POLITECNICO DI MILANO
DIPARTIMENTO DI ELETTRONICA, INFORMAZIONE E BIOINGEGNERIA
DOCTORAL PROGRAMME IN INFORMATION TECHNOLOGY

A RANDOMIZED MODEL STRUCTURE SELECTOR FOR COMPLEX DYNAMICAL SYSTEMS

Doctoral Dissertation of:
Federico Bianchi

Supervisor:
Prof. Luigi Piroddi

Tutor:
Prof. Simone Garatti

Chair of the Doctoral Program:
Prof. Barbara Pernici

2019 – XXXII Cycle

To my family

“La mente non è un vaso da riempire, ma un fuoco da accendere.”

— Plutarco

Ringraziamenti

Eccomi giunto alla conclusione di questo mio dottorato, dopo tre anni intensi. Il risultato scientifico di cui sono orgoglioso è stato riassunto in questo elaborato. Ciò che invece non traspare da queste pagine, è quello che questo cammino fatto anche di persone e incontri ha rappresentato dal punto di vista umano. Voglio quindi prendermi questo poco spazio per ringraziare tutte quelle persone che hanno lasciato un segno.

Il primo ringraziamento, non perchè dovuto, ma sentito, va a Luigi, mia guida in questo viaggio. Una persona mite, sempre presente e disponibile, prodiga di consigli e idee, ma mai invadente. A lui devo principalmente questo dottorato, perchè si è speso in prima persona affinchè io potessi prenderne parte. Da Luigi ho cercato di apprendere molto: la dedizione al lavoro, la voglia di migliorarsi sempre e il non sottovalutare mai i risultati ottenuti. Un grazie anche a Maria, Alessandro e Simone per avermi dato la possibilità di collaborare con loro.

In secondo luogo, il mio ringraziamento va alla mia famiglia, che mi ha sempre supportato nelle mie decisioni e mi è sempre stata vicina. Grazie a Elena, mia compagna di vita, per avermi spesso spronato a dare il meglio di me e per aver rappresentato un importante valvola di sfogo.

Grazie ad Andrea, insostituibile compagno di ufficio, ma soprattutto amico sincero. Un grazie anche al gruppo ATG del Polo di Como, per aver condiviso lieti momenti conviviali e di svago.

Infine, un ringraziamento a tutti i miei amici che in un modo o nell'altro sono riusciti a strapparmi una risata anche nei momenti di maggiore tensione.

La conferma della bontà dell'esperienza formativa vissuta e del lavoro fatto arriverà col tempo, ne sono certo. Sin da subito invece posso felicemente dire che come persona mi sento molto arricchito, e questo grazie a tutte queste persone.

Sommario

In questa tesi si affronta il problema relativo alla selezione della struttura del modello (MSS, dall'inglese *Model Structure Selection*) di un sistema dinamico, quando l'apprendimento del modello basato sui dati è perseguito con metodi parametrici. Come noto, il tentativo di dare soluzione a tale problema risulta particolarmente arduo a causa della natura combinatoria dello stesso che, in linea di principio, richiede di esplorare in modo esaustivo lo spazio delle possibili strutture di modello, al fine di selezionare la migliore. Tale spazio cresce però rapidamente all'aumentare del numero dei termini di modello considerati, rendendo l'approccio esaustivo applicabile solo su problemi a ridotta scala. Sono state pertanto proposte svariate strategie con l'obiettivo di esplorare in modo intelligente lo spazio delle strutture di modello: euristiche basate su una costruzione incrementale della struttura, algoritmi ispirati al principio della selezione naturale ed evoluzione biologica, tecniche basate su regolarizzazione, nonché approcci probabilistici. Seguendo tale ultima impostazione, questa tesi estende un recente approccio probabilistico originariamente proposto per la risoluzione del problema MSS nel caso di sistemi non lineari, al caso di dati distribuiti, sistemi non lineari commutati e anche alla stima della matrice di covarianza di processo in applicazioni basate sul filtro di Kalman.

Nel caso in cui i dati siano distribuiti tra più agenti e non possano essere raccolti in un'unità centrale - come generalmente si assume nei problemi di apprendimento del modello basato sui dati - il problema MSS e la stima dei parametri devono essere risolti in modo cooperativo. Considerando modelli parametrici appartenenti alla famiglia NARX, in questa tesi si affrontano, dunque, queste due problematiche ricorrendo a uno schema risolutivo di tipo distribuito che mira a raggiungere congiuntamente un valore comune tra gli agenti sia per la struttura del modello che per le stime dei parametri, sfruttando la riformulazione probabilistica del problema MSS.

I sistemi (non lineari) a commutazione sono caratterizzati da dinamiche continue che rappresentano il comportamento del sistema in diverse condizioni operative (modalità), indicizzate da un segnale di commutazione discreto. In questo caso, il problema MSS comprende la selezione di una struttura di modello per ciascuna modalità e anche la ricostruzione del segnale di commutazione. Sfortunatamente, poiché le commutazioni

di modalità possono avvenire in modo arbitrario nel tempo, lo spazio delle strutture di modello cresce rapidamente con il numero di dati, aggravando così in modo significativo il problema MSS. Questa tesi propone, al riguardo, un metodo di identificazione iterativo che allevia la crescente complessità combinatoria adottando un approccio a due fasi, che possono essere così sintetizzate: nella prima fase, la ricostruzione del segnale di commutazione, la selezione delle strutture NARX e la stima dei parametri sono risolte congiuntamente mediante una strategia di campionamento-e-valutazione, fissando a priori i possibili istanti di commutazione; nella seconda fase, il posizionamento degli istanti di commutazione viene perfezionato sulla base delle prestazioni del modello ottenuto nella fase precedente.

L'apprendimento di modelli basato sui dati svolge un ruolo importante anche nel contesto del filtraggio, in quanto le statistiche sul rumore di output e di processo, necessarie per la formulazione del filtro di Kalman, sono generalmente sconosciute e devono essere pertanto stimate dai dati. Un problema particolarmente impegnativo è la stima delle statistiche del rumore di processo che tiene conto delle dinamiche di sistema non modellate e su cui in genere non è disponibile alcuna conoscenza a priori. Questa tesi affronta il problema di selezionare una parametrizzazione adatta per la matrice di covarianza del rumore di processo, considerando questo problema come un caso particolare del problema MSS.

Abstract

THIS thesis addresses the problem of choosing suitable model structures for dynamical systems when the data-driven model learning is pursued with parametric methods. The *model structure selection* (MSS) problem is known to be challenging due to its combinatorial nature which requires in principle to exhaustively search for the model terms to be included into the model within a space that might be large. Accordingly, many strategies have been proposed with the aim of exploring in a smart way the model structure space, ranging from greedy incremental policies, regularization based techniques, evolutionary methods, and probabilistic approaches. This thesis extends a recent randomized approach based on a probabilistic reformulation of the MSS problem for nonlinear systems, to the case of distributed data, switched nonlinear systems, and also to the estimation of the process covariance matrix in Kalman filter applications.

When data are distributed among multiple agents and cannot be made centrally available, the MSS and the parameter estimation tasks have to be solved cooperatively. Within the NARX modeling framework, we address this issue by resorting to a distributed scheme which aims at reaching a common value for both the model structure and the parameter estimates in an integrated fashion, taking advantage from the probabilistic reformulation of the MSS problem.

Switched (nonlinear) systems are characterized by the interaction between continuous and discrete dynamics, the former representing the system behavior in different operational conditions (modes), indexed by a discrete switching signal. In this case, the MSS problem encompasses the selection of a model structure for each mode, and also the reconstruction of the switching signal. Unfortunately, since switchings can occur arbitrarily in time, the model structure space grows rapidly with the number of data, thus aggravating significantly the MSS problem. This thesis proposes an iterative identification method which alleviates the combinatorial complexity by adopting a two-stage approach. More precisely, in the first stage, candidate mode switching instants are fixed and adopted to jointly reconstruct the switching signal and solve the NARX structure and parameter identification via a sample-and-evaluate strategy; in the second stage, the positioning of the switching instants is refined.

Data-driven model learning plays an important role also in the Kalman filtering context, where the output and process noise statistics are generally unknown and must to be estimated from data. A particularly challenging problem is the estimation of the process noise statistics that account for the unmodeled dynamics on which typically no prior knowledge is available. This thesis addresses the problem of choosing a suitable parameterization for the process noise covariance matrix, viewing it as a specialization of a classical MSS problem.

The solutions we proposed in this thesis to the three mentioned problems are supported by the results obtained in several simulation studies.

Contents

CHAPTER 1

Introduction

THE art of inferring models from observed data goes back to the work of Gauss and Legendre in the late 18th and early 19th century. It developed rapidly, achieving success in many disciplines, and particularly in the engineering field, where one is generally interested in a quantitative assessment of the behavior of a dynamical system through a mathematical description of it. This data-driven estimation approach allowed to extend model-based formal tools of analysis and synthesis to broader fields of application, for which first principles modeling is not suitable to obtain reliable models. Indeed, there are two ways of constructing mathematical models:

- *Mathematical modeling*. Analytic approach which employs prior knowledge and physical insight about the system to describe its dynamic behavior.
- *System identification*. Experimental approach, whereby a model is learnt from data collected by means of suitable experiments on the system.

In this thesis, the identification from experimental data of dynamical systems is considered with focus on *parametric methods*, [74]. These techniques rely on the a-priori assumption that the true model belongs to a desired *model class* $\mathcal{M}$, in contrast with *non-parametric methods* which try to estimate directly the impulse or frequency response of the system. In the parametric case, when the model structure is known, the identification problem consists essentially in estimating a finite number of unknown parameters. The parameter estimation task involves the minimization of a suitable cost function with respect to the parameter vector. For example, in the prediction error minimization (PEM) framework the one-step ahead prediction error is minimized. In that context, parameter estimates are guaranteed to be unbiased if the model structure exactly matches that of the underlying system. Moreover, for specific model classes

(*e.g.,* ARX models), algorithms of the Least Squares (LS) family can be employed for parameter estimation, which greatly simplifies the estimation task. If, on the other hand, the model structure is unknown, one has also to carry out the task of *model structure selection* (MSS). The aim of MSS is to find the form of the dependence between the data, within a family of functions which is usually parameterized by means of a finite-dimensional parameter vector. This is a combinatorial problem which consists in selecting, among all the model terms that could be included into the model, only those providing good accuracy in approximating the system behavior, but at the same time preserving model parsimony. Ultimately, MSS is one of the most difficult tasks in the identification problem in view of its combinatorial nature, and hence it has roused great interest in the scientific community. Most of the proposed model selection algorithms consist of a policy to explore the model structure space and compare different model structures in terms of their performance. The naive exhaustive approach of generating all possible structures and comparing their performance (which is, in fact, what one typically does with linear models) is hardly applicable in practice for complex systems, such as nonlinear and hybrid systems, due to the explosion of the search space - a problem often referred to as *curse of dimensionality* [2, 98]. Also, statistical indices such as the Akaike information criterion (AIC), the final prediction error (FPE), the minimum description length (MDL), the Bayesian information criterion (BIC), etc., which are used in the linear case to estimate the correct model structure balancing model accuracy and complexity, are unsuitable in these cases since there is no a simple relation between model accuracy and model size [92]. Finally, regularization approaches, such as the LASSO (Least Absolute Shrinkage Selection Operator), can help reducing the set of candidate model terms, but are not suitable for exact model selection [22, 53]. Therefore, not surprisingly, most efforts in the literature have been devoted to the development of heuristic search methods aimed at identifying a parsimonious model at an affordable cost. Several approaches have been proposed, ranging from greedy incremental strategies to evolutionary algorithms. This thesis investigates instead the use of randomized techniques based on a probabilistic reformulation of the selection problem. This approach operates by introducing (independent) Bernoulli or Categorical variables to represent the probability that model terms are present in the true model structure. The distributions of such stochastic variables are tuned based on the information gathered from a population of extracted models, according to a randomized approach. This general scheme is considered in this thesis, where the problem of MSS is investigated with respect to the identification of nonlinear systems via distributed computation, the identification of hybrid (nonlinear) systems, and also to the estimation of the process noise covariance matrix in state estimation problems in the Kalman filter setting. In the following, these three problems are briefly introduced, highlighting the relative challenges.

Distributed Nonlinear Model Identification - Several model classes have been proposed in the literature to represent nonlinear systems [21, 49, 53, 107]. In this thesis, we focus on the Nonlinear AutoRegressive with eXogenous input (NARX) class [70, 71], which naturally extends that of linear ARX models. Within this model class, the nonlinear mapping between data is often represented by means of a functional expansion involving lagged inputs and outputs. A popular choice for the form of the functional expansion is the polynomial one, which yields a *linear-in-the-parameters* model structure

that is particularly convenient for parameter estimation purposes. On the downside, the model complexity of the NARX class grows rapidly with the model order and nonlinearity degree, and this motivates the interest in the problem of MSS for such model class. The problem of identifying a model of a nonlinear system from input/output observations is typically formulated as an optimization problem over all available data that are collected by a central unit, in the same operating conditions. However, the massive diffusion of networked systems is changing this paradigm: data are collected separately by multiple agents and cannot be made available to some central unit due, *e.g.,* to band limitations or privacy constraints. Therefore, we address this novel set-up and consider the case in which multiple agents are cooperatively aiming at identifying a model for a system, by local computations based on private data sets. This problem is particularly challenging because the combinatorial nature of the MSS problem hampers the application of classical distributed schemes. Here, we propose a method that overcomes this limit by adopting a probabilistic reformulation of the model structure selection problem and seeking the consensus among agents on both the model structure and the parameter estimates at the same time.

Identification of Hybrid Nonlinear Systems - There are many physical processes whose behavior is characterized by different continuous dynamics (modes) among which the system can switch according to some discrete dynamics. For example, in electrical circuits, continuous phenomena, such as the charging of capacitors, are interrupted by switches or diodes. In a thermal control process a thermostat is used to control the temperature by switching on or off heating or cooling devices. In technological systems, such as a robotic system or a component mounter, the continuous dynamics, representing the behavior of the physical and mechanical part, interact with the discrete dynamics, that account for the software and logical behavior. More in general, complex systems exhibit different continuous dynamics as individual components are switched on or off. In such cases, a single dynamical model, even if nonlinear, is often not sufficient to capture the real dynamics of the system. Hybrid dynamical systems (HSs) [45, 76, 114], whose behavior can be described by the interaction of time- and event-driven dynamics, provide a unified framework for the representation of such cases. Most research regarding the identification of hybrid systems (HSI) has focused on switched affine (SA) and piecewise affine (PWA) models due to their universal approximation properties and their simple interpretation. Indeed, they provide the simplest extensions of continuous systems that can handle hybrid phenomena. In SA systems, the discrete state is an exogenous finite-valued input which determines the switching between different continuous affine dynamics, whereas in PWA systems the switching mechanism is determined by a polyhedral partition of the (continuous) state-input domain. The optimization problem induced by the identification task is of a mixed-integer type, since it involves the identification of discrete variables (representing the mapping of the samples to the modes and the model structure associated to each mode), as well as continuous ones (the parameters of the models describing the continuous dynamics associated to the various system conditions). Many approaches have been proposed over the last two decades for the case of *affine* dynamics, [44, 94]. Surprisingly fewer works have tackled the case of *nonlinear* continuous dynamics associated to the modes, in spite of its importance in modeling complex applications.

Indeed, if no *a priori* information on the number of modes is available, one can in principle identify an arbitrarily high number of local linear models (and switchings among them) in order to achieve a good model accuracy. However, this prevents the identification of the real dynamics of the hybrid system and hinders its physical interpretation. It also greatly aggravates the combinatorial complexity of the optimization problem, due to the increasing number of switchings. In this thesis, we consider the identification of switched nonlinear systems in input-output form, namely Switched Nonlinear ARX (SNARX), in the case of unknown model structures. We propose a black-box iterative identification method, where each iteration is characterized by two stages. In the first stage the identification problem is addressed assuming that mode switchings can occur only at predefined time instants, while in the second one the candidate mode switching locations are refined. This strategy allows to significantly reduce the combinatorial complexity of the problem, thus allowing an efficient solution of the optimization problem using a randomized method.

Structure Selection of the Process Noise Covariance Matrix in Kalman Filter Applications - A third problem in which MSS is crucial is the estimation of the process noise covariance matrix in state estimation problems in the Kalman filter setting, [33, 46, 59]. Kalman filtering for linear systems is known to provide the minimum variance estimation error, under the assumption that the model dynamics is known, [60]. While many system identification tools are available for computing the system matrices from experimental data, estimating the statistics of the output and process noises is still an open problem which significantly impacts the filter performance. In fact, although some methods based on maximum likelihood and correlation approaches have been proposed in the literature, it turns out that the existing techniques are either too computationally expensive or not accurate enough. Above all, in many papers the process and output covariance matrices are provided as a *prior knowledge* or their estimation is "declassified" into an *empirical tuning* problem in which diagonal parameterizations are often assumed, for simplicity. Our study indicates that this assumption does not always provide the best compromise between computational complexity and tracking accuracy. This evidence encouraged us to further investigate this problem leading to an algorithm for the selection of the structure of the process noise covariance matrix, which actually is an adaptation of the same optimization algorithm originally employed for MSS in nonlinear identification applications.

1.1 Thesis outline

The thesis is structured as follows. Chapter 2 gives an introduction to system identification, formalizes the identification problem for nonlinear systems and hybrid systems with emphasis on the problem of choosing a suitable model structure. It also presents the Kalman filtering problem for linear systems. Finally, it contains an overview of different approaches to the MSS problem. Chapter 3 describes the randomized model structure selection method which will be used to address the three problems investigated in this thesis. The latter are discussed in the next three Chapters: namely, Chapter 4 deals with the distributed nonlinear model identification, Chapter 5 with the identification of hybrid nonlinear systems, and Chapter 6 with the structure selection of the pro-

cess noise covariance matrix in Kalman filter applications, respectively. Lastly, some conclusions are drawn in Chapter 7, and guidelines for future research are suggested.

Note that part of the content of Chapters 3 and 5 has been published in:

F. Bianchi, M. Prandini, and L. Piroddi. A randomized approach to switched nonlinear systems identification. *18th IFAC Symposium on System Identification, SYSID 2018*, 2018

F. Bianchi, M. Prandini, and L. Piroddi. A randomized two-stage iterative method for switched nonlinear systems identification. *Nonlinear Analysis: Hybrid Systems*, 35:100818, 2020

The contribution in Chapter 4 appears in:

F. Bianchi, A. Falsone, M. Prandini, and L. Piroddi. Nonlinear system identification with model structure selection via distributed computation. *58th IEEE Conference on Decision and Control*, 2019

Finally, the content of Chapter 6 can be found in:

F. Bianchi, S. Formentin, and L. Piroddi. Process noise covariance estimation via stochastic approximation. *International Journal of Adaptive Control and Signal Processing*, 2019

F. Bianchi, S. Formentin, and L. Piroddi. Structure selection of noise covariance matrices for linear kalman filter design. *Accepted for publication in the proceedings of the 2020 European Control Conference*, 2020

1.2 Contributions

The main contributions of this thesis are:

- A procedure for the identification of nonlinear systems via distributed computation which jointly addresses the MSS problem and the parameter estimation. In particular, the cooperative solution of a MSS problem seems to be innovative w.r.t. the state of the art since only a few attempts to deal with this problem are documented in the literature.
- An iterative two-stage procedure for the identification of switched nonlinear systems. In general, only a few works have tackled the case of nonlinear continuous dynamics associated to the system modes. Furthermore, most of them are nonparametric, while in this thesis the identification has been addressed from a parametric point of view, including a MSS process in the identification procedure.
- An heuristic method for the structure selection of noise covariance matrices in Kalman filter applications, which contrasts with the common approach of choosing diagonal parameterizations. To our knowledge, this is the first time that the

covariance matrix structure selection problem is posed (and a solution is proposed for it).

CHAPTER 2

Preliminaries and problem formulation

THIS chapter introduces system identification with focus on the problem of model structure selection (MSS). The model classes of interest for this thesis are briefly reviewed. Emphasis is given in particular to the identification from experimental data of nonlinear systems and hybrid (nonlinear) systems. A section is also dedicated to the estimation of the process noise covariance matrix in the context of linear Kalman filtering. As will be discussed, these three problems all entail a MSS, which motivates the interest of this thesis for this specific task.

2.1 System identification

Inferring models from observed data is a fundamental task in science, [43, 54], and has been addressed in different application areas, [31, 75]. Particularly, in the systems and control area, the techniques answering to this problem are known under the collective term *system identification* [74, 108], in that they involve the estimation problem of a *dynamical system* based on data which have been observed from the system itself. Starting from the model it is then possible to develop formal tools of analysis and synthesis for the system, to perform complex tasks such as prediction, fault detection, control design. System identification is an experimental approach, in contrast with *mathematical modeling*, where prior knowledge and physical insight about the system are used to describe the dynamic behavior of a system. Specifically, system identification requires the knowledge of observed inputs $u(t)$ and outputs $y(t)$ taken from experiments performed on a dynamical system $\mathcal{S}$

$$\text{input data: } \{u(1), u(2), \dots, u(N)\} \tag{2.1}$$

$$\text{output data: } \{y(1), y(2), \dots, y(N)\} \tag{2.2}$$

and the objective is to look for a model $\mathcal{M}(\boldsymbol{\vartheta})$, where $\boldsymbol{\vartheta}$ is a finite-dimensional vector of unknown parameters, that can explain the relationship between input and output. When $\boldsymbol{\vartheta}$ spans a set of feasible values Θ, we obtain a parameterized set of models or a *model structure* $\mathcal{M}$, each different $\boldsymbol{\vartheta} \in \Theta$ corresponding to a different model. Generally speaking, the model structure is a parameterized mapping which projects past inputs and outputs $\mathcal{D}^{t-1}$ to the space of the model outputs:

$$\hat{y}(t|\boldsymbol{\vartheta}) = g\left(\mathcal{D}^{t-1}; \boldsymbol{\vartheta}\right), \tag{2.3}$$

where

$$\mathcal{D}^{t-1} = \{u(1), y(1), \ldots, u(t-1), y(t-1)\}. \tag{2.4}$$

Amongst all the possible parameterizations, we are looking for the one that provides the best approximation of $y(t)$ in terms of an identification criterion. The parameter estimation task deals with this approximation problem, minimizing with respect to $\boldsymbol{\vartheta}$ a *loss function*, defined as:

$$\mathcal{L}(y, \hat{y}) : \mathbb{R}^2 \to \mathbb{R}, \tag{2.5}$$

such that $\mathcal{L}(y, y) = 0$. For example, in the prediction error minimization (PEM) framework, $\mathcal{L}$ is constructed as a function of the one-step ahead prediction error (for dynamical systems) which is the difference between the system output y and the model prediction (2.3). Usually, $\mathcal{L}(y, \hat{y})$ is evaluated on an independent set of data, not employed for training purposes, to establish the robustness and generalization properties of the model. When a *testing data set* is not available, cross-validation techniques can be used, [62]. Also, statistical indices such as the final prediction error (FPE), the Akaike information criterion (AIC), and the Bayesian information criterion (BIC) can be used to adjust the computed training error according to the model complexity and the size of $\mathcal{D}$, providing a way to assess how the model would perform on unseen data.

As formulated before, the identification problem consists essentially in solving a continuous optimization problem to find the parameterization that minimizes the loss function $\mathcal{L}(y, \hat{y})$, *i.e.,* the *Parameter Estimation* (PE) task. More in general, however, one has to choose a suitable model structure within a *class of model structures*. This leads to solving also a discrete optimization problem to select the most appropriate model structure within the chosen set of candidate ones, which makes the problem a mixed-integer one. We refer to this second problem as *Model Structure Selection* (MSS), to distinguish it from PE. There are several factors that can drive the MSS problem, such as:

- *Flexibility*. The chosen model structure should allow to describe most of the system dynamics. This factor is influenced by both the number of free model terms and how they enter into the model.
- *Parsimony*. The model should include the smallest number of free model terms needed to adequately represent the true system.

While the MSS problem is trivially solved in linear problems by exhaustively calculating all alternatives and comparing them to find the best one, *e.g.,* using the previously mentioned indices FPE, AIC, and BIC, the same approach does not apply to complex model identification problems, due to the combinatorial explosion.

2.1.1 Identification of nonlinear systems

Several classes of model structure have been proposed in the literature to cover the many types of existing nonlinear systems [21]. In this thesis, we focus on the Nonlinear AutoRegressive with eXogenous input (NARX) class [70, 71].

The NARX model is defined as

$$\hat{y}(t|\boldsymbol{\vartheta}) = g(\boldsymbol{x}(t); \boldsymbol{\vartheta}), \tag{2.6}$$

where $\boldsymbol{x}(t)$ is a finite-dimensional vector

$$[y(t-1), \cdots, y(t-n_y), u(t-1), \cdots, u(t-n_u)] \in \mathbb{R}^{n_y+n_u}, \tag{2.7}$$

with n_y and n_u being suitable maximum lags. The nonlinear mapping g is then often represented by means of a nonlinear functional expansion:

$$g(\boldsymbol{x}(t); \boldsymbol{\vartheta}) = \boldsymbol{\varphi}(t)\boldsymbol{\vartheta} = \sum_{j=1}^{n} \vartheta_j \varphi_j(t), \tag{2.8}$$

where $\boldsymbol{\varphi}(t) = \varphi(\boldsymbol{x}(t))$ is a mapping that projects the observations onto a finite-dimensional space (the basis functions space), *i.e.,* $\boldsymbol{\varphi}(t) : \mathcal{X} \subseteq \mathbb{R}^{n_y+n_u} \rightarrow \mathcal{F} \subseteq \mathbb{R}^n$. This mapping is particularly convenient in that it results in a *linear-in-the-parameters* model and it configures a linear regression problem (all nonlinearities are confined in $\boldsymbol{\varphi}(t)$):

$$y(t) = \boldsymbol{\varphi}(t)\boldsymbol{\vartheta} + e(t), \tag{2.9}$$

where the additive term $e(t)$ is a zero-mean stochastic process independent of $u(t)$ and $y(t)$, that accounts for the unpredictable part of the system. As a consequence, the current output $y(t)$ is not an exact function of past data. This naturally leads to interpreting the available observations as being a finite length realization of a stochastic process. This stochastic nature is explicitly represented by $e(t)$. Within this framework, the parameter estimation problem can be solved by employing algorithms of the Least Squares (LS) family. Indeed, using a squared loss function:

$$\mathcal{L}(y, \hat{y}) = \frac{1}{N} \sum_{t=1}^{N} \left(y(t) - \boldsymbol{\varphi}(t)\boldsymbol{\vartheta}\right)^2, \tag{2.10}$$

the estimation problem can be formulated as the following optimization problem:

$$\min_{\boldsymbol{\vartheta}} \mathcal{L}(y, \hat{y}), \tag{2.11}$$

which yields the closed form solution:

$$\hat{\boldsymbol{\vartheta}} = \left(\boldsymbol{\Phi}^T\boldsymbol{\Phi}\right)^{-1}\boldsymbol{\Phi}^T\boldsymbol{y} \in \mathbb{R}^n, \tag{2.12}$$

where $\boldsymbol{\Phi} = [\boldsymbol{\varphi}(1)^T, \ldots, \boldsymbol{\varphi}(N)^T]^T \in \mathbb{R}^{N\times n}$ and $\boldsymbol{y} = [y(1), \ldots, y(N)]^T \in \mathbb{R}^N$, provided that the matrix $\boldsymbol{\Phi}^T\boldsymbol{\Phi}$ is positive definite

Remark 1. *The matrix $\mathbf{\Phi}^T\mathbf{\Phi}$ is by construction always symmetric semi-definite positive. When the data matrix $\mathbf{\Phi}$ is not full rank, $\mathbf{\Phi}^T\mathbf{\Phi}$ turns out to be singular, and therefore there does not exist a unique solution to the equation*

$$\left(\mathbf{\Phi}^T\mathbf{\Phi}\right)\boldsymbol{\vartheta} = \mathbf{\Phi}^T\boldsymbol{y}. \tag{2.13}$$

Note however, that if the identification experiment is well designed, e.g., *using a persistently exciting input signal, then $\mathbf{\Phi}$ will have full rank, [108].*

To analyze the statistical properties of the estimate (2.12), additional assumptions on the data are needed. Assume that data have been generated according to the model (2.9) and to the unknown parameter $\boldsymbol{\vartheta}^\circ$, and that $e(t)$ is a white noise process of zero mean and variance σ^2. It follows that

$$\text{var}\left(\hat{\boldsymbol{\vartheta}}\right) = \left(\mathbf{\Phi}^T\mathbf{\Phi}\right)^{-1}\sigma^2 \tag{2.14}$$

and

$$\hat{\boldsymbol{\vartheta}} \sim \mathcal{N}\left(\boldsymbol{\vartheta}^\circ, \left(\mathbf{\Phi}^T\mathbf{\Phi}\right)^{-1}\sigma^2\right). \tag{2.15}$$

Typically the variance σ^2 is unknown and one has to estimate it by

$$\hat{\sigma}^2 = \frac{1}{N-n}\sum_{t=1}^{N}\left(y(t) - \hat{y}(t)\right)^2. \tag{2.16}$$

It follows that

$$(N-n)\,\hat{\sigma}^2 \sim \sigma^2\chi^2_{N-n}, \tag{2.17}$$

where χ^2_{N-n} is a chi-squared distribution with $N-n$ degrees of freedom, and that $\hat{\boldsymbol{\vartheta}}$ and $\hat{\sigma}^2$ are statistically independent. These distributional properties of $\hat{\boldsymbol{\vartheta}}$ can be used to form a hypothesis test to evaluate the confidence intervals for the parameters ϑ_j°. For example, one can test the null hypothesis that $\vartheta_j^\circ = 0$, by analyzing the *Z-score* associated to $\hat{\vartheta}_j$:

$$z_j = \frac{\hat{\vartheta}_j}{\hat{\sigma}\sqrt{v_j}}, \tag{2.18}$$

where v_j denotes the jth diagonal element of $\mathbf{\Phi}^T\mathbf{\Phi}$, which follows a t distribution with $N-n$ degrees of freedom. A large absolute value of z_j will lead to reject the null hypothesis. On the contrary, if $|z_j|$ is less than $t_{\alpha/2,N-n}$, which is the critical value of a t distribution with $N-n$ degrees of freedom and confidence level α, there is no sufficient evidence in the data to reject the null hypothesis and hence the corresponding parameter ϑ_j° can be removed from the model.

There are several families of basis functions that can be used in (2.8) to approximate any continuous function on a compact domain to a given precision level, provided that the expansion is endowed with sufficient degrees of freedom (*i.e.*, n is large enough). Polynomials, splines, multi-layer perceptron neural networks (NN), radial basis function (RBF), wavelets are examples of such universal approximating functions. As argued in [21], a popular choice in nonlinear identification is the polynomial functional expansion which is a linear combination of all monomials of $\boldsymbol{x}(t)$ up to a given order

n_d, *i.e.,* the regressors $\varphi_j(t)$ are of the form $x_1^{k_1}x_2^{k_2}\cdots x_l^{k_l}$, where $l = |\boldsymbol{x}| = n_y + n_u$, with $\sum_{i=1}^{l} k_i \leq n_d$ and $k_i \geq 0$. This functional expansion extends gracefully from linear models and typically allows an easier model interpretation. Indeed, the terms in the model are often associated to physical aspects of the system and so the parameters can be interpreted as importance factors for the relative physical phenomena. Furthermore, nonlinear combinations of physical variables reveal the existence of specific nonlinear dependencies. Other functional expansions do not share this feature. For example, artificial NNs are very powerful function approximators, but do not provide transparent models.

Unfortunately, polynomial expansions suffer from the well known *curse of dimensionality*, in that the number of terms grows rapidly with the number of elementary arguments $(n_u + n_y)$ and the degree of the polynomial n_d. However, it is common experience that polynomial models with few terms can provide highly accurate and robust models. It is therefore crucial to identify and select the essential terms of a model, *i.e.,* select the model structure, which motivates the interest in MSS algorithms.

Let the structure of a NARX model be coded in a vector $\boldsymbol{s} \in \{0,1\}^n$, where $s_j = 1$ if the j-th regressor belongs to the model structure (and $s_j = 0$ otherwise). The nonlinear system identification within the polynomial NARX class can be stated as follows.

Problem 1. *Given a data set $\mathcal{D}^N = \{(y(t), u(t))\}_{t=1}^N$ and the model orders n_y and n_u, and the degree of nonlinearity n_d, estimate the model structure $\boldsymbol{s}$ (chosen among the non-redundant structures [1]) and the parameter vector $\boldsymbol{\vartheta}$, so as to*

$$\begin{aligned} &\textit{minimize} \ \ \frac{1}{N}\sum_{t=1}^{N}\left(y(t) - \boldsymbol{\varphi}(t)\boldsymbol{\vartheta}\right)^2, \\ &\textit{subject to } \vartheta_j = 0 \textit{ if } s_j = 0, j = 1, \ldots, n, \end{aligned} \tag{2.19}$$

where $n = \frac{(n_d+n_y+n_u)!}{n_d!(n_y+n_u)!}$.

Problem 2.19 typically involves all data at once, assuming that data have been collected by a single entity. However, when data are collected separately by multiple entities and cannot be made available to a central unit, due to *e.g.,* privacy or communication constraints, the problem of identifying the same model based on separately available data sets arises. If the entities collecting data have computing and communication capabilities, one can formulate the problem in a distributed computation framework, where a network of agents are cooperatively solving the identification problem by local optimization. However, the problem of identifying the structure and parameters of the system has a mixed discrete and continuous nature, which hampers the application of classical distributed schemes, such as those based on the subgradient, [87], and on proximal minimization, [79]. In Chapter 4, we address this issue by resorting to a distributed scheme which aims at reaching a common value for both the model structure and the parameter estimates within the NARX modeling framework, where each agent i has its own data set and its own cost function $\mathcal{L}_i : \mathbb{R}^n \to \mathbb{R}$ to assess the quality of the model in terms of its parameterization $\boldsymbol{\vartheta} \in \mathbb{R}^n$ (which also encompasses the model

[1] Regressor redundancy can be tackled *e.g.* by introducing a regularization term, or by applying an *a posteriori* t-test on the estimated parameter vector to detect terms that are statistically indistinguishable from 0

structure, in that only the terms included in the model have nonzero parameters). The K agents aim at reaching consensus on a common value for $\boldsymbol{\vartheta}$ that optimizes $\sum_{i=1}^{K} \mathcal{L}_i(\boldsymbol{\vartheta})$, but without sharing their local data sets and costs.

2.1.2 The bias-variance dilemma

The Gauss-Markov theorem states that the LS estimator $\hat{\boldsymbol{\vartheta}}$ in (2.12) has minimum variance, among all the unbiased estimators, assuming that the model (2.9) is correct. However, when the model structure is unknown, one is confronted with the so called *bias-variance dilemma*.

Suppose we create several models starting each time from a new set of collected data. Due to the randomness in the underlying data sets, the resulting models will have different prediction performance. The bias measures how differ in general the model predictions from the correct value, while the variance describes how much the predictions vary between different realizations of the model. Essentially, dealing with bias and variance is really about dealing with over- and under-fitting. High bias can cause a model to miss the relevant relations between the inputs and the target outputs (under-fitting). High variance can cause over-fitting, *i.e.,* the model is so accurate that it fits the data including the noise effects, rather than capturing the real system dynamics. Hence, there is a trade-off between bias and variance, since in general the bias is reduced and the variance is increased as the model complexity is increased. Within the polynomial NARX model class, assuming *a priori* information about the model orders and the degree of nonlinearity, this trade-off deals with the problem of choosing which model terms to include in the model.

In principle, one could include all the available regression terms φ_j into the model and estimate the parameter vector $\boldsymbol{\vartheta}$ in a classical LS approach. It can be proved that in this case the variance of $\hat{y}$ for a given vector $\boldsymbol{\varphi}$, is:

$$\text{var}\left(\boldsymbol{\varphi}\hat{\boldsymbol{\vartheta}}\right) = \sigma^2 \left(\boldsymbol{\varphi}\boldsymbol{R}^{-1}\right)\left(\boldsymbol{\varphi}\boldsymbol{R}^{-1}\right)^T, \tag{2.20}$$

where $\boldsymbol{R} \in \mathbb{R}^{n\times n}$ is the upper-triangular matrix coming from the Cholesky decomposition of the matrix $\boldsymbol{\Phi}^T\boldsymbol{\Phi} \in \mathbb{R}^{n\times n}$, *i.e.,* $\boldsymbol{\Phi}^T\boldsymbol{\Phi} = \boldsymbol{R}^T\boldsymbol{R}$.

Property 1. *According to* (2.20)*, the variance of the predicted value $\hat{y}$ is the sum of squares of the elements of the n-dimensional vector $\boldsymbol{\varphi}\boldsymbol{R}^{-1}$.*

Now let us consider the case in which one wants to predict y using only the first $k < n$ terms φ_i. Let us write

$$\boldsymbol{\Phi} = \left(\boldsymbol{\Phi}_A, \boldsymbol{\Phi}_B\right), \tag{2.21}$$

where $\boldsymbol{\Phi}_A$ consists of the first k columns of $\boldsymbol{\Phi}$, and $\boldsymbol{\Phi}_B$ contains the remaining $(n-k)$ columns. It turns out that:

$$\text{var}\left(\boldsymbol{\varphi}\hat{\boldsymbol{\vartheta}}\right) \geq \text{var}\left(\boldsymbol{\varphi}_A\hat{\boldsymbol{\vartheta}}_A\right), \tag{2.22}$$

where

$$\text{var}\left(\boldsymbol{\varphi}_A\hat{\boldsymbol{\vartheta}}_A\right) = \sigma^2 \left(\boldsymbol{\varphi}_A\boldsymbol{R}_A^{-1}\right)\left(\boldsymbol{\varphi}_A\boldsymbol{R}_A^{-1}\right)^T, \tag{2.23}$$

with $\boldsymbol{\varphi}_A$ denoting the first k elements of vector $\boldsymbol{\varphi}$, $\hat{\boldsymbol{\vartheta}}_A$ being the corresponding LS estimate, and $\boldsymbol{R}_A$ being composed by the first k rows and columns of matrix $\boldsymbol{R}$. The result in (2.22) is due to Property 1.

Assuming that the true model is in the form (2.9), then

$$\begin{aligned}\mathbb{E}\left[\hat{\boldsymbol{\vartheta}}_A\right] &= \left(\boldsymbol{\Phi}_A^T\boldsymbol{\Phi}_A\right)^{-1}\boldsymbol{\Phi}_A^T\boldsymbol{\Phi}\boldsymbol{\vartheta} \\ &= \left(\boldsymbol{\Phi}_A^T\boldsymbol{\Phi}_A\right)^{-1}\boldsymbol{\Phi}_A^T\left(\boldsymbol{\Phi}_A,\boldsymbol{\Phi}_B\right)\boldsymbol{\vartheta} \\ &= \left(\boldsymbol{\Phi}_A^T\boldsymbol{\Phi}_A\right)^{-1}\left(\boldsymbol{\Phi}_A^T\boldsymbol{\Phi}_A,\boldsymbol{\Phi}_A^T\boldsymbol{\Phi}_B\right)\boldsymbol{\vartheta} \\ &= \boldsymbol{\vartheta}_A + \left(\boldsymbol{\Phi}_A^T\boldsymbol{\Phi}_A\right)^{-1}\left(\boldsymbol{\Phi}_A^T\boldsymbol{\Phi}_B\right)\boldsymbol{\vartheta}_B\end{aligned} \tag{2.24}$$

where $\boldsymbol{\vartheta}_A$ and $\boldsymbol{\vartheta}_B$ are respectively the first k and the last $(n-k)$ elements of $\boldsymbol{\vartheta}$. The second term in the R.H.S. of (2.24) represents the bias introduced in the first k least-squares coefficients due to the omission of the remaining $(n-k)$ coefficients. The bias in estimating y given $\boldsymbol{\varphi}$ and $\boldsymbol{\vartheta}_A$ is hence:

$$\boldsymbol{\varphi}\boldsymbol{\vartheta} - \mathbb{E}\left[\boldsymbol{\varphi}_A\hat{\boldsymbol{\vartheta}}_A\right] = \left[\boldsymbol{\varphi}_B - \boldsymbol{\varphi}_A\left(\boldsymbol{\Phi}_A^T\boldsymbol{\Phi}_A\right)^{-1}\left(\boldsymbol{\Phi}_A^T\boldsymbol{\Phi}_B\right)\right]\boldsymbol{\vartheta}_B \tag{2.25}$$

In general, as more terms are added to a model, the bias is reduced, at the cost of an increasing variance. However, if a term has no predictive value, then adding that term merely increases the variance. Conversely, there are situations in which one can accept an increase in bias if it helps in reducing significantly the estimator variance, as will be discussed in Section 2.3.

2.1.3 Identification of hybrid systems

Hybrid systems (HS) are dynamical systems whose behavior can be described by the interaction of several time-driven continuous dynamics indexed by event-driven discrete dynamics (discrete state), [45, 76, 114]. HSs provide a unified framework for the representation of technological systems whose physical and mechanical part is described by continuous models such as differential or difference equations describe the, while the software and logical behavior is well modeled by finite-state machines or Petri nets. Also many real physical processes exhibiting both fast and slow changing behaviors can be described by HS models. Finally, a nonlinar dynamical system can be approximated by switching among various linear models.

Most research regarding the identification of hybrid systems (HSI) has focused on switched affine (SA) and piecewise affine (PWA) models due to their universal approximation properties and their simple interpretation. Indeed, they provide the simplest extensions of continuous systems that can handle hybrid phenomena. The difference between them relies on how the switching mechanism is implemented. In SA systems, the switching between different continuous affine dynamics is governed by a finite-valued input which represents the discrete state, whereas in PWA systems the switching mechanism is determined by a polyhedral partition of the (continuous) state-input domain. The input-output counterparts of these system classes are Switched ARX (SARX) and PieceWise affine ARX (PWARX), respectively. The optimization problem induced by the identification task is of the mixed-integer type, since it involves the identification of discrete variables (representing the mapping of the samples to the modes and the model structure associated to each mode), as well as continuous ones (the parameters of the models describing the continuous dynamics associated to the various system conditions).

In Chapter 5, we investigate the identification of Switched Nonlinear ARX (SNARX) models, defined as a collection of NARX systems (2.9) indexed by a finite-valued switching signal $\boldsymbol{\sigma}$. Given a regression vector $\boldsymbol{\varphi}$ of size n, common to all the continuous modes, the output predictor of a SNARX model is defined as

$$\hat{y}(t|\boldsymbol{\vartheta}) = \boldsymbol{\varphi}(t)\boldsymbol{\vartheta}^{(\sigma(t))}, \tag{2.26}$$

where $\sigma(t) \in \{1, \dots, N_M\}$ is the value taken by the switching signal at time t, that defines which mode is active at that time instant, N_M is the number of modes, and $\boldsymbol{\vartheta}^{(i)}$ is the parameter vector defining the dynamics of the i-th NARX mode. According to the coding of the NARX model structure introduced in Section 2.1.1, the overall SNARX model structure can be encoded in a $n \times N_M$ matrix $S = [\boldsymbol{s}^{(1)}, \dots, \boldsymbol{s}^{(N_M)}] \in \mathcal{S} = \{0, 1\}^{n \times N_M}$, which is the collection of the structures of the NARX models that are associated with its N_M modes.

Given a data set of time-ordered and consecutive input-output samples of a SNARX, a finite-valued switching signal assigns each sample to a specific mode. For the purpose of the SNARX model identification, the SNARX model structure needs to be extended to include the mode switching signal $\boldsymbol{\sigma} = [\sigma(1), \dots, \sigma(N)] \in \Sigma = \{1, \dots, N_M\}^N$, such that $\sigma(t) = i$ if sample t is attributed to mode i. Notice that the identification of $\sigma(t)$ amounts to segmenting the data in consecutive portions, attributing each subperiod to the appropriate mode. A SNARX model structure is thus expressed by a pair $\lambda = (\boldsymbol{\sigma}, S)$ taking values in $\Lambda = \Sigma \times \mathcal{S}$. The quality of a SNARX model with structure λ is given by the value of the loss function corresponding to its optimal parameterization:

$$\begin{aligned} \mathcal{L}(\lambda) = \min_{\{\boldsymbol{\vartheta}^{(i)}\}_{i=1}^{N_M}} & \frac{1}{N} \sum_{t=1}^{N} \sum_{i=1}^{N_M} \beta_t^{(i)} \cdot \left(y(t) - \boldsymbol{\varphi}(t)\boldsymbol{\vartheta}^{(i)} \right)^2, \\ & \text{subject to } \vartheta_j^{(i)} = 0 \text{ if } s_j^{(i)} = 0, j = 1, \dots, n, i = 1, \dots, N_M, \end{aligned} \tag{2.27}$$

where $\beta_t^{(i)}$ is a binary variable encoding the sample-mode mapping provided by $\boldsymbol{\sigma}$:

$$\sigma(t) = i \iff \beta_t^{(i)} = 1. \tag{2.28}$$

Consistently with most of the literature on switched system identification, we here address the case where Assumption 1 holds.

Assumption 1. *The number of modes N_M is known.*

Under Assumption 1, the SNARX identification problem can be stated as follows.

Problem 2. *Given a data set of time-ordered and consecutive input-output samples $\mathcal{D} = \{(y(t), u(t))\}_{t=1}^{N}$, estimate the model structures $\boldsymbol{s}^{(i)}$ (chosen among the non-redundant structures [2]) and parameterizations $\boldsymbol{\vartheta}^{(i)}$, $i = 1, \dots, N_M$, of the mode dynamics, as well as the switching signal $\boldsymbol{\sigma}$, so as to minimize the fit criterion* (2.27).

If there exists only one λ minimizing (2.27), this can be written as

$$\lambda^\star = (\boldsymbol{\sigma}^\star, S^\star) = \arg\min_{\lambda \in \Lambda} \mathcal{L}(\lambda). \tag{2.29}$$

[2] See Note 1 on redundancy of the parameterization.

The optimization problem (2.29) is a mixed integer program, which is typically computationally intractable due to its combinatorial complexity. Indeed, the SNARX structure λ involves $N \times N_M$ binary variables for $\boldsymbol{\sigma}$, plus $n \times N_M$ for S. Typically, N is the factor most affecting the combinatorial complexity of the problem, since $N \gg n, N_M$. As a consequence, the sample-mode mapping is the most critical aspect of the problem, since switchings can occur at arbitrary times. However, denoting by $\mathcal{T}_s^\circ \subseteq \{1, \dots, N\}$ the set of switching time instants in the observed data, it is typically true that $|\mathcal{T}_s^\circ| \ll N$.

2.2 Kalman filtering

Model structure selection is crucial also in the estimation of the process noise covariance matrix in state estimation problems in the Kalman filter setting, [33,46,59], as will be discussed in the following. Kalman filtering is probably the most widespread tool for designing virtual sensors and state estimators in dynamical systems [59, 60, 99]. Within the linear framework, it can be shown that the Kalman filter provides the minimum variance state estimator when all noise acting on the system are Gaussian, under the assumption that the employed model accurately describes the real system dynamics. The model is characterized as a stochastic dynamical system in the state-space form, that includes both a model of the process relating the inputs to the outputs and a noise model accounting for both noise effects and unmodeled dynamics. Such noise model is a crucial complement to the process model, which is typically the result of some approximations. This is even more true in practical applications where the physics underlying the systems is unknown or too complex to model from first principles. System identification plays a key role in such conditions and has been extensively applied to compute the system matrices from data, resulting in several well-established techniques. On the other hand, a relatively smaller effort has been devoted to characterize the noise model, which is a key factor affecting the quality of the state estimator. In many papers the process noise covariance matrix $\boldsymbol{Q}$ and the measurement noise covariance matrix $\boldsymbol{R}$ are provided as *prior knowledge* or their estimation is "declassified" into an *empirical tuning* problem [41]. Recently, the joint estimation of the state and the covariance matrices (CMs) from data has been more thoroughly investigated and several algorithms have been proposed. Some of these techniques are formulated within an *adaptive control* framework, where a feedback mechanism is used to update the CMs based on the quality of state estimation. A common feature of all the existing algorithms is that the CMs are assumed to be *retrievable* from data, *i.e.*, that the data are informative enough to estimate them and that the matrices are correctly parameterized. However, as observed in [83], tuning the CMs may require more degrees of freedom than tuning the optimal gain of a Kalman filter, which poses an identifiability issue on the noise CMs. Indeed, these matrices cannot be estimated univocally in such conditions. This problem can be dealt with in two ways. A first strategy consists in estimating directly the optimal gain through an iterative process, bypassing completely the estimation of $\boldsymbol{Q}$ and $\boldsymbol{R}$. However, this approach is suitable if only the state estimation is of interest. A second strategy is that of simplifying the structure of $\boldsymbol{Q}$ and $\boldsymbol{R}$ to match the degrees of freedom of the Kalman gain matrix. For example, a diagonal structure is often adopted for these matrices

(thereby assuming incorrelation of the individual noise components), which greatly reduces the number of free elements. While this often works out satisfactorily, it is not always the most appropriate choice, leading to an unnecessary loss of filtering accuracy.

