Università degli Studi di Padova

Dipartimento di Ingegneria dell'Informazione

Advances in System Identification:

Gaussian Regression and Robot Inverse Dynamics Learning

Ph.D. candidate
Diego Romeres

Advisor
Prof. Alessandro Chiuso

Co-Advisor
Prof. Gianluigi Pillonetto

Director & Coordinator
Prof. Matteo Bertocco

Ph.D. School in
Information Engineering

Department of
Information Engineering
University of Padova
2017

Abstract

Nonparametric Gaussian regression models are powerful tools for supervised learning problems. Recently they have been introduced in the field of system identification as an alternative to classical parametric models used in prediction error methods. The focus of this thesis is the analysis and the extension of linear Gaussian regression models and their applications to the identification of the inverse dynamics of robotic platforms.

When Gaussian processes are applied to linear systems identification, according to the Bayesian paradigm the impulse response is modeled a priori with a Gaussian distribution encoding the desired structural properties of the dynamical system (e.g. smoothness, BIBO stability, sparsity, etc.). The inference on the impulse response estimate is obtained through the posterior distribution which combines the information of the a priori distribution together with the information given by the data.

The Bayesian framework naturally allows the adaptation of the model class and its complexity while also accounting for uncertainty and noise, thus providing a robust mean to trade bias versus variance. On the other hand, one disadvantage of these nonparametric methods is that their aim to identify directly the impulse response of the predictor model does not guarantee the stability of the forward model. These general advantages and disadvantages inspired the research on this manuscript.

Gaussian Regression and Parametric PEM: a Comparison. The term of comparison for these Gaussian regression models will be the classical parametric technique. In addition to an analysis of the two approaches in terms of error in fitting the impulse response estimates, we are interested in comparing the confidence intervals around these estimates. A new definition of the confidence intervals is proposed in order to pave the way for a fair comparison between the two approaches. Numerical simulations show that the Bayesian estimates have higher prediction performance.

Online Gaussian Regression. In an on-line system identification setting, new data become available at given time steps and real-time estimation requirements have to be satisfied. The goal is to compute the model estimate with low and fixed computational complexity and a reduced memory storage. We developed a tailored Bayesian procedure which updates the quantities to compute the marginal likelihood and the impulse response estimate iteratively and performs the estimation of the hyperparameters by computing only one iteration of a suitable optimization algorithm to maximize the marginal likelihood. Both quasi-Newton methods and EM algorithm are adopted as optimization algorithms. When time-varying systems are considered, the property of 'forgetting the past data' is required. Accordingly we propose two schemes: the usage of a temporal window which slides over the data and the usage of a forgetting factor which is a variable that exponentially decreases the weight of the old data. In particular, we propose to consider the forgetting factor both as a fixed constant or as an estimating variable. The proposed nonparametric procedures have satisfactory performances compared to the batch algorithm and outperform the classical parametric approaches both in terms of computational time and adherence of the model estimate to the true one.

Enforcing model stability in nonparametric Gaussian regression. The main idea of the Bayesian approach is to frame linear system identification as predictor estimation in an infinite dimensional space with the aid of regularization techniques. This approach is based on the prediction error minimization and can guarantee the identification of stable predictors. Unfortunately, the stability of the predictors does not guarantee the stability of the impulse response of the forward model in general. Various techniques are successfully proposed to guarantee the stability of this model.

Online semiparametric learning for inverse dynamics modeling. Dynamic models can be obtained from the first principles of mechanics, using the so called Rigid Body Dynamics. This approach results in a parametric model in which the values of physically meaningful parameters must be provided in order to complete the fixed structure of the model. Alternatively, the nonparametric Gaussian regression modeling can be employed extrapolating the dynamics directly from the experimental data, without making any unrealistic approximation on the physical system (e.g. assuming linear frictions models, ignoring the dynamics of the hydraulic actuators, etc.). Nevertheless, nonparametric models deteriorate their performance when predicting unseen data that are not in the "neighbourhood" of the training dataset. In order to exploit the advantages of both techniques, semiparametric models which combine the parametric and the nonparametric models are analyzed.

Acknowledgement

To my family Chiara, Giuseppe, Irene, Elisa, Davide and Viola who supported me through all these years and without who this thesis would not have been possible.

To my Advisor Alessandro, who introduced me with passion to the world of system identification and wisely guided in these years of research.

To all the members of Office 330 and the other PhD and PostDoc researchers who not only helped and inspired my research but filled the working place of new friends. In particular, I thank Giulia for all the idea shared, the long philosophical discussions about Bayesian system identification and the nice moments spent together.

To all the persons and my numerous and irreplaceable friends who recently or in the past have been present in my life and made this thesis possible.

Contents

Notational Conventions

Symbols

n	Dimension of the impulse response
N	Number of data
θ	Parameter vector
θ_0	Parameters of the true system
Θ	Parameter space
d	Dimension of the parameters θ
η	Hyperparameters
d_η	Dimension of the hyperparameters θ
$y(t)$	System output at time t
$u(t)$	System input at time t
$e(t)$	System noise at time t, typically assumed to be a white noise
$\mathcal{D}^N$	Dataset composed of N input-output data pairs
f^*	True predictor model
g	Impulse response from u to y
h	Impulse response from e to y
t^-	Vector of the past time instants $t^- = t-1, t-2, \ldots$
$y(t^-)$	Vector of the past measurements at time t, $y(t^-) = [y(t-1)\, y(t-2)\, \ldots]^\top$
$\hat{y}(t\|t^-)$	Predicted value at time instant t based on the data up to time $t-1$
I_N	Identity matrix of dimension N
$\chi^2(d)$	Chi-squared distribution with $d-$degree of freedom
$O_{n\times d}$	Matrix composed of zeros with dimension $n \times d$
$\mathcal{M}$	Model structure
$\mathcal{M}(\theta)$	Model structure depending on the parameters θ

Operators

$\mathbb{E}[\cdot]$	Expectation
$\mathbb{E}[\cdot\|\cdot]$	Conditional expectation
$\mathbb{E}_q[\cdot]$	Expectation w.r.t. the probability density q
$Pr\{A\}$	Probability of the event A
$\mathcal{N}(\mu, \Sigma)$	Multivariate Gaussian (normal) distribution with mean vector μ and covariance matrix Σ
$\mathcal{U}(a, b)$	Uniform distribution with support defined in the interval $[a, b]$
$p(x)$	Probability density function of the random variable x
$p_x(x')$	Probability density function of the random variable x evaluated at x'. This notation will be used only when necessary, the notation $p(x)$ is in general preferred.
$\otimes$	Kronecker product
$\arg\min_x f(x)$	Optimal solution of the minimization problem of $f(x)$
$\arg\max_x f(x)$	Optimal solution of the maximization problem of $f(x)$
A^{-1}	Inverse of the matrix A
$\mathbb{R}$	Set of real numbers
$\mathbb{R}^n$	Euclidean n-dimensional space
$\mathbb{R}^{n \times m}$	Space of real matrices with n rows and m columns
$\mathbb{N}$	Set of natural numbers
$\mathbb{Z}$	Set of integers

Acronyms

ARMA	AutoRegressive Moving Average
ARMAX	AutoRegressive Moving Average with eXternal input
BIBO	Bounded Input Bounded Output
EB	Empirical Bayes
FBS	Full Bayes Sampling
FIR	Finite Impulse Response
LTI	Linear Time Invariant
LS	Least Squares
MAP	Maximum A Posteriori
MCMC	Markov Chain Monte Carlo
PEM	Prediction Error Methods
PPEM	Parametric Prediction Error Methods
RPPEM	Recusrsive Parametric Prediction Error Methods
RKHS	Reproducing Kernel Hilbert Space
NPPEM	NonParametric Prediction Error Methods
NPGR	NonParametric Gaussian Regression
SGP	Scaled Gradient Projection
SISO	Single-Input-Single-Output
w.l.o.g.	Without loss of generality
w.r.t.	With respect to

1 Introduction

System identification is concerned with the problem of estimating *dynamical systems* from data measurements.

The systems are the entities that describe every cause and effect reaction. They can be described by suitable mathematical laws called *models*. As a consequence, the models can be seen as the mathematical description of the phenomena we experience in our daily life. The wide spread class of dynamical systems is described by sets of differential equations in the continuous-time case and of difference equations in the discrete-time one.

The standard set-up of a system identification problem involves sets of input data which excite the system under consideration and sets of output data which record the response produced by the system. System identification has the fascinating aim to build models for the underlying system from the observed data.

The art of building models from observed data is treated in several scientific fields

like robotics, machine learning, statistics, data mining, econometrics, neuroscience, biology, and industry to report only a few of them. Back in the 50-th the term "System Identification" has been coined in Zadeh (1956) for the Automatic Control area.

The majority of the approaches proposed to face the system identification problem can be categorized in two groups.

The first one is called *Prediction Error Methods* (PEM). A fundamental property that distinguishes the dynamical systems is the temporal relation to the data, i.e., the future data depends on the past data. Consequently, a natural manner for validating the quality of the system identification procedure is to evaluate the prediction capability of the estimated models. The aim of prediction error methods is to minimize a scalar cost function, depending on the data and on the model, which represents the prediction error.

A classical approach is to provide a specific parametric structure to the models. This turns the system identification problem into the identification of the model parameters. The pioneer works of this approach are Åström and Bohlin (1966); Söderström and Stoica (1989); Ljung (1999).

Recently, a new approach has been introduced in the system identification community which, rather than postulating a parametric model structure, aims at estimating the model in a possible infinite dimensional space. This method is based on the Gaussian regression framework and admits a Bayesian interpretation. While Gaussian regression is a well known technique in the Machine Learning community Rasmussen and Williams (2006), its application to system identification problems has appeared only lately and it represents a fundamental turning point for the community. The pioneer works of this approach are Pillonetto and De Nicolao (2010); Pillonetto, Chiuso, and Nicolao (2011a).

The second main group is called *Subspace methods*. In this paradigm, the models are not obtained by the optimization of a cost function, but the relation between input and output is characterized by the evolution of a state variable in the so called state-space models. See among the others Overschee and Moor (1995); Viberg (1995); Katayama (2006); Qin (2006); Chiuso (2007).

The focus of this dissertation is on the nonparametric Gaussian regression framework, which offers a new effective tool for system identification to tackle the famous bias-variance dilemma. Indeed, while the classical parametric PEM (PPEM) are concerned with the search of the best model structure, which requires a tradeoff between accuracy of the estimate and model complexity, the nonparametric PEM (NPPEM) allow to account for the model complexity directly in the estimation procedure. The purpose of the dissertation is to give new insights on the Gaussian regression technique applied to system identification, to analyze pros and cons of both theoretical and applicative

aspects and to offer extensions to the current state-of-art. The classical PEM are also discussed as a benchmark to refer to. The subspace methods instead are not treated since the comparison is meant only within PEM appraoches.

System identification plays also a fundamental role in robotic applications where accurate models are needed for high performance control design. Indeed, estimation of the inverse dynamics is a challenging problem that finds a direct application in robotic control. The inverse dynamics model can be used as a feedforward term in classical closed loop control schemes, improving the performances in tracking desired trajectories and reducing the gain of the controller. It is known, that parametric models often rely on too restrictive simplification of the physical model to effectively describe the dynamics of the robot. A valuable option is given by the use of semiparametric techniques which could allow to exploit the advantages of both parametric and nonparametric methods.

1.1 Outline

The dissertation aims at providing extensions to the system identification techniques based on Gaussian regression, focusing on theoretical aspects and on a robotic application.

A brief description of each chapter follows.

Chapter 2 presents the prediction error problem for system identification. The problem is mathematically formalized and the two main techniques to face this problem are described, namely the classical parametric prediction error method (PPEM) and the nonparametric prediction error method (NPPEM). The last method is also known as Gaussian regression or Bayesian inference or regularization approach. The chapter concludes highlighting the pros and cons of NPPEM that are discussed in the subsequent chapters.

Chapter 3 performs a comparison between PPEM and NPPEM in terms of both the uncertainty and the accuracy of the estimators. The intrinsic difference between the two approaches leads to a new definition of confidence regions in order to evaluate the uncertainty property. These regions are obtained through sampling techniques and are denoted as "Particle" confidence sets.

The presented results are based on the paper:

Prando G., Romeres D., Pillonetto G., and Chiuso A. Classical vs. bayesian methods for linear system identification: point estimators and confidence sets. In *Proc. of ECC*, 2016a

Chapter 4 introduces an extension of the NPPEM to cope with the problem of the online system identification. In this framework, new data becomes available at each sampling time interval and the estimates need to be updated exploiting the new information before the subsequent data becomes available. Real-time algorithms based on efficient update of the quantities, related to the data, are developed. Particular attention is given to time-varying systems and a comparison with the recursive PPEM is provided. The results of this chapter are based on the papers:

Romeres D., Prando G., Pillonetto G., and Chiuso A. Online bayesian system identification. In *Proc. of ECC*, 2016b

Prando G., Romeres D., and Chiuso A. Online identification of time-varying systems: a bayesian approach. In *Proc. of IEEE CDC*, 2016b

Chapter 5 deals with stability issues that arise when dealing with NPPEM. Indeed, recent works show how these methods can be characterized to estimate naturally stable predictors. Yet stability of predictor models does not guarantee the stability of the simulation models. Several algorithms are proposed to guarantee the stability of both models. The results of this chapter are based on the paper:

Romeres D., Pillonetto G., and Chiuso A. Identification of stable models via nonparametric prediction error methods. In *Control Conference (ECC), 2015 European*, pages 2044–2049. IEEE, 2015

Chapter 6 considers the problem of learning the inverse dynamics of a robotic platform in an online scenario. The semiparametric models, which are a combination of the nonparametric and parametric models, are considered. It is shown how these models can be still considered as a Gaussian process. Their effectiveness is demonstrated in the data collected from the humanoid robot iCub. Moreover, all the inverse dynamics models used in robotics rely on physical quantities that are often not available, namely joint velocities and accelerations. The latter quantities have to be obtained through numerical differentiation from the joint positions measured by the sensor. Since numerical differentiation presents several numerical problems, new derivative free models are proposed. The results of this chapter are based on the paper:

Romeres D., Zorzi M., and Chiuso A. Online semi-parametric learning for inverse dynamics modeling. In *Proc. of IEEE CDC*, 2016c

2

System Identification Overview

This chapter is meant to give the theoretical system identification tools that will be used in the remainder of the dissertation. In particular, two main approaches to tackle the prediction error minimization problem are discussed: the classical parametric approach and the Gaussian regression framework.

The scientific literature about these topics is extremely extended; as such this is only a brief introduction without any aim of completeness. Thus, in this chapter the main concepts necessary to understand the reminder of the dissertation are introduced, providing however to the reader, a general overview on what it has already been done and referring to appropriate literature.

Section 2.1 introduces the problem faced by system identification. In Section 2.2, the prediction error problem is specified, while Section 2.3 and Section 2.4 introduce the two main approaches to tackle this problem, namely, the classical parametric methods and the nonparametric Gassian regression framework. Finally, in Section 2.5 discussions of

the advantages and disadvantages of the nonparametric methods over the parametric ones is carried out.

2.1 System Identification Problem

The core of the system identification process is the estimation of a model from measurement data. This dissertation considers the identification of discrete-time causal dynamical systems: the focus will be on linear and nonlinear, time invariant as well as time-varying systems dependently on the specific problem we will look at. In this section the general system identification problem is formalized using a model that includes all the previously mentioned cases; furthermore, the linear case will be also explicitly discussed due to its relevance for the future chapters.

Let $u(t)$, $y(t) \in \mathbb{R}$ be, respectively, the measured input and output signals of a Single-Input-Single-Output (SISO) dynamical system[1]. Given a finite collection of input-output data points $\mathcal{D}^N := \{\mathcal{D}(t)\}_{t=1}^N = \{u(t), y(t)\}_{t=1}^N$, system identification aims at estimating an accurate (under some criteria) model to describe the phenomenon under consideration.

Consider the time series $\{u(t)\}_{t\in\mathbb{N}}$ and $\{y(t)\}_{t\in\mathbb{N}}$ as jointly stationary zero-mean stochastic process, the *one-step ahead predictor* can be defined in a general form as

$$\hat{y}^*(t|t^-) = f_t^*(y(t^-), u(t^-)) := \mathbb{E}\left[y(t)|y(t^-), u(t^-)\right] \tag{2.1}$$

where the conditional expectations are assumed to be always well defined, t^- is the vector of all the past time instants and f_t^* is the "true" one-step-ahead predictor and the subscript t indicates the time variance of the function f^*.

The definition of a strictly causal predictor, i.e, it does not depend upon $u(t)$, grants the description of the *prediction (or forward) model* in the innovation form

$$y(t) = f_t^*(y(t^-), u(t^-)) + e(t) \tag{2.2}$$

where $\{e(t)\}$ is the *innovation* signal and it is defined as

$$e(t) := y(t) - \hat{y}^*(t|t^-) = y(t) - \mathbb{E}\left[y(t)|y(t^-), u(t^-)\right] \tag{2.3}$$

The innovation sequence $\{e(t)\}$ is, by construction, a martingale difference sequence Hannan and Deistler (1988) w.r.t. the sigma algebra generated by past measurements $y(t^-)$, $u(t^-)$. In this thesis we shall also postulate that $e(t)$ is considered to be a zero

[1] The theory presented in this chapter could be extended to the multi-input multi-output case, but for ease of notation the exposition is restricted to the single-input single-output case.

mean Gaussian process with variance

$$var\left(e(t)\right) = var\left(e(t)|y(t^-),\, u(t^-)\right) = var\left(y(t)|y(t^-),\, u(t^-)\right) = \sigma^2. \tag{2.4}$$

The **System Identification Problem** can be cast as the problem of estimating the one-step-ahead predictor f_t^* from the available input-output data pairs by satisfying a specific criterion: in this dissertation the criterion used is the minimization of the *prediction error* and it will be described in Section 2.2.

"Stability" Assumption

In principle, model (2.2) belongs to an infinite-dimensional space. The main issue that arises working in a infinite dimensional framework is that the problem of finding estimators of the predictor f_t^* from data might be an ill-posed inverse problem Tikhonov and Arsenin (1977).

However, it is commonly considered that in physical systems the effect of a pair of data $(y(s), u(s))$ over $y(t)$ with $t > s$ decreases as $t - s$ goes to infinity. This concept of fading memory is related to the concept of BIBO stability in the predictor impulse responses of linear dynamical systems. In nonlinear systems the concept of stability is more involved and out of the scope of this dissertation, the interest reader is referred to e.g., Chen (2004); Lakshmikantham, Leela, and Martynyuk (1990); Bai, Tempo, Liu, et al. (2007).

For this reasons, it is general practice to estimate a finite-length predictor, which means that f_t^* is assumed to be dependent only on a finite number of previous temporal lags, denoted from here on as n. The length n has to be chosen large enough to capture the dynamics of the system.

Notice that all the results presented in the following could be also formulated in the infinite dimension thanks to recent identification techniques that aim to search for candidate estimators in a suitable Reproducing Kernel Hilbert Space (RKHS) Aronszajn (1950); Saitoh (1988); Kimeldorf and Wahba (1971) where the norms act as a regularizer; "stability" of the predictor (e.g. in linear dynamical systems making sure that the estimated impulse responses are BIBO stable with probability 1), can also be accounted for. See Pillonetto et al. (2011a); Pillonetto and De Nicolao (2010) for the linear case and Pillonetto, Quang, and Chiuso (2011b) for the nonlinear case.

Linear Dynamical Systems

System (2.2) is a general description that contains nonlinear and time-varying models. Clearly, the particular case of linear time invariant (LTI) systems (see e.g. the highly cited books Ljung (1999); Söderström and Stoica (1989)) extensively used throughout this dissertation, is included. In this class of systems, the outputs are described by a linear transformation of the inputs and corrupted by an additive noise, the innovation defined in (2.3). That is:

$$y(t) = \sum_{k=1}^{\infty} g(k)u(t-k) + \sum_{k=1}^{\infty} h(k)e(t-k) \tag{2.5}$$

where $g := \{g(k)\}_{k\in\mathbb{N}}$ is the so called *impulse response*, which is the response of the system when an impulse signal is fed into the systems. Notice that in this general case the additive noise can be seen as the realization of another LTI system with input the innovation $\{e(t)\}_{k\in\mathbb{N}}$ and impulse response $h := \{h(k)\}_{k\in\mathbb{N}}$ which will be called the *error impulse response*. For normalization reasons and w.l.o.g. it is assumed that $h(0) = 1$.

An equivalent representation of (2.5) is given in terms of the transfer functions

$$y(t) = G(z)u(t) + H(z)e(t) \tag{2.6}$$

where

$$G(z) = \sum_{k=1}^{\infty} g(k)z^{-k}, \quad H(z) = \sum_{k=1}^{\infty} h(k)z^{-k} \tag{2.7}$$

Hereafter, it is assumed that $G(z)$ is stable and $H(z)$ is stable and minimum phase, that is, both $H(z)$ and its inverse are causal and stable (i.e. all the poles and zeros of $H(z)$ are inside the unit circle).

The one-step-ahead predictor associated to (2.6) and corresponding to the general description in the linear case of (2.1) is given by

$$\hat{y}(t|t^-) = H^{-1}(z)\left[(H(z)-1)y(t) + G(z)u(t)\right] \tag{2.8}$$

$$= W^y(z)y(t) + W^u(z)u(t) \tag{2.9}$$

see (Ljung, 1999, Chp. 3) for the derivation.

Notice that the predictor transfer functions can also be described as:

$$W^y(z) = \sum_{k=1}^{\infty} w^y(k)z^{-k}, \quad W^u(z) = \sum_{k=1}^{\infty} w^u(k)z^{-k} \tag{2.10}$$

where $w^y := \{w^y(k)\}_{k\in\mathbb{N}}$ and $w^u := \{w^u(k)\}_{k\in\mathbb{N}}$ are the predictor impulse responses.

Since w^y and w^u are BIBO transfer functions, the linear predictor model (2.11) can be approximated in a finite dimensional space as

$$\hat{y}(t|t^-) \approx \sum_{k=1}^{n} w^y(k) z^{-k} y(t) + \sum_{k=1}^{n} w^u(k) z^{-k} u(t) \tag{2.11}$$

which means that the impulse responses $w^y \in \mathbb{R}^n$ and $w^u \in \mathbb{R}^n$ are truncated to the finite space of dimension n. Referring to the previous discussion about the "stability" assumption, this approximation into a finite space is up to an arbitrary small error because the systems considered are BIBO stable (recall that $\{y(t), u(t)\}_{k\in\mathbb{Z}}$ are jointly stationary), which means the coefficients $w^y(k)$ decays to 0 as $k \to \infty$.

A straightforward and useful way to reformulate the predictor model (2.11) is to rewrite the predictor model in a linear regression form

$$Y = \Phi f + E \tag{2.12}$$

where $f := \left[w^{y^\top}, w^{u^\top}\right]^\top$ is the finite dimensional impulse response and

$$\Phi(y(t^-), u(t^-)) = \Phi_N = \Phi := \begin{bmatrix} \phi(1)^\top \\ \vdots \\ \phi(N)^\top \end{bmatrix} \tag{2.13}$$

is the regressor matrix with $\phi(t) = [-y(t-1) \ldots -y(t-n)\, u(t-1) \ldots u(t-n)]^\top \in \mathbb{R}^{2n\times 1}$. Notice that the structure of $\Phi(\cdot)$ is in general dependent on the linear model considered, see Chen, Ohlsson, and Ljung (2012), Pillonetto et al. (2011a) for details. Finally, the output vector, the error vectors and the input vector are defined as

$$Y = Y_N := \begin{bmatrix} y(1) \\ \vdots \\ y(N) \end{bmatrix}, \quad E = E_N := \begin{bmatrix} e(1) \\ \vdots \\ e(N) \end{bmatrix}, \quad U = U_N := \begin{bmatrix} u(1) \\ \vdots \\ u(N) \end{bmatrix} \in \mathbb{R}^N$$

The subscript N will be omitted in the sequel unless specifically needed to avoid confusion.

2.2 Prediction Error Methods

The Prediction Error Methods (PEM) are probably the most widespread approach to identification of dynamical systems. They have been introduced in the system identification community by the seminal paper Åström and Bohlin (1966) which was considering parametric SISO ARMAX models. Nowadays, this approach in the parametric perspective is well established and it has extensively developed both in the Control and in the Statistics community, as it can be demonstrated by the books Ljung (1999); Söderström and Stoica (1989); Hannan and Deistler (1988); Box, Jenkins, Reinsel, and Ljung (2015); Brockwell and Davis (2013).

The interest of this identification approach is on the ability of predicting unobserved data. This focus arises naturally when considering that in dynamical system the future is a function of the past.

In order to solve the system identification problem discussed in Section 2.1 a model class, $\mathcal{M}$, for the predictor f_t^* in (2.2) has to be selected. A model represents a hypothesis we are making on the system. The choice of the model class $\mathcal{M}$ is fundamental and significant difference may arise between parametric and nonparametric approach that are discussed in Sections 2.3 and 2.4.

In the PEM framework, the optimal f_t is found by minimizing an appropriate average loss.

The prediction error problem results to be:

$$\hat{f}_t^{PEM} = \arg\min_{f_t \in \mathcal{M}} \frac{1}{N} \sum_{t=1}^{N} l_t(y(t) - f_t(y(t^-), u(t^-))) \tag{2.14}$$

where $l_t : \mathbb{R} \to \mathbb{R}^+$ is the loss function and $e(t) = y(t) - f_t(y(t^-), u(t^-))$ is the prediction error.

The most commonly used criteria is called *mean squared error* and it is defined by specifying the loss function as:

$$l_t(\cdot) = (y(t) - f_t(y(t^-), u(t^-)))^2 \tag{2.15}$$

The solution of problem (2.14) with loss function (2.15) is called *mean squared predictor*, see for more details (Ljung, 1999, Chp. 3) and (Söderström and Stoica, 1989, Chp. 7).

However, problem (2.14) may be in general ill conditioned making the estimation of the predictor difficult. This ill conditioning might be due to the fact that the predictor f_t^* lives in a finite dimensional space of dimension n and this integer has to be large

enough to include all the dynamical behaviour of the system, which in principle could be even larger than the number of data available N. In the literature, several attempts have been made to tackle this problem, the most relevant either impose a parametric structure to the model class or use a nonparametric approach that can be described in a probabilistic framework (Gaussian regression / Bayesian formulation) or in a deterministic one (regularization). These two approaches, parametric and nonparametric, are outlined in the Sections 2.3 and 2.4.

Maximum Likelihood Approach

The PEM problem is strictly related to the *Maximum Likelihood* approach which aims at maximizing the likelihood function, $p(Y|f_t)$ i.e., the probability density function of the data given the model class.

Consider model (2.2) and the dataset $\mathcal{D}^N$ then the likelihood function is:

$$p_y(Y|f_t) = \prod_{t=1}^{N} p_e(y(t) - f_t(y(t^-), u(t^-))|f_t(y(t^-), u(t^-))) = \prod_{t=1}^{N} p_e(e(t)) \tag{2.16}$$

Notice, that these equivalences are exact only after a sufficient long transient depending on the choice of the initial conditions. However for ease of exposition this error is not treated here, see (Ljung, 1999, Chp. 7) for more details.

The maximum likelihood estimator (MLE) can therefore be written as

$$\begin{aligned} \hat{f}_t^{MLE} &= \arg\max_{f_t} p_y(Y|f_t) \\ &= \arg\max_{f_t} \prod_{t=1}^{N} p_e(y(t) - f_t(y(t^-), u(t^-))) \\ &\equiv \arg\min_{f_t} \frac{1}{N} \sum_{t=1}^{N} -\log p_e(e(t)) \end{aligned} \tag{2.17}$$

which yields the equivalence between the PEM and the MLE estimator when the cost function in (2.14) matches the cost function in (2.17), i.e., $l_t(e(t)) = -\log p_e(e(t))$.

The logarithmic transformation taken in (2.17) is particularly interesting when considering $e(t)$ an independent Gaussian random variable with zero mean and variance σ_e^2. Indeed, the likelihood function becomes a Gaussian $p_y(Y|f_t) \sim \mathcal{N}\left(y|f_t, \sigma_e^2 I_N\right)$ and the MLE corresponds to the mean squared predictor

$$\begin{aligned}\hat{f}_t^{MLE} &= \arg\max_{f_t} p_y(Y|f_t) \\ &= \arg\min_{f_t} \sum_{t=1}^{N} (y(t) - f_t(y(t^-), u(t^-)))^2 + \frac{N}{2}\log\sigma_e^2 \end{aligned} \tag{2.18}$$

2.3 Parametric PEM: the Classical Approach

In the classical parametric identification framework, it is assumed that the system to identify belongs to a specific model class $\mathcal{M}$ characterized through a finite set of parameters $\theta \in \Theta \subset \mathbb{R}^d$, i.e., $\mathcal{M}(\theta)$. Hence, the prediction model (2.2) becomes

$$\mathcal{M}(\theta) : y(t) = f_t(y(t^-), u(t^-), \theta) + e(t) = \hat{y}_\theta(t|t^-) + e(t) \tag{2.19}$$

where $\hat{y}_\theta(t|t^-)$ is the parametrization of the one-step-ahead predictor induced by the choice of the model class $\mathcal{M}(\theta)$ and $e(t)$ is the prediction error.

Given the dataset $\{\mathcal{D}^N\}$ the PEM problem (2.14) turns into the problem of estimating the parameters θ by minimizing a scalar function of the prediction error, that is

$$\hat{\theta}_{PEM} = \arg\min_{\theta\in\Theta} \frac{1}{N}\sum_{t=1}^{N} l_t\left(y(t) - \hat{y}_\theta(t|t^-)\right) \tag{2.20}$$

The choice of the model structure is a crucial step in the identification procedure to effectively tackle problem (2.20) and it requires two main steps:

1. The selection of the "type" of model class which is the a priori information imposed to the model that can be nonlinear, linear, input-output transfer functions, parametrized state-space models, etc. The huge literature on nonlinear modeling is only sketched here to give the reader an idea of the main research directions that have been proposed.

   The Volterra series, which can be seen as an expansion of the system with coefficients, called Volterra kernels given by 'higher-order impulse responses', Volterra (2005); Ikeara (1951); the Wiener-Hammerstain models which are a block-oriented sub case of the Volterra series and exploit the physics of the systems, see the survey Billings (1980) or (Ljung, 1999, Chp. 5); the Neural Networks, which are based on the idea of imitating the network of neurones in the brain, Kubat (1999); Hunt, Sbarbaro, Żbikowski, and Gawthrop (1992); Nelles (2013); Bishop (2006); the family

of NARMAX models Billings (2013) and the Linear Time-Varying modeling which are able to describe nonlinear dynamics by considering the model parameters as time-varying functions of a signal called scheduling variable, Rugh and Shamma (2000); Bachnas, Tóth, Ludlage, and Mesbah (2014).

In this dissertation, the discussion will be restricted to the parametric class of linear dynamical systems considering input-output models; further details on classical parametric PEM will be given within this class in Section 2.3.1. The state space approach mentioned in Chapter 1 is not treated here.

2. The selection of "size" of the model. Once the 'type' of the model class is fixed it is likewise important to choose the dimension d of the parameters θ, or in other words the complexity of the model, using the available data. This step is fundamental because it implies the control on the famous 'bias-variance tradeoff' that will be discussed in Section 2.5. The model order selection step is typically accomplished by estimating models with different complexities and choosing the one with highest performance according to some criteria. The most used criteria are: the information criteria (AIC, BIC, MDL), cross-validation, SURE estimator, bootstrap and C_p. See (Ljung, 1999, Chp. 16) for an extensive overview of these methods and Efron (2004) for the latter method.

