. 2019, pp. 1–3.

[36] Roland Ryf et al. "Transmission over Randomly-Coupled 4-Core Fiber in Field-Deployed Multi-Core Fiber Cable". In: *2020 European Conference on Optical Communications (ECOC)*. IEEE. 2020, pp. 1–4.

[37] Luis Felipe Florenzan Reyes et al. "Data-driven efficient digital signal processing over a field trial space-division multiplexed fiber-optic transmission". In: *2022 International Conference on Computer Communications and Networks (ICCCN)*. IEEE. 2022, pp. 1–8.

[38] Luis Felipe Florenzan Reyes et al. "Learning Markov models of fading channels in wireless control networks: a regression trees based approach". In: *2021 29th Mediterranean Conference on Control and Automation (MED)*. IEEE. 2021, pp. 232–237.

[39] Luis Felipe Florenzan Reyes, Francesco Smarra, and Alessandro D'Innocenzo. "Reduced SARX modeling and control via Regression Trees". In: *2022 American Control Conference (ACC)*. IEEE. 2022, pp. 2110–2115.

[40] Reyes Luis Felipe Florenzan et al. "Control-Aware Dynamic Edge Computing for Real-Time Target Tracking in UAV Systems". In: *International Conference on Acoustics, Speech, and Signal Processing*. IEEE. 2023.

[41] Peter J. Winzer and David T. Neilson. "From Scaling Disparities to Integrated Parallelism: A Decathlon for a Decade". In: *Journal of Lightwave Technology* 35.5 (2017), pp. 1099–1115.

[42] Peter J. Winzer, David T. Neilson, and Andrew R. Chraplyvy. "Fiber-optic transmission and networking: the previous 20 and the next 20 years [Invited]". In: *Opt. Express* 26.18 (Sept. 2018), pp. 24190–24239. URL: `http://opg.optica.org/oe/abstract.cfm?URI=oe-26-18-24190`.

[43] René-Jean Essiambre et al. "Capacity Limits of Optical Fiber Networks". In: *Journal of Lightwave Technology* 28.4 (2010), pp. 662–701.

[44] Benjamin J. Puttnam, Georg Rademacher, and Ruben S. Luís. "Space-division multiplexing for optical fiber communications". In: *Optica* 8.9 (Sept. 2021), pp. 1186–1203. URL: http://opg.optica.org/optica/abstract.cfm?URI=optica-8-9-1186.

[45] Roland Ryf and Cristian Antonelli. "Space-Division Multiplexing". In: *Springer Handbook of Optical Networks*. Springer, 2020, pp. 353–393.

[46] Bruce Thompson. *Canonical correlation analysis: Uses and interpretation*. 47. Sage, 1984.

[47] Md Saifuddin Faruk and Kazuro Kikuchi. "Adaptive frequency-domain equalization in digital coherent optical receivers". In: *Optics Express* 19.13 (2011), pp. 12789–12798.

[48] S Randel et al. "MIMO-based signal processing of spatially multiplexed 112-Gb/s PDM-QPSK signals using strongly-coupled 3-core fiber". In: *2011 37th European Conference and Exhibition on Optical Communication*. IEEE. 2011, pp. 1–3.

[49] Michael Rice. *Digital communications: a discrete-time approach*. Pearson Education India, 2009.

[50] Ian T Jolliffe and Jorge Cadima. "Principal component analysis: a review and recent developments". In: *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences* 374.2065 (2016), p. 20150202.

[51] C. Lu et al. "Real-Time Wireless Sensor-Actuator Networks for Industrial Cyber-Physical Systems". In: *Proceedings of the IEEE* 104.5 (2016), pp. 1013–1024.

[52] A. Ahlen et al. "Toward Wireless Control in Industrial Process Automation: A Case Study at a Paper Mill". In: *IEEE Control Systems Magazine* 39.5 (2019), pp. 36–57.

[53] Luca Schenato et al. "Foundations of control and estimation over lossy networks". In: *Proceedings of the IEEE* 95.1 (2007), pp. 163–187.

[54] Y. Zacchia Lun and A. D'Innocenzo. "Stabilizability of Markov jump linear systems modeling wireless networked control scenarios". In: *2019 IEEE 58th Conference on Decision and Control (CDC)*. 2019, pp. 5766–5772.

[55] W. Turin and R. van Nobelen. "Hidden Markov modeling of flat fading channels". In: *IEEE Journal on Selected Areas in Communications* 16.9 (1998), pp. 1809–1817.