In Chapter 6, we analyze in more detail the introduced structural simplification approach, showing that the diagonal parameterization of the CMs does not always provide the best compromise between computational complexity and tracking accuracy, and accordingly we propose an algorithm to suitably select the structure of $\boldsymbol{Q}$ (the $\boldsymbol{R}$ can be directly deduced from the sensors characteristics).

2.3 Model structure selection

The identification problems defined so far all entail a MSS, *i.e.,* the problem of finding the optimal subset of model terms among a given set of candidates, according to some performance criterion. The latter is in general related to the accuracy of the model [84]. As suggested by the bias-variance analysis carried out in Section 2.1.2, MSS is particularly challenging since there is not an analytical way to find the optimal model complexity. Instead, we must use a measure of the prediction error, and explore differing levels of model complexity and then choose the most appropriate complexity level. In doing so, one has to consider that a prediction error based measure, *e.g.,* the squared loss function in (2.10), is computed using the training data that were used to fit the model, and so it will decrease as model complexity increases, but the same measure computed on unseen data may not. This motivates the usage of a test data set, not used for training purposes, to validate the estimated model, or the use of cross-validation techniques when test data are not available. These techniques allow to estimate the prediction error on unseen data. Alternatively, the loss function can be modified to include a regularization term penalizing the model complexity.

In the following, the main approaches for solving MSS problems are briefly introduced, highlighting their pros and cons, and the possible relations between them.

2.3.1 Best subset selection methods

The most intuitive approach for solving the MSS, is certainly that of considering exhaustively all possible model structures, estimate the relative model parameterizations, and compare them in terms of the criterion of choice. That is the case of the Best Subset Selection (BSS) approach. Specifically, with reference to problem 2.19, an LS regression model is fitted on the training data for each possible combination of the n regressors, and one then looks at all of the resulting $2^n - 1$ models, with the aim of finding the *best* one. In practice, for each $0 < k \leq n$, one estimates all the models with exactly $(n - k)$ non-zero LS coefficients ϑ_j, and one picks the *best* among all these $\binom{n}{k}$ models. Then, among all the resulting n best models, a single model is selected by evaluating them on a testing data set $\mathcal{D}_{test}$ or according to some criterion, *e.g.,* AIC, BIC. Note that a price is paid in terms of variance for selecting the best subset of each size, thus resulting in possible over-fitting.

Let us introduce a mathematical formulation of the BSS scheme which will be used to compare this approach with other approaches. Specifically, at each step k, the fol-

lowing optimization problem has to be solved:

$$\min_{\boldsymbol{\vartheta}} \sum_{t=1}^{N} \left(y(t) - \sum_{j=1}^{n} \vartheta_j \varphi_j(t) \right)^2 \tag{2.30}$$
$$\text{subject to } \sum_{j=1}^{n} \mathbb{1}\left(\vartheta_j \neq 0\right) \leq k$$

where the indicator function

$$\mathbb{1}\left(\vartheta_j \neq 0\right) = \begin{cases} 1, & \text{if } \vartheta_j \neq 0 \\ 0, & \text{otherwise} \end{cases}. \tag{2.31}$$

The optimal structure $\boldsymbol{s}^\star$, can be retrieved directly from the estimated optimal parameterization $\boldsymbol{\vartheta}^\star$ solving problem (2.30) as:

$$s_j^\star = \begin{cases} 0, & \text{if and only if } \vartheta_j^\star = 0, \\ 1, & \text{otherwise.} \end{cases} \tag{2.32}$$

While this approach is conceptually very simple, it is computationally infeasible for large values of n, since it is an NP-hard problem, [85]. Also, note that the resulting optimization problem (2.30) is not convex due to the presence of the indicator function. This motivates the interest in computational tractable alternatives.

2.3.2 Stepwise methods

The methods belonging to this class, try to mitigate the computational issue of the BSS approach, by reducing the number of candidate models to evaluate and compare. Specifically, in the forward stepwise method, the model is built incrementally, starting from an empty model and adding one model term at a time. At each step the term that gives the greatest additional improvement to the fit is added to the model, thus involving the fitting of $(n-k)$ models at the kth iteration. Note that in practice, the full model is never computed, but one stops when an *a-priori* defined maximum model size is reached, or a minimum performance requirement for the model is met. The backward stepwise method adopts instead a decremental approach starting from a full initial model.

Unlike the BSS approach, in this case only $\frac{n(n+1)}{2}$ models have to be fitted at most, but there is no guarantee to find the true best model. Note that stepwise methods implement a more constrained search than the BSS approach, and will have lower variance (but perhaps more bias), thus mitigating also the over-fitting issue.

The forward-regression orthogonal estimator (FROE) [20] represents a milestone in the research on model structure selection algorithms, and several variants of this method have been proposed in the literature [48, 72, 78, 98, 116]. The FROE adopts an incremental greedy scheme where the regressors are rated by means of the error reduction ratio (ERR) criterion. The FROE also uses a smart scheme based on orthogonal least squares (OLS) to decouple the estimation of the parameters associated to additional terms from that of the parameters related to terms already included in the model. The

drawbacks of the FROE have been extensively reviewed in the literature (see, *e.g.*, the discussion in [98]). Most of these are related with the fact that the ERR provides a local estimation of the importance of a specific regressor, which appears to vary significantly depending on the considered model structure.

2.3.3 Shrinkage methods

The idea behind these methods is to estimate a full model by exploiting all the available n model terms, but constraining the model coefficients. In particular, they are based on a convex penalized relaxation of problem (2.30).

Ridge regression

The following problem is addressed:

$$\begin{aligned} &\min_{\boldsymbol{\vartheta}} \sum_{t=1}^{N} \left(y(t) - \sum_{j=1}^{n} \vartheta_j \varphi_j(t) \right)^2 \\ &\text{subject to } \sum_{j=1}^{n} \vartheta_j^2 \leq k \end{aligned} \tag{2.33}$$

or equivalently

$$\min_{\boldsymbol{\vartheta}} \sum_{t=1}^{N} \left(y(t) - \sum_{j=1}^{n} \vartheta_j \varphi_j(t) \right)^2 + \lambda \sum_{j=1}^{n} \vartheta_j^2, \tag{2.34}$$

where $\lambda > 0$ is a parameter governing the trade-off between model accuracy and model complexity. A bias-variance perspective suggests that as λ increases, the estimator variance will decrease while the bias will increase. Recalling that the LS estimator (which corresponds to the case $\lambda = 0$) is unbiased, we are accepting to introduce a bias in the estimate, if this helps in significantly decreasing the variance, [52].

Given a value of λ, this method requires the estimation of a single model which will be computed in closed form as

$$\hat{\boldsymbol{\vartheta}}^{ridge} = \left(\boldsymbol{\Phi}^T \boldsymbol{\Phi} + \lambda \boldsymbol{I}_n \right)^{-1} \boldsymbol{\Phi}^T \boldsymbol{y}, \tag{2.35}$$

where $\boldsymbol{\Phi} = [\boldsymbol{\varphi}(1)^T, \ldots, \boldsymbol{\varphi}(N)^T]^T \in \mathbb{R}^{N \times n}$ and $\boldsymbol{I}_n$ denotes the identity matrix of order n. However, unfortunately, this will always lead to a full model, *i.e.,* no coefficients will be exactly zero. For a better understanding of this approach, especially in comparison with the classical LS estimate, consider the singular value decomposition (SVD) of the regression matrix $\boldsymbol{\Phi}$:

$$\boldsymbol{\Phi} = \boldsymbol{U} \boldsymbol{D} \boldsymbol{V}^T, \tag{2.36}$$

where $\boldsymbol{U} \in \mathbb{R}^{N \times n}$ and $\boldsymbol{V} \in \mathbb{R}^{n \times n}$ are orthogonal matrices, and $\boldsymbol{D} \in \mathbb{R}^{n \times n}$ is a diagonal matrix whose diagonal elements are the singular values of $\boldsymbol{\Phi}$. It results that:

$$\begin{aligned} \boldsymbol{\Phi} \hat{\boldsymbol{\vartheta}}^{LS} &= \boldsymbol{\Phi} \left(\boldsymbol{\Phi}^T \boldsymbol{\Phi} \right)^{-1} \boldsymbol{\Phi}^T \boldsymbol{y} = \boldsymbol{U} \boldsymbol{U}^T \boldsymbol{y} \\ &= \sum_{j=1}^{n} \boldsymbol{u}_j \boldsymbol{u}_j^T \boldsymbol{y}, \end{aligned} \tag{2.37}$$

and

$$\begin{aligned}\boldsymbol{\Phi}\hat{\boldsymbol{\vartheta}}^{ridge} &= \boldsymbol{\Phi}\left(\boldsymbol{\Phi}^T\boldsymbol{\Phi} + \lambda I_n\right)^{-1}\boldsymbol{\Phi}^T\boldsymbol{y} \\ &= \boldsymbol{U}\boldsymbol{D}\left(\boldsymbol{D}^2 + \lambda I_n\right)^{-1}\boldsymbol{D}\boldsymbol{U}^T\boldsymbol{y} \\ &= \sum_{j=1}^{n}\boldsymbol{u}_j\frac{d_j^2}{d_j^2+\lambda}\boldsymbol{u}_j^T\boldsymbol{y}.\end{aligned} \tag{2.38}$$

Therefore, unlike LS regression, the ridge regression rescales the coordinates of $\boldsymbol{y}$ with respect to the orthonormal basis $\boldsymbol{U}$ of $\boldsymbol{\Phi}$, by a factor which is proportional to the singular values, *i.e.,* the smaller d_j^2, the bigger the shrinking.

LASSO

Another convex penalized relaxation of the BSS (2.30) is represented by the LASSO estimate [29, 113], defined by:

$$\begin{aligned}&\min_{\boldsymbol{\vartheta}}\sum_{t=1}^{N}\left(y(t)-\sum_{j=1}^{n}\vartheta_j\varphi_j(t)\right)^2 \\ &\text{subject to }\sum_{j=1}^{n}|\vartheta_j|\le k\end{aligned} \tag{2.39}$$

or equivalently

$$\min_{\boldsymbol{\vartheta}}\sum_{t=1}^{N}\left(y(t)-\sum_{j=1}^{n}\vartheta_j\varphi_j(t)\right)^2+\lambda\sum_{j=1}^{n}|\vartheta_j|. \tag{2.40}$$

Unlike ridge regression, LASSO has not a closed form solution, and computing the LASSO solution amounts to solving a quadratic programming problem (except if the matrix $\boldsymbol{\Phi}$ is orthonormal), which requires the use of efficient solvers. Also, because of the different nature of the constraint, LASSO translates each coefficient by a constant factor λ and some of them are truncated at zero. Under various conditions on the regression matrix $\boldsymbol{\Phi}$ and N, n, $\boldsymbol{\vartheta}$ it can be shown that LASSO leads to a sparse model with good predictive performance, [43, 105], thus actually solving the MSS problem in a continuous domain. On the contrary, a critical aspect is the selection of the optimal λ value, which is usually done by cross-validation, [22]. Also, by the nature of the ℓ_1 norm, LASSO may lead to biased estimates of the regression coefficients, [13].

2.3.4 Dimension reduction methods

All the methods discussed so far, aim to reduce the estimator variance selecting a suitable subset of all the model terms φ_j, considering them in their original form. Instead, the dimension reduction methods operate with $p < n$ linear combinations of the φ_j's. They first transform the model terms and then fit a LS model using the transformed terms.

Let $z_1, z_2, \ldots, z_p$ being the $p < n$ linear combinations of the φ_j's computed as:

$$z_i = \sum_{j=1}^{n}\phi_{ji}\varphi_j, \tag{2.41}$$

where $\phi_{1i}, \phi_{2i}, \ldots, \phi_{ni}$, $i = 1, 2, \ldots, p$, are some constants. It can be shown that:

$$\sum_{i=1}^{p} \beta_i z_i = \sum_{j=1}^{n} \vartheta_j \varphi_j, \tag{2.42}$$

where

$$\vartheta_j = \sum_{i=1}^{p} \beta_i \phi_{ji}. \tag{2.43}$$

As in ridge regression and the LASSO, the dimension reduction serves to constrain the estimated ϑ_j coefficients, since now they must take the form (2.43).

The Principal Component Regression (PCR) [55, 56] belongs to this family. It exploits the Principal Component Analysis (PCA), taking the first p directions along which the φ_j's vary the most as z_i's, under the assumption that y also varies most in these directions. Apparently, the PCR does not perform a real MSS since all the original model terms are used in computing the z_i terms, and hence it works similarly to ridge regression. Nevertheless, being $p < n$, the computational complexity is reduced.

2.3.5 Evolutionary methods

Evolutionary algorithms (EAs) are search and optimisation methods, inspired by the principles of natural selection and population genetics. They allow a flexible representation of the decision variables, thus allowing their application to decision problems characterized by a large search space, leading to their diffusion also in the systems and control community, [39]. The term EAs denotes a broad family of methods, such as genetic algorithms (GAs) and genetic programming (GP), and particle swarm based optimization methods (PSO). The main idea behind these algorithms is to encode each solution of the problem, which can then be easily manipulated by means of operators, some of which are based on a designed cost function. They are population-based iterative algorithms, in which an initial set of candidate solutions is manipulated in order to effectively explore the entire solution space, with the aim of reaching the optimal solution.

Apparently, EAs can deal with the MSS problem, where the presence or absence of a model term is easily encoded as a binary decision variable, as reported in [40, 78, 103, 115, 119, 120]. Consider, *e.g.,* the case of a GA applied to the MSS problem.

A typical GA requires a genetic representation of the solution domain and a fitness function to evaluate the solution domain. A standard representation of each candidate solution is a binary string. Assuming for example to have a candidate regressor set composed by 10 terms, *i.e.,* $\boldsymbol{\varphi} = [\varphi_1, \varphi_2, \ldots, \varphi_{10}]$. The model structure relative to the model $y(t) = \vartheta_1\varphi_1(t) + \vartheta_3\varphi_3(t) + \vartheta_7\varphi_7(t) + \vartheta_{10}\varphi_{10}(t)$ can be represented by the following binary string

$$1010001001 \tag{2.44}$$

New candidate model structures are obtained by means of genetic operators:

- *Crossover*: two strings are randomly selected from a previously defined pool using a given criterion (*tournament selection*, *roulette wheel selection* and others). The bit position from which the two strings will be splitted is randomly selected and the obtained substrings finally combined. For example, consider the following

strings: i)1010001001 and ii) 1110101011. Assuming to split from the fifth bit, the two resulting strings will be: i)1010001001 and ii)1110101001.

- *Mutation*: randomly flip a bit.

The fitness function assumes a key role in the selection phase: individual solutions are selected through a fitness-based process, where solutions with high fit value are typically more likely to be selected. In other words, the idea is to combine good solutions hoping to find a better one. In this way, the population will move toward better solutions. GAs are simple to implement, but there are some complexity issues. In particular, how model portions that have evolved to represent good solutions can be protected from further destructive mutation or crossover operations. So, there is a trade-off between exploration of the search space and exploitation of the current good results.

2.3.6 Bayesian methods

The methods belonging to this class address the MSS problem from a statistical perspective, starting from the specification of a probability model for the random variable $\boldsymbol{Y}$:

$$\boldsymbol{Y} \sim p\left(\boldsymbol{y}|\boldsymbol{\vartheta}\right), \tag{2.45}$$

where $\boldsymbol{\vartheta}$ is the parameter vector, and p is a probability density function. $p\left(\boldsymbol{y}|\boldsymbol{\vartheta}\right)$ is the likelihood of the value $\boldsymbol{y}$ for variable $\boldsymbol{Y}$ as a function of the parameters ϑ_j, $j = 1, 2, \ldots, n$. Accordingly, the model identification problem can be stated as that of finding the $\boldsymbol{\vartheta}$ value which maximizes the likelihood (or functions based on it), given the observed $\boldsymbol{y}$ value. If one specifies the prior probability $p(\boldsymbol{\vartheta})$, the joint probability of $\boldsymbol{y}$ and $\boldsymbol{\vartheta}$ is:

$$p(\boldsymbol{y}, \boldsymbol{\vartheta}) = p(\boldsymbol{y}|\boldsymbol{\vartheta})p(\boldsymbol{\vartheta}), \tag{2.46}$$

and according to the Bayes' rule, one obtains the posterior probability for the parameters given the data as:

$$\begin{aligned} p(\boldsymbol{\vartheta}|\boldsymbol{y}) &= \frac{p(\boldsymbol{y}, \boldsymbol{\vartheta})}{p(\boldsymbol{y})} \\ &= \frac{p(\boldsymbol{y}|\boldsymbol{\vartheta})p(\boldsymbol{\vartheta})}{p(\boldsymbol{y})}, \end{aligned} \tag{2.47}$$

where $p(\boldsymbol{y}) = \int p(\boldsymbol{y}|\boldsymbol{\vartheta})p(\boldsymbol{\vartheta})d\boldsymbol{\vartheta}$. Therefore, $p(\boldsymbol{\vartheta}|\boldsymbol{y}) \propto p(\boldsymbol{y}|\boldsymbol{\vartheta})p(\boldsymbol{\vartheta})$, but the normalization factor $p(\boldsymbol{y})$ cannot be easily evaluated due to the integral. To fill this gap, several sampling methods have been proposed to numerically compute posterior distributions, [112].

Markov chain Monte Carlo methods

One popular sampling method family is the *Markov chain Monte Carlo* (MCMC). The idea behind these methods is to set up a Markov chain whose stationary distribution is $p(\boldsymbol{\vartheta}|\boldsymbol{y})$. Then, one can simulate a random sequence of states from that Markov chain that is long enough to (almost) reach the steady state and then keep some generated states as samples drawn from $p(\boldsymbol{\vartheta}|\boldsymbol{y})$. The *Metropolis-Hasting* (MH) [32] algorithm allows to construct a Markov chain such that its stationary distribution is exactly the desired distribution $p(\boldsymbol{\vartheta}|\boldsymbol{y})$.

The MH provides the basis for the reversible jump Markov chain Monte Carlo (RJMCMC) method, [7], that has been applied to the identification of NARX models. One of the advantages of this approach, is that it naturally provides a framework for quantifying model uncertainty (in both parameters and structure). The basic idea of this method is to move between the states of a Markov chain, which represent models of k terms. Three basic operations have been defined to move between states:

- Birth move: sampling of unselected terms to include in the current model, *i.e.,* $k' = k + 1$.
- Death move: sampling of previously selected terms to remove from the current model, *i.e.,* $k' = k - 1$.
- Update: updating the parameters of the existing terms, along with the variance, *i.e.,* $k' = k$.

At each iteration one of this operations is randomly attempted according to some probabilities and its result is accepted or rejected according to an acceptance ratio. The probabilities of performing such operations are updated according to the likelihood that the size of the real model is larger or smaller than the current model. This method solves jointly both the model structure selection and the parameter estimation problems, performing a global search of the term space and naturally including a pruning method to remove incorrect terms. However appealing it may be, the joint solution of the two problems appears to be an extremely difficult task, essentially because the value of the same parameter can vary greatly depending on the structure of the model in which the corresponding regressor appears, so that the distribution of the parameter values over models is typically very complex. Thus, everytime a birth or death move takes place, a discontinuous variation of the optimal parameters generally occurs, affecting the parameter update process.

Although MCMC methods tend to be accurate asymptotically, they are also computationally intensive and can be prohibitively slow, because they tend to rely on large numbers of samples, and may exhibit slow convergence to a stationary distribution.

2.3.7 An illustrative example

Consider the following nonlinear dynamical system [7]:

$$y(t) = 0.7y(t-1)u(t-1) - 0.5y(t-2) - 0.7y(t-2)u(t-2)^2 + 0.6u(t-2)^2 + e(t), \quad (2.48)$$

where the input signal $u(t)$ is uniformly distributed in the range $(-1, 1)$ and the noise $e(t)$ is a white Gaussian noise with variance 0.004. Let the candidate regressor set be composed by all the monomials of degree less or equal to 3 obtained from $\boldsymbol{x}(t) = [y(t-1), \cdots, y(t-4), u(t-1), \cdots, u(t-4)]$. This amounts to $n = 165$ possible candidate model terms among which to search for the optimal subset. A data set of $N = 500$ samples was generated. The MSS approaches presented previously have been applied on this example, except the BSS approach which is computationally infeasible for the considered regressor set.

Forward stepwise methods - the FROE

Table 2.1 reports the terms selected by the FROE method, listed in the same order as they have been selected according to their importance in the model. The terms reported represent the final model structure, as selected according to the following BIC-based rule:

$$\boldsymbol{s}^k \text{ such that } \texttt{BIC}(\boldsymbol{s}^{k+1}) \geq \texttt{BIC}(\boldsymbol{s}^k),$$

where $\boldsymbol{s}^k$ is the structure at iteration k.

As can be noticed, the constant term, selected at the second step, is incorrect. This is presumably due to the local estimation of the importance of a specific term, which depends on the model structure selected so far.

Table 2.1: *Illustrative example: Model structure selection results with the FROE method.*

Sel. order	Model term	ERR	Parameter Estimate	Correct term?
1	$y(t-2)$	0.4127	-0.5078	Yes
2	1	0.2863	0.0014	No
3	$u(t-2)^2$	0.1696	0.5901	Yes
4	$y(t-1)u(t-1)$	0.1132	0.6868	Yes
5	$y(t-2)u(t-2)^2$	0.0104	-0.6660	Yes

Shrinkage methods - Ridge regression

Figure 2.1 shows the ridge coefficients, as the tuning parameter λ varies. The optimal λ value has been chosen by 5-fold cross-validation and it is represented by the vertical red dashed line. As expected, a full model has been obtained, with several coefficients close to 0.

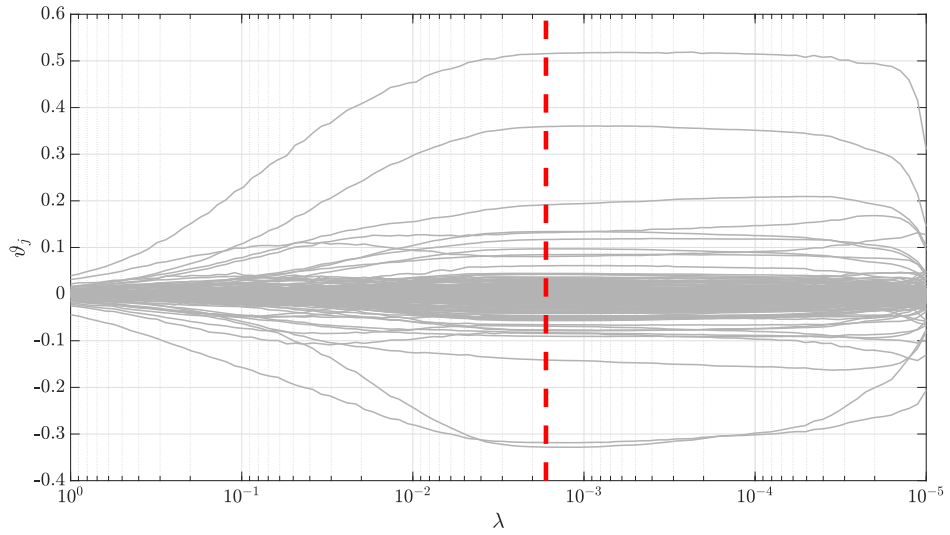

Figure 2.1: *Illustrative example: Profiles of ridge coefficients, as the tuning parameter λ is varied. The red dashed line indicates the optimal λ value chosen by cross-validation.*

Shrinkage methods - the LASSO

Figure 2.2 shows the LASSO coefficients, as the tuning parameter λ varies. Again, the optimal λ value has been chosen by 5-fold cross-validation. The LASSO solves the MSS problem returning a sparse model. However, as can be noticed in Table 2.2, several spurious terms have been added to the final best model.

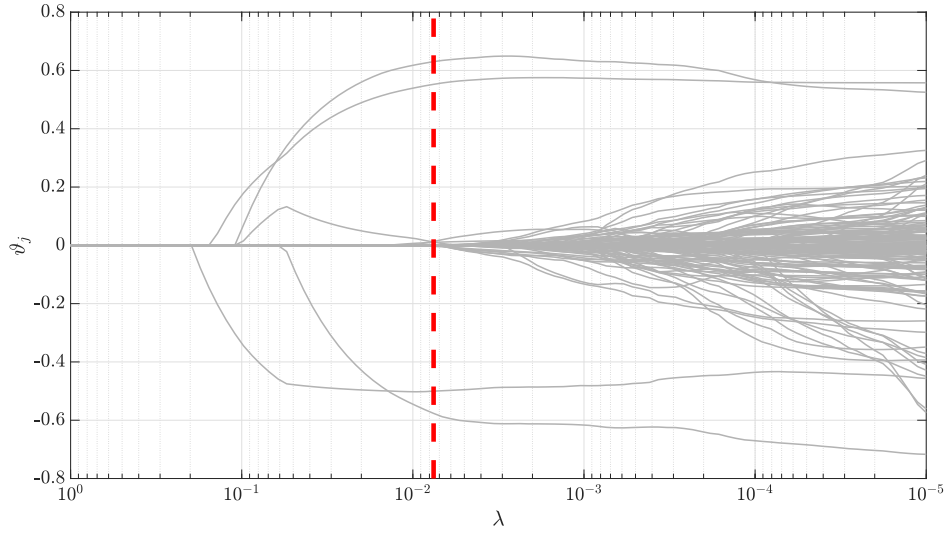

Figure 2.2: *Illustrative example: Profiles of the LASSO coefficients, as the tuning parameter λ is varied. The red dashed line indicates the optimal λ value chosen by cross-validation.*

Table 2.2: *Illustrative example: Model structure selection results with the LASSO method.*

Model term	Parameter Estimate	Correct term?
1	0.0090	No
$y(t-2)$	-0.5002	Yes
$y(t-1)u(t-1)$	0.6302	Yes
$u(t-2)^2$	0.5524	Yes
$y(t-2)u(t-1)^2$	-0.0035	No
$y(t-2)u(t-2)^2$	-0.5765	Yes
$y(t-4)u(t-1)^2$	0.0133	No
$y(t-4)u(t-2)^2$	0.0115	No
$u(t-1)^3$	0.0101	No
$u(t-1)u(t-3)^2$	0.0019	No

Dimension reduction methods - the PCR

The number of retained components has been chosen by imposing a minimum value of 2% on the total variance explained by each principal component. Accordingly, only $p = 12$ components have been considered, see Figure 2.3. Like ridge regression, also the PCR results in a full model, but this time several spurious terms have non-negligible coefficients, Figure 2.4. This issue is partially mitigated if a larger data set is employed.

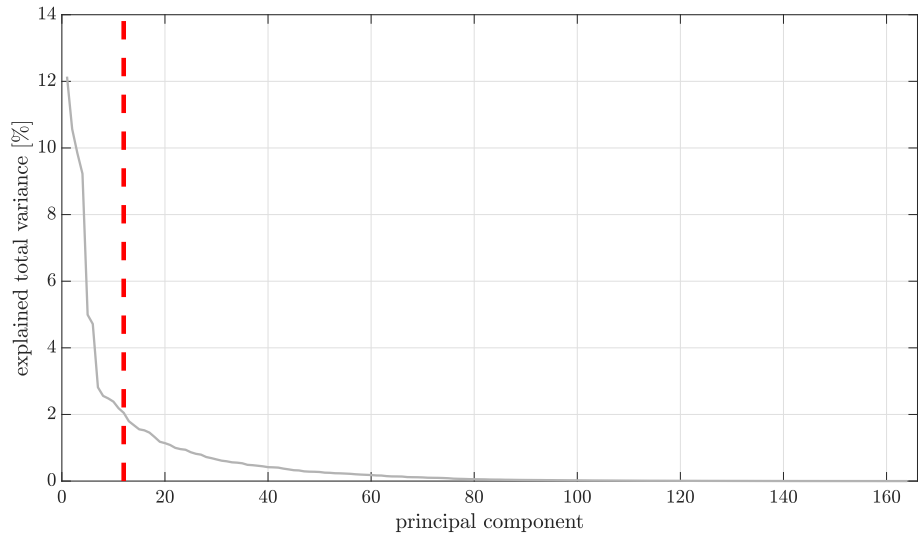

Figure 2.3: *Illustrative example: Percentage of the total variance explained by each principal component. The red dashed line indicates the optimal number of considered principal components.*

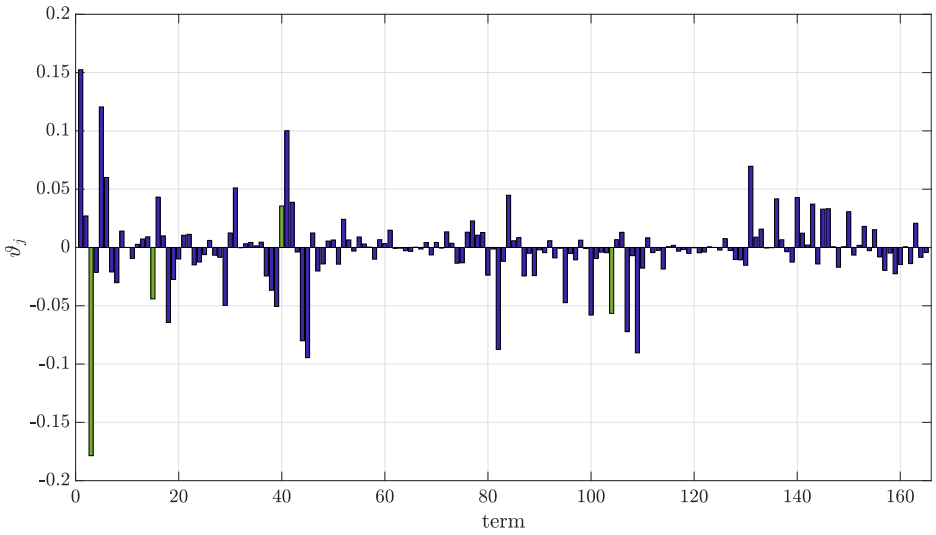

Figure 2.4: *Illustrative example: PCR coefficients. Green vertical bars correspond to the true model terms.*

Evolutionary methods - Genetic Algorithm

A GA as described in Section 2.3.5 has been implemented. Specifically, the BIC criterion has been used as fitness function, the tournament selection criterion and the single point crossover technique (which creates a random binary vector and selects the genes where the vector is a 1 from the first parent, and the genes where the vector is a 0 from the second parent, and combines the genes to form the child) have been chosen, the population size has been set to 200, and the mutation rate has been set to 0.1. To partially preserve good solutions from further destructive mutation and crossover operations, the *elitism* technique has been employed, which involves carrying a small proportion of the fittest candidates, unchanged, over to the next generation.

Table 2.3 reports the model structures selected by the GA, running the algorithm 10 times on the same data realization. Apparently, even if the correct terms always appear in the selected structures, only 3 times the final model is correct. In the remaining 7 cases, an extra term has been selected.

Bayesian methods - the RJMCMC

Table 2.4 presents directly the results reported in [7], in which the RJMCMC has been employed in the identification of NARMAX models. Specifically, the RJMCMC approach retrieved the correct model structure 7 times out of 10 runs. It is worth noting that the algorithm was run with 30000 iterations and a burn in period of 5000 iterations, confirming the fact that in general the MCMC based approaches are computationally very expensive.

Table 2.3: *Illustrative example: Model structure selection results obtained from* 10 *runs of the genetic algorithm on the same data realization. Wrong selected terms are highlighted.*

Model index	Term 1	Term 2	Term 3	Term 4	Term 5	Correct model?
1	$y(t-2)$	$y(t-1)u(t-1)$	$u(t-2)^2$	$y(t-2)u(t-2)^2$	$u(t-1)^3$	No
2	$y(t-2)$	$y(t-1)u(t-1)$	$u(t-2)^2$	$y(t-2)u(t-2)^2$	$u(t-1)^3$	No
3	$y(t-2)$	$u(t-1)$	$y(t-1)u(t-1)$	$u(t-2)^2$	$y(t-2)u(t-2)^2$	No
4	$y(t-2)$	$y(t-1)u(t-1)$	$u(t-2)^2$	$y(t-2)u(t-2)^2$	-	Yes
5	$y(t-2)$	$y(t-1)u(t-1)$	$u(t-2)^2$	$y(t-2)u(t-2)^2$	-	Yes
6	$y(t-2)$	$y(t-1)u(t-1)$	$u(t-2)^2$	$y(t-2)u(t-2)^2$	$u(t-1)^3$	No
7	$y(t-2)$	$y(t-1)u(t-1)$	$u(t-2)^2$	$y(t-2)u(t-2)^2$	$u(t-1)^3$	No
8	$y(t-2)$	$y(t-1)u(t-1)$	$u(t-2)^2$	$y(t-2)u(t-2)^2$	-	Yes
9	$y(t-2)$	$u(t-1)$	$y(t-1)u(t-1)$	$u(t-2)^2$	$y(t-2)u(t-2)^2$	No
10	$y(t-2)$	$y(t-1)u(t-1)$	$u(t-2)^2$	$y(t-2)u(t-2)^2$	$u(t-1)^3$	No

Table 2.4: *Illustrative example: Model structure selection results obtained from* 10 *runs of the RJMCMC method on the same data realization. These results are reported in [7]. Wrong selected term are highlighted.*

Model index	Term 1	Term 2	Term 3	Term 4	Term 5	Correct model?
1	$y(t-1)u(t-1)$	$y(t-2)$	$y(t-2)u(t-2)^2$	$u(t-2)^2$	-	Yes
2	$y(t-1)u(t-1)$	$y(t-2)^3$	$y(t-4)u(t-2)^2$	$u(t-2)^2$	$y(t-3)u(t-3)$	No
3	$y(t-1)u(t-1)$	$y(t-2)$	$y(t-2)u(t-2)^2$	$u(t-2)^2$	-	Yes
4	$y(t-1)u(t-1)$	$y(t-2)$	$y(t-2)u(t-2)^2$	$u(t-2)^2$	-	Yes
5	$y(t-1)u(t-1)$	$y(t-4)$	$y(t-2)u(t-2)^2$	$u(t-2)^2$	$u(t-4)^2$	No
6	$y(t-1)u(t-1)$	$y(t-2)$	$y(t-2)u(t-2)^2$	$u(t-2)^2$	-	Yes
7	$y(t-1)u(t-1)$	$y(t-2)$	$y(t-2)u(t-2)^2$	$u(t-2)^2$	-	Yes
8	$y(t-1)u(t-1)$	$y(t-2)$	$y(t-2)u(t-2)^2$	$u(t-2)^2$	-	Yes
9	$y(t-1)u(t-1)$	$y(t-2)$	$y(t-2)u(t-2)^2$	$u(t-2)^2$	-	Yes
10	$y(t-3)u(t-1)$	$y(t-2)u(t-4)^2$	$y(t-2)u(t-2)^2$	$u(t-2)^2$	$u(t-1)u(t-3)^2$	No

CHAPTER 3

A randomized model structure selection method

THIS chapter considers the Randomized Model Structure Selection (RaMSS) method, [14, 36]. The RaMSS method is a randomized model structure selection approach for NARX model identification, based on a probabilistic representation of the model structure, whereby a probability distribution representing the likelihood of each model structure to be the true one is iteratively refined by a sample-and-evaluate procedure until convergence to a limit distribution that can be associated to a specific NARX model structure. Specifically, a collection of independent Bernoullian distributions is employed in the RaMSS to account for the presence (or absence) of each regressor in the model.

In the following, we directly present this method in a more generic form than that used in the seminal paper [36], which will allow us to extend it in order to cope with the problems we investigate in this thesis. Then, the presented randomized approach is applied to the illustrative example discussed in Section 2.3.7, with reference to the identification of NARX models, as originally done in [36].

3.1 A probabilistic framework for combinatorial optimization problems

Let $\boldsymbol{x} = (x_1, x_2, \ldots, x_n)$ be a collection of n discrete variables variables with $x_j \in \mathcal{X}_j = \{1, \ldots, m_j\}$, $j = 1, \ldots, n$. Consider a combinatorial optimization problem where the goal is to find a value of $\boldsymbol{x}$ that maximizes a given performance index $\mathcal{J} : \mathcal{X} \to \mathbb{R}^+$, with $\mathcal{X} = \mathcal{X}_1 \times \ldots \times \mathcal{X}_n$. If such a value is unique, we can define it as:

$$\boldsymbol{x}^\star = (x_1^\star, \ldots, x_n^\star) = \arg\max_{\boldsymbol{x} \in \mathcal{X}} \mathcal{J}(\boldsymbol{x}). \tag{3.1}$$

Let us introduce a random variable $\gamma_j \sim \texttt{Categorical}(\boldsymbol{\pi}_j)$ [1] for each term x_j, where $\boldsymbol{\pi}_j = (\pi_j^{(1)}, \ldots, \pi_j^{(m_j)})$ and $\pi_j^{(i)}$ represents the probability that x_j takes the i-th value ($\sum_{i=1}^{m_j} \pi_j^{(i)} = 1$). If we assume that the γ_j variables are independent[2], then the probability that the collection of random variables $\boldsymbol{\gamma} = (\gamma_1, \ldots, \gamma_n)$ takes the value $\boldsymbol{x} = (x_1, \ldots, x_n) \in \mathcal{X}$ is uniquely defined by $\boldsymbol{\pi} = (\boldsymbol{\pi}_1, \ldots, \boldsymbol{\pi}_n)$. More precisely, we have that

$$\mathbb{P}_{\boldsymbol{\gamma}}(\boldsymbol{x}) = \prod_{j=1}^{n} \prod_{i=1}^{m_j} \left(\pi_j^{(i)}\right)^{\beta_j^{(i)}}, \tag{3.2}$$

where $\beta_j^{(i)} = 1$ if $x_j = i$, and 0 otherwise.

The expected performance of $\boldsymbol{\gamma}$ can then be computed as follows:

$$\mathbb{E}_{\mathbb{P}_{\boldsymbol{\gamma}}}[\mathcal{J}(\boldsymbol{\gamma})] = \sum_{\boldsymbol{x} \in \mathcal{X}} \mathcal{J}(\boldsymbol{x}) \mathbb{P}_{\boldsymbol{\gamma}}(\boldsymbol{x}). \tag{3.3}$$

The value of $\mathbb{E}_{\mathbb{P}_{\boldsymbol{\gamma}}}[\mathcal{J}(\boldsymbol{\gamma})]$ is a function of $\mathbb{P}_{\boldsymbol{\gamma}}$, and its maximum is obtained if the distribution $\mathbb{P}_{\boldsymbol{\gamma}}$ is such that all the probability mass is concentrated on $\boldsymbol{x}^\star$, which can be obtained for an appropriate choice of the parameters in $\boldsymbol{\pi}$. In view of this, the value $\boldsymbol{x}^\star$ that maximizes $\mathcal{J}(\boldsymbol{x})$ can be also obtained as:

$$\boldsymbol{x}^\star = \arg\max_{\boldsymbol{x} \in \mathcal{X}} \mathbb{P}^\star_{\boldsymbol{\gamma}}(\boldsymbol{x}), \tag{3.4}$$

where

$$\mathbb{P}^\star_{\boldsymbol{\gamma}} = \arg\max_{\mathbb{P}_{\boldsymbol{\gamma}}} \mathbb{E}_{\mathbb{P}_{\boldsymbol{\gamma}}}[\mathcal{J}(\boldsymbol{\gamma})]$$

is called the target limit distribution. Now, let

$$\delta_j^{(i)} = \mathbb{E}_{\mathbb{P}_{\boldsymbol{\gamma}}}[\mathcal{J}(\boldsymbol{\gamma}) | \gamma_j = i] - \mathbb{E}_{\mathbb{P}_{\boldsymbol{\gamma}}}[\mathcal{J}(\boldsymbol{\gamma}) | \gamma_j \neq i] \tag{3.5}$$

for $i = 1, \ldots, m_j$, $j = 1, \ldots, n$, where the conditional expectations are set equal to 0 if the conditional event has 0 probability to happen. Index $\delta_j^{(i)}$, compares the average performance of those solutions having x_j taking the i-th value, with all the remaining ones.

Theorem 1. *Let $\mathbb{P}_{\boldsymbol{\gamma}}$ be the probability distribution over $\mathcal{X}$ defined according to* (3.2)*. Then, there exists $\varrho \in (0, 1)$ such that if $\mathbb{P}_{\boldsymbol{\gamma}}(\boldsymbol{x}^\star) \geq \varrho > \max_{\boldsymbol{x} \in \mathcal{X} \setminus \{\boldsymbol{x}^\star\}} \frac{\mathcal{J}(\boldsymbol{x})}{\mathcal{J}(\boldsymbol{x}^\star)}$ it holds that $\delta_j^{(i)} > 0$ if $x_j^\star = i$ and $\delta_j^{(i)} < 0$ otherwise, $i = 1, \ldots, m_j$, $j = 1, \ldots, n$.*

Proof. The proof goes along the lines of that reported in Appendix A.1 in [36], extending that result to Categorical distributions with more than two outcomes. The proof is here reported for the sake of clarity within the notation introduced in this thesis.

Consider first the case $x_j^\star = i$. Then, the index $\delta_j^{(i)}$ (3.5) can be bounded from below as follows:

$$\delta_j^{(i)} \geq \mathcal{J}(\boldsymbol{x}^\star) \mathbb{P}_{\boldsymbol{\gamma}}(\boldsymbol{x}^\star) - \bar{\mathcal{J}}_j^{(i)}, \tag{3.6}$$

[1] A categorical random variable can take one of n possible values (or categories), with the probability of each category separately specified. The outcomes are often numbered for convenience, *e.g.* from 1 to n. The parameters specifying the probabilities of each possible outcome must be in the range $[0, 1]$, and must sum to 1. The categorical distribution is the generalization of the Bernoulli distribution for $n > 2$.

[2] The introduced probability distribution quantifies our belief regarding the fact that $x_j^\star$ takes a specific value. By assuming the independence of the γ_j variables, we are not letting our belief regarding one specific variable affect the belief for the remaining ones.

where $\bar{\mathcal{J}}_j^{(i)} = \max_{\boldsymbol{x}\in\mathcal{X}:x_j\neq i} \mathcal{J}(\boldsymbol{x})$. Indeed, for the first term in the RHS of (3.5),

$$\mathbb{E}_{\mathbb{P}_{\boldsymbol{\gamma}}}[\mathcal{J}(\boldsymbol{\gamma})|\gamma_j = i] = \sum_{\boldsymbol{x}\in\mathcal{X}:x_j=i} \mathcal{J}(\boldsymbol{x})\mathbb{P}_{\boldsymbol{\gamma}}(\boldsymbol{x}) \geq \mathcal{J}(\boldsymbol{x}^\star)\mathbb{P}_{\boldsymbol{\gamma}}(\boldsymbol{x}^\star), \tag{3.7}$$

where the inequality follows upon observing that $\boldsymbol{x}^\star \in \{\boldsymbol{x} \in \mathcal{X} : x_j = i\}$ and that $\mathcal{J}(\boldsymbol{x}) \geq 0$.
On the other hand, the second term in the RHS of (3.5),

$$\mathbb{E}_{\mathbb{P}_{\boldsymbol{\gamma}}}[\mathcal{J}(\boldsymbol{\gamma})|\gamma_j \neq i] \leq \bar{\mathcal{J}}_j^{(i)}, \tag{3.8}$$

by definition. Therefore, applying the bounds 3.7 and 3.8 in (3.5), one obtains (3.6).

A similar reasoning applies for the case $x_j^\star \neq i$, leading to the following bound:

$$\delta_j^{(i)} \leq \tilde{\mathcal{J}}_j^{(i)} - \mathcal{J}(\boldsymbol{x}^\star)\mathbb{P}_{\boldsymbol{\gamma}}(\boldsymbol{x}^\star), \tag{3.9}$$

where $\tilde{\mathcal{J}}_j^{(i)} = \max_{\boldsymbol{x}\in\mathcal{X}:x_j=i} \mathcal{J}(\boldsymbol{x})$. Indeed, for the first term in the RHS of (3.5),

$$\mathbb{E}_{\mathbb{P}_{\boldsymbol{\gamma}}}[\mathcal{J}(\boldsymbol{\gamma})|\gamma_j = i] \leq \tilde{\mathcal{J}}_j^{(i)}, \tag{3.10}$$

by definition.
The second term can be bounded as

$$\mathbb{E}_{\mathbb{P}_{\boldsymbol{\gamma}}}[\mathcal{J}(\boldsymbol{\gamma})|\gamma_j \neq i] = \sum_{\boldsymbol{x}\in\mathcal{X}:x_j\neq i} \mathcal{J}(\boldsymbol{x})\mathbb{P}_{\boldsymbol{\gamma}}(\boldsymbol{x}) \geq \mathcal{J}(\boldsymbol{x}^\star)\mathbb{P}_{\boldsymbol{\gamma}}(\boldsymbol{x}^\star). \tag{3.11}$$

Therefore, applying the bounds 3.10 and 3.11 in (3.5), one obtains (3.9).

Now, under the assumption that $\boldsymbol{x}^\star$ is unique, if one sets

$$\varrho > \max_{\boldsymbol{x}\in\mathcal{X}\setminus\{\boldsymbol{x}^\star\}} \frac{\mathcal{J}(\boldsymbol{x})}{\mathcal{J}(\boldsymbol{x}^\star)}$$

and $\mathbb{P}_{\boldsymbol{\gamma}}(\boldsymbol{x}^\star) \geq \varrho$, one obtains that $\delta_j^{(i)} > 0$ if $x_j^\star = i$, from bound 3.6. On the other hand, $\delta_j^{(i)} < 0$ if $x_j^\star \neq i$, from bound 3.9. □

Theorem 1 suggests that, when $\mathbb{P}_{\boldsymbol{\gamma}}$ is sufficiently close to $\mathbb{P}_{\boldsymbol{\gamma}}(\boldsymbol{x}^\star)$, then the sign of $\delta_j^{(i)}$ provides a reliable information for tuning the $\pi_j^{(i)}$ parameters towards those in $\mathbb{P}_{\boldsymbol{\gamma}}(\boldsymbol{x}^\star)$. This information can then be used to iteratively refine $\pi_j^{(i)}(k)$ (where k is the iteration index) according to the following update rule:

$$\pi_j^{(i)}(k+1) = \pi_j^{(i)}(k) + \chi\delta_j^{(i)}, \tag{3.12}$$

where $\chi > 0$. In order for the Categorical distribution to be well defined, a normalization step is required after the application of (3.12), so that $0 \leq \pi_j^{(i)}(k+1) \leq 1$ and $\sum_{i=1}^{m_j} \pi_j^{(i)}(k+1) = 1$.

Theorem 2. *Let $\mathbb{P}_{\boldsymbol{\gamma}}$ be the probability distribution over $\mathcal{X}$ defined according to* (3.2)*, and assume that $\boldsymbol{\pi}$ is such that $\mathbb{P}_{\boldsymbol{\gamma}}(\boldsymbol{x}^\star) \geq \varrho$, ϱ being a value for which Theorem 1 holds. Then, the local convergence to the target limit distribution $\mathbb{P}_{\boldsymbol{\gamma}}^\star$ is guaranteed by the iterative application of* (3.12) *starting from $\boldsymbol{\pi}$.*

Proof. Let $\mathbb{P}_{\boldsymbol{\gamma}}^{(k)}$ be the probability distribution associated with the probability matrix $\boldsymbol{\pi}$ at iteration k. Assuming that $\mathbb{P}_{\boldsymbol{\gamma}}^{(k)}(\boldsymbol{x}^\star) \geq \varrho$, where ϱ makes the condition of Theorem (1) valid. Then one obtains that:

$$\begin{cases} \delta_j^{(i)} > 0 & \forall j : x_j^\star = i \\ \delta_j^{(i)} < 0 & \forall j : x_j^\star \neq i \end{cases}$$

and therefore, according to (3.12) and recalling that $\chi > 0$:

$$\begin{cases} \pi_j^{(i)}(k+1) = \pi_j^{(i)}(k) + \chi\delta_j^{(i)} > \pi_j^{(i)}(k) & \forall j : x_j^\star = i \\ \pi_j^{(i)}(k+1) = \pi_j^{(i)}(k) + \chi\delta_j^{(i)} < \pi_j^{(i)}(k) & \forall j : x_j^\star \neq i \end{cases}$$

Recalling that:

$$\mathbb{P}_{\boldsymbol{\gamma}}^{(k)}(\boldsymbol{x}) = \prod_{j=1}^{n}\prod_{i=1}^{m}\left(\pi_j^{(i)}(k)\right)^{\beta_j^{(i)}}$$

it follows that

$$\mathbb{P}_{\boldsymbol{\gamma}}^{(k+1)}(\boldsymbol{x}^\star) > \mathbb{P}_{\boldsymbol{\gamma}}^{(k)}(\boldsymbol{x}^\star) > \varrho.$$

We then have a sequence of strictly monotonically increasing scalars that are bounded from above by 1, which entails that $\lim_{k\to\infty} \mathbb{P}_{\boldsymbol{\gamma}}^{(k)}(\boldsymbol{x}^\star) = 1$. □

3.2 Implementation issues

In practice only an approximate sampled version of $\delta_j^{(i)}$ in (3.5) can be computed, since the exact computation of the conditional expectations would require to exhaustively explore the entire solution space $\mathcal{X}$. Therefore, at each iteration, N_p candidate solutions $\boldsymbol{x}$ are sampled from $\mathbb{P}_\gamma$ and evaluated according to the performance index $\mathcal{J}(\boldsymbol{x})$. The probabilities $\pi_j^{(i)}$ are hence updated according to (3.12), where sample estimates are used in place of the expected values, and the algorithm proceeds to the next iteration. The algorithm ends when a stopping criterion is met. This can either be associated with a maximum number of iterations, or a practical convergence of the $\pi_j^{(i)}$ parameters, which is achieved when the relative difference between the $\pi_j^{(i)}$ calculated at subsequent iterations is lower than a given threshold.