2.3.1 Linear Dynamical Systems

Consider the 'true' linear model (2.6); the Parametric PEM (PPEM) approach specifies a parametric model structure depending on the parameters θ, $\mathcal{M}(\theta)$ which describes the transfer functions $G(z)$ and $H(z)$ as $G_\theta(z)$ and $H_\theta(z)$, respectively.

$$y(t) = G_\theta(z)u(t) + H_\theta(z)e(t) \tag{2.21}$$

As mentioned earlier, this parametrization induces a parametrization of the one-step-ahead predictor, $\hat{y}_\theta(t|t^-)$. The most common predictor (and the one that will be used from here on) is the *mean square predictor* which minimizes the variance of the prediction error

$$\hat{\theta}_{PEM} = \arg\min_{\theta\in\Theta} V(\theta, \mathcal{D}^N) = \arg\min_{\theta\in\Theta} \frac{1}{N}\sum_{t=1}^{N} \left(y(t) - \hat{y}_\theta(t|t^-)\right)^2 \tag{2.22}$$

The PEM problem is now translated into the estimation of the parameters $\hat{\theta}_{PEM}$ which can be used to determine the one-step-ahead predictor. Considering the prediction model the mean square predictor can be explicitly formulated as

$$\hat{y}_\theta(t|t^-) = W_\theta^y(z)y(t) + W_\theta^u(z)u(t) \tag{2.23}$$
$$= (1 - H_\theta^{-1}(z))y(t) + H_\theta^{-1}(z)G_\theta(z)u(t) \tag{2.24}$$

see (Ljung, 1999, Sec. 3.2) and (Söderström and Stoica, 1989, Sec. 7.3) for the derivation.

The model class 'type' selection for linear parametric approaches is a well established topic. The first idea that one might have is to set the parameters equal to the coefficients of the numerator and denominator of the transfer functions in model (2.21). Starting from this basic idea several types of models can be formulated, the most famous are: Finite Impulse Response (FIR), Output-Error (OE), AutoRegressive with eXogenous input (ARX), AutoRegressive Moving Average with eXogenous input (ARMAX) and Box Jenkins (BJ). These types of models can be found in any system identification textbook, the reader is referred to (Ljung, 1999, Chp. 4).

Many interesting properties of these estimators are derived using asymptotic arguments, i.e. when $N \to \infty$. For instance, for Gaussian innovations $e(t)$ and for fixed model complexity, these methods have proved to be asymptotically *efficient* and *consistent*. However, model complexity, which strongly affects their effectiveness, has to be estimated from the data. Some of the approaches commonly exploited for this purpose have been already mentioned for the nonlinear case; here are recalled the Information Criteria (AIC/FPE, BIC/MDL, etc.) because are derived from asymptotic arguments. From these considerations a natural question arises: how many data have to be considered for these asymptotic properties to be reliable in a finite-sample domain? The answer is not general and could be really application-dependent. Asymptotic properties will be treated in Section 3.2.

Once $\hat{\theta}_{PEM}$ has been determined, the corresponding predictor impulse response estimate $f_{\hat{\theta}_{PEM}}$ can be computed.

2.3.2 Online Approach

The extension of parametric batch approaches to an online setting relies on Recursive Least Squares (or pseudo LS) methods. In the online setting we assume that at time $t = i$, an estimate of the parameters $\hat{\theta}^{(i)}$, based only on the data up to time i, is available and the estimate has to be updated when new data arrive. The update has to be efficiently computed in order to get the new estimate, $\theta^{(i+1)}$, within the sampling interval.

Suppose that at time $i + 1$ a new input-output data pair $\mathcal{D}(i + 1)$ is provided, two possibilities to compute the estimate can be considered: solving the "complete" problem

(2.22) which may not meet the "real time" computational performance required or update $\hat{\theta}^{(i)}$ using the recursive formula:

$$\hat{\theta}^{(i+1)} = \hat{\theta}^{(i)} - \mu^{(i+1)} Q^{(i+1)^{-1}} \nabla_\theta V_{i+1}(\hat{\theta}^{(i)}, \mathcal{D}(i+1)) \tag{2.25}$$

where $\nabla_\theta V_{i+1}(\hat{\theta}^{(i)}, \mathcal{D}(i+1))$ denotes the gradient of the loss function computed in the previous estimate and in the new data; $\mu^{(i+1)} \in \mathbb{R}$ and $Q^{(i+1)} \in \mathbb{R}^{d \times d}$ are appropriate scalings which depend on the specific algorithm which is adopted (see Ljung and Söderström (1983) and (Ljung, 1999, Chp. 11) for further details). Notice that (2.25) is simply a scaled gradient step w.r.t. the loss function $V_{i+1}(\hat{\theta}^{(i)}, \mathcal{D}(i+1))$.

In order to cope with *time-varying* systems, a possible strategy involves the inclusion of a *forgetting factor* $\bar{\gamma}$ in the loss function:

$$V_N^{\bar{\gamma}}(\theta, \mathcal{D}^N) = \frac{1}{2} \sum_{t=1}^{N} \bar{\gamma}^{N-t} \left(y(t) - \hat{y}_\theta(t|t^-) \right)^2, \qquad \bar{\gamma} \in]0, 1] \tag{2.26}$$

In this way old measurements become less relevant for the computation of the estimate. A recursive update of the estimate $\hat{\theta}^{(i)}$ (as the one in (2.25)) can be derived (Ljung (1999), Ch. 11).

As an alternative, a sliding window approach can be adopted: at each time step only the last N_w data are used for computing the current estimate (with N_w being the window length). However, since this approach does not admit an update rule as the one in (2.25), the computational complexity of the new estimate will depend on the window length. At each time step, a new estimate has to be estimated from scratch, thus significantly slowing down the method. Hence, the estimation of multiple models within the sampling interval has to be computed in order to find the best estimate, this may be computationally expensive, making this procedure possibly not suited for the online identification of time-varying systems.

2.4 Nonparametric PEM: Gaussian Regression

Gaussian regression (GR) has been treated from the scientific community since decades. The initial theory applied to time series appears in the fifties in the book Wiener (1949) and some of the first applications have been in geostatistics under the name 'kriging', Matheron (1973) and in meteorology, Daley (1993). These works were then extended in the books Cressie (2015); Ripley (2005)(originally published in the eighties and nineties). The explicit application of Gaussian process in the regression context can be found in the highly cited works of O'Hagan and Kingman (1978); Sacks, Welch, Mitchell, and Wynn

(1989); Santner, Williams, and Notz (2013). Finally, Gaussian regression for function approximation has been introduced in the machine learning community by Williams and Rasmussen (1996); Rasmussen and Williams (2006). A more detailed historical literature review can be found in Rasmussen and Williams (2006).

The contributions in the machine learning community inspired the recent development in system identification: in the latest years new nonparametric techniques to face the PEM estimation problem (2.14) have been defined. These methods have attracted considerable attention because they go beyond the classical PPEM described in Section 2.3. In particular, the candidate models are searched for in infinite dimensional model classes, thus avoiding to perform the delicate 'order selection' step needed in parametric methods. Needless to say, this is not entirely free of difficulties, since an alternative way to control the model complexity, i.e., to face the so called bias-variance tradeoff, needs to be found. It has been shown in the recent literature Pillonetto and De Nicolao (2010); Pillonetto et al. (2011a); Chen et al. (2012) that the apparatus of Reproducing Kernel Hilbert Spaces (RHKS) or, equivalently, Bayesian Statistics provide powerful tools to face this trade-off, see also Chiuso (2016); Pillonetto, Dinuzzo, Chen, Nicolao, and Ljung (2014).

The multiple connections of the Gaussian regression framework applied to the system identification PEM problem created several ways to name these techniques. First, the model "type" selection described for classical parametric approach is no longer needed because *NonParametric Prediction Error Methods* (NPPEM) are based on a mathematical tools that does not impose a parametric structure, from which the adjective 'nonparametric'. Second, the connection between Gaussian regression and the regularized kernel methods applied to function estimation, see e.g. Wahba (1990); Rasmussen and Williams (2006), arises the name of *Regularized methods*. Third, the recent works Pillonetto and De Nicolao (2010); Pillonetto et al. (2011a); Chen et al. (2012) enlightened how Gaussian regression applied to system identification relies on Bayesian inference, thus the name *Bayesian methods*.

Consider now the general problem statement considered in Section 2.1, when the NPPEM are adopted the predictor f_t^* in the simulation model (2.2) is interpreted as the realization of a Gaussian random field, see e.g. (Rasmussen and Williams, 2006, Chp. 2) denoted by f_t. In the spirit of the Bayesian philosophy the aim of this identification procedure is to estimate the posterior distribution of f_t given the available data $\mathcal{D}^N$, $p(f_t|\mathcal{D}^N)$. From hereafter, the dependency on the inputs U will be omitted, consequently the posterior distribution is defined as $p(f_t|Y)$.

The a priori probability distribution given to f_t in Bayesian terminology is called

prior

$$f_t \sim p_\eta(f_t) \tag{2.27}$$

and in general depends upon some unknown parameters $\eta \in \Omega \subset \mathbb{R}^{d_\eta}$, called *hyperparameters*, which need to be estimated from data. The subscript η denotes that the probability distribution of f_t is a function of the hyperparameters.

In the Gaussian regression framework $p_\eta(f_t)$ is assumed to be Gaussian, implying that f_t can be defined through only its second order statistics, i.e., its mean and covariance. That is:

$$f_t \sim \mathcal{N}(\mu(\cdot), k_\eta(\cdot,\cdot)) \tag{2.28}$$

where $k_\eta(\cdot,\cdot)$ is the covariance function defined as

$$k_\eta(\mathcal{D}(t), \mathcal{D}(s)) := cov(f_t(\mathcal{D}(t)), f_t(\mathcal{D}(s))) \tag{2.29}$$

The corresponding covariance matrix $K_\eta(\mathcal{D}^N, \mathcal{D}^N) \in \mathbb{R}^{N\times N}$ is defined as the matrix of the covariances evaluated at all the pairs of $(\mathcal{D}(t), \mathcal{D}(s))$ points with $t, s = [1, N]$. This matrix is called *kernel matrix* in the Machine Learning community, see (Rasmussen and Williams, 2006; Scholkopf and Smola, 2001). The kernel matrix is a symmetric positive semi-definite matrix:

$$K_\eta(\mathcal{D}(t), \mathcal{D}(s)) = K_\eta^\top(\mathcal{D}(t), \mathcal{D}(s)) \geq 0$$

If not differently specified we rely on the common assumption that the prior mean in (2.28) is set to 0, i.e., $\mu(\cdot) = 0$

The choice of the structure of K_η and the estimation of η are crucial points because the quality and the features of the final estimate are encoded in this choice. Loosely speaking, this choice can be seen as the counterpart of the model selection in the PPEM procedure.

According to the Bayesian paradigm, once a prior on the predictor has been selected and a dataset, $\mathcal{D}^N$, is available the *posterior* of the predictor is defined as following

$$p_\eta(f_t|\mathcal{D}^N) = \frac{p(\mathcal{D}^N|f_t)p_\eta(f_t)}{p_\eta(\mathcal{D}^N)} \tag{2.30}$$

where $p(\mathcal{D}^N|f_t)$ and of $p_\eta(\mathcal{D}^N)$ are the *likelihood* and *marginal likelihood* function, respectively. In order to simplify the notation, in what follows, the symbol $\mathcal{D}^N$ will be replaced in the notation with Y; therefore, we shall use $p_\eta(Y)$ and $p_\eta(f_t|Y)$ in lieu of $p_\eta(\mathcal{D}^N)$ and $p_\eta(f_t|\mathcal{D}^N)$, respectively.

The marginal likelihood is defined as:

$$p_\eta(Y) = \int p(Y|f_t)p_\eta(f_t)df_t \tag{2.31}$$

This quantity is a fundamental tool that Bayesian inference provides to estimate the hyperparameters, as it will be seen in Section 2.4.3.

Given the posterior distribution, the predictor *minimum variance estimate*, (Pillonetto and De Nicolao, 2010; Pillonetto et al., 2011a,b) is given by:

$$\begin{aligned}\hat{f}_t &= \int f_t p_\eta(f_t|Y)df_t \\ &= \int\int f_t p(f_t|Y,\eta)p(\eta|Y)df_t d\eta \\ &= \int \mathbb{E}\left[f_t|Y,\eta\right] p(\eta|Y)d\eta\end{aligned} \tag{2.32}$$

where $\mathbb{E}\left[f_t|Y,\eta\right]$ is the conditional estimate of f_t when η are fixed.

Unfortunately, in a general framework these integrals are not analytically tractable and it is necessary to resort to effective approximations of the posterior, e.g. stochastic techniques as Markov Chain Monte Carlo (MCMC) methods or analytical approximations. These approximations lead to different approaches, such as the Full Bayesian Sampling (FBS) and the so-called Empirical Bayes (EB) estimators, described in Section 2.4.1.

2.4.1 Posterior Approximation

In this section, two fundamental approximations of the posterior in Bayesian inference are presented.

Considering the Bayesian hierarchical model described in Section 2.4, the problem of determining the posterior distribution can be formulated as the problem of determining the hyperparameters of the prior. Consequently, an approximation of the posterior distribution corresponds to an approximation of the hyperparameters distribution.

In terms of solutions to the PEM problem, the approximations on the hyperparameters lead to different approximations of the minimum variance estimate defined in (2.32) that will be named FBS and EB, accordingly to the paradigm used. In practice, what we are wondering is:

"How can we approximate the integral $\int p(f_t|Y,\eta)p(\eta|Y)d\eta$ in order to compute the Bayesian minimum variance estimate?"

Full Bayes Sampling

The Fully Bayesian paradigm is in principle interested in determining the whole posterior distribution. As mentioned earlier, the posterior in general cannot be computed and one possible approximation relies on a sampling approximation technique; from which the name Full Bayes Sampling (FBS).

In this context, η is considered as a random vector with distribution $p(\eta|Y)$ and the posterior is approximated as follows

$$p_\eta(f_t|Y) = \int p(f_t|Y,\eta)p(\eta|Y)d\eta \approx \sum_{i=1}^{T} p(f_t|Y,\eta^{(i)}) \tag{2.33}$$

where $\eta^{(i)}, i = [1,T]$ are samples from the distribution $p(\eta|Y)$. How to perform the sampling is a user's choice. One possibility is to adopt he stochastic simulation technique MCMC Gilks, Richardson, and Spiegelhalter (1995); Andrieu, Doucet, and Holenstein (2010); Ninness and Henriksen (2010) which is a well known techniques to efficiently sample from unknown distributions.

From this approximation, the minimum variance estimate of f_t can be computed. Notice that one interesting property of this estimator is that is takes into account also the variability of the hyperparameters, for further discussion see Magni, Bellazzi, and Nicolao (1998); Pillonetto and Bell (2007) and the results of Chapter 3.

In Chapters 3 and 5 two implementations of the FBS approximation will be formulated.

Empirical Bayes

A common choice of the Empirical Bayes (EB) approach (Robbins, 1958) is based on the idea that, the parameters (or hyperparameters), at the highest level of the Bayesian hierarchical model, are fixed to a value estimated from the data, e.g., maximizing the (marginal) likelihood.

In the Bayesian framework described in Section 2.4 the Empirical Bayes estimate we consider is based on the assumption that the marginal on the hyperparameters $p(\eta|Y)$ is approximated by a delta-function centred at its mode $\hat{\eta}$; under this approximation the outer integral in (2.32) is trivially equal to $\mathbb{E}[f_t|Y,\eta]$ evaluated at $\hat{\eta}$. According to this distribution only one set of hyperparameters has to be estimated and it will be denoted by $\hat{\eta}$.

Under the assumption that the error affecting the output data is an additive independent identically distributed zero mean Gaussian noise with covariance σ^2 and the hyperparameters are fixed to a certain value, $\hat{\eta}$, the vector composed by $[f_t(\mathcal{D}^N), y(1), \ldots, y(N)]$

is jointly Gaussian. This is true for any point in the domain space of $\mathcal{D}^N$, including also the ones not available in the dataset $\mathcal{D}^N$. In practical applications this consideration is of interest when one wants to make prediction on a point in the input space not seen yet. Therefore, considering a point $\mathcal{D}^* := (y(t), u(t)) \in \mathbb{R}^2$ with the joint distribution between the predictor and the available data is defined as

$$\begin{bmatrix} f_t(\mathcal{D}^*) \\ Y \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} 0_N \\ 0_N \end{bmatrix}, \begin{bmatrix} K_{\hat{\eta}}(\mathcal{D}^*, \mathcal{D}^*) & K_{\hat{\eta}}(\mathcal{D}^*, \mathcal{D}^N) \\ K_{\hat{\eta}}(\mathcal{D}^N, \mathcal{D}^*) & \hat{K}_{\hat{\eta}}(\mathcal{D}^N, \mathcal{D}^N) + \sigma^2 I_N \end{bmatrix} \right) \tag{2.34}$$

Consequently, from basic calculus on conditioning jointly Gaussian random variable, the posterior distribution $p_{\hat{\eta}}(f_t|Y)$ is Gaussian

$$p_{\hat{\eta}}(f_t|Y) = \mathcal{N}\left(f_t|\mu^{post}, \Sigma^{post}\right) \tag{2.35}$$

where

$$\mu^{post} = K_{\hat{\eta}}(\mathcal{D}^*, \mathcal{D}^N)\left(K_{\hat{\eta}}(\mathcal{D}^N, \mathcal{D}^N) + \sigma^2 I_N\right)^{-1} Y \tag{2.36}$$

$$\Sigma^{post} = K_{\hat{\eta}}(\mathcal{D}^*, \mathcal{D}^*) - K_{\hat{\eta}}(\mathcal{D}^*, \mathcal{D}^N)\left(K_{\eta}(\mathcal{D}^N, \mathcal{D}^N) + \sigma^2 I_N\right)^{-1} K_{\hat{\eta}}(\mathcal{D}^N, \mathcal{D}^*) \tag{2.37}$$

Finally, the EB estimator (see Robbins (1958); Carlin and Louis (1997); Pillonetto and Chiuso (2015)) of f_t coincides with the posterior mean and it can be written as

$$\hat{f}_t^{EB} := \mathbb{E}\left[f_t|Y, \hat{\eta}\right] = \mu^{post} \tag{2.38}$$

The posterior mean obtained in this setting is also called the *maximum a posteriori* (MAP) estimator of f_t. It follows that the minimum variance estimator and the MAP estimators coincide.

In machine learning this approach is also called Gaussian regression functional space view, see (Rasmussen and Williams, 2006, Chp. 2).

Remark 2.4.1. The noise variance σ^2 can be treated as a hyper-parameter or estimated by solving a low-bias LS estimate of f_t^*. Anyhow, it is unknown and it has to be estimated in order to compute an estimate of f_t^*.

2.4.2 Connection with Regularized PEM

There is an interesting link between the probabilistic Bayesian inference and the deterministic framework of *regularized* problems.

The minimum variance estimate in (2.38) coincides with the MAP estimator, consequently it can be formulated as

$$\hat{f}_t^{MAP} = \mu^{post} = \arg\max_{f_t} p_\eta(f_t|Y) \tag{2.39}$$

$$= \arg\max_{f_t} p_\eta(Y|f_t)p_\eta(f_t) \tag{2.40}$$

$$\equiv \arg\min_{f_t} -\log p_\eta(Y|f_t) - \log p_\eta(f_t) \tag{2.41}$$

$$\equiv \arg\min_{f_t} \frac{1}{\sigma^2}\|Y - f_t\|^2 - \log p_\eta(f_t) \tag{2.42}$$

The first fundamental observation that this formulation highlights is that Gaussian regression aims to solve a prediction error problem. Indeed, the first term in expression (2.42) coincides exactly with the mean square loss function in (2.15); the estimate $\hat{f}_t^{MAP}$ is an approximation of the mean squared predictor.

As discussed earlier, problem (2.14) is an ill-conditioned problem and expression (2.42) suggests that one way to see how Gaussian regression tackle this issue is to add a regularization term that is given by the prior knowledge assumed on the system.

Under the assumption of a Gaussian prior distribution (2.28) the regularization problem (2.42) becomes

$$\hat{f}_t^{REG} = \hat{f}_t^{MAP} = \arg\min_{f_t} \|Y - f_t\|^2 + \sigma^2 f_t^\top K_{\hat{\eta}}^{-1}(\mathcal{D}^N, \mathcal{D}^N) f_t \tag{2.43}$$

where $\hat{\eta}$ indicates that η is fixed to a certain value, typically estimated from the data.

Expression (2.43) is now a l_2-type regularized problem. The solution to the prediction error problem is now restricted to a set of estimates such that the term $f_t^\top K_\eta^{-1} f_t$ is "small". This underlines the importance of the choice of the prior: the kernel matrix gives the directions where the solution has to be searched for in the space and reduce the ill-conditioning of the problem.

The regularized inverse problem (2.43) is also known as Tikhonov-regularization Tikhonov and Arsenin (1977).

For future use, it is also pointed out that the regularized (or MAP, or minimum variance) estimate coincides with

$$\hat{f}_t^{REG} = \sum_{i=1}^{N} c_i K_{\hat{\eta}}(\mathcal{D}(t), \mathcal{D}(i)) \tag{2.44}$$

where c_i is the i-th component of the vector

$$c = \left(K_{\hat{\eta}}(\mathcal{D}^N, \mathcal{D}^N) + \sigma^2 I_N\right)^{-1} Y \in \mathbb{R}^N \tag{2.45}$$

This result follows from the representer theorem, see Kimeldorf and Wahba (1971); Wahba (1990).

2.4.3 Hyperparameters Tuning

The last question we need to answer is: 'How can we estimate the hyperparameters η from the data ?'

Marginal Likelihood

The Bayesian framework offers directly a tool to estimate the hyperparameters. Indeed, the marginal likelihood, $p_\eta(Y)$, defined in (2.31) expresses the likelihood of the hyperparameters given the data, once the unknown model component f_t has been integrated out.

Under the assumption that f_t and the innovation are Gaussian and independent (see Section 2.1 for more details), the marginal density can be computed in closed form, as discussed in Pillonetto and De Nicolao (2010); Pillonetto et al. (2011a) for linear systems and generalized in Pillonetto et al. (2011b) to the nonlinear case, and is given by

$$p_\eta(Y) = \exp\left(-\frac{1}{2}\log(\det[2\pi\Sigma_Y(\eta)]) - \frac{1}{2}Y^\top \left(\Sigma_Y(\eta)\right)^{-1} Y\right) \tag{2.46}$$

where

$$\Sigma_Y(\eta) := K_\eta(\mathcal{D}^N, \mathcal{D}^N) + \sigma^2 I_N \tag{2.47}$$

is the prior covariance on the noisy observations and $\sigma^2 := Var\{e(t)\}$ is the variance of the innovation process.

The hyperparameters vector η is estimated by minimizing the negative log marginal likelihood:

$$\begin{aligned}
\hat{\eta}_{ML} &= \arg\max_\eta p_\eta(Y) \\
&\equiv \arg\min_\eta -\log p_\eta(Y) \\
&\equiv \arg\min_\eta \log(\det[2\pi\Sigma_Y(\eta)]) + Y^\top \left(\Sigma_Y(\eta)\right)^{-1} Y
\end{aligned} \tag{2.48}$$

Notice that the optimization of the marginal likelihood is equivalent to the maximization on the posterior of the hyperparameters given the data $p(\eta|Y)$ once a non-informative prior on the hyperparameters is considered, i.e., all the sets of hyperparameters have a uniform distribution.

The robustness of this approach has been discussed in Aravkin, Burke, Chiuso, and Pillonetto (2012); Carli, Chen, Chiuso, Ljung, and Pillonetto (2012); Pillonetto and Chiuso (2015).

Cross Validation

In a deterministic framework, such as the regularization, an alternative technique to estimate the hyperparameters is known as *Cross Validation* (CV). The dataset used for the identification is split into two data sets: the training set and the validation set. The goal is to estimate the set of hyperparameters having the best performance on the prediction of unseen data, accordingly to some criteria. The training set is used to estimate the model for different values of the hyperparameters and the validation set is used to verify the prediction capability. This kind of validation is called hold-out validation (James, Witten, Hastie, and Tibshirani, 2013, Chapter 6).

This is probably the simplest version of CV, an extension could consider the splitting of the dataset into several subsets. Each subset is considered in turn the training set and the others as validation sets. The prediction capability obtained in the validation sets are somehow averaged in all the cases to decide the best model. One common technique of this kind is called *k-fold*.

These are only few of the possible variants of CV, some others are e.g., PRESS and GCV. Moreover, in the deterministic framework there can be also other techniques as the C_p statistics or the SURE estimator. See (Pillonetto et al., 2014, Sec. 14) for an overiview.

2.4.4 Linear Dynamical Systems

Consider the linear model (2.12) and assume the noise signal e is a white independent Gaussian noise, then all the theory and results obtained for NPPEM for the more general problem (2.2) still hold. Nevertheless, it is of interest to report explicitly the main results and formula in the class of LTI systems, due to the wide usage of linear systems in the literature and also in the remainder of this dissertation.

$$Y = \Phi f + E \tag{2.49}$$

where $f := \left[w^{y^\top}, w^{u^\top}\right]^\top$ is the infinite dimensional impulse response and Φ is defined as in (2.13).

The impulse response f is modelled as a zero mean Gaussian process Rasmussen and Williams (2006) with covariance given by the Kernel matrix K_η

$$p_\eta(f) \sim \mathcal{N}(f|0, K_\eta) \tag{2.50}$$

Under the assumption that the innovation is Gaussian and independent of f and because of the linearity, then of Y and f are jointly Gaussian yielding also a Gaussian posterior for a fixed η:

$$p_\eta(f|Y) \sim \mathcal{N}(\mu_f^{post}(\eta), \Sigma_f^{post}(\eta)) \tag{2.51}$$
$$\mu_f^{post}(\eta) = \mathbb{E}[f|Y, \eta] := K_\eta \Phi^\top (\Phi K_\eta \Phi^\top + \sigma^2 I_n)^{-1} Y \tag{2.52}$$
$$\Sigma_f^{post}(\eta) := K_\eta - K_\eta \Phi^\top (\Sigma_Y(\eta))^{-1} \Phi K_\eta \tag{2.53}$$

where the a priori covariance of the data is

$$\Sigma_Y(\eta) = \Phi K_\eta \Phi^\top + \sigma^2 I_N \tag{2.54}$$

and $\sigma^2 := Var\{e(t)\}$ is the variance of the innovation process.

Hence, the minimum variance estimate $\hat{f}^{EB}$ can be computed in closed form using (2.52)

$$\hat{f}^{EB} = \mu_f^{post}(\hat{\eta}) = K_{\hat{\eta}} \Phi^\top (\Sigma_Y(\eta))^{-1} Y \tag{2.55}$$

In this framework also the marginal $p_\eta(Y)$ can be computed as in (2.46) and its closed form results to be:

$$p_\eta(Y) = \exp\{-0.5(\ln|2\pi\Sigma_Y(\eta)| + Y^\top (\Sigma_Y(\eta))^{-1} Y)\} \tag{2.56}$$

In machine learning this approach is called the Gaussian regression weight-space view (Rasmussen and Williams, 2006, Chp. 2).

As pointed out in Section 2.4.2 the minimum variance Empirical Bayes estimate can be interpreted in the regularization framework, which results in:

$$\hat{f}^{REG} = \hat{f}^{EB} = := \mathbb{E}\left[f|Y,\hat{\eta}\right] \tag{2.57}$$

$$= \arg\min_{f\in\mathbb{R}^n} \left(Y-\Phi\mathbf{f}\right)^\top \left(Y-\Phi f\right) + \sigma^2 f^\top K_{\hat{\eta}}^{-1} f \tag{2.58}$$

$$= \left(\Phi^\top\Phi + \sigma^2 K_{\hat{\eta}}^{-1}\right)^{-1} \Phi^\top Y \tag{2.59}$$

2.4.5 Online Approach

On the contrary to the parametric approach, there is not a standard online procedure for NPEM. In the following, an overview of the main works present in literature is reported.

Unfortunately, Gaussian regression requires computations that scale with $O(N^3)$ for training, as can be seen from equations (2.36)-(2.38). In order to reduce the computational complexity, several sparse approximations have been proposed in the recent years (see for example Lawrence, Seeger, and Herbrich (2002); Smola and Bartlett (2001); Snelson and Ghahramani (2006); Tresp (2000); Williams and Seeger (2001); Ranganathan, Yang, and Ho (2011); Csató and Opper (2002). The main idea of these approximations is to select only a fixed limited number of data based on some criteria, consequently the computational complexity can be arbitrarily reduced. However, most of these approximations operate in a batch mode, assuming that all the data are available and performing the computation offline.

Based on similar ideas also online approaches have been proposed in the literature. We mention the nonparametric algorithm selecting a sparse subset of training data points (i.e. dictionary), Nguyen-Tuong and Peters (2011a) and the local Gaussian process regression approach proposed in Nguyen-Tuong, Seeger, and Peters (2009). In Gijsberts and Metta (2011) the complexity is kept constant approximating the kernel function using so called "random features", Rahimi and Recht (2007); Quinonero-Candela and Rasmussen (2005).

Only few approaches propose methods to sequentially update data. In Gilks et al. (1995) the new available data are clustered in a sequential manner that leads to the final estimate. However, the number of clusters and the number of data of clusters have to be carefully tuning accordingly to the application. In Hartikainen and Särkkä (2010) and in De Nicolao, Ferrari-Trecate, and Lecchini (1998), GR is seen as a Kalman filtering that scales with $O(N)$ for specific choice of the kernel. In Hartikainen and Särkkä (2010) and Huber (2014) a method called Recursive Gaussian Process (RGP) is proposed. Gaussian regression is seen as a Bayesian filtering problem, where the regression function is represented by means of a finite set of basis vectors. The update of the estimate can be computed recursively thanks to this fixed number of basis vectors. This method has been

applied to system identification of nonlinear functions in Prüher and Simandl (2014).

2.5 Motivations: Advantages and Disadvantages of Gaussian Regression

The purpose of this section is to discuss some of the main issues that have to be faced when dealing with system identification and that have inspired this manuscript.

The discussion focuses on the points where we believe that the Gaussian regression framework could effectively outperform the parametric approach or where we encountered limitations in this nonparametric approach; these points are the motivations of the work in the remainder of this dissertation.

Bias-Variance Tradeoff

Selecting the model complexity, e.g. trading bias versus variance is an important aspect which makes the identification of a system given a finite number of data still an open issue.

Even in the "easy" linear system identification, which is sometimes considered to be a mature field (in particular for PPEM which are by now well developed and understood, see e.g. Ljung (1999); Söderström and Stoica (1989); Pintelon and Schoukens (2012)) facing in an effective manner the bias-variance dilemma trading model complexity vs. data fit is still a challenge. The recent regularization methods for system identification are offering new effective tools to tackle this issue, see e.g. Pillonetto and De Nicolao (2010); Banbura, Giannone, and Reichlin (2010); Pillonetto et al. (2011a); Chen et al. (2012); Pillonetto et al. (2014); Chiuso (2016); Rasmussen and Williams (2006).