[56] Jeff A Bilmes. "A Gentle Tutorial of the EM Algorithm and its Application to Parameter Estimation for Gaussian Mixture and Hidden Markov Models (No. TR-97-021)". In: *Berkeley, CA: International Computer Science Institute and Univ. Calif. Berkeley* (1998).

[57] R Lawrance and A Rabiner. "A tutorial on hidden Markov models and selected applications in speech recognition". In: *Proceedings of the IEEE* 77.2 (1989), pp. 257–286.

[58] Kavé Salamatian and Sandrine Vaton. "Hidden Markov Modeling for Network Communication Channels". In: *SIGMETRICS Perform. Eval. Rev.* 29.1 (June 2001), pp. 92–101. URL: https://doi.org/10.1145/384268.378439.

[59] P Sadeghi et al. “Finite-state Markov modeling of fading channels”. In: *IEEE Signal Processing Magazine* 57 (2008).

[60] Yuriy Zacchia Lun et al. “On the impact of accurate radio link modeling on the performance of WirelessHART control networks”. In: *IEEE INFOCOM 2020-IEEE Conference on Computer Communications*. IEEE. 2020, pp. 2430–2439.

[61] Francesco Smarra et al. “Data-driven switching modeling for MPC using Regression Trees and Random Forests”. In: *Nonlinear Analysis: Hybrid Systems* 36 (2020), p. 100882.

[62] IEEE 802.15.4-2006. *IEEE Standard for Information technology – Local and metropolitan area networks – Specific requirements – Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low Rate Wireless Personal Area Networks (WPANs)*. Standard. IEEE, Sept. 2006.

[63] Andrea Goldsmith. *Wireless communications*. Cambridge university press, 2005.

[64] Carlo Fischione, Fabio Graziosi, and Fortunato Santucci. “Approximation for a Sum of On-Off Log-Normal Processes With Wireless Applications”. In: *IEEE Trans. Commun.* 55.9 (Sept. 2007), pp. 1822–1822.

[65] K. E. Baddour and N. C. Beaulieu. “Autoregressive modeling for fading channel simulation”. In: *IEEE Transactions on Wireless Communications* 4.4 (July 2005), pp. 1650–1662.

[66] Steven M Kay. *Modern spectral estimation*. Pearson Education India, 1988.

[67] Daniele Bernardini and Alberto Bemporad. “Stabilizing model predictive control of stochastic constrained linear systems”. In: *IEEE Transactions on Automatic Control* 57.6 (2011), pp. 1468–1480.

[68] Andrea Garulli, Simone Paoletti, and Antonio Vicino. “A survey on switched and piecewise affine system identification”. In: *IFAC Proceedings Volumes* 45.16 (2012), pp. 344–355.

[69] Simone Paoletti et al. “Identification of hybrid systems a tutorial”. In: *European journal of control* 13.2-3 (2007), pp. 242–260.

[70] Jonas Sjöberg et al. “Nonlinear black-box modeling in system identification: a unified overview”. In: *Automatica* 31.12 (1995), pp. 1691–1724.

[71] Francesco Smarra et al. “Data-driven model predictive control using random forests for building energy optimization and climate control”. In: *Applied energy* 226 (2018), pp. 1252–1272.

[72] Fabien Lauer. “On the complexity of piecewise affine system identification”. In: *Automatica* 62 (2015), pp. 148–153.

[73] Fabien Lauer. “On the complexity of switching linear regression”. In: *Automatica* 74 (2016), pp. 80–83.

[74] Valentina Breschi, Dario Piga, and Alberto Bemporad. “Piecewise affine regression via recursive multiple least squares and multicategory discrimination”. In: *Automatica* 73 (2016), pp. 155–162.

[75] Alberto Bemporad et al. “Fitting jump models”. In: *Automatica* 96 (2018), pp. 11–21.

[76] Laurent Bako et al. “A recursive identification algorithm for switched linear/affine models”. In: *Nonlinear Analysis: Hybrid Systems* 5.2 (2011), pp. 242–253.

[77] A. Bemporad et al. “A bounded-error approach to piecewise affine system identification”. In: *IEEE Transactions on Automatic Control* 50.10 (2005), pp. 1567–1580.

[78] Leo Breiman et al. *Classification and regression trees*. CRC press, 1984.

[79] Vittorio De Iuliis et al. “A Comparison of Classical Identification and Learning-Based Techniques for Cyber-Physical Systems”. In: *2021 29th Mediterranean Conference on Control and Automation (MED)*. IEEE. 2021, pp. 179–185.