The convergence speed is affected by the choice of the step size χ in (3.12). Indeed, too small values would slow down the algorithm, while too large values might cause instability. Therefore, in [35, 36], the authors proposed to update χ at each iteration according to the performance dispersion in the extracted population of solutions. Specifically,

$$\chi = \frac{1}{10\left(\mathcal{J}_{\text{best}} - \overline{\mathcal{J}}\right) + 0.1} \tag{3.13}$$

where $\mathcal{J}_{\text{best}}$ and $\overline{\mathcal{J}}$ are, respectively, the best value and the mean value for $\mathcal{J}$ evaluated on the extracted samples for $\boldsymbol{\gamma}$. The rationale behind this adaptive rule is to allow the algorithm to freely explore the solution space in the early stages, in order to gather as much information as possible. In this exploration phase, however, the correction terms $\delta_j^{(i)}$ may vary erratically, and thus their influence in the update equations has to

be limited. Later on, when the suggested corrections become more stable, the step size should be incremented to accelerate convergence.

Regarding the initialization of $\pi_j^{(i)}(0)$, in absence of any *a priori* information about the optimal solution $\boldsymbol{x}^\star$, it is reasonable to assume that all possible outcomes of x_j are equally probable, *i.e.,* $\pi_j^{(i)} = 1/m_j$, $i = 1, \ldots, m_j$.

The proposed approach has some features in common with GAs, and others EAs, in that it exploits randomness in choosing the values for the discrete variables and in that it processes populations of candidate solutions. However, these two methodologies differ significantly on how the population evolves in time. In our framework, the fitness of each possible value for x_j is evaluated from an aggregated analysis of the whole population. Specifically, all individuals of the population contribute to this evaluation, either reinforcing or discouraging the assignment of the i-th value to x_j. Then,- the new population is generated from scratch, based on the aggregate information derived from the current population. Instead, in GAs, only the fittest individuals in the current population are selected and manipulated in order to generate the new population.

3.3 Identification of NARX models

The identification of NARX models can be easily dealt within the presented probabilistic approach, as originally proposed in [36]. In particular, note that the RaMSS method in [36] is a specialization of (3.1)–(3.12) when x_j is a binary variable, and $\gamma_j \sim \texttt{Bernoullian}(\boldsymbol{\pi}_j)$, whose success probability $\boldsymbol{\pi}_j$ represents the belief that x_j takes value 1, *i.e.* that the regressor φ_j is present in the model. This probability has been named Regressor Inclusion Probability, or RIP, and in absence of any *a priori* information about the correct model structure, it is common to initially set all the RIPs to an equal small value, thus encouraging the extraction of sparse models in the early stages of the algorithm, whose iteration is now detailed.

To avoid ambiguity with the NARX notation, let $\boldsymbol{s} \in \Sigma = \{0, 1\}^n$ encode a candidate solution *i.e.,* a model structure, while $\boldsymbol{x}(t)$ denotes the finite-dimensional vector of lagged input and output, as usual. Under the assumption that $\gamma_1, \ldots, \gamma_n$ are independent[3] from each other, at each iteration N_p candidate model structures $\boldsymbol{s}$ are sampled from

$$\mathbb{P}_\gamma(\boldsymbol{\gamma} = \boldsymbol{s}) = \prod_{j:s_j=1} \boldsymbol{\pi}_j \prod_{j:s_j=0} (1 - \boldsymbol{\pi}_j)\,, \tag{3.14}$$

and their parameters are estimated by means of an LS procedure. Then, for each model all statistically non-significant regressors are removed via a t-test (see Section 2.1.1), and the parameters re-estimated after the removal. The resulting models are evaluated according to the performance index:

$$\mathcal{J}(\boldsymbol{s}) = e^{-K\mathcal{L}(y,\hat{y})}, \tag{3.15}$$

with $\mathcal{L}(y, \hat{y})$ defined in (2.10), where the one-step-ahead predictor $\hat{y}$ is computed according to the extracted model structure $\boldsymbol{s}$, and K is a scaling parameter ($K = 1$ in the following). Exponential indices can facilitate the discrimination between models with similar performance by amplifying their difference [110], thus improving the structure

[3] see note 2.

selection process. Finally, the RIPs are updated according to (3.12), which is actually similar to a gradient-based update rule. Indeed, note that for all the Bernoullian variables γ_j, $j = 1, \dots, n$, the following equation holds:

$$\mathbb{E}_{\mathbb{P}_{\boldsymbol{\gamma}}}[\mathcal{J}(\boldsymbol{\gamma})] = \boldsymbol{\pi}_j \mathbb{E}_{\mathbb{P}_{\boldsymbol{\gamma}}}[\mathcal{J}(\boldsymbol{\gamma})|\gamma_j = 1] + (1 - \boldsymbol{\pi}_j)\, \mathbb{E}_{\mathbb{P}_{\boldsymbol{\gamma}}}[\mathcal{J}(\boldsymbol{\gamma})|\gamma_j = 0]. \tag{3.16}$$

Now, deriving (3.16) w.r.t. $\boldsymbol{\pi}_j$, one obtains :

$$\left.\frac{\partial \mathbb{E}_{\mathbb{P}_{\boldsymbol{\gamma}}}[\mathcal{J}(\boldsymbol{\gamma})]}{\partial \boldsymbol{\pi}_j}\right| = \mathbb{E}_{\mathbb{P}_{\boldsymbol{\gamma}}}[\mathcal{J}(\boldsymbol{\gamma})|\gamma_j = 1] - \mathbb{E}_{\mathbb{P}_{\boldsymbol{\gamma}}}[\mathcal{J}(\boldsymbol{\gamma})|\gamma_j = 0], \tag{3.17}$$

which is the definition of δ_j in (3.5). However, an important technicality is missing: δ_j is computed on the set of non-redundant models (due to the t-test), which is a subset of the entire model structure space on which $\mathbb{P}_{\boldsymbol{\gamma}}$ is actually defined. Therefore, δ_j is not directly interpretable as the gradient of $\mathbb{E}_{\mathbb{P}_{\boldsymbol{\gamma}}}[\mathcal{J}(\boldsymbol{\gamma})]$ w.r.t. $\boldsymbol{\pi}_j$. Nevertheless, the local convergence to $\mathbb{P}^{\star}_{\boldsymbol{\gamma}}$ is still guaranteed by Theorem 2.

Algorithm 2 reports the whole randomized identification procedure.

Algorithm 1 NARX model generation with Randomized MSS

Require: $\{(\boldsymbol{\varphi}(t), \boldsymbol{y}(t)), t = 1, \cdots, N\}, n, \boldsymbol{\pi}$
Ensure: $\boldsymbol{s}, \hat{\boldsymbol{\vartheta}}$

$\boldsymbol{\Psi(t)} \leftarrow [\,]$;
$\boldsymbol{s} \leftarrow [\,]$;
$m \leftarrow 0$
for $j = 1$ **to** n **do**
  Extract r_j from Bernoullian(π_j); ▷ Generate terms
  $\boldsymbol{s} \leftarrow [\boldsymbol{s},\ r_j]$
  if $r_j = 1$ **then**
    $\boldsymbol{\Psi(t)} \leftarrow [\boldsymbol{\Psi(t)},\ \varphi_j(t)]$;
    $m \leftarrow m + 1$;
  end if
end for
$\hat{\boldsymbol{\vartheta}} \leftarrow \left(\sum_{t=1}^{N} \boldsymbol{\Psi(t)}^T \boldsymbol{\Psi(t)}\right)^{-1} \sum_{t=1}^{N} \boldsymbol{\Psi(t)}^T y(t)$; ▷ LS estimate
$\boldsymbol{V} \leftarrow \left(\sum_{t=1}^{N} \boldsymbol{\Psi(t)}^T \boldsymbol{\Psi(t)}\right)^{-1}$;
$\hat{\sigma}^2 \leftarrow \frac{1}{N-m} \sum_{t=1}^{N} \left(y(t) - \boldsymbol{\Psi(t)}\hat{\boldsymbol{\vartheta}}\right)^2$;
for $i = 1$ **to** m **do** ▷ Remove redundant regressors
  $z_i \leftarrow \frac{\hat{\vartheta}_i}{\hat{\sigma}\sqrt{\boldsymbol{V}_{ii}}}$; ▷ t-test
  if $|z_i| < t_{\alpha/2, N-m}$ **then**
    Remove regressor $\Psi_i(t)$ from $\boldsymbol{\Psi(t)}$;
    $s_i \leftarrow 0$;
  end if
end for
$\hat{\boldsymbol{\vartheta}} \leftarrow \left(\sum_{t=1}^{N} \boldsymbol{\Psi(t)}^T \boldsymbol{\Psi(t)}\right)^{-1} \sum_{t=1}^{N} \boldsymbol{\Psi(t)}^T (t) y(t)$; ▷ LS estimate

3.3.1 Continuation of illustrative example

Figure 3.1 illustrates a typical run of the RaMSS algorithm when it is applied to the illustrative example (2.48). Specifically, the evolution over iterations of the RIPs and

Algorithm 2 NARX identification procedure with Randomized MSS

Require: $\{(\boldsymbol{\varphi}(t), \boldsymbol{y}(t)), t = 1, \cdots, N\}, n, N_p, \boldsymbol{\pi}, \pi_{min}, \pi_{max}, K, \alpha, \varepsilon$
Ensure: $\boldsymbol{\pi}$

$\boldsymbol{\pi} \leftarrow \frac{1}{n} \cdot \mathbf{1}_{n\times 1}$;
repeat
  for $p = 1$ **to** N_p **do**
    $\{\boldsymbol{s}^{(p)}, \hat{\boldsymbol{\vartheta}}\} \leftarrow$ Algorithm 1; ▷ Generate NARX model
    $\mathcal{J}^{(p)} \leftarrow e^{-K\mathcal{L}(y,\hat{y})}$; ▷ Model evaluation
  end for
  for $j = 1$ **to** n **do** ▷ Update π_j
    $\mathcal{J}^{\oplus} \leftarrow 0$; $n^{\oplus} \leftarrow 0$; $\mathcal{J}^{\ominus} \leftarrow 0$; $n^{\ominus} \leftarrow 0$;
    for $p = 1$ **to** N_p **do**
      if $s_j^{(p)} = 1$ **then**
        $\mathcal{J}^{\oplus} \leftarrow \mathcal{J}^{\oplus} + \mathcal{J}^{(p)}$; $n^{\oplus} \leftarrow n^{\oplus} + 1$;
      else
        $\mathcal{J}^{\ominus} \leftarrow \mathcal{J}^{\ominus} + \mathcal{J}^{(p)}$; $n^{\ominus} \leftarrow n^{\ominus} + 1$;
      end if
    end for
    $\chi \leftarrow \frac{1}{10\left(\mathcal{J}_{\text{best}} - \overline{\mathcal{J}}\right)+0.1}$;
    $\pi_j \leftarrow \pi_j + \chi\left(\frac{\mathcal{J}^{\oplus}}{\max(n^{\oplus},1)} - \frac{\mathcal{J}^{\ominus}}{\max(n^{\ominus},1)}\right)$;
    $\pi_j \leftarrow \max\left(\min\left(\pi_j, \pi_{max}\right), \pi_{min}\right)$; ▷ Saturation
  end for
until Stopping criterion

of the average model size (AMS) are reported. Apparently, the RIPs associated to the correct model terms are steadily increasing until they reach 1, while all other RIPs are squashed toward 0 after a transient. Worth noticing, some spurious parameters initially exhibit non-negligible RIPs, but as the algorithm proceeds, it is able to reject them based only on the information gathered from partially correct models. Indeed, as can be noticed by the evolution of the AMS computed on the extracted model structures at each iteration, small and incomplete models are typically analyzed. Also, the final AMS is essentially monotonically increasing to the correct value, emphasizing the incremental nature of the selection procedure which starts from very small models (mainly due to the initial choice made for the RIPs) and progressively increases the size of the explored models.

Table 3.1: *Illustrative example: Model structure selection results obtained from* 10 *runs of the RaMSS on the same data realization.*

	RaMSS - $N_p = 200$
Correct selection	100%
# of Iterations	31
Elapsed Time [sec]	6.56
Maximum AMS	3.99
Final AMS	3.98
Explored Models	731.3

Table 3.1 reports the aggregated results obtained from 10 runs of the RaMSS over the same data set of $N = 500$ input/output pairs. The following statistics have been studied:

correctness: percentage of exact final model selections,

number of iterations: number of performed algorithm iterations,

elapsed time: time required to obtain the final model,

maximum AMS: maximum model size amongst all the mean sizes computed at each iteration,

final AMS: average model size of the last iteration,

explored models: number of distinct models explored by the algorithm.

As can be noticed, the algorithm retrieved the correct model structure in all runs by exploring a very tiny fraction of the entire solution space. Indeed, on average, only 731.3 distinct models have been tested among the 2^{165} possible ones. Furthermore, the obtained maximum and final AMS values confirm the capability of the RaMSS to extract useful information from small and partially correct models.

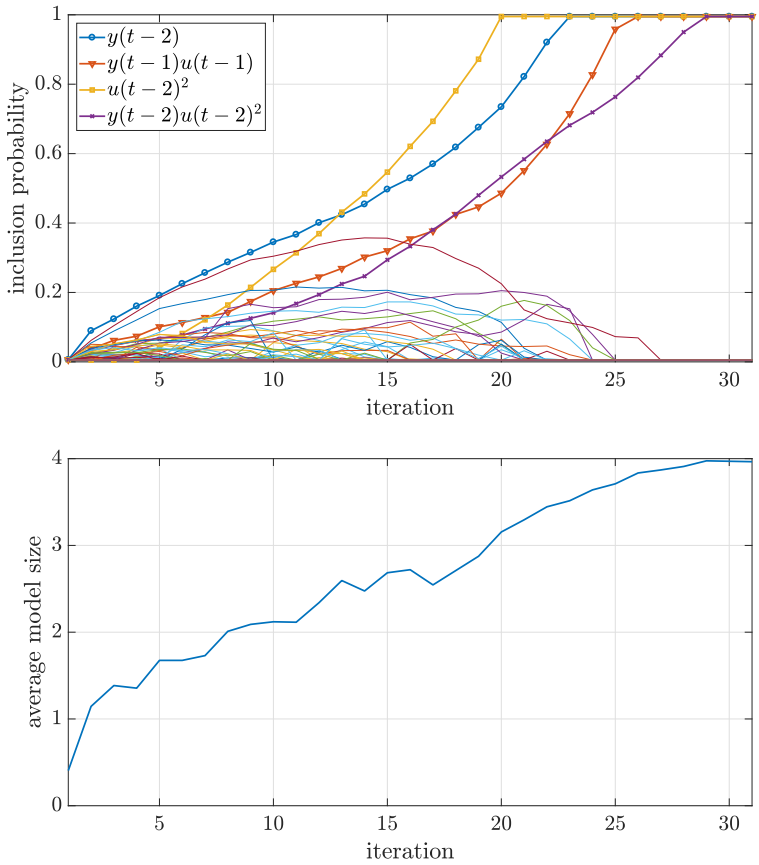

Figure 3.1: *Illustrative example: RaMSS, a typical run. RIPs (top) and AMS (bottom) evolution over* 31 *iterations.*

CHAPTER 4

Identification of nonlinear systems via distributed computation

THIS chapter deals with the identification of NARX systems when data are collected separately by multiple entities and cannot be made available to a central unit. Consider for example the case in which a system is monitored by several agents, each one collecting its own data set which cannot be make centrally available due to *e.g.,* privacy or communication constraints. Or else, several data sets are available, each of them coming from a specific independent experiment performed on the system. In this case, joining the data could not make sense at all, since *e.g.,* the chronological order of the data points would be lost, thus impairing the application of several identification approaches. Or simply, one wants to split the whole data set in smaller chunks and spread the computation over a network of calculators to distribute the load and speed up the task. In this chapter, we address this novel set-up and consider the case in which multiple agents are cooperatively aiming at identifying a model for a nonlinear system, by local computations based on private data sets. The problem of identifying the structure and parameters of the system has a mixed discrete and continuous nature, which hampers the application of classical distributed schemes. Based on the randomized MSS approach described in Chapter 3, we reformulate the MSS problem in terms of the optimization of a common probability distribution over the space of all possible model structures, thus transforming the purely combinatorial MSS task into a continuous optimization problem. Based on this reformulation, we develop a distributed computation method to address both MSS and parameter estimation.

In this chapter, after reviewing the state of the art, we introduce the novel NARX model identification algorithm with distributed computation. The algorithm perfor-

mance is illustrated through the analysis of some numerical examples and a real application involving weather data, *e.g.,* humidity and temperature. Some conclusive remarks end the chapter.

4.1 State of the art

In the literature, there are various examples of *linear-in-the-parameters* regression problems solved according to distributed approaches, as documented *e.g.* in [47, 81, 111].

In [47], a distributed message passing algorithm for performing regression within a sensor network has been proposed based on a distributed application of Gaussian elimination to solve linear systems. In particular, the authors addressed the problem of extracting complete information about the shape and structure of sensor data, while still using as less as possible the communication capability of the network. To do that, they exploited the redundancy in readings from a sensor over time and also the redundancy between measurements performed by different nodes. Specifically, they describe the sensor data through a kernel-based linear regression model (depending on both time and spatial location of the nodes), whose parameters are estimated by applying the LS method with a root mean squared error as the optimization metric. In [81], the authors proposed three algorithms to estimate the regression coefficients via a consensus-based reformulation of the LASSO, in order to cope with distributed training data. In particular, the optimization problem with the LASSO criterion is first reformulated into a separable form, which is then iteratively minimized using the alternating-direction method of multipliers (ADMM) [24] so as to gain the desired degree of parallelization. The novel algorithms differ on how this iterative minimization is performed, *i.e.,* with a distributed quadratic programming or a distributed coordinate descent approach. In [111], the distributed sum optimization (DSO) problem [101] is addressed. In this problem, each agent has a unique and local objective function and the network goal is to minimize the sum of these functions over a constrained set. The authors proposed a distributed gradient-based update in combination with a consensus-based tracking step inspired by [90]. This general framework has been applied to the problem of vertically and horizontally distributed regression in large peer to peer systems.

However, none of the mentioned methods deals explicitly with MSS, which introduces discrete decision variables in the optimization problem, thus making it hard to solve. In this respect, there are only a few attempts to solve the MSS problem in a distributed fashion. Recently, in [73, 117] the authors have proposed an extension of OFR-type algorithms to select a common-structure sparse model from multiple data sets, within the NARX modeling framework. The rationale behind this technique is to evaluate independently the importance of each term in each data set, and then selecting that term which maximizes the (weighted) average importance. The selected term is hence removed from the candidate set, and the procedure is repeated. Once the common structure has been selected, the final parameter estimate is the (weighted) average of the LS estimates obtained from all the data sets, which is however not guaranteed to be optimal according to any global criterion.

Finally, it should be mentioned that a distributed MSS method for NARX models has also been proposed in [4], although from a different perspective. Indeed, here the

distribution applies to the set of candidate model terms, that are split among the agents, as opposed to the data. Specifically, each agent performs the MSS on its assigned subset of model terms over the whole set of data. Then, the agents exchange information regarding the most promising terms each of them has found. These are used to augment each private subset of model terms and a new MSS round is executed by each agent. This iterative process is repeated until convergence to a common model structure.

4.2 Problem statement

We assume that K data sets $\mathcal{D}_i = \{\big(y^{(i)}(t), u^{(i)}(t)\big)\}_{t=1}^{N_i}$ with length N_i, $i = 1, \dots, K$, have been collected from system (2.9) separately by K agents, possibly in different experimental set-ups. Let $\sigma_1^2, \dots, \sigma_K^2$ be the corresponding output process variances. We can then formulate the identification of $\boldsymbol{\vartheta}$ as the following multi-agent optimization problem:

$$\min_{\boldsymbol{\vartheta}} \mathcal{L}(\boldsymbol{\vartheta}) = \min_{\boldsymbol{\vartheta}} \sum_{i=1}^{K} \mathcal{L}_i(\boldsymbol{\vartheta}), \tag{4.1}$$

with the function $\mathcal{L}_i : \mathbb{R}^n \to \mathbb{R}$ defined as

$$\mathcal{L}_i(\boldsymbol{\vartheta}) = \frac{1}{\sigma_i^2} \sum_{t=1}^{N_i} \big(y^{(i)}(t) - \boldsymbol{\varphi}^{(i)}(t)\boldsymbol{\vartheta}\big)^2 + C\|\boldsymbol{\vartheta}\|_0 N_i, \tag{4.2}$$

where the first term accounts for the accuracy of the identified NARX model $\boldsymbol{\vartheta}$ on the data set $\mathcal{D}_i$, and the second for the model complexity (as measured by the zero norm of $\boldsymbol{\vartheta}$, $\|\boldsymbol{\vartheta}\|_0 = \text{card}\{\vartheta_j : \vartheta_j \neq 0\}$, *i.e.,* the number of non-zero entries of that vector, which corresponds to the actual model size). The latter is a regularization term introduced to prevent redundant terms from entering the model structure. In (4.2), parameter $C > 0$ tunes the accuracy-complexity trade-off. We shall denote with $\boldsymbol{\vartheta}^\star$ the optimal solution of (4.1) and by $\mathcal{L}^\star$ the corresponding optimal cost $\sum_{i=1}^{K} \mathcal{L}_i(\boldsymbol{\vartheta}^\star)$. Problem (4.1) can be handled within the probabilistic framework presented in Chapter 3.

Specifically, let $\boldsymbol{s} \in \Sigma = \{0, 1\}^n$ encode a model structure. Then its performance can be measured as the cost associated to the best parametrization compatible with this structure, *i.e.*,

$$\mathcal{J}(\boldsymbol{s}) = \min_{\boldsymbol{\vartheta} \in \Theta_{\boldsymbol{s}}} \mathcal{L}(\boldsymbol{\vartheta}) \tag{4.3}$$

where $\Theta_{\boldsymbol{s}} = \{\boldsymbol{\vartheta} : \vartheta_j = 0, \forall j : s_j = 0\}$. Let us further denote as

$$\boldsymbol{s}^\star = \arg\min_{\boldsymbol{s} \in \Sigma} \mathcal{J}(\boldsymbol{s}) \quad \text{and} \quad \mathcal{J}^\star = \min_{\boldsymbol{s} \in \Sigma} \mathcal{J}(\boldsymbol{s})$$

the best model structure and the corresponding (optimal) performance. Now, associate a Bernoulli random variable γ_j to each s_j, $j = 1, \dots, n$, whose success probability $\pi_j \in [0, 1]$ represents the belief that s_j takes value 1, *i.e.* that the regressor φ_j is present in the model. The overall discrete random variable $\boldsymbol{\gamma} = [\gamma_1, \dots, \gamma_n]$ takes values in Σ according to

$$\mathbb{P}_{\boldsymbol{\gamma}}(\boldsymbol{\gamma} = \boldsymbol{s}) = \prod_{j:s_j=1} \pi_j \prod_{j:s_j=0} (1 - \pi_j), \tag{4.4}$$

By definition of expectation, the average performance of $\boldsymbol{\gamma}$ is given by

$$\mathbb{E}_{\mathbb{P}_{\boldsymbol{\gamma}}}[\mathcal{J}(\boldsymbol{\gamma})] = \sum_{\boldsymbol{s}\in\Sigma} \mathcal{J}(\boldsymbol{s})\mathbb{P}_{\boldsymbol{\gamma}}(\boldsymbol{\gamma} = \boldsymbol{s}), \tag{4.5}$$

and, if we let $\mathbb{P}_{\boldsymbol{\gamma}}$ vary over all possible distributions over Σ, we notice that the minimum value of (4.5) as a function of $\mathbb{P}_{\boldsymbol{\gamma}}$ is obtained by making all probability mass concentrate on the true model. Formally, denoting as

$$\mathbb{P}^{\star}_{\boldsymbol{\gamma}} = \arg\min_{\mathbb{P}_{\boldsymbol{\gamma}}} \ \mathbb{E}_{\mathbb{P}_{\boldsymbol{\gamma}}}[\mathcal{J}(\boldsymbol{\gamma})], \tag{4.6}$$

it holds that $\mathbb{P}^{\star}_{\boldsymbol{\gamma}}(\boldsymbol{\gamma} = \boldsymbol{s}^{\star}) = 1$ and $\mathbb{P}^{\star}_{\boldsymbol{\gamma}}(\boldsymbol{\gamma} = \boldsymbol{s}) = 0$ for all $\boldsymbol{s} \in \Sigma \setminus \{\boldsymbol{s}^{\star}\}$. For Bernoullian $\boldsymbol{\gamma}$ variables, the update rule (3.12) can be further detailed [1]:

$$\boldsymbol{\pi}(k+1) = \boldsymbol{\pi}(k) - \alpha(k)\ \nabla_{\boldsymbol{\pi}}\mathbb{E}_{\mathbb{P}_{\boldsymbol{\gamma}}}[\mathcal{J}(\boldsymbol{\gamma})]\big|_{\boldsymbol{\pi}(k)} \tag{4.7}$$

where $\nabla_{\boldsymbol{\pi}} = \left[\frac{\partial}{\partial\pi_1}, \ldots, \frac{\partial}{\partial\pi_n}\right]^T$,

$$\left.\frac{\partial\mathbb{E}_{\mathbb{P}_{\boldsymbol{\gamma}}}[\mathcal{J}(\boldsymbol{\gamma})]}{\partial\pi_j}\right|_{\boldsymbol{\pi}(k)} = \mathbb{E}_{\mathbb{P}^k_{\boldsymbol{\gamma}}}[\mathcal{J}(\boldsymbol{\gamma})|\gamma_j = 1] - \mathbb{E}_{\mathbb{P}^k_{\boldsymbol{\gamma}}}[\mathcal{J}(\boldsymbol{\gamma})|\gamma_j = 0], \tag{4.8}$$

and $\mathbb{P}^k_{\boldsymbol{\gamma}}$ is given by (4.4) when $\boldsymbol{\pi} = \boldsymbol{\pi}(k)$.

4.3 An intuitive extension based on distributed computation

Given that (4.7) represents a gradient descent algorithm and considering the separable structure of $\mathcal{L}(\boldsymbol{\vartheta})$ in (4.1), one might be tempted to derive a distributed version of (4.7) as follows:

$$\begin{aligned}\bar{\boldsymbol{\pi}}(k) &= \frac{1}{K}\sum_{i=1}^{K}\boldsymbol{\pi}^{(i)}(k)\\ \boldsymbol{\pi}^{(i)}(k+1) &= \bar{\boldsymbol{\pi}}(k) - \alpha(k)\ \nabla_{\boldsymbol{\pi}}\mathbb{E}_{\mathbb{P}_{\boldsymbol{\gamma}}}[\mathcal{J}_i(\boldsymbol{\gamma})]\big|_{\bar{\boldsymbol{\pi}}(k)}\end{aligned} \tag{4.9}$$

where $\boldsymbol{\pi}^{(i)}$ represents agent i's local estimate of the common $\boldsymbol{\pi}$ vector and

$$\left.\frac{\partial\mathbb{E}_{\mathbb{P}_{\boldsymbol{\gamma}}}[\mathcal{J}_i(\boldsymbol{\gamma})]}{\partial\pi_j}\right|_{\bar{\boldsymbol{\pi}}(k)} = \mathbb{E}_{\bar{\mathbb{P}}^k_{\boldsymbol{\gamma}}}[\mathcal{J}_i(\boldsymbol{\gamma})|\gamma_j = 1] - \mathbb{E}_{\bar{\mathbb{P}}^k_{\boldsymbol{\gamma}}}[\mathcal{J}_i(\boldsymbol{\gamma})|\gamma_j = 0], \tag{4.10}$$

with $\mathcal{J}_i(\boldsymbol{s})$ defined as in (4.3) with $\mathcal{L}_i(\boldsymbol{\vartheta})$ in place of $\mathcal{L}(\boldsymbol{\vartheta})$ and $\bar{\mathbb{P}}^k_{\boldsymbol{\gamma}}$ given by (4.4) when $\boldsymbol{\pi} = \bar{\boldsymbol{\pi}}(k)$. Notice that the method allows each agent to locally compute $\nabla_{\boldsymbol{\pi}}\mathbb{E}_{\mathbb{P}_{\boldsymbol{\gamma}}}[\mathcal{J}_i(\boldsymbol{\gamma})]\big|_{\bar{\boldsymbol{\pi}}(k)}$ based on its own dataset only. Briefly, at each step the local estimates $\boldsymbol{\pi}^{(i)}(k)$ are averaged and each agent performs an iteration of rule (4.7) starting from the common average $\bar{\boldsymbol{\pi}}(k)$.

[1] Differently from [36], we adopt the penalty term $C\|\boldsymbol{\vartheta}\|_0 N$ in (4.2) instead of a statistical test to prune redundant regressors from the model, which allows us to interpret the RIPs update rule in (3.12) as an exact gradient-based update rule (see Section 3.3 for further details).

Unfortunately, this intuitively simple strategy presents some drawbacks. Indeed, if we compute the average of $\boldsymbol{\pi}^{(1)}(k+1), \dots, \boldsymbol{\pi}^{(K)}(k+1)$, we obtain

$$\begin{aligned}
\bar{\boldsymbol{\pi}}(k+1) &= \frac{1}{K}\sum_{i=1}^{K} \boldsymbol{\pi}^{(i)}(k+1) \\
&= \frac{1}{K}\sum_{i=1}^{K} \bar{\boldsymbol{\pi}}(k) - \alpha(k)\frac{1}{K}\sum_{i=1}^{K} \nabla_{\boldsymbol{\pi}} \mathbb{E}_{\mathbb{P}_{\boldsymbol{\gamma}}}[\mathcal{J}_i(\boldsymbol{\gamma})]\big|_{\bar{\boldsymbol{\pi}}(k)} \\
&= \bar{\boldsymbol{\pi}}(k) - \frac{\alpha(k)}{K}\, \nabla_{\boldsymbol{\pi}} \left[\sum_{i=1}^{K} \mathbb{E}_{\mathbb{P}_{\boldsymbol{\gamma}}}[\mathcal{J}_i(\boldsymbol{\gamma})]\right]\Bigg|_{\bar{\boldsymbol{\pi}}(k)} \\
&= \bar{\boldsymbol{\pi}}(k) - \frac{\alpha(k)}{K}\, \nabla_{\boldsymbol{\pi}} \mathbb{E}_{\mathbb{P}_{\boldsymbol{\gamma}}} \left[\sum_{i=1}^{K} \mathcal{J}_i(\boldsymbol{\gamma})\right]\Bigg|_{\bar{\boldsymbol{\pi}}(k)},
\end{aligned}$$

which means that algorithm (4.9) is minimizing the cost function $\mathbb{E}_{\mathbb{P}_{\boldsymbol{\gamma}}}\left[\sum_{i=1}^{K} \mathcal{J}_i(\boldsymbol{\gamma})\right]$, rather than $\mathbb{E}_{\mathbb{P}_{\boldsymbol{\gamma}}}[\mathcal{J}(\boldsymbol{\gamma})]$ as in (4.5). The two cost functions can actually be different, considering that for all $\boldsymbol{s} \in \Sigma$ the following holds:

$$\sum_{i=1}^{K} \mathcal{J}_i(\boldsymbol{s}) = \sum_{i=1}^{K} \min_{\boldsymbol{\vartheta}^{(i)} \in \Theta_{\boldsymbol{s}}} \mathcal{L}_i(\boldsymbol{\vartheta}^{(i)}) \leq \min_{\boldsymbol{\vartheta} \in \Theta_{\boldsymbol{s}}} \sum_{i=1}^{K} \mathcal{L}_i(\boldsymbol{\vartheta}) = \mathcal{J}(\boldsymbol{s}), \tag{4.11}$$

where we used $\boldsymbol{\vartheta}^{(i)}$ with superscript i in the left hand side of (4.11) to emphasize the fact that the optimal parameterizations might be different from one agent to another.

Albeit showing that minimizing $\mathbb{E}_{\mathbb{P}_{\boldsymbol{\gamma}}}\left[\sum_{i=1}^{K} \mathcal{J}_i(\boldsymbol{\gamma})\right]$ does not necessarily go into the direction of minimizing $\mathbb{E}_{\mathbb{P}_{\boldsymbol{\gamma}}}[\mathcal{J}(\boldsymbol{\gamma})]$, (4.11) provides us with a precious intuition on how to modify algorithm (4.9) to correctly distribute (4.7). Indeed, if we could force all agents to agree on a common $\boldsymbol{\vartheta}$ while evaluating $\mathcal{J}_i(\boldsymbol{s})$ for any $\boldsymbol{s}$, then we could turn (4.11) into an equality and correct the method to make it minimize $\mathbb{E}_{\mathbb{P}_{\boldsymbol{\gamma}}}[\mathcal{J}(\boldsymbol{\gamma})]$.

4.4 A distributed scheme

Inspired by the proximal algorithm in [79], we propose to correct (4.9) modifying how the agents assess the performance of a generic model structure $\boldsymbol{s}$, and specifically replacing $\mathcal{J}_i(\boldsymbol{s})$ in (4.9) with

$$\mathcal{J}_{i,k}(\boldsymbol{s}) = \min_{\boldsymbol{\vartheta}^{(i)} \in \Theta_{\boldsymbol{s}}} \mathcal{L}_i(\boldsymbol{\vartheta}^{(i)}) + \frac{\rho(k)}{2}\|\boldsymbol{\vartheta}^{(i)} - \bar{\boldsymbol{\vartheta}}(k)\|_2^2, \tag{4.12}$$

where the additional proximal term $\|\boldsymbol{\vartheta}^{(i)} - \bar{\boldsymbol{\vartheta}}(k)\|_2^2$ [95] penalizes the distance of the parameter estimate $\boldsymbol{\vartheta}^{(i)}$ computed by agent i from the average parameter vector $\bar{\boldsymbol{\vartheta}}(k)$ and $\rho(k) > 0$ tunes the trade-off between the agent's local cost $\mathcal{L}_i(\boldsymbol{\vartheta}^{(i)})$ and the disagreement among the agents.

The calculation of quantity $\bar{\boldsymbol{\vartheta}}(k)$ requires some further explanation. This quantity represents a "common" parameter vector among the agents, obtained, *e.g.*, by averaging the current best parameterizations of the agents. However, there is not an obvious characterization of such best local parametrization, since each agent is endowed

with a probability distribution over the model collection Σ, as opposed to a specific model structure. To calculate $\bar{\boldsymbol{\vartheta}}(k)$ we therefore associate first to each agent the single model structure $\hat{\boldsymbol{s}}^{(i)}(k)$ that currently has the highest probability of being the optimal one, according to the local probability distribution. Then, we find for each agent the parametrization $\hat{\boldsymbol{\vartheta}}^{(i)}(k)$ minimizing (4.12) with $\boldsymbol{s} = \hat{\boldsymbol{s}}^{(i)}(k)$. Finally, we average the minimizers $\hat{\boldsymbol{\vartheta}}^{(i)}(k)$, $i = 1, \dots, K$ to get $\bar{\boldsymbol{\vartheta}}(k)$.

Accordingly, the overall proposed method is as follows:

$$\begin{aligned}
\bar{\boldsymbol{\vartheta}}(k) &= \frac{1}{K}\sum_{i=1}^{K} \hat{\boldsymbol{\vartheta}}^{(i)}(k) \\
\bar{\boldsymbol{\pi}}(k) &= \frac{1}{K}\sum_{i=1}^{K} \boldsymbol{\pi}^{i}(k) \\
\boldsymbol{\pi}^{(i)}(k+1) &= \bar{\boldsymbol{\pi}}(k) - \alpha(k)\, \nabla_{\boldsymbol{\pi}} \mathbb{E}_{\mathbb{P}_{\boldsymbol{\gamma}}}[\mathcal{J}_i(\boldsymbol{\gamma})]\big|_{\bar{\boldsymbol{\pi}}(k)} \\
\hat{\boldsymbol{s}}^{(i)}(k+1) &= \arg\max_{\boldsymbol{s}\in\Sigma} \mathbb{P}^{(i)}_{\boldsymbol{\gamma}}(k+1) \\
\hat{\boldsymbol{\vartheta}}^{(i)}(k+1) &= \arg \min_{\boldsymbol{\vartheta}^{(i)} \in \Theta_{\hat{\boldsymbol{s}}^{(i)}(k+1)}} \mathcal{L}_i(\boldsymbol{\vartheta}^{(i)}) + \frac{\rho(k)}{2}\|\boldsymbol{\vartheta}^{(i)} - \bar{\boldsymbol{\vartheta}}(k)\|_2^2,
\end{aligned} \tag{4.13}$$

where a saturation is applied to ensure $\pi_j^{(i)}(k+1) \in [0,1]$, $j = 1, \dots, n$.

Intuitively, if we increase $\rho(k)$ at a proper rate, we can push the agents towards the (common) $\bar{\boldsymbol{\vartheta}}(k)$ while they keep optimizing their own local objective functions. Once the $\hat{\boldsymbol{\vartheta}}^{i}(k)$ are sufficiently close to each other, then it holds that

$$\sum_{i=1}^{K} \mathcal{J}_i(\boldsymbol{s}) \approx \mathcal{J}(\boldsymbol{s}),$$

which implies that the method is actually minimizing $\mathbb{E}_{\mathbb{P}_{\boldsymbol{\gamma}}}[\mathcal{J}(\boldsymbol{\gamma})]$.

Algorithm 4 reports the whole identification procedure.

4.5 Numerical examples

In the following we considered two different scenarios, where either all the agents have access to homogenously obtained data (*i.e.*, data resulting from experiments in similar conditions, with equal input and noise signal characterizations), or one of them has data from a different type of experiment on the unknown system.

Model selection was carried out over a candidate regressor set including all monomials with lags not larger than $n_y = n_u = 3$ and maximum degree $n_d = 3$, amounting to $n = 84$ terms overall. The initial $\pi_j^{(i)}$, $\forall j, i$, were set to $\pi_j^{(i)}(0) = 1/n$. The stepsize α in (4.13) is a decreasing function of time according to the following rule [26]:

$$\alpha(k) = \beta/\sqrt{k},\ \beta > 0. \tag{4.14}$$

The C (see (4.2)) and β values have been suitably selected for each studied NARX system by performing a small sensitivity analysis for several pairs (C, β) w.r.t. the correctness of the MSS and the number of iterations required to reach consensus.

Algorithm 3 Local computation - i-th agent

Require: $n, N_p, \pi_{min}, \pi_{max}, C, \rho, \alpha, \bar{\boldsymbol{\pi}}, \bar{\boldsymbol{\vartheta}}$
Ensure: $\boldsymbol{\pi}^{(i)}, \hat{\boldsymbol{\vartheta}}^{(i)}$
Data: $\mathcal{D}_i = \{(\boldsymbol{\varphi}^{(i)}(t), \boldsymbol{y}^{(i)}(t)), t = 1, \cdots, N_i\}$

$\sigma_i^2 \leftarrow \frac{\sum_{t=1}^{N_i} \left(y^{(i)}(t)\right)^2}{N_i}$;
for $p = 1$ **to** N_p **do**
  $\boldsymbol{\Psi(t)} \leftarrow [\,]$;
  $\boldsymbol{s}^{(p)} \leftarrow [\,]$;
  $\bar{\boldsymbol{\vartheta}}^{(p)} \leftarrow [\,]$;
  $m \leftarrow 0$;
  for $j = 1$ **to** n **do**
    Extract r_j from Bernoullian($\bar{\pi}_j$); ▷ Generate terms according to $\bar{\boldsymbol{\pi}}$
    $\boldsymbol{s} \leftarrow [\boldsymbol{s},\ r_j]$
    if $r_j = 1$ **then**
      $\boldsymbol{\Psi(t)} \leftarrow [\boldsymbol{\Psi(t)},\ \varphi_j^{(i)}(t)]$;
      $\bar{\boldsymbol{\vartheta}}^{(p)} \leftarrow [\bar{\boldsymbol{\vartheta}}^{(p)}, \bar{\boldsymbol{\vartheta}}_j]$;
      $m \leftarrow m + 1$;
    end if
  end for
  $\boldsymbol{\vartheta}^{(i)} \leftarrow \left(\sum_{t=1}^{N_i} \boldsymbol{\Psi(t)}^T \boldsymbol{\Psi(t)} + \rho \mathbf{1}_{m\times m}\right)^{-1} \left(\sum_{t=1}^{N_i} \boldsymbol{\Psi(t)}^T y^{(i)}(t) + \rho \bar{\boldsymbol{\vartheta}}^{(p)}\right)$;
  ▷ LS estimate with $\boldsymbol{\vartheta}$ consensus
  $\mathcal{J}_i^{(p)} \leftarrow \frac{1}{\sigma_i^2} \sum_{t=1}^{N_i} \left(y^{(i)}(t) - \boldsymbol{\Psi(t)} \boldsymbol{\vartheta}^{(i)}\right)^2 + C\|\boldsymbol{\vartheta}^{(i)}\|_0 N_i$; ▷ Model evaluation
end for
for $j = 1$ **to** n **do** ▷ Update π_j
  $\mathcal{J}^{\oplus} \leftarrow \infty$; $n^{\oplus} \leftarrow 0$; $\mathcal{J}^{\ominus} \leftarrow \infty$; $n^{\ominus} \leftarrow 0$;
  for $p = 1$ **to** N_p **do**
    if $s_j^{(p)} = 1$ **then**
      $\mathcal{J}^{\oplus} \leftarrow \mathcal{J}^{\oplus} + \mathcal{J}_i^{(p)}$; $n^{\oplus} \leftarrow n^{\oplus} + 1$;
    else
      $\mathcal{J}^{\ominus} \leftarrow \mathcal{J}^{\ominus} + \mathcal{J}_i^{(p)}$; $n^{\ominus} \leftarrow n^{\ominus} + 1$;
    end if
  end for
  $\pi_j^{(i)} \leftarrow \bar{\pi}_j - \alpha \left(\frac{\mathcal{J}^{\oplus}}{\max(n^{\oplus},1)} - \frac{\mathcal{J}^{\ominus}}{\max(n^{\ominus},1)}\right)$;
  $\pi_j^{(i)} \leftarrow \max\left(\min\left(\pi_j^{(i)}, \pi_{max}\right), \pi_{min}\right)$; ▷ Saturation
end for
$\hat{\boldsymbol{s}}^{(i)} \leftarrow \lfloor \boldsymbol{\pi}^{(i)} \rceil$; ▷ Compute the most probable $\boldsymbol{s}$ according to $\boldsymbol{\pi}^{(i)}$
$\bar{\boldsymbol{\vartheta}}^{(i)} \leftarrow [\,]$;
$m \leftarrow 0$;
for $j = 1$ **to** n **do**
  if $\hat{s}_j^{(i)} = 1$ **then**
    $\boldsymbol{\Psi(t)} \leftarrow [\boldsymbol{\Psi(t)},\ \varphi_j^{(i)}(t)]$;
    $\bar{\boldsymbol{\vartheta}}^{(i)} \leftarrow [\bar{\boldsymbol{\vartheta}}^{(i)}, \bar{\boldsymbol{\vartheta}}_j]$;
    $m \leftarrow m + 1$;
  end if
end for
$\hat{\boldsymbol{\vartheta}}^{(i)} \leftarrow \left(\sum_{t=1}^{N_i} \boldsymbol{\Psi(t)}^T \boldsymbol{\Psi(t)} + \rho \mathbf{1}_{m\times m}\right)^{-1} \left(\sum_{t=1}^{N_i} \boldsymbol{\Psi(t)}^T y^{(i)}(t) + \rho \bar{\boldsymbol{\vartheta}}^{(i)}\right)$;

Algorithm 4 NARX identification - distributed scheme

Require: $n, K, N_p, \pi_{min}, \pi_{max}, C, \rho, \alpha$
Ensure: $\bar{\boldsymbol{\pi}}, \bar{\boldsymbol{\vartheta}}$

$\boldsymbol{\pi}^{(i)} \leftarrow \frac{1}{n} \cdot \mathbf{1}_{n\times 1}$;
$\hat{\boldsymbol{\vartheta}}^{(i)} \leftarrow \mathbf{0}_{n\times 1}$;
repeat
  $\bar{\boldsymbol{\vartheta}} = \frac{1}{K}\sum_{i=1}^{K} \hat{\boldsymbol{\vartheta}}^{(i)}$;
  $\bar{\boldsymbol{\pi}} \leftarrow \frac{1}{K}\sum_{i=1}^{K} \boldsymbol{\pi}^{(i)}$;
  parfor $i = 1$ **to** K **do** ▷ Gather information from agents
    $\{\boldsymbol{\pi}^{(i)},\ \hat{\boldsymbol{\vartheta}}^{(i)}\} \leftarrow$ run Algorithm 3;
  end parfor
until Stopping criterion

A Monte Carlo approach has been employed to approximate the expected values appearing in the proposed algorithm (4.13), with their sampled counterparts. More precisely, at each iteration each agent draws $N_p = 1000$ sample model structures from $\mathbb{P}_{\boldsymbol{\gamma}}$, evaluates them in terms of the $\mathcal{J}_{i,k}(\boldsymbol{s})$, and calculates the appropriate sampled averages.

To account for the randomization inherent in the algorithm, a Monte Carlo analysis has been carried out in all the experiments, by running the algorithm 100 times on the same data sets.

All the tests have been performed in MATLAB 2017a environment, on an HP ProBook 650 G1 CORE i7-4702MQ CPU @2.20 GHz with 8GB of RAM.

4.5.1 Experiment 1

We considered the following benchmark systems taken from the literature [1,7,22,116]:

$$S_1\text{: } y(t) = -1.7y(t-1) - 0.8y(t-2) + u(t-1) + 0.8u(t-2) + e(t),$$
with $u(t) \sim WUN(-2, 2)$, $e(t) \sim WGN(0, 0.01)$

$$S_2\text{: } y(t) = 0.7y(t-1)u(t-1) - 0.5y(t-2) - 0.7y(t-2)u(t-2)^2 + 0.6u(t-2)^2 + e(t),$$
with $u(t) \sim WUN(-1, 1)$, $e(t) \sim WGN(0, 0.04)$

$$S_3\text{: } y(t) = 0.8y(t-1) + 0.4u(t-1) + 0.4u(t-1)^2 + 0.4u(t-1)^3 + e(t),$$
with $u(t) \sim WGN(0, 0.333)$, $e(t) \sim WGN(0, 0.1)$

$$S_4\text{: } y(t) = 0.25u(t-1) + 0.75y(t-2) - 0.2y(t-2)u(t-1) + e(t),$$
with $u(t) \sim WGN(0, 0.25)$, $e(t) \sim WGN(0, 0.02)$

where $WGN(\eta, \sigma^2)$ is a White Gaussian Noise with mean η and variance σ^2, while $WUN(a, b)$ denotes a White Uniform Noise defined in the interval $[a, b]$. The employed C values are: $C_{S_1} = C_{S_2} = C_{S_3} = 0.01$, $C_{S_4} = 0.001$. $\beta_{S_1} = \beta_{S_2} = \beta_{S_3} = 0.01$, $\beta_{S_4} = 0.1$, in (4.14), see Table 4.1. We adopted an increasing factor $\rho(k) = 2k$ in (4.13), with

k being the iteration index. Four data sets of length 2000 have been generated, one for each agent.

The proposed algorithm (4.13) has been applied to this case, and the aggregated results are reported in Table 4.2, where each cell reports the average value of the corresponding parameter. Specifically, the following statistics have been considered: the correctness of the structure selection, the elapsed time and number of iterations required to reach consensus, and the mean square error (MSE) of the parameter estimates. Table 4.3 displays the average parameter estimates.

The proposed algorithm performed well in all cases both regarding the model structure selection and the estimation of the parameters.