The bias-variance dilemma takes root in the fact that the residuals in the training data are not a good measure of the estimate capability in predicting unseen data. In the interest of only the training data the estimator would tend to interpolate the points creating an *overfitting* effect and likely an *underfitting* in the test data.

Assume the data are generated from a model as (2.2), the error is a zero mean independent Gaussian noise and we are provided with an estimate of the predictor $\hat{f}(\mathcal{D}^0)$, at the point $\mathcal{D}^0 := (y(t), u(t))$ then the *mean squared error* can be decomposed as:

$$MSE = \mathbb{E}\left[\left(f_t^*(\mathcal{D}^0) - \hat{f}(\mathcal{D}^0)\right)^2\right] \tag{2.60}$$

$$= \left(f_t^*(\mathcal{D}^0) - \mathbb{E}\left[\hat{f}(\mathcal{D}^0)\right]\right)^2 + \mathbb{E}\left[\left(\hat{f}(\mathcal{D}^0) - \mathbb{E}\left[\hat{f}(\mathcal{D}^0)\right]\right)^2\right] \tag{2.61}$$

$$= \text{bias}^2 + \text{variance}$$

where (2.60) follows from the independence of the noise and the model and (2.61) because the true model $f_t^*(\mathcal{D}^0)$ is a deterministic quantity.

The final expression of the MSE (2.61) is composed of the sum of two quantities: the square of the bias term, which is the difference between the true model and the expectation of the estimate $\hat{f}(\mathcal{D}^0)$ w.r.t. the randomness in the training data and the variance of the estimate $\hat{f}(\mathcal{D}^0)$.

These two terms can be controlled through the estimation procedure and it is well known that as the complexity of the estimator increases the bias term tends to decrease while the variance tends to increase. How to trade between these two quantities should be ideally based on the minimization of the MSE on the test data, which is clearly unavailable. Unfortunately, the training error is a bad estimator of the MSE, indeed, while the former decreases with the increase of the model complexity the latter has been shown to have its minimum (as a function of the complexity) in a 'middle' point between low and high complexity, see e.g., (Hastie, Tibshirani, and Friedman, 2008, Chp. 2 and 7)

In the parametric approach, these considerations have a clear explanation once the model "type" has been selected: on the one hand, a complex model (i.e. with a 'high' number of parameters) guarantees an accurate adherence to the training data, on the other hand, a simple model (i.e. with a 'low' number of parameters) results to be more flexible in describing the unseen data. In the nonparametric approach, after the selection of the Kernel structure (that somehow corresponds to the choice of the parametric model 'type') the model complexity is regulated by the value of the hyperparameters. This way of controlling the model complexity has been experimented to bring advantages in some situations w.r.t. the parametric approach. Moreover, the NPPEM presented in Section 2.4 have typically a smaller number of hyperparameters w.r.t. the number of parameters of PPEM leading to optimization problems in a smaller dimension space.

A further insight on the possible superiority of NPPEM methods can be seen in linear models when introducing the concept of *error model*, see e.g., Goodwin, Gevers, and Ninness (1992). The error model is the displacement between the assumed class of models and the true system. In machine learning this concept appears e.g. in (Hastie

et al., 2008, Chp. 7) where the bias term in equation (2.61) is split as:

$$\text{bias}^2 = \text{bias}^2_{\text{model}} + \text{bias}^2_{\text{estimate}} \tag{2.62}$$

with $\text{bias}^2_{\text{model}}$ a function of the error model and $\text{bias}^2_{\text{estimate}}$ a function of the error returned by the estimation procedure.

In the linear NPPEM, it has been shown that when kernels of the Stable Spline family are considered Pillonetto and De Nicolao (2010); Pillonetto et al. (2011a), the error model goes to zero because the class of functions generated from these kernels is sufficiently rich to describe all the possible stable impulse responses. Therefore, the tradeoff is made between the $\text{bias}_{\text{estimate}}$ and the variance while keeping null the $\text{bias}_{\text{model}}$. In the PPEM this is in general not true. Indeed, a zero error model is obtained only in the case that the selected model "type" coincides with the true system. Notice that choosing always the most general class cannot be a solution because this increases the model complexity.

Model complexity selection is actually one of the crucial steps in any parametric procedure: experimental evidence has shown that parametric approaches may give rather unreliable results when model complexity is not fixed but has rather to be determined from data. Furthermore, many useful properties concerning the parameter estimators are derived under asymptotic conditions, i.e. assuming to deal with infinite data lengths. For instance, most criteria for determining model complexity are based on asymptotic arguments.

All the previous considerations are still open issues in the system identification community. Chapter 3 will be dedicated to an extensive comparison between PPEM and NPPEM, with the final goal of validating some of the points previously discussed.

Moreover, the criticality of the model selection complexity emerges especially in the *online* framework, where new data become available as the time goes. Indeed, model complexity has to be modified in response to the changes of the system dynamics. In general, dealing with parametric model classes in which the order changes over time is a non-trivial issue, for which no clear guide-lines exist. In addition, classical complexity selection rules may not be applicable in online settings, because of the excessive computational effort they require. The recursive methods outlined in Section 2.3.2 do not take into account the possibility of changes in the dimension of the parameters.

In Section 2.4.5 it has been discussed that NPPEM are for their nature less suitable to recursive procedures, therefore they also present issues for applications in online settings. However, in this framework model complexity is tuned in a continuous manner by estimating the hyperparameters which describe the prior distribution and consequently the size of the class of models. This property joint with a typical reduced dimensionality

of the domain of the hyperparameters makes the NPPEM particularly appealing for the online identification of time-varying systems. This extension of NPPEM is discussed in Chapter 4.

System Stability

It has been shown, that for linear dynamical systems NPPEM can guarantee the identification of stable predictors based on an appropriate choice of the prior, see Pillonetto and De Nicolao (2010); Pillonetto et al. (2011a). Unluckily, the stability of the predictors does not guarantee the stability of the impulse responses of the forward model, as it can be seen observing the relation between the transfer functions of the two models in (2.23).

In PPEM approaches this issue is solved imposing the stability requirements of both forward and predictor model in the a priori structure of the parameters.

In control theory terminology we are dealing with the problem of having a stable closed loop system and a possible unstable open loop plant. This problem can affect scientific area also outside the control community since in some applications working with the open loop plant can be more of a interest than the closed loop system. Some preliminary and successful techniques are discussed in Chapter 5.

Local and Global Models

The previous discussion has highlighted how NPPEM effectively faces the bias-variance tradeoff, exploiting data-driven procedures which allow to tune the model complexity in a continuous manner. Yet the price to be paid is that, typically, nonparametric models for nonlinear systems can only provide good local approximation, in the neighbourhood of input locations visited by the training data. Their prediction performance deteriorates significantly when tested on input locations which are far (in a suitable metric) from those visited in the training phase. On the other hand, parametric models can be based on physical considerations thus providing, in principle, global approximation properties. The advantages of both methods can be successfully combined, to some extent, in semiparamteric models. In Chapter 6 these models are analyzed and extended in a robotic application.

3

Gaussian Regression and Parametric PEM: a Comparison

A model can never be the perfect description of a real system. Consequently, the evaluation of the quality of the estimate obtained by an identification procedure is fundamental. In linear dynamical systems, when the knowledge of the real system is available, e.g. in simulative experiments, a natural performance index to test an identification algorithm is given by the fit of the impulse response estimate to the real one. Nevertheless, the information given by this index can be misleading if not combined with the precise knowledge of the uncertainty bounds around the impulse response.

This chapter focuses on the comparison between the linear PPEM, defined in section 2.3.1 and the linear Gaussian regression defined in section 2.4.4. In particular, the focus will be on the uncertainty sets which can be determined under the two approaches. The comparison between the confidence intervals derived under a frequentist framework and the credible intervals defined under a Bayesian paradigm is a widely discussed topic (see

e.g. Jaynes and Kempthorne (1976); Efron (2005)); the comparison carried out in our contribution will be restricted to the system identification framework.

In PPEM, the asymptotic theory has been widely exploited to derive the statistical properties of the estimate and therefore to construct a confidence region around it, see (Ljung, 1999, Chp 9) among the others. However, in practice the amount of available data is limited: assessing the reliability of these confidence regions under finite sample situations is of crucial importance and has been discussed for instance in Goodwin et al. (1992); Weyer, Williamson, and Mareels (1999); Campi and Weyer (2002); Garatti, Campi, and Bittanti (2004). Some authors have also explored the possibility to define non-asymptotic confidence regions for parametric system identification procedures (see e.g. Campi and Weyer (2005); Csáji, Campi, and Weyer (2015)).

It should be also recalled that the asymptotic derivation of confidence regions assume that the model class is fixed to the correct one, while in practice this is estimated from the available data, making the PPEM estimator a Post Model Selection Estimator (PMSE): Leeb and Potscher (2005) have pointed out how the asymptotic analysis becomes rather delicate in this case.

On the other hand, NPPEM relying on the Bayesian inference perform an implicit model selection step, thus not requiring the user to explicitly select the complexity of the model to be estimated. Furthermore, under the Bayesian framework, confidence regions can be directly derived from the posterior distribution, without relying on the asymptotic theory. Of course, the quality of these confidence sets directly depends on the goodness of the chosen prior.

Contribution

1. The PPEM and NPPEM are based on different paradigms that lead to the formulation of the solution to the prediction error minimization problem either in the space of the parameters or in the space of the impulse responses. We believe that the latter space is the correct one to perform the comparison and we define and evaluate the quality of confidence sets in it. In order to include all the methods we propose and define "particle" confidence sets, i.e., based on sampling procedures.

   The evaluation of the quality of the two PEM methods is carried out considering also the precision of the point estimators.

2. In addition to the comparison between parametric and nonparametric approaches we are also interested in other two analysis.

First, focusing on the NPPEM the effectiveness of the Empirical Bayes and of the Full Bayes Sampling paradigms is evaluated.

Second, focusing on the PPEM, the quality of the confidence sets relying on the asymptotic distribution and on the likelihood function are counterposed.

The chapter is organized as follows. Section 3.1 reports the problem statement. In Section 3.2 a brief summary of the confidence sets arising from the asymptotic and likelihood distribution in the parametric approach is illustrated, while in Section 3.3 presents the confidence sets that arise from the EB and FBS posterior distribution approximation. In Section 3.4 the proposed definition of confidence set is formalized. Section 3.5 provides an experimental comparison of the parametric and nonparametric methods in terms of point estimators and confidence sets, while Section 3.6 offers some final remarks and conclusions.

3.1 Problem Statement

Consider, for the sake of the exposition, the SISO Output-Error model (a simplified version of model (2.6) with $H(z) = 1$) :

$$y(t) = [g * u](t) + e(t) \tag{3.1}$$

where $y(t)$, $u(t) \in \mathbb{R}$ are respectively the measurable input and output, $e(t)$ is a zero mean Gaussian white noise uncorrelated to $u(t)$ and $g(t)$ is the model impulse response. Also consider $\{u(t)\}$ and $\{y(t)\}$ as jointly stationary zero-mean stochastic processes. Thus, the assumptions of the system identification problem in Section 2.1 hold.

Given a finite set of input-output data points $\mathcal{D}^N = \{u(t), y(t)\}_{t=1}^N$, we are interested in

1. estimating the impulse response $g(t)$ as solution of the PEM problem described in Section 2.2,

2. determining a (random) set which is likely to include the unknown true $g(t)$. This set is generally referred to as *confidence set*,

3. comparing on both the previous points, the PPEM and NPPEM illustrated in Sections 2.3.1 and 2.4.4, respectively.

The questions we are trying to answer are: how accurate are the estimated model w.r.t. the true system? Is it possible to find a common "fair" framework to compare the parametric and nonparametric approaches?

3.2 Confidence Sets of Classical Parametric PEM

Asymptotic

The analysis of this section assumes that an infinite number of data is available, i.e., $N \to \infty$. It is worth mentioning that the limit properties of the PPEM estimate (2.22) are related to the chosen criteria function $V(\theta, \mathcal{D}^N)$, see (Ljung, 1999, Chp. 8).

Consider the PPEM estimate (2.22), under the assumption that the true system belongs to the chosen model class $\mathcal{M}(\theta)$ and some other mild assumptions (e.g. $\hat{\theta}_{PEM}$ gives rise to a uniformly stable model and the given data $\{y(t)\}, \{u(t)\}$ are jointly quasi-stationary signals), it holds that as $N \to \infty$, $\hat{\theta}_{PEM}$ is a *consistent* and *efficient* estimator. This means that with increasing number of data, $\hat{\theta}_{PEM}$ converges to the true system and its covariance matrix approaches the Cramér-Rao limit, so that no unbiased estimator can be better.

In mathematical formula this can be written as

$$\hat{\theta}_{PEM} \to \mathcal{N}\left(\theta_0, \frac{\Sigma_\theta}{N}\right), \quad \text{as } N \to \infty \tag{3.2}$$

where θ_0 is the unique value in Θ such that

$$\hat{\theta}_{PEM} \to \theta_0, \quad \text{w.p. 1 as } N \to \infty \tag{3.3}$$

and

$$\Sigma_\theta = \sigma^2 \left\{ \lim_{N\to\infty} \frac{1}{N} \sum_{t=1}^{N} \mathbb{E}\left[\psi(t,\theta_0)\psi^\top(t,\theta_0)\right] \right\}^{-1} \tag{3.4}$$

$$\psi(t,\theta_0) = \frac{d}{d\theta}\hat{y}_\theta(t|t^-)|_{\theta=\theta_0} \tag{3.5}$$

Notice that, in case of Gaussian innovations Σ_θ coincides with the Cramer-Rao lower bound, thus proving the aforementioned asymptotic efficiency of the PEM estimators. The interested reader is referred to (Ljung, 1999, Chp. 8,9) for more details and extensions to different criteria function $V_N(\mathcal{D}^N, \theta)$.

Once $\hat{\theta}_{PEM}$ has been determined from $\mathcal{D}^N$, the given N input-output pairs, the asymptotic covariance (3.4) can be approximated as

$$\hat{\Sigma}_\theta = V_N(\mathcal{D}^N, \hat{\theta}_{PEM}) \left\{ \frac{1}{N} \sum_{t=1}^{N} \psi(t,\hat{\theta}_{PEM})\psi^\top(t,\hat{\theta}_{PEM}) \right\}^{-1} \tag{3.6}$$

$$\psi(t,\hat{\theta}_{PEM}) = \frac{d}{d\theta}\hat{y}_\theta(t|t^-)|_{\theta=\hat{\theta}_{PEM}} \tag{3.7}$$

It follows that the asymptotic distribution of the estimator can be approximated as

$$p_N(\cdot) \sim \mathcal{N}\left(\hat{\theta}_{PEM}, N^{-1}\hat{\Sigma}_\theta\right) \tag{3.8}$$

Considering distribution (3.8), it is known that the quantity $N(\hat{\theta}_{PEM}-\theta)^\top \hat{\Sigma}_\theta^{-1}(\hat{\theta}_{PEM}-\theta)$ is distributed as a $\chi^2(d)$, which is a chi-squared distribution with $d-$degree of freedom, where d is the dimension of $\hat{\theta}_{PEM}$.

Hence, the ellipsoidal confidence set around the estimate $\hat{\theta}_{PEM}$ with coverage of the $1-\alpha$ percentile, for a fixed probability level α consists of

$$\mathcal{E}_\alpha^{PPEM+ASYMP} := \{\theta \in \mathbb{R}^d : (\hat{\theta}_{PEM}-\theta)^\top \hat{\Sigma}_\theta^{-1}(\hat{\theta}_{PEM}-\theta) \leq \chi_{\alpha,d}\} \tag{3.9}$$

where $\chi_{\alpha,d}$ is the value for which $Pr\left(\chi^2(d) < \chi_{\alpha,d}\right) = \alpha$.

Likelihood Sampling

As an alternative, instead of relying on the approximation (3.6) to the asymptotic covariance (3.4), one could define a confidence set sampling from the likelihood function $p(Y|\theta,\hat{\sigma}^2)$, with $\hat{\sigma}^2$ being a noise variance estimate (obtained e.g. through a Least-Squares model). In fact, assuming a flat prior distribution $p(\theta)$ for the parameters, the likelihood function is proportional to the posterior distribution:

$$p(\theta|Y,\hat{\sigma}^2) \propto p(Y|\theta,\hat{\sigma}^2) = (2\pi\hat{\sigma}^2)^{-N/2}\exp\left\{-\frac{1}{2\hat{\sigma}^2}\sum_{t=1}^{N}(y(t)-\hat{y}_\theta(t|t^-))^2\right\} \tag{3.10}$$

An MCMC algorithm is designed in order to obtain T samples $\theta^{(i)}$ from (3.10). The interested reader is referred to the book Gilks et al. (1995) for an exhaustive explanation of the standard MCMC algorithms. Here, there are reported only the three quantities that need to be specified in order to perform these types of algorithms. First, the target distribution is the posterior distribution defined in (3.10). Second, the proposal density is given by a random walk with increments regulated by the asymptotic distribution (3.8). This can appear counter intuitive, but the asymptotic distribution is a convenient proposal candidate, since it expresses an appropriate distribution over the parameter in opportune conditions, as mentioned in the previous section. Recall that, MCMC algorithms are proved to return the correct target distribution, after a sufficiently high number of samples, independently from the chosen proposal. Third, the acceptance probability for each candidate sample $\theta^{(i)}$ is $\beta(\theta^{(i)},\theta^{(i-1)}) = \min\left(1, \frac{p(Y|\theta^{(i)},\hat{\sigma}^2)}{p(Y|\theta^{(i-1)},\hat{\sigma}^2)}\right)$.

Once the samples $\{\theta^{(i)}\}$, $i \in [1, T]$ are obtained, a subset of them is selected based on the posterior distribution values in order to determine a sampling form of a confidence set, called "Particle" confidence set.

The α-level "Particle" confidence set associated with the $\alpha-$fraction of the highest probability $\left\{p_N(\theta^{(i)})\right\}$ is

$$\mathcal{C}_\alpha^{PPEM+LIK} := \left\{\theta^{(i)} : p(\theta^{(i)}|Y, \hat{\sigma}^2) \geq p_\alpha^{PEM+LIK}, \theta^{(i)} \in \Theta\right\} \tag{3.11}$$

where $p_\alpha^{PEM+LIK}$ is the $(1-\alpha)$-percentile of the set $\left\{p_T(\theta^{(i)})\right\}$, $i = [1, T]$. The set is denoted by the abbreviation $PPEM + LIK$ to emphasize its connection with likelihood function.

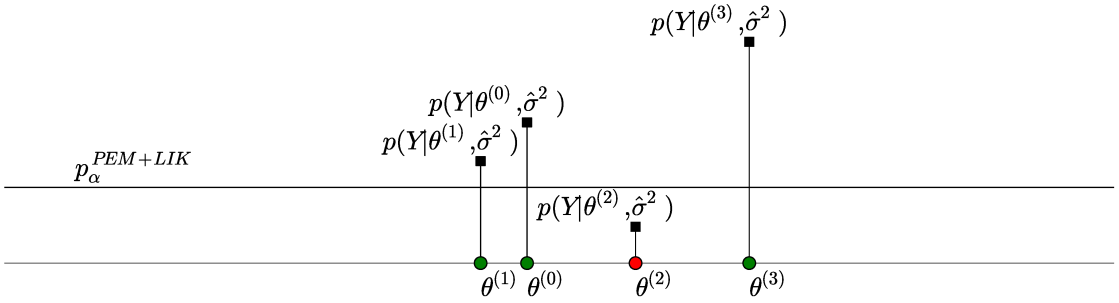

Figure 3.1: An illustration of the "particle" confidence set $C_\alpha^{PPEM+LIK}$ is represented. The parameters $\theta^{(i)}$ are the samples obtained by the MCMC algorithm, at each of them is associated a value of the likelihood function (which is proportional to the posterior) and a selection among of them is performed based on a threshold, $p_\alpha^{PEM+LIK}$. The green $\theta^{(i)}$ are the ones which build the confidence set and the red ones are instead discarded. The purpose of this image is only to give a visualization of the confidence set and it is not a rigorous and general representation.

As previously said, sampling techniques allow to avoid approximations of asymptotic expressions. However, they are still approximations of the true uncertainty associated to the estimated parameter $\hat{\theta}_{PEM}$. Indeed, for the definition of the previous confidence sets, it has been assumed that the model class $\mathcal{M}(\theta)$ and the model complexity are fixed, even if in practice model selection is performed using the available data. That is, $\hat{\theta}_{PEM}$ is a so-called post-model-selection estimator (PMSE): in order to define a more accurate

confidence set, we should take into account also the uncertainty related to the model selection step. However, as emphasized in Leeb and Potscher (2005), the finite-sample distribution of a PMSE generally has a quite intricate shape; moreover, even if one tries to estimate it through a sampling method, one has to recall that the finite-sample distribution of a PMSE is not uniformly close to its asymptotic limit (3.4).

Remark 3.2.1. The reader might wonder why a bootstrap procedure has not been adopted. Indeed, a bootstrap approach could be considered to obtain the samples to compute a particle confidence set in the parameters. The idea behind the bootstrap is that if the model that generated the data is known, it could be possible to generate several datasets corresponding to different realizations of the error, estimate models from these datasets and compute the desired statistics on them. This is clearly not possible since the true model is in general unknown and the only available information are the data $\mathcal{D}^N$. Bootstrap techniques therefore resort on estimating a model from the available data using a PPEM technique, from this model an estimation error and a variance of the error can be computed. At this point, the estimated model can be simulated with an additive noise sampled from the distribution of the previously estimated error. See e.g.,Efron and Tibshirani (1994); Zoubir and Boashash (1998) for an extensive traction of the bootstrap methods. In this way several datasets are generated as in the original idea, then from each dataset a model can be estimated which is a candidate for the particle confidence set.

Notice that this is different from the likelihood sampling above proposed. The difference consists in the fact that in our approach the parameters sets are sampled from a distribution (the posterior) in the bootstrap technique "samples" of datasets are generated and then there is a further estimation step to obtain the parameters samples. A discussion could be set in which of the two methods is more efficient. We rely in the likelihood sampling, which consists in sampling from the posterior distribution, because this is strictly connected with the way confidence sets are naturally defined in the Bayesian framework, as it will be more clear in the next sections. This allows a more fair comparison.

3.3 Confidence Sets of Bayesian Identification Methods

Within the Bayesian framework, the confidence of the final estimator is described by the posterior density $p(g|Y)$. The Empirical Bayes (EB) and the Full Bayesian Sampling (FBS) estimators are derived in Section 2.4.1 from different approximations of the posterior, therefore also different definitions of confidence set are associated to the two

estimators.

Empirical Bayes (EB)

When the Emprical Bayes approach is considered, the posterior $p_\eta(g|Y)$ is the Gaussian distribution defined in (2.51) with η fixed to $\hat{\eta}_{EB}$. Hence, one can define the ellipsoidal confidence region in $\mathbb{R}^n$, with n being the length of the estimated impulse response, i.e. $\hat{g}_{EB} \in \mathbb{R}^n$:

$$\mathcal{E}_\alpha^{EB} := \left\{ x \in \mathbb{R}^n : (x - \hat{g}_{EB})^\top \Sigma_g^{post}(\hat{\eta}_{EB})^{-1}(x - \hat{g}_{EB}) \leq \chi_{\alpha,n} \right\} \tag{3.12}$$

For a fixed probability level α, $\chi_{\alpha,n}$ is the value for which $\Pr(\chi^2(n) < \chi_{\alpha,n} = \alpha$. $\mathcal{E}_\alpha^{EB}$ defines the region in which a sample from $p(g|Y)$ will end up with probability α.

Full Bayes Sampling (FBS)

The Full Bayesian Sampling approach has the advantage that treats η as a random variable and aims to reconstruct the joint distribution of g and η. Therefore in principle, the FBS estimate should be closer to the true system than the EB estimate (under the assumption that the a priori Bayesian model is correct). As a disadvantage, in general it requires a much higher computational effort which, when the marginal posterior $p(\eta|Y)$ is sufficiently peaked, may not be counterbalanced by a significant performance increase.

As discussed in Section 2.4.1 FBS can be implemented through a MCMC sampling algorithm. Here, the FBS estimator of the impulse response g is obtained by an Adaptive Metropolis-Hastings (AM) algorithm, an adaptive version of the more famous Metropolis-Hastings algorithm; see Gilks et al. (1995), Haario, Saksman, and Tamminen (2001). We choose to use this adaptive version instead of the classical counterpart because it allows to tune the proposal distribution exploiting the new knowledge which becomes available through the sampling. This property renders the method robust w.r.t. the initial choice of the proposal distribution.

Recall that the target is to compute the posterior distribution of the impulse response given the data which cannot be computed analytically. For this reason, we tackle the problem by approximating the posterior as

$$p(g|Y) = \int_\eta p_\eta(g|Y)p(\eta|Y)\,d\eta \simeq \frac{1}{N}\sum_{i=1}^{N} p(g|Y,\eta^{(i)}) \tag{3.13}$$

where $p(g|Y,\eta^{(i)})$ is the posterior density (2.51) when the hyperparameters are fixed equal to $\eta^{(i)}$.

In order to do this, we need to design an MCMC algorithm to draw samples $\eta^{(i)}$ from $p(\eta|Y)$. Observe that:

$$p(\eta|Y) = \frac{p_\eta(Y)p(\eta)}{p(Y)} \propto p_\eta(Y) \tag{3.14}$$

where we have assumed that $p(\eta)$ is a non informative prior distribution. Thus, by using (2.56) we can evaluate $p(\eta|Y)$ apart from the normalization constant $p(Y)$.

As mentioned earlier, we have exploited Adaptive Metropolis-Hastings algorithm proposed in Haario et al. (2001) to obtain the samples $\eta^{(i)}$. At each iteration i, the algorithm adopts a Gaussian proposal distribution, $q_i(\cdot)$, centred at the previous sample $\eta^{(i-1)}$ and with a covariance matrix, Π_i, which is adaptively updated based on the samples $\eta^{(1)}, ..., \eta^{(i-1)}$. The updating recursion formula for the covariance matrix given in Haario et al. (2001) is:

$$\Pi_{i+1} = \frac{i-1}{i}\Pi_i + \frac{s_{d_\eta}}{i}(i\bar{\eta}^{(i)}\bar{\eta}^{(i)^\top} + \eta^{(i)}\eta^{(i)^\top} + \epsilon I_{d_\eta}) \tag{3.15}$$

where $\bar{\eta}^{(i)}$ is the mean after i samples, s_{d_η} is a regularization parameter, d_η is the dimension of the hyperparamters and $\epsilon > 0$ is an arbitrarily small constant. The value of the regularization parameter s_{d_η} has been initially set to $s_{d_\eta} = \frac{2.4^2}{d_\eta}$, a value which gives good mixing properties in the Metropolis chain under the assumption of Gaussian target and proposal (as shown in Gelman, Roberts, and Gilks (1996)). Successively, s_{d_η} has been empirically adjusted in order to guarantee an acceptance rate around 30% for the AM algorithm.

The covariance matrix of the proposal density has been initialized with the inverse of the Hessian matrix of the marginal likelihood (2.56) computed at the mode $\hat{\eta}_{EB}$, as it was successfully used e.g., in Pillonetto and Chiuso (2009).

The algorithm we implemented in order to obtain the FBS estimate $\hat{g}_{FB}$ is outlined in Algorithm 1. The chain length T and the burn-in length have been determined by applying twice the method proposed in Raftery and Lewis (1992).

Remark 3.3.1. In addition to the minimum variance estimate (3.16) also the Maximum a Posteriori estimate have been estimated, i.e.,:

$$\bar{g}_{FB} = \arg\max_{g^{(i)}} p(g|Y) \tag{3.17}$$

However, the results are analogous to the minimum variance estimate and therefore omitted.

Algorithm 1 FBS estimate through an AM algorithm

Sample hyperparameters through an AM algorithm
Inputs: $\hat{\eta}_{EB}$, Π_0 and b_{in} the burn-in length

1: **Init:** The proposal density is set to $q_0(\cdot) = \mathcal{N}(\hat{\eta}_{EB}, \Pi_0)$
2: **for** i = 1 **to** $b_{in} + T$ **do**

- Sample η from $q_i(\cdot|\eta^{(i-1)}) \sim \mathcal{N}(\eta^{(i-1)}, \Pi_i)$
- Sample u from a uniform distribution on $[0, 1]$
- Set
$$\eta^{(i)} = \begin{cases} \eta & \text{if } u \leq \frac{p(Y|\eta)p(\eta)}{p(Y|\eta^{(i-1)})p(\eta^{(i-1)})} \\ \eta^{(i-1)} & \text{otherwise} \end{cases}$$
- Compute Π_{i+1} according to equation (3.15).

3: Retain the last T samples $\eta^{(i)}$ which are (approximately) samples from $p(\eta|Y)$.

Estimate the impulse response:
4: **for** i = 1 **to** T **do**

- Compute $\mu_g^{post}(\eta^{(i)})$, $\Sigma_g^{post}(\eta^{(i)})$ as in (2.52), (2.53).
- Sample $g^{(i)}$ from $\mathcal{N}(\mu_g^{post}(\eta^{(i)}), \Sigma_g^{post}(\eta^{(i)}))$

5: The samples $g^{(i)}$ obtained above are samples from $p(g|Y)$. Compute $\hat{g}_{FB}$ as:

$$\hat{g}_{FB} = N^{-1} \sum_{i=1}^{N} g^{(i)} \tag{3.16}$$

Confidence Set The FBS estimator above described exploits the sample approximation of the posterior distribution in (3.13). The approximated approximated distribution is not Gaussian and it is not possible to define an ellipsoidal confidence region as in the EB case. However, an appropriate α-level confidence set can be given by:

$$\mathcal{S}_{\alpha}^{FBS} := \{g^{(i)} \in \mathbb{R}^n : N^{-1} \sum_{j=1}^{N} p(g^{(i)}|Y, \eta^{(j)}) \geq p_{\alpha}^{FBS}\} \tag{3.18}$$

where p_{α}^{FBS} is the $(1-\alpha)$-percentile of the set $\{N^{-1}\sum_{j=1}^{N} p(g^{(i)}|Y,\eta^{(j)});\ i = [1, N]\}$. Which means that S_{α}^{FBS} is a "Particle" confidence set composed by the impulse response samples $g^{(i)}$ associated with the α-fraction of the highest values of the posterior (3.13).

Figure 3.2 gives a qualitative visualization of the confidence set S_{α}^{FBS}.