[80] Vittorio De Iuliis et al. “On the Stability of Switched ARX Models, with an Application to Learning via Regression Trees”. In: *IFAC-PapersOnLine* 54.5 (2021), pp. 61–66.

[81] Felix Bünning et al. “Experimental demonstration of data predictive control for energy optimization and thermal comfort in buildings”. In: *Energy and Buildings* 211 (2020), p. 109792.

[82] Giovanni D Di Girolamo et al. “Data-driven optimal predictive control of seismic induced vibrations in frame structures”. In: *Structural Control and Health Monitoring* 27.4 (2020), e2514.

[83] Francesco Smarra et al. “Learning models for seismic-induced vibrations optimal control in structures via random forests”. In: *Journal of Optimization Theory and Applications* 187.3 (2020), pp. 855–874.

[84] Tobias Geyer, Fabio D Torrisi, and Manfred Morari. “Optimal complexity reduction of polyhedral piecewise affine systems”. In: *Automatica* 44.7 (2008), pp. 1728–1740.

[85] Frank J Christophersen et al. “Controller complexity reduction for piecewise affine systems through safe region elimination”. In: *2007 46th IEEE Conference on Decision and Control*. IEEE. 2007, pp. 4773–4778.

[86] Steven L Brunton and J Nathan Kutz. *Data-driven science and engineering: Machine learning, dynamical systems, and control*. Cambridge University Press, 2019.

[87] Mingxing Tan, Ruoming Pang, and Quoc V. Le. *EfficientDet: Scalable and Efficient Object Detection*. 2020. arXiv: `1911.09070 [cs.CV]`.

[88] Tsung-Yi Lin et al. *Microsoft COCO: Common Objects in Context*. 2015. arXiv: `1405.0312 [cs.CV]`.

[89] Reza Yazdani et al. “The dark side of DNN pruning”. In: *2018 ACM/IEEE 45th Annual International Symposium on Computer Architecture (ISCA)*. IEEE. 2018, pp. 790–801.

[90] Joseph Redmon and Ali Farhadi. *YOLOv3: An Incremental Improvement*. 2018. arXiv: `1804.02767 [cs.CV]`.

[91] Jiasi Chen and Xukan Ran. “Deep learning with edge computing: A review”. In: *Proceedings of the IEEE* 107.8 (2019), pp. 1655–1674.

[92] Davide Callegaro, Marco Levorato, and Francesco Restuccia. "SeReMAS: Self-Resilient Mobile AutonomousSystems Through Predictive Edge Computing". In: *CoRR* abs/2105.15105 (2021). arXiv: 2105.15105. URL: https://arxiv.org/abs/2105.15105.

[93] Heting Liu and Guohong Cao. "Deep Reinforcement Learning-Based Server Selection for Mobile Edge Computing". In: *IEEE Transactions on Vehicular Technology* 70.12 (2021), pp. 13351–13363.

[94] Xiaoheng Deng et al. "Energy-Efficient UAV-Aided Target Tracking Systems Based on Edge Computing". In: *IEEE Internet of Things Journal* (2021).

[95] Oswaldo Luiz Valle Costa, Marcelo Dutra Fragoso, and Ricardo Paulino Marques. *Discrete-time Markov jump linear systems*. Springer Science & Business Media, 2006.

[96] Yuriy Zacchia Lun, Alessandro D'Innocenzo, and Maria Domenica Di Benedetto. "Robust LQR for time-inhomogeneous Markov jump switched linear systems". In: *IFAC-PapersOnLine* 50.1 (2017). 20th IFAC World Congress, pp. 2199–2204.

[97] Fuhui Zhou et al. "Mobile Edge Computing in Unmanned Aerial Vehicle Networks". In: *IEEE Wireless Communications* 27.1 (2020), pp. 140–146.

[98] Matthias Mueller, Neil Smith, and Bernard Ghanem. "A benchmark and simulator for uav tracking". In: *European conference on computer vision*. Springer. 2016, pp. 445–461.

[99] Teppo Luukkonen. "Modelling and control of quadcopter Independent research project in applied mathematics". In: *Espoo* 22 (2011), p. 22.

[100] MATLAB - v. R2019a. *Parallel Computing Toolbox*. Natick, Massachusetts, 2019. URL: https://it.mathworks.com/help/pdf_doc/parallel-computing/distcomp.pdf.