Table 4.1: *Experiment 1: sensitivity analysis for parameters C and β. For each pair (C, β), with $C \in [0.001, 0.01, 0.1, 1, 10]$ and $\beta \in [0.01, 0.1, 1, 10]$, an MC analysis has been carried out in all the experiments, by running the algorithm 20 times on the same data sets of 2000 samples. Each cell reports the average value and the standard deviation (over the MC runs) of $\frac{1}{K}\sum_{i=1}^{K} MSE_i$, where MSE_i is the mean square prediction error computed on data set $\mathcal{D}_i$ by employing the identified model $\bar{\boldsymbol{\vartheta}}$. The average number of iterations required to reach consensus is also reported.*

		Mean squared error $\pm$ standard deviation (# of iterations)			
		β^1	β^2	β^3	β^4
S_1	C^1	0.0098 $\pm$ 4.6E-4 (839.8)	0.0098 $\pm$ 4.6E-4 (49.9)	0.0110 $\pm$ 4.8E-4 (17.9)	0.0110 $\pm$ 4.8E-4 (18.2)
	C^2	0.0098 $\pm$ 4.6E-4 (42.6)	0.0179 $\pm$ 8.0E-4 (17.4)	0.0354 $\pm$ 1.5E-3 (19.2)	0.0432 $\pm$ 1.7E-3 (20.8)
	C^3	0.3180 $\pm$ 7.6E-3 (16.7)	0.2754 $\pm$ 7.1E-3 (17.5)	0.2523 $\pm$ 6.9E-3 (18.1)	0.3173 $\pm$ 7.7E-3 (19.9)
	C^4	1.7291 $\pm$ 4.7E-2 (112.8)	2.0507 $\pm$ 4.7E-2 (173.1)	2.1634 $\pm$ 5.4E-2 (206.4)	1.8788 $\pm$ 5.5E-2 (157.9)
	C^5	3.0149 $\pm$ 5.6E-2 (870.4)	2.9045 $\pm$ 6.6E-2 (568.5)	3.3639 $\pm$ 8.5E-2 (503.9)	3.0400 $\pm$ 7.5E-2 (578.3)
S_2	C^1	0.0039 $\pm$ 1.8E-4 (816)	0.0039 $\pm$ 1.8E-4 (45.9)	0.0053 $\pm$ 2.1E-4 (15.8)	0.0053 $\pm$ 2.1E-4 (17.2)
	C^2	0.0039 $\pm$ 1.8E-4 (34.7)	0.0046 $\pm$ 2.2E-4 (14.5)	0.0078 $\pm$ 2.2E-4 (14.8)	0.0081 $\pm$ 2.1E-4 (19.1)
	C^3	0.0126 $\pm$ 1.6E-3 (13.1)	0.0126 $\pm$ 1.6E-3 (13.9)	0.0126 $\pm$ 1.6E-3 (13.9)	0.0144 $\pm$ 1.9E-3 (14.1)
	C^4	0.1204 $\pm$ 2.2E-2 (795.7)	0.1184 $\pm$ 2.02E-2 (480.7)	0.1166 $\pm$ 2.1E-2 (387.5)	0.1229 $\pm$ 2.2E-2 (561.5)
	C^5	0.1334 $\pm$ 2.4E-2 (928)	0.1342 $\pm$ 2.4E-2 (833)	0.1259 $\pm$ 2.1E-2 (769.8)	0.1270 $\pm$ 2.2E-2 (443.1)
S_3	C^1	0.0982 $\pm$ 5.1E-3 (878)	0.0982 $\pm$ 5.1E-3 (48.8)	0.1120 $\pm$ 5.7E-3 (17.3)	0.1120 $\pm$ 5.6E-3 (20.4)
	C^2	0.0982 $\pm$ 5.1E-3 (42.3)	0.1115 $\pm$ 6.3E-3 (17.1)	0.1320 $\pm$ 6.1E-3 (17.4)	0.1320 $\pm$ 6.7E-3 (17.9)
	C^3	0.1652 $\pm$ 1.2E-2 (14)	0.1815 $\pm$ 1.6E-2 (17.4)	0.1768 $\pm$ 1.6E-2 (18)	0.1681 $\pm$ 1.3E-2 (17.5)
	C^4	0.4079 $\pm$ 4.6E-2 (101.9)	0.5619 $\pm$ 8.0E-2 (201.1)	0.4912 $\pm$ 6.0E-2 (117.4)	0.4988 $\pm$ 6.4E-2 (64.6)
	C^5	1.2759 $\pm$ 2.1E-1 (922.3)	1.0511 $\pm$ 1.6E-1 (669.9)	1.1390 $\pm$ 1.8E-1 (915.9)	1.0743 $\pm$ 2.3E-1 (630.6)
S_4	C^1	0.0196 $\pm$ 1.0E-3 (922.4)	0.0196 $\pm$ 1.0E-3 (46.6)	0.0201 $\pm$ 1.2E-3 (17.8)	0.0201 $\pm$ 1.2E-3 (21.3)
	C^2	0.0203 $\pm$ 1.2E-3 (42.6)	0.0203 $\pm$ 1.2E-3 (16)	0.0203 $\pm$ 1.2E-3 (17)	0.0203 $\pm$ 1.2E-3 (17.2)
	C^3	0.0203 $\pm$ 1.2E-3 (12.5)	0.0216 $\pm$ 1.4E-3 (13.8)	0.0210 $\pm$ 1.3E-3 (16.5)	0.0216 $\pm$ 1.4E-3 (15.2)
	C^4	0.0483 $\pm$ 3.0E-3 (212.1)	0.0571 $\pm$ 3.5E-3 (455.4)	0.0648 $\pm$ 4.1E-3 (522.8)	0.0598 $\pm$ 3.7E-3 (395.3)
	C^5	0.0734 $\pm$ 4.5E-3 (810.5)	0.0727 $\pm$ 4.5E-3 (663.6)	0.0766 $\pm$ 4.9E-3 (681.1)	0.0814 $\pm$ 5.0E-3 (682)

4.5.2 Experiment 2

In this experiment, we considered the following system:

$$S_5\textbf{:}\ y(t) = 0.5y(t-1) + 0.8u(t-2) + 0.1u(t-1)^2 + e(t),\ e(t) \sim WGN(0, 0.01).$$

Table 4.2: *Experiment 1: average statistics.*

	S_1	S_2	S_3	S_4
Correct selection	100%	100%	100%	100%
# of Iterations	45.9	33.6	42.3	45.8
Elapsed Time [sec]	27.8	12.3	24.7	24.9
MSE on parameter estimate	3.9E-7	1.3E-5	2.2E-5	2.3E-4

Table 4.3: *Experiment 1: average parameter estimates.*

S_1	True	-1.7	-0.8	1	0.8
	Estimated	-1.6993	-0.8002	0.9999	0.7990
S_2	True	0.7	-0.5	-0.7	0.6
	Estimated	0.6987	-0.4977	-0.7066	0.6007
S_3	True	0.8	0.4	0.4	0.4
	Estimated	0.7975	0.4069	0.4057	0.3983
S_4	True	0.25	0.75	-0.2	
	Estimated	0.2528	0.7414	-0.1751	

Again, 4 data sets $\mathcal{D}_i$, $i = 1, \ldots, 4$ were collected, of length 5000 each, but this time the data are originated from different experimental set-ups. Specifically, in the first 3 experiments, $u(t) \sim WGN(0, 0.01)$, while in the last, $u(t) \sim WGN(0, 1)$.

The peculiarity of this example lies in the impossibility to identify the full model structure based on the data sets $\mathcal{D}_1$, $\mathcal{D}_2$, or $\mathcal{D}_3$, since the input amplitude is insufficient to excite the nonlinear dynamics in the model. On the other hand, the nonlinear dynamics is fully excited when $u(t) \sim WGN(0, 1)$ is employed (data set $\mathcal{D}_4$). To see this, check Table 4.4, which reports the first eight terms selected by the OFR method, applied separately to each data set. The model terms are listed in the same order as they have been selected, which reflects their importance in the model. The terms in bold represent the final model structure, as selected according to the BIC criterion. Apparently, while the correct model structure is identified for $\mathcal{D}_4$ (albeit with a redundant term), only the linear sub-model is correctly selected in the other three cases, and the nonlinear missing term is not even among those suggested by the OFR immediately after the two correct regressors.

The explained difficulty of this example causes the failure of approaches such as that explained in [73] (denoted as PRESS-based OFR), as documented in Table 4.5, where the final model structure has been selected according to the Average-BIC criterion defined in [73]. Again, only the linear sub-part of the model has been correctly identified, while the nonlinear term has been masked by the prevailing linear data, and has been selected only as fourth term, to be rejected by the Average-BIC criterion.

Table 4.6 displays the aggregated results obtained by running our algorithm. The design parameters were set to $\lambda = 0.005$ and $\beta = 0.01$. Apparently, the algorithm proposed in the paper fruitfully combines the information gathered from all the data

Table 4.4: *Experiment 2: Model structure selection results with the OFR method on different data sets.*

Sel. order	$\mathcal{D}_1$ only	$\mathcal{D}_2$ only	$\mathcal{D}_3$ only	$\mathcal{D}_4$ only
1	$\boldsymbol{u(t-2)}$	$\boldsymbol{u(t-2)}$	$\boldsymbol{u(t-2)}$	$\boldsymbol{u(t-2)}$
2	$\boldsymbol{y(t-1)}$	$\boldsymbol{y(t-1)}$	$\boldsymbol{y(t-1)}$	$\boldsymbol{y(t-1)}$
3	$y(t-2)y(t-3)$	$y(t-3)$	$y(t-2)u(t-2)u(t-3)$	$\boldsymbol{u(t-1)^2}$
4	$y(t-3)u(t-3)$	$u(t-2)u(t-3)$	$u(t-2)^2u(t-3)$	$\boldsymbol{y(t-3)u(t-2)u(t-3)}$
5	$u(t-3)$	$y(t-3)u(t-2)u(t-3)$	$y(t-2)u(t-1)^2$	$y(t-1)u(t-2)^2$
6	$y(t-1)y(t-3)$	$u(t-1)^2u(t-2)$	$u(t-1)^2u(t-2)$	$u(t-1)^2u(t-2)$
7	$y(t-1)u(t-1)u(t-3)$	$y(t-1)u(t-1)$	$y(t-3)u(t-1)u(t-2)$	$y(t-3)u(t-1)$
8	$u(t-1)^2$	$u(t-1)u(t-3)$	$u(t-2)u(t-3)$	$u(t-1)^2u(t-3)$

Table 4.5: *Experiment 2: Model identification results with the PRESS-based OFR.*

Sel. order	Model Term	Par. estimate
1	$\boldsymbol{u(t-2)}$	0.794143
2	$\boldsymbol{y(t-1)}$	0.508702
3	$y(t-2)u(t-2)u(t-3)$	–
4	$u(t-1)^2$	–

sets, ultimately leading to the identification of the correct model structure and accurate parameter estimates.

Table 4.6: *Experiment 2: average statistics of the proposed approach.*

Correct selection	100%
# of Iterations	222.6
Elapsed Time [sec]	85.5
Selected model terms	$y(t-1),\ u(t-2),\ u(t-1)^2$
Parameter estimates	$0.5050,\ 0.7983,\ 0.0977$
MSE on parameter estimates	1.1E-5

Figure 4.1 shows the evolution for all agents of the parameter associated to the nonlinear term (denoted $\vartheta^i_{u(t-1)^2}$), during a single execution of the algorithm, using $\rho(k) = 2k$ in (4.13). As expected, the 4-th agent immediately recognizes the importance of the nonlinear term and provides a very accurate estimate of the corresponding parameter, while the others are slower, given that their data does not clearly emphasize the nonlinearity. However, they are still able to reach a consensus on the presence of that term and its parameter value, which is very close to the true one.

Figure 4.2 shows the squared prediction error (SPE) values obtained by employing the models identified by the PRESS-based OFR (see Table 4.5) and by the proposed method (see Table 4.6) to perform the one-step-ahead prediction for the validation data sets $\{\tilde{\mathcal{D}}_i\}_{i=1}^4$, which have been collected in the same experimental set-ups described before. As expected, both models provide similar prediction accuracy on $\{\tilde{\mathcal{D}}_i\}_{i=1}^3$,

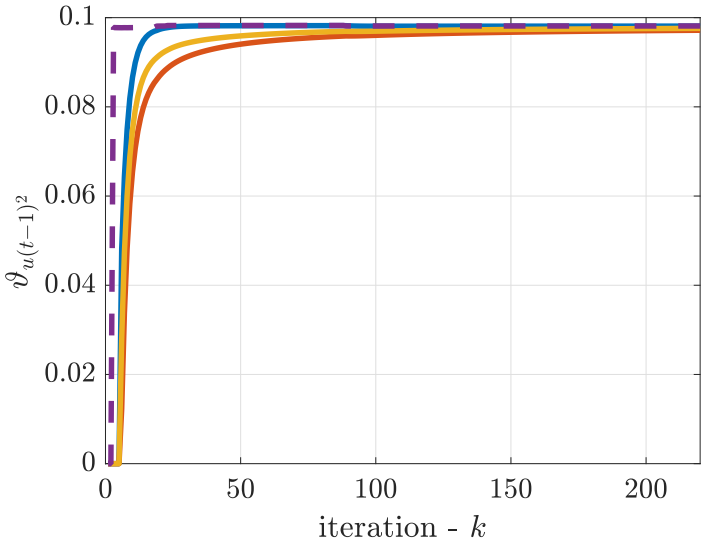

Figure 4.1: *Experiment 2: Nonlinear parameter estimation results with $\rho(k) = 2k$ for the proposed approach. The dashed curve is associated to agent 4.*

while there are significant discrepancies between the predictions obtained for the fourth data set. Specifically, the mean SPE obtained for $\tilde{\mathcal{D}}_4$ with the PRESS-based OFR model is 0.0506 w.r.t. 0.0102 obtained by employing the model resulting from the application of the proposed approach. This difference is even more significant in case of open-loop output simulation of the identified models (see Figure 4.3). Indeed, in this case the PRESS-based OFR model results in a mean squared simulation error (SSE) on $\tilde{\mathcal{D}}_4$ equal to 0.1033 w.r.t. 0.0139.

4.6 A case study

The proposed identification procedure was applied to the data collected from the experimental setup presented in [47]. Data have been collected from 54 Mica2Dot sensors with weather boards deployed in the Intel Berkeley Research lab between February 28-th and April 5-th, 2004. The sensors collected, among all, humidity and temperature values once every 31 seconds, along with timestamped topology information about the sensor network. Data was collected using the TinyDB in-network query processing system, built on the TinyOS platform. The sensors were arranged in the lab according to the diagram shown in Figure 4.4. The data measured in the lab are quite complex: different areas of the lab are exposed to the sun at different times of the day, and the measurements exhibit spatial correlation, but they also include local variations due to the proximity to windows and air conditioning vents. Inspired by [47], we identified 5 regions, in each of which the readings are assumed to be highly correlated (see Table 4.7, noting that some sensors are not considered).

It is reasonable to think of sensors in the same region as devices (agents) which are monitoring the same physical system, and hence trying to model in a distributed way the temperature as a function of the available measurements of the temperature itself and humidity. Accordingly, we ran our distributed scheme on each region separately,

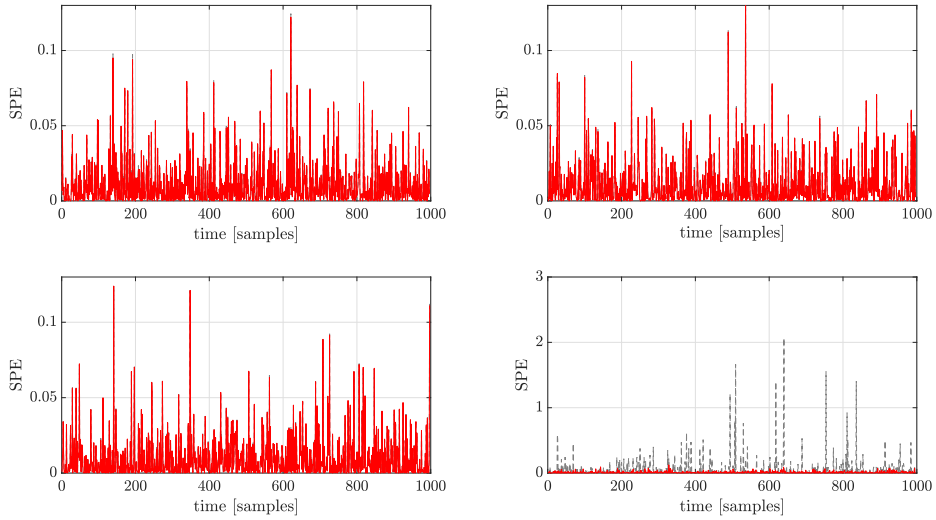

Figure 4.2: *Experiment 2: squared prediction error (SPE) values obtained by employing the models identified by the PRESS-based OFR (see Table 4.5) and by the proposed method (see Table 4.6) to perform the one-step-ahead prediction for the validation data sets* $\{\tilde{\mathcal{D}}_i\}_{i=1}^4$ *(solid line: proposed approach, dashed: PRESS-based OFR). From top to bottom, left to right: data sets* $\{\tilde{\mathcal{D}}_i\}_{i=1}^4$

.

Table 4.7: *A case study: mapping between sensors and identified regions. Note that some sensors are not considered here.*

Region	Sensor identifier
1	$\{44, 45, 46, 47, 48, 49, 50\}$
2	$\{33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43\}$
3	$\{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$
4	$\{26, 27, 28, 29, 30, 31, 32\}$
5	$\{15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25\}$

considering each sensor as an agent with its own private data set. Table 4.8 indicates for each region those sensors that have been used for model learning and those for model validation.

We considered data sets of an entire day, the 8-th of March, and we tried to identify a NARX model with $n_d = 2$, and $n_y = n_u = 2$, where n_y and n_u denote, respectively, the maximum lag for the autoregressive component, *i.e.,* the temperature, and for the exogenous input, *i.e.,* humidity. Figure 4.5 reports temperature data obtained from the sensors in Region 1 over a one day period. This proves the presence of significant local variations also in the measurements coming from the same region, especially in the period ranging from 6 AM to 6 PM, and also that the temperature exhibits different behaviors in different times of the day. Accordingly, we applied the identification procedure considering 4 time periods, from midnight to 6 AM, from 6 AM to noon, from noon to 6 PM, and from 6 PM to midnight. Figure 4.6 illustrates the evolution in time

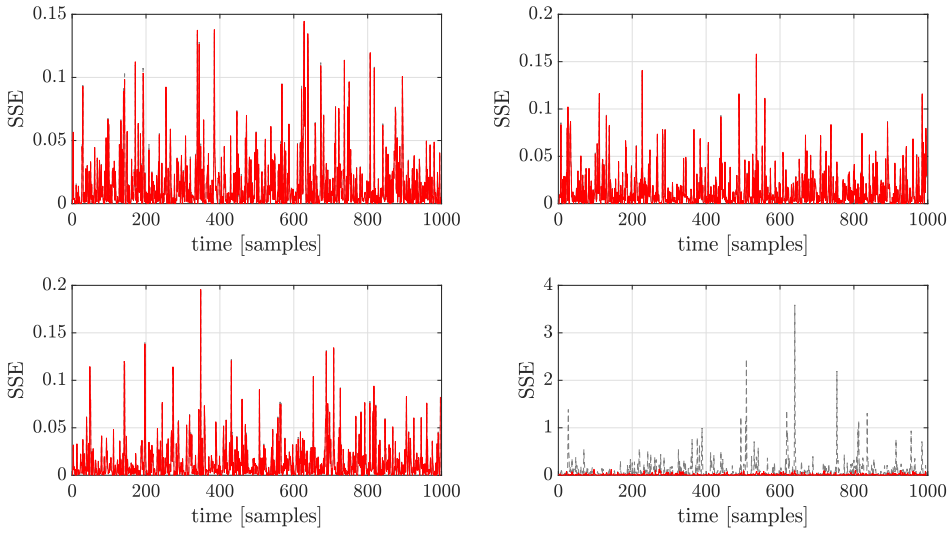

Figure 4.3: *Experiment 2: squared simulation error (SSE) values obtained by employing the models identified by the PRESS-based OFR (see Table 4.5) and by the proposed method (see Table 4.6) to open-loop output simulation for the validation data sets* $\{\tilde{\mathcal{D}}_i\}_{i=1}^4$ *(solid line: proposed approach, dashed: PRESS-based OFR). From top to bottom, left to right: data sets* $\{\tilde{\mathcal{D}}_i\}_{i=1}^4$

.

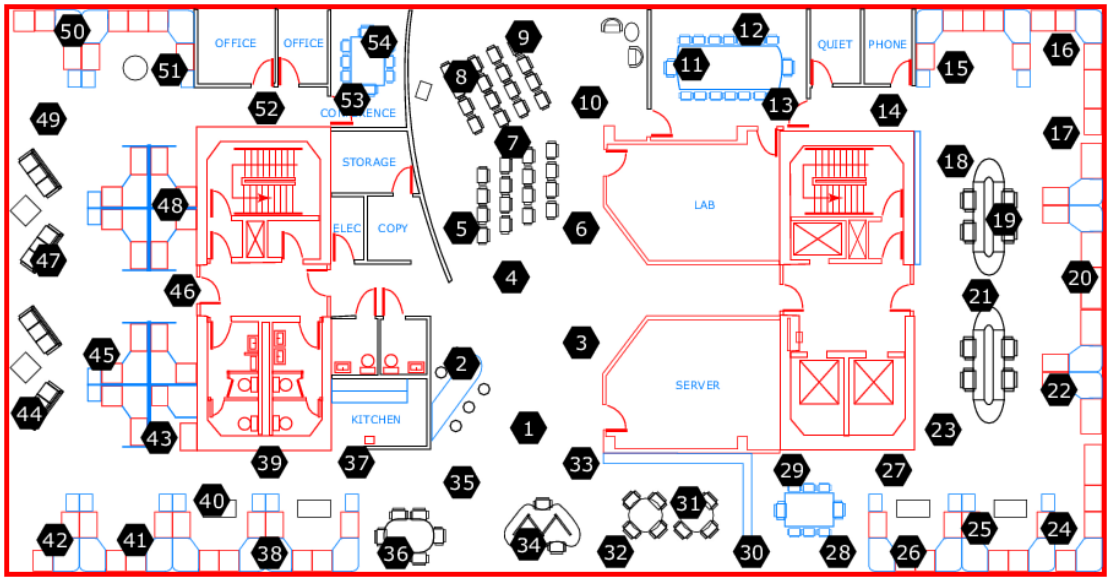

Figure 4.4: *A case study: sensors location at the Intel Berkeley Research lab.*

of temperature data in the remaining regions. Apparently, during the night, *i.e.,* from 6 PM to midnight, and in the early morning, *i.e.,* from midnight to 6 AM, the temperature curves are similar in different regions, while during the working hours there are significant differences between them. This happens also for the humidity, whose evolution in time is shown in Figure 4.7. Each data set was subsampled to a data set with $1/8$-th of the original data before running the identification procedure.

The design parameters were set as follows: $C = 0.001$, $\beta = 1$, $\rho(k) = 2k$, and

Table 4.8: *A case study: sensors that have been used for model learning and those for model validation. Note that some sensors are not used due to errors in the collected data.*

Region	Sensor identifier (model learning)	Sensor identifier (model validation)
1	$\{45, 47, 48, 49\}$	$\{44, 50\}$
2	$\{34, 35, 36, 38, 39, 42\}$	$\{33, 43\}$
3	$\{3, 5, 7, 9\}$	$\{1, 10\}$
4	$\{27, 28, 29, 30, 31\}$	$\{26, 32\}$
5	$\{18, 19, 20, 21, 22, 23, 24\}$	$\{17, 25\}$

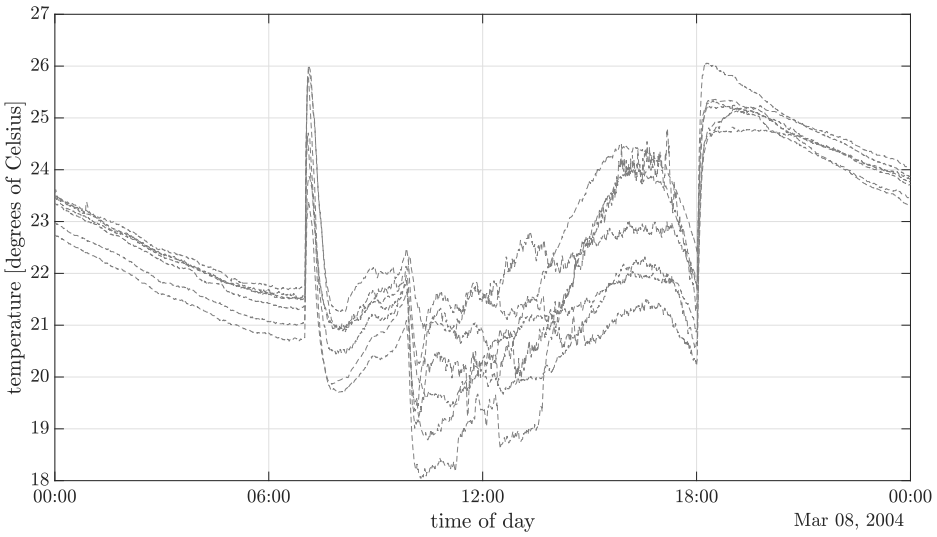

Figure 4.5: *A case study: temperature data obtained from sensors in Region* 1 *over one day period.*

$N_p = 1000$. Data $y(t)$ and $u(t)$ have been standardized to have zero mean and a variance of one. Table 4.9 reports, for each region, the models identified in different time periods and the corresponding averaged (among agents) mean squared error. As expected, models identified from data belonging to the first and fourth time period are similar among regions. Conversely, the models corresponding to the second and third time period, differ significantly among regions. In particular, while simple ARX models are in general enough to adequately approximate the system behavior in the first and fourth time period, more complex models, which include also nonlinear model terms, are needed to model the local temperature variations in the period ranging from 6 AM to 6 PM.

The identified models have been compared with the trivial predictor $\hat{y}(t) = y(t-1)$ which often represents a valid competitor when the system dynamic changes slowly in time as *e.g.,* in case of the temperature, and the obtained results are reported in Table 4.10. Apparently, one can always take advantage in employing the identified models w.r.t. the trivial predictor, as can be easily noted by the reported MSE reduction values in the last column of Table 4.10. This evidence confirms that the considered subsampled ($1/8$-th subsampling factor) data set is a valid benchmark for studying the

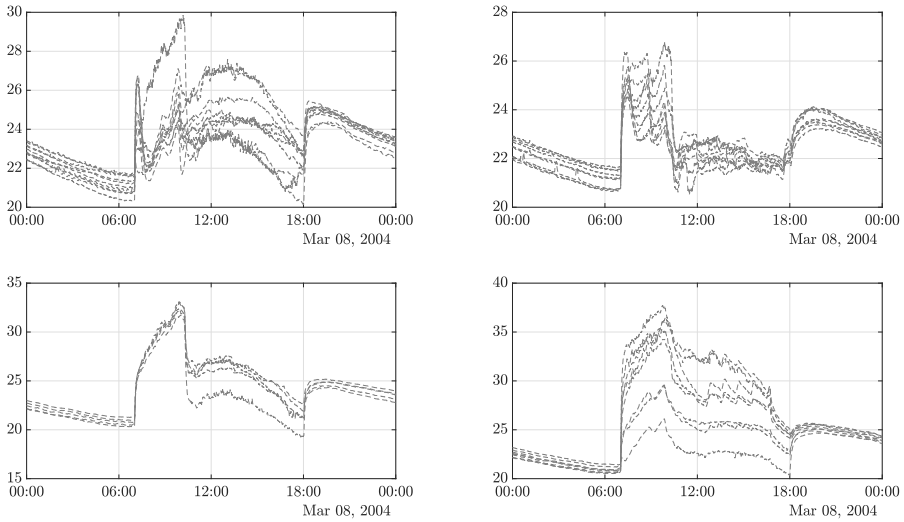

Figure 4.6: *A case study: temperature data obtained from sensors over one day period. From top-left to bottom-right, Region* 2*, Region* 3*, Region* 4*, Region* 5*.*

MSS problem.

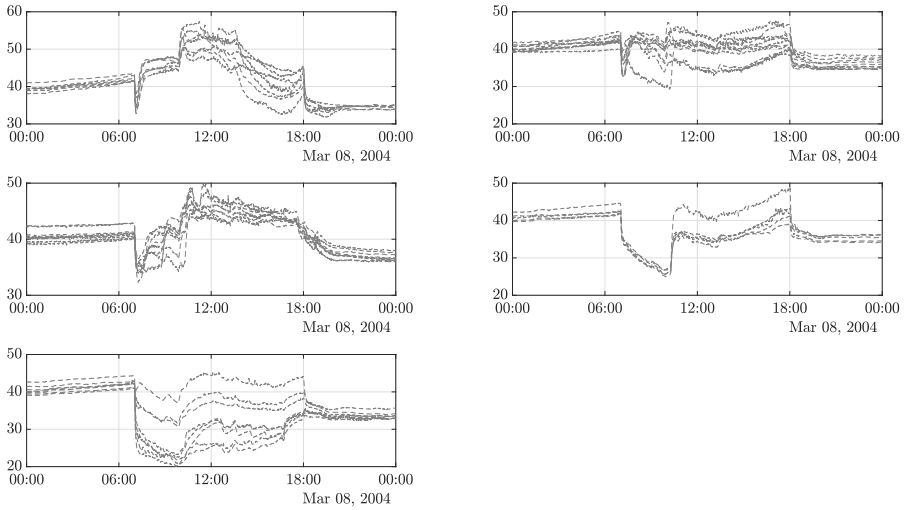

Figure 4.7: *A case study: humidity data obtained from sensors over one day period. On the x-axis is reported the time of day, while in the y-axis is reported the percentage of humidity. From top-left to bottom-right, Region* 1*, Region* 2*, Region* 3*, Region* 4*, Region* 5*.*

The identified models have been validated on data collected from those sensors that have been excluded from the identification process, as reported in Table 4.8. The obtained results are presented in Table 4.11, and it easy to note that in general the identified models generalize well on unseen data, leading to mean squared errors comparable with those resulting from the training phase. However, this is not always the case, such as for sensor 10 in the second and third time period. Nevertheless, this is not surpris-

Table 4.9: *A case study: identified model in each time period, from data collected by sensors in each region. Data have been standardized to have zero mean and a variance of one.*

Region	Period	Identified model	Mean Squared Error
1	$00:00-06:00$	$\hat{y}(t) = -0.0630 + 0.6059y(t-1) + 0.3831y(t-2)$	0.0037
	$06:00-12:00$	$\hat{y}(t) = 1.1494y(t-1) - 0.4428y(t-2) - 0.2215u(t-1)$	0.0910
	$12:00-18:00$	$\hat{y}(t) = 0.0819 + 0.9556y(t-1) - 0.1945y(t-2)^2 - 0.4376y(t-2)u(t-1) - 0.2821u(t-1)^2$	0.0345
	$18:00-00:00$	$\hat{y}(t) = 1.1773y(t-1) - 0.1776y(t-2)$	0.0087
2	$00:00-06:00$	$\hat{y}(t) = -0.0636 + 0.7376y(t-1) + 0.2457y(t-2)$	0.0101
	$06:00-12:00$	$\hat{y}(t) = 0.9989y(t-1) + 0.0816u(t-2)$	0.0746
	$12:00-18:00$	$\hat{y}(t) = -0.0662 + 0.8379y(t-1) + 0.2906y(t-2) + 0.1220u(t-2)$	0.0257
	$18:00-00:00$	$\hat{y}(t) = 0.7753y(t-1) + 0.1970y(t-2) + 0.1929u(t-2)$	0.0363
3	$00:00-06:00$	$\hat{y}(t) = -0.0512 + 0.6915y(t-1) + 0.2915y(t-2)$	0.0117
	$06:00-12:00$	$\hat{y}(t) = 1.2297y(t-1) - 0.3043y(t-2)$	0.0670
	$12:00-18:00$	$\hat{y}(t) = 1.2429y(t-1) - 0.3091u(t-1)$	0.0908
	$18:00-00:00$	$\hat{y}(t) = 0.9424y(t-1) + 0.1547u(t-2)$	0.0251
4	$00:00-06:00$	$\hat{y}(t) = -0.0819 + 0.6006y(t-1) + 0.3687y(t-2)$	0.0080
	$06:00-12:00$	$\hat{y}(t) = 0.7043y(t-1) - 0.2042y(t-2) - 0.4343u(t-1) - 0.2943y(t-2)^2 + 0.3165u(t-2)^2$	0.0301
	$12:00-18:00$	$\hat{y}(t) = -0.0637 + 0.8011y(t-1) + 0.4607y(t-2) + 0.2414u(t-2)$	0.0135
	$18:00-00:00$	$\hat{y}(t) = 0.8157y(t-1) + 0.1304y(t-2) + 0.1850u(t-2)$	0.0522
5	$00:00-06:00$	$\hat{y}(t) = -0.0436 + 0.6974y(t-1) + 0.2942y(t-2)$	0.0112
	$06:00-12:00$	$\hat{y}(t) = 0.9324y(t-1) - 0.2481u(t-1) + 0.2183u(t-2)$	0.0296
	$12:00-18:00$	$\hat{y}(t) = 0.9225y(t-1) + 0.1476y(t-2) + 0.1105u(t-2) - 0.0361y(t-2)^2$	0.0261
	$18:00-00:00$	$\hat{y}(t) = 0.9619y(t-1) + 0.1304u(t-2)$	0.0284

ingly, since the proposed distributed scheme aims to trading-off between the performance of each local model and those of the global model, and it has to facing with the high variations among sensor data, see Figure 4.8. For the sake of completeness, Figures 4.9 and 4.10 show the one-step-ahead prediction for the validation (standardized) data collected by sensors 1 and 10, and sensors 44 and 50 in the second and third period, respectively.

4.7 Conclusions

A novel distributed scheme with model structure selection was proposed for nonlinear system identification using the NARX model representation. The proposed approach relies on the standing assumption that there are multiple data sets collected from several experiments which cannot be made centrally available, and hence the identification problem has to be solved by distributing the computation among agents. We address this issue by resorting to a distributed scheme which aims at reaching a common value for both the model structure and the parameter estimates in an integrated fashion, after having reformulated the MSS problem in the probabilistic framework discussed in Chapter 3. Its performance was evaluated using Monte Carlo simulations over two different scenarios: in the former the data were originated from the same experimental set-ups, while in the latter different input signals have been considered to generate the data. In both cases, the algorithm was capable of retrieving the correct structure and parameterization of the process model, in a computationally efficient way. Furthermore, it was shown that the presented method outperforms an OFR-based competitor in terms of reliability of the structure selection. Finally, the proposed identification procedure

Table 4.10: *A case study: comparison between the identified models and the trivial predictor* $\hat{y}(t) = y(t-1)$. *Data have been standardized to have zero mean and a variance of one.*

Region	Period	Mean Squared Error (identified model)	Mean Squared Error (trivial predictor)	Evaluation
1	$00:00-06:00$	0.0037	0.0064	-42.19%
	$06:00-12:00$	0.0910	0.1330	-31.58%
	$12:00-18:00$	0.0345	0.0396	-12.88
	$18:00-00:00$	0.0087	0.0100	-13%
2	$00:00-06:00$	0.0101	0.0143	-29.37
	$06:00-12:00$	0.0746	0.0813	-8.24%
	$12:00-18:00$	0.0257	0.0344	-25.29
	$18:00-00:00$	0.0363	0.0502	-27.69%
3	$00:00-06:00$	0.0117	0.0136	-13.97%
	$06:00-12:00$	0.0670	0.0831	-19.37%
	$12:00-18:00$	0.0908	0.1114	-18.49%
	$18:00-00:00$	0.0251	0.0490	-48.78%
4	$00:00-06:00$	0.0080	0.0117	-31.62%
	$06:00-12:00$	0.0301	0.0409	-26.41
	$12:00-18:00$	0.0135	0.0172	-21.51%
	$18:00-00:00$	0.0522	0.0738	-29.27%
5	$00:00-06:00$	0.0112	0.0119	-5.88%
	$06:00-12:00$	0.0296	0.0336	-11.90%
	$12:00-18:00$	0.0261	0.0301	-13.29
	$18:00-00:00$	0.0284	0.0476	-40.34%

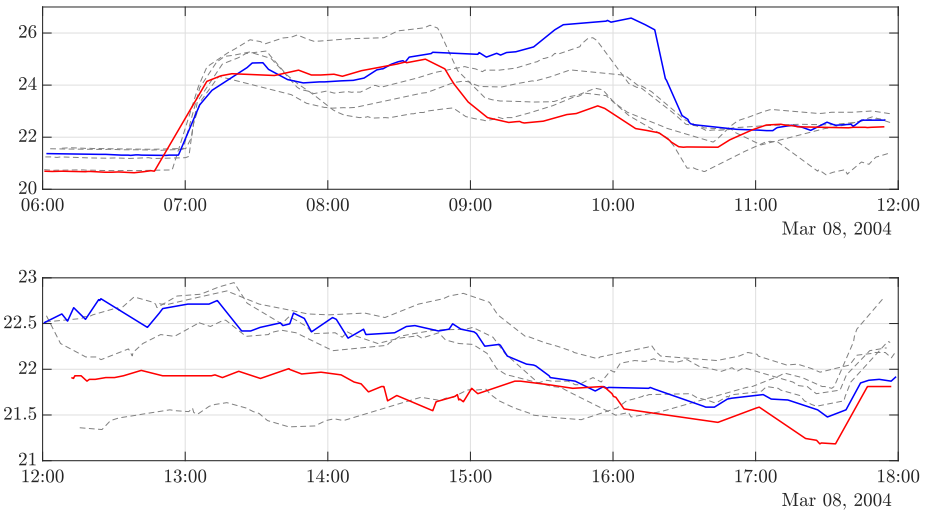

Figure 4.8: *A case study: temperature data in Region* 3*, in the second and third time period. On the* x*-axis is reported the time of day, while in the* y*-axis is reported the temperature in degrees of Celsius. Blue and red solid lines correspond respectively to sensor* 1 *and* 10*.*

Table 4.11: *A case study: model validation. Data have been standardized to have zero mean and a variance of one.*

Region	Period	Sensor identifier	Mean Squared Error	Sensor identifier	Mean Squared Error
1	$00:00-06:00$	44	0.0048	50	0.0045
	$06:00-12:00$	44	0.0938	50	0.0854
	$12:00-18:00$	44	0.0317	50	0.0123
	$18:00-00:00$	44	0.0079	50	0.0061
2	$00:00-06:00$	33	0.0067	43	0.0101
	$06:00-12:00$	33	0.0399	43	0.0909
	$12:00-18:00$	33	0.0367	43	0.0433
	$18:00-00:00$	33	0.0372	43	0.0083
3	$00:00-06:00$	1	0.0043	10	0.0069
	$06:00-12:00$	1	0.0523	10	0.1138
	$12:00-18:00$	1	0.0778	10	0.1185
	$18:00-00:00$	1	0.0147	10	0.0209
4	$00:00-06:00$	26	0.0033	32	0.0077
	$06:00-12:00$	26	0.0136	32	0.1080
	$12:00-18:00$	26	0.0061	32	0.0292
	$18:00-00:00$	26	0.0123	32	0.0194
5	$00:00-06:00$	17	0.0050	25	0.0051
	$06:00-12:00$	17	0.0415	25	0.0511
	$12:00-18:00$	17	0.0172	25	0.0177
	$18:00-00:00$	17	0.1003	25	0.0143

was applied to benchmark case study concerning temperature data collected from 54 sensors deployed in the Intel Berkeley Research lab [47]. The obtained results show the potential of the proposed algorithm also for this real application.

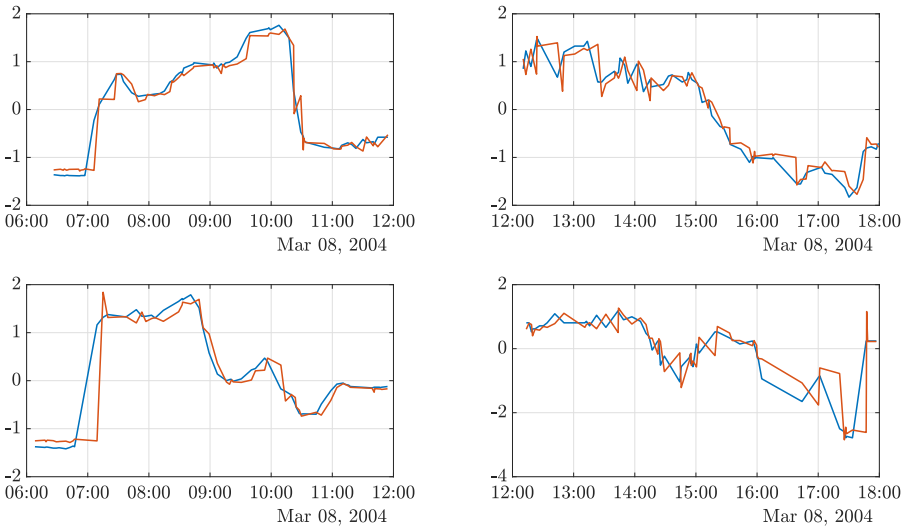

Figure 4.9: *A case study: one-step-ahead prediction for the validation data collected by sensors* 1 *(top) and* 10 *(bottom) in the second (left) and third (right) period. Real data (blue) and prediction (red). Data have been standardized to have zero mean and a variance of one.*

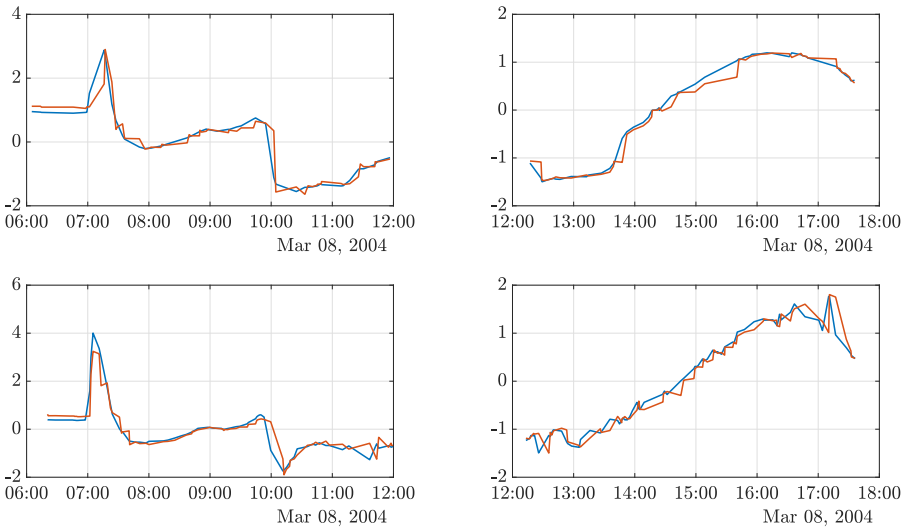

Figure 4.10: *A case study: one-step-ahead prediction for the validation data collected by sensors* 44 *(top) and* 50 *(bottom) in the second (left) and third (right) period. Real data (blue) and prediction (red). Data have been standardized to have zero mean and a variance of one.*

CHAPTER 5

Identification of nonlinear hybrid systems

THIS chapter addresses the identification of discrete time switched nonlinear systems, which are collections of discrete time nonlinear continuous systems (modes) indexed by a finite-valued variable defining the current mode. In particular, we consider the class of Switched Nonlinear AutoRegressive eXogenous (Switched NARX, or SNARX) models, where the continuous dynamics are represented by NARX models. Given a set of input-output data, the identification of a SNARX model for the underlying system involves the simultaneous identification of the mode sequence and of the NARX model associated to each mode, configuring a mixed integer non-convex optimization problem, hardly solvable in practice due to the large combinatorial complexity. In this chapter, we introduce an iterative randomized approach for the segmentation of time-ordered data observed from Switched Nonlinear ARX (SNARX) models. More in detail, the proposed method consists of a two-stage procedure repeated at each iteration, the first stage addressing the SNARX identification problem based on the current set of candidate switching times, and the second aiming at the refinement of such set. The sequence of the SNARX model identification and refinement stages is repeated until convergence, ideally, to the SNARX model that best describes the available data (target model). The described iterative two-stage method requires that the number of modes is *a priori* known (a commonly adopted assumption in the literature), whereas the structure of the NARX models associated with the modes is not assumed to be known. The proposed method is capable of operating with noisy data and can deal with medium/large datasets.

The rest of the chapter is organized as follows. Section 5.1 briefly reviews the state of the art. Section 5.2 provides a general overview of the proposed two-stage procedure, which is then detailed in Sections 5.2.1 and 5.2.2. Some simulation examples are

presented in Section 5.3. Section 5.4 discusses the application of the proposed procedure to the problem of data-driven model learning of a component placement process in a pick-and-place machine, which requires to extend the method to PWARX systems. Finally, some concluding remarks end the chapter.

5.1 State of the art

Many approaches have been proposed over the last two decades for the case of *affine* continuous dynamics (see, *e.g.*, [94] and [44] for a comprehensive review). These methods can be roughly classified into two categories, depending on how the optimization problem is tackled. Some methods adopt a solution strategy which addresses the full problem, optimizing simultaneously over both continuous and discrete variables, [10], [104], [77], [86], [5], [96], [91], [89], [80], [28], while other methods deal separately with the mode and structure classification and the parameter estimation tasks, [37], [58], [97], [51].

In the first category, the bounded-error approach in [10] imposes that the identification error is finite bounded and it addresses the identification problem as the partition into a minimum number of feasible subsystems for a set of linear complementary inequalities derived from data. The procedure in [104] solves directly the optimization problem by using mixed-integer linear or quadratic programming which is guaranteed to converge to a global minimum. However, this approach is computationally affordable only for small problems. The algebraic approach [77] proposes a continuous approximation based on a polynomial constraint which does not depend on the inference of the hybrid state and the switching mechanism. In [5], the author poses the identification as a combinatorial ℓ_0 optimization problem, then relaxed into a convex ℓ_1-norm minimization problem. The underlying idea is to find among all submodels that of achieves, over the whole dataset, the sparsest vector of fitting errors. An ℓ_1-relaxation is used also in [89], where the general optimization takes the form of a least squares problem with sum-of-norm regularization. Instead, in [96] the ℓ_0-minimization problem is relaxed into a sequence of convex semidefinite programming problems, whose solutions are guaranteed to convergence to the optimal solution of the original problem. A swarm-based approach is used in [80] to handle the non-convex nature of the optimization problem. The approach in [28] proposes a multi-model least-squares technique to iteratively solve the clustering and fitting of affine functions to data. Then, a second stage is devoted to the computation of a polyhedral partition of the regressor space based on multi-class discrimination methods. The same authors recently proposed in [9] a framework for fitting jump models which encompasses also the case of piecewise affine models. The proposed coordinate descent algorithm alternates between the estimation of the model parameters, and the data classification.

As for the methods that deal separately with the classification and estimation tasks, a clustering-based approach is presented in [37], where a K-means like algorithm is used to partition the feature vectors space which implicitly leads to the accomplishment of data classification. Then, the parameters are estimated from aggregated data through weighted least squares. The method in [58] exploits the Bayesian inference to estimate the model parameters. After this phase, each data point is attributed to the mode that most likely generated it. The Bayesian framework is adopted also in [97]

where submodels impulse responses are described by stable spline stochastic models enhanced with further hyperparameters representing the classification variables. Data classification is thus carried out in a marginal likelihood optimization approach which is efficiently approximated by using a Markov chain Monte Carlo scheme. A three steps procedure is presented in [51]. The first two steps estimate the discrete sequence via sum-of-norm regularized least squares and expectation maximization clustering, while the last step estimates the models from classified data.

Surprisingly fewer works have tackled the case of *nonlinear* continuous dynamics associated to the modes, in spite of its importance in modeling complex applications. Indeed, if no *a priori* information on the number of modes is available, one can in principle identify an arbitrarily high number of local linear models (and switchings among them) in order to achieve a good model accuracy. However, this prevents the identification of the real dynamics of the hybrid system and hinders its physical interpretation. It also greatly aggravates the combinatorial complexity of the optimization problem, due to the increasing number of switchings. An attempt to deal with nonlinear HSs is documented in [63], where a method based on kernel regression and Support Vector Machines (SVMs) is discussed. In this setting, the number of variables over which the optimization is carried out grows rapidly with the number of data N and the number of modes N_M, according to $2N_M\,(N+1)$, and hence this method can deal only with relatively small problems. A reformulation of the optimization problem in a continuous framework is studied in [65] and [66], thus allowing the use of efficient solvers and enabling the solution of larger problems. The efficiency of this method is further improved in [68] by introducing fixed-size kernel submodels. In [64], the authors proposed an extension of the sum-of-norms approach described in [89] to piecewise systems with nonlinear dynamics, based again on kernel functional expansions. The method employs a convex cost function containing an accuracy term (quantifying the quality of fit of each local model on the assigned data samples), a term penalizing the local model complexity, and a variational term which controls the overall complexity as a function of the number of local models. Note that, in case of time-ordered and consecutive data, the proposed approach is similar to that in [34] which addresses the segmentation of ordered data obtained from nonlinear dynamical systems. In [6] the identification problem is first formulated as a sparse optimization problem and then relaxed in a convex form by approximating the ℓ_0 norm with the ℓ_1 norm. A sufficient condition guaranteeing the optimality of the relaxed convex problem solution was provided only under a noiseless assumption. The notion of robust sparsity is introduced in [69] to extend the applicability of the previous method to the noisy case. On the down side, the method requires the careful setting of several parameters (*e.g.*, the factor that defines the trade-off between model complexity and accuracy, or the weights used to improve the sparsity of the solution), which appears to be far from trivial.

It is worth noticing that most of the aforementioned approaches are nonparametric in that they are based on kernel functional expansions. Instead, here the identification problem has been addressed from a parametric perspective using nonlinear models of the NARX/NARMAX class [70, 71], where the nonlinear functions are represented as finite-dimensional parameterized polynomial expansions. Indeed, this is a very popular approach in black-box nonlinear model identification [21], provided the identification

procedure includes a model structure selection (MSS) process to tackle the curse of dimensionality issue that is inherent to polynomial expansions. Polynomial nonlinear models of this type have several attractive features (see *e.g.* [21] for a more detailed discussion), among which the ability to represent a wide range of nonlinear systems using a small number of parameters, the easy interpretability, and the amenability to nonlinear frequency analysis using generalized frequency response functions.