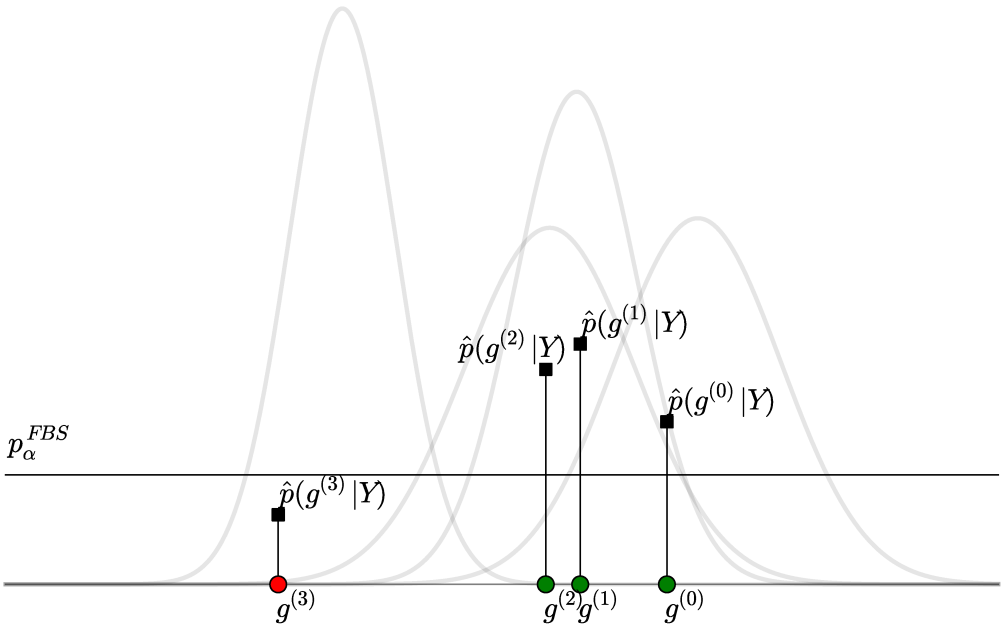

Figure 3.2: Illustration of the Full Bayesian Sampling confidence set. The green impulse responses $g^(i)$ with $i = \{0, 1, 2, 3\}$ represent the samples that belong to the confidence set, the red impulse response represents the samples discarded.

3.4 A Common Framework: "Particle" Confidence Sets on the Impulse Response Space

The confidence sets for the parametric PEM and Bayesian methods presented in Sections 3.2 and 3.3, respectively, show how the characteristics of these two approaches lead to

different confidence sets both in terms of domain (parameter space or impulse response space) and typology ("Particle" or closed sets). Thus making the comparison between the two particularly delicate.

Indeed, the parametric confidence sets are defined in the space of the parameters of dimension d and the asymptotic approach lead to the ellipsoidal region, $\mathcal{E}_\alpha^{PPEM+ASYMP}$ in (3.9), while the likelihood approach lead to the "Particle" region, $\mathcal{C}_\alpha^{PPEM+LIK}$ in (3.11). Instead, the Bayesian confidence sets are defined in the space of the impulse responses of dimension n and the EB approach lead to the ellipsoidal region, $\mathcal{E}_\alpha^{EB}$ in (3.12), while the FBS approach leads to a "Particle" region, $\mathcal{S}_\alpha^{FBS}$ in (3.18).

In order to evaluate and compare the quality of all the approaches in a common framework we propose to define the confidence sets as "Particle" regions in the impulse response space.

Recall that, the impulse response is the output obtained exciting the systems with a Kronecker-delta and it corresponds to the inverse $\mathcal{Z}$-transform of the transfer functions of the systems. The map that goes from the parameters to the impulse response is therefore a nonlinear map and it depends on the specific model class $\mathcal{M}(\theta)$. This map can generally defined as:

$$\begin{aligned} \mathcal{I} : \Theta &\rightarrow \mathbb{R}^n \\ \theta &\mapsto g \end{aligned} \tag{3.19}$$

Our belief is that performing the comparison in the impulse response space is a fair choice, since the impulse response explicitly describes the input-output relation of the system to be identified. A comparison in the parameter space would have required a model reduction step on the Bayesian estimates: we believe that this step is more delicate than the non-linear transformation we had to apply on parametric estimates in order to map parameter estimators to impulse response estimators.

Notice that the FBS confidence set is already in the desired form for the comparison. In the following it is shown how the confidence regions of the three remaining approaches are transformed and they will be indicated with the symbol $\mathcal{S}_\alpha^X$, where X denotes the specific approach.

Asymptotic

Observe that the covariance (3.6) describes the approximated asymptotic confidence set in $\mathbb{R}^d$, the space of the parameters θ. In order to map the confidence set (3.9) in a

"Particle" region in the impulse response space a Monte-Carlo technique is adopted.

First, several samples $\theta^{(i)}$ are drawn from the approximated asymptotic distribution $p_N(\theta)$ defined in (3.8) and among them the one that belongs to the ellipsoid (3.9) are retained.

Second, for each of the selected $\theta^{(i)}$ a model $\mathcal{M}(\theta^{(i)})$ is built and the associated impulse response $g^{(i)}$ of length n is computed accordingly to the map defined in (3.19).

The "Particle" confidence set is composed by all the $g^{(i)}$ with $i = [1, T]$, i.e:

$$\mathcal{S}_{\alpha}^{PPEM+ASYMP} = \left\{ g^{(i)} = \mathcal{I}(\theta^{(i)}) \in \mathbb{R}^n : \theta^{(i)} \in \mathcal{E}_{\alpha}^{PPEM+ASYMP} \right\} \tag{3.20}$$

where $\mathcal{E}_{\alpha}^{PPEM+ASYMP}$ is the ellipsoidal confidence set defined in (3.9).

In Figure 3.3 a qualitative illustration of the procedure to compute the confidence set (3.20) is presented.

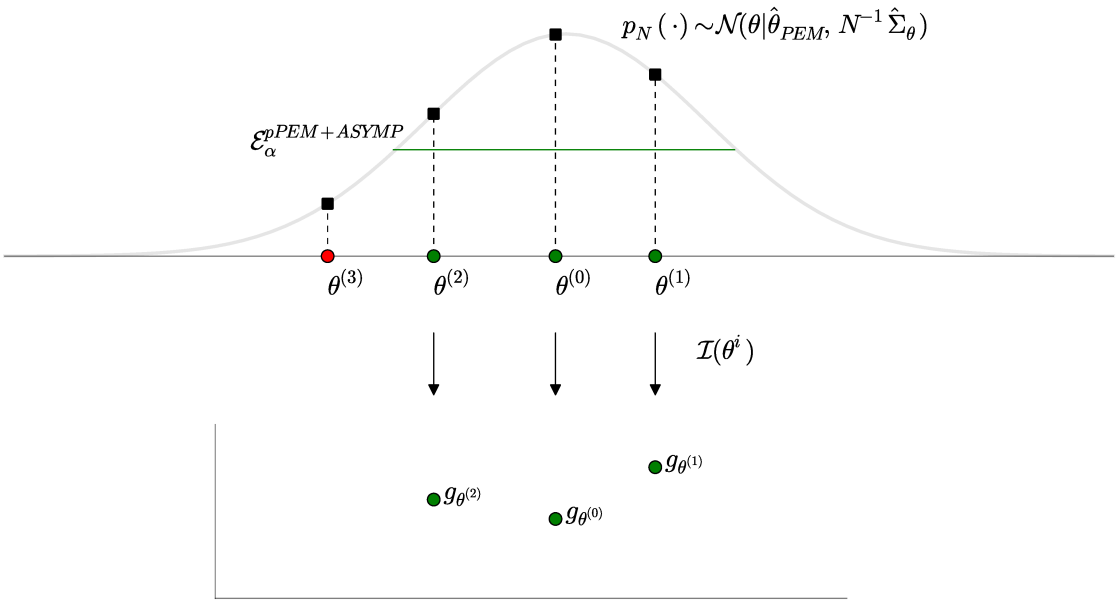

Figure 3.3: Illustration of the "Particle" confidence set in the impulse response space for the asymptotic distribution of the PPEM. The Monte-Carlo samples $\theta^{(i)}$ are obtained from the asymptotic posterior distribution$p_N(\theta)$. The green $\theta^{(i)}$ represent the samples accepted by the $p_{\alpha}^{PPEM+ASYMP}$ threshold and the red $\theta^{(i)}$ represent the discarded ones. The black arrows represent the function $\mathcal{I}(\theta^(i))$ that maps the parameter samples in the impulse response space. Finally, the green impulse responses are the samples that compose the "Particle" confidence set $\mathcal{S}_{\alpha}^{PPEM+ASYMP}$.

Likelihood

It has been shown that the likelihood distribution $p(\theta|Y,\hat{\sigma}^2)$ yields the "Particle" confidence set $\mathcal{C}_\alpha^{PPEM+LIK}$ in (3.11) defined over the parameter space. Hence, a set of samples $\theta^{(i)}$ with $i=[1,T]$ is available.

For each $\theta^{(i)} \in \mathcal{C}_\alpha^{PPEM+LIK}$ the corresponding impulse responses $g^{(i)}$ are computed using the map (3.19). These impulse responses compose the "Particle" confidence set:

$$\mathcal{S}_\alpha^{PPEM+LIK} = \left\{ g^{(i)} = \mathcal{I}(\theta^{(i)}) \in \mathbb{R}^n : \theta^{(i)} \in \mathcal{C}_\alpha^{PPEM+LIK} \right\} \tag{3.21}$$

Figure 3.4 recalls Figure 3.1 and shows a qualitative illustration of how the confidence set $\mathcal{S}_\alpha^{PPEM+LIK}$ is computed starting from (3.11).

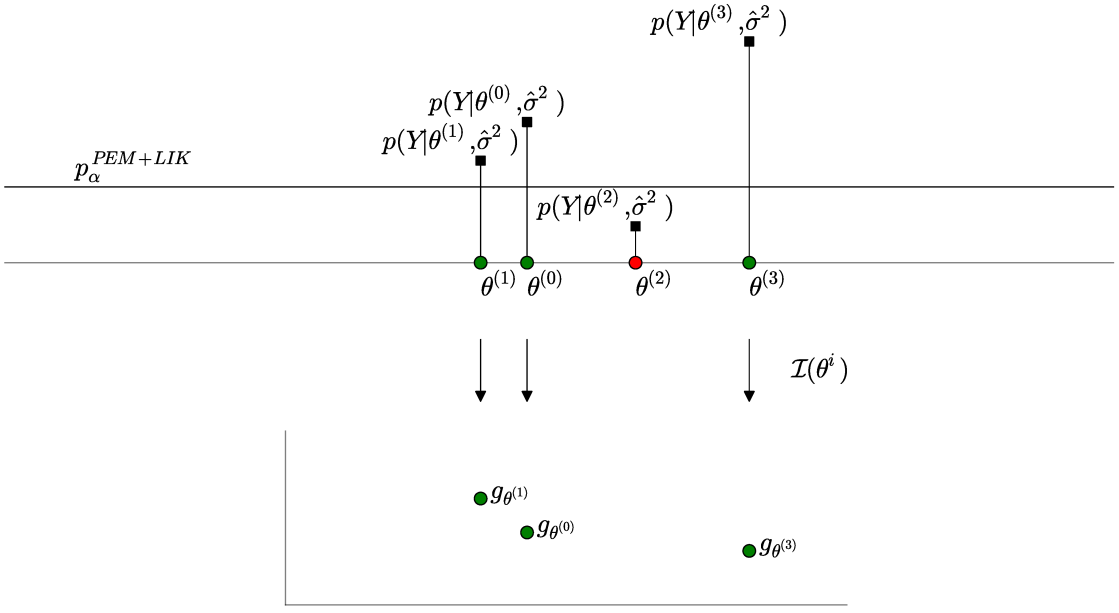

Figure 3.4: Illustration of the "Particle" confidence set in the impulse response space from the likelihood distribution of the PPEM. The $\theta^{(i)}$ are the Monte-Carlo samples obtained from $p(\cdot|Y,\hat{\sigma}^2)$. The green $\theta^{(i)}$ represent the sample accepted by the $p_\alpha^{PPEM+LIK}$ threshold and the red ones represents the discarded samples. The black arrows represent the function $\mathcal{I}(\theta^(i))$ that maps the parameter samples in the impulse response space. Finally, the green impulse responses are the samples that compose the "Particle" confidence set $\mathcal{S}_\alpha^{PPEM+LIK}$.

Empirical Bayes

The Empirical Bayes method is based on modeling directly the impulse response of the system, consequently, its confidence set (3.12) is an ellipsoid with domain defined on the impulse response space. In order to have a "Particle" region, the posterior distribution

$p_{\hat{\eta}_{EB}}(g|Y)$ defined in (2.51), with η fixed to $\hat{\eta}_{EB}$, is sampled using a Monte-Carlo approach and only the samples which belong to $\mathcal{E}_\alpha^{EB}$ are retained. That is:

$$\mathcal{S}_\alpha^{EB} = \{g^{(i)} \in \mathbb{R}^n : g^{(i)} \in \mathcal{E}_\alpha^{EB}\} \tag{3.22}$$

where $\mathcal{E}_\alpha^{EB}$ was defined in (3.12).

Remark 3.4.1. At this point one could argue that the sets S_α^X, where X denotes a generic method among the ones previously illustrated, that is, PPEM+LIK, PPEM+ASYMP, EB and FBS, are only "sample" approximations of a confidence set, while one may be interested in having a bounded region, in the impulse response space, as a confidence set. In the case of the EB estimator this region is directly defined since the posterior distribution is Gaussian, thus naturally leading to ellipsoidal confidence regions (3.12). For all the other estimators, it is in principle possible to build outer approximations of the confidence sets e.g. building a minimum size set which includes all the points in S_α^X; examples are the convex hull or an ellipsoid. The convex hull can be computed with off-the-shelf algorithms (such as the Matlab routine `convhulln.m`), while the smallest ellipsoid (in terms of sum of squared semi-axes length) can be found solving the following problem:

$$\begin{aligned} P_\alpha^{opt}, c_\alpha^{opt} := \quad & \arg\min_{P,c} \quad \text{Trace } P \\ & s.t. \qquad \begin{bmatrix} P & (g^{(i)} - c) \\ (g^{(i)} - c)^\top & 1 \end{bmatrix} \succ 0, \\ & g^{(i)} \qquad \in S_\alpha^X \end{aligned} \tag{3.23}$$

See Calafiore (2002) for further details. The corresponding ellipsoid is

$$\mathcal{E}_\alpha^{opt} = \left\{ x \in \mathbb{R}^n : (x - c_\alpha^{opt})^\top (P_\alpha^{opt})^{-1} (x - c_\alpha^{opt}) \leq 1 \right\} \tag{3.24}$$

However, the computation of the convex hull as well as the solution of the optimization problem (3.23) become computationally intractable for moderate ambient space and sample sizes. E.g. when the impulse response lives in $\mathbb{R}^n$, $n = 100$ and the set S_α^X contains thousands of points (as in the situation we are facing), these computations are prohibitive with off-the-shelf methods. To overcome this issue, we tried to approximate the optimal ellipsoid $\mathcal{E}_\alpha^{opt}$ by using the sample mean $\bar{g}_{S_\alpha^X}$ and the sample covariance $\Sigma_{S_\alpha^X}$

of the elements in S_α^X; namely:

$$\begin{aligned} \mathcal{E}_\alpha^X &= \left\{ x \in \mathbb{R}^n : (\Pi_\alpha^X)^\top \Sigma_{S_\alpha^X}^{-1} d_\alpha^X \leq k_\alpha^X \right\}, \\ d_\alpha^X &= x - \bar{g}_{S_\alpha^X} \end{aligned} \tag{3.25}$$

where k_α^X is a constant appropriately chosen so that all the elements of S_α^X fall within $\mathcal{E}_\alpha^X$. However, it can be observed that these ellipsoids are rather rough approximations of the sets S_α^X. E.g., inspecting 2D sections of the n-dimensional ellipsoids, it can be seen that often the axis orientation was not correct, thus leading to sets which are much larger than needed. This fact was mainly observed for the confidence sets related to PEM estimates.

These observations suggest that the quality of the confidence sets obtained through the ellipsoidal approximation (3.25) would have been highly dependent on the quality of the fitted ellipsoid. Therefore, we concluded that a comparison among the different estimators, based on this kind of confidence set, would have led to unreliable results; therefore such results have not been reported.

3.5 Simulations Results

The experiment consists in a Monte-Carlo simulation with 200 runs. At each run, a model such as (3.1) is estimated together with its confidence for the PPEM and NPPEM. The quality of the estimators are compared in terms of both the impulse response fit and the accuracy of the corresponding confidence set, determined as illustrated in Section 3.4.

Data

The data-bank of systems and input-output data used in our experiments have been introduced in Chen, Andersen, Ljung, Chiuso, and Pillonetto (2014). In particular, we applied the identification techniques to the data set "D4" which is briefly described in the following.

The data set consists of 30th order random SISO dicrete-time systems having all the poles inside a circle of radius 0.95. These systems were simulated with a unit variance band-limited Gaussian signal with normalized band $[0, 0.8]$. A zero mean white Gaussian noise, with variance adjusted so that the Signal to Noise Ratio (SNR) is always equal to 1, was then added to the output data. Refer to Chen et al. (2014) for further details on dataset "D4". We consider three different data lengths: $N_1 = 250$, $N_2 = 500$, $N_3 = 2500$.

In addition, we experimented also the data set "D2" from the same data-bank, in

which the input signals are not filtered with a low pass filter, and the data set "S1D2" introduced in Chen et al. (2012) which have a different SNR. The results obtained from these data sets are similar to the ones obtained from dataset "D4" and therefore are not reported here.

Estimators

Parametric PEM. The MATLAB routine `oe` to implement the PPEM procedure is adopted. Model selection has been performed through BIC criterion, since it generally outperforms AIC. We will denote this estimator as PPEM+BIC.

Moreover, as a reference we also consider an oracle estimator, denoted by PPEM+OR, which has the (unrealistic) knowledge of the impulse response of the true system, g: among the OE models with complexity ranging from 2 to 30, it selects the one which gives the best fit to g.

Nonparametric PEM. The Bayesian estimates have been obtained adopting a zero-mean Gaussian prior with a covariance matrix (kernel) given by the so-called DC-kernel:

$$K_\eta^{DC}(k,j) = c\rho^{|k-j|}\lambda^{(k+j)/2} \tag{3.26}$$

where $c \geq 0$, $0 \leq \lambda \leq 1$ and $|\rho| \leq 1$ are the hyperparameters which form the set $\eta = \{c, \rho, \lambda\}$. For further details on the meaning of these hyperparameters and on the properties they induce in the estimated impulse response we refer to Chen et al. (2012), where the DC kernel has been proposed. The length n of the estimated impulse responses has been set to 100.

For ease of notation, we will use the apex (or the subscript) X to denote a generic estimator among the ones previously illustrated, that is, PPEM+BIC, PPEM+OR, EB and FBS.

Impulse Response Fit

As a first comparison, we would like to evaluate the ability of the considered identification techniques on the reconstruction of the true impulse response. Thus, for each estimated system and for each estimator X we compute the so-called impulse response fit:

$$\mathcal{F}^X = 100 \cdot \left(1 - \frac{\|g - \hat{g}_X\|_2}{\|g\|_2}\right) \tag{3.27}$$

where g, $\hat{g}_X$ are the true and the estimated impulse responses of the considered system.

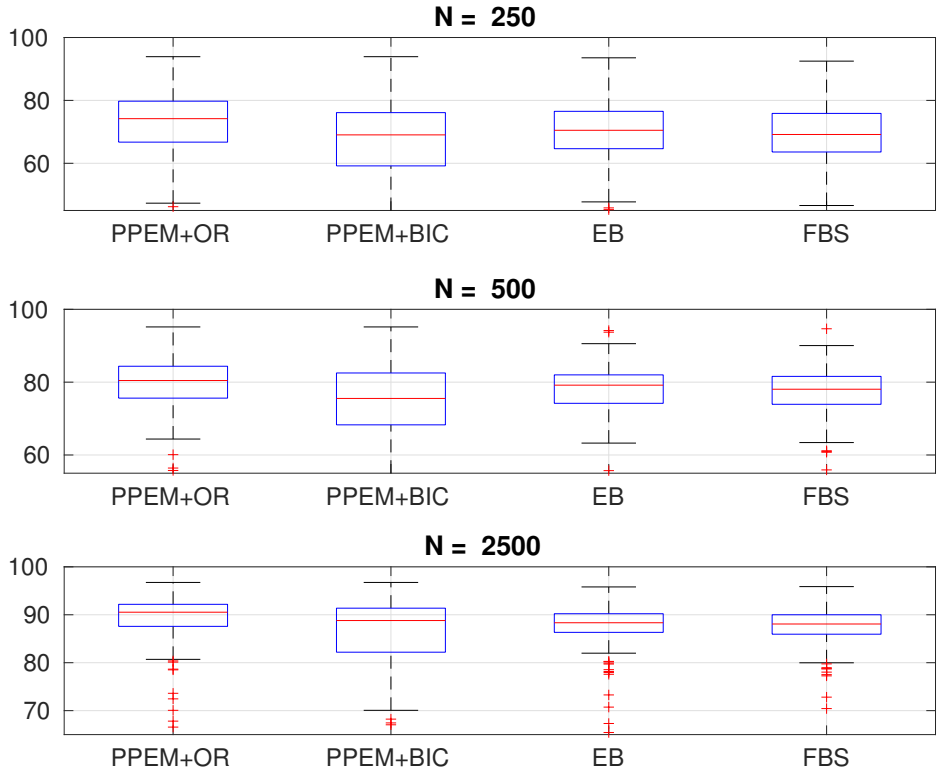

Figure 3.5: Monte Carlo results. Boxplots of the impulse response fit for the compared identification techniques and for different data lengths N.

Figure 3.5 and Table 3.1 displays the boxplots and the average of index (3.27) for the 4 estimators and for increasing data lengths N.

	PPEM+OR	PPEM+BIC	EB	FBS
Fit Mean N = 250	71.4341	56.2997	69.9300	68.2640
Fit Mean N = 500	78.3256	67.1082	77.5604	76.7928
Fit Mean N = 2500	88.8431	74.8353	87.0588	85.9443

Table 3.1: Comparison of average impulse response fit and for different data lengths N.

The oracle estimator PPEM+OR sets an upper bound on the achievable performance by parametric methods; we can note that EB performs remarkably well, with only a slightly inferior fit. The FBS estimator performs similarly to EB, but it requires the implementation of a MCMC, which is highly computationally expensive. These results suggest that the marginal posterior $p(\eta|Y)$ is sufficiently well peaked to be approximated by a delta function (meaning that $p(g|Y) \simeq p(g|Y, \hat{\eta}_{EB})$).

The PPEM+BIC estimator has weaker performances: a lower median and a long tail of systems with low fit (cut out from the figure for readability reasons) are obtained. This is most likely due to the low pass characteristics of the input signal, which make the order estimation step particularly delicate. Indeed, in the dataset "S1D2" where the inputs were Gaussian white noises, PPEM+BIC performed similar to the Bayesian estimators.

Note that, as expected, when a larger number of data is available the fit of the 4 estimators improve and in particular, the parametric estimate for $N = 2500$, becomes competitive with the Bayesian ones. Recall that if the order model is chosen correctly the parametric estimates are consistent.

Confidence Set Indexes

The confidence sets which have been introduced in Section 3.4 associated to the estimators proposed previously are: $S_\alpha^{PPEM+OR+ASYMP}$, $S_\alpha^{PPEM+OR+LIK}$, $S_\alpha^{PPEM+BIC+ASYMP}$, $S_\alpha^{PPEM+BIC+LIK}$, S_α^{EB} and S_α^{FBS}. As before, S_α^X will generically denote one of them.

In the simulations we present, we have set $\alpha = 0.95$. Furthermore, the number of samples N that are used to construct the above-mentioned confidence sets takes different values for each of the considered Monte-Carlo runs. Specifically, it has been set as the maximum chain length of the three MCMC algorithms exploited in our setting (i.e. the MCMC algorithms used for Likelihood Sampling for the two PEM estimators and the AM used to compute the Full Bayes estimator). Recall that for each of these algorithms, the chain length and the burn-in length have been set by applying twice the method proposed in Raftery and Lewis (1992).

Given that the confidence sets we consider are only approximations of a "true" α-level confidence set, our aim is to study how well they perform both in term of "coverage" (how often does the α-level confidence set contain the "true" value?) as well as of size (how big is an α-level confidence set?). Unfortunately, since our sets are only defined through a set of points, it is not possible to define a notion of inclusion (does the true system belong to the confidence set?) and as a proxy to this we thus consider the following index which measures the relative distance from the true system and the closest point within the confidence set:

1. *Coverage Index*: For a fixed probability level α, it is given by

$$\mathcal{I}_1^X(\alpha) := \min_{x \in S_\alpha^X} \frac{\|x - g\|_2}{\|g\|_2} \tag{3.28}$$

where g denotes the true impulse response. For future analysis the usage of the concept "coverage" will be meant as in definition (3.28).

2. *Confidence Set Size*: It evaluates the area of the interval which includes all the impulse responses contained in S_α^X. Let us define the vectors $\bar{g}_X \in \mathbb{R}^n$ and $\underline{g}_X \in \mathbb{R}^n$ whose j-entries are $\bar{g}_X(j) := \max_i g^{(i)}(j)$ and $\underline{g}_X(j) := \min_i g^{(i)}(j)$, respectively, with $g^{(i)} \in S_\alpha^X$; the index we consider is defined as:

$$\mathcal{I}_2^X(\alpha) = \sum_{j=1}^{n} \bar{g}_X(j) - \underline{g}_X(j) \tag{3.29}$$

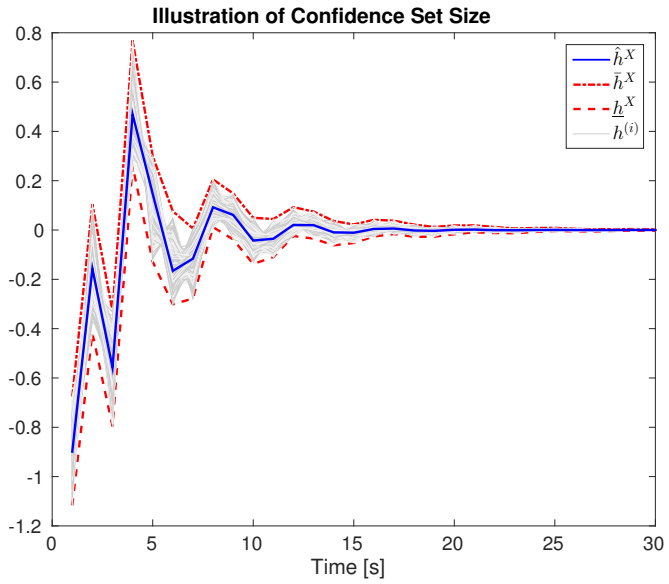

Figure 3.6: Illustration of the idea of the *Confidence set size* index for a single system. The blue line represents the point estimator $\hat{g}_X$, the dashed red line and the dot-dashed red line represent the lower values $\underline{g}_X$ and the upper values $\bar{g}_X(j)$ of the confidence set, respectively, and the gray lines represent the impulse responses samples within the confidence set.

Referring to Figure 3.6, a large confidence set is more likely to contain the true impulse response, giving a low value of $\mathcal{I}_1^X(\alpha)$, but it will also denote a high uncertainty in the returned estimate, thus leading to a large value of $\mathcal{I}_2^X(\alpha)$.

Figure 3.7 illustrates the boxplots for index (3.28) when different data lengths N are considered. The Bayesian confidence sets have higher coverage performances then the parametric ones equipped with BIC. The unique exception is for the PPEM+BIC ASYMP confidence set when the data length is $N = 2500$, that is, when the asymptotic theory is more reliable. Their accuracy is comparable with the one achieved by the

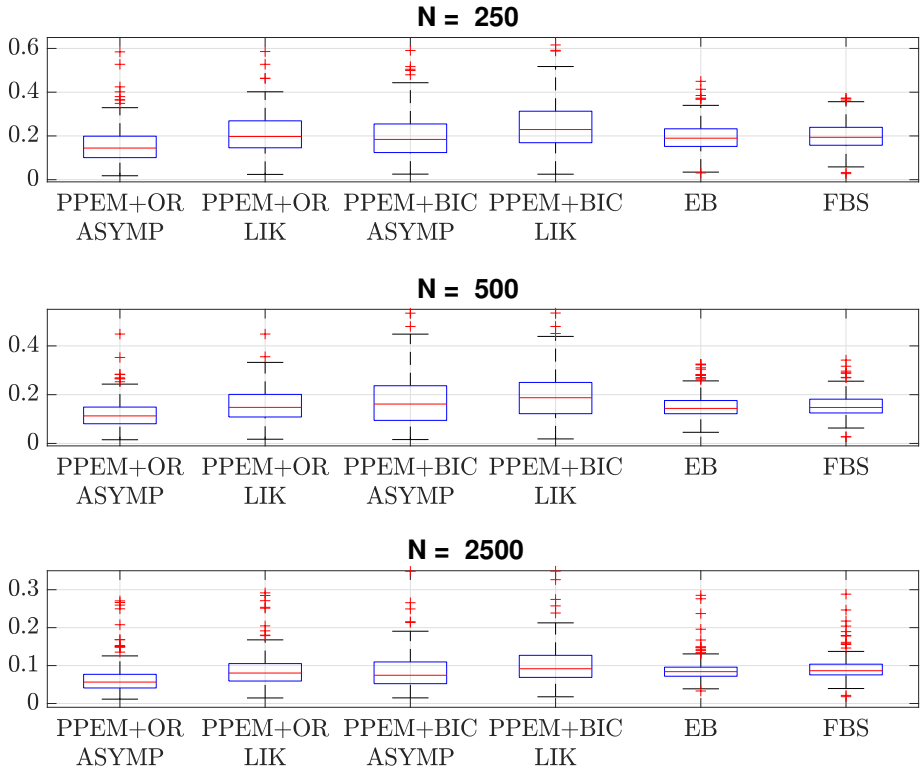

Figure 3.7: Monte Carlo results. Boxplots of the *Coverage Index* for the compared identification techniques and for different amounts of data in the dataset.

PPEM+OR LIK confidence set, which is favoured by the knowledge of the true system. No substantial differences are detected between the two Bayesian approaches we compare.

Among the parametric confidence sets, as expected, the PPEM+OR outperform the PPEM+BIC, whereas surprisingly, the asymptotic confidence sets outperform the PPEM+BIC which are built precisely for finite data lengths. This result can be explained analysing also index (3.29) displayed in Figure 3.8, the discussion is therefore postponed. Note that the asymptotic confidence set have, correctly, a significant improvement for larger data lengths.

Figure 3.8 illustrates the boxplots for index (3.29), for different data lengths.

The EB confidence set has a slightly smaller size than the FBS, which is rather obvious since in the latter also uncertainty related to the hyper-parameters is accounted for.

In this case the parametric approaches equipped with the likelihood sampling return the smallest confidence sets, even smaller than the Bayesian ones. However, the coverage index in Figure 3.7 shows that they are less accurate than the Bayesian one.