5.2 A randomized two-stage iterative method

To ease the discussion, we report here the SNARX identification problem which has been detailed in Section 2.1.3:

$$\lambda^\star = (\boldsymbol{\sigma}^\star, S^\star) = \arg\min_{\lambda \in \Lambda} \mathcal{L}(\lambda), \tag{5.1}$$

where λ represents the SNARX model structure. This structure involves $N \times N_M$ binary variables for $\boldsymbol{\sigma}$ (the mode switching signal), plus $n \times N_M$ for S (the collection of NARX model structures, *i.e.,* $S = [\boldsymbol{s}^{(1)}, \dots, \boldsymbol{s}^{(N_M)}] \in \mathcal{S} = \{1, 2\}^{n \times N_M}$, where $s_j^{(i)} = 1$ if the j-th regressor belongs to the model structure of mode i, and $s_j^{(i)} = 2$ otherwise). Regarding this optimization problem, we argued that it is a mixed integer program and also that is typically computationally intractable due to its combinatorial complexity. In particular, we observed that the most critical aspect of this problem is the sample-mode mapping, since switchings can occur at arbitrary times. However, it is typically true that the number of true switching time instants in the observed data is significantly less than N.

In view of this, we address the SNARX identification problem (5.1) using an iterative two-stage approach, where at each iteration the identification is first carried out by restricting the possible switching occurrences at a limited (small) number of time instants, and then, based on the results of this operation, the set of allowed switching times is refined. The restriction of the candidate switchings is crucial in reducing the combinatorial complexity of the optimization problem associated to the identification task performed in the first stage, thus allowing its solvability. More in detail, it induces a partition of the data into (a small number of) sub-periods, each of which is associated to a mode, and the NARX model associated to each mode can be identified based on all the data segments labeled with it.

In the first stage a randomized algorithm is employed for the estimation of the SNARX model best fitting the available data, given that the switching locations are restricted to be in the set $\mathcal{T}_s = \{t_k\}_{k=1}^{N_s}$, with $1 < t_1 < t_2 < \ldots < t_{N_s} \leq N$ and $N_s \ll N$. The information resulting from the first stage is used to refine the switchings positioning defined by $\mathcal{T}_s$ before a new execution of the first stage is carried out. This is done by means of a split-and-merge procedure designed to finitely tune the number and the location of the candidate switching times. The rationale is to add further possible switching times in the neighborhood of detected switchings, while at the same time removing candidate switching locations that were not identified as such. By iterating this two-stage procedure, one can progressively improve the identification of the switching locations as well as that of the NARX models associated to the modes.

The two stages are described in detail in the next two sections.

5.2.1 First stage: identification

The first stage of the method is an application to the SNARX model class of the randomized method presented in Chapter 3, under the assumption that switchings can occur only at specific time instants.

Let $I_1 = \{t \mid 1 \leq t < t_1\}$, $I_k = \{t \mid t_{k-1} \leq t < t_k\}$, $k = 2, \ldots, N_s$, and $I_{N_s+1} = \{t \mid t_{N_s} \leq t \leq N\}$, be the $N_s + 1$ time intervals induced by $\mathcal{T}_s$. Define also the corresponding set of admissible switching signals:

$$\Sigma_{\mathcal{T}_s} = \{\boldsymbol{\sigma} | \sigma(t') = \sigma(t''),\ \forall t', t'' \in I_k,\ k = 1, \ldots, N_s + 1\}.$$

One can associate a mode κ_k to each time interval I_k, $k = 1, \ldots, N_s + 1$, and define vector $\boldsymbol{\kappa} = [\kappa_1, \ldots, \kappa_{N_s+1}] \in \{1, \ldots, N_M\}^{N_s+1}$. Then, with a slight abuse of notation, the SNARX model structure λ can be re-parameterized as $\lambda = (\boldsymbol{\kappa}, S)$. Accordingly, our goal is to find the best SNARX model with switching signal in $\Sigma_{\mathcal{T}_s}$:

$$\lambda^\star = (\boldsymbol{\kappa}^\star, S^\star) = \arg\max_{\lambda \in \Lambda} \mathcal{J}(\lambda), \tag{5.2}$$

where we set $\Lambda = \{1, \ldots, N_M\}^{N_s+1} \times \{1, 2\}^{n \times N_M}$, and $\mathcal{J}(\lambda) : \Lambda \to [0, 1]$ is the following performance index:

$$\mathcal{J}(\lambda) = e^{-K_\lambda \mathcal{L}(\lambda)}, \tag{5.3}$$

with $K_\lambda > 0$ being a scaling parameter.

Note that problem (5.2) has the same structure of (3.1) and therefore can be addressed in the probabilistic framework described in Chapter 3. To this purpose, let $\boldsymbol{\gamma} = (\boldsymbol{\xi}, \boldsymbol{\rho})$, where $\boldsymbol{\xi}$ is a discrete random variable taking values in $\{1, \ldots, N_M\}^{N_s+1}$ according to $\mathbb{P}_{\boldsymbol{\xi}}$ that accounts for the mode switchings, and $\boldsymbol{\rho}$ is a discrete variable taking values in $\{1, 2\}^{n \times N_M}$ according to $\mathbb{P}_{\boldsymbol{\rho}}$ that accounts for the structures of the N_M modes.

If we assume that the mode switching and the local model structures are independent, we can express $\mathbb{P}_{\boldsymbol{\gamma}}$ as:

$$\mathbb{P}_{\boldsymbol{\gamma}}(\lambda) = \mathbb{P}_{\boldsymbol{\xi}}(\boldsymbol{\kappa}) \cdot \mathbb{P}_{\boldsymbol{\rho}}(S), \tag{5.4}$$

where $\lambda = (\boldsymbol{\kappa}, S) \in \Lambda$.

Parametrization of $\mathbb{P}_{\boldsymbol{\xi}}$

The random variable $\boldsymbol{\xi}$ is a vector of $N_s + 1$ random variables ξ_k, $k = 1, \ldots, N_s + 1$, each one representing the mode associated to the corresponding time interval I_k. We can then introduce vector $\boldsymbol{\eta}_k = [\eta_k^{(1)}, \ldots, \eta_k^{(N_M)}]$, where $\eta_k^{(i)}$ represents the probability of assigning mode i to sub-period I_k and is denoted as Mode Extraction Probability (MEP) in the following. Clearly, $\sum_{i=1}^{N_M} \eta_k^{(i)} = 1$.

If we assume independence between the random variables ξ_k, $k = 1, \ldots, N_s + 1$, then, the probability distribution of $\boldsymbol{\xi}$ is given by

$$\mathbb{P}_{\boldsymbol{\xi}}(\boldsymbol{\kappa}) = \mathbb{P}_{\boldsymbol{\xi}}([\kappa_1, \ldots, \kappa_{N_s+1}]) = \prod_{k=1}^{N_s+1} \prod_{i=1}^{N_M} \left(\eta_k^{(i)}\right)^{b_k^{(i)}}, \tag{5.5}$$

where $b_k^{(i)} = 1$ if $\kappa_k = i$, and 0 otherwise. $\mathbb{P}_{\boldsymbol{\xi}}$ is uniquely defined by matrix

$$\boldsymbol{\eta} = \begin{bmatrix} \boldsymbol{\eta}_1 \\ \vdots \\ \boldsymbol{\eta}_{N_s+1} \end{bmatrix} \in \mathbb{R}^{(N_s+1)\times N_M}. \tag{5.6}$$

Parametrization of $\mathbb{P}_{\boldsymbol{\rho}}$

We associate each regressor φ_j in each mode i to a Categorical distribution[1] $\rho_{j,i} \sim \texttt{Categorical}(\boldsymbol{\mu}_{j,i})$, where $\boldsymbol{\mu}_{j,i} = (\mu_{j,i}^{(1)}, \mu_{j,i}^{(2)})$. The outcome 1 encodes the case that φ_j is present in the i-th local model structure ($s_j^{(i)} = 1$), while the outcome 2 encodes the case that φ_j is absent ($s_j^{(i)} = 2$). Clearly, $\mu_{j,i}^{(1)} + \mu_{j,i}^{(2)} = 1$. The probabilities $\mu_{j,i}^{(1)}$ are denoted as Regression Inclusion Probabilities (RIPs).

The collection of all parameters $\mu_{j,i}^{(1)}$, and $\mu_{j,i}^{(2)}$, $j = 1, \ldots, n$, $i = 1, \ldots, N_M$ defines a matrix

$$\boldsymbol{\mu} = \begin{bmatrix} \boldsymbol{\mu}_1^{(1)} & \boldsymbol{\mu}_1^{(2)} \\ \vdots & \vdots \\ \boldsymbol{\mu}_{N_M}^{(1)} & \boldsymbol{\mu}_{N_M}^{(2)} \end{bmatrix} \in \mathbb{R}^{(N_M \cdot n)\times 2}, \tag{5.7}$$

where $\boldsymbol{\mu}_i^{(l)} = [\mu_{1,i}^{(l)}, \ldots, \mu_{n,i}^{(l)}]^T \in \mathbb{R}^n$, $i = 1, \ldots, N_M$, $l = 1, 2$. If the random variables $\rho_{j,i}$, $j = 1, \ldots, n$, are independent, then the probability distribution $\mathbb{P}_{\boldsymbol{\rho}^{(i)}}$ of $\boldsymbol{\rho}^{(i)} = [\rho_{1,i}, \ldots, \rho_{n,i}]^T$ is given by

$$\mathbb{P}_{\boldsymbol{\rho}^{(i)}}(\boldsymbol{s}^{(i)}) = \mathbb{P}_{\boldsymbol{\rho}^{(i)}}([s_1^{(i)}, \ldots, s_n^{(i)}]) = \prod_{j:\varphi_j \in \boldsymbol{s}^{(i)}} \mu_{j,i}^{(1)} \prod_{j:\varphi_j \notin \boldsymbol{s}^{(i)}} \mu_{j,i}^{(2)} = \prod_{j=1}^{n} \prod_{l=1}^{2} \left(\mu_{j,i}^{(l)}\right)^{\zeta_{j,i}^{(l)}},$$

where $\zeta_{j,i}^{(l)} = 1$ if $s_j^{(i)} = l$, and 0 otherwise.
Under the assumption of independence between mode structures, we then have that the probability distribution of the random vector $\boldsymbol{\rho}$ associated with the SNARX model structure is given by

$$\mathbb{P}_{\boldsymbol{\rho}}(S) = \prod_{i=1}^{N_M} \mathbb{P}_{\boldsymbol{\rho}^{(i)}}(\boldsymbol{s}^{(i)}) = \prod_{i=1}^{N_M} \prod_{j=1}^{n} \prod_{l=1}^{2} \left(\mu_{j,i}^{(l)}\right)^{\zeta_{j,i}^{(l)}}. \tag{5.8}$$

Therefore, $\mathbb{P}_{\boldsymbol{\rho}}$ is uniquely defined by matrix $\boldsymbol{\mu}$ in (5.7).

Tuning of $\mathbb{P}_{\boldsymbol{\gamma}}$

The overall probability distribution $\mathbb{P}_{\boldsymbol{\gamma}}$ in (5.4) is parameterized by $\eta_k^{(i)}$ and $\mu_{j,i}^{(l)}$, $k = 1, \ldots, N_s + 1$, $j = 1, \ldots, n$, $i = 1, \ldots, N_M$, and $l = 1, 2$. By setting

$$\boldsymbol{\pi} = \left(\boldsymbol{\eta}_1, \ldots, \boldsymbol{\eta}_{N_s+1}, \boldsymbol{\mu}_{1,1}, \ldots, \boldsymbol{\mu}_{n,1}, \ldots, \boldsymbol{\mu}_{1,N_M}, \ldots, \boldsymbol{\mu}_{n,N_M}\right),$$

[1] Notice that a Categorical distribution with only two outcomes is equivalent to a Bernoullian distribution.

$\mathbb{P}_{\boldsymbol{\gamma}}$ can be succinctly written in the form of (3.2) as

$$\mathbb{P}_{\boldsymbol{\gamma}}(\lambda) = \prod_{j=1}^{N_s+1+N_M\cdot n} \prod_{i=1}^{m_j} \left(\pi_j^{(i)}\right)^{\beta_j^{(i)}}, \tag{5.9}$$

where

$$m_j = \begin{cases} N_M, & j \leq N_s + 1 \\ 2, & \text{otherwise} \end{cases},$$

and the $\beta_j^{(i)}$ values are the element of a vector $\boldsymbol{\beta}$ defined as

$$\begin{aligned}\boldsymbol{\beta} = [&b_1^{(1)}, \ldots, b_1^{(N_M)}, \ldots, b_{N_s+1}^{(1)}, \ldots, b_{N_s+1}^{(N_M)}, \\ &\zeta_{1,1}^{(1)}, \zeta_{1,1}^{(2)}, \ldots, \zeta_{n,1}^{(1)}, \zeta_{n,1}^{(2)}, \ldots, \zeta_{1,N_M}^{(1)}, \zeta_{1,N_M}^{(2)}, \ldots, \zeta_{n,N_M}^{(1)}, \zeta_{n,N_M}^{(2)}].\end{aligned}$$

In this view, the results stemming from Theorems 1 and 2 can be used to develop suitable tuning rules for $\mathbb{P}_{\boldsymbol{\gamma}}$, as follows.

The randomized procedure involves extracting and evaluating samples $\lambda = (\boldsymbol{\kappa}, S)$ of the random variable $\boldsymbol{\gamma} = (\boldsymbol{\xi}, \boldsymbol{\rho})$, according to the distribution $\mathbb{P}_{\boldsymbol{\gamma}}$, to gather information for tuning $\mathbb{P}_{\boldsymbol{\gamma}}$. To update the MEP $\eta_k^{(i)}$ we employ a sampled version of the index:

$$\delta_k^{(i)} = \mathbb{E}_{\mathbb{P}_{\boldsymbol{\gamma}}}\left[\mathcal{J}(\boldsymbol{\gamma})|\xi_k = i\right] - \mathbb{E}_{\mathbb{P}_{\boldsymbol{\gamma}}}\left[\mathcal{J}(\boldsymbol{\gamma})|\xi_k \neq i\right], \tag{5.10}$$

which compares the average performance of $\boldsymbol{\gamma}$ in case mode i is assigned to time period I_k with the average performance of $\boldsymbol{\gamma}$ in the opposite case. If $\delta_k^{(i)} > 0$ it pays off to apply the mentioned mode assignment. Since, in practice, index $\delta_k^{(i)} > 0$ can only be calculated in an approximate sampled version, we use this information in a conservative way, defining the following tuning rule:

$$\eta_k^{(i)} \leftarrow \eta_k^{(i)} + \chi\delta_k^{(i)}, \tag{5.11}$$

where the step size $\chi > 0$ is a design parameter.

Similarly, we update $\mu_{j,i}^{(l)}$ based on an aggregate index that weighs the advantages of picking regressor φ_j for mode i:

$$\ell_{j,i}^{(l)} = \mathbb{E}_{\mathbb{P}_{\boldsymbol{\gamma}}}\left[\mathcal{J}(\boldsymbol{\gamma})|\rho_{j,i} = l\right] - \mathbb{E}_{\mathbb{P}_{\boldsymbol{\gamma}}}\left[\mathcal{J}(\boldsymbol{\gamma})|\rho_{j,i} \neq l\right]. \tag{5.12}$$

Index $\ell_{j,i}^{(l)}$ compares the average performance of $\boldsymbol{\gamma}$ in case φ_j is included in the model structure for mode i with the average performance of $\boldsymbol{\gamma}$ in the opposite case. As with $\delta_k^{(i)}$, only an approximate sampled version of $\ell_{j,i}^{(l)}$ can be calculated in practice, which motivates the use of an update law which balances the prior knowledge with the new estimate of the index:

$$\mu_{j,i}^{(l)} \leftarrow \mu_{j,i}^{(l)} + \chi\ell_{j,i}^{(l)}. \tag{5.13}$$

The iterative application of the above update rules guarantee (local) convergence to the target limit distribution

$$\mathbb{P}_{\boldsymbol{\gamma}}^{\star} = \arg\max_{\mathbb{P}_{\boldsymbol{\gamma}}} \mathbb{E}_{\mathbb{P}_{\boldsymbol{\gamma}}}[\mathcal{J}(\boldsymbol{\gamma})]$$

as stated in Theorem 2.

Guidelines for parameter settings

As discussed in Section 3.2, choosing the correct step size χ in the update of the MEPs and RIPs is crucial for the convergence speed of the algorithm. Accordingly, we adopt here the adaptive tuning rule introduced in (3.13).

The convergence speed of the algorithm is also influenced by the choice of K_λ in (5.3). As an alternative to a classical trial-and-error approach, we here provide a simple tuning procedure for K_λ, designed to allow a better discrimination between models with similar performance. Let us denote by $OM(x) = \lfloor \log_{10}(x) \rfloor$ the order of magnitude of a nonnegative number x. Parameter K_λ is tuned at the first iteration of the algorithm according to the minimum $OM(\mathcal{L}(\lambda))$, computed based on the extracted SNARX models. Specifically,

$$K_\lambda = 10^{-(\min(OM(\mathcal{L}(\lambda)))+1)}. \tag{5.14}$$

Regarding the initialization of the probability distribution, we set the parameters $\mu_{j,i}^{(l)}$ to equal small values, to encourage the extraction of small models at the early stages of the algorithm. As for the $\eta_k^{(i)}$, in the absence of any a-priori assumption on the switching signal, we attribute equal probabilities $\eta_k^{(i)} = 1/N_M$ to all modes in each sub-period I_k.

Concerning the choice of $\mathcal{T}_s$, we initially place the candidate switching time instants uniformly over $\{1, N\}$, dividing the time horizon in sub-periods of equal length. In choosing this placement, one can take advantage from the *a priori* knowledge on the minimum dwell time of the system in a mode. Indeed, in practical applications, where the mode switching is caused by activation/deactivation of devices and system reconfiguration, switchings cannot generally occur at consecutive time steps and a certain time must be allowed to pass between switchings. If such information is available, the maximum sub-period length should be upper bounded by the minimum dwell time, so that at most one switching can occur inside a given sub-period, thus reducing the number of mixed sub-periods, as discussed in Section 5.2.2. Notice also that a significant reduction of the combinatorial complexity can be leveraged for what concerns the switching signal (only switching signals that do not violate the minimum dwell time are acceptable).

An heuristic implementation

Let $N_i = \sum_{t=1}^{N} \beta_t^{(i)}$ denote the number of samples in the data set that are associated with mode i. The performance index $\mathcal{L}(\lambda)$ in (2.27) can be explicitly expressed in terms of the contribution of each mode as

$$\mathcal{L}(\lambda) = \frac{1}{N} \sum_{i:\, N_i \neq 0} N_i \cdot \mathcal{L}^{(i)}(\boldsymbol{\kappa}, \boldsymbol{s}^{(i)}),$$

where $\mathcal{L}^{(i)}(\boldsymbol{\kappa}, \boldsymbol{s}^{(i)})$ measures the accuracy of the model of the i-th mode. Index $\mathcal{L}^{(i)}(\boldsymbol{\kappa}, \boldsymbol{s}^{(i)})$ is well-defined if $N_i \neq 0$ and is given by:

$$\begin{aligned} \mathcal{L}^{(i)}(\boldsymbol{\kappa}, \boldsymbol{s}^{(i)}) = & \min_{\boldsymbol{\vartheta}^{(i)}} \frac{1}{N_i} \sum_{t=1}^{N} \beta_t^{(i)} \cdot \left(y(t) - \boldsymbol{\varphi}(t) \boldsymbol{\vartheta}^{(i)} \right)^2 \\ & \text{subject to } \vartheta_j^{(i)} = 0 \text{ if } s_j^{(i)} = 2, j = 1, \ldots, n. \end{aligned} \tag{5.15}$$

The convergence speed of the algorithm can be improved by updating the probability distribution associated to the model structures of the modes (see equation (5.12)) *separately* for each mode, based on a local performance index of the following type:

$$\mathcal{J}^{(i)}(\boldsymbol{\kappa}, \boldsymbol{s}^{(i)}) = e^{-K_i \mathcal{L}^{(i)}(\boldsymbol{\kappa}, \boldsymbol{s}^{(i)})}, \tag{5.16}$$

as opposed to the full $\mathcal{J}(\lambda)$. In expression (5.16) $K_i > 0$ is a design parameter that can be tuned similarly to (5.14):

$$K_i = 10^{-\left(\min\left(OM\left(\mathcal{L}^{(i)}(\boldsymbol{\kappa}, \boldsymbol{s}^{(i)})\right)\right)+1\right)}. \tag{5.17}$$

As a result, the update term $\ell_{j,i}^{(l)}$ is modified as follows:

$$\tilde{\ell}_{j,i}^{(l)} = \mathbb{E}_{\mathbb{P}_{\boldsymbol{\gamma}}}\left[\mathcal{J}^{(i)}(\boldsymbol{\kappa}, \boldsymbol{\rho}^{(i)}) | \rho_{j,i} = l, \boldsymbol{\xi} = \boldsymbol{\kappa}\right] - \mathbb{E}_{\mathbb{P}_{\boldsymbol{\gamma}}}\left[J^{(i)}(\boldsymbol{\kappa}, \boldsymbol{\rho}^{(i)}) | \rho_{j,i} \neq l, \boldsymbol{\xi} = \boldsymbol{\kappa}\right], \tag{5.18}$$

which, with reference to mode i, compares the average performance of model structures that include φ_j with that of the remaining structures. Observe that the performance evaluation depends on the switching signal as well, which defines the segments of the data-set that are assigned to mode i. The resulting RIP update law is:

$$\mu_{j,i}^{(l)} \leftarrow \mu_{j,i}^{(l)} + \nu_i \tilde{\ell}_{j,i}^{(l)} \tag{5.19}$$

where $\nu_i > 0$ is the step size for mode i defined (similarly to (3.13)) as:

$$\nu_i = \frac{1}{10\left(\mathcal{J}_{\text{best}}^{(i)} - \overline{\mathcal{J}}^{(i)}\right) + 0.1} \tag{5.20}$$

with $\mathcal{J}_{\text{best}}^{(i)}$ and $\overline{\mathcal{J}}^{(i)}$ being respectively, the best value and the mean value for $\mathcal{J}^{(i)}$ evaluated on the extracted samples for $\boldsymbol{\gamma}$.

In this case, the local convergence of $\mathbb{P}_{\boldsymbol{\gamma}}$ to the target limit distribution $\mathbb{P}_{\boldsymbol{\gamma}}^{\star}$ is not guaranteed, essentially due to possible sign differences between $\ell_{j,i}^{(l)}$ and $\tilde{\ell}_{j,i}^{(l)}$. To show this, consider first the relation between $\mathcal{J}(\lambda)$ and $\mathcal{J}^{(i)}\left(\boldsymbol{\kappa}, \boldsymbol{s}^{(i)}\right)$, $i = 1, \ldots, N_M$:

$$\mathcal{J}(\lambda) = e^{-K_\lambda \mathcal{L}(\lambda)} = e^{-K_\lambda \cdot [\frac{1}{N} \sum_{i:\, N_i \neq 0} N_i \cdot \mathcal{L}^{(i)}(\boldsymbol{\kappa}, \boldsymbol{s}^{(i)})]} = \prod_{i=1}^{N_M} \left[\mathcal{J}^{(i)}\left(\boldsymbol{\kappa}, \boldsymbol{s}^{(i)}\right)\right]^{\frac{N_i}{N}\frac{K_\lambda}{K_i}} \tag{5.21}$$

Based on (5.21) and under the assumption of independence between modes, and between regressors, one can reformulate the first term in the RHS of (5.12) as:

$$\mathbb{E}_{\mathbb{P}_{\boldsymbol{\gamma}}}[\mathcal{J}(\boldsymbol{\gamma}) | \rho_{j,i} = l] = \sum_{\boldsymbol{\kappa}} \mathbb{P}_{\boldsymbol{\xi}}(\boldsymbol{\kappa}) \left[\mathbb{E}\left[\left(\mathcal{J}^{(i)}\left(\boldsymbol{\kappa}, \boldsymbol{s}^{(i)}\right)\right)^{\frac{N_i}{N}\frac{K_\lambda}{K_i}} | \rho_{j,i} = l, \boldsymbol{\xi} = \boldsymbol{\kappa}\right] \cdot \prod_{n \neq i}^{N_M} \mathbb{E}\left[\left(\mathcal{J}^{(i)}\left(\boldsymbol{\kappa}, \boldsymbol{s}^{(i)}\right)\right)^{\frac{N_i}{N}\frac{K_\lambda}{K_i}} | \boldsymbol{\xi} = \boldsymbol{\kappa}\right]\right]. \tag{5.22}$$

Similarly,

$$\mathbb{E}_{\mathbb{P}_{\boldsymbol{\gamma}}}[\mathcal{J}(\boldsymbol{\gamma}) | \rho_{j,i} \neq l] = \sum_{\boldsymbol{\kappa}} \mathbb{P}_{\boldsymbol{\xi}}(\boldsymbol{\kappa}) \left[\mathbb{E}\left[\left(\mathcal{J}^{(i)}\left(\boldsymbol{\kappa}, \boldsymbol{s}^{(i)}\right)\right)^{\frac{N_i}{N}\frac{K_\lambda}{K_i}} | \rho_{j,i} \neq l, \boldsymbol{\xi} = \boldsymbol{\kappa}\right] \cdot \prod_{n \neq i}^{N_M} \mathbb{E}\left[\left(\mathcal{J}^{(i)}\left(\boldsymbol{\kappa}, \boldsymbol{s}^{(i)}\right)\right)^{\frac{N_i}{N}\frac{K_\lambda}{K_i}} | \boldsymbol{\xi} = \boldsymbol{\kappa}\right]\right]. \tag{5.23}$$

By substituting (5.22) and (5.23) in (5.12), one obtains:

$$\mathbb{E}_{\mathbb{P}_{\boldsymbol{\gamma}}}[\mathcal{J}(\boldsymbol{\gamma})|\rho_{j,i}=l]-\mathbb{E}_{\mathbb{P}_{\boldsymbol{\gamma}}}[\mathcal{J}(\boldsymbol{\gamma})|\rho_{j,i}\neq l]=$$

$$\sum_{\boldsymbol{\kappa}}\mathbb{P}_{\boldsymbol{\xi}}(\boldsymbol{\kappa})\left[\mathbb{E}\left[\left(\mathcal{J}^{(i)}\left(\boldsymbol{\kappa},\boldsymbol{s^{(i)}}\right)\right)^{\frac{N_i}{N}\frac{K_\lambda}{K_i}}|\rho_{j,i}=l,\boldsymbol{\xi}=\boldsymbol{\kappa}\right]-\right. \tag{5.24}$$

$$\left.-\,\mathbb{E}\left[\left(\mathcal{J}^{(i)}\left(\boldsymbol{\kappa},\boldsymbol{s^{(i)}}\right)\right)^{\frac{N_i}{N}\frac{K_\lambda}{K_i}}|\rho_{j,i}\neq l,\boldsymbol{\xi}=\boldsymbol{\kappa}\right]\right]\cdot\prod_{n\neq i}^{N_M}\mathbb{E}\left[\left(\mathcal{J}^{(i)}\left(\boldsymbol{\kappa},\boldsymbol{s^{(i)}}\right)\right)^{\frac{N_i}{N}\frac{K_\lambda}{K_i}}|\boldsymbol{\xi}=\boldsymbol{\kappa}\right]. \tag{5.25}$$

That is, the sign of each single $\tilde{\ell}_{j,i}^{(l)}$ $\left(\text{the inner part of the summation over }\boldsymbol{\kappa}\right)$ may be different from that of $\ell_{j,i}^{(l)}$.

However, as discussed in Section 5.2.1, the experimental evidence indicates that this occurs relatively seldom and scarcely affects the overall identification results (see Table 5.2 and Figure 5.2). This justifies the adoption of this heuristic version of the algorithm in view of its more favorable computational characteristics.

Algorithm 5 reports the whole identification procedure carried out in the first stage.

Example 1: $\mathcal{T}_s^\circ \subseteq \mathcal{T}_s$

Recalling that $\mathcal{T}_s^\circ$ identifies the set of true switching time instants, it can happen that $\mathcal{T}_s^\circ \subseteq \mathcal{T}_s$ or, more frequently, $\mathcal{T}_s^\circ \nsubseteq \mathcal{T}_s$. We discuss here the former condition, while the latter one is the subject of the next subsection.

Consider the following SNARX system [63], which switches between a linear mode 1:

$$y(t)=-0.905y(t-1)+0.9u(t-1)+e(t),$$

and a nonlinear mode 2:

$$y(t)=-0.4y(t-1)^2+0.5u(t-1)+e(t),$$

where $e(t)$ is a zero mean Gaussian noise of variance 0.012 and $u(t)$ is uniformly distributed in the interval $[0,1]$. An observation window of $N=2000$ samples has been collected, which contains 4 switchings, at $t=400$ (from mode 1 to mode 2), $t=1500$ (from mode 2 to mode 1), $t=1600$ (from mode 1 to mode 2), and $t=1700$ (from mode 2 to mode 1), so that $\mathcal{T}_s^\circ=\{400,1500,1600,1700\}$. In the absence of any *a priori* information regarding the candidate switching times, we uniformly divide the time horizon in 20 sub-periods of length 100, setting $t_k=100k$, $k=1,\ldots,19$. Notice that, while this hugely simplifies the combinatorial complexity of the problem, more than 1 million different possible switching signals are nevertheless compatible with the defined $\mathcal{T}_s$. In this case study, the set of pre-defined candidate switchings includes the true ones. The design parameters have been set to $n_y=n_u=n_d=2$ (for a total of 15 possible regressors for each NARX model). Furthermore, the initial MEPs are all set to 0.5, and the initial RIPs to 0.0667.

Worth noting, the algorithm extracts useful information on the model from partially correct extracted models. To emphasize this property, consider Figure 5.1 which shows the probability distribution state, in terms of the scalar parameters $\eta_k^{(i)}$ and $\mu_{j,i}^{(l)}$, $k=1,\ldots,N_s+1$, $j=1,\ldots,n$, $i=1,\ldots,N_M$, $l=1,2$, obtained by interrupting the algorithm well before convergence, at the iteration when the correct model is first extracted. All the information gathered up to this point to tune the probability distribution

Algorithm 5 First stage: identification algorithm

Require: $\{(\boldsymbol{\varphi}(t), \boldsymbol{y}(t)), t = 1, \cdots, N\}$,
$n, N_M, \mathcal{T}_s^{(0)} = \{t_1, \ldots, t_{N_s}\}, N_p, K_\lambda, K_i, \mu_{min}, \mu_{max}, \eta_{min}, \eta_{max}, \varepsilon$
Ensure: $\boldsymbol{\eta}, \boldsymbol{\mu}$

$\boldsymbol{\eta}_k \leftarrow \frac{1}{N_M} \cdot \mathbf{1}_{N_M \times 1}$;
$\boldsymbol{\mu}_i^{(1)} \leftarrow \frac{1}{n} \cdot \mathbf{1}_{n \times 1}$;
Define the time intervals $I_k, k = 1, \ldots, N_s$ according to $\mathcal{T}_s^{(0)}$;
repeat
  for $p = 1$ **to** N_p **do**
    $\boldsymbol{\kappa}^{(p)} \leftarrow [\,]$;
    for $k = 1$ **to** N_s **do** ▷ Generate switching signal
      Extract $\kappa_k^{(p)}$ from Categorical($\boldsymbol{\eta}_k$);
      $\boldsymbol{\kappa}^{(p)} \leftarrow [\boldsymbol{\kappa}^{(p)},\ \kappa_k^{(p)}]$;
    end for
    for $i = 1$ **to** N_M **do**
      $\tilde{I} \leftarrow \{I_k | \kappa_k^{(p)} = i\}$; ▷ Aggregate data
      $\mathcal{D}^{(i)} \leftarrow \{(\boldsymbol{\varphi}(t), \boldsymbol{y}(t)) | t \in \tilde{I}\}$;
      $N_i \leftarrow |\mathcal{D}^{(i)}|$;
      $\{\boldsymbol{s}_p^{(i)}, \hat{\boldsymbol{\vartheta}}^{(i)}\} \leftarrow$ run Algorithm 1 on data $\mathcal{D}^{(i)}$ and RIPs $\boldsymbol{\mu}_i$; ▷ Generate NARX model
      Compute $\mathcal{L}_p^{(i)}$ according to (5.15); ▷ Mode evaluation
      $\mathcal{J}_p^{(i)} \leftarrow e^{-K_i \mathcal{L}_p^{(i)}}$;
    end for
    $\mathcal{L}_p \leftarrow \frac{1}{N} \sum_{i:N_i \neq 0} N_i \mathcal{L}_p^{(i)}$; ▷ SNARX model evaluation
    $\mathcal{J}_p \leftarrow e^{-K_\lambda \mathcal{L}_p}$;
  end for
  $\chi \leftarrow \frac{1}{10(\mathcal{J}_{\text{best}} - \overline{\mathcal{J}}) + 0.1}$;
  for $k = 1$ **to** N_s **do** ▷ Update $\boldsymbol{\eta}_k$
    for $i = 1$ **to** N_M **do**
      $\mathcal{J}^{\oplus} \leftarrow 0$; $n^{\oplus} \leftarrow 0$; $\mathcal{J}^{\ominus} \leftarrow 0$; $n^{\ominus} \leftarrow 0$;
      for $p = 1$ **to** N_p **do**
        if $\boldsymbol{\kappa}_k^{(p)} = i$ **then**
          $\mathcal{J}^{\oplus} \leftarrow \mathcal{J}^{\oplus} + \mathcal{J}_p$; $n^{\oplus} \leftarrow n^{\oplus} + 1$;
        else
          $\mathcal{J}^{\ominus} \leftarrow \mathcal{J}^{\ominus} + \mathcal{J}_p$; $n^{\ominus} \leftarrow n^{\ominus} + 1$;
        end if
      end for
      $\eta_k^{(i)} \leftarrow \eta_k^{(i)} + \chi \left(\frac{\mathcal{J}^{\oplus}}{\max(n^{\oplus}, 1)} - \frac{\mathcal{J}^{\ominus}}{\max(n^{\ominus}, 1)} \right)$;
      $\eta_k^{(i)} \leftarrow \max\left(\min\left(\eta_k^{(i)}, \eta_{max}\right), \eta_{min}\right)$; ▷ Saturation
    end for
    $\eta_k^{(i)} \leftarrow \frac{\eta_k^{(i)}}{\sum_{i=1}^{N_M} \eta_k^{(i)}}$; ▷ Normalization
  end for

for $i = 1$ **to** N_M **do** ▷ Update $\boldsymbol{\mu}_i$ - heuristic implementation
  $\nu_i \leftarrow \frac{1}{10\left(\mathcal{J}_{\text{best}}^{(i)} - \overline{\mathcal{J}}^{(i)}\right)+0.1}$;
  for $j = 1$ **to** n **do**
    $\mathcal{J}^{\oplus} \leftarrow 0$; $n^{\oplus} \leftarrow 0$; $\mathcal{J}^{\ominus} \leftarrow 0$; $n^{\ominus} \leftarrow 0$;
    for $p = 1$ **to** N_p **do**
      if $s_j^{(p)} = 1$ **then**
        $\mathcal{J}^{\oplus} \leftarrow \mathcal{J}^{\oplus} + \mathcal{J}_p^{(i)}$; $n^{\oplus} \leftarrow n^{\oplus} + 1$;
      else
        $\mathcal{J}^{\ominus} \leftarrow \mathcal{J}^{\ominus} + \mathcal{J}^{(p)}$; $n^{\ominus} \leftarrow n^{\ominus} + 1$;
      end if
    end for
    $\mu_{j,i}^{(1)} \leftarrow \mu_{j,i}^{(1)} + \nu_i \left(\frac{\mathcal{J}^{\oplus}}{\max(n^{\oplus},1)} - \frac{\mathcal{J}^{\ominus}}{\max(n^{\ominus},1)} \right)$;
  end for
  $\mu_{j,i}^{(1)} \leftarrow \max\left(\min\left(\mu_{j,i}^{(1)}, \mu_{max}\right), \mu_{min}\right)$; ▷ Saturation
end for
until Stopping criterion

is based on extracted SNARX models none of which has the correct structure. All the same, this information appears to be sufficient to drive the algorithm toward the true model structure $\lambda^\star$. Indeed, some of the sub-periods have been already mapped on the correct mode with high confidence and the algorithm is looking for the model structure S on a restricted area of the solution space $\mathcal{S}$ which actually contains $S^\star$. This confirms the effectiveness of the chosen parametrization of $\mathbb{P}_{\boldsymbol{\gamma}}$ and of the proposed tuning rules. It proves also that the result in Theorem 2 is somewhat conservative, since in this example the algorithm is converging toward the target limit distribution even if it has been initialized with $\mathbb{P}_{\boldsymbol{\gamma}}(\lambda^\star) \cong 0$.

Table 5.1 reports some aggregate results obtained from 100 runs of the algorithm on the same data realization. The proposed algorithm performs well in both the sample-mode assignment and the local NARX model identification, and it does so by exploring a small fraction of the total number of possible switching signals and models. As for the nonlinear mode, the algorithm sporadically (2 times out of 100) fails to select the nonlinear term $y(t-1)^2$ in favor of $y(t-1)$, for a slight performance loss. Indeed, $\mathcal{L}^{(2)}$ takes the value 0.0119 for the wrong local model and the value 0.0118 for the correct one, causing the algorithm to be trapped in the found local minimum due to the almost negligible difference between them. It is worth noticing that despite the occasional failures in identifying mode 2, the algorithm has always been able to capture from the data the existence of two different modes, and to assign them correctly to the sub-periods.

A similar MC analysis has been carried out by considering this time the heuristic implementation introduced in Section 5.2.1. As one can note from Table 5.2, which reports the aggregated results of this analysis, the heuristic implementation provides comparable results in terms of accuracy, albeit at a lower computational cost. Figure 5.2 compares the two versions of the proposed algorithm, by enumerating the occurrences of a sign difference between the two update factors $\ell_{j,i}^{(l)}$ and $\tilde{\ell}_{j,i}^{(l)}$ over the MC runs. The frequency of these events decreases with iterations, so that no significant differences are expected in the algorithm outcomes at convergence. Based on this evidence, the

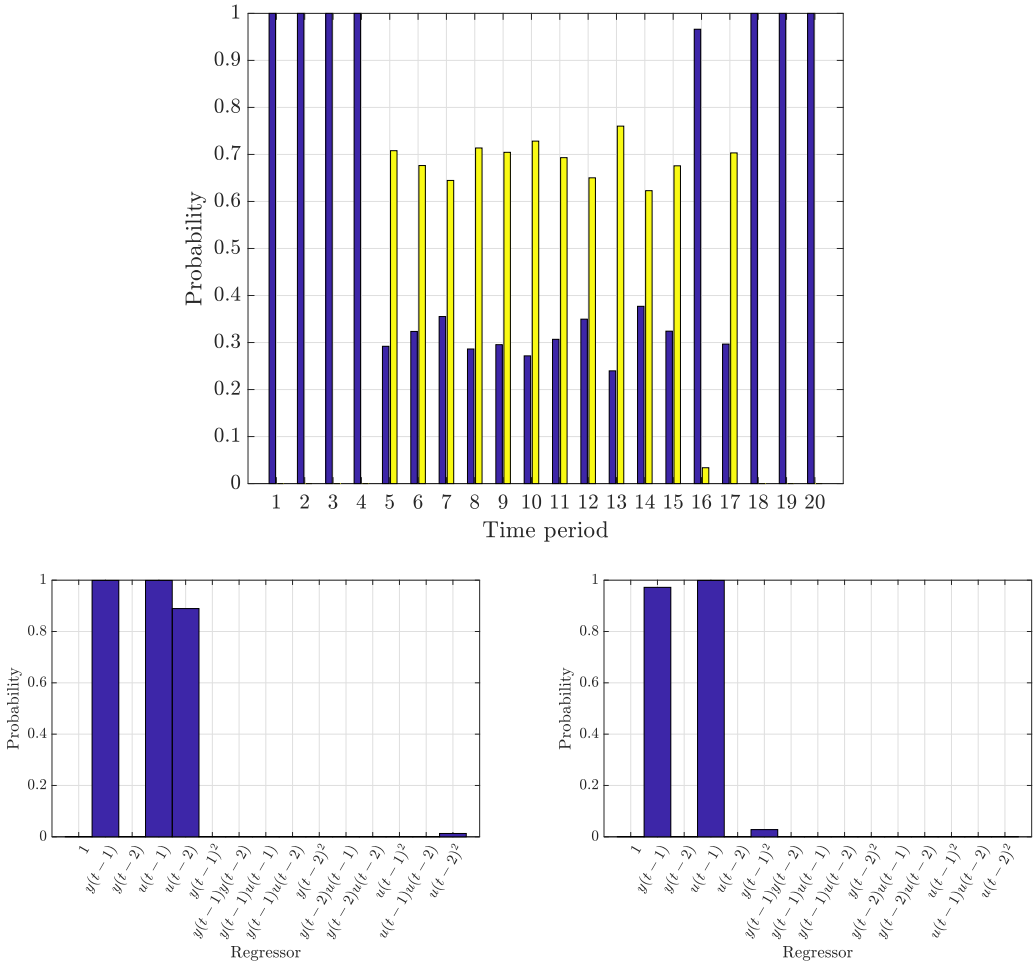

Figure 5.1: *Example 1: MEP and RIP values at the iteration when the correct model structure is first extracted. Top: MEPs of modes 1 (blue) and 2 (yellow). Bottom, from left to right: RIPs of mode 1, RIPs of mode 2.*

heuristic implementation has been employed in the rest of the paper for computational convenience.

Example 1 (contd.): $\mathcal{T}_s^\circ \nsubseteq \mathcal{T}_s$

Suppose now that the switchings occur at $t = 350$ (from mode 1 to mode 2), $t = 1450$ (from mode 2 to mode 1), $t = 1600$ (from mode 1 to mode 2), and $t = 1750$ (from mode 2 to mode 1). Notice that using the previously defined uniform placement of the switching times, only one of the true switchings is encompassed, while the others occur exactly in the middle of the 3rd, 15th, and 18th sub-periods.

Table 5.3 reports the results of a single run of the identification method. Apparently, the presence of sub-periods assigned to mode 1 but containing also samples associated to mode 2 prevents the algorithm from correctly identifying the local model assigned to the first mode (a redundant regressor is added to the model, although with a very small coefficient, indicating its relatively smaller importance). Despite this failure in estimating the linear local model, the method performs well in assigning the samples

Table 5.1: *Example 1: $\mathcal{T}_s^\circ \subseteq \mathcal{T}_s$. Monte Carlo simulation results.*

Average elapsed time [s]	41.25
Percentage of correct selection of $\boldsymbol{\kappa}$	100%
Average # of explored switching sequences	12520
Total # of allowed switching sequences	1048576
Percentage of correct selection of $\boldsymbol{s}^{(1)}$	100%
Average # of explored model structures for mode 1	790.62
Total # of possible model structures for mode 1	32768
Percentage of correct selection of $\boldsymbol{s}^{(2)}$	98%
Average # of explored model structures for mode 2	1005.9
Total # of possible model structures for mode 2	32768

Table 5.2: *Example 1: $\mathcal{T}_s^\circ \subseteq \mathcal{T}_s$. Monte Carlo simulation results - heuristic implementation.*

Average elapsed time [s]	30.16
Percentage of correct selection of $\boldsymbol{\kappa}$	100%
Average # of explored switching sequences	11156
Total # of allowed switching sequences	1048576
Percentage of correct selection of $\boldsymbol{s}^{(1)}$	100%
Average # of explored model structures for mode 1	646.27
Total # of possible model structures for mode 1	32768
Percentage of correct selection of $\boldsymbol{s}^{(2)}$	95%
Average # of explored model structures for mode 2	752.42
Total # of possible model structures for mode 2	32768

to the modes. Indeed, the obtained $\kappa^\star$ is correct in 17 out of 20 periods and yields a 50% correct classification of the samples in the remaining three sub-periods. This error (which involves 150 out of 2000 samples, *i.e.* 7.5% of the data) is unavoidable given the placement of the true switchings exactly in the middle of the allowed sub-periods.

Discussion

The presented first stage identification method is effective in both mode assignment and model estimation, provided that $\mathcal{T}_s^\circ \subseteq \mathcal{T}_s$, while an unavoidable approximation error is experienced otherwise. In general, no *a priori* information on the switching times is available and, in principle, a switching could occur at any time instant in $\{1, 2, \dots, N\}$. In order to encompass this case one could arbitrarily enlarge $\mathcal{T}_s$ towards $\{1, \dots, N\}$. However, the complexity of the resulting combinatorial problem rapidly increases with the cardinality of $\mathcal{T}_s$ that is employed, making it computationally intractable to sample the set $\{1, 2, \dots, N\}$ too densely. This poses a practical limit on the modeling accuracy that can be achieved with the method described in this section, since with a sparse $\mathcal{T}_s$ a poor resolution on the switching times is typically obtained, which in turn influences the quality of the identified models (that are tuned on data not fully belonging to the

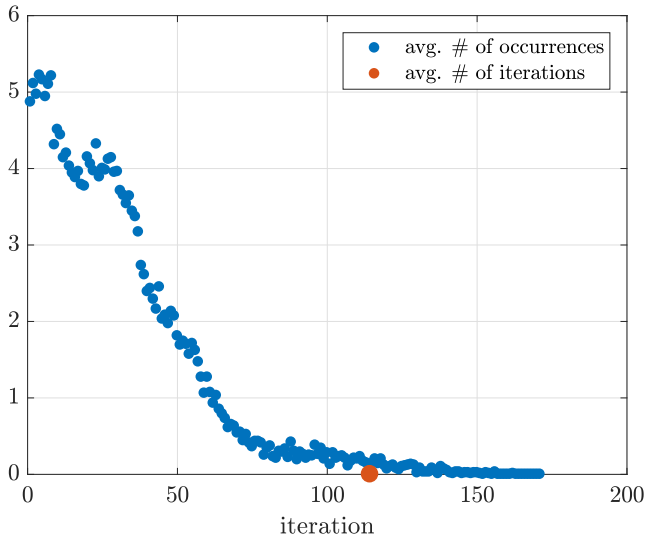

Figure 5.2: *Example 1: average number of occurrences of a sign difference between $\ell_{j,i}^{(l)}$ and $\tilde{\ell}_{j,i}^{(l)}$ in the MC runs at each iteration. The red marker indicates the average number of iterations required to solve the identification problem.*

Table 5.3: *Example 1: $\mathcal{T}_s^\circ \not\subseteq \mathcal{T}_s$. Single run results.*

$\mathcal{L}(\lambda)$	0.0154
$\mathcal{L}^{(1)}(\lambda)$	0.0198 ($N_1 = 900$)
$\mathcal{L}^{(2)}(\lambda)$	0.0119 ($N_2 = 1100$)
Detected switching times	$400,\ 1400,\ 1600,\ 1700$
sub-periods assigned to mode 1	I_k, $k = 1, \dots, 4, 15, 16, 18, \dots, 20$
sub-periods assigned to mode 2	I_k, $k = 5, \dots, 14, 17$
Sample classification error	7.5%
Regressors mode 1	$y(t-1),\ u(t-1),\ u(t-2)$
Parameters mode 1	$-0.9041,\ 0.8363,\ 0.0566$
Regressors mode 2	$u(t-1),\ y(t-1)^2$
Parameters mode 2	$0.5093,\ -0.4137$

appropriate modes), and motivates the introduction of the second stage.

5.2.2 Second stage: refinement

Rather than extending $\mathcal{T}_s$ to improve the accuracy of the model, we here suggest to refine it based on the outcome of the identification procedure and then iterate the process. The refinement stage is aimed at improving the resolution of $\mathcal{T}_s$ where required, at the same time keeping its size under control. This is achieved by adopting a denser sampling of the time horizon in the vicinity of the detected switchings and a sparser sampling elsewhere. Notice that, besides improving the resolution of the estimated switching instants, it is also expected that the improvement in the sample-mode assign-

ment will also positively impact the accuracy of the identified local models.

The rationale behind the refinement of $\mathcal{T}_s$ follows from the observations listed below:

- Let two adjacent sub-periods be assigned to different modes, say $\kappa_k = 1$ and $\kappa_{k+1} = 2$. This suggests that the majority of the samples of the first period can be ascribed to mode 1 and similarly that most of the samples in the second period indeed belong to mode 2. This indicates that there is at least one switching between modes 1 and 2 in the time interval spanned by the set $I_k \cup I_{k+1}$, but not necessarily at the common boundary (t_k). Therefore, adding new candidate switching times in the vicinity of t_k may improve the resolution of the algorithm.
- Let two adjacent sub-periods be assigned to the same mode, say $\kappa_k = \kappa_{k+1} = 1$. Then, in the same assumptions as before, no switching from mode 1 to another one can occur in the vicinity of the intermediate point t_k. It is therefore possible to disregard t_k altogether as a candidate switching time.
- Occasionally, the identification procedure may fail to converge to a limit distribution regarding a specific sub-period, so that multiple MEPs have non-zero values. This typically occurs when the sub-period contains data of different modes. In these situations, splitting further the sub-period into smaller sub-periods may facilitate the algorithm in taking its decisions.

Let $\mathcal{T}_s^{(r)}$ be the set of allowed switching time instants at the rth iteration of the overall procedure. Then, after the execution of the identification phase in the first stage, the refinement phase of the mode switching times consists in defining $\mathcal{T}_s^{(r+1)}$ based on the results of the rth identification. $\mathcal{T}_s^{(r+1)}$ is calculated according to the following steps, starting from an empty set:

1. *Detection of switchings*. A switching is detected at t_k if $\kappa_k \neq \kappa_{k+1}$ (*i.e.* two consecutive sub-periods have been assigned to different modes). Accordingly, let $\mathcal{V} = \{t_k \in \mathcal{T}_s^{(r)} | \kappa_k \neq \kappa_{k+1}\}$ be the set of detected switchings.
2. *Detection of unresolved sub-periods*. Sub-period I_k is marked as *unresolved* if the identification algorithm was unable to converge to a limit distribution for ξ_k, within the allotted iterations (although the MEP of one mode could still be significantly larger than the others to allow for a meaningful mode assignment). The auxiliary set $\mathcal{U} \subseteq \mathcal{T}_s^{(r)}$ includes the starting times of such unresolved sub-periods.
3. *Split phase: part a*. For each $t \in \mathcal{V}$, three candidate switching locations are added to $\mathcal{T}_s^{(r+1)}$. More precisely, $\mathcal{T}_s^{(r+1)} \leftarrow \mathcal{T}_s^{(r+1)} \cup \{t^-, t, t^+\}$, with $t^- = t - w$ and $t^+ = t + w$, where w is a design parameter.
4. *Split phase: part b*. For each $t \in \mathcal{U}$, let $t' = \min_{\{t_k \in \mathcal{T}_s^{(r)} | t_k > t\}} t_k$. Now, if $d = t' - t \geq 2$, then $\mathcal{T}_s^{(r+1)} \leftarrow \mathcal{T}_s^{(r+1)} \cup \{t, t^+, t'\}$, where $t^+ = t + \lceil \frac{d}{2} \rceil$. Otherwise, $\mathcal{T}_s^{(r+1)} \leftarrow \mathcal{T}_s^{(r+1)} \cup \{t, t'\}$.
5. *Merge phase*. The elements of $\mathcal{T}_s^{(r)}$ not in $\mathcal{V}$ or $\mathcal{U}$ are not carried over to $\mathcal{T}_s^{(r+1)}$, and are therefore discarded. By doing so, we are implicitly merging consecutive sub-periods, which are assumed not to include mode switchings, according to the current model.

Regarding the split procedure, a possible choice is to use the same w value for each detected switching, setting $w(r+1) = \alpha \min_k |I_k^{(r)}|$, where $I_k^{(r)}$, $k = 1, \ldots, N_s + 1$ are the sub-periods induced by $\mathcal{T}_s^{(r)}$ and $0 < \alpha < 1$ (*e.g.*, $\alpha = 0.5$ to get new sub-periods half as large as the smallest sub-periods of the previous iteration).

The rationale behind the processing of the unresolved sub-periods is as follows. Since the absence of convergence is typically due to the simultaneous presence in a sub-period of an initial portion associated to a mode followed by samples from a different one, the time interval is split into two equal parts to increase the mode unbalance in both time intervals and thus facilitate the mode assignment. However, if the original unresolved sub-period is too short, the time interval is not further divided, trusting that the progressive improvements in the identification of the local models (thanks to the refined positioning of the switchings) will allow the full convergence to a limit distribution in the subsequent iterations.

Guidelines for parameter settings

The results of the previous identification phase can also be used to set the initial MEPs and RIPs more appropriately before repeating the identification procedure. Indeed, if a sub-period was previously assigned to a specific mode with high confidence (*i.e.*, the corresponding MEP was close to 1 at the previous iteration), then this information should be preserved in the new execution, by setting the corresponding MEP to a large value. All the same, we apply a discounting factor to allow the identification algorithm some flexibility to consider also alternative mode assignments. On the other hand, the MEPs associated to unresolved sub-periods or to newly generated sub-periods (from t^- to t and from t to t^+) are set to be equal for all modes. The following rules formalize these considerations:

- *Detected switchings.* For each $t \in \mathcal{V}$, $\eta_{t^-}^{(i)} = \eta_t^{(i)} = 1/N_M$, $i = 1 \ldots N_M$, while $\eta_{t^+}^{(i)} = p$ for $i = \sigma_{t^+}$ and $\eta_{t^+}^{(j)} = \frac{1-p}{N_M - 1}$, for all other modes, where p is a design parameter (*e.g.*, $p = 0.7$) representing the desired confidence level.
- *Unresolved switchings.* For each $t \in \mathcal{U}$, $\eta_t^{(i)} = 1/N_M$, $i = 1 \ldots N_M$, while $\eta_{t'}^{(i)} = p$ for $i = \sigma_{t'}$ and $\eta_{t'}^{(i)} = \frac{1-p}{N_M - 1}$, for all other modes. Furthermore, if t^+ exists, $\eta_{t^+}^{(i)} = 1/N_M$, $i = 1 \ldots N_M$.

Regarding the RIPs, they are all set to $1/n$ at each iteration, where n is the number of regressors.

Example 1 (contd.): Refinement stage

A typical execution of the refinement stage is illustrated in Figure 5.3, as a continuation of the last example discussed in Section 5.2.1. In the identification stage, as already discussed, mode switchings were identified at times 400, 1400, 1600, and 1700, three of which being approximations of the true ones, given the coarse division of the time horizon in $\mathcal{T}_s$. The refinement stage halves the 4th, 5th, 14th, 15th, 16th, 17th, and 18th intervals ($w = 50$), and removes the redundant time points separating equal mode assignments, yielding $\mathcal{T}_s^{(2)} = \{350, 400, 450, 1350, 1400, 1450, 1550, 1600, 1650, 1700, 1750\}$. Notice that the total number of switching times has decreased from

19 to 11, thanks to the merging phase. The smallest time sub-periods generated by the refinement are initialized with MEPs assigning the same *a priori* probability to all modes, while the MEPs of the other ones (where a clear decision was made in the first run) are only partially discounted to allow some further flexibility to the algorithm. Notice that based on $\mathcal{T}_s^{(2)}$ a much sharper detection of the true switching times is indeed possible.

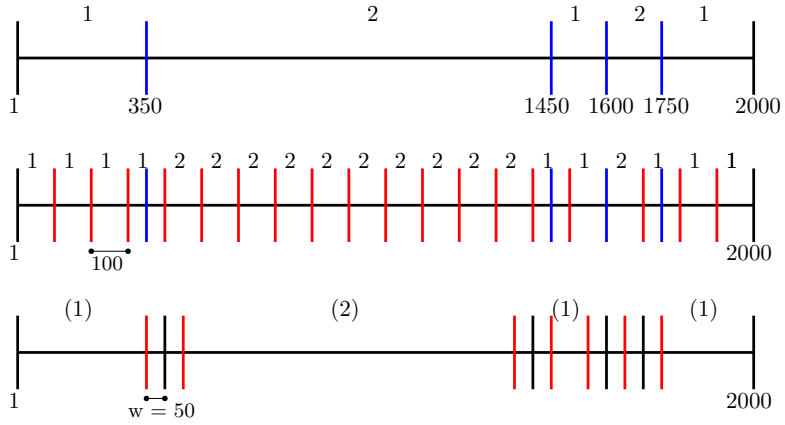

Figure 5.3: *Example 1, refinement stage: real switching signal (top), identified switching instants (middle) and updated set of allowed switchings (bottom). Blue bars indicate actual switchings, black bars the detected switching instants, and red bars the candidate switchings. The mode corresponding to each sub-period is reported on the top of each plot: modes indicated in brackets are those whose MEP will be set to a larger value in that sub-period for the next identification stage.*

5.3 Numerical examples

In this section several simulation examples are discussed to show the effectiveness of the proposed iterative method. First, the presented procedure is applied to the example introduced in Section 5.2.1 to illustrate the effect of repeatedly iterating stages 1 and 2 (Section 5.3.1). Some robustness and computational load analyses have also been carried out on the same example. A second system identification problem is discussed in Section 5.3.3, which is not trivial due to the presence of local models with the same structure but different parameterizations. A third, more complex case study is also considered.

All tests have been performed in a MATLAB 2017a environment, on an HP ProBook 650 G1 CORE i7-4702MQ CPU @2.20 GHz with 8GB of RAM.

5.3.1 Example 1 (contd.): Two-stage procedure

Let us apply the iterative two-stage procedure to the illustrative example discussed in Section 5.2.1. The design parameters of the identification phase, as well as the initial MEPs, RIPs, and candidate switching locations are set as done previously (20 sub-periods of 100 samples are initially defined). The design parameters for the refinement stage have been set to $\alpha = 0.5$ and $p = 0.7$.

Table 5.4 presents the aggregated results of 100 Monte Carlo (MC) runs. Notice, first of all, that both local model structures have been estimated correctly 100% of

the times. Furthermore, the low accuracy in the selection of the switching sequence selection (see Table 5.4) is only apparent, the errors in the estimation of the switching time instants being in fact rather small. This can be a appreciated by inspection of Figure 5.4 (top), which shows the distribution of the detected switchings over the MC runs, indicating that the number and position of the switchings are in fact quite accurately estimated, thanks to the refinement procedure. Figure 5.4 shows also the aggregated results in terms of classification error rate on the training set (percentage of misclassified samples), and the normalized accuracy criterion

$$FIT = 100\,(1 - \|\hat{\boldsymbol{y}} - \boldsymbol{y}\|_2 / \|\boldsymbol{y} - \bar{y}\mathbb{1}\|_2)\,, \tag{5.26}$$

where $\boldsymbol{y}$ is the vector containing the target outputs, $\bar{y}$ being the mean value, and $\hat{\boldsymbol{y}}$ is the vector of the outputs predicted using the estimated mode switching signal $\boldsymbol{\sigma}$.

With reference to the same example we also ran a comparative analysis with the non-parametric approach of [69], which extends the method presented in [63] from which the SNARX system used in this example has been taken. In particular, among the four methods proposed in [69] to fix the submodel size and limit the number of optimization variables, we chose the Feature Vector Selection (FVS) method. To describe the two modes we considered a linear kernel and a RBF kernel, respectively, exploiting (as done in [63]) the prior knowledge that one submodel is linear and the other is nonlinear. To produce the results presented in the paper we tested various combinations of the design parameters σ (the STD of the RBF kernels) and C (which governs the trade-off between model complexity and model accuracy), obtaining the best results for $\sigma = 0.1$ and $C = 100$. An MC analysis was carried out and the aggregate results are reported in Figure 5.5. The values of the FIT criterion are roughly in the same range as with the proposed algorithm, albeit with a much larger variance. However, the more striking difference is in the sample classification accuracy, which is significantly larger than with the proposed algorithm. This is a remarkable aspect, considering also that with the non-parametric approach we have taken advantage of the *a priori* knowledge about the linearity of one of the submodels. One reason for this performance difference lies in the fact that the non-parametric method operates on a sample-by-sample basis, resulting in a very fragmented mode mapping of the time history (unless some sort of post-processing is applied). This does not happen with our method, since it exploits the time-ordering of the collected data to solve the sample-mode mapping process, by applying a segmentation in a relatively small number of subperiods. In the light of the large classification error, the occasional high FIT models obtained with the non-parametric approach might be interpreted as a manifestation of overfitting behavior. Finally, the considered non-parametric approach on average required 123.9 seconds to solve the identification task.

5.3.2 Example 1 (contd.): Robustness and computational load analysis

To show the robustness of the proposed method with respect to the initial choice of the switching instants, a MC simulation was carried out on Example 1[2], initializing $\mathcal{T}_s^{(0)}$ randomly. Specifically, at each run the candidate switching instants are set to $t_k = 100k + v_k$, $k = 1, \ldots, 19$, where v_k is a zero mean white Gaussian noise with

[2]Where the values of the design parameters are not reported explicitly, those used in Section 5.3.1 are considered.

Table 5.4: *Example 1 (contd.): $\mathcal{T}_s^\circ \nsubseteq \mathcal{T}_s$. MC analysis.*

Average elapsed time [s]	284.26
Percentage of correct selection of $\boldsymbol{\kappa}$	61.62%
Average # of explored sequences	11791
Total # of allowed switching sequences	1048576
Percentage of correct selection of $\boldsymbol{s}^{(1)}$	100%
Average # of explored models for mode 1	654
Total # of possible model structures for mode 1	32768
Percentage of correct selection of $\boldsymbol{s}^{(2)}$	100%
Average # of explored model structures for mode 2	758
Total # of possible model structures for mode 2	32768

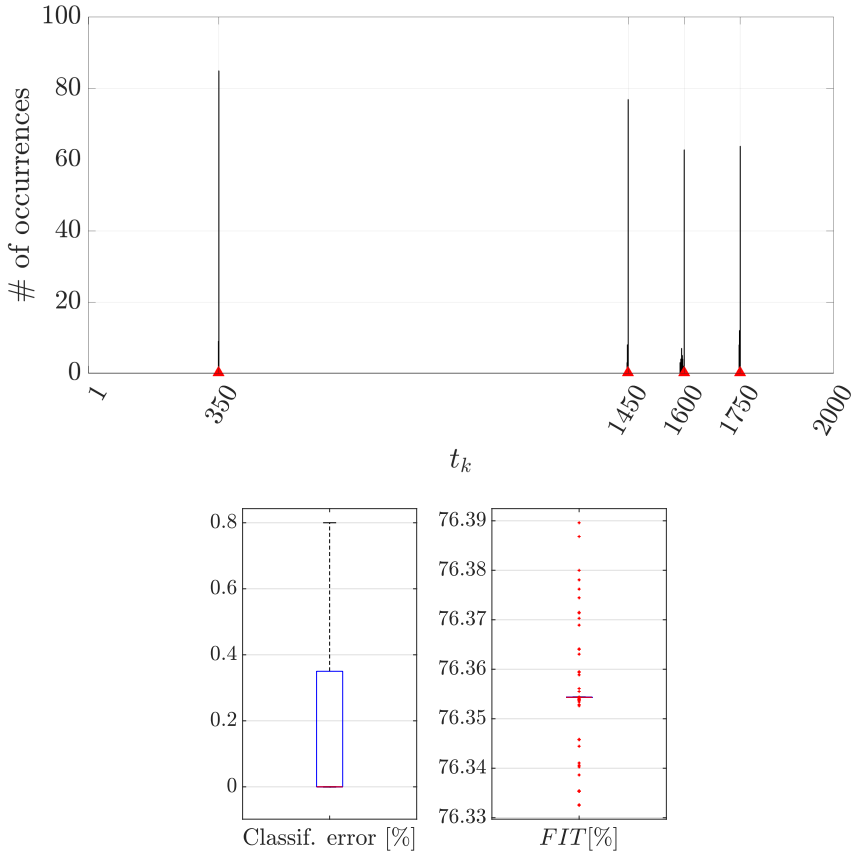

Figure 5.4: *Example 1 (contd.): $\mathcal{T}_s^\circ \nsubseteq \mathcal{T}_s$, proposed method. Top: distribution of the detected switching time instants for the proposed method (red markers represent the true switching instants). Bottom: boxplots showing the distributions of the classification error rate and the FIT criterion on the training set.*

standard deviation 10. As can be noticed from Figure 5.6, the classification error rate is generally below 1% and in any case lower than 5%, and the obtained distribution of the detected switching instants shows that the algorithm performs reasonably well in

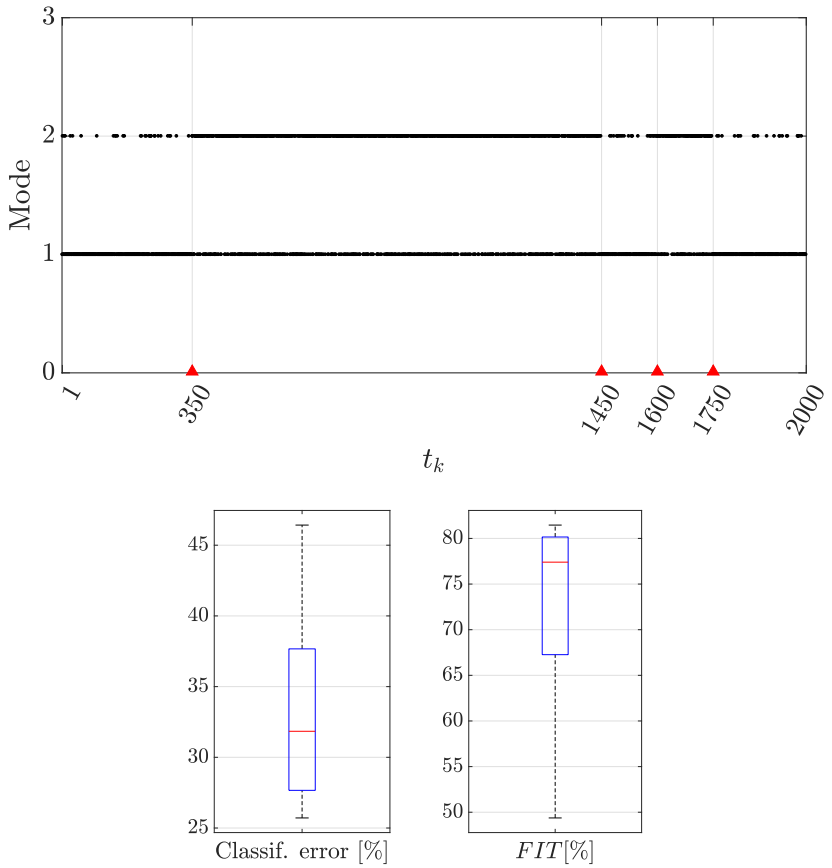

Figure 5.5: *Example 1 (contd.): $\mathcal{T}_s^\circ \nsubseteq \mathcal{T}_s$, non-parametric approach described in [69]. Top: Sample-mode mapping (single run). Bottom: boxplots showing the distributions of the classification error rate and the FIT criterion on the training set.*

the data segmentation task, leading to accurate models. Indeed, the overall accuracy as described by the FIT index is not distant from what found previously.

We also analyzed the robustness of the proposed approach as the noise level increases (using a fixed $\mathcal{T}_s^{(0)}$, with $t_k = 100k$, $k = 1, \ldots, 19$, as done originally). For each data realization (*i.e.* different SNR level), 10 runs were carried out, the aggregated results being summarized in Table 5.5. As expected, the performance of the method in terms of FIT decreases significantly as the noise variance increases. Interestingly enough, the classification error rate increases very slowly and remains well below 1% in all the examined range.

Finally, a computational load analysis for an increasing number of switchings was carried out, by analyzing data-sets of different length obtained from the system of Example 1. Assuming that the system switches between the two modes every 100 instants (starting from $\kappa_1^\circ = 1$), the number of switching instants t_k in $\mathcal{T}_s^\circ$ grows proportionally to N. As done previously, we initialized $\mathcal{T}_s^{(0)}$ randomly. An MC simulation was carried out by running the algorithm 30 times for each data realization with different random initializations of $\mathcal{T}_s^{(0)}$, and repeating for different N values. Figure 5.7 shows how the elapsed time varies with the number of switching time instants. As expected, this is the

most critical factor which affects the computational burden.

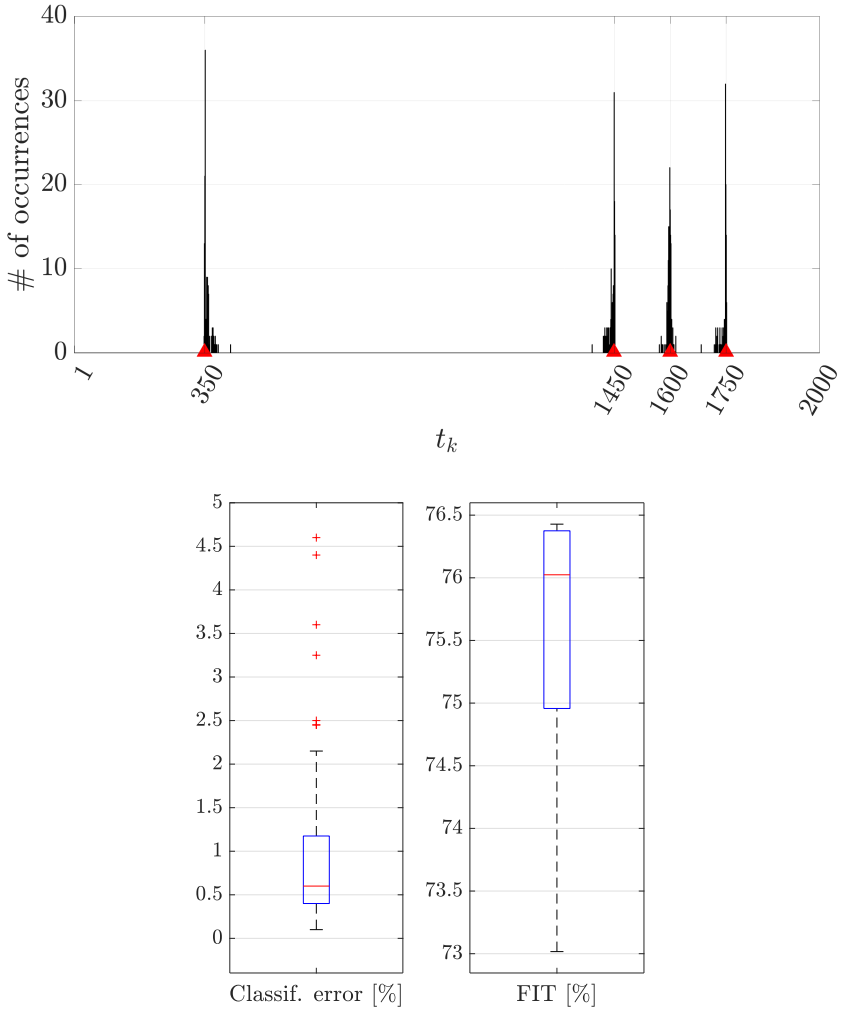

Figure 5.6: *Example 1 (contd.): robustness w.r.t. the initial choice of the switching instants. Distribution of the detected switching time instants (red markers represent the true switching instants), and boxplots demonstrating the classification error rate on the training set and the FIT criterion.*

Table 5.5: *Example 1 (contd.): robustness w.r.t. the noise level. MC analysis, mean values and variances.*

Noise σ	0.01	0.0422	0.0744	0.1067	0.1389	0.1711	0.2033
Train Cl. Err. [%]	0 (0)	0 (0)	0 (0)	0.044 (0.018)	0.23 (0.043)	0.34 (0.042)	0.34 (0.043)
FIT[%]	96.44 (1.36)	89.85 (0)	82.98 (0)	76.85 (2.97E-5)	71.59 (6.57E-5)	67.19 (2.84E-4)	63.54 (3.29E-4)

5.3.3 Example 2: a switching system with equal local model structures

The aim of this example is to assess how the method fares in the identification of the overall process model when the local models have the same structure. Consider thus

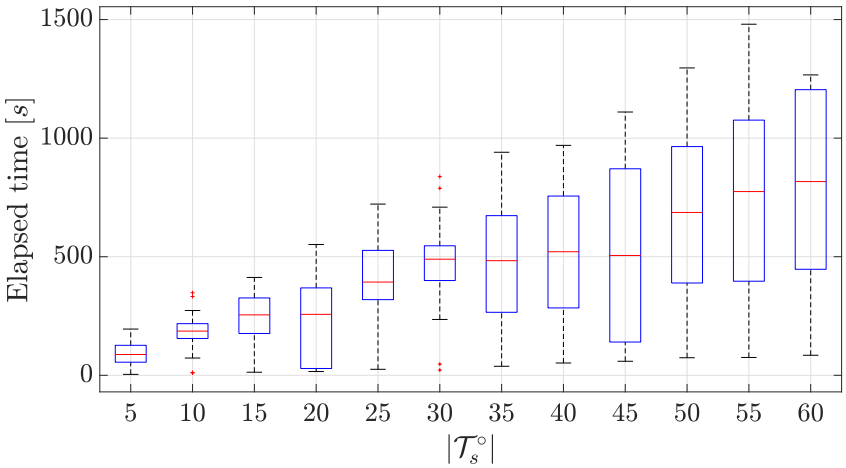

Figure 5.7: *Example 1 (contd.): elapsed time as a function of the number of switchings.*

the system presented in [51]:

$$y(t) = \vartheta^{(i)} y(t-1) - 0.7y(t-2) + u(t-1) - 0.5u(t-2) + e(t), \tag{5.27}$$

which consists of $N_M = 4$ local models, that are almost identical apart from one parameter that takes the values $\vartheta^{(1)} = -1.5$, $\vartheta^{(2)} = -1$, $\vartheta^{(3)} = -0.5$, and $\vartheta^{(4)} = 0.5$, respectively. The input signal $u(t)$ is a ± 1 Pseudo-Random Binary Sequence (PRBS), while the noise is an i.i.d. Gaussian process, $e(t) \sim \mathcal{N}(0, \sigma^2)$, with $\sigma = 0.5$. A dataset of 2500 input-output samples is available during which 6 mode switchings occur, according to $\mathcal{T}_s^\circ = \{400,\ 810,\ 1270,\ 1500,\ 1830,\ 2150\}$ and following the mode sequence $\boldsymbol{\kappa}^\circ = [1,\ 2,\ 3,\ 2,\ 3,\ 4,\ 1]$.

We compare our method with the SON-EM method described in [51], which turned out to fare well w.r.t. some of the latest developments in identification for linear switched systems (for details see [51]). Among others, the SON-EM outperforms (on the considered examples) the RANdom SAmple Consensus (RANSAC) method [38] which has been adapted in [51] for hybrid systems. In order to have a fair comparison, we here assume that the model structure of the modes is fixed as for the SON-EM method (the correct regressors $y(t-1)$, $y(t-2)$, $u(t-1)$, and $u(t-2)$, are employed and the NARX model structure selection part is skipped). Both methods address the estimation of all 4 parameters (not just $\vartheta^{(i)}$), for each mode. The initial set of candidate switching locations is defined as $\mathcal{T}_s^{(0)} = \{100, 200, \ldots, 2400\}$, inducing a uniform subdivision of the dataset in 25 sub-periods of 100 samples. The design parameters for the refinement stage are set to $\alpha = 0.5$ and $p = 0.25$.

An MC analysis has been carried out, running the algorithm 100 times on the same data realization. It turned out that 92% of the detected switching sequences contained the correct number of time instants. The distribution of the detected switching time instants for these sequences is reported in Figure 5.8. These results show that the proposed method performs well in detecting the switchings, in fact the best run yields $\mathcal{T}_s^\star = \{400,\ 797,\ 1250,\ 1500,\ 1830,\ 2146\}$ which proves to be quite close to the real one $\mathcal{T}_s^\circ$. Overall, the maximum and the mean sample classification error are respectively 5.96% and 1.54% for the MC runs resulting in a $\mathcal{T}_s^\circ$ with the correct cardinality.

In the remaining 8% of detected switching sequences, 6 of them missed only the switching at time $t = 400$, while the other 2 cases resulted in a completely wrong $\mathcal{T}_s^\star$.

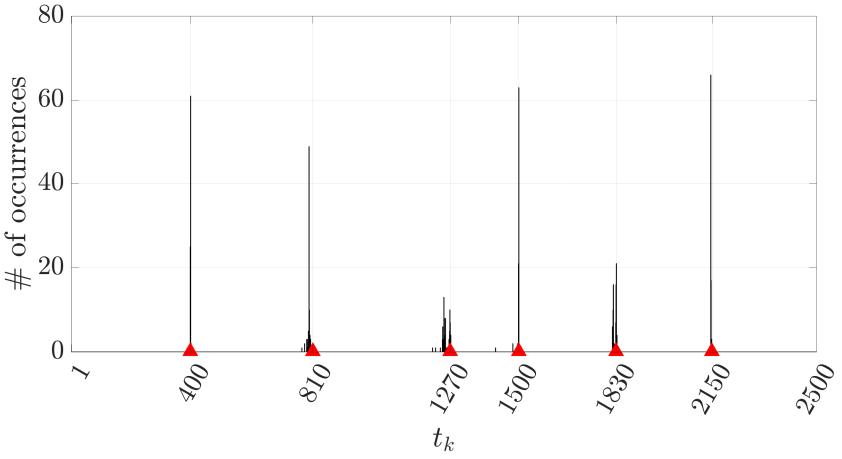

Figure 5.8: *Example 2: Distribution of the detected switching time instants (red markers represent the true switching instants).*

Figure 5.9 compares the estimates of $\vartheta^{(i)}$ on a single run obtained with the proposed method and the SON-EM method [51]. It is noteworthy that both methods captured well all the switching time instants and provided good parameter estimates, thus showing that the proposed method equals in terms of performance one of the most recent and promising methods. For the considered run, Table 5.6 reports the performance of the identified hybrid model at each iteration, the detected switchings, the sample-mode classification for each sub-period and the corresponding classification error on the training set. From a computational complexity viewpoint, we compared the two methods in terms of the time required to solve the identification task. It turned out that our method is more demanding w.r.t. the SON-EM, *i.e.,* on average our method lasted 238.56 seconds against 20.6063.

Finally, we compared the two methods also in terms of accuracy in one-step-ahead prediction and open-loop output simulation on the training data. Regarding the open-loop output simulation, the FIT values have been computed by considering the simulated response as $\hat{\boldsymbol{y}}$ in (5.26). It resulted that the two methods have similar capabilities, both in prediction and simulation, thus further proving the effectiveness of the proposed approach. Specifically, the obtained values for the SON-EM method are $FIT^{pred} - train = 74,1933$ and $FIT^{sim} - train = 46.0552$, whereas the proposed approach yields $FIT^{pred} - train = 72.9288$ and $FIT^{sim} - train = 45.2753$.

Table 5.6: *Example 2: Performance of a single run over iterations.*

r	$\mathcal{L}^{(1)}$	$\mathcal{L}^{(2)}$	$\mathcal{L}^{(3)}$	$\mathcal{L}^{(4)}$	$\mathcal{L}$	t_1	t_2	t_3	t_4	t_5	t_6	t_7	$\boldsymbol{\kappa}$	Error
1	0.2467	0.9951	0.2838	0.4848	0.4569	400	800	1300	1500	1800	2100	2200	[1,2,3,2,3,4,2,1]	6.8%
2	0.2469	0.2603	0.3217	0.2459	0.2742	400	800	1250	1500	1850	2150	–	[1,2,3,2,3,4,1]	1.4%
3	0.2469	0.2275	0.2615	0.2934	0.2527	400	800	1275	1500	1825	2150	–	[1,2,3,2,3,4,1]	0.8%
4	0.2469	0.2275	0.2615	0.2934	0.2527	400	800	1275	1500	1825	2150	–	[1,2,3,2,3,4,1]	0.8%
5	0.2469	0.2315	0.2542	0.2934	0.2513	400	813	1269	1500	1825	2150	–	[1,2,3,2,3,4,1]	0.36%
6	0.2529	0.2315	0.2542	0.2951	0.2533	400	813	1269	1500	1826	2147	–	[1,2,3,2,3,4,1]	0.48%
7	0.2829	0.2317	0.2570	0.2735	0.2514	400	810	1269	1500	1827	2147	–	[1,2,3,2,3,4,1]	0.28%

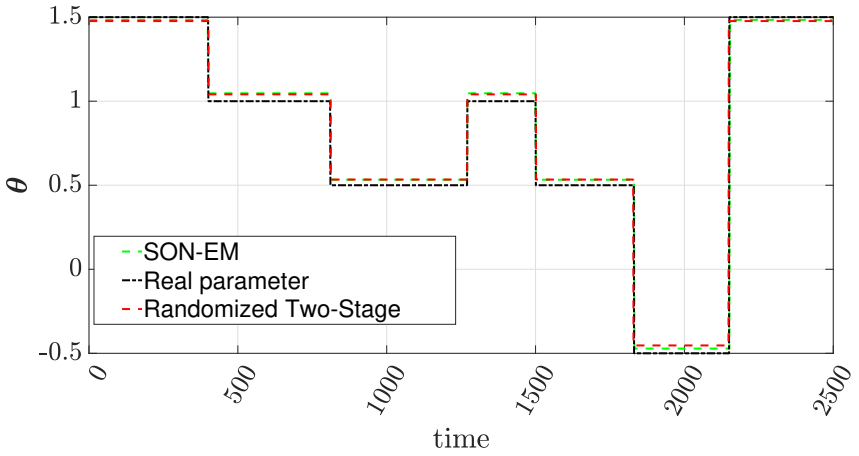

Figure 5.9: *Example 2: Identification of parameter $\vartheta^{(i)}$ with the proposed method and the SON-EM method.*

5.3.4 Example 3: A 3-mode SNARX case, with nonlinear modes

In this study, the following system has been considered:

$$\begin{aligned}
\text{mode } 1 : y(t) &= 0.5y(t-1) + 0.8u(t-2) \\
&\quad + u(t-1)^2 - 0.3y(t-2)^2 + e(t) \\
\text{mode } 2 : y(t) &= 0.2y(t-1)^3 - 0.5y(t-2) \\
&\quad - 0.7y(t-2)u(t-2)^2 + 0.6u(t-2)^2 + e(t) \\
\text{mode } 3 : y(t) &= 0.4y(t-1)^3 + 0.5y(t-2) \\
&\quad - 0.7y(t-2)u(t-2)^2 + 0.6u(t-2)^2 + e(t)
\end{aligned}$$

where $e(t)$ is a zero mean Gaussian noise of variance 0.01 and $u(t)$ is uniformly distributed in the interval $[-1, 1]$. Notice that two of the three nonlinear local models have the same model structure (but one different parameter). An observation window of $N = 3400$ samples has been collected, which contains 5 switchings, at locations $\mathcal{T}_s^\circ = \{500, 1030, 2115, 2740, 3000\}$, and corresponding to the mode switching sequence $\kappa^\circ = [1, 2, 1, 3, 2, 3]$.

An MC analysis has been carried out considering an initial set of candidate switchings defined as $\mathcal{T}_s^{(0)} = \{200, 400, \ldots, 3200\}$, which induces 17 sub-periods of 200

samples. Furthermore, the initial MEPs are all set to 0.33, and the initial RIPs to $1/n = 0.0061$. Regarding the NARX model structure selection, the candidate regressor set is defined by $n_d = 3$, $n_y = n_u = 4$, which makes it abundantly oversized (the model orders are overestimated), amounting to $n = 165$ regressors. Finally, $\alpha = 0.5$ and $p = 0.7$, for the refinement stage.

Table 5.7 reports the aggregated results of 50 MC runs. Apparently, the model structures of all the modes have been detected with a quite high accuracy (over 94%), despite the large combinatorial complexity of the involved model selection problems. Furthermore, Figure 5.10 illustrates the robustness of the algorithm in estimating the switching locations. Indeed, in the best case, a $\mathcal{T}_s^\star = \{499,\ 1029,\ 2112,\ 2739,\ 2998\}$ was obtained, whereas an error of only 1.6% was obtained regarding the sample classification in the worst run of the MC study.

Table 5.8 reports the results of a single run, indicating specifically the performance of the identified hybrid model at each iteration, the detected switchings, the sample-mode classification (for each sub-period) and the corresponding percentage error. Furthermore, Table 5.9 reports for each mode the percentage of misclassified samples. As can be noticed, the first identification stage results in an inaccurate model mainly because of the initial coarse uniform placement of the switching candidate time instants, which leads to a large sample classification error mainly for the first and third mode (see $r = 1$ in Table 5.9). The algorithm adapts the structure selection by extracting the correct terms plus some extra ones in order to take into account for the misclassified samples (see $r = 1$ in Table 5.10). Notwithstanding this, the first identified switching signal $\boldsymbol{\sigma}$ is already close to the real discrete dynamics. The subsequent refinement stages (and the identification phases) progressively improve both the local and the global performance leading to a very accurate final hybrid model (both in terms of the continuous and the discrete dynamics). It is apparent that as the switching signal is more accurately estimated, the accuracy of the local models also improves, since they are estimated on more appropriate data sets. Indeed, from the fourth iteration on the sample classification errors are lower than 1% for all modes (see Table 5.9) and the extracted structures are correct (see Table 5.10).

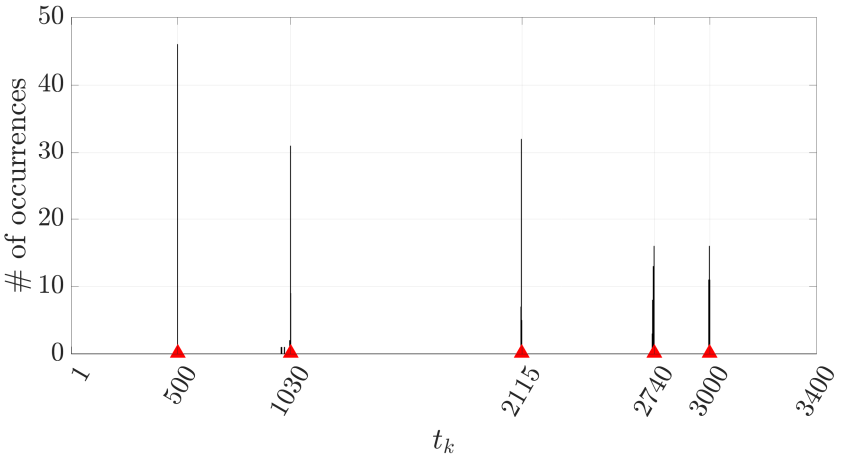

Figure 5.10: *Example 3: Distribution of the detected switching time instants (red markers represent the true switching instants).*

Table 5.7: *Example 3: MC analysis.*

Average elapsed time [s]	1247
Percentage of $\boldsymbol{\kappa}$ of correct length	100%
Average # of explored sequences	5042
Total # of allowed switching sequences	131072
Percentage of correct selection of $\boldsymbol{s}^{(1)}$	94%
Average # of explored models for mode 1	6634.3
Total # of possible model structures for mode 1	$4.6768{\cdot}10^{49}$
Percentage of correct selection of $\boldsymbol{s}^{(2)}$	96%
Average # of explored model structures for mode 2	6377.2
Total # of possible model structures for mode 2	$4.6768{\cdot}10^{49}$
Percentage of correct selection of $\boldsymbol{s}^{(3)}$	94%
Average # of explored model structures for mode 3	6144.4
Total # of possible model structures for mode 3	$4.6768{\cdot}10^{49}$

Table 5.8: *Example 3: Performance over iterations on a single run.*

r	$\mathcal{L}^{(1)}$	$\mathcal{L}^{(2)}$	$\mathcal{L}^{(3)}$	$\mathcal{L}$	t_1	t_2	t_3	t_4	t_5	$\boldsymbol{\kappa}$	Error
1	0.0395	0.0097	0.0117	0.0257	600	1000	2200	2800	3000	[1,2,1,3,2,3]	7.86%
2	0.0274	0.0097	0.0106	0.0190	550	1000	2150	2750	3000	[1,2,1,3,2,3]	3.57%
3	0.0179	0.0097	0.0107	0.0138	525	1025	2125	2750	3000	[1,2,1,3,2,3]	2.86%
4	0.0135	0.0104	0.0097	0.0116	512	1025	2112	2737	3000	[1,2,1,3,2,3]	0.66%
5	0.0121	0.0105	0.0097	0.0110	505	1032	2112	2737	3000	[1,2,1,3,2,3]	0.37%
6	0.0110	0.0104	0.0097	0.0104	501	1028	2112	2737	3000	[1,2,1,3,2,3]	0.26%
7	0.0098	0.0106	0.0096	0.0098	499	1030	2114	2739	3000	[1,2,1,3,2,3]	0.09%

Table 5.9: *Example 3: Sample classification error over iterations on a single run.*

r	Mode 1	Mode 2	Mode 3
1	11.94%	0%	5.45%
2	8.85%	0%	0.91%
3	2.42%	0%	0.89%
4	1.06%	0%	0.89%
5	0.32%	0.63%	0.27%
6	0.19%	0.38%	0.27%
7	0%	0.25%	0.09%

Table 5.10: *Example 3: Model structure selection over iterations (true regressors in bold face).*

r	Regressors of mode 1	Regressors of mode 2	Regressors of mode 3
1	$\boldsymbol{y(t-1)}$, $\boldsymbol{u(t-2)}$, $y(t-1)u(t-2)$, $\boldsymbol{y(t-2)^2}$, $y(t-3)u(t-2)$, $\boldsymbol{u(t-1)^2}$, $u(t-2)^2$, $u(t-2)u(t-3)$, $u(t-2)u(t-4)$, $y(t-2)u(t-2)u(t-4)$, $y(t-2)u(t-4)^2$, $y(t-3)^2u(t-2)$, $y(t-4)^2u(t-2)$, $u(t-2)^3$, $u(t-2)u(t-3)^2$	$\boldsymbol{y(t-2)}$, $\boldsymbol{u(t-2)^2}$, $\boldsymbol{y(t-1)^3}$, $\boldsymbol{y(t-2)u(t-2)^2}$	$\boldsymbol{y(t-2)}$, $y(t-2)y(t-4)$, $\boldsymbol{u(t-2)^2}$, $\boldsymbol{y(t-1)^3}$, $y(t-2)y(t-4)^2$, $\boldsymbol{y(t-2)u(t-2)^2}$, $u(t-4)^2$
2	$\boldsymbol{y(t-1)}$, $\boldsymbol{u(t-2)}$, $y(t-1)u(t-2)$, $\boldsymbol{y(t-2)^2}$, $y(t-3)u(t-2)$, $\boldsymbol{u(t-1)^2}$, $u(t-2)u(t-3)$, $u(t-2)u(t-4)$, $y(t-2)u(t-4)^2$, $y(t-3)^2u(t-2)$, $y(t-4)^2u(t-2)$, $u(t-2)^3$, $u(t-2)u(t-3)^2$	$\boldsymbol{y(t-2)}$, $\boldsymbol{u(t-2)^2}$, $\boldsymbol{y(t-1)^3}$, $\boldsymbol{y(t-2)u(t-2)^2}$	$\boldsymbol{y(t-2)}$, $\boldsymbol{u(t-2)^2}$, $\boldsymbol{y(t-1)^3}$, $\boldsymbol{y(t-2)u(t-2)^2}$
$3\cdots7$	$\boldsymbol{y(t-1)}$, $\boldsymbol{u(t-2)}$, $\boldsymbol{y(t-2)^2}$, $\boldsymbol{u(t-1)^2}$	$\boldsymbol{y(t-2)}$, $\boldsymbol{u(t-2)^2}$, $\boldsymbol{y(t-1)^3}$, $\boldsymbol{y(t-2)u(t-2)^2}$	$\boldsymbol{y(t-2)}$, $\boldsymbol{u(t-2)^2}$, $\boldsymbol{y(t-1)^3}$, $\boldsymbol{y(t-2)u(t-2)^2}$

5.4 A case study

The proposed identification procedure was applied to the data collected from the experimental setup introduced in [57] to simulate a component placement process operated by a pick-and-place machine. This machine is used to automatically place electronic components on printed circuit boards (PCBs). Specifically, assuming that the mounting head which carries the component is positioned correctly above the PCB, it moves downwards until the component impacts with the PCB, then releases the component and returns to the upper retracted position. The PCB exhibits certain elasticity properties, depending on the material. This process is characterized by switchings between different modes of operation. This motivates the interest in searching for the model in the form of a hybrid system as discussed in [10, 57, 58]. In particular, in [57], the dynamics of the experimental setup show four different operational modes:

upper saturation: the mounting head is in the upper retracted position, *i.e.,* it cannot move upward;

free mode: the mounting head is neither in contact with the PCB, nor in the upper retracted position;

impact mode: the mounting head is in contact with the PCB, but it is not in the lower extended position;

lower saturation: the mounting head is in the lower extended position, *i.e.,* it cannot move downward.

Note that the switch between the impact and free modes does not occur always at the same head position, because of the movement of the PCB. A physical model of the experimental setup is reported in Figure 5.11. The mounting head is represented by the mass M. The springs c_1 and c_2 simulate elasticity. The linear and dry frictions are represented respectively by (d_1, d_2), and (f_1, f_2). The system input is the voltage applied to the motor that drives the mounting head, represented by the force F. The input signal can be chosen in a way that only the operational modes of interest are sufficiently excited. The output of the system is the position of the mounting head.

5.4.1 Piecewise ARX (PWARX) identification

The presented system is better represented by a piece-wise (nonlinear) autoregressive exogenous (PW[N]ARX) model, than a S[N]ARX, due to the switching mechanism.

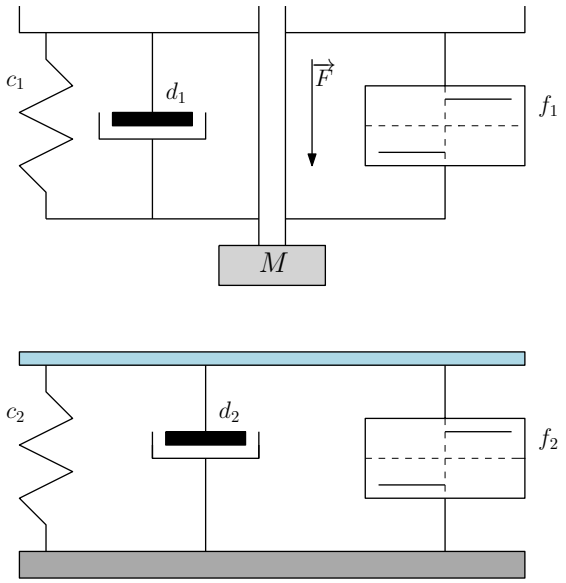

Figure 5.11: *Case study: pick-and-place machine, physical model.*

Indeed, the system switches between modes according to the current continuous state, rather than according to a time-dependent finite-valued signal. Accordingly, one must perform a partition of the state-input domain (or, in our case, of the regressors domain) into regions, in order to complete the identification. More in general, the region estimation is required also in case of a switched system whenever one wants to validate the identified model on an unseen data set.

Recall that in the NARX modeling framework, the nonlinear mapping g is often represented as:

$$g(\boldsymbol{x}(t);\boldsymbol{\vartheta}) = \boldsymbol{\varphi}(t)\boldsymbol{\vartheta} = \sum_{j=1}^{n} \vartheta_j \varphi_j(t),$$

where $\boldsymbol{\varphi}(t) : \mathcal{X} \subseteq \mathbb{R}^{n_y+n_u} \to \mathcal{F} \subseteq \mathbb{R}^n$, with n_y and n_u being suitable maximum lags for the output and input signals, respectively. In particular, for nonlinear polynomial expansions, for a given maximum degree n_d of the polynomial, the regressor set $\mathcal{F}$ has dimensionality $n = \frac{(n_d+n_y+n_u)!}{n_d!(n_y+n_u)!}$.

The identified finite-valued switching signal $\boldsymbol{\sigma}$ implicitly induces the classification of the N data points $(y(t), \boldsymbol{\varphi}(t))$ into N_M clusters $\mathcal{C}^{(i)} = \{(y(t), \boldsymbol{\varphi}(t)) : \sigma(t) = i\}$, $i = 1, \ldots, N_M$ and $t = 1, \ldots, N$. Accordingly, the corresponding sets of regression vectors are defined as:

$$\mathcal{R}^{(i)} = \{\boldsymbol{\varphi}(t) : (y(t), \boldsymbol{\varphi}(t)) \in \mathcal{C}^{(i)}\},\ i = 1, \ldots, N_M,\ t = 1, \ldots, N \tag{5.28}$$

The region estimation problem consists in finding a complete polyhedral partition $\{\mathcal{F}^{(i)}\}_{i=1}^{N_M}$ of the regressor set $\mathcal{F}$ such that $\mathcal{R}^{(i)} \subseteq \mathcal{F}^{(i)}$ for all $i = 1, \ldots, N_M$. Each polyhedral region $\mathcal{F}^{(i)}$ is defined by means of a set of linear inequalities, as follows:

$$\mathcal{F}^{(i)} = \{\boldsymbol{\varphi} \in \mathbb{R}^n : H^{(i)}\boldsymbol{\varphi} \preceq 0\}, \tag{5.29}$$

where $H^{(i)} \in \mathbb{R}^{q_i \times n}$, with q_i being the number of linear equalities defining the i-th region, and $\preceq$ denotes component-wise inequality. Therefore, the region estimation problem involves finding a set of hyperplanes $\mathcal{H}$:

$$\mathcal{H} = \{\boldsymbol{\varphi} \in \mathbb{R}^n : H_j^{(i)}\boldsymbol{\varphi} = 0\},\ i = 1, \ldots, N_M,\ j = 1, \ldots, q_i. \tag{5.30}$$

A linear classifier (hyperplane) separating without errors the samples in $\mathcal{R}^{(i)}$ from those in $\mathcal{R}^{(j)}$, with $i \neq j$, does not always exist if the two sets have intersecting convex hulls, *e.g.,* due to possible clustering errors. Therefore, the goal is to find a separating hyperplane which minimizes the number of misclassified samples or a suitable cost function associated with the errors.