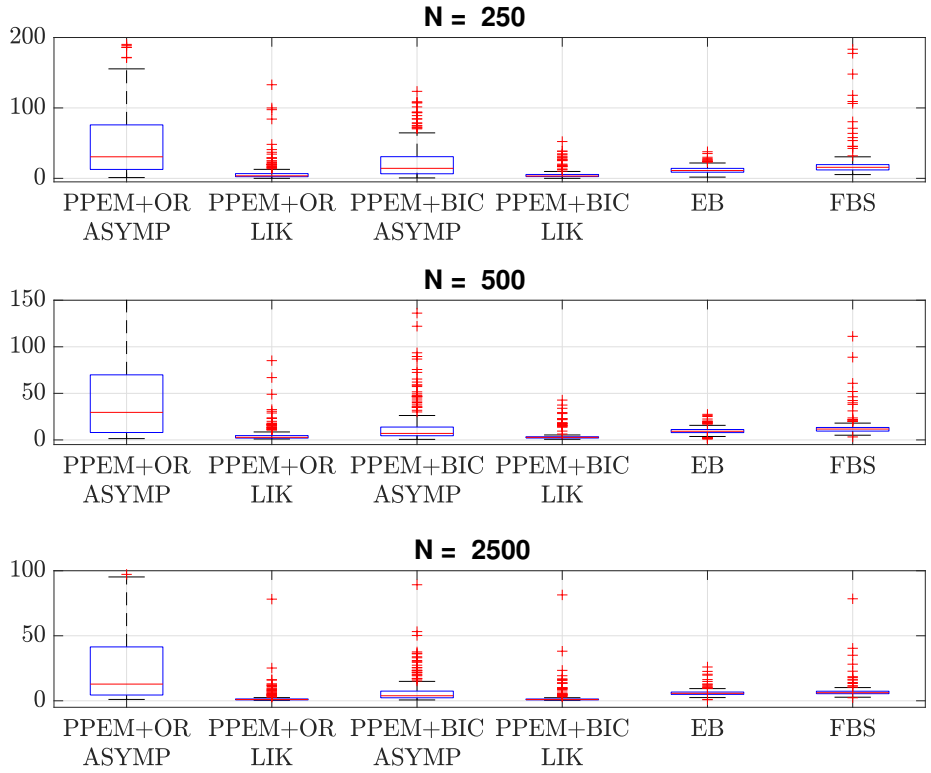

Figure 3.8: Monte Carlo results. Boxplots of the *Confidence Set Size* for the compared identification techniques and for different data lengths N.

Furthermore, notice that the two PPEM+OR confidence sets are larger than the ones returned by the PPEM+BIC estimator: this can be explained from the fact that PPEM+OR tends to select higher-order models, thus bringing more uncertainty into the estimated systems. Comparing the *Asymptotic* and the *Likelihood Sampling* confidence sets it is clear that the latter is the more precise one. Indeed, the *Asymptotic* confidence set is an approximation which holds for large data sets, while the *Likelihood Sampling* is correct for any finite sample size; however, this improvement comes at a rather high computational price needed to run the MCMC sampler.

This explains why the asymptotic confidence sets outperform the likelihood ones in the metric (3.28): being much larger they have higher coverage performances. Analysing the size and coverage properties of the likelihood confidence sets they seems to be too much self-confident giving a small uncertainty to their estimate but with not satisfactory performances in terms of coverage.

It is important to note that the asymptotic theory does not take into account stability issues: namely, the confidence set derived from the Gaussian asymptotic distribution (3.2)

could contain unstable impulse responses. Therefore the sampling procedure described in Section 3.2 could yield to diverging confidence set size. In order to avoid this problem we truncated the asymptotic Gaussian distribution within the stability region. Clearly, this fact shows an intrinsic problem of the asymptotic theory. We should also recall that the asymptotic as well as likelihood based confidence intervals do not account for uncertainty in the order estimation step.

By comparing the results in both Figures 3.7-3.8 we can conclude that: among the feasible identification methods, EB and FBS are preferable taking into account performances in terms of both coverage and size. In this case there seems to be no gain in using the more computationally expensive FBS.

3.6 Conclusions

For the best of our knowledge this is the first tentative of an exhausting comparison between the PPEM and the NPPEM accounting also the confidence set and we believe in the fairness of the proposed method and experiments. However, the proposed framework does not pretend to be the only possible solution, how to perform the comparison can be still a matter of discussions, other possibilities can be achieved and contrasting opinions about the fairness of the comparison can arise.

The achieved results complement previous findings showing that Bayesian methods not only outperform parametric methods in terms of point estimators, but also provide better approximations for uncertainty regions.

From the limited experience acquired in the simulations proposed, there seems to be very little advantage in using Full Bayes Sampling approaches which entail a much higher computational load than Empirical Bayes methods.

It is interesting to note that Bayesian estimators and their confidence sets are competitive even with the parametric methods equipped with an oracle which has the knowledge of the true impulse response.

Finally, with regard to the parametric techniques, the confidence sets obtained from the likelihood probability distribution are in general more accurate than the ones returned by the “asymptotic” approximation.

Discussion

The results obtained in this chapter confirm the general discussion on the bias-variance tradeoff and model complexity carried in Section 2.5. The benefits of using the NPPEM w.r.t. the PPEM in terms of tradeoff among the accuracy of the confidence sets, the size

of the confidence sets and the accuracy of the point estimators reflect the effectiveness of the former techniques to tackle the bias-variance tradeoff.

4

Online Gaussian Regression

The identification of time-varying systems plays a key role in different applications, such as adaptive and model predictive control, where a satisfactory real-time tracking of the system to be controlled is necessary. In addition, the detection of changes or drifts in plant parameters is crucial in terms of process monitoring and fault detection.

Online system identification and the estimation of time-varying systems are typically strictly connected problems: one would like to exploit the new data that become available in order to track real-time possible changes in the system dynamics, e.g., situations in which a sensor provides new measurements at fixed time intervals.

Recursive parametric prediction error method (RPPEM), a variant of the classical parametric PEM Ljung (1999); Ljung and Söderström (1983), represents nowadays a well-established technique, through which the current estimate can be efficiently updated, as soon as new data are provided, see Section 2.3.2 for a brief description. RPPEM approaches rely on recursive least-squares (or pseudo LS) routines, which compute the

parameter estimate by minimizing a function of the prediction errors (Ljung, 1999, Chp. 11).

An extension of these approaches for the identification of time-varying systems involves the adoption of a forgetting factor, through which old data become less relevant in the estimation criterion. Convergence and stability properties of Forgetting Factor RPPEM have been well-studied within the system identification community Lozano (1983); Bittanti, Bolzern, and Campi (1990); Guo, Ljung, and Priouret (1993); Dasgupta and Huang (1987). Alternative approaches model the coefficients trajectories and regard them as stochastic processes, thus exploiting Kalman filtering for parameter estimation Guo (1990). Within this research direction, some authors considered the approximation of the parameters time evolution through a combination of some bases sequences, e.g. wavelet basis, Tsatsanis and Giannakis (1993).

The above-mentioned parametric procedures share the criticality of the model selection complexity. It is well known that selecting the model complexity is a critical issue in parametric system identification (Ljung, 1999; Söderström and Stoica, 1989; Pillonetto and De Nicolao, 2010; Pillonetto et al., 2011a; Chen et al., 2012; Pillonetto et al., 2014) and it gets more critical in the recursive framework, in particular when the system under analysis is time-varying. In fact, model complexity selection rules, which trade model complexity versus fit, may turn out to give different answers as new data become available; of course if the "true system" is also time varying one should actually expect that also the estimator follows these variations. Dealing with parametric model classes in which the order changes over time is definitely a delicate (and possibly nontrivial) issue.

The nonparametric Gaussian regression techniques, recently introduced in the system identification community by Pillonetto et al. (2011a, 2014), see Section (2.4.4), do not offer a structure naturally suitable for recursive update as it is for PPEM. For this reason, there is not a standard technique for recursive NPPEM. See Section 2.4.5 for an overview on the literature.

In this Chapter, the NPPEM framework is extended by introducing an incremental procedure, which is suitable for an online setting and for coping with time invariant and time-varying systems.

Contributions

We propose an online NPPEM identification procedure, with fixed[1] computational complexity and memory storage, in which

[1] i.e., independent on the number of data available.

1. the hyperparameters are recursively estimated through one-step-updates of an algorithm optimizing the marginal likelihood,

2. the system impulse response estimate is computed with fixed computational complexity,

3. time-varying changes of the system dynamics can be identified.

1. The one-step-updates of the hyperparameters are obtained by gradient-based as well as EM-based algorithms and comparisons among these methods will be provided through simulation results in terms of both accuracy and computational time. Some connections between EM-based, gradient-based methods and iteratively reweighted schemes will be also provided, showing that there is a strong similarity among these seemingly different approaches.

2. This result is straightforward a consequence of the efficient hyperparameters update at point a). Since the hyperparameters become available with fixed computational complexity by applying standard recursive rules also the computation of the impulse response is independent on the number of data.

3. In order to deal with time-varying systems, three approaches, relying on the use of a forgetting factor or of a sliding window over the data, are proposed. In particular, we investigate the estimation of the forgetting factor by treating it as a hyperparameter of the Bayesian inference procedure. These techniques are experimentally compared with the classical parametric counterparts described in Section 2.4.5: the results appear favourable and promising for the methods we propose.

The chapter is organized as follows. Section 4.1 presents the problem statement of the online setup, while Section 4.2 provides the online identification algorithm for linear NPPEM and some first experimental results to validate the method. Section 4.3 focuses on the techniques to extend the proposed method to the estimation of time-varying systems. In Section 4.4 final experiment showing the efficiency of the proposed setting are presented. Finally, in Section 4.4 future research directions and connections to the the bias-variance dilemma are drawn.

4.1 Problem Statement

Consider a dynamical system described through a SISO Output-Error model, i.e.,:

$$y(t) = [g_t * u](t) + e(t), \quad y(t),\ u(t) \in \mathbb{R} \tag{4.1}$$

where $e(t)$ is assumed to be a zero-mean Gaussian noise with variance σ^2 and $g_t(t)$ denotes the model impulse response and is assumed to be time-varying i.e., its coefficients might vary along with time passing by.

Model (4.1) cannot be considered as a simplified version of model (2.6) considering $H(z) = 1$ because here the coefficients of the impulse response are allowed to change along with time. Anyway, model (4.1) is a particular case of model (2.2).

System identification techniques are designed in order to estimate the impulse response g of the system, once a set $\mathcal{D} = \{u(t), y(t)\}_{t=1}^{N}$ of measurements of its input and output signals is provided.

In this chapter an "online" setting is considered, in which a new set of input-output measurements becomes available every T time steps. Specifically, let us define the variable $i := k/T$ by assuming, w.l.o.g., that k is a multiple of T and the $i^{th}-$dataset as $\mathcal{D}_i = \{u(t), y(t)\}_{t=(i-1)T+1}^{iT}$. The variable i is the cardinality of the datasets referred to the data points $[k - T, \dots, k]$. In the remaining of the chapter, the superscript $^{(i)}$ will denote quantities that are computed after dataset $\mathcal{D}_i$ becomes available.

The framework is such: at time k an impulse response estimate $\hat{g}^{(i)}$ has been computed using the data coming from a collection of previous datasets $\bigcup_{l=1}^{i} \mathcal{D}_l = \{u(t), y(t)\}_{t=1}^{iT}$, at time $k + 1$ new data $\mathcal{D}_{i+1}$ are provided and the aim is to update the previous estimate $\hat{g}^{(i)}$ by exploiting them. Furthermore, online applications typically require that the new estimate is available before the new dataset $\mathcal{D}_{i+2}$ is provided, thus limiting the computational complexity and the memory storage of the adopted estimation methods.

The interesting case of study is when the underlying system undergoes certain variations that have to be tracked: this situation could often arise in practice, as a consequence of variations in the physical system e.g. internal heating up, alteration of the masses (e.g. after grasping an object in a robotic platform), aging, weather conditions, etc.

For these reasons the time-varying model (4.1) is considered. However, time invariant systems, as a special case of the time-varying, are also considered.

4.2 Online Efficient Regularization Update

Consider the online setting outlined in Section 4.1. As pointed out there, identification procedures that are suitable for online applications need to be inexpensive in terms of both execution time and of memory storage. From the computational cost perspective, the critical step in NPPEM outlined in Section 2.4 is the marginal likelihood optimization in (2.48). Indeed, this step is typically performed by adopting iterative routines, such

as 1st or 2nd order optimization algorithms or the Expectation-Maximization (EM) algorithm. These methods may require a large number of iterations before reaching convergence, thus possibly making the estimation routine outlined in Section 2.4 too slow for being applied in an online setting. When applied to the marginal likelihood optimization problem (2.48), each iteration of these algorithms has a computational complexity of $O(n^3)$ due to the objective function evaluation.

In this regard, let us define for ease of notation, the so called "negative marginal log likelihood" from the definition in (2.56) as

$$f_N(\eta) := -\ln p_\eta(Y_N) = Y_N^\top \Sigma_Y(\eta)^{-1} Y_N + \ln \det \Sigma_Y(\eta) \tag{4.2}$$

$$\Sigma_Y(\eta) = \Phi_N K_\eta \Phi_N^\top + \sigma^2 I_N \tag{4.3}$$

where $\Phi_N \in \mathbb{R}^{N\times n}$:

$$\Phi_N := \begin{bmatrix} u(0) & u(-1) & \cdots & u(-n+1) \\ \vdots & \ddots & \ddots & \vdots \\ u(N) & u(N-1) & \cdots & u(N-n+1) \end{bmatrix} \tag{4.4}$$

and where the kernel is defined as

$$K_\eta = \lambda K_\beta, \quad \lambda \in \mathbb{R},\ \beta \in \mathbb{R}^{d_\eta - 1} \tag{4.5}$$

$$\Omega = \left\{ \eta = [\lambda, \beta] \in \mathbb{R}^{d_\eta} : \lambda \geq 0,\ 0 \leq \beta \leq 1 \right\} \tag{4.6}$$

The hyperparameters domain is kernel dependent and Ω in (4.6) is defined for the family of the stable splines kernels. However, the choice of the kernel is not restrictive for the theory developed in the reminder of the chapter. The estimation of the hyperparameters via maximization of the mariginal likelihood described in (2.48) is equivalent to:

$$\hat{\eta} = \arg\min_{\eta\in\Omega} -\ln p_\eta(Y_N) = \arg\min_{\eta\in\Omega} f_N(\eta) \tag{4.7}$$

The approach followed here, to adapt the "batch" technique described in Section 2.4 to the online framework, is the one proposed in Romeres, Prando, Pillonetto, and Chiuso (2016a): at time $k+T$, when new data $\mathcal{D}_{i+1} = \{u(t), y(t)\}_{t=iT+1}^{(i+1)T}$ are provided, the hyperparameters estimate $\hat{\eta}^{(i)}$ is set equal to the value returned after just one iteration of a 1st order optimization algorithm (or of the EM algorithm) applied to solve problem (4.7).

These iterative algorithms are initialized with the previous estimate $\hat{\eta}^{(i+1)}$ (obtained using the data $\bigcup_{l=1}^{i} \mathcal{D}_l$) which is likely to be close to a local optimum of the objective function $f_{iT}(\eta) \equiv f_k(\eta)$. If the number of new data $T << k$, it is reasonable to suppose that $\arg\min_{\eta\in\Omega} f_{iT}(\eta) \approx \arg\min_{\eta\in\Omega} f_{(i+1)T}(\eta)$. Therefore, by performing only one iteration of the chosen optimization algorithm, we expect $\hat{\eta}^{(i+1)}$ to be sufficiently close to a local optimum of $f_{(i+1)T}(\eta)$.

In other words, the algorithm that we are proposing can be seen as an online tracking of a local optimum point of the marginal likelihood that is changing when new data become available.

Let us define the following data matrices, useful for the formulation of the recursive procedure

$$R^{(i+1)} := \Phi_{(i+1)T}^{\top}\Phi_{(i+1)T} = R^{(i)} + \left(\Phi_{iT+1}^{(i+1)T}\right)^{\top} \Phi_{iT+1}^{(i+1)T} \tag{4.8}$$

$$\widetilde{Y}^{(i+1)} := \Phi_{(i+1)T}^{\top}Y_{(i+1)T} = \widetilde{Y}^{(i)} + \left(\Phi_{iT+1}^{(i+1)T}\right)^{\top} Y_{iT+1}^{(i+1)T} \tag{4.9}$$

$$\overline{Y}^{(i+1)} := Y_{(i+1)T}^{\top}Y_{(i+1)T} = \overline{Y}^{(i)} + \left(Y_{iT+1}^{(i+1)T}\right)^{\top} Y_{iT+1}^{(i+1)T} \tag{4.10}$$

where $Y_{(i+1)T} = [y(1) \ \cdots \ y((i+1)T)]^{\top} \in \mathbb{R}^{(i+1)T}$, while $Y_{iT+1}^{(i+1)T} = [y(iT+1) \ \cdots \ y((i+1)T)]$; Φ_i has been defined in (4.4) while $\Phi_{iT+1}^{(i+1)T}$ has the same structure of matrix (4.4) but it contains the data from $iT-n+1$ to $(i+1)T$, i.e.,

$$\Phi_{iT+1}^{(i+1)T} := \begin{bmatrix} u(iT+1) & u(iT) & \cdots & u(iT-n+1) \\ \vdots & \ddots & \ddots & \vdots \\ u((i+1)T) & u(i+1)T-1) & \cdots & u(i+1)T-n+1) \end{bmatrix} \tag{4.11}$$

The computational cost of (4.8)-(4.10) is, $O(n^2T)$, $O(nT)$ and $O(T^2)$, respectively.

Definition (4.2) shows that the computation of the negative marginal log likelihood depends on the number of data. However, in Chen and Ljung (2013) a robust way for computing (4.2) is shown, which in the online setting can be evaluated with computational complexity of order $O(n^3)$. That is:

$$f_{(i+1)T}(\eta) = ((i+1)T-n)\ln\sigma^2 + 2\ln|S| + \left(\frac{\overline{Y}^{(i+1)}}{\sigma^2} - \frac{\|S^{-1}L^{\top}\widetilde{Y}^{(i+1)}\|_2^2}{\sigma^2}\right) \tag{4.12}$$

where L and S are Cholesky factors, defined as

$$K_\eta =: LL^\top = \lambda L_\beta L_\beta^\top, \quad SS^\top := \sigma^2 I_n + L^\top R^{(i+1)} L \tag{4.13}$$

whose computation is $O(n^3)$. The definition of L_β, the Cholesky factor of the kernel without considering the scaling factor λ, has been added for future use.

Algorithm 2 summarizes the implementation of the whole procedure.

Algorithm 2 Online Bayesian System Identification

Inputs: previous estimates $\left\{\hat{\eta}^{(i)}, \hat{\eta}^{(i-1)}\right\}$, previous data matrices $\left\{R^{(i)}, \widetilde{Y}^{(i)}, \overline{Y}^{(i)}\right\}$, new data $\mathcal{D}_{i+1} = \{u(t), y(t)\}_{t=iT+1}^{(i+1)T}$

1: $R^{(i+1)} \leftarrow R^{(i)} + \left(\Phi_{iT+1}^{(i+1)T}\right)^\top \Phi_{iT+1}^{(i+1)T}$

2: $\widetilde{Y}^{(i+1)} \leftarrow \widetilde{Y}^{(i)} + \left(\Phi_{iT+1}^{(i+1)T}\right)^\top Y_{iT+1}^{(i+1)T}$

3: $\overline{Y}^{(i+1)} \leftarrow \overline{Y}^{(i)} + \left(Y_{iT+1}^{(i+1)T}\right)^\top Y_{iT+1}^{(i+1)T}$

4: $\widehat{g}_{LS}^{(i+1)} \leftarrow R^{(i+1)^{-1}} \widetilde{Y}^{(i+1)}$

5: $\hat{\sigma}^{(i+1)^2} \leftarrow \frac{1}{iT-n}\left(\bar{Y}^{(i+1)} - 2\widetilde{Y}^{(i+1)^\top}\widehat{g}_{LS}^{(i+1)} + \widehat{g}_{LS}^{(i+1)^\top} R^{(i+1)} \widehat{g}_{LS}^{(i+1)}\right)$

6: Compute $\hat{\eta}^{(i+1)}$ through 1-step Marginal Likelihood maximization initialized with $\hat{\eta}^{(i)}$ and $\hat{\eta}^{(i-1)}$

7: $\widehat{g}^{(i+1)} \leftarrow \left(R^{(i+1)} + \hat{\sigma}^{(i+1)^2} K_{\hat{\eta}^{(i+1)}}^{-1}\right)^{-1} \widetilde{Y}^{(i+1)}$

Output: $\widehat{g}^{(i+1)}$

The key step of the procedure outlined in Algorithm 2 is the hyperparameter estimation at step 6. Regarding the computational complexity of the remaining steps in Algorithm 2, the most demanding ones are Steps 4 and 7, which are both $O(n^3)$, because of the matrix inversion that has to be computed. If the new dataset $\mathcal{D}_{i+1}$ consists on only few ($<< n$) input-output pairs, then Shermann-Morrison formula can be exploited to compute $R^{(i+1)^{-1}}$ with a complexity of $O(n^2)$.

Furthermore, notice that the memory storage requirements of Algorithm 2 are $O(n^2)$, thanks to the updates at Steps 1-3.

4.2.1 1-Step Marginal Likelihood Maximization

Two different approaches are considered to solve problem (4.7), the 1st order optimization methods (also known as gradient methods) and the EM algorithm, which is suited to compute maximum likelihood solutions for models having latent variables. As anticipated in the previous section, only one iteration of these algorithms will be performed, in order

to address the requirement that the online setting imposes, Algorithm 2 step 6.

The approaches are now described.

Gradient Descent Methods

The gradient descent methods search for a local minimum of a function taking the steps proportional to the negative of the gradient function at the current point. A common approach to choose these steps is given by the Newton's method, where the step size is chosen proportional to the inverse of the Hessian of the function at the current point. Specifically, in our setup, Newton's update rule for the hyperparameters would be

$$\eta^{k+1} = \eta^k - \gamma \left(H(\eta^k) \right)^{-1} \nabla f(\eta^k) \tag{4.14}$$

where $H(\eta^k)$ and $\nabla f(\eta^k)$ are the Hessian and the gradient of $f(\eta^k)$, respectively, $\gamma \in (0, 1)$ is scalar to control the step length and k refers to the k-th iteration of the Newton's algorithm. This approach does not suit our framework because the Hessian matrix is computationally too expensive to obtain.

The algorithms considered belong to the family of Quasi-Newton methods (introduced in Davidon (1991); J. E. Dennis and Moré (1977)), where only an approximation of the inverse Hessian is computed.

Quasi-Newton methods approximate the Hessian by using only gradient information. Different algorithms can be derived according to the specific Hessian approximation that is chosen. They essentially differ in the way they attempt to satisfy the so-called *secant equation* (Nocedal and Wright, 2006, Chp. 6):

$$B^{(i)} w^{(i-1)} = r^{(i-1)} \tag{4.15}$$

where $B^{(i)}$ represents the approximation to the inverse Hessian computed at $\hat{\eta}^{(i)}$, while

$$r^{(i-1)} = \hat{\eta}^{(i)} - \hat{\eta}^{(i-1)}, \qquad w^{(i-1)} = \nabla f_{(i+1)T}(\hat{\eta}^{(i)}) - \nabla f_{(i+1)T}(\hat{\eta}^{(i-1)})$$

Recall for the following of this Chapter, that the superscript $^{(i)}$ refers to the value taken by a certain quantity after the datasets $\bigcup_{l=1}^{i} \mathcal{D}_l$ have been seen; it does not refer to the iteration number of the considered gradient method (since we are performing just one iteration).

The one-step implementation of a gradient method is summarized in Algorithm 3.

The update rule for $\hat{\eta}^{(i)}$ in Algorithm 3 Step 10 is a Quasi-Newton method and the approximation of inverse Hessian at Step 4 becomes crucial. According to this

Algorithm 3 1-step Gradient Method

Inputs: previous estimates $\left\{\hat{\eta}^{(i)}, \hat{\eta}^{(i-1)}\right\}$, $\nabla f_{iT}(\hat{\eta}^{(i-1)})$, $R^{(i+1)}$, $\widetilde{Y}^{(i+1)}$, $\overline{Y}^{(i+1)}$, $\hat{\sigma}^{(i+1)^2}$

Parameters initialization: $c = 10^{-4}$ and $\delta = 0.4$

1: Compute $\nabla f_{(i+1)T}(\hat{\eta}^{(i)})$

2: $r^{(i-1)} \leftarrow \hat{\eta}^{(i)} - \hat{\eta}^{(i-1)}$

3: $w^{(i-1)} \leftarrow \nabla f_{(i+1)T}(\hat{\eta}^{(i)}) - \nabla f_{(i+1)T}(\hat{\eta}^{(i-1)})$

4: Compute the inverse Hessian approximation $B^{(i)}$ using one among Algorithm 4,5,6

5: Project onto Ω: $z \leftarrow \Pi_{\Omega,W}\left(\hat{\eta}^{(i)} - B^{(i)}\nabla f_{(i+1)T}(\hat{\eta}^{(i)})\right)$

6: $\Delta\hat{\eta}^{(i)} \leftarrow z - \hat{\eta}^{(i)}$

7: $\gamma \leftarrow 1$

8: **while** $f_{(i+1)T}(\hat{\eta}^{(i)} + \gamma\Delta\hat{\eta}^{(i)}) \leq f_{(i+1)T}(\hat{\eta}^{(i)}) + c\gamma\nabla f_{(i+1)T}(\hat{\eta}^{(i)})^\top\Delta\hat{\eta}^{(i)}$ **do**

9: $\quad\gamma \leftarrow \delta\gamma$

10: $\hat{\eta}^{(i+1)} \leftarrow \hat{\eta}^{(i)} + \gamma\Delta\hat{\eta}^{(i)}$

Output: $\hat{\eta}^{(i+1)}$

approximation, the projection operator $\Pi_{\Omega,W}$ onto the feasible set Ω in Algorithm 3 Step 5 changes; namely, it is defined as:

$$\Pi_{\Omega,W}(z) = \arg\min_{x\in\Omega}(x-z)^\top W(x-z) \tag{4.16}$$

and the matrix W takes different values according to how $B^{(i)}$ is computed.

The loop in Step 8 ensures that the value of the cost function decreases at each iteration; these steps are called the Armijo bactracking loop, see (Bertsekas, 1995, Ch. 2).

The purpose of this work is not to define a new efficient approximation of the inverse Hessian, but to compare successful techniques adopted in literature in order to achieve high performance in the online system identification problem outlined in Section 4.1.

In the following three possible procedures to approximate the inverse Hessian are illustrated.

Barzilai-Borwein

This approach sets $B^{(i)} = \alpha^{(i)}I_{d_\eta}$, with d_η the dimension of the hyperparameters and $\alpha^{(i)} > 0$ is computed as a variant of Barzilai-Borwein (BB) rules, Barzilai and Borwein (1988), proposed in Bonettini, Chiuso, and Prato (2015) . In practice, the $\alpha^{(i)}$ are

determined as the solution of one of the problems:

$$\alpha_1^{(i)} := \arg\min_{\alpha} \|\alpha r^{(i-1)} - w^{(i-1)}\|^2 = \frac{r^{(i-1)^\top} r^{(i-1)}}{r^{(i-1)^\top} w^{(i-1)}} \tag{4.17}$$

$$\alpha_2^{(i)} := \arg\min_{\alpha} \|r^{(i-1)} - \alpha^{-1} w^{(i-1)}\|^2 = \frac{r^{(i-1)^\top} w^{(i-1)}}{w^{(i-1)^\top} w^{(i-1)}} \tag{4.18}$$

It is known by recent literature that the inverse Hessian is approximated more accurately by adaptively alternating the two solutions $\alpha_1^{(i)}$ and $\alpha_2^{(i)}$ and bounding them into a prefixed interval. Our implementation follows the one proposed in (Bonettini et al., 2015, Sec 4.1) and outlined in Algorithm 4.

In this case, the matrix W in the projection $\Pi_{\Omega,W}$ (4.16) is set equal to the identity matrix I_{d_η}.

Algorithm 4 Barzilai-Borwein Alternation Strategy

Inputs: $\tau^{(i)}, r^{(i-1)}, w^{(i-1)}$
Set $0 < \alpha_{min} < \alpha_{max}$
$\alpha_1 \leftarrow (r^{(i-1)^\top} r^{(i-1)})/(r^{(i-1)^\top} w^{(i-1)})$
$\alpha_2 \leftarrow (r^{(i-1)^\top} w^{(i-1)})/(w^{(i-1)^\top} w^{(i-1)})$
$\tilde{\alpha}_1 \leftarrow \min\{\max\{\alpha_{min}, \alpha_1\}, \alpha_{max}\}$
$\tilde{\alpha}_2 \leftarrow \min\{\max\{\alpha_{min}, \alpha_2\}, \alpha_{max}\}$
if $\tilde{\alpha}_2/\tilde{\alpha}_1 \leq \tau^{(i)}$ **then**
    $\alpha^{(i)} \leftarrow \tilde{\alpha}_2$
    $\tau^{(i+1)} \leftarrow 0.9\tau^{(i)}$
else
    $\alpha^{(i)} \leftarrow \tilde{\alpha}_1$
    $\tau^{(i+1)} \leftarrow 1.1\tau^{(i)}$
Outputs: $B^{(i)} = \alpha^{(i)} I_{d_\eta}$, $\tau^{(i+1)}$

In the first iteration of Algorithm 4 the scalar τ is initialized in the range $\tau^{(1)} \in (0, 1)$. This alternating strategy has been successfully applied in different convex, nonlinear, constrained problems in Barzilai and Borwein (1988); Bonettini, Zanella, and Zanni (2009); Bonettini and Prato (2010); Bonettini (2011) and for the maximization of the marginal likelihood in (4.7) in Bonettini et al. (2015).

Scaled Gradient Projection (SGP)

The inverse Hessian in this approach is approximated as:

$$B^{(i)} = \alpha^{(i)} D^{(i)}, \qquad \alpha^{(i)} > 0, \quad D^{(i)} \in \mathbb{R}^{d \times d} \tag{4.19}$$

The step-size $\alpha^{(i)}$ is set by alternating the Barzilai-Borwein rules and $D^{(i)}$ is a scaling matrix whose choice depends on the objective function and on the constraints set of the problem we are considering.

Our implementation follows the one proposed in (Bonettini et al., 2015, Sec. 4.2), where $D^{(i)}$ is a diagonal matrix, namely, $D^{(i)} = blockdiag(D_\lambda^{(i)}, D_\beta^{(i)})$ where $D_\lambda^{(i)} \in \mathbb{R}$ and $D_\beta^{(i)} \in \mathbb{R}^{(d-1)\times(d-1)}$ respectively denote the scaling matrices built for the two components of η.

The definition of matrix $D_\lambda^{(i)}$ in relation to the constraint $\lambda \geq 0$ is of interest also for future consideration and therefore it is briefly outlined. For the derivation of $D_\beta^{(i)}$ refer to (Bonettini et al., 2015, Sec. 4.2).

The definition of $D_\lambda^{(i)}$ relies on the gradient decomposition:

$$\begin{aligned} \nabla_\lambda f_{(i+1)T}(\eta) &= V_\lambda(\eta) - U_\lambda(\eta) \\ V_\lambda(\eta) &= \nabla_\lambda \ln |\Sigma_Y(\eta)| > 0 \\ U_\lambda(\eta) &= -\nabla_\lambda {Y^{(i+1)}}^\top \Sigma_Y(\eta) Y^{(i+1)} \geq 0 \end{aligned} \tag{4.20}$$

where ∇_λ denotes the gradient w.r.t. λ. Notice that the inequalities in (4.20) hold because of the positive semi-definiteness of K_β defined in (4.5) which occurs in the construction of $\Sigma_Y(\eta)$ (see (2.54)). In view of decomposition (4.20), the 1st order optimality conditions w.r.t. λ for problem (4.7),

$$\lambda \nabla_\lambda f_{(i+1)T}(\eta) = 0, \quad \lambda \geq 0, \quad \nabla_\lambda f_{(i+1)T}(\eta) \geq 0 \tag{4.21}$$

can be rewritten as the fixed point equation $\lambda = \lambda U_\lambda(\eta)/V_\lambda(\eta)$. By exploiting the fixed point update method, we can then define

$$D_\lambda^{(i)} = \min\{\max\{d_{min},\ \hat{\lambda}^{(i)}(V_\lambda(\hat{\eta}^{(i)}))^{-1}\},\ d_{max}\} \tag{4.22}$$

Refer to Bonettini et al. (2015) for a more detailed derivation.

Algorithm 5 summarizes how $B^{(i)}$ at Step 4 of Algorithm 3 is computed through SGP. In this case $\Pi_{\Omega,W}$ at Step 5 of Algorithm 3 is defined setting $W = {D^{(i)}}^{-1}$.

Algorithm 5 Scaled Gradient Projection Algorithm (SGP)

Inputs: $\nabla f_{(i+1)T}(\hat{\eta}^{(i)}),\ \tau^{(i)}, r^{(i-1)}, w^{(i-1)}$
Set the values of d_{min} and d_{max} such that $0 < d_{min} < d_{max}$
1: $V_\lambda(\hat{\eta}^{(i)}) \leftarrow \nabla_\lambda \ln\det(\Sigma_Y(\eta))\Big|_{\eta=\hat{\eta}^{(i)}}$
2: $U_\lambda(\hat{\eta}^{(i)}) \leftarrow -\nabla_\lambda {Y^{(i+1)}}^\top \Sigma_Y(\eta) Y^{(i+1)}\Big|_{\eta=\hat{\eta}^{(i)}}$
3: $D_\lambda^{(i)} \leftarrow \min\{\max\{d_{min},\ \hat{\lambda}^{(i)}(V_\lambda(\hat{\eta}^{(i)}))^{-1}\},\ d_{max}\}$
4: Compute $V_\beta(\hat{\eta}^{(i)}) > 0$ and $U_\beta(\hat{\eta}^{(i)}) > 0$ s.t. $\nabla_\beta f_{(i+1)T}(\hat{\eta}^{(i)}) = V_\beta(\hat{\eta}^{(i)}) - U_\beta(\hat{\eta}^{(i)})$
5: Compute $D_\beta^{(i)}$ as illustrated in (Bonettini et al., 2015, Sec. 4.2)
6: $D^{(i)} \leftarrow blockdiag(D_\lambda^{(i)}, D_\beta^{(i)})$
7: Run Algorithm 4 to compute $\alpha^{(i)},\ \tau^{(i+1)}$
Outputs: $B^{(i)} = \alpha^{(i)} D^{(i)},\ \tau^{(i+1)}$

BFGS

When adopting the inverse Hessian approximation provided by the BFGS algorithm, $B^{(i)}$ at step 4 of Algorithm 3 is computed as the unique solution of

$$\begin{aligned} &\min_B \|B - B^{(i-1)}\|_M \\ &\quad s.t.\ B = B^\top,\ B \succ 0, \\ &\quad Bw^{(i-1)} = r^{(i-1)} \end{aligned} \tag{4.23}$$

where $\|A\|_M = \|M^{1/2} A M^{1/2}\|_F$ denotes the weighted Frobenius norm, with M chosen such that $Mr^{(i-1)} = w^{(i-1)}$, see (Nocedal and Wright, 2006, Chp. 6). The unique solution $B^{(i)}$ to problem (4.23) is given by

$$B^{(i)} = \rho^{(i-1)} r^{(i-1)} {r^{(i-1)}}^\top + \left(I - \rho^{(i-1)} r^{(i-1)} {w^{(i-1)}}^\top\right) B^{(i-1)} \left(I - \rho^{(i-1)} w^{(i-1)} {r^{(i-1)}}^\top\right)$$

where $\rho^{(i-1)} = 1/\left({w^{(i-1)}}^\top r^{(i-1)}\right)$.
Algorithm 6 summarizes the implementation of BFGS. The projection operator $\Pi_{\Omega,W}$ is in this case defined with $W = I_{d_\eta}$.

Algorithm 6 BFGS

Inputs: $B^{(i-1)},\ r^{(i-1)},\ w^{(i-1)}$
1: $\rho \leftarrow 1/({w^{(i-1)}}^\top r^{(i-1)})$
2: $B^{(i)} \leftarrow \rho r^{(i-1)} {r^{(i-1)}}^\top + \left(I - \rho r^{(i-1)} {w^{(i-1)}}^\top\right) B^{(i-1)} \left(I - \rho w^{(i-1)} {r^{(i-1)}}^\top\right)$
Outputs: $B^{(i)}$

EM Algorithm

As already said in the beginning, another method to update $\hat{\eta}^{(i)}$ is the Expectation-Maximization (EM) algorithm which is used to compute maximum likelihood solutions for models having latent variables. Recall that at Step 6 of Algorithm 2 we need to compute $\hat{\eta}^{(i+1)}$ by maximizing

$$p(Y_{(i+1)T}|\eta) = \mathbb{E}_{p(g|\eta)}\left[p(Y_{(i+1)T}, g|\eta)\right]$$

where $\mathbb{E}_q$ denotes the expectation w.r.t. the probability distribution q. Hence, in our setting g plays the role of the latent variable. Consider the following decomposition (Bishop, 2006, Chp. 9):

$$\begin{aligned} \ln p(Y_{(i+1)T}|\eta) &= \mathcal{L}(q(g), \eta) + KL(q(g)||p(g|Y_{(i+1)T}, \eta)) \\ \mathcal{L}(q(g), \eta) &= \mathbb{E}_{q(g)}\left[\frac{p(Y_{(i+1)T}, g|\eta)}{q(g)}\right] \end{aligned} \tag{4.24}$$

where $\mathcal{L}(q, \eta)$ represents a lower bound for $\ln p(Y_{(i+1)T}|\eta)$, while $KL(\cdot||\cdot)$ denotes the Kullback-Leibler divergence between two probability distributions.

A standard EM algorithm finds the optimal value for η by alternating the Expectation (E) and the Maximization (M) steps until convergence is reached. The idea behind the algorithm is that, instead of maximizing the marginal likelihood, which in general is a nontrivial problem, it is possible to maximize the lower bound $\mathcal{L}(q(g), \eta)$ first w.r.t. $q(g)$, which corresponds to the E-step and then w.r.t. η which corresponds to the M-step. This algorithm is proven to converge to a local optimum of the marginal likelihood.

Specifically, the E-step arises from fixing $\hat{\eta}^{(i)}$ in $\mathcal{L}(q(g), \eta)$ and maximizing it w.r.t. $q(g)$. It is easy to see that

$$\mathcal{L}(p(g|Y_{(i+1)T}, \hat{\eta}^{(i)}), \eta) = \max_{q(g)} \mathcal{L}(q(g), \hat{\eta}^{(i)}) \tag{4.25}$$

since $KL(q(g)||p(g|Y_{(i+1)T}, \hat{\eta}^{(i)})) = 0$ when $q(g)$ equals the posterior distribution obtained with $\hat{\eta}^{(i)}$. It can be directly computed as

$$\mathcal{L}\left(p(g|Y_{(i+1)T}, \hat{\eta}^{(i)}), \eta\right) = \mathbb{E}_{p(g|Y_{(i+1)T}, \hat{\eta}^{(i)})}\left[\ln p(Y_{(i+1)T}, g|\eta) - \ln p(g|Y_{(i+1)T}, \hat{\eta}^{(i)})\right] \tag{4.26}$$

$$\begin{aligned} &= \mathbb{E}_{p(g|Y_{(i+1)T}, \hat{\eta}^{(i)})}\left[\ln p(Y_{(i+1)T}|g, \eta) + \ln p(g|\eta)\right] \\ &\quad - \mathbb{E}_{p(g|Y_{(i+1)T}, \hat{\eta}^{(i)})}\left[\ln p(g|Y_{(i+1)T}, \hat{\eta}^{(i)})\right] \end{aligned}$$

Recalling that $p(Y_{(i+1)T}|g,\eta) \sim \mathcal{N}(\Phi^{(i+1)}g, \sigma^2 I_{(i+1)T})$, using the prior $p(g|\eta)$ in (4.5) and assuming a non-informative prior on η, it results that

$$\begin{aligned}\mathcal{L}\left(p(g|Y_{(i+1)T}, \hat{\eta}^{(i)}), \eta\right) = &\frac{1}{2}\ln|P^{(i)}| + \frac{n}{2} - \frac{(i+1)T}{2}\ln\sigma^2 \\ &- \frac{1}{2\sigma^2}\left(\overline{Y}^{(i+1)} - 2\widetilde{Y}^{(i+1)^\top}\widehat{g}^{(i)} + \mathrm{tr}\{R^{(i+1)}P^{(i)}\} + \widehat{g}^{(i)^\top}R^{(i+1)}\widehat{g}^{(i)}\right) \\ &- \frac{1}{2}\left(\mathrm{tr}\{K_\eta^{-1}P^{(i)}\} + \widehat{g}^{(i)^\top}K_\eta^{-1}\widehat{g}^{(i)}\right) - \frac{1}{2}\ln|K_\eta| \end{aligned} \tag{4.27}$$

where $\widehat{g}^{(i)} = \frac{1}{\sigma^2}P^{(i)}\widetilde{Y}^{(i+1)}$ is the impulse response estimate and

$$P^{(i)} = (\sigma^{-2}R^{(i+1)} + K^{-1}_{\hat{\eta}^{(i)}})^{-1} \tag{4.28}$$

In the M-step of the EM algorithm the update of the hyperparameters is computed as:

$$\hat{\eta}^{(i+1)} = \arg\max_{\eta\in\Omega} \mathcal{L}(p(g|Y^{(i+1)}, \hat{\eta}^{(i)}), \eta) \tag{4.29}$$

According to our "1-step" approach, when EM is adopted at Step 6 of Algorithm 2, only one E-step and one M-step are performed. The 1-step EM algorithm is summarized in Algorithm 7.

Algorithm 7 EM

Inputs: $\hat{\eta}^{(i)}$, $R^{(i+1)}$, $\widetilde{Y}^{(i+1)}$, $\overline{Y}^{(i+1)}$, $\hat{\sigma}^{(i+1)^2}$

1: **E-step:** Compute $\mathcal{L}\left(p(g|Y^{(i+1)}, \hat{\eta}^{(i)}), \eta\right)$ as in (4.25)

2: **M-step:** $\hat{\eta}^{(i+1)} \leftarrow \arg\max_{\eta\in\Omega} \mathcal{L}(p(g|Y^{(i+1)}, \hat{\eta}^{(i)}), \eta)$

Outputs: $\hat{\eta}^{(i+1)}$

4.2.2 Connection with Existing Methodologies

In this section it is considered the case that hyperparameter β in (4.5) has been fixed to $\hat{\beta}$ and only the scaling factor λ has to be updated by the identification procedure.

Under this assumption, two connections of the EM algorithm can be verified:

1. the EM update rule coincides with a gradient-based update if a specific step-size is chosen,

2. the EM algorithm is equivalent to the iterative reweighted methods, which have been introduced for compressed sensing applications Candes, Wakin, and Boyd (2008); Chartrand and Yin (2008).

Connection between EM and Gradient Methods

Consider the EM update in (4.29) and assume the Kernel is $K_\eta = \lambda K_{\hat{\beta}}$ (i.e., β is fixed). Then, the optimization problem (4.29) can be reformulated as

$$\begin{aligned}\hat{\lambda}_{EM}^{(i+1)} &= \arg\max_{\lambda\in\mathbb{R}^+} \ \mathcal{L}(p(g|Y^{(i+1)}, \hat{\eta}^{(i)}), \lambda) \\ &= \arg\max_{\lambda\in\mathbb{R}^+} \ -\ln|\lambda K_{\hat{\beta}}| - \frac{1}{\lambda}\mathrm{tr}\{K_{\hat{\beta}}^{-1}P^{(i)}\} - \widehat{g}^{(i)\top}K_{\hat{\beta}}^{-1}\widehat{g}^{(i)}\end{aligned} \tag{4.30}$$

from which by imposing that $\frac{d}{d\lambda}\mathcal{L}(p(g|Y^{(i+1)}, \hat{\eta}^{(i)}), \lambda) = 0$ the EM update can be computed as

$$\hat{\lambda}_{EM}^{(i+1)} = \frac{1}{n}\left[\widehat{g}^{(i)\top}K_{\hat{\beta}}^{-1}\widehat{g}^{(i)} + \mathrm{tr}\left\{(K_{\hat{\beta}})^{-1}P^{(i)}\right\}\right] \tag{4.31}$$

Notice that the first term in the update rule (4.31) corresponds to the current approximation of the value of λ which asymptotically maximizes the Marginal Likelihood, i.e. $\hat{\lambda}^* = \frac{1}{n}g^\top K_{\hat{\beta}}^{-1}g$, with g denoting the true impulse response Aravkin et al. (2012). Whereas, the second term in (4.31) accounts for the uncertainty in the λ estimate, due to the use of a finite amount of data.

Consider now the gradient update rule for $\hat{\lambda}^{(i+1)}$ to solve problem (4.7)

$$\hat{\lambda}_{GR}^{(i+1)} = \hat{\lambda}^{(i)} - \alpha_\lambda^{(i)}\nabla_\lambda f_{(i+1)T}(\hat{\lambda}^{(i)}) \tag{4.32}$$

Since the quantity to optimize is a scalar the step-size is also a scalar, namely $\alpha_\lambda^{(i)}$.

The following result can now be proven.

Lemma 4.2.1. *Consider the optimization problem (4.7) where $\eta = \lambda$, that is, the only hyperparameter to estimate is the scaling factor of the Kernel $K_\eta = \lambda K_{\hat{\beta}}$, and the two update rules (4.31) and (4.32). Then, the following result holds.*

If $\alpha_\lambda^{(i)} = \frac{(\hat{\lambda}^{(i)})^2}{n}$ in (4.32), then

$$\hat{\lambda}_{GR}^{(i+1)} = \hat{\lambda}_{EM}^{(i+1)} \tag{4.33}$$

Proof: We want to show that computing explicitly $f_{(i+1)T}(\hat{\lambda}^{(i)})$ the gradient update equals the EM update (4.32) (4.31).

Letting $\eta = \lambda$ and fixing β to $\hat{\beta}$ function (4.2) can be rewritten as:

$$f_{(i+1)T}(\lambda) = Y_{(i+1)T}^\top \left(\Phi_{(i+1)T}\lambda K_{\hat{\beta}}\Phi_{(i+1)T}^\top + \sigma^2 I_{(i+1)T}\right)^{-1} Y_{(i+1)T} \tag{4.34}$$

$$+ \ln\det\left(\Phi_{(i+1)T}\lambda K_{\hat{\beta}}\Phi_{(i+1)T}^\top + \sigma^2 I_{(i+1)T}\right) \tag{4.35}$$

Using the Woodbury matrix identity and the definition of $P^{(i)}$ in (4.28)

$$\begin{aligned}\left(\Phi_{(i+1)T}\lambda K_\beta\Phi_{(i+1)T}^\top + \sigma^2 I_{(i+1)T}\right)^{-1} &= \frac{I_{(i+1)T}}{\sigma^2} - \frac{\Phi_{(i+1)T}}{\sigma^2}\left((\lambda K_\beta)^{-1} + \frac{\Phi_{(i+1)T}^\top\Phi_{(i+1)T}}{\sigma^2}\right)^{-1}\frac{\Phi_{(i+1)T}^\top}{\sigma^2} \\ &= \frac{I_{(i+1)T}}{\sigma^2} - \frac{1}{\sigma^4}\Phi_{(i+1)T}P^{(i)}\Phi_{(i+1)T}^\top\end{aligned}$$

the first term in the marginal likelihood can be rewritten as

$$Y_{(i+1)T}^\top \left(\Phi_{(i+1)T}\lambda K_{\hat{\beta}}\Phi_{(i+1)T}^\top + \sigma^2 I_{(i+1)T}\right)^{-1} Y_{(i+1)T} = \frac{\overline{Y}^{(i+1)}}{\sigma^2} - \frac{\widetilde{Y}^{(i+1)^\top}}{\sigma^2}P^{(i)}\frac{\widetilde{Y}^{(i+1)}}{\sigma^2}$$

The gradient of $f_{(i+1)T}(\hat{\lambda}^{(i)})$ w.r.t. λ can be rewritten as

$$\begin{aligned}\nabla_\lambda f_{(i+1)T}(\lambda) &= -\frac{\widetilde{Y}^{(i+1)^\top}}{\sigma^2}\frac{\partial P^{(i)}}{\partial\lambda}\frac{\widetilde{Y}^{(i+1)}}{\sigma^2} + \mathrm{Tr}\left\{\left(\Phi_{(i+1)T}\lambda K_\beta\Phi_{(i+1)T}^\top + \sigma^2 I\right)^{-1}\Phi_{(i+1)T}K_\beta\Phi_{(i+1)T}^\top\right\} \\ &= \frac{\widetilde{Y}^{(i+1)^\top}}{\sigma^2}P^{(i)}\left(-\frac{K_\beta^{-1}}{\lambda^2}\right)P^{(i)}\frac{\widetilde{Y}^{(i+1)}}{\sigma^2} + \mathrm{Tr}\left\{\left(\Phi_{(i+1)T}\lambda K_\beta\Phi_{(i+1)T}^\top + \sigma^2 I\right)^{-1}\Phi_{(i+1)T}K_\beta\Phi_{(i+1)T}^\top\right\} \\ &= -\frac{1}{\lambda^2}\frac{\widetilde{Y}^{(i+1)^\top}P^{(i)}}{\sigma^2}K_\beta^{-1}\frac{P^{(i)}\widetilde{Y}^{(i+1)}}{\sigma^2} + \mathrm{Tr}\left\{\left(\Phi_{(i+1)T}\lambda K_\beta\Phi_{(i+1)T}^\top + \sigma^2 I\right)^{-1}\Phi_{(i+1)T}K_\beta\Phi_{(i+1)T}^\top\right\}\end{aligned}$$

The second term in the gradient can be rewritten by means of the inversion lemma:

$$
\begin{aligned}
&\mathrm{Tr}\left\{\left(\Phi_{(i+1)T}\lambda K_\beta \Phi_{(i+1)T}^\top + \sigma^2 I\right)^{-1}\Phi_{(i+1)T}K_\beta\Phi_{(i+1)T}^\top\right\}\\
&\quad=\mathrm{Tr}\left\{\left(\frac{I}{\sigma^2}-\frac{1}{\sigma^4}\Phi_{(i+1)T}P^{(i)}\Phi_{(i+1)T}^\top\right)\Phi_{(i+1)T}K_\beta\Phi_{(i+1)T}^\top\right\}\\
&\quad=\mathrm{Tr}\left\{\frac{\Phi_{(i+1)T}}{\sigma^2}P^{(i)}\left[\left((\lambda K_\beta)^{-1}+\frac{\Phi_{(i+1)T}^\top\Phi}{\sigma^2}\right)K_\beta\Phi_{(i+1)T}^\top-\frac{\Phi_{(i+1)T}^\top\Phi_{(i+1)T}K_\beta\Phi_{(i+1)T}^\top}{\sigma^2}\right]\right\}\\
&\quad=\mathrm{Tr}\left\{\frac{\Phi_{(i+1)T}}{\sigma^2}P^{(i)}\frac{\Phi_{(i+1)T}^\top}{\lambda}\right\}\\
&\quad=\frac{1}{\lambda}\mathrm{Tr}\left\{\frac{\Phi_{(i+1)T}^\top\Phi}{\sigma^2}\left(\frac{\Phi_{(i+1)T}^\top\Phi}{\sigma^2}+(\lambda K_\beta)^{-1}\right)^{-1}\right\}\\
&\quad=\frac{1}{\lambda}\mathrm{Tr}\left\{\left(\frac{\Phi_{(i+1)T}^\top\Phi_{(i+1)T}}{\sigma^2}+(\lambda K_\beta)^{-1}-(\lambda K_\beta)^{-1}\right)\left(\frac{\Phi_{(i+1)T}^\top\Phi_{(i+1)T}}{\sigma^2}+(\lambda K_\beta)^{-1}\right)^{-1}\right\}\\
&\quad=\frac{1}{\lambda}\mathrm{Tr}\left\{\left(\frac{\Phi_{(i+1)T}^\top\Phi_{(i+1)T}}{\sigma^2}+(\lambda K_\beta)^{-1}\right)\left(\frac{{}^\top\Phi_{(i+1)T}}{\sigma^2}+(\lambda K_\beta)^{-1}\right)^{-1}-(\lambda K_\beta)^{-1}P^{(i)}\right\}\\
&\quad=\frac{n}{\lambda}-\frac{1}{\lambda^2}\mathrm{Tr}\left\{K_\beta^{-1}P^{(i)}\right\}
\end{aligned}
$$

Eventually, the gradient function of $f_{(i+1)T}(\lambda)$ w.r.t. λ evaluated in $\lambda=\hat{\lambda}^{(i)}$ is

$$\nabla_\lambda f_{(i+1)T}(\hat{\lambda}^{(i)})=n(\hat{\lambda}^{(i)})^{-1}-(\hat{\lambda}^{(i)})^{-2}\mathrm{Tr}\{K_{\hat{\beta}}^{-1}P^{(i)}\}-(\sigma^2\hat{\lambda}^{(i)})^{-2}\widetilde{Y}^{(i+1)^\top}P^{(i)}K_{\hat{\beta}}^{-1}P^{(i)}\widetilde{Y}^{(i+1)}$$

Replacing this expression and $\alpha_\lambda^{(i)}=\frac{(\hat{\lambda}^{(i)})^2}{n}$ into (4.32) the equality (4.33) is proven. □

Connection between EM and Iterative Reweighted Methods

Iterative reweighted methods have been recently introduced in the compressive sensing field in order to improve the recovery of sparse solutions. The focus of this section is on the ℓ_2-reweighted scheme that has been proposed in Wipf and Nagarajan (2010) for Sparse Bayesian Learning (SBL) Tipping (2001).

Consider the optimization problem (4.7) that in the current setting (i.e., with β fixed) results in

$$\min_{\lambda\geq 0}-\ln p(Y_{(i+1)T}|\lambda)=\min_{\lambda\geq 0}Y_{(i+1)T}^\top\Sigma_Y(\lambda)^{-1}Y_{(i+1)T}+\ln\det\Sigma_Y(\lambda)$$

In (Tipping, 2001, App. A) it has been shown that

$$Y_{(i+1)T}^{\top}\Sigma_Y(\lambda)^{-1}Y_{(i+1)T} = \min_g \frac{1}{\sigma^2}\|Y_{(i+1)T} - \Phi_{(i+1)T}\, g\|_2^2 + g^{\top}(\lambda K_{\hat{\beta}})^{-1}g$$

Thus, the minimization problem becomes

$$\begin{aligned}\min_{\lambda\geq 0} -\ln p(Y_{(i+1)T}|\lambda) &= \min_{\lambda\geq 0, g} \frac{1}{\sigma^2}\|Y_{(i+1)T} - \Phi_{(i+1)T} g\|_2^2 + g^{\top}(\lambda K_{\hat{\beta}})^{-1}g + \ln\det\Sigma_Y(\lambda) \\ &= \min_g \frac{1}{\sigma^2}\|Y_{(i+1)T} - \Phi_{(i+1)T}\, g\|_2^2 + \zeta(g)\end{aligned}$$

where

$$\zeta(g) = \min_{\lambda\geq 0} g^{\top}(\lambda K_{\hat{\beta}})^{-1}g + \ln\det\Sigma_Y(\lambda)$$

is a non-separable penalty function, since it can not be expressed as a summation over functions of the individual impulse response coefficients $g(j)$ with $j = [1, n]$. Furthermore, it is a non-decreasing concave function of $g^2 := [g(1)^2 \ \cdots \ g(n)^2]^{\top}$, thus allowing to employ iterative reweighted ℓ_2 schemes to minimize the function above. Namely,

$$\begin{aligned}\zeta(g) &\leq g^{\top}(\lambda K_{\hat{\beta}})^{-1}g + \ln\det\Sigma_Y(\lambda) \\ &= g^{\top}(\lambda K_{\hat{\beta}})^{-1}g + \ln\det(\lambda K_{\hat{\beta}}) + \ln\det\left(\frac{\Phi_{(i+1)T}^{\top}\Phi_{(i+1)T}}{\sigma^2} + (\lambda K_{\hat{\beta}})^{-1}\right) + (i+1)T\ln\sigma^2 \qquad (4.36) \\ &\leq g^{\top}(\lambda K_{\hat{\beta}})^{-1}g + \ln\det(\lambda K_{\hat{\beta}}) + z\lambda^{-1} - s^*(z) + (i+1)T\ln\sigma^2 \qquad (4.37)\end{aligned}$$

where $s^*(z)$ denotes the concave conjugate of $s(a) := \ln\det\left(\frac{\Phi_{(i+1)T}^{\top}\Phi_{(i+1)T}}{\sigma^2} + aK_{\hat{\beta}}^{-1}\right)$, $a = \lambda^{-1}$, given by:

$$s^*(z) = \min_a za - \ln\det\left(\frac{\Phi_{(i+1)T}^{\top}\Phi_{(i+1)T}}{\sigma^2} + aK_{\hat{\beta}}^{-1}\right), \ a = \lambda^{-1}$$

Notice that in (4.36) the Silvester's determinant identity is used and the bound (4.37) holds for all $z, \lambda \geq 0$.

Hence, it follows

$$\min_{\lambda\geq 0} -\ln p(Y_{(i+1)T}|\lambda) =$$
$$= \min_{\lambda\geq 0, z\geq 0, g} \frac{1}{\sigma^2}\|Y_{(i+1)T} - \Phi_{(i+1)T} g\|_2^2 + g^\top (\lambda K_{\hat{\beta}})^{-1} g + \ln\det(\lambda K_{\hat{\beta}}) + z\lambda^{-1} - s^*(z) \tag{4.38}$$

where the irrelevant terms for the optimization problem have been omitted.

Now that the minimization problem in the reweighted ℓ_2 scheme has been defined, the analogies with the two steps of the EM algorithm can be stated. Specifically, recall that the E-step in the EM is equivalent to solving problem (4.25) whose solution is given by the posterior distribution of g given $\hat{\lambda}^{(i)}$, i.e. $p(g|Y_{(i+1)T}, \hat{\lambda}^{(i)})$. Analogously, solving (4.38) w.r.t. g for fixed $\hat{\lambda}^{(i)}$ leads to an a-posteriori estimate, namely the Empirical Bayes estimator $\widehat{g}^{(i+1)} = \mathbb{E}[g|Y_{(i+1)T}, \hat{\lambda}^{(i)}]$, which coincides with the Maximum a Posteriori estimator of g.

On the other hand, solving (4.38) for fixed $\widehat{g}^{(i)}$ leads to

$$\hat{\lambda}^{(i+1)} = \frac{1}{n}\left(\widehat{g}^{(i)\top} K_{\hat{\beta}}^{-1} \widehat{g}^{(i)} + z^*\right) \tag{4.39}$$

where Wipf and Nagarajan (2010)

$$z^* = \frac{\partial}{\partial a} \ln\det\left(\frac{\Phi_{(i+1)T}^\top \Phi_{(i+1)T}}{\sigma^2} + aK_{\hat{\beta}}^{-1}\right) = \mathrm{Tr}\left\{P^{(i)} K_{\hat{\beta}}^{-1}\right\}$$

Thus, the update (4.39) coincides with the M-step in (4.29).

4.2.3 Simulations with Time Invariant Dynamical Systems

The purpose of this section is to evaluate some preliminary performance of Algorithm 2 in a set-up easier than the one proposed in Section 4.1 because time invariant systems (a particular case of time-varying systems) are considered. Moreover, it is aimed to show which of the methods proposed in Section 4.2.1 to compute the unique step in the marginal likelihood optimization outperforms the others.

Data

The experiment consisted of 200 Monte Carlo runs, in each of them a random SISO discrete-time system has been generated through the Matlab routine `drmodel.m`. The system orders have been randomly chosen in the range $[5, 10]$, while the systems poles

are all inside a circle of radius 0.95.

The input signal is a unit variance band-limited Gaussian signal with normalized band $[0, 0.8]$. A zero mean white Gaussian noise, with variance adjusted so that the Signal to Noise Ratio (SNR) is always equal to 5, has been added to the output data. For each Monte Carlo run, a data set of $N = 5000$ input-output pairs has been generated, while the length of the online upcoming datasets $\mathcal{D}_i$ has been chosen to be $T = 10$.

Estimators

The procedures that perform only one iteration of the iterative algorithms SGP, BB, BFGS and EM (illustrated in Algorithms 4-7)), are also compared to the standard iterative algorithm which estimates the hyperparameters running the optimization algorithm until convergence. In the following, the former procedures will be denoted 1-STEP, while we will refer to the latter one as OPT. The OPT procedure exploits the SGP algorithm to maximize the Marginal Likelihood.

The OPT procedure corresponds to the so called "batch" procedure equipped with an ad-hoc initialization of the optimization problem (4.7) provided by the previous hyperparameters estimate of the online procedure SGP and with the recursive update of the data depending matrices, see Algorithm 2 steps 1-3 to reduce the computational time.

In the experiments, a zero-mean Gaussian prior with a covariance matrix given by the so-called TC-kernel, see Chen et al. (2012), is adopted:

$$K_{\eta}^{TC}(k, j) = \lambda \min(\beta^k, \beta^j) \tag{4.40}$$

where $\lambda \geq 0$ and $0 \leq \beta \leq 1$ are the hyperparameters collected in $\eta = [\lambda,\ \beta]$. The length n of the estimated impulse responses has been set to 80.

In the interest of exploring the solutions with higher computational time performance of the online updates, two versions of BFGS, SGP, BB, EM are proposed.

- Update λ and β. Both the hyperparameters in η are updated whenever a new dataset $\mathcal{D}_i$ becomes available.
- Update only λ. Only the scaling factor λ is updated, retaining β fixed to its initial value. This methodology reflects the framework where the theoretical results in Section 4.2.2 has been achieved.

It is clear that the second case allows a faster computation, at the expenses of a less precise impulse response estimator.

In addition, two cases of the EM version in which only λ is updated are considered:

- $EM1$, where $\widehat{\lambda}^{(i+1)} = \frac{1}{n}\widehat{g}^{(i)\top} K_{\hat{\beta}}^{-1}\widehat{g}^{(i)}$, which is the current approximation of the asymptotically optimal value.

- $EM2$, where the update corresponds to (4.31).

The aim is to show a comparison between the asymptotic theory and the EM update, see e.g. Bottegal, Aravkin, Hjalmarsson, and Pillonetto (2014); notice that the second term of (4.31) tends to zero when the number of data tends to infinity.

Performance

As a first comparison, the adherence of the impulse response estimate to the true one is evaluated. Thus, for each estimated system and for each procedure the impulse response fit is performed:

$$\mathcal{F}(\widehat{g}) = 100 \cdot \left(1 - \frac{\|g - \widehat{g}\|_2}{\|g\|_2}\right) \tag{4.41}$$

where g, $\widehat{g}$ are the true and the estimated impulse responses of the considered system, respectively.

Figure 4.1 shows the impulse response fits (4.41) achieved in the Monte-Carlo simulations along with the increase of the number of observed data. OPT procedure is compared with the 1-STEP SGP, BB, BFGS and EM. On the left hand side the obtained results optimizing both hyperparameters in η are reported, while the results on the right hand side are obtained by updating only λ.

All the 1-STEP procedures which update both hyperparameters perform remarkably well, with the fit index being almost equivalent to the one obtained with the OPT procedure. This suggests that the full optimization of problem (4.7) does not bring any particular advantage in terms of fit in the online setting. Notice that we are taking a sort of worst case approximation since we stop the optimization algorithm after only 1 step: some more advanced techniques could be considered (e.g. an early stopping criterion Yao, Rosasco, and Caponnetto (2007)). The 1-STEP updates optimizing only λ, as expected, perform worse than the other update technique, having a bigger variance and slightly inferior performance in terms of median. However, their behaviour is comparable to the one when both hyperparameters are updated, therefore depending on the application this technique can be taken in consideration. The only exception is represented by EM1 which achieves inferior fits, but it is expected that also this update reaches the same performance when the number of data tends to infinity.

The second comparison is done in terms of cumulative computational time of the procedures, see Figure 4.2 and Table 4.1.

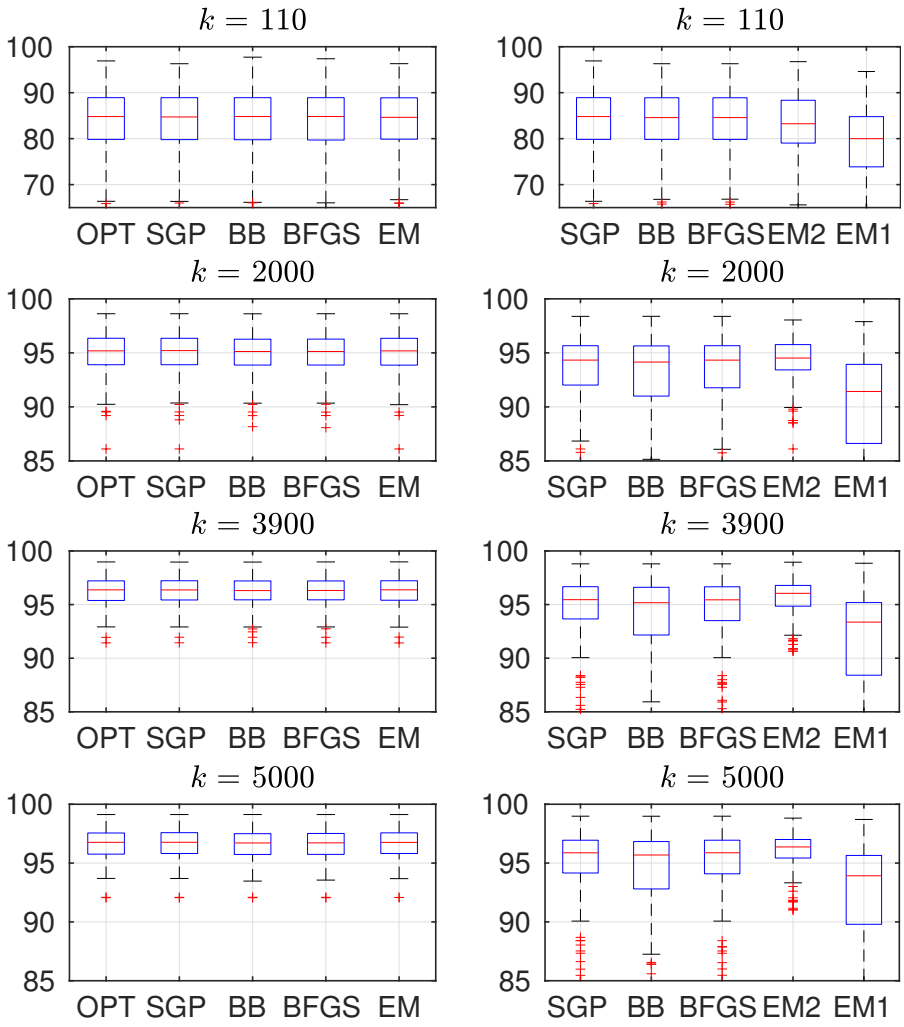

Figure 4.1: Monte Carlo results. *Left:* Boxplots of the impulse response fit obtained updating both hyperparameters in η. *Right*: Boxplots of the impulse response fit obtained updating only λ.

The OPT procedure, as expected, is much slower than the 1-STEP procedures. This

	Update λ and β					Update only λ				
	OPT	SGP	BB	BFGS	EM	SGP	BB	BFGS	EM2	EM1
mean	163.1	0.56	0.93	1.19	0.57	0.31	0.60	0.45	0.18	0.30
std	18.45	0.13	0.16	0.36	0.11	0.06	0.13	0.25	0.06	0.92

Table 4.1: MC results. Mean and standard deviation (std) of the cumulative computational time after $N = 5000$ data have been used.

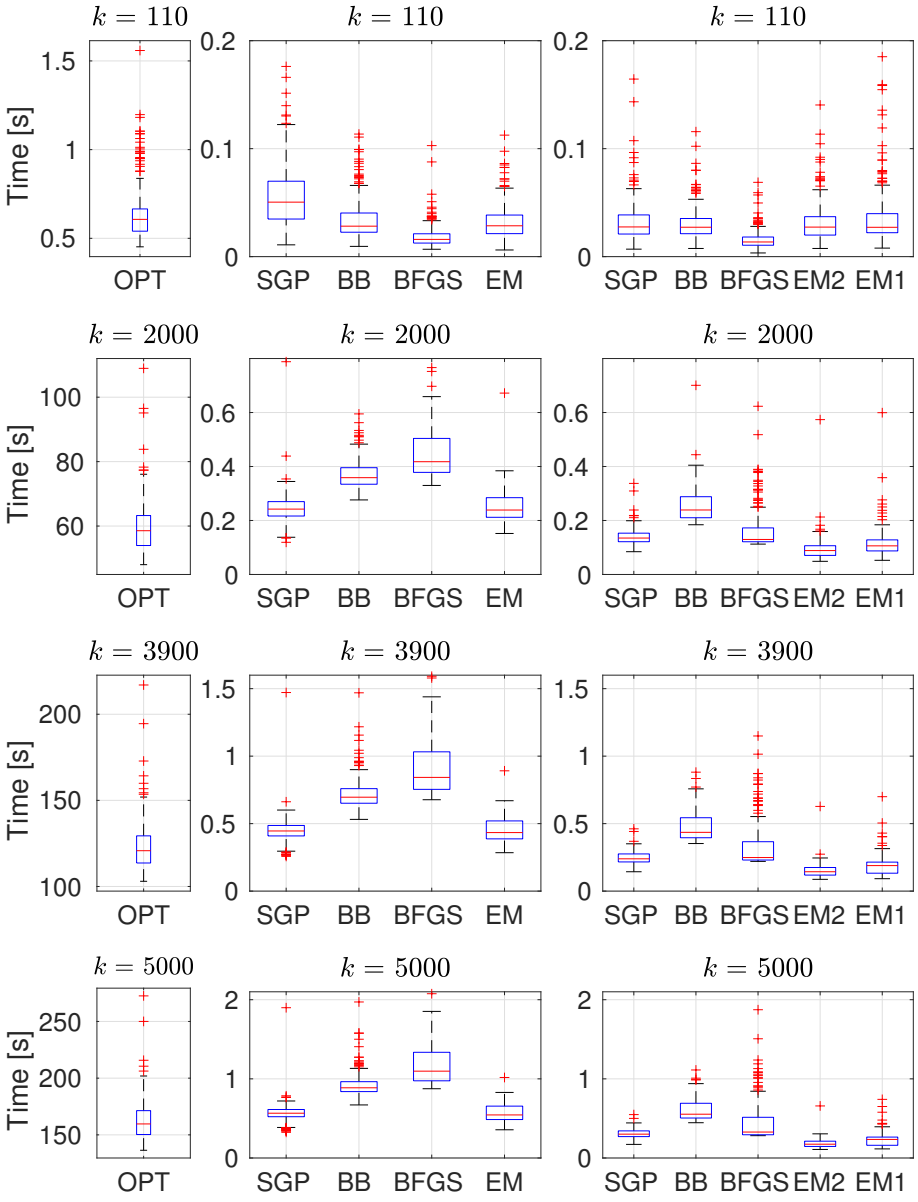

Figure 4.2: Monte Carlo results. Boxplots of the cumulative computational time. Each row of plots corresponds to the situation after T data are viewed. *Left:* OPT procedure. *Mid:* 1-STEP optimization of both hyperparameters. *Right:* 1-STEP optimization only of λ (β is fixed).

could suggest that the 1-STEP procedures we consider appear to be excellent candidates for real-time applications. Indeed, these techniques perform comparably in terms of fit w.r.t. the OPT procedure, but demanding a computational time which is two or three order of magnitude faster; furthermore the difference in terms of computational time

diverges in favour of the 1-STEP procedure with the increase of the number of data seen. Among the 1-STEP procedures SGP and EM provide the fastest updates: this is surprisingly positive for the EM update since only λ has a closed form update, while β is the solution of a maximization problem; indeed, in the right hand side of Figure 4.2, where only λ is updated, EM1 and EM2 outperform SGP. The update BB is a particular case of SGP, where $D^{(i)} = I$ (see Section 4.2.2), but it is significantly slower: this is due to the backtracking loop at Step 8 in Algorithm 3. The right hand side of Figure 4.2 shows the advantage of updating only λ: the cumulative computational time is inferior.

Finally, Figure 4.3 reports the evolution of the fit and of the hyperparameters estimates, for a single system, when new datasets of different lengths arrive. In this experiment, datasets $\mathcal{D}_i$ of lengths $T = 1, 10, 50$ are considered. It is of interest to observe that in terms of both fit and hyperparameters update, the performance of the 1-step techniques match closely the performance of the OPT procedure. The graph is cut after 3000 data to highlight the transitory behaviour. As expected, the transitory is longer and more accentuated in the case of $T = 50$, particularly in the behaviour of λ. However, this does not affect the behaviour of the fit performance significantly.

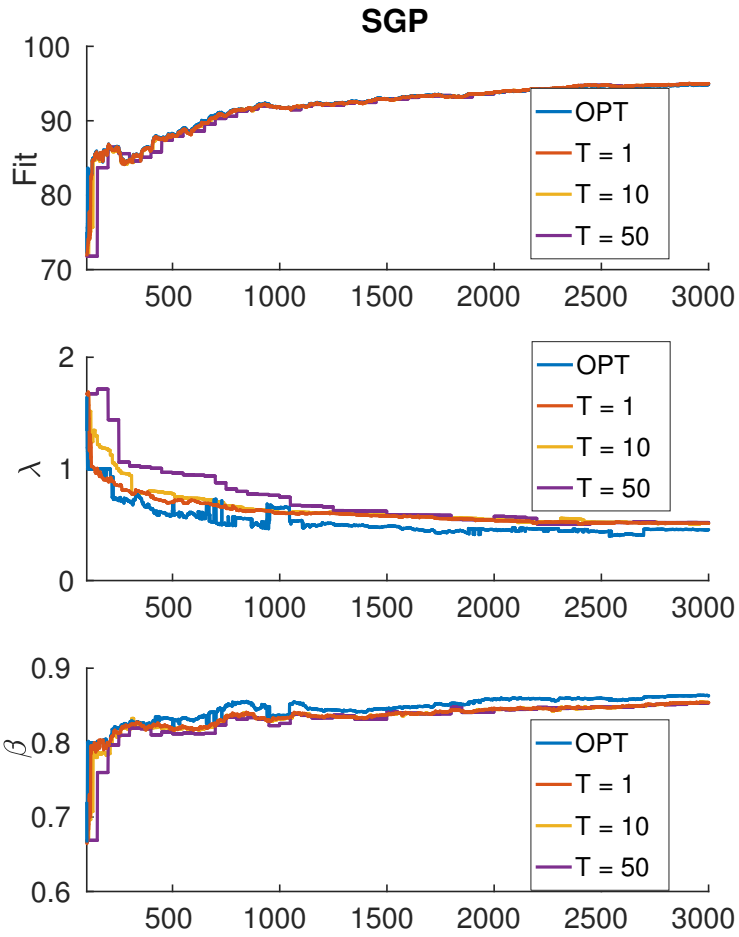

Figure 4.3: Comparison of OPT and a 1-STEP SGP update with different length T of the dataset $\mathcal{D}_i$ in the online identification of one system.

4.3 Time-Varying Dynamical Systems

The methodology proposed in Section 4.2 and validated for time invariant dynamical systems in Section 4.2.3 is here extended to time-varying dynamical systems.

The NPPEM methods have good flexibility properties in adapting to the data, as discussed in Section 2.5 and experimented in Chapter 3 and thanks to the update rules proposed in Section 4.2 they have also an efficient way to include into the estimators the information of the new data in the online scenario. All of these feature are fundamental to cope with time-varying dynamical systems. In this section it is added to NPPEM the ability to "forget" past data that would deteriorate the quality of the estimators due to the time variance of the system.

In the following, three routines are presented which combine the "online Gaussian

regression estimation" proposed in Section 4.2 with tools through which past data are disregarded or become less relevant to the current estimation.

4.3.1 Fixed Forgetting Factor

Following a classical practice in parametric system identification (see Sec. 2.3.2), a forgetting factor $\bar{\gamma} \in]0, 1]$ is applied to the available data, in order to base the estimation mainly on the more recent data. Accordingly, the first k data are generated from the following linear model:

$$\bar{Q}_k Y_k = \bar{Q}_k \Phi_k g + E \tag{4.42}$$

where $\bar{Q}_k \bar{Q}_k =: \bar{\Gamma}_k$ and $\bar{\Gamma}_k := diag\left(\bar{\gamma}^{k-1}, \bar{\gamma}^{k-2}, ..., \bar{\gamma}^0\right)$ and $E = [e(1)...e(k)]^\top \sim \mathcal{N}(Y_k|0, \sigma^2 I_k)$. Consequently, the Gaussian regression estimate at time k is obtained by adapting the regularized regression criterion (2.58) as:

$$\begin{aligned} \widehat{g}_{\bar{\gamma}} &:= \underset{g\in\mathbb{R}^n}{\arg\min} \sum_{t=1}^{k} \bar{\gamma}^{k-t} \left(y(t) - \Phi_t^t g\right)^2 + \sigma^2 g^\top K_{\hat{\eta}}^{-1} g \qquad (4.43) \\ &= \underset{g\in\mathbb{R}^n}{\arg\min} \left(Y_k - \Phi_k g\right)^\top \bar{\Gamma}_k \left(Y_k - \Phi_k g\right) + \sigma^2 g^\top K_{\hat{\eta}}^{-1} g \\ &= (\Phi_k^\top \bar{\Gamma}_k \Phi_k + \sigma^2 K_{\hat{\eta}}^{-1})^{-1} \Phi_k^\top \bar{\Gamma}_k Y_k \qquad (4.44) \end{aligned}$$

and estimating the hyperparameters by solving:

$$\hat{\eta}^{(i)} = \underset{\eta\in\Omega}{\arg\min}\ Y_k^\top \bar{Q}_k \Sigma_Y^{\bar{\gamma}}(\eta)^{-1} \bar{Q}_k Y_k + \ln\det \Sigma_Y^{\bar{\gamma}}(\eta) \tag{4.45}$$

$$\Sigma_Y^{\bar{\gamma}}(\eta) = \bar{Q}_k \Phi_k K_\eta \Phi_k^\top \bar{Q}_k + \sigma^2 I_k \tag{4.46}$$

Algorithm 8 illustrates the online implementation of the identification procedure based on equations (4.44) and (4.45). In particular, it assumes that at time k the estimates $\widehat{g}^{(i)}$ and $\hat{\eta}^{(i)}$ are the solutions of (4.43) and (4.45), respectively. These estimates are then updated online when the new dataset $\mathcal{D}_{i+1}$ is provided. Once $\bar{\gamma}$ is chosen by the user, it is inserted in the data matrices

$$R_{\bar{\gamma}}^{(i+1)} := \Phi_{(i+1)T}^{\top} \bar{\Gamma}_{(i+1)T} \Phi_{(i+1)T}, \tag{4.47}$$

$$\widetilde{Y}_{\bar{\gamma}^{(i+1)}} := \Phi_{(i+1)T}^{\top} \bar{\Gamma}_{(i+1)T} Y_{(i+1)T}, \tag{4.48}$$

$$\overline{Y}_{\bar{\gamma}^{(i+1)}} := Y_{(i+1)T}^{\top} \bar{\Gamma}_{(i+1)T} Y_{(i+1)T} \tag{4.49}$$

updated at steps 1-3 of the algorithm.

Algorithm 8 Online Bayesian System Identification: Fixed Forgetting Factor

Inputs: forgetting factor $\bar{\gamma}$, previous estimates $\{\hat{\eta}^{(i)}, \hat{\eta}^{(i-1)}\}$, previous data matrices $\{R_{\bar{\gamma}}^{(i)}, \widetilde{Y}_{\bar{\gamma}}^{(i)}, \overline{Y}_{\bar{\gamma}}^{(i)}\}$, new data $\mathcal{D}_{i+1} = \{u(t), y(t)\}_{t=iT+1}^{(i+1)T}$

1: $R_{\bar{\gamma}}^{(i+1)} \leftarrow \bar{\gamma}^T R_{\bar{\gamma}}^{(i)} + \left(\Phi_{iT+1}^{(i+1)T}\right)^{\top} \bar{\Gamma}_T \; \Phi_{iT+1}^{(i+1)T}$

2: $\widetilde{Y}_{\bar{\gamma}}^{(i+1)} \leftarrow \gamma^T \widetilde{Y}_{\gamma}^{(i)} + \left(\Phi_{iT+1}^{(i+1)T}\right)^{\top} \bar{\Gamma}_T \; Y_{iT+1}^{(i+1)T}$

3: $\overline{Y}_{\bar{\gamma}}^{(i+1)} \leftarrow \bar{\gamma}^T \overline{Y}_{\bar{\gamma}}^{(i)} + \left(Y_{iT+1}^{(i+1)T}\right)^{\top} \bar{\Gamma}_T \; Y_{iT+1}^{(i+1)T}$

4: $\widehat{g}_{LS}^{(i+1)} \leftarrow {R_{\bar{\gamma}}^{(i+1)}}^{-1} \widetilde{Y}_{\bar{\gamma}}^{(i+1)}$

5: $\hat{\sigma}^{(i+1)^2} \leftarrow \frac{1}{(i+1)T-n} \left(\bar{Y}_{\bar{\gamma}}^{(i+1)} - 2\widetilde{Y}_{\bar{\gamma}}^{(i+1)^{\top}} \widehat{g}_{LS}^{(i+1)} + \widehat{g}_{LS}^{(i+1)^{\top}} R_{\bar{\gamma}}^{(i+1)} \widehat{g}_{LS}^{(i+1)}\right)$

6: $\hat{\eta}^{(i+1)} \leftarrow \arg\min_{\eta \in \Omega} \; f_{(i+1)T}(\eta)$ (use Algorithm 3)

7: $\widehat{g}^{(i+1)} \leftarrow \left(R_{\bar{\gamma}}^{(i+1)} + \hat{\sigma}_{\bar{\gamma}}^{(i+1)^2} K_{\hat{\eta}^{(i+1)}}^{-1}\right)^{-1} \widetilde{Y}_{\bar{\gamma}}^{(i+1)}$

Output: $\widehat{g}^{(i+1)}$, $\hat{\eta}^{(i+1)}$

We should stress that in this setting the forgetting factor $\bar{\gamma}$ has to be a priori chosen by the user and even if some range of values, $\bar{\gamma} = [0.95, 0.99]$, have been suggested in literature , Ljung (1999), the 'correct' value is data dependent and it has to be empirically chosen in each application. In this regard, we propose to estimate it as a hyperparameter in the next section.

4.3.2 Treating the Forgetting Factor as a Hyperparameter

The Bayesian framework provides the user with the possibility to treat the forgetting factor as a hyperparameter and therefore to estimate it by standard techniques.

Accordingly, the first k data are generated from the following linear model:

$$Y_k = \Phi_k g + E_\gamma, \quad E_\gamma = [e_\gamma(1), ..., e_\gamma(k)]^{\top} \sim \mathcal{N}(0, \sigma^2 \Gamma_k^{-1}) \tag{4.50}$$

where the $G_k G_k =: \Gamma_k$ and $\Gamma_k := diag\left(\gamma^{k-1}, \gamma^{k-2}, ..., \gamma^0\right)$.

Therefore, treating the forgetting factor as a hyperparameter is equivalent to modeling the noise with a non-constant variance and to giving to the diagonal entries of the

covariance matrix an exponential decaying structure.

Notice that model (4.50) is equivalent to model (4.42) but considering the forgetting factor as an hyperparameter and not as a fixed variable. The hyperparameters can be computed by solving:

$$\hat{\eta}^{(i)}, \hat{\gamma}^{(i)} = \underset{\eta\in\Omega,\gamma\in(0,1]}{\arg\min} \; f_k(\eta,\gamma) \tag{4.51}$$

$$f_k(\eta,\gamma) = Y_k^\top G_k \Sigma_Y(\eta,\gamma)^{-1} G_k Y_k + \ln\det\Sigma_Y(\eta,\gamma) - \ln\det\Gamma_k \tag{4.52}$$

$$\Sigma_Y(\eta,\gamma) = G_k \Phi_k K_\eta \Phi_k^\top G_k + \sigma^2 I_k \tag{4.53}$$

The online implementation of this approach is detailed in Algorithm 9, where

$$R_{\hat{\boldsymbol{\gamma}}}^{(i)} := \left(\Phi_{(i-1)T+1}^{iT}\right)^\top \widehat{\Gamma}_T^{(i)} \Phi_{(i-1)T+1}^{iT}, \tag{4.54}$$

$$\widetilde{Y}_{\hat{\boldsymbol{\gamma}}}^{(i)} := \Phi_{(i-1)T+1}^{iT^\top} \widehat{\Gamma}_T^{(i)} Y_{(i-1)T+1}^{iT}, \tag{4.55}$$

$$\overline{Y}_{\hat{\boldsymbol{\gamma}}}^{(i)} := Y_{(i-1)T+1}^{iT^\top} \widehat{\Gamma}_T^{(i)} Y_{(i-1)T+1}^{iT} \tag{4.56}$$

with $\widehat{\Gamma}_T^{(i)} = diag((\hat{\gamma}^{(i)})^{T-1}, .., (\hat{\gamma}^{(i)})^0)$.

Algorithm 9 Online Bayesian System Identification: Forgetting Factor as a hyperparameter

Inputs: previous estimates $\{\hat{\eta}^{(i)}, \hat{\eta}^{(i-1)}, \hat{\gamma}^{(i)}, \hat{\gamma}^{(i-1)}\}$, previous data matrices $\{R_{\hat{\boldsymbol{\gamma}}}^{(i)}, \widetilde{Y}_{\hat{\boldsymbol{\gamma}}}^{(i)}, \overline{Y}_{\hat{\boldsymbol{\gamma}}}^{(i)}\}$, new data $\mathcal{D}_{i+1} = \{u(t), y(t)\}_{t=iT+1}^{(i+1)T}$

1: $R_\gamma^{(i+1)} \leftarrow \gamma^T R_{\hat{\boldsymbol{\gamma}}}^{(i)} + \left(\Phi_{iT+1}^{(i+1)T}\right)^\top \Gamma_T \; \Phi_{iT+1}^{(i+1)T}$

2: $\widetilde{Y}_\gamma^{(i+1)} \leftarrow \gamma^T \widetilde{Y}_{\hat{\boldsymbol{\gamma}}}^{(i)} + \left(\Phi_{iT+1}^{(i+1)T}\right)^\top \Gamma_T \; Y_{iT+1}^{(i+1)T}$

3: $\overline{Y}_\gamma^{(i+1)} \leftarrow \gamma^T \overline{Y}_{\hat{\boldsymbol{\gamma}}}^{(i)} + \left(Y_{iT+1}^{(i+1)T}\right)^\top \Gamma_T \; Y_{iT+1}^{(i+1)T}$

4: $\widehat{g}_{LS}^{(i+1)} \leftarrow (R_{\hat{\boldsymbol{\gamma}}}^{(i)})^{-1} \widetilde{Y}_{\hat{\boldsymbol{\gamma}}}^{(i)}$

5: $\hat{\sigma}^{2^{(i+1)}} \leftarrow \frac{1}{(i+1)T-n}\left(\overline{Y}_{\hat{\boldsymbol{\gamma}}}^{(i)} - 2(\widetilde{Y}_{\hat{\boldsymbol{\gamma}}}^{(i)})^\top \widehat{g}_{LS}^{(i+1)} + (\widehat{g}_{LS}^{(i+1)})^\top R_{\hat{\boldsymbol{\gamma}}}^{(i)} \widehat{g}_{LS}^{(i+1)}\right)$

6: $\hat{\eta}^{(i+1)}, \hat{\gamma}^{(i+1)} \leftarrow \arg\min_{\eta\in\Omega,\gamma\in(0,1]} \; f_{(i+1)T}(\eta,\gamma)$ (use Algorithm 3)

7: $\widehat{g}^{(i+1)} \leftarrow \left(R_{\hat{\boldsymbol{\gamma}}}^{(i+1)} + \hat{\sigma}^{2^{(i+1)}} \; K_{\hat{\eta}^{(i+1)}}^{-1}\right)^{-1} \widetilde{Y}_{\hat{\boldsymbol{\gamma}}}^{(i+1)}$

Output: $\widehat{g}^{(i+1)}$, $\hat{\eta}^{(i+1)}$

4.3.3 Sliding Window

As a third approach to cope with time-varying systems in the Gaussian Regression framework, a "sliding window" over the data is proposed: whenever a new dataset $\mathcal{D}_{i+1}$ is provided, the new impulse response estimate $\widehat{g}_w^{(i+1)}$ and the corresponding hyperparameters $\hat{\eta}_w^{(i+1)}$ are computed using the last N_w input-output data pairs. The window length, N_w, has to be chosen by the user and it has a similar trading-off role, between the tracking of rapidly-changing system parameters and the estimation accuracy, as the forgetting factor does in the previous sections.

Notice that when the sliding window approach is adopted, there are no recursive equations analogous to (4.8)-(4.9)-(4.10). However, the computational complexity of the algorithm is still suitable for online applications since it is fixed and it can be regulated by choosing the length of N_w.

Algorithm 10 outlines how the regularization/Bayesian framework can be equipped with this technique in an "online" setting.

Algorithm 10 On-Line Bayesian System Identification - Sliding Window

Inputs: previous estimates $\left\{\hat{\eta}_w^{(i)}, \hat{\eta}_w^{(i-1)}\right\}$, new data $\mathcal{D}_{i+1} = \{u(t), y(t)\}_{t=iT+1}^{(i+1)T}$, windowed data $\mathcal{D}_{i+1}^w = \{u(t), y(t)\}_{t=iT-N_w+T}^{(i+1)T}$

1: $R_w^{(i+1)} \leftarrow \left(\Phi_{iT-N_w+T}^{(i+1)T}\right)^\top \Phi_{iT-N_w+T}^{(i+1)T}$

2: $\widetilde{Y}_w^{(i+1)} \leftarrow \left(\Phi_{iT-N_w+T}^{(i+1)T}\right)^\top Y_{iT-N_w+T}^{(i+1)T}$

3: $\overline{Y}_w^{(i+1)} \leftarrow \gamma \left(Y_{iT-N_w+T}^{(i+1)T}\right)^\top Y_{iT-N_w+T}^{(i+1)T}$

4: $\widehat{g}_{LS}^{(i+1)} \leftarrow {R_w^{(i+1)}}^{-1} \widetilde{Y}_w^{(i+1)}$

5: $\hat{\sigma}_w^{(i+1)^2} \leftarrow \frac{1}{(i+1)T-n}\left(\bar{Y}_w^{(i+1)} - 2\widetilde{Y}_w^{(i+1)^\top}\widehat{g}_{LS}^{(i+1)} + \widehat{g}_{LS}^{(i+1)^\top} R_w^{(i+1)} \widehat{g}_{LS}^{(i+1)}\right)$

6: $\hat{\eta}^{(i+1)}, \hat{\gamma}^{(i+1)} \leftarrow \arg\min_{\eta\in\Omega,\gamma\in(0,1]} f_{(i+1)T}(\eta,\gamma)$ (use Algorithm 3)

7: $\widehat{g}_w^{(i+1)} \leftarrow \left(R_w^{(i+1)} + \hat{\sigma}_w^{(i+1)^2} K_{\hat{\eta}^{(i+1)}}^{-1}\right)^{-1} \widetilde{Y}_w^{(i+1)}$

Output: $\widehat{g}_w^{(i+1)}$

The computational complexity of Algorithm 10 is $O(N_w^3)$ and it is determined by the steps 7, while the required memory storage is $O(N_w^2)$ for the matrices in step 1.

4.4 Simulations Results

The purpose of this section is to test the standard online algorithms for PPEM described in Sections 2.3.2 and the online algorithms for NPPEM for time-varying systems proposed in Section 4.3.

The experiments will focus on three main points.

First, the performance of the 1-step projected gradient methods and the 1-step EM algorithm, outlined in Section 4.2, are compared here while coping with time-variant systems; the algorithms will be called "SGP","BB","BFGS","EM". A comparison of these techniques have been already analysed in Section 4.2.3 for time invariant systems, showing a superiority of the approach based on SGP and EM, see Section 4.2.1. For this reason, the results will be only briefly discussed.

Second, the three different routines proposed in Section 4.3 are tested while dealing with time-varying systems; from here on, we will use the acronyms "TC FF" when a fixed forgetting factor is adopted, "TC est FF" when the forgetting factor is estimated as a hyperparameter and "TC W" when a sliding window is used.

Third, the online PPEM and NPPEM approaches are compared.

Data

A Monte Carlo study experiment over 200 time-varying systems has been performed. Each system generated a data set of 3000 input-output measurement pairs created as follows: the first 1000 data are generated from one system contained in the data-bank "D4" (proposed in Chen et al. (2014)), while the remaining 2000 data are generated by perturbing the D4-system with two additional poles and zeros. These additional poles are generated so that the order of the D4-system actually changes, which means that no zero-pole cancellations apply and that they lead to a variation in the frequency response, thus creating a significant switch on the data generating system at time $k = 1001$.

The data-bank "D4" consists of 30th order random SISO dicrete-time systems having all the poles inside a circle of radius 0.95. These systems are simulated with a unit variance band-limited Gaussian signal with normalized band $[0, 0.8]$. A zero mean white Gaussian noise, with variance adjusted so that the Signal to Noise Ration (SNR) is always equal to 1, is then added to the output data.

The above-mentioned experiments have been applied also on the data-bank "D2" (proposed in Chen et al. (2014)), which only differs from "D4" because the input signal is not filtered. The obtained results are analogous to the one reported for dataset "D4"; therefore, they are not reported.

Estimators

The Recursive PPEM estimators are computed with the `roe` Matlab routine, using the BIC criterion for the model complexity selection. In the following, this estimator will be denoted as "PPEM BIC". Furthermore, as a benchmark we introduce the parametric

oracle estimator, called "PPEM OR", which selects the model complexity by choosing the order model that gives the best fit to the impulse response of the true system. The order selection is performed every time a new dataset becomes available: multiple models with orders ranging from 1 to 20 are estimated and the order selection is performed according to the two criteria described above.

Regarding the methods relying on Bayesian inference, a zero-mean Gaussian prior is adopted with a covariance matrix (kernel) given by the so-called "TC"-kernel defined in (4.40). The length n of the estimated impulse responses is set to 100. In the following, we will use the acronym "TC" to refer to these methods. Furthermore, the notation "OPT" will refer to the standard Bayesian procedure, in which the SGP algorithm adopted to optimize the marginal likelihood $f_k(\eta)$ is run until the relative change in $f_k(\eta)$ is less than 10^{-9}. From here on, the online counterpart will be referred to as the "1-step ML".

Performance

For each of the 200 Monte Carlo runs, the identification algorithms are initialized using the first batch of data $\mathcal{D}_{init} = \{u(t), y(t)\}_{t=1}^{300}$; in the Bayesian procedures, the routines adopted for the optimization of the marginal likelihood are run until convergence in the initial step. After this initial step, the estimators are updated every $T = 10$ time steps, when new data $\mathcal{D}_{i+1} = \{u(t), y(t)\}_{t=iT}^{(i+1)T}$ are provided. The forgetting factor in the "TC FF" and "PPEM" methods is set to 0.998, while its estimation in "TC est FF" method is initialized with 0.995; "TC W" methods adopt a window length $N_w = 800$. The chosen values of the forgetting factor and of the window length are comparable in the amount of data they take in consideration, accordingly to Ljung (1999).

The performance we are interested in, regards the adherence of the estimated impulse responses to the true ones and the computational cumulative time.

The adherence index we choose is the impulse response fit:

$$\mathcal{F}(\widehat{g}) = 100 \cdot \left(1 - \frac{\|g - \widehat{g}\|_2}{\|g\|_2}\right) \tag{4.57}$$

where g, $\widehat{g}$ are the true and the estimated impulse responses of the considered system, respectively.

As previously mentioned, the first comparison is in the "1-step marginal likelihood optimization" algorithms. Table 4.2 summarizes the performance in terms of mean and standard deviation achieved in the Monte Carlo study after the estimators "SGP", "BB", "BFGS" and "EM" have been updated (every $T = 10$ new data) using all the $k = 3000$ input-output measurements in each data set. For each dataset in the MC study the

impulse response fit has been averaged over time.

FF	FIT mean	FIT std	Time mean	Time std
TC SGP	69.35	8.25	0.44	0.03
TC BB	69.30	8.23	1.31	0.10
TC BFGS	69.33	8.46	1.80	0.38
TC EM	66.98	14.00	0.43	0.05

Table 4.2: "1-step marginal likelihood optimization" algorithms: mean and standard deviation over the 200 data sets of the impulse response fit and the computational cumulative time after all the $\mathcal{D} = \{u(t), y(t)\}_{t=1}^{3000}$ are processed.

The SGP algorithms outperform all the others when comparing both the impulse response fit and the computational cumulative time. The comparison has been checked also after a different amount of data k has been seen and in the overall SGP was the outperforming technique. For this reason, in the following analysis only the SGP technique will be considered and the acronym "TC" will refer to the online Bayesian estimates updated with SGP.

At this point, Gaussian regression algorithms can be compared the classical recursive PPEM proposed in Section 4.3 and 2.3.2, respectively. Figure 4.4 shows the performance in terms of impulse response fit at five selected time instants.

As expected, using only the first batch of data $\mathcal{D}_{init}$, the "TC" estimators outperform the recursive PPEM: it is well known that the regularized/Bayesian estimators are particularly efficient when a reduced amount of data is available. It is interesting to note that even when $k = 1000$ the "TC" methods reach a performance regime that slightly outperform the ideal parametric estimator "PPEM OR".

Until time $k = 1000$, the performance among the "TC" methods are similar since both the forgetting factor and the window cover a comparable amount of the data and the data considered so far can be associated to a time-invariant system.

After the switch in the data generating system, which occurs at $k = 1001$, performance are subjected to an abrupt degradation, since most of the data that are passed to the estimators are generated from the 'wrong' system. The "TC FF" methods are faster in recovering the fit performance than "TC W": this is because the latter equally weights data before and after the switch, while through the forgetting factor effect, less importance is given to the data from the original system.

When $k = 1800$ the "TC W" estimators are in the ideal situation because they are fed only with data coming from the current system; instead, the cost function in "TC FF" methods still takes into account also data from the old system (even if scaled down).

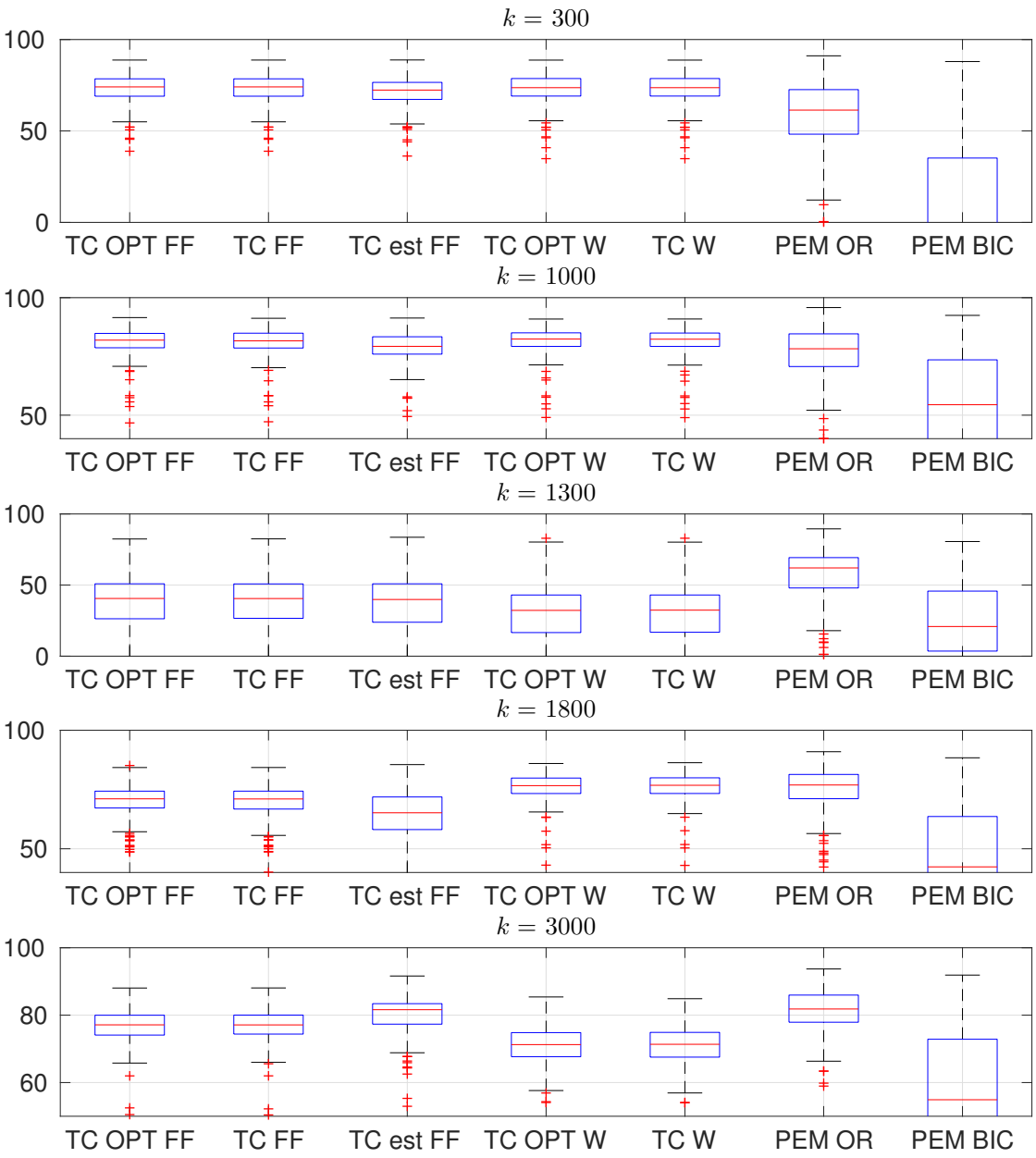

Figure 4.4: Impulse response fit $\mathcal{F}(\widehat{g})$ achieved at five time instants k (corresponding to the number of data available for the estimation).

It is interesting to note that when $k = 3000$ the "TC FF" methods perform better than the "TC W": this gives an empirical evidence that the old (rescaled) data appearing in the loss function of "TC FF" methods are still relevant to the computation of the estimate. These results suggest a general qualitative guide-line: the use of the forgetting factor seems preferable when the systems to be identified are either varying 'rapidly' or 'very slowly' and the sliding window approach appears a good choice in between these (arbitrary defined) behaviours.

The conclusions derived for $k = 1800$ and $k = 3000$ might seem to go against the fact that the length of the window and the decay given by the forgetting factor have been chosen to cover a similar amount of data, but this fact is true only indicatively, and the reported results give a better understanding on the behaviour of the two methodologies.

The "1-step ML" procedures and the correspondent "OPT" routines provide analogous performance at each time step k, validating the method we propose to perform online estimation and confirming the results obtained in Section 4.2.3.

Among the regularization/Bayesian routines,"TC est FF" seems to be preferable: indeed, after the switch, it recovers the fit performance a bit slower than "TC FF" but faster than "TC W"; on the other hand, at regime it outperforms all the other approaches because it can choose forgetting factor values that retain a larger amount of data.

The unrealistic "PPEM OR" represents the reference on the achievable performance of the PPEM estimators; it outperforms the "TC" methods in the transient after the switch, while it has comparable performance at regime. Whereas, the recursive "PPEM BIC" estimator performs very poorly.

	TC					PPEM	
	OPT FF	FF	est FF	OPT W	W	OR	BIC
mean	6.70	0.44	0.9	6.44	0.85	18.44	18.44
std	1.28	0.03	0.37	1.03	0.08	0.69	0.69

Table 4.3: Mean and standard deviation over 200 data sets of the computational cumulative time after the data $\mathcal{D} = \{u(t), y(t)\}_{t=1}^{3000}$ are processed.

Finally, Table 4.3 summarizes the computational cumulative time of the proposed algorithms in terms of mean and standard deviation after the estimators are fed with all the data $\mathcal{D} = \{u(t), y(t)\}_{t=1}^{3000}$.

The "1-step ML" procedures are one order of magnitude faster than the corresponding "OPT" ones and two orders of magnitude faster than the recursive "PPEM" estimators. This confirm how the "1-step ML" procedures are the most appealing techniques for online applications.

4.5 Conclusions

In this chapter, the recently introduced system identification techniques relying on Gaussian Regression have been extended to the online identification of dynamical systems.

In order to meet real-time requirements, the reduction of the computational time required to update the impulse response estimate becomes essential. In a Bayesian estimation procedure, the most demanding step in terms of computational complexity is the marginal likelihood optimization required to determine the hyperparameters estimate. Hence, an efficient version of different iterative procedures, typically used to solve the marginal likelihood maximization problem, has been proposed: the hyperparameters are updated by performing only one iteration of these procedures, each time a new dataset becomes available.

Moreover, in order to cope with time-varying dynamical systems, three approaches, based on the use of a forgetting factor or of a sliding window over the data, have been proposed. We also investigate the estimation of the forgetting factor by treating it as an hyperparameter of the Bayesian inference procedure.

The experimental results appear very promising, even for practical contexts. We believe that the preliminary investigation performed in this work may pave the way for further research in the online identification of dynamical systems using Gaussian regression.

Discussion

The methodology and the experiments run in this chapter have confirmed what discussed in Section 2.5.

The powerful capability revealed by the NPPEM methods of trading the model complexity in a continuous a manner through the estimation of the hyperparameters appears to be one key point to successful online applications for system identification.

Indeed, under the computational point of view, model complexity selection in PPEM requires the solution of several possibly high-dimensional non-convex optimization problems at each update, whereas this step is much faster in NPPEM methods since it reduces to tuning few hyperparameters.

The results in terms of impulse response fit obtained in this Chapter together with the results in Chapter 3 give evidence that NPPEM effectively outperform classical PPEM in facing the bias-variance tradeoff.

Future Works

Future research directions could consider:

- the recursive update of the Bayesian estimate, resembling the update available for parametric techniques; consequently, the impulse response would depend on the multiple hyperparameters estimated at different time steps,
- the derivation of an appropriate criteria to select the number of optimization iterations to be performed in the maximization of the marginal likelihood. Indeed, the a priori choice of making only one iteration is arbitrary.
- the attempt of finding an approximation of the hyperparameters update, in order to further reduce the computational complexity,
- the proof of the consistency of the estimators obtained with this one step update rule.

5

Enforcing Model Stability in Nonparametric Gaussian Regression

5.1 Introduction

The recent nonparametric Gaussian regression methods applied to linear system identification describe the unknown system directly in terms of impulse response. It has been shown in the recent literature Pillonetto et al. (2011a); Chen et al. (2012); Pillonetto and De Nicolao (2010) that this approach supported by Bayesian Statistics provide powerful tools to face the bias-variance tradeoff in the identification problem.

The paper Pillonetto et al. (2011a) has shown how these infinite dimensional model classes can be used for identification of linear systems in the framework of prediction error methods, leading naturally to stable predictors.

However, as discussed in Section 2.5, the stability of the predictor model does not necessarily guarantee stability of the so called "forward" model.

In control theory terminology, the predictor is the closed-loop model and the forward model is the open-loop model plant.

It can be argued that in general, imposing stability of the open-loop model is quite a natural requirement given that the majority of the real-world systems are stable. Moreover, in many contexts, not necessarily directly related to the control theory, what is eventually used is the open-loop model.

As a matter of fact, we faced this stability issue when performing identification on a real data set from EEG recordings. A physical insight in this case suggests that the transfer function describing the link between potentials in different brain locations were expected to be stable, while the identified models were not.

Contributions

The motivations given by the link to the real-world applications appear quite notable. Hence, in this chapter we tackle the problem of identifying stable forward models when NPPEM are used. Four possible solutions to this problem are described and compared.

1. Inspired by the works Chilali and Gahinet (1996); Miller and de Callafon (2013) the so-called "LMI-constraint" approach is adapted to constrain the eigenvalues of the estimated model within the unit circle.

2. A penalty term is added in the optimization procedure of the "classic" Stable-Spline algorithm Pillonetto et al. (2011a); Pillonetto and De Nicolao (2010). This penalty smoothly imposes the stability constrain to the eigenvalue of the forward model with the maximum absolute value.

3. The posterior distribution over the impulse responses that exclude all the unstable model is considered and it is called a "stable posterior". This distribution cannot be analytically computed and is obtained in a sampling form through a Markov Chain Monte Carlo approach (MCMC). Two possible solutions are achieved thorugh this technique: the minimum variance estimate and the maximum a posteriori estimate.

The last two techniques have the advantage, w.r.t. the first one, of being integrated directly inside the pre-existing optimization problem and do not simply post-process the estimates. An extensive simulation study comparing these techniques will be provided.

The chapter is structured as follows. In Section 5.2, the statement of the stable forward model identification problem is formulated. Sections 5.3, 5.4 and 5.5 report the procedure of the four different stabilization approaches used to tackle the problem described in Section 5.2. The experiment to validate the used techniques and the results are described in Section 5.6. Finally, our conclusions are drawn in Section 5.7.

5.2 Problem Statement

Consider the dataset $\mathcal{D}^N := \{\mathcal{D}(t)\}_{t=1}^N = \{u(t),\, y(t)\}_{t=1}^N$ the discrete causal time-invariant model defined in (2.6) called forward model

$$y(t) = G(z)u(t) + H(z)e(t) \tag{5.1}$$

and the one-step ahead predictor model (2.23)

$$\hat{y}(t|t^-) = H^{-1}(z)\left[(H(z)-1)y(t) + G(z)u(t)\right] \tag{5.2}$$
$$= W^y(z)y(t) + W^u(z)u(t) \tag{5.3}$$

where

$$W^y(z) = \sum_{k=1}^{n} w^y(k)z^{-k}, \quad W^u(z) = \sum_{k=1}^{n} w^u(k)z^{-k} \tag{5.4}$$

and $n \in \mathbb{N}$ is the length of the impulse responses.

Recall the assumptions made in Section 2.1 that are: $G(z)$ is stable and $H(z)$ is stable and minimum phase, i.e., both $H(z)$ and its inverse are causal and stable (all the poles and zeros of $H(z)$ are inside the unit circle).

In classic linear PPEM described in Section 2.3.1 the parameters $\theta \in \mathbb{R}^d$ are usually constrained in a domain Θ so as to account for prior knowledge such as stability of $G_\theta(z)$, $H_\theta(z)$ and $H_\theta^{-1}(z)$. Thus, the stability of both the forward model (5.1) and of the predictor model (5.2) are guaranteed within these parametric methods.

The NPPEM approaches described in Section 2.4 aim to find the estimators $\hat{w}^y$ and $\hat{w}^u$ of $w^y := \{w^y(k)\}_{k=1}^n$ and $w^u := \{w^u(k)\}_{k=1}^n$. How to find these estimates has been shown in Section 2.4.4. The estimates were derived for an equivalent formulation of the model; here we report, for the sake of the understanding, the equivalent estimates explicitly distinguishing the two impulse responses w^y and w^u. Specifically, w^y and w^u are assumed to be independent with the same covariance matrix, that is:

$$K_\eta(\mathcal{D}(t), \mathcal{D}(s)) = cov\left(w^y(t), w^y(s)\right) = cov\left(w^u(t), w^u(s)\right),$$
$$p_\eta(w^y, w^u) = p_\eta(w^y)p_\eta(w^u)$$

Under the assumption that the innovation process is Gaussian and independent of w^y and w^u, the marginal $p_\eta(Y)$ and the posterior $p_\eta(w^y, w^u|Y)$ are Gaussian. Then, following the Empirical Bayes paradigm and fixing the hyperparameters to their estimated

value $\hat{\eta}_{ML}$, see Section 2.4.3, the impulse responses estiamte are found as:

$$\hat{w}^y := E_{\hat{\eta}_{ML}}[w^y|Y] \quad \hat{w}^u := E_{\hat{\eta}_{ML}}[w^u|Y] \tag{5.5}$$

where $E_{\hat{\eta}_{ML}}[\cdot|\cdot]$ denotes the minimum variance estimator having fixed $\eta = \hat{\eta}_{ML}$.

Unfortunately, BIBO stability of the predictor impulse responses $\hat{w}^y$ and $\hat{w}^u$ does not guarantee BIBO stability of the transfer function estimates $G(z)$ and $H(z)$ of the forward model (5.1)

$$\hat{G}(z) := \frac{\sum_{k=1}^{n} \hat{w}^u(k) z^{-k}}{1 - \sum_{k=1}^{n} \hat{w}^y(k) z^{-k}} = \sum_{k=1}^{\infty} g(k) z^{-k}, \;\; \hat{H}(z) := \frac{1}{1 - \sum_{k=1}^{n} \hat{w}^y(k) z^{-k}} = \sum_{k=1}^{\infty} h(k) z^{-k} \tag{5.6}$$

Indeed, BIBO stability of the sequences $\hat{w}^y = \{\hat{w}^y(k)\}_{k\in\mathbb{Z}^+}$ and $\hat{w}^u = \{\hat{w}^u(k)\}_{k\in\mathbb{Z}^+}$ has no relation with stability of $\hat{G}(z)$ and $\hat{H}(z)$ which, if no cancellations occur, depends on the zeros of the polynomial $1 - \hat{W}^y(z) = 1 - \sum_{k=1}^{n} \hat{w}^y(k) z^{-k}$.

Recall that $g := \{g(k)\}_{k=\mathbb{Z}^+}$ and $h := \{h(k)\}_{k=\mathbb{Z}^+}$ are the impulse responses of the forward model, obtained by inverse $\mathcal{Z}$ transform of $G(z)$ and $H(z)$, respectively.

Thus, the problem of identifying stable forward models via NPPEM can be formulated as follows:

Problem 5.2.1. *Given the data $\{u(t), y(t)\}_{t=1}^{N}$, estimate the impulse response coefficients $\{\hat{w}^y(k)\}_{k\in[1,n]}$ and $\{\hat{w}^u(k)\}_{k\in[1,n]}$ of the predictor model so that the transfer functions of the forward model, $\hat{G}(z)$ and $\hat{H}(z)$ in (5.6), are BIBO stable. A sufficient generic*[1] *condition for this to happen is that*

$$\begin{gathered} A(z) = 1 - \hat{W}^y(z) = z^n(1 - \textstyle\sum_{k=1}^{n} \hat{w}^y(k) z^{-k}) = z^n - [z^{n-1} \dots 1]\hat{w}^y \\ \hat{w}^y := [\hat{w}^y(1), \hat{w}^y(2), \dots, \hat{w}^y(n)]^\top \end{gathered} \tag{5.7}$$

is (Schur) stable, i.e., has all roots inside the open unit disc of the complex plane $\mathbb{C}$.

Three techniques are described and compared in the following sections to achieve this aim. For each technique, Problem 5.2.1 is reformulated accordingly in order to understand the different ideas and approaches proposed.

[1] i.e., if no cancellations occur, which is generic for estimated impulse responses.

5.3 Stabilization via LMI constraint

The first stabilization technique is based on formulating stability of the model (5.6) as a constraint on the eigenvalues of the companion matrix of $A(z)$ in (5.7). This constraint can be characterized in terms of Linear Matrix Inequalities (LMI) as discussed in Chilali and Gahinet (1996) and used later on in Miller and de Callafon (2013) to enforce stable models in subspace identification. Accordingly, Problem 5.2.1 can be formulated as follows.

Problem 5.3.1 (Reformulation)**.** *Given a preliminary estimate* $\widetilde{w}^y := [\widetilde{w}^y(1), .., \widetilde{w}^y(n)]^\top$, *find a vector of coefficients* $\hat{w}^y$ *so that*

$$\hat{w}^y = \underset{w^y \in \mathcal{W}_\mathcal{D}}{\arg\min} \|w^y - \widetilde{w}^y\|^2 \tag{5.8}$$

where the set $\mathcal{W}_\mathcal{D} := \{w^y \in \mathbb{R}^n : |\lambda| < 1 \ \forall \lambda \ s.t. \ \hat{A}(\lambda) = 0, \hat{A}(z) = z^n - [z^{n-1} \dots 1]\hat{w}^y\}$, *can be described by an LMI constraint as discussed below.*

The quadratic programming problem (5.8) with LMI constraints can be easily solved by available software, e.g., the *CVX Toolbox* in Matlab has been used, see Grant, Boyd, and Ye (2006) for more details.

The estimator $\widetilde{w}^y$ can be, in principle, any estimator of w^y. In this chapter, the Empirical Bayes estimate (5.5) is adopted and it will be denoted as $\widetilde{w}^y := \widetilde{w}^y_{EB}$.

It should be observed that the use of the 2-norm in (5.8) is entirely arbitrary and, in fact, considering some form of model approximation error (e.g. difference of output predictors) would be preferable. In addition, when $\widetilde{w}^y$ is the outcome of a preliminary estimation step, a principled solution would require accounting for the distribution of $\widetilde{w}^y$, e.g., weighting the difference $(w^y - \widetilde{w}^y)$ by the inverse of the variance of $\widetilde{w}^y$, Wahlberg (1989). However, this brings in some technical difficulties related to the formulation of the quadratic problem, therefore, it is still a subject of research.

Formulation of the LMI constraint

Let $\mathbf{D}$ be the open unit disc of the complex plane:

$$\mathbf{D} = \{z \in \mathbb{C} : |z| < 1\} \tag{5.9}$$

It is well known, see e.g. Chilali and Gahinet (1996), that the set $\mathbf{D}$ can be expressed in terms of the matrix polynomial

$$q_{\mathbf{D}}(z) = I_2 + \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} z + \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix} \bar{z} \tag{5.10}$$

as

$$\mathbf{D} = \{z \in \mathbb{C} \; s.t. \; q_{\mathbf{D}}(z) > 0\}$$

In Chilali and Gahinet (1996), the authors show that a matrix Q has all its eigenvalues in the LMI region $\mathbf{D}$ if and only if there exists $P = P^\top \geq 0$ s.t.

$$M(Q, P) = I_2 \otimes P + \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \otimes (QP) + \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix} \otimes (QP)^\top \geq 0 \tag{5.11}$$

Let $\Psi(w^y)$ be the companion matrix of w^y

$$\Psi(w^y) := \begin{bmatrix} 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & 1 \\ -w^y(n) & -w^y(n-1) & -w^y(n-2) & \dots & -w^y(1) \end{bmatrix} \in \mathbb{R}^{n \times n}$$

Accordingly to (Miller and de Callafon, 2013, Theorem 1), which presents small variations w.r.t. the original central theorem in Chilali and Gahinet (1996), the LMI constraint to guarantee the stability of $\Psi(w^y)$ and therefore of w^y is $M(\Psi(w^y), P) \geq 0$, where $M(\Psi(w^y), P)$ is defined in (5.11). Unfortunately, $M(\Psi(w^y), P)$ is not linear in w^y and P since their product appears. This nonlinearity calls for a parametrization of the constraint similarly to what has been done in Miller and de Callafon (2013).

In order to specify the parametrization, we introduce the quantities $P \in \mathbb{R}^{n \times n}$, $\Psi \in \mathbb{R}^n$ such that $\psi := Pw^y$ and $J := [O_{n \times 1} \, I_{n-1}]$, and define

$$M(\psi, P) := M(\Psi(w^y), P), \quad \text{with } w^y = P^{-1}\psi$$

i.e.,

$$M(\psi, P) = I_2 \otimes P + \left(\begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \otimes \begin{bmatrix} JP \\ \Psi^\top \end{bmatrix} \right) + \left(\begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \otimes \begin{bmatrix} JP \\ \Psi^\top \end{bmatrix} \right)^\top \tag{5.12}$$

which is linear in ψ and P.

Now, the stability property can be enforced adopting the LMI constraint $M(\psi, P) \geq 0$,

where $M(\psi, P)$ has been defined in (5.12).

Thus, problem 5.3.1 can be formalized as:

$$\begin{aligned} \hat{\psi}, \hat{P} = & \arg\min_{w^y, P} \|\psi - P\hat{w}_B^y\|^2 \\ \text{s. t.} \quad & M(\psi, P) \geq 0 \\ & Tr(P) = n \\ & P = P^\top \geq 0 \end{aligned} \tag{5.13}$$

where the constraint $Tr(P) = n$ is added to improve the numerical conditioning, see Miller and de Callafon (2013) for further details.

The solution $\hat{w}^y$ of Problem 5.3.1 is finally computed as:

$$\hat{w}^y = \hat{P}^{-1}\hat{\psi} \tag{5.14}$$

In the remaining of the chapter the model $\hat{G}(z)$ obtained by plugging in (5.6) the estimators $\hat{w}^y$ and $\hat{w}^u_{EB}$ obtained respectively from (5.14) and the EB procedure in (5.5), will be called "LMI" model.

5.4 Stabilization via Penalty Function

The second stabilization technique is developed to act directly inside the Gaussian regression procedure. As discussed in Section 2.4, a crucial step is the estimation of the hyperparameter vector η, that can be done e.g. through marginal likelihood optimization (2.48). It turns out that some hyperparameters η may lead to estimators (5.5) which do not correspond to stable models $\hat{G}(z)$ and $\hat{H}(z)$. Thus, one possible remedy is to restrict the set of admissible hyperparameters to a subset Ω_S which leads to stable models. This is not entirely trivial as the estimators (and thus the set Ω_S) depend on the measured data Y, U. Accordingly, Problem 5.2.1 can be formulated as follows.

Problem 5.4.1 (Reformulation)**.** *Estimate the hyperparameters η restricting the search*

$$\hat{\eta} = \arg\max_{\eta \in \Omega_S} p_\eta(Y) = \arg\min_{\eta \in \Omega_S} -\ln p_\eta(Y) \tag{5.15}$$

to the set $\Omega_S = \{\eta | \hat{A}(z)$ Stable $\}$, i.e., the set of hyperparameters which leads to stable models $\hat{G}(z)$, $\hat{H}(z)$.

Since the set Ω_S cannot be determined a priori because it is data dependent, a penalty term to the the criterion in (5.15) has been added. This penalty function can be

interpreted as a barrier to push the estimate $\hat{\eta}$ into Ω_S, or equivalently, to keep $\hat{\eta}$ away from the set of hyperparameters η which leads to an unstable $A(z)$.

Denote with $A_\eta(z)$ the polynomial $A(z)$ in (5.7) built with the estimator

$$\hat{w}_\eta^y := \mathbb{E}_\eta[w^y|Y], \tag{5.16}$$

which is to indicate that $\hat{w}_\eta^y$ is obtained with the specific hyperparameters $\hat{w}_\eta^y$ and define the dominant root of $A_\eta(z)$ as $\bar{\rho}_\eta := \max|\sigma(A_\eta(z))|$, where $\sigma(A(z))$ denotes the set of roots of the polynomial $A(z)$.

Next, the penalty function $J(\bar{\rho}_\eta)$ can be defined:

$$J(\bar{\rho}_\eta) = \frac{1}{(\alpha(\delta - \bar{\rho}_\eta))^\alpha} - \frac{1}{(\alpha\delta)^\alpha} \tag{5.17}$$

where $\delta \geq 1$ is a scalar which determines the limit point corresponding to an infinite value of the function and α is a positive scalar which adjusts the steepness of the function.

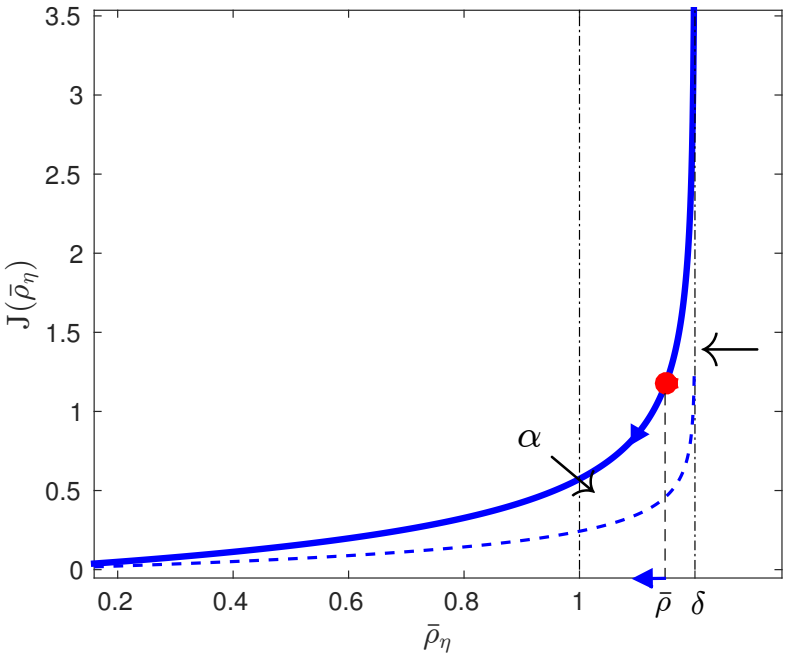

Figure 5.1: Representation of the penalty function $J(\bar{\rho}_\eta)$. The red bullet represents the value of the penalty function associated to a specific $\bar{\rho}$ in an illustrative example of an unstable polynomial $A_\eta(z)$. The blue arrows show the effects of the penalty function on $\bar{\rho}$ while estimating the hyperparameters. The black arrows show the effects of changing the parameters α and δ.

The penalty function (5.17) is illustrated in Figure 5.1 and it can be seen that it

diverges ($J(\bar{\rho}_\eta) \to \infty$) when $\bar{\rho} \to \delta$ and $J(\bar{\rho}_\eta) \to 0$ when $\bar{\rho} \to 0$. Thus, when (5.17) is added to the minimization problem (5.15), the effect is of penalizing the solutions η which yields $\bar{\rho}_\eta$ outside