In general, the linear separation of the N_M sets $\mathcal{R}^{(1)}, \ldots, \mathcal{R}^{(N_M)}$ can be accomplished with two different approaches:

1. Constructing a linear classifier for each pair $\left(\mathcal{R}^{(i)}, \mathcal{R}^{(j)}\right)$, with $i \neq j$. This amounts to solving $\frac{N_M(N_M-1)}{2}$ two-class classification problems with classical *e.g.,* support vector machines (SVM) [30] or robust linear programming (RLP) [11], and then eliminating *a posteriori* possible redundant hyperplanes. This approach is computationally attractive since each classification problem involves only the data corresponding to the considered pair $\left(\mathcal{R}^{(i)}, \mathcal{R}^{(j)}\right)$. However, the resulting regions are not guaranteed to form a complete polyhedral partition of $\mathcal{F}$.

2. Constructing a piecewise linear classifier which discriminates among N_M classes. A possible way to tackle this problem is to solve N_M two-class classification problems, each of which is aimed at separating the samples in $\mathcal{R}^{(i)}$ from those in all the remaining sets, *i.e.,* according to a *one-vs-rest* strategy [102]. In this case, the N_M two-class classification sub-problems can be tackled by RLP or SVM, as before. Another option is to search directly for the piecewise linear function by solving a single optimization problem. This can be accomplished by *e.g.,* multicategory SVM (M-SVM) [27] or multicategory RLP (M-RLP) [12]. The multiclass linear separation approach 2 is more computationally expensive than approach 1 since it involves all data at once, but it results in a complete partition of $\mathcal{F}$.

For a more comprehensive review of two-class and multiclass classification techniques, see [93, 118].

5.4.2 Identification with free and impact modes

A data record of 15 s is available, in which only the *impact* and *free* modes have been excited. The waveforms in the experimental setup were sampled at 4 kHz. The data used for identification purposes are obtained by re-sampling the original signals at 50 Hz. The resulting data set consists of 750 points, divided in two overlapping sets of 500 points: the former $[0, \ldots, 500]$ is used for identification, while the latter $[250, \ldots, 750]$ is used for validation of the identified models (see Figure 5.12). In all the analysis which follows, the design parameters for the identification and refinement stages have been set to $N_p = 100$, initially $t_k = 5k$ with $k = 1, \ldots, 99$, the initial MEPs are all set to $1/N_M$, the initial RIPs to $1/n$, $\alpha = 0.5$, and $p = 0.7$. Once the whole identification phase ended, the region estimation task has been carried out by tackling the problem with SVMs and a *one-vs-rest* strategy, and the identified model has been tested on the

validation data.

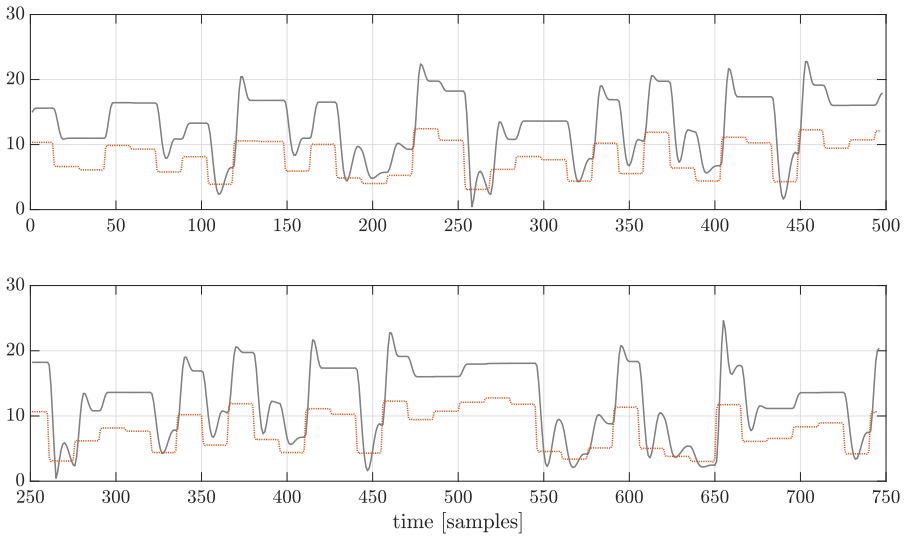

Figure 5.12: *Case study: data set used for identification (top) and validation (bottom). The solid and dashed lines represent the system output (position of the mounting head) and the scaled input (voltage applied to the motor), respectively.*

Switched models with two modes have been identified by considering different settings for the model orders. Table 5.11 reports the identified models and the FIT criterion (5.26) evaluated on both the training and validation data. For the first model m_1 the parameters $n_d = 1$, $n_y = 2$, and $n_u = 1$ have been used. The one-step-ahead prediction performance on the validation data set is shown in Figure 5.13. The difference between the system response and the model prediction is small and visible only in some time periods, *e.g.,* from 200 to 300, as experienced also in [57]. In this period, the system response seems to be unaffected by the variations of the input, particularly around $t = 250$, mainly due to the presence of the friction. Conversely, the predicted response exhibits small steps. As for the reconstructed active modes, they apparently capture well the physical operational modes. Indeed, mode 1 and mode 2 can be respectively mapped on the impact and free modes.

A second model m_2 has been identified with $n_d = 1$, $n_y = n_u = 2$. Note from Table 5.11 that the term $u(t-2)$ has not been included in any submodel, and that the identified model is close to m_1. This reflects on the obtained predicted response and reconstructed switching signal, that do not exhibit significant differences from that corresponding to model m_1, see Figure 5.14. As for m_1, mode 1 and mode 2 can be respectively mapped on the impact and free modes.

An analysis has been carried out by considering also a switched model m_3 with three linear modes, and $n_y = 2$, $n_u = 1$, with the hope of improving the identification by modeling also the transition between the impact and free modes. However, by looking at the $FIT - test$ value reported in Table 5.11, the further mode does not bring any significant improvements w.r.t. the previous models with two modes. Rather, it hampers the physical interpretation of the reconstructed switching signal, indeed now the impact

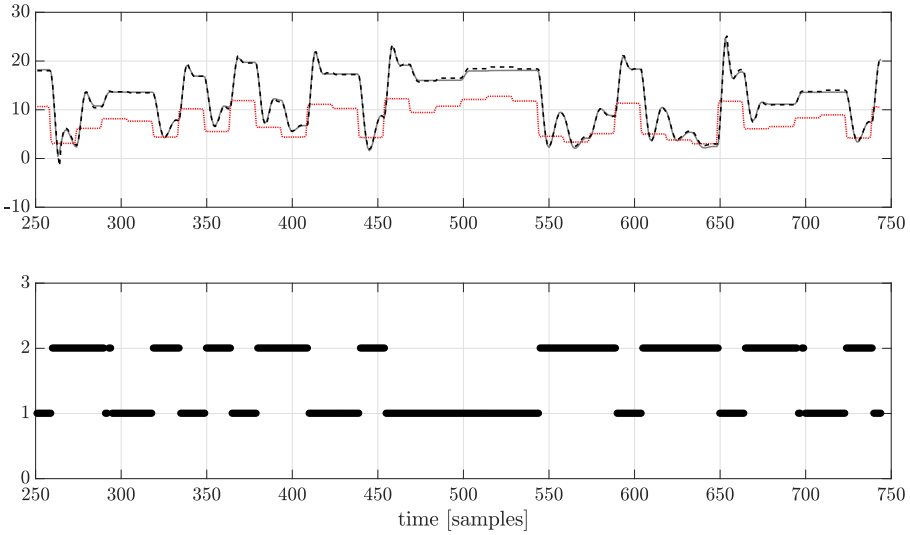

Figure 5.13: *Case study: validation of model m_1. (Top) One-step-ahead prediction for the validation data (solid: system response, dashed: model prediction, dotted: input). (Bottom) Active mode at each time instant.*

mode seems to be mapped indifferently on modes 2 and 3 (*e.g.,* see the time interval $(200, 300)$ in Figure 5.15). This proves that there is no evidence in the data supporting the need of a switched model with more than two modes.

To compare our results with those reported in [57], model m_1 has been identified a second time fixing the ARX model structures so as to include all the $n = 4$ model terms, see Table 5.12.

The results of the open-loop output simulation of the identified models are shown in Figure 5.17. For all models, the response is still similar to the one of the real system, but now differences are more evident. In particular, at the time interval $(200, 300)$, the discrepancy between the system response and the simulated one is even more significant than in the prediction case discussed before with reference to the model m_1. This further confirms the difficulties of the identified models, as already experienced in [57], in suitably reproducing the system behavior in this interval. The $FIT - test$ values, computed by considering the simulated response as $\hat{\boldsymbol{y}}$ in (5.26), are $76,61\%$, $77,83\%$, and $79,36\%$, respectively for the identified models m_1, m_2, and m_3.

Table 5.11: *Case study: identified models.*

Model	Parameters	Identified model	$FIT - train$ [%]	$FIT - test$ [%]
m_1	$N_M = 2,\ n_d = 1,\ n_y = 2,\ n_u = 1$	$\hat{y}^{(1)}(t) = 0.594 + 1.324y(t-1) - 0.613u(t-1) - 41.763u(t-2)$ $\hat{y}^{(2)}(t) = 1.514y(t-1) - 0.728y(t-2) - 34.579u(t-1)$	94.71	93.86
m_2	$N_M = 2,\ n_d = 1,\ n_y = n_u = 2$	$\hat{y}^{(1)}(t) = 0.625 + 1.344y(t-1) - 0.626y(t-2) - 39.614u(t-1)$ $\hat{y}^{(2)}(t) = 1.509y(t-1) - 0.735y(t-2) - 37.65u(t-1)$	94.66	94.12
m_3	$N_M = 3,\ n_d = 1,\ n_y = 2,\ n_u = 1$	$\hat{y}^{(1)}(t) = -0.226 + 1.579y(t-1) - 0.784y(t-2) - 37.222u(t-1)$ $\hat{y}^{(2)}(t) = 1.29y(t-1) - 0.604y(t-2) - 51.61u(t-1)$ $\hat{y}^{(3)}(t) = 0.796 + 1.39y(t-1) - 0.643y(t-2) - 32.622u(t-1)$	95.64	94.13

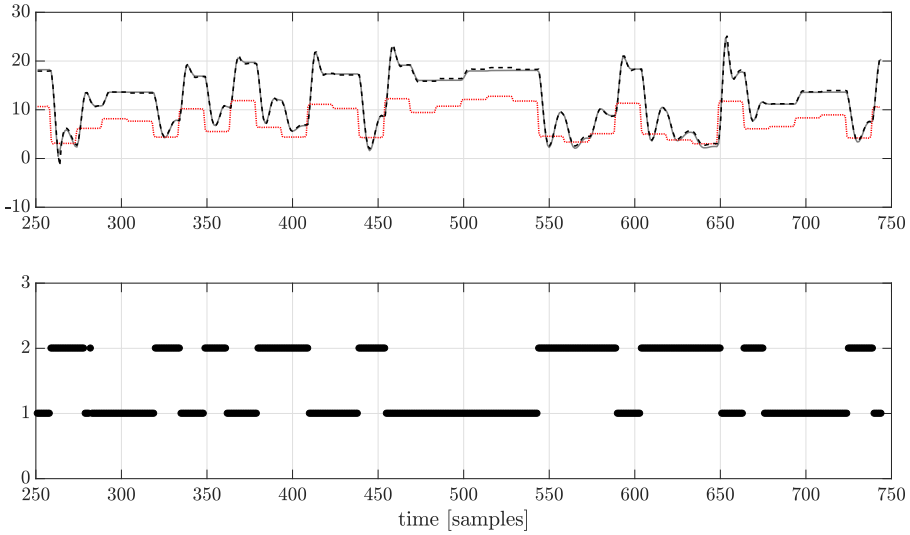

Figure 5.14: *Case study: validation of model m_2. (Top) One-step-ahead prediction for the validation data (solid: system response, dashed: model prediction, dotted: input). (Bottom) Active mode at each time instant.*

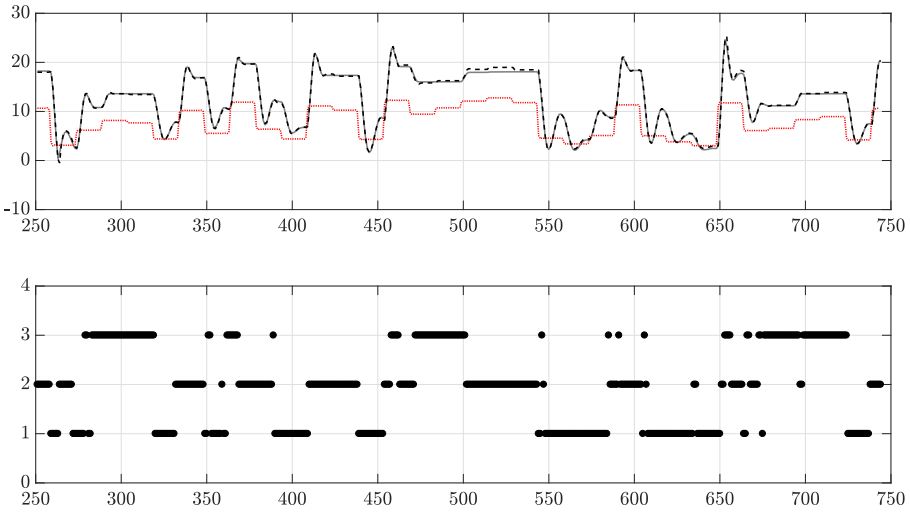

Figure 5.15: *Case study: validation of model m_3. (Top) One-step-ahead prediction for the validation data (solid: system response, dashed: model prediction, dotted: input). (Bottom) Active mode at each time instant.*

Table 5.12: *Case study: identified models with fixed NARX structures:* $m_1 : \{1,\ y(t-1),\ y(t-2),\ u(t-1)\}$. *Comparison with [57].*

Model	Parameters	Identified model	Identified model [57]
m_1	$N_M = 2,\ n_d = 1,\ n_y = 2,\ n_u = 1$	$\boldsymbol{\vartheta}^{(1)}$: $[0.606,\ 1.356,\ -0.633,\ -39.133]$ $\boldsymbol{\vartheta}^{(2)}$: $[-0.226,\ 1.498,\ -0.727,\ -41.456]$	$\boldsymbol{\vartheta}^{(1)}$: $[0.303,\ 1.4,\ -0.63,\ -34.412]$ $\boldsymbol{\vartheta}^{(2)}$: $[-0.718,\ 1.587,\ -0.768,\ -44.864]$

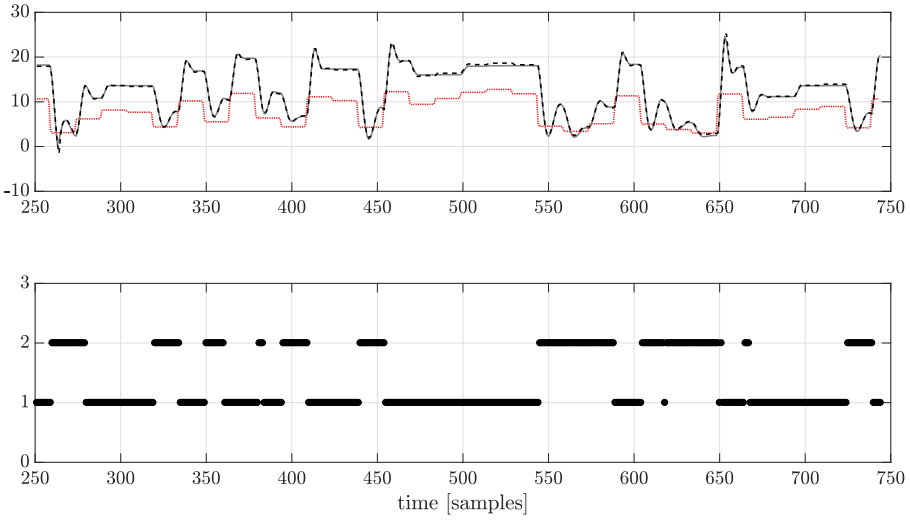

Figure 5.16: *Case study: validation of model m_1 identified by fixing the two NARX model structures. (Top) One-step-ahead prediction for the validation data (solid: system response, dashed: model prediction, dotted: input). (Bottom) Active mode at each time instant.*

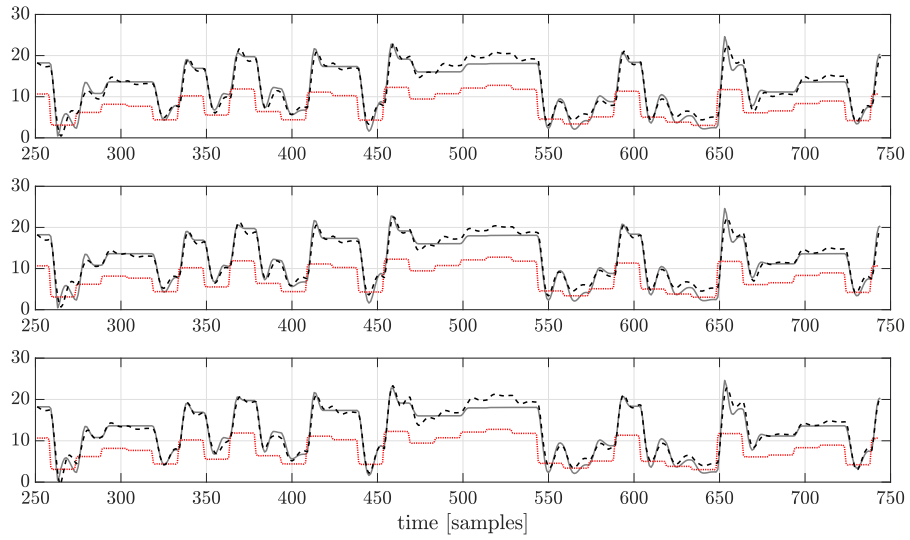

Figure 5.17: *Case study: open-loop output simulation of all PWARX models. (Top) simulated output for model m_1. (Center) simulated output for model m_2. (Bottom) simulated output for model m_3. Solid line: system response, dashed: simulated output, dotted: input.*

5.5 Conclusions

We considered the identification of switched nonlinear autoregressive exogenous (SNARX) models, and proposed an iterative method that addresses the challenge of the simultaneous identification of the mode switching sequence and of the NARX model associated to each mode. The proposed method alleviates the combinatorial complexity related to the identification of the switching signal by adopting a two-stage ap-

proach. More precisely, in the first stage, candidate mode switching instants are fixed and adopted to segment the input/output data and jointly solve the mode assignment and the NARX structure and parameter identification tasks; in the second stage, the candidate mode switching instants are refined. The combinatorial optimization problem in the first stage is addressed using a computationally attractive randomized method where the mode assignment and the SNARX model structure are modeled through discrete probability distributions that are progressively tuned via a sample-and-evaluate strategy, until convergence to a limit distribution concentrated on the best SNARX model of the system generating the observed data. Numerical examples show the effectiveness of the proposed method. Moreover, the proposed identification procedure has been validated also on the problem of data-driven model learning of a component placement process in a pick-and-place machine. This problem is better handled in the PWARX modeling framework, and hence we added a further step that addresses the partition of the regressor space into regions based on the data classification induced by the reconstructed switching signal, in order to validate the identified models on unseen data.

CHAPTER 6

Process noise covariance estimation in Kalman filtering

In modern engineering applications involving dynamical systems, model-based methods like Kalman filtering are usually employed to estimate the current state values from the corresponding outputs [46]. However, even in the linear time-invariant framework, the Kalman estimation of the state is proven to be optimal (*i.e.*, it minimizes the variance of the estimation error) only if the model *coincides* with the mathematical description of the system [60]. Then, in practical applications where no prior knowledge on the physics of the system is available and a state space model needs to be identified from data, modeling errors can jeopardize the filtering performance. In this view, many techniques have been proposed for the identification of the deterministic components of the state-space model. Conversely, less attention has been devoted to the estimation of the noise model which is expressed in terms of the noise Covariance Matrices (CMs) $\boldsymbol{Q}$ and $\boldsymbol{R}$, acting respectively on the state and output equations. This is quite surprisingly, since, while the variance of the output noise can be roughly estimated by looking at the output spectrum, the process noise statistics are generally unknown, given that the process noise model accounts for the unmodeled dynamics on which typically no prior knowledge is available.

This chapter treats the estimation of the process noise covariance matrix for linear Kalman filter design with an unusual emphasis on the problem of structure selection. Indeed, the approaches proposed for the identification of the noise statistics typically assume that the CMs are correctly parameterized and hence retrievable from data. However, as observed in [83], when one addresses the joint estimation of the state and the CMs from data, an identifiability issue may arise that prevents the estimation of all the elements of $\boldsymbol{Q}$. In such conditions, the state estimation is usually carried out

by directly estimating the Kalman gain without knowing $\boldsymbol{Q}$ and $\boldsymbol{R}$, or by assuming simplified structures for the CMs. In the latter case, the CMs are often assumed ot be diagonal. While this often seems satisfactory and coherent with physical evidences on the system there are cases in which a diagonal structure is ill suited. Consider, *e.g.,* distributed systems with various interconnected devices, whereby only state variables associated to a device or to neighboring devices are correlated. Here, the resulting $\boldsymbol{Q}$ is sparse, but typically not diagonal. To adequately address this issue, we propose an algorithm to suitably select the structure of the $\boldsymbol{Q}$ matrix to be employed in the filter design procedure. We want to stress here that this approach goes beyond the state estimation problem, trying to answer to the more general problem of quantifying the uncertainty affecting the identified process model.

The rest of the chapter is organized as follows. Section 6.1 introduces the problem statement which Section 6.2 refers to in order to briefly reviews the state of the art concerning the estimation of $\boldsymbol{Q}$. Section 6.3 highlights the mentioned identifiability issue by means of some simple but significant numerical examples. The case of diagonal parametrization of the $\boldsymbol{Q}$ is discussed in Section 6.4, while the algorithm for structure selection is provided in Section 6.5. Finally, some simulation examples are presented in Section 6.6, followed by some concluding remarks.

6.1 Problem statement

We consider linear time-invariant discrete-time dynamic stochastic systems with additive white Gaussian noise (shortly WGN) in the following state-space representation:

$$\begin{aligned} \boldsymbol{x}(k+1) &= \boldsymbol{F}\boldsymbol{x}(k) + \boldsymbol{v}(k) \\ \boldsymbol{y}(k) &= \boldsymbol{H}\boldsymbol{x}(k) + \boldsymbol{w}(k) \end{aligned} \tag{6.1}$$

where $\boldsymbol{x}(k) \in \mathbb{R}^n$ is the state vector, $\boldsymbol{y}(k) \in \mathbb{R}^p$ is the output vector, $\boldsymbol{v} \sim WGN(0, \boldsymbol{Q})$ is the process noise with $\boldsymbol{Q} = \boldsymbol{Q}^T \in \mathbb{R}^{n\times n}$ and $\boldsymbol{Q} \succeq 0$ and $\boldsymbol{w} \sim WGN(0, \boldsymbol{R})$ indicates the measurement noise with $\boldsymbol{R} = \boldsymbol{R}^T \succ 0$ and $\boldsymbol{R} \in \mathbb{R}^{p\times p}$. $\boldsymbol{F}$ and $\boldsymbol{H}$ are the dynamic and output matrices, respectively. We assume that $\boldsymbol{v}$ and $\boldsymbol{w}$ are uncorrelated. To ease the discussion, no deterministic exogenous inputs are here considered, without loss of generality. Let $\hat{\boldsymbol{x}}(k|k)$ be the optimal filter, that is the estimate of $\boldsymbol{x}(k)$ given the outputs $\boldsymbol{y}$ up to the current discrete time instant k. This estimation is carried out in two phases: the predictive phase and the update phase. In the *predictive phase*, the measurements up to the previous time instant $(k-1)$ are used to predict the value of the states (and the outputs) at the current instant k, as well as $\boldsymbol{P}(k)$, which is the covariance matrix of the *state estimation error*:

$$\hat{\boldsymbol{x}}(k|k-1) = \boldsymbol{F}\hat{\boldsymbol{x}}(k-1|k-1) \tag{6.2}$$

$$\hat{\boldsymbol{y}}(k|k-1) = \boldsymbol{H}\hat{\boldsymbol{x}}(k|k-1) \tag{6.3}$$

$$\boldsymbol{P}(k|k-1) = \boldsymbol{F}\boldsymbol{P}(k-1|k-1)\boldsymbol{F}^T + \boldsymbol{Q} \tag{6.4}$$

The *update phase* starts as soon as the measurement relative to the current time instant (that is being estimated) is available. Both the estimates of $\boldsymbol{x}(k)$ and $\boldsymbol{P}(k)$ are adjusted

taking into account this new information:

$$\hat{\boldsymbol{x}}(k|k) = \hat{\boldsymbol{x}}(k|k-1) + \boldsymbol{K}(k)\left[\boldsymbol{y}(k) - \boldsymbol{H}\hat{\boldsymbol{x}}(k|k-1)\right] \tag{6.5}$$

$$\boldsymbol{P}(k|k) = \left[\boldsymbol{I} - \boldsymbol{K}(k)\boldsymbol{H}\right]\boldsymbol{P}(k|k-1) \tag{6.6}$$

where the Kalman gain $\boldsymbol{K}(k)$ is defined as:

$$\boldsymbol{K}(k) = \boldsymbol{P}(k|k-1)\boldsymbol{H}^T\left[\boldsymbol{H}\boldsymbol{P}(k|k-1)\boldsymbol{H}^T + \boldsymbol{R}\right]^{-1} \tag{6.7}$$

Notice that the *a priori* estimate of $\boldsymbol{x}(k)$ calculated in the predictive phase is adjusted by a correction term which equals the estimation error $\boldsymbol{e}(k) = \boldsymbol{y}(k) - \boldsymbol{H}\hat{\boldsymbol{x}}(k|k-1)$, also called *innovation*, weighted by $\boldsymbol{K}(k)$. When both phases are concluded, index k is incremented and the procedure iterated. An initialization for $\hat{\boldsymbol{x}}(0|-1)$ and $\boldsymbol{P}(0|-1)$ is required.

Property 2. *Let $\boldsymbol{\varepsilon}(k) = \boldsymbol{x}(k) - \hat{\boldsymbol{x}}(k|k)$ and consider the quadratic loss function:*

$$\mathcal{L}(\boldsymbol{\varepsilon}(k)) = \boldsymbol{\varepsilon}(k)^T\boldsymbol{\varepsilon}(k). \tag{6.8}$$

The filter $\hat{\boldsymbol{x}}(k|k)$ defined in Eq. (6.5) *is the minimum error variance filter,* i.e., *it minimizes the average loss or risk:*

$$\mathcal{R}(\hat{\boldsymbol{x}}(k|k)) \equiv \mathbb{E}\left[\mathcal{L}(\boldsymbol{\varepsilon}(k))\right], \tag{6.9}$$

provided that the two CMs are known.

Proof. By Theorem 1 in [59], the minimum error variance filter $\hat{\boldsymbol{x}}(k|k)$ is the conditional expectation

$$\hat{\boldsymbol{x}}(k|k) = \mathbb{E}\left[\boldsymbol{x}(k)|\boldsymbol{y}(0), \ldots, \boldsymbol{y}(k)\right],$$

which, under the constraint that the filter be linear, takes the form of Eq. (6.5) (see Theorem 2 in [59]). □

Furthermore, it can be easily proved (see Section 3.1 in [3]) that the minimum error variance filter has the following property:

$$\mathcal{R}(\hat{\boldsymbol{x}}(k|k)) = \text{trace}\{\boldsymbol{P}(k|k)\}, \tag{6.10}$$

which motivates the choice of the trace of the state estimation covariance matrix as an optimality criterion in the sequel.

In the following, we denote as $\mathcal{R}(\hat{\boldsymbol{x}}(k|k), \boldsymbol{Q})$ the risk value associated with the filter $\hat{\boldsymbol{x}}(k|k)$ computed employing the given matrix $\boldsymbol{Q}$ in the design procedure.

6.2 State of the art

The available approaches proposed for the joint estimation of the state and the CMs, can be classified into two main categories. Feedback methods estimate simultaneously the unknown CMs and the state by extending the system with further states which represent the parameters of the unknown CMs. In feedback-free methods a sub-optimal filter is employed to estimate the states, and then the prediction error of the latter is used to calculate the CMs. The main drawback of feedback methods is the fact that

the extended system is nonlinear. Feedback-free methods are then often preferred, in view of their usually lower computational burden. Accordingly, in this section, we will briefly review only the main feedback-free methods. For further details on feedback methods, see [33].

The class of methods mainly employed are the so-called *correlation-based methods*. Such techniques are based on the analysis of the *innovation* sequence $\{\boldsymbol{e}(k)\}_{k=1}^{N}$, generated by a steady-state Kalman filter starting from an arbitrary initial condition $\hat{\boldsymbol{x}}(0|-1)$ and a gain $\boldsymbol{K}$ selected such that the matrix $\bar{\boldsymbol{F}} = \boldsymbol{F} - \boldsymbol{FKH}$ is stable. The pioneer of these methods is the work presented in [83] (denoted here as Indirect Correlation Method, ICM), in which the noise CMs are estimated using a three-step procedure with a classical least squares (LS) approach. The method is based on a system of N_E (user-defined) linear matrix equations stemming from the auto-covariance function (ACF):

$$C_j = \text{cov}[\boldsymbol{e}(k), \boldsymbol{e}(k-j)] = \begin{cases} \boldsymbol{HPH}^T + \boldsymbol{R}, & j = 0 \quad \text{(6.11a)} \\ \boldsymbol{H}\bar{\boldsymbol{F}}^{j-1}\boldsymbol{F}(\boldsymbol{PH}^T - \boldsymbol{K}C_0), & j > 0 \quad \text{(6.11b)} \end{cases}$$

where steady-state $\boldsymbol{P}$ is given by the solution to the Lyapunov equation

$$\boldsymbol{P} = \bar{\boldsymbol{F}}\boldsymbol{P}\bar{\boldsymbol{F}}^T + \boldsymbol{FKRK}^T\boldsymbol{F}^T + \boldsymbol{Q}. \tag{6.12}$$

Specifically, given the estimate of the ACF computed from the innovation sequence as

$$\hat{C}_j = \frac{1}{N-j}\sum_{k=j}^{N} \boldsymbol{e}(k)^T\boldsymbol{e}(k-j),\ j = 0, \ldots, N_E - 1, \tag{6.13}$$

where N denotes the number of collected data, the following three steps are performed:

1. Compute the LS estimate $\hat{\boldsymbol{P}}\hat{\boldsymbol{H}}^T$ of $\boldsymbol{PH}^T$ from $\hat{C}_j$, $j = 1, \ldots, N_E - 1$, with $C_0 = \hat{C}_0$, according to (6.11b).
2. Based on $\hat{\boldsymbol{P}}\hat{\boldsymbol{H}}^T$ and $C_0 = \hat{C}_0$, compute the estimate $\hat{\boldsymbol{R}}$ of $\boldsymbol{R}$, from (6.11a).
3. Compute the LS estimate $\hat{\boldsymbol{Q}}$ of $\boldsymbol{Q}$ from (6.12), substituting $\hat{\boldsymbol{P}}\hat{\boldsymbol{H}}^T$ for $\boldsymbol{PH}^T$ and multiplying both sides of (6.12) by $\boldsymbol{H}$ and $\boldsymbol{H}^T$, respectively.

Remark 2. *Notice that the estimation of $\boldsymbol{R}$ is decoupled from that of $\boldsymbol{Q}$, since the sampled version of C_j, namely $\hat{C}_j$ in* (6.13)*, is employed to compute the system of N_E equations according to* (6.11).

A second method, denoted here Weighted Correlation Method (WCM), is introduced in [8]. This approach is still based on the processing of the ACF, similarly to [83], but employs a suitable parameterization of the noise CMs and the ACF. More precisely, the unknown CMs are expressed as linear combinations of known (user-defined) basis matrices $\boldsymbol{Q}^{(i)}$ and $\boldsymbol{R}^{(i)}$, $i = 1, \ldots, M$:

$$\boldsymbol{Q} = \sum_{i=1}^{M} \theta(i)\boldsymbol{Q}^{(i)},\ \boldsymbol{R} = \sum_{i=1}^{M} \theta(i)\boldsymbol{R}^{(i)},$$

where $\theta^{(i)}$, $i = 1, \ldots, M$, are unknown weights to be estimated. In a similar fashion, the ACF is now defined as a linear function of the unknown weights $\theta(i)$:

$$C_j = \text{cov}[\boldsymbol{e}(k), \boldsymbol{e}(k-j)] = \sum_{i=1}^{M} \mathcal{F}_i(t,j)\theta(i),$$

where $\mathcal{F}_i(t,j)$ is appropriately defined (see Equation (16) in [8]).

Remark 3. *Assuming for example $n = p = 2$, the basis matrices $\boldsymbol{Q}^{(i)}$ and $\boldsymbol{R}^{(i)}$, $i = 1, \ldots, M$ can be selected as follows:*

$$\boldsymbol{Q}^{(1)} = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}, \boldsymbol{Q}^{(2)} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \boldsymbol{Q}^{(3)} = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix},$$
$$\boldsymbol{R}^{(4)} = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}, \boldsymbol{R}^{(5)} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \boldsymbol{R}^{(6)} = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix},$$

$$\boldsymbol{Q}^{(4)} = \boldsymbol{Q}^{(5)} = \boldsymbol{Q}^{(6)} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$$
$$\boldsymbol{R}^{(1)} = \boldsymbol{R}^{(2)} = \boldsymbol{R}^{(3)} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix},$$

This justifies the use of a single vector of weights $\theta \in \mathbb{R}^M$ for both the $\boldsymbol{Q}^{(i)}$s and the $\boldsymbol{R}^{(i)}$s.

Finally, the method presented in [88] (denoted here as Direct Correlation Method, DCM), and extended in [100], estimates the CMs in a single step, by reformulating the relations of [83] in such a form that the three intermediate steps can be replaced by the resolution of a single LS problem.

The three mentioned methods appear to be very computational efficient, since they only need a single-point estimation following a classical LS approach. On the other hand, these methods have some limitations in the estimation of $\boldsymbol{Q}$ (they can estimate up to $n \cdot p$ elements [1]), and they require the full knowledge of its structure (*i.e.*, the number and position of the nonzero elements). Furthermore, they are based on the assumption that the sampled ACF $\hat{C}_j$ approaches its true value. However, this is not generally true for finite data sets, especially due to the dependence of the innovation sequence on the *a priori* defined gain $\boldsymbol{K}$.

The *maximum-likelihood (ML) methods* are based on the maximization of a likelihood function over the collected data. The method in [61] (denoted Input-Output Correlation Method, IOCM) is based on the minimization of the innovation related to an

[1] With reference to the ICM method, only the estimate of PH^T instead of P can be obtained by the ACF. Thus, only $n \cdot p$ linear matrix equations in the unknown components of $\boldsymbol{Q}$ are available.

input-output ARMAX model of order $\mathcal{O}$. Assuming that $\mathcal{O} = 1$, $n = p$, and $\boldsymbol{H} = \boldsymbol{I}_n$, the ARMAX model can be reformulated as in (6.1). A ML step is first required to compute the covariance and the cross-covariances of the innovation, that are needed to estimate the noise CMs.

A classical ML approach is instead adopted in [106] (here denoted MLM), where a negative log-likelihood function is directly maximized w.r.t. the unknown CMs:

$$L(\boldsymbol{Q}, \boldsymbol{R}) = \texttt{ln}(|\boldsymbol{Q}|) + \texttt{ln}(|\boldsymbol{R}|) + \sum_{k=1}^{N} \left(\boldsymbol{x}(k) - \boldsymbol{F}\boldsymbol{x}(k-1)\right)^T \boldsymbol{Q}^{-1} \left(\boldsymbol{x}(k) - \boldsymbol{F}\boldsymbol{x}(k-1)\right) + \sum_{k=0}^{N} \left(\boldsymbol{y}(k) - \boldsymbol{H}\boldsymbol{x}(k)\right)^T \boldsymbol{R}^{-1} \left(\boldsymbol{y}(k) - \boldsymbol{H}\boldsymbol{x}(k)\right).$$

Therefore, compared to the previous method, no intermediate steps are needed to recover the optimal CMs. However, the *a priori* setting of the unknown CMs is crucial for the convergence of the method and the accuracy of the final estimate.

In general, although they generally provide more accurate solutions than correlation-based methods and they do not have limitations on the number of free parameters that can be estimated, ML methods involve a much more significant computational load, mainly due to the computation of the gradient descent direction from data over the parameter space.

6.3 Estimation of Q

The general idea behind all methods devoted to the *joint* estimation of the state and the CMs, is to exploit the existing relation between the CMs, the state vector $\boldsymbol{x}(k)$ and $\boldsymbol{P}(k|k)$. From Property 2 and relation (6.10), a possible way to solve the $\boldsymbol{Q}$ estimation problem is to address it as an optimization problem:

$$\underset{\boldsymbol{Q}}{\text{minimize}} \quad \mathcal{R}(\hat{\boldsymbol{x}}(k|k), \boldsymbol{Q}), \tag{6.14}$$

with $\mathcal{R}(\hat{\boldsymbol{x}}(k|k), \boldsymbol{Q})$ defined as in (6.9). Indeed, assuming that the observed data $\mathcal{D} = \{(\boldsymbol{x}(k), \boldsymbol{y}(k)\}_{k=1}^{N}$ [2] have been generated using $\boldsymbol{v} \sim WGN(0, \boldsymbol{Q}^\circ)$, it follows that

$$\mathcal{R}(\hat{\boldsymbol{x}}(k|k), \boldsymbol{Q}^\circ) \leq \mathcal{R}(\hat{\boldsymbol{x}}(k|k), \boldsymbol{Q}), \forall\, Q \neq \boldsymbol{Q}^\circ.$$

However, it is not always possible to retrieve $\boldsymbol{Q}^\circ$ from $\mathcal{D}$ due to structural issues. Indeed, as discussed in [83], for a system of order n with p outputs, Property 3 holds.

Property 3. *Given a system in the form* (6.1) *with* n *states and* p *outputs, if*

$$p < \lceil \frac{(n+1)}{2} \rceil, \tag{6.15}$$

then there exist infinite optimal solutions for problem (6.14).

Proof. The lower bound in (6.15) follows by imposing that the DOFs associated to $\boldsymbol{Q}$ be at most equal the number of $\boldsymbol{K}(k)$ components, *i.e.,* $\frac{n(n+1)}{2} \leq np$. □

[2] We assume that the available dataset includes also the state measurement. However, this is not a strong assumption in many practical applications, see, *e.g.*, the Kalman-filter based roll angle estimation of [23] and [42].

Property 3 implies that the q_{ij}, $i, j = 1, \ldots, n$ are not exactly retrievable from data when the estimation of the state *and* the CMs is jointly addressed, if over-parameterization occurs. We now exemplify this identifiability issue by means of some simple numerical examples.

Consider *e.g.* the following scalar system:

$$S_1 : \begin{cases} x(k+1) = 0.5x(k) + v(k) \\ y(k) = 1.5x(k) + w(k) \end{cases} \tag{6.16}$$

with $Q^\circ = 0.1$, $R = \text{var}[y(k)]/10$, $x(0) = 0$.

A Monte Carlo (MC) study has been carried out analyzing 1000 different noise realizations of length $N = 10000$. On each run, the optimization problem (6.14) has been solved on an unbounded domain using the `fminsearch` Matlab routine [82]. Figure 6.1 shows the distribution of the obtained optimal Q values w.r.t. the noise realization. In general, there is an error in the estimation of Q° induced by the specific noise realization. However, the resulting distribution is centered on the correct value, indicated by the black triangular marker.

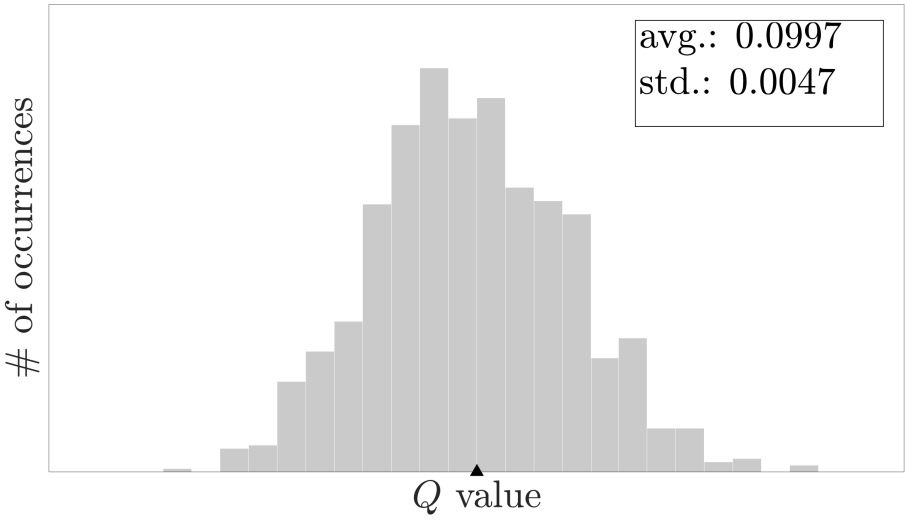

Figure 6.1: *S_1: Distribution of the optimal Q values over* 1000 *MC runs with different noise realizations. The black triangular marker indicates the true value,* i.e., $Q^\circ = 0.1$.

Consider now *e.g.* the following second order system:

$$S_2 : \begin{cases} \boldsymbol{x}(k+1) = \begin{bmatrix} 0.9 & -0.4 \\ 0.2 & 0.9 \end{bmatrix} \boldsymbol{x}(k) + \boldsymbol{v}(k) \\ y(k) = \begin{bmatrix} 0.5 & 0 \end{bmatrix} \boldsymbol{x}(k) + \boldsymbol{w}(k) \end{cases} \tag{6.17}$$

with $\boldsymbol{Q}^\circ = \begin{bmatrix} 0.0125 & 0.15 \\ 0.15 & 10 \end{bmatrix}$, $R = \frac{\text{var}[y(k)]}{10}$, $\boldsymbol{x}(0) = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$.

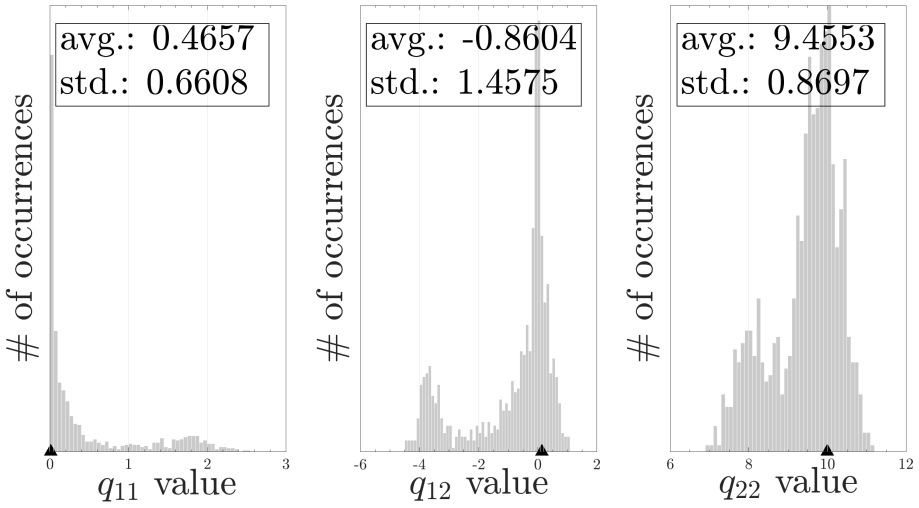

Figure 6.2: *S_2: Distribution of the optimal values for q_{11}, q_{12}, and q_{22} over 1000 MC runs with different noise realizations. The black triangular markers indicate the true values.*

An MC study has been carried out as before. Figure 6.2 shows the distribution of the optimal values of the coefficients of $\boldsymbol{Q}$ w.r.t. the noise realization. As previously, the identification accuracy depends on the noise realization, but in this case, the resulting distributions are *not* centered on the "true" values. This experimental evidence demonstrates that it is not always possible to deduce directly from $\mathcal{D}$ the $\boldsymbol{Q}^\circ$, not even asymptotically, when the estimation of the state *and* the CMs is jointly addressed, if over-parameterization occurs. Indeed, for the considered system S_2, the Kalman gain $\boldsymbol{K}^\circ(k)$ is a 2×1 vector, whereas the $\boldsymbol{Q}$ matrix has 3 degrees of freedom (DOFs). Therefore, the same $\boldsymbol{K}^\circ(k)$ is obtained for infinite values of $\boldsymbol{Q}$ (which are all optimal).

Note that, despite the result stated in Property 3, a full analysis of the conditions under which the noise CMs elements can be estimated is still missing in the literature. This motivates the interest in estimating directly the optimal gain of a Kalman filter independently of the knowledge of $\boldsymbol{Q}$, as proposed in [83], or in the usage of $\boldsymbol{Q}$s with simplified structures. Indeed, notice that condition (6.15) is automatically satisfied if *e.g.,* a diagonal structure is chosen for $\boldsymbol{Q}$, since in that case $\boldsymbol{Q}$ has only n DOFs which is at most equal to the number np of elements of the Kalman gain. Figure 6.3 exemplifies this concept, where the experiment on S_2 is repeated with a diagonal $\boldsymbol{Q}$, *i.e.* with only 2 DOFs.

We next focus on the problem of choosing a suitable structure for $\boldsymbol{Q}$ when it is unknown. This motivates the selection of the numerical optimization in (6.14), w.r.t correlation methods which provide an analytical solution to the $\boldsymbol{Q}$ estimation problem but that do not handle structural constraints.

6.4 Which structure for Q?

A common practice adopted by several methods is to simplify the structure of matrix $\boldsymbol{Q}$ to be diagonal, with obvious computational advantages. While this often works out sat-

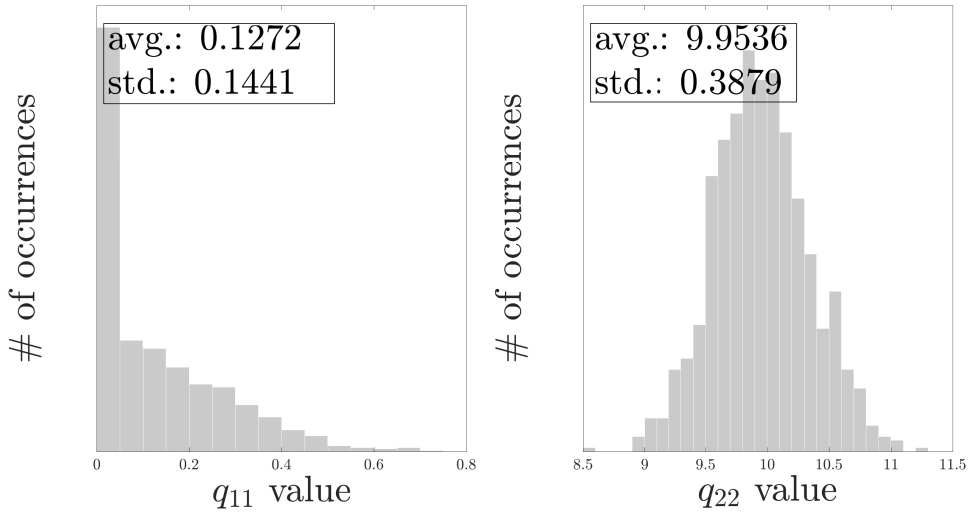

Figure 6.3: *S_2: Distribution of the optimal values for q_{11} and q_{22} over 1000 MC runs with different noise realizations. In this case, over-parameterization does not occur since a diagonal structure is adopted for $\boldsymbol{Q}$.*

isfactorily because it reduces the degrees of freedom of $\boldsymbol{Q}$ to n, it is not always the most appropriate choice, and may lead to an unnecessary loss of filtering accuracy. To investigate this issue in more detail, we here address the following constrained optimization problem:

$$\begin{aligned} \underset{\boldsymbol{Q}}{\text{minimize}} \quad & \mathcal{R}(\hat{\boldsymbol{x}}(k|k), \boldsymbol{Q}) \\ \text{subject to} \quad & \boldsymbol{Q} \in \mathcal{Q}_\kappa, \end{aligned}$$

where $\mathcal{Q}_\kappa$ is the set of all positive semidefinite matrices $\boldsymbol{Q} \in \mathbb{R}^{n \times n}$ with $\kappa \leq \frac{n(n+1)}{2}$ free parameters. To ensure that $\boldsymbol{Q} = \boldsymbol{Q}^T \succeq 0$, $\boldsymbol{Q}$ is parameterized using a Cholesky decomposition:

$$\boldsymbol{Q} = \boldsymbol{Z}\boldsymbol{Z}^T, \tag{6.18}$$

where $\boldsymbol{Z}$ is a lower triangular matrix, and the optimization process is carried out over the parameters z_{ij}. As a result, the structural constraints on $\boldsymbol{Q}$ translate to conditions on the parameters z_{ij} of the Cholesky factors. For example, for $n = 3$ one has that:

$$\boldsymbol{Q} = \boldsymbol{Z}\boldsymbol{Z}^T = \begin{bmatrix} z_{11}^2 & z_{11}z_{21} & z_{11}z_{31} \\ z_{21}z_{11} & z_{21}^2 + z_{22}^2 & z_{21}z_{31} + z_{22}z_{32} \\ z_{31}z_{11} & z_{31}z_{21} + z_{32}z_{22} & z_{31}^2 + z_{32}^2 + z_{33}^2 \end{bmatrix}. \tag{6.19}$$

Therefore, the constraint $z_{21}z_{31} + z_{22}z_{32} = 0$ must be applied in the optimization problem to ensure that $q_{23} = q_{32} = 0$, and so on. More simply, a diagonal $\boldsymbol{Q}$ is obtained by setting Z to be diagonal ($z_{21} = z_{31} = z_{32} = 0$).

Let us consider the third order system described below:

$$S_3 : \begin{cases} \boldsymbol{x}(k+1) = \begin{bmatrix} 0.0218 & 0.9243 & -0.2750 \\ 0.4645 & -0.2466 & -0.8076 \\ 0.8451 & 0.1167 & 0.4530 \end{bmatrix} \boldsymbol{x}(k) + \boldsymbol{v}(k) \\ \boldsymbol{y}(k) = \begin{bmatrix} 0.9936 & 0 & 0.6539 \end{bmatrix} \boldsymbol{x}(t) + \boldsymbol{w}(k) \end{cases} \tag{6.20}$$

with

$$\boldsymbol{Q}^{\circ} = \begin{bmatrix} 6.3557 & 6.2921 & -0.7910 \\ 6.2921 & 6.5128 & -0.0420 \\ -0.7910 & -0.0420 & 6.7963 \end{bmatrix},$$

$R = \text{var}[y(k)]/10$, $\boldsymbol{x}(0) = [0\ 0\ 0]^T$. As done previously, an observation window of $N = 10000$ samples has been considered. By a simple visual inspection of $\boldsymbol{Q}$, it should be clear that a diagonal structure will miss the contribution of q_{12} which is comparable with that of q_{ii}, $i = 1, 2, 3$.

An exhaustive analysis over all possible matrix structures has been carried out and the aggregated results are reported in Figure 6.4. In particular, for each structure, 100 MC runs have been carried out on the same data realization for different random initializations of the optimization solver, and the identified optimal $\boldsymbol{Q}$s have been validated over a test set. In this plot, the black marker refers to the best diagonal solution, while the red marker refers to $\mathcal{R}(\hat{\boldsymbol{x}}(k|k), \boldsymbol{Q}^{\circ})$. Observing the parameterizations with 3 DOFs, one can note that the full diagonal case does not represent the best solution and more efficient *non-diagonal* structures can be pursued with the same number of parameters. Indeed, the minimum $\mathcal{R}(\hat{\boldsymbol{x}}(k|k), \boldsymbol{Q})$ associated to the diagonal case (d), computed on a test set, is 42.6770 w.r.t. 40.3954 (i.e., a relative error of 5.6482%), which corresponds to the best *non-diagonal* (nd) structure with 3 DOFs. The two corresponding identified matrices are listed below:

$$\boldsymbol{Q}_d = \begin{bmatrix} 2.3232 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 11.7474 \end{bmatrix}, \ \boldsymbol{Q}_{nd} = \begin{bmatrix} 6.2337 & 8.2759 & 0 \\ 8.2759 & 10.9879 & 0 \\ 0 & 0 & 0 \end{bmatrix}.$$

It is worth noting that both these matrices make the pair $\left(\boldsymbol{F}, \sqrt{\boldsymbol{Q}}\right)$ uncontrollable, which results in the Kalman filter neglecting the effect of new measurements on the state corresponding to the row of zeros. Unusual as it may seem, this is perfectly legitimate and does not limit *per se* the applicability of the Kalman filter (see Section 3.1 in [109]).

As for S_2, an identifiability issue arises also for S_3 due to over-parameterization, when the optimization is carried out on a $\boldsymbol{Q}$ matrix with more than 3 DOFs. This prevents one from obtaining the $\boldsymbol{Q}^{\circ}$ in the unconstrained case (*i.e.* with 6 DOFs).

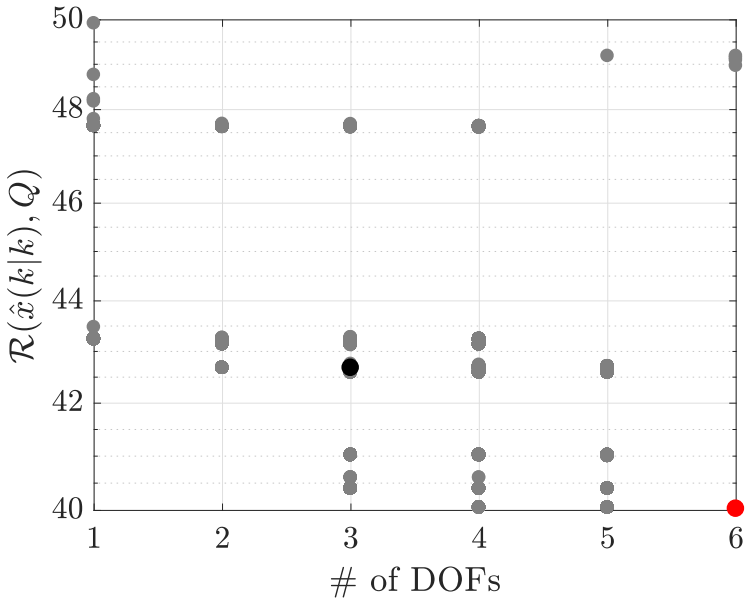

Figure 6.4: *S_3: estimation error corresponding to different parameterizations. The black marker represents the diagonal case, while the red one the case $\mathcal{R}(\hat{\boldsymbol{x}}(k|k), \boldsymbol{Q}^{\circ})$. The reported values have been computed on a test data-set.*

6.5 Matrix structure selection method

The analysis carried out in the previous sections shows that while the $\boldsymbol{Q}$ matrix should be endowed with a limited number of degrees of freedom, the diagonal structure does not necessarily always provide the best possible approximation of the real $\boldsymbol{Q}$. Building on this observation, we next propose an approach to address the matrix structure selection of $\boldsymbol{Q}$, which ultimately results in a MSS problem. Accordingly, we tackle this problem within the probabilistic framework described in Chapter 3.

Let $\boldsymbol{\vartheta} = [z_{11}\ z_{21}\ z_{22}\ ...]^T$ be a vector containing all the $N_T = \frac{n(n+1)}{2}$ parameters z_{ij}, $i \geq j$, in lexicographical order. Let also $\boldsymbol{s}$ be a binary vector taking values in the set $\Sigma = \{0, 1\}^{N_T}$, where the k-th element s_k encodes the presence (or absence) of ϑ_k in the model structure. For example, a third order $\boldsymbol{Q}$ matrix with the following structure:

$$Q = \begin{bmatrix} q_{11} & 0 & 0 \\ 0 & q_{22} & q_{23} \\ 0 & q_{23} & q_{33} \end{bmatrix}$$

can be obtained by selecting $\vartheta_1 = z_{11}$, $\vartheta_3 = z_{22}$, $\vartheta_5 = z_{32}$ (see (6.19)), *i.e.* by employing a Z factor with the structure $\boldsymbol{s} = [1\ 0\ 1\ 0\ 1\ 0]$. Notice that the same structure for $\boldsymbol{Q}$ could have also been achieved using a larger structure $\boldsymbol{s} = [1\ 0\ 1\ 0\ 1\ 1]$, encompassing also the term z_{33}, but the inclusion of this additional parameter does not lead to any improvements in terms of accuracy.

We aim to find the matrix $\boldsymbol{Q}$ with the *smallest number of non-zero parameters* that provides an "acceptable" performance. Accordingly, we can formulate the structure

selection problem as follows:

$$\underset{s}{\text{minimize}} \quad \mathcal{R}(\hat{\boldsymbol{x}}(k|k), \boldsymbol{Q}(\boldsymbol{s})) + \lambda \mathcal{P}(\boldsymbol{s}), \tag{6.21}$$

where $\mathcal{R}(\hat{\boldsymbol{x}}(k|k), \boldsymbol{Q}(\boldsymbol{s}))$ is defined in (6.9), $\boldsymbol{Q}(\boldsymbol{s})$ being a symmetric semidefinite positive matrix with structure induced by $\boldsymbol{s}$, $\mathcal{P}(\boldsymbol{s}) : \Sigma \to \mathbb{R}^+$ is a penalization term, and parameter λ determines the relative importance of the two sub-objectives. Since $\boldsymbol{s}$ is a binary vector, the most intuitive choice for the penalization term is the zero-norm $\|\boldsymbol{s}\|_0 = \text{card}\{s_i : s_i \neq 0\}$, which actually coincides in this specific case with the ℓ_1 norm $\|\boldsymbol{s}\|_1 = \sum_{i=1}^{N_T} |s_i|$. In view of the convexity of the ℓ_1 norm, this is preferred to the ℓ_0 norm [25].

In order to be compliant with the problem formulation in (3.1), we will reformulate problem (6.21) as follows

$$\underset{\boldsymbol{s}}{\text{maximize}} \quad \mathcal{J}(\boldsymbol{s}) = e^{-\beta \cdot [\mathcal{R}(\hat{\boldsymbol{x}}(k|k), \boldsymbol{Q}(\boldsymbol{s})) + \lambda \mathcal{P}(\boldsymbol{s})]}, \tag{6.22}$$

where $\beta > 0$ is a design parameter. The exponential form of the cost function provides a natural normalization in the interval $(0, 1]$, and allows a sharper discrimination between structures with similar performance [110]. In the following, we will denote as $\boldsymbol{s}^\star$ the optimal solution of problem (6.22). We will further assume that $\mathcal{J}(\boldsymbol{s}) < \mathcal{J}(\boldsymbol{s}^\star), \forall \boldsymbol{s} \in \Sigma \setminus \{\boldsymbol{s}^\star\}$. An exhaustive solution of the optimization problem (6.22) for all possible values of $\boldsymbol{s}$ is not convenient for high order systems, due to the complexity of the combinatorial part of the problem. Indeed, there are 2^{N_T} different values for vector $\boldsymbol{s}$, and, in turn, N_T grows as the square of the system order n. For this purpose we adopt the probabilistic reformulation presented in Section 3.1, by assigning a probability to each possible value of the structure vector to be the target solution $\boldsymbol{s}^\star$. To this end, let $\boldsymbol{\gamma}$ be a discrete variable taking values in Σ according to the probability distribution $\mathbb{P}_{\boldsymbol{\gamma}}$:

$$\mathbb{P}_{\boldsymbol{\gamma}} = \prod_{i:s_i=1} \pi_i \prod_{i:s_i=0} (1 - \pi_i), \tag{6.23}$$

where π_i is the success probability of the Bernoulli random variable $\gamma_i \sim \texttt{Bernoullian}(\pi_i)$ associate to the term ϑ_i. Accordingly, ϑ_i is present in the extracted Cholesky factor $\boldsymbol{Z}$ (*i.e.* $s_i = 1$) if $\gamma_i = 1$ and is absent if $\gamma_i = 0$. Parameter μ_i represents the confidence level that ϑ_i belongs to the target structure $\boldsymbol{s}^\star$. Starting from an initial tentative distribution (*e.g.,* such that all possible structures have equal probability), we then employ a *sample-and-evaluate* approach to progressively update the probability distribution, enhancing the probability that high-performance structures are extracted.

More precisely, at each iteration, several sample structures are extracted from $\mathbb{P}_{\boldsymbol{\gamma}}$ (each structure is obtained by extracting one sample for each Bernoulli distribution) and evaluated. The latter operation amounts to solving problem 6.14 with the proper constraints on the structure of $\boldsymbol{Q}$, and yields the parameters of the matrix $\boldsymbol{Q}$ with the assigned structure that maximizes the filter performance on the collected data set of N state-measurement pairs $\mathcal{D}$. Then, the retrieved $\boldsymbol{Q}$ is used to evaluate the corresponding structure $\boldsymbol{s}$ through $\mathcal{J}(\boldsymbol{s})$. Finally, one updates π_i, for each term ϑ_i, according to the sign of the difference between the average performance of the structures that contain it, weighted with the respective probabilities as given by $\mathbb{P}_{\boldsymbol{\gamma}}$, and the corresponding average performance of the remaining models weighted in probability. Specifically, let

$$\delta_i = \mathbb{E}_{\mathbb{P}_{\boldsymbol{\gamma}}}[\mathcal{J}(\boldsymbol{\gamma}) | \gamma_i = 1] - \mathbb{E}_{\mathbb{P}_{\boldsymbol{\gamma}}}[\mathcal{J}(\boldsymbol{\gamma}) | \gamma_i = 0], \tag{6.24}$$

one updates π_i according to the sign of the sampled version of δ_i:

$$\pi_i(r+1) = \pi_i(r) + \chi\delta_i, \tag{6.25}$$

where χ is a step size and r denotes the current iteration. The overall algorithm is reported in Algorithm 6.

Remark 4. *The update of π_i at line 22 is based on the computation of the average performance of those structures containing the term ϑ_i, through the variables $\mathcal{J}^{\oplus}$ and $n^{\oplus}$ defined in Algorithm 6, and the average performance of the remaining models, through $\mathcal{J}^{\ominus}$ and $n^{\ominus}$.*

Algorithm 6 Matrix structure selection algorithm

Require: $\{(\boldsymbol{x}(k), \boldsymbol{y}(k)), k = 1, \cdots, N\}, N_T, N_p, \beta, \lambda, \gamma, \pi_{min}, \pi_{max}$
Ensure: $\boldsymbol{\pi}$

1: $\boldsymbol{\pi} \leftarrow \frac{1}{N_T} \cdot \mathbf{1}_{N_T \times 1}$;
2: **repeat**
3: **for** $p = 1$ **to** N_p **do** ▷ Generate structures
4: $\boldsymbol{s}^{(p)} = [\,]$;
5: **for** $i = 1$ **to** N_T **do**
6: Extract t_i from Bernoullian(π_i); ▷ Generate terms
7: $\boldsymbol{s}^{(p)} \leftarrow [\boldsymbol{s}^{(p)},\ t_i]$;
8: **end for**
9: Define $\boldsymbol{Q}^{(p)}$ according to structure $\boldsymbol{s}^{(p)}$;
10: $\{\boldsymbol{Q}^{(p)},\ \mathcal{R}(\hat{\boldsymbol{x}}(k|k), \boldsymbol{Q}^{(p)})\}$ CM estimation;
11: $\mathcal{J}^{(p)} \leftarrow e^{-\beta \cdot \left[\mathcal{R}(\hat{\boldsymbol{x}}(k|k), \boldsymbol{Q}^{(p)}) + \lambda \mathcal{P}(\boldsymbol{s}^{(p)})\right]}$; ▷ Structure evaluation
12: **end for**
13: **for** $i = 1$ **to** N_T **do** ▷ Update π_i
14: $\mathcal{J}^{\oplus} \leftarrow 0$; $n^{\oplus} \leftarrow 0$; $\mathcal{J}^{\ominus} \leftarrow 0$; $n^{\ominus} \leftarrow 0$;
15: **for** $p = 1$ **to** N_p **do**
16: **if** $s_i^{(p)} = 1$ **then**
17: $\mathcal{J}^{\oplus} \leftarrow \mathcal{J}^{\oplus} + \mathcal{J}^{(p)}$; $n^{\oplus} \leftarrow n^{\oplus} + 1$;
18: **else**
19: $\mathcal{J}^{\ominus} \leftarrow \mathcal{J}^{\ominus} + \mathcal{J}^{(p)}$; $n^{\ominus} \leftarrow n^{\ominus} + 1$;
20: **end if**
21: **end for**
22: $\pi_i \leftarrow \pi_i + \gamma \left(\frac{J^{\oplus}}{\max(n^{\oplus},1)} - \frac{J^{\ominus}}{\max(n^{\ominus},1)} \right)$;
23: $\pi_i \leftarrow \max\left(\min\left(\pi_i, \pi_{max}\right), \pi_{min}\right)$; ▷ Saturation
24: **end for**
25: **until** Stopping criterion

6.6 Numerical examples

In this example we consider a system S_4 in the form of (6.1) with 5 states and outputs, described by the following matrices:

$$\boldsymbol{F} = \begin{bmatrix} 0.0964 & 0.1577 & 0.2783 & -0.0983 & 0.1116 \\ 0.1354 & 0.0687 & 0.1230 & -0.1257 & -0.2379 \\ 0.2974 & 0.0913 & -0.1032 & -0.0937 & 0.0525 \\ -0.1010 & -0.1266 & -0.0891 & 0.4644 & -0.0420 \\ 0.0863 & -0.2380 & 0.0884 & -0.0410 & -0.0409 \end{bmatrix},$$

$$\boldsymbol{H} = \begin{bmatrix} 0.2380 & -0.5470 & 2.0726 & 0 & 1.4756 \\ -0.0458 & 0 & -0.7593 & 0.9201 & -1.5044 \\ 0.0523 & 1.7044 & -1.1369 & -0.0254 & 0.8159 \\ 0 & -0.1391 & 0 & -1.4746 & -0.3703 \\ 0.1182 & 0.0666 & 0.3504 & 2.1646 & 0.1038 \end{bmatrix}.$$

The data were generated using a sparse $\boldsymbol{Q}$ matrix:

$$\boldsymbol{Q} = \begin{bmatrix} 0.3854 & 0.1160 & 0 & -0.4516 & 0 \\ 0.1160 & 0.3225 & 0 & -0.0832 & -0.0643 \\ 0 & 0 & 0.1000 & 0 & 0 \\ -0.4516 & -0.0832 & 0 & 0.6249 & -0.0165 \\ 0 & -0.0643 & 0 & -0.0165 & 0.0251 \end{bmatrix},$$

and

$$\boldsymbol{R} = \text{diag}\left(0.1055,\ 0.1402,\ 0.0978,\ 0.1916,\ 0.3802\right).$$

A sparse $\boldsymbol{Q}$ can be representative, *e.g.*, of distributed systems with various interconnected devices, whereby only state variables associated to a device or to neighboring devices are correlated.

To account for the randomization inherent in the MSS algorithm 6, an MC study has been carried out by running the algorithm 100 times on the same data realization. The algorithm has been set up with $N_p = 40$, $\beta = 1$, $\mu_{min} = 0.001$, and $\mu_{max} = 0.999$. A common issue in solving optimization problems which include regularization terms is the selection of the optimal regularization weight λ. A popular method for the selection of this parameter is the *L-curve criterion* [67] [50]. It is based on the study of the *L-curve* which is a log-log plot of the norm of a regularized solution versus the norm of the corresponding residual. It is a convenient graphical tool for displaying the trade-off between the size of a regularized solution and its fit to the given data, as the regularization parameter varies. According to the performed *L-curve* analysis (see Figure 6.5), a λ value equal to 0.02 is used. Finally, as suggested in [36], we adopt a

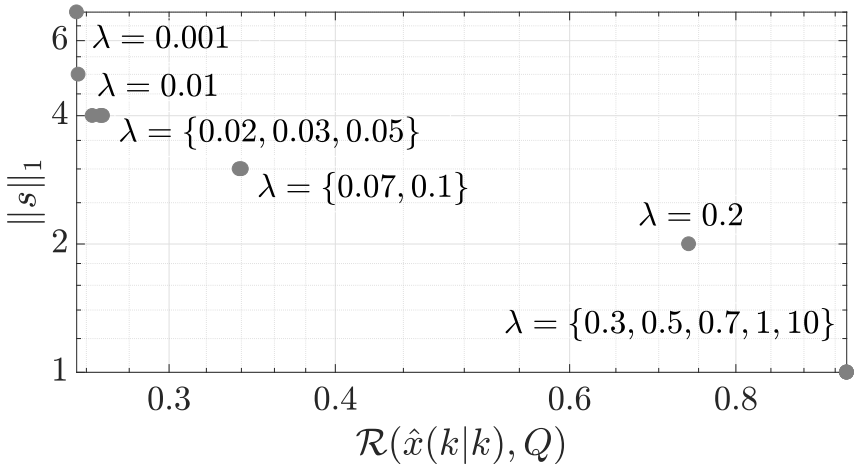

Figure 6.5: *MSS - L-*curve*. The values have been computed on the training data-set.*

time varying step size χ with the following law:

$$\chi = \frac{1}{10\left(\mathcal{J}_{\text{best}} - \overline{\mathcal{J}}\right) + 0.1} \tag{6.26}$$

where $\mathcal{J}_{\text{best}} = \texttt{max}\left(\mathcal{J}^{(p)}\right)$ and $\overline{\mathcal{J}} = \texttt{mean}\left(\mathcal{J}^{(p)}\right)$.

The aggregated results are reported in Table 6.1. One can note that the most frequently selected structure corresponds to $\boldsymbol{Q}$ matrices with 5 DOFs in the form:

$$\boldsymbol{Q} = \begin{bmatrix} q_{11} & 0 & 0 & q_{14} & 0 \\ 0 & q_{22} & 0 & 0 & 0 \\ 0 & 0 & q_{33} & 0 & 0 \\ q_{14} & 0 & 0 & q_{44} & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

In the remaining 3% of cases, the algorithm selected the following structures (defined in terms of the z_{ij} terms):

$$\begin{aligned} &(z_{11},\ z_{41},\ z_{22},\ z_{42},\ z_{43}), \\ &(z_{11},\ z_{41},\ z_{22},\ z_{42},\ z_{33}), \\ &(z_{11},\ z_{21},\ z_{22},\ z_{42},\ z_{43}). \end{aligned}$$

It is worth mentioning that the algorithm converges to the mentioned solution by exploring a tiny fraction of the Σ space, *i.e.,* 611.29 explored solutions (on average) w.r.t. 32768 possible structures.

Figure 6.6 shows a comparison between a portion of the real x_1 trajectory and those estimated by employing in the filter design the true $\boldsymbol{Q}$ (denoted Q_{true}), the CM estimated based on the structure suggested by the MSS algorithm (Q_{opt}), as well as the optimal diagonal one (Q_{diag}). The curves displayed in Figure 6.6 have been computed on a test data-set. Apparently, the lack of information about q_{14} in the diagonal structure, prevents the filter from estimating accurately the x_1 trajectory. On the

other hand, the filter based on the structure-optimized Q_{opt} yields state estimates very close to those obtained with Q_{true}. This also reflects on the corresponding values of $\mathcal{R}(\hat{\boldsymbol{x}}(k|k), \boldsymbol{Q})$, respectively $\mathcal{R}(\hat{\boldsymbol{x}}(k|k), Q_{\text{true}}) = 0.2280$, $\mathcal{R}(\hat{\boldsymbol{x}}(k|k), Q_{\text{opt}}) = 0.2557$, and $\mathcal{R}(\hat{\boldsymbol{x}}(k|k), Q_{\text{diag}}) = 0.5123$.

Table 6.1: *MSS - Monte Carlo study results.*

Average elapsed time [s]	381.34
Average number of iterations	50.19
Average # of explored matrix structures	611.29
Total # of possible matrix structures	$2^{15} = 32768$
Selected structure (z_{ij} terms)	$(z_{11},\ z_{41},\ z_{22},\ z_{33})$
Extraction of the selected structure [%]	97

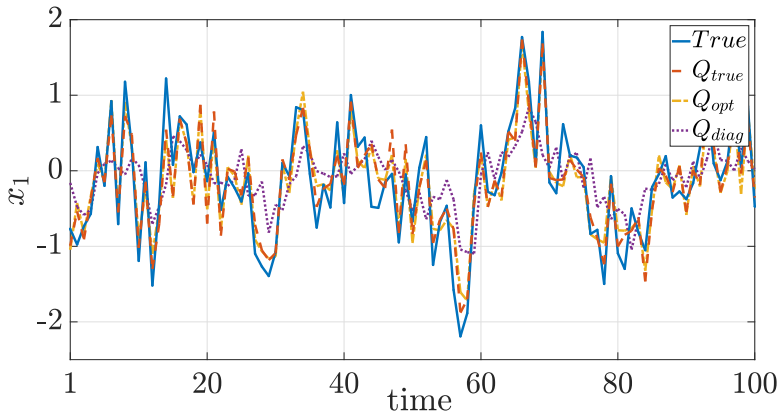

Figure 6.6: *MSS - State estimation: real and estimated trajectories for state variable x_1. The state estimates are obtained by employing Q_{true}, Q_{opt}, and Q_{diag}. All the trajectories are computed on a test data-set.*

6.7 Conclusions

We investigated three main aspects regarding the identification of the noise process CM in Kalman filtering problems. First, we discussed the identifiability issues associated to the estimation of $\boldsymbol{Q}$ from data, related to the over-parametrization in the multidimensional case. Indeed, as already observed in [83], the tuning of the CMs is based on the estimation of the Kalman gain, which usually provides less degrees of freedom than those associated with the CM. Even if this is a well known problem, the literature lacks a full analysis of the conditions under which the noise CMs elements can be estimated. A possible motivation to this deficiency may be the capability of directly estimating the Kalman gain independently of the knowledge of the CMs [83], if one is addressing a state estimation problem. Or even, the possibility in several applications to employ satisfactorily CMs with diagonal structures, thus overcoming the mentioned identifiability issue. However, the noise model is itself is a crucial complement of the process model,

which typically results from some approximations, and hence one may be interested in assessing the confidence in the identified model *per se*. This key observation opened a window onto the problem of choosing a suitable parametrization for $\boldsymbol{Q}$. Accordingly, as a second point of discussion, we questioned the common practice of assuming a diagonal structure for $\boldsymbol{Q}$, showing that this is not always the best choice. Finally, building upon the last observation, we developed a method for solving explicitly the structure selection problem, by decoupling it from the estimation problem. The proposed method relies on the fact that the matrix structure selection problem is a specialization of a classical MSS problem, and hence we tackled it with a randomized method which allows to handle its combinatorial complexity.

CHAPTER 7

Concluding remarks

7.1 Conclusions

In this thesis, the problem of model structure selection (MSS) for dynamical systems has been investigated with reference to the identification of nonlinear systems when data are distributed among agents and that of switched nonlinear systems, and the estimation of the process noise statistics in Kalman filtering. Each scenario has its own peculiarities that impact on the MSS problem, which is a complex combinatorial problem *per se*. The proposed solutions are all based on a probabilistic reformulation of the selection problem whereby a probability distribution is defined over the model structure space. Accordingly, we proposed adequate *sample-and-evaluate* procedures that allow to finitely tune the parameters of the introduced probability distribution in order to concentrate its probability mass on a target structure.

With reference to the cooperative identification of a NARX model from data that are distributed across multiple agents, we proposed a distributed scheme to address jointly the MSS and the parameter estimation. Specifically, the conceived scheme exploits the probabilistic reformulation to transform the purely combinatorial MSS task into a continuous optimization problem which is tackled in a way akin to distributed algorithms based on sub-gradient and proximal minimization. While the algorithm convergence in the case of centrally available data has been proved, the method, in its current form, lacks a formal theoretical proof. Nonetheless, the obtained results both on numerical examples and on a real application show the potential of the proposed algorithm.

The MSS problem for switched systems encompasses the selection of a model structure for each mode and also the reconstruction of the switching signal. This latter task has a detrimental impact on the combinatorial complexity, if no *a priori* knowledge on the discrete nature of the system is available. The rationale behind the proposed

two-stage procedure is that of keeping under control the problem complexity by alternating between an identification phase with a limited number of fixed switching times and a switching locations refinement step. This is done by means of a Monte Carlo sampling approach, where the probability to extract a model is expressed separately in terms of extraction of switching sequences and submodel structures. The local convergence of this approach has been proved. The proposed approach has been validated on benchmark examples regarding the identification of both linear and nonlinear switched systems, and it fared well also with respect to existing approaches based on *e.g.,* kernel regression and Support Vector Machines, or Sum of Norm regularization. Numerical evidence showed that the method can be extended fruitfully to piecewise systems if a region estimation step is carried out once the identification procedure ends.

In state reconstruction problems, an incorrect description of the model stochastic properties significantly affects the final filtering performance. A typical practical assumption is that the process and noise covariances can be parameterized as diagonal matrices. However, we showed that this is not always the best compromise between computational complexity and tracking accuracy, and hence a matrix structure selection problem arises. We considered this selection problem as a specialization of a classical MSS problem, and we treated it within the presented probabilistic framework. The effectiveness of the proposed approach is illustrated by means of some numerical examples.

7.2 Future works

The work in this thesis can be improved and extended in the following directions:

- The core of the proposed methods is an heuristic approach to combinatorial problems arising in system identification, based on a continuous relaxation of the integer variables via discrete probability distributions. It would be interesting to investigate if this idea applies also to other problems characterized by having a mixed-integer nature.
- Regarding the identification of NARX models via distributed computation, a theoretical proof of the algorithm convergence is still missing.
- An on-line version of the proposed identification procedure for switched nonlinear systems would deserve attention.
- One could study the structure selection for the covariance matrices in Kalman filter applications when the underlying system in nonlinear.

"The important thing is not to stop questioning.
Curiosity has its own reason for existing."

— Albert Einstein

List of Figures

List of Tables

List of Algorithms

Bibliography

[1] L. A. Aguirre, B. H. G. Barbosa, and A. P. Braga. Prediction and simulation errors in parameter estimation for nonlinear systems. *Mechanical Systems and Signal Processing*, 24(8):2855–2867, 2010.

[2] L. A. Aguirre and S. A. Billings. Dynamical effects of overparametrization in nonlinear models. *Physica*, 80(4):26–40, 1995.

[3] B. D. Anderson and J. B. Moore. Optimal filtering. *Englewood Cliffs*, 21:22–95, 1979.

[4] M. Avellina, A. Brankovic, and L. Piroddi. Distributed randomized model structure selection for NARX models. *International Journal of Adaptive Control and Signal Processing*, 31(12):1853–1870, 2017.

[5] L. Bako. Identification of switched linear systems via sparse optimization. *Automatica*, 47(4):668–677, 2011.

[6] L. Bako, K. Boukharouba, and S. Lecoeuche. An l_0–l_1 norm based optimization procedure for the identification of switched nonlinear systems. In 49^{th} *IEEE Conference on Decision and Control*, pages 4467–4472, 2010.

[7] T. Baldacchino, S. R. Anderson, and V. Kadirkamanathan. Computational system identification for Bayesian NARMAX modeling. *Automatica*, 49:2641–2651, 2013.

[8] P. R. Bélanger. Estimation of noise covariance matrices for a linear time-varying stochastic process. *Automatica*, 10(3):267–275, 1974.

[9] A. Bemporad, V. Breschi, D. Piga, and S. P. Boyd. Fitting jump models. *Automatica*, 96:11–21, 2018.

[10] A. Bemporad, A. Garulli, S. Paoletti, and A. Vicino. A bounded-error approach to piecewise affine system identification. *IEEE Transactions on Automatic Control*, 50(10):1567–1580, 2005.

[11] K. P. Bennett and O. L. Mangasarian. Robust linear programming discrimination of two linearly inseparable sets. *Optimization methods and software*, 1(1):23–34, 1992.

[12] K. P. Bennett and O. L. Mangasarian. Multicategory discrimination via linear programming. *Optimization methods and Software*, 3(1-3):27–39, 1994.

[13] D. Bertsimas, A. King, and R. Mazumder. Best subset selection via a modern optimization lens. *The annals of statistics*, 44(2):813–852, 2016.

[14] F. Bianchi, A. Falsone, M. Prandini, and L. Piroddi. A randomised approach for NARX model identification based on a multivariate Bernoulli distribution. *International Journal of Systems Science*, 48(6):1203–1216, 2017.

[15] F. Bianchi, A. Falsone, M. Prandini, and L. Piroddi. Nonlinear system identification with model structure selection via distributed computation. *58th IEEE Conference on Decision and Control*, 2019.

[16] F. Bianchi, S. Formentin, and L. Piroddi. Process noise covariance estimation via stochastic approximation. *International Journal of Adaptive Control and Signal Processing*, 2019.

[17] F. Bianchi, S. Formentin, and L. Piroddi. Structure selection of noise covariance matrices for linear kalman filter design. *Accepted for publication in the proceedings of the 2020 European Control Conference*, 2020.

[18] F. Bianchi, M. Prandini, and L. Piroddi. A randomized approach to switched nonlinear systems identification. *18th IFAC Symposium on System Identification, SYSID 2018*, 2018.

[19] F. Bianchi, M. Prandini, and L. Piroddi. A randomized two-stage iterative method for switched nonlinear systems identification. *Nonlinear Analysis: Hybrid Systems*, 35:100818, 2020.

[20] S. Billings, S. Chen, and M. Korenberg. Identification of MIMO nonlinear systems using a forward-regression orthogonal estimator. *International Journal of Control*, 49:2157–2189, 1989.

[21] S. A. Billings. *Nonlinear System Identification: NARMAX Methods in the Time, Frequency, and Spatio-Temporal Domains*. Wiley, 2013.

[22] M. Bonin, V. Seghezza, and L. Piroddi. NARX model selection based on simulation error minimisation and LASSO. *IET Control Theory & Applications*, 4(7):1157–1168, 2010.

[23] I. Boniolo, S. Savaresi, and M. Tanelli. Roll angle estimation in two-wheeled vehicles. *IET Control Theory & Applications*, 3(1):20–32, 2009.

[24] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–122, 2011.

[25] S. Boyd and L. Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[26] S. Boyd, L. Xiao, and A. Mutapcic. Subgradient methods. *Lecture notes of EE392o, Stanford University, Autumn Quarter*, 2004:2004–2005, 2003.

[27] E. J. Bredensteiner and K. P. Bennett. Multicategory classification by support vector machines. In *Computational Optimization*, pages 53–79. Springer, 1999.

[28] V. Breschi, D. Piga, and A. Bemporad. Piecewise affine regression via recursive multiple least squares and multicategory discrimination. *Automatica*, 73:155–162, 2016.

[29] S. S. Chen, D. L. Donoho, and M. A. Saunders. Atomic decomposition by basis pursuit. *SIAM review*, 43(1):129–159, 2001.

[30] C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.

[31] M. Deistler. System identification and time series analysis: Past, present, and future. In *Stochastic Theory and Control*, pages 97–109. Springer, 2002.

[32] L. Devroye. Nonuniform random variate generation. *Handbooks in operations research and management science*, 13:83–121, 2006.

[33] J. Duník, O. Straka, O. Kost, and J. Havlík. Noise covariance matrices in state-space models: A survey and comparison of estimation methods - Part I. *International Journal of Adaptive Control and Signal Processing*, 31(11):1505–1543, 2017.

[34] T. Falck, H. Ohlsson, L. Ljung, J. A. Suykens, and B. De Moor. Segmentation of time series from nonlinear dynamical systems. *IFAC Proceedings Volumes*, 44(1):13209–13214, 2011.

[35] A. Falsone, L. Piroddi, and M. Prandini. A novel randomized approach to nonlinear system identification. In *53rd IEEE Conference on Decision and Control*, pages 6516–6521. IEEE, 2014.

[36] A. Falsone, L. Piroddi, and M. Prandini. A randomized algorithm for nonlinear model structure selection. *Automatica*, 60:227–238, 2015.

[37] G. Ferrari-Trecate, M. Muselli, D. Liberati, and M. Morari. A clustering technique for the identification of piecewise affine systems. *Automatica*, 39(2):205–217, 2003.

[38] M. A. Fischler and R. C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[39] P. J. Fleming and R. C. Purshouse. Evolutionary algorithms in control systems engineering: a survey. *Control Engineering Practice*, 10(11):1223–1241, 2002.

[40] C. M. Fonseca and P. J. Fleming. Non-linear system identification with multiobjective genetic algorithms. *IFAC Proceedings Volumes*, 29(1):1169–1174, 1996.

[41] S. Formentin and S. Bittanti. An insight into noise covariance estimation for Kalman filter design. *IFAC Proceedings Volumes*, 47(3):2358–2363, 2014.

[42] S. Formentin, I. Boniolo, P. Lisanti, C. Spelta, and S. M. Savaresi. Fault detection in roll angle estimation for two-wheeled vehicles. *IFAC Proceedings Volumes*, 45(24):54–59, 2012.

[43] J. Friedman, T. Hastie, and R. Tibshirani. *The elements of statistical learning*, volume 1. Springer series in statistics, 2001.

[44] A. Garulli, S. Paoletti, and A. Vicino. A survey on switched and piecewise affine system identification. In 16^{th} *IFAC Symposium on System Identification*, pages 344–355, Brussels, Belgium, July 11-13 2012.

[45] R. Goebel, R. G. Sanfelice, and A. R. Teel. Hybrid dynamical systems. *IEEE Control Systems Magazine*, 29(2):28–93, 2009.

[46] M. S. Grewal. Kalman filtering. In *International Encyclopedia of Statistical Science*, pages 705–708. Springer, 2011.

[47] C. Guestrin, P. Bodik, R. Thibaux, M. Paskin, and S. Madden. Distributed regression: an efficient framework for modeling sensor network data. In *Third International Symposium on Information Processing in Sensor Networks*, pages 1–10. ACM, 2004.

[48] Y. Guo, L. Z. Guo, S. A. Billings, and H. L. Wei. An iterative orthogonal forward regression algorithm. *International Journal of Systems Science*, 46:776–789, 2015.

[49] R. Haber and H. Unbehauen. Structure identification of nonlinear dynamic systems – a survey on input/output approaches. *Automatica*, 26(4):651–677, 1990.

[50] P. C. Hansen. Analysis of discrete ill-posed problems by means of the l-curve. *SIAM review*, 34(4):561–580, 1992.

[51] A. Hartmann, J. M. Lemos, R. S. Costa, J. Xavier, and S. Vinga. Identification of switched ARX models via convex optimization and expectation maximization. *Journal of Process Control*, 28:9–16, 2015.

[52] A. E. Hoerl and R. W. Kennard. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12(1):55–67, 1970.

[53] X. Hong, R. Mitchell, S. Chen, C. Harris, K. Li, and G. Irwin. Model selection approaches for non-linear system identification: a review. *International Journal of Systems Science*, 39(10):925–946, 2008.

[54] G. James, D. Witten, T. Hastie, and R. Tibshirani. *An introduction to statistical learning*, volume 112. Springer, 2013.

[55] I. Jolliffe. *Principal component analysis*. Springer, 2011.

[56] I. T. Jolliffe. A note on the use of principal components in regression. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 31(3):300–303, 1982.

[57] A. L. Juloski, W. Heemels, and G. Ferrari-Trecate. Data-based hybrid modelling of the component placement process in pick-and-place machines. *Control Engineering Practice*, 12(10):1241–1252, 2004.

[58] A. L. Juloski, S. Weiland, and W. Heemels. A Bayesian approach to identification of hybrid systems. *IEEE Transactions on Automatic Control*, 50(10):1520–1533, 2005.

[59] R. E. Kalman. A new approach to linear filtering and prediction problems. *Journal of Basic Engineering*, 82(1):35–45, 1960.

[60] R. E. Kalman and R. S. Bucy. New results in linear filtering and prediction theory. *Journal of Basic Engineering*, 83(1):95–108, 1961.

[61] R. Kashyap. Maximum likelihood identification of stochastic linear systems. *IEEE Transactions on Automatic Control*, 15(1):25–34, 1970.

[62] R. Kohavi et al. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Ijcai*, volume 14, pages 1137–1145. Montreal, Canada, 1995.

[63] F. Lauer and G. Bloch. Switched and piecewise nonlinear hybrid system identification. In *International Workshop on Hybrid Systems: Computation and Control*, pages 330–343, 2008.

[64] F. Lauer and G. Bloch. Piecewise smooth system identification in reproducing kernel hilbert space. In *53rd IEEE Conference on Decision and Control*, pages 6498–6503. IEEE, 2014.

[65] F. Lauer, G. Bloch, and R. Vidal. Nonlinear hybrid system identification with kernel models. In 49^{th} *IEEE Conference on Decision and Control*, pages 696–701, 2010.

[66] F. Lauer, G. Bloch, and R. Vidal. A continuous optimization framework for hybrid system identification. *Automatica*, 47(3):608–613, 2011.

[67] C. Lawson and R. J. R, Hanson. *Solving least squares problems*, volume 15. Siam, 1995.

[68] V. L. Le, G. Bloch, and F. Lauer. Reduced-size kernel models for nonlinear hybrid system identification. *IEEE Transactions on Neural Networks*, 22(12):2398–2405, 2011.

[69] V. L. Le, F. Lauer, L. Bako, and G. Bloch. Learning nonlinear hybrid systems: from sparse optimization to support vector regression. In *Proceedings of the* 16^{th} *International Conference on Hybrid systems: computation and control*, pages 33–42, 2013.

[70] I. J. Leontaritis and S. A. Billings. Input-output parametric models for non-linear systems part I: deterministic non-linear systems. *International Journal of Control*, 41(2):303–328, 1985.

[71] I. J. Leontaritis and S. A. Billings. Input-output parametric models for non-linear systems part II: stochastic non-linear systems. *International Journal of Control*, 41(2):329–344, 1985.

[72] K. Li, J. Peng, and G. Irwin. A fast nonlinear model identification method. *IEEE Transactions on Automatic Control*, 50(8):1211–1216, 2005.

[73] P. Li, H.-L. Wei, S. A. Billings, M. A. Balikhin, and R. Boynton. Nonlinear model identification from multiple data sets using an orthogonal forward search algorithm. *Journal of Computational and Nonlinear Dynamics*, 8(4):041001, 2013.

[74] L. Ljung. *System identification: theory for the user*. Prentice Hall, 1999.

[75] L. Ljung. Perspectives on system identification. *Annual Reviews in Control*, 34(1):1–12, 2010.

[76] J. Lunze and F. Lamnabhi-Lagarrigue. *Handbook of hybrid systems control: theory, tools, applications*. Cambridge University Press, 2009.

[77] Y. Ma and R. Vidal. Identification of deterministic switched ARX systems via identification of algebraic varieties. In *International Workshop on Hybrid Systems: Computation and Control*, pages 449–465, 2005.

[78] K. Mao and S. Billings. Variable selection in non-linear systems modelling. *Mechanical Systems and Signal Processing*, 13(2):351–366, 1999.

[79] K. Margellos, A. Falsone, S. Garatti, and M. Prandini. Distributed constrained optimization and consensus in uncertain networks via proximal minimization. *IEEE Transactions on Automatic Control*, 63(5):1372–1387, May 2018.

[80] I. Maruta, T. Sugie, and T.-H. Kim. Identification of multiple mode models via distributed particle swarm optimization. In *Proceedings of the* 18^{th} *IFAC World Congress*, pages 7743–7748, Milano, Italy, Aug. 28 – Sept. 2 2011.

[81] G. Mateos, J. A. Bazerque, and G. B. Giannakis. Distributed sparse linear regression. *IEEE Transactions on Signal Processing*, 58(10):5262–5276, 2010.

[82] Matlab optimization toolbox, Version 7.6. The MathWorks, Natick, MA, USA.

[83] R. Mehra. On the identification of variances and adaptive Kalman filtering. *IEEE Transactions on Automatic Control*, 15(2):175–184, 1970.

[84] A. Miller. *Subset selection in regression*. Chapman and Hall/CRC, 2002.

[85] B. K. Natarajan. Sparse approximate solutions to linear systems. *SIAM Journal on Computing*, 24(2):227–234, 1995.

[86] S. Nazari, Q. Zhao, and B. Huang. An improved algebraic geometric solution to the identification of switched ARX models with noise. In *Proceedings of the American Control Conference*, pages 1230–1235, 2011.

[87] A. Nedic and A. Ozdaglar. Distributed subgradient methods for multi-agent optimization. *IEEE Transactions on Automatic Control*, 54(1):48–61, 2009.

[88] B. J. Odelson, M. R. Rajamani, and J. B. Rawlings. A new autocovariance least-squares method for estimating noise covariances. *Automatica*, 42:303–308, 2006.

[89] H. Ohlsson and L. Ljung. Identification of switched linear regression models using sum-of-norms regularization. *Automatica*, 49(4):1045–1050, 2013.

[90] R. Olfati-Saber and J. S. Shamma. Consensus filters for sensor networks and distributed sensor fusion. In *Proceedings of the 44th IEEE Conference on Decision and Control*, pages 6698–6703. IEEE, 2005.

[91] N. Ozay, M. Sznaier, C. M. Lagoa, and O. I. Camps. A sparsification approach to set membership identification of switched affine systems. *IEEE Transactions on Automatic Control*, 57(3):634–648, 2012.

[92] P. Palumbo and L. Piroddi. Seismic behaviour of bruttress dams: nonlinear modeling of a damaged buttress based on ARX/NARX models. *Journal of Sound and Vibration*, 239:405–422, 2000.

[93] S. Paoletti. *Identification of piecewise affine models*. PhD thesis, Dipartimento di Ingegneria dell'Informazione, Univ. Siena, Siena, Italy, 2004.

[94] S. Paoletti, A. L. Juloski, G. Ferrari-Trecate, and R. Vidal. Identification of hybrid systems a tutorial. *European Journal of Control*, 13(2–3):242–260, 2007.

[95] N. Parikh, S. Boyd, et al. Proximal algorithms. *Foundations and Trends in Optimization*, 1(3):127–239, 2014.

[96] D. Piga and R. Tóth. An SDP approach for ℓ_0-minimization: Application to ARX model segmentation. *Automatica*, 49(12):3646–3653, 2013.

[97] G. Pillonetto. A new kernel-based approach to hybrid system identification. *Automatica*, 70:21–31, 2016.

[98] L. Piroddi and W. Spinelli. An identification algorithm for polynomial NARX models based on simulation error minimization. *International Journal of Control*, 76(17):1767–1781, 2003.

[99] G. Pozzi, S. Formentin, P. Lluka, and S. Bittanti. A Kalman filtering approach to traffic flow estimation in computer networks. *IFAC-PapersOnLine*, 51(15):37–42, 2018.

[100] M. R. Rajamani and J. B. Rawlings. Estimation of the disturbance structure from data using semidefinite programming and optimal weighting. *Automatica*, 45(1):142–148, 2009.

[101] S. S. Ram, A. Nedić, and V. V. Veeravalli. Distributed stochastic subgradient projection algorithms for convex optimization. *Journal of optimization theory and applications*, 147(3):516–545, 2010.

[102] R. Rifkin and A. Klautau. In defense of one-vs-all classification. *Journal of machine learning research*, 5(Jan):101–141, 2004.

[103] K. Rodriguez-Vazquez, C. M. Fonseca, and P. J. Fleming. Identifying the structure of nonlinear dynamic systems using multiobjective genetic programming. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, 34(4):531–545, 2004.

[104] J. Roll, A. Bemporad, and L. Ljung. Identification of piecewise affine systems via mixed-integer programming. *Automatica*, 40(1):37–50, 2004.

[105] T. Sapatinas. Statistics for high-dimensional data, 2012.

[106] R. H. Shumway and D. S. Stoffer. Time series regression and exploratory data analysis. In *Time series analysis and its applications*, pages 47–82. Springer, 2011.

[107] J. Sjöberg, Q. Zhang, L. Ljung, A. Benveniste, B. Delyon, P. Y. Glorennec, H. Hjalmarsson, and A. Juditsky. Nonlinear black-box modeling in system identification: a unified overview. *Automatica*, 31(12):1691–1724, 1995.

[108] T. Söderström and P. Stoica, editors. *System Identification*. London: Prentice-Hall, 1989.

[109] B. Southall, B. F. Buxton, and J. A. Marchant. Controllability and observability: Tools for Kalman filter design. In *British Machine Vision Conference*, pages 1–10, 1998.

[110] J. Speyer, J. Deyst, and D. Jacobson. Optimization of stochastic linear systems with additive measurement and process noise using exponential performance criteria. *IEEE Transactions on Automatic Control*, 19(4):358–366, 1974.

[111] S. Sundhar Ram, A. Nedić, and V. V. Veeravalli. A new class of distributed optimization algorithms: Application to regression of distributed data. *Optimization Methods and Software*, 27(1):71–88, 2012.

[112] R. Tempo, G. Calafiore, and F. Dabbene. *Randomized algorithms for analysis and control of uncertain systems: with applications*. Springer Science & Business Media, 2012.

[113] R. Tibshirani. Regression shrinkage and selection via the LASSO. *Journal of the Royal Statistical Society: Series B (Methodological)*, 58(1):267–288, 1996.

[114] A. J. Van Der Schaft and J. M. Schumacher. *An introduction to hybrid dynamical systems*, volume 251. Springer London, 2000.

[115] B. C. Wallet, D. J. Marchette, J. L. Solka, and E. J. Wegman. A genetic algorithm for best subset selection in linear regression. *Computing Science and Statistics*, pages 545–550, 1997.

[116] H.-L. Wei and S. Billings. Model structure selection using an integrated forward orthogonal search algorithm assisted by squared correlation and mutual information. *International Journal of Modelling, Identification and Control*, 3(4):341–356, 2008.

[117] H.-L. Wei and S. A. Billings. Improved model identification for non-linear systems using a random subsampling and multifold modelling (RSMM) approach. *International Journal of Control*, 82(1):27–42, 2009.

[118] I. H. Witten, E. Frank, M. A. Hall, and C. J. Pal. *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann, 2016.

[119] I. M. Yassin, M. N. Taib, N. A. Rahim, M. K. M. Salleh, and H. Z. Abidin. Particle swarm optimization for NARX structure selection - application on DC motor model. In *2010 IEEE Symposium on Industrial Electronics and Applications (ISIEA)*, pages 456–462. IEEE, 2010.

[120] I. M. Yassin, M. N. Taib, A. Zabidi, H. A. Hassan, and H. Z. Abidin. Comparison between NARX parameter estimation methods with binary particle swarm optimization-based structure selection method. In *IEEE Congress on Evolutionary Computation*, pages 1–7. IEEE, 2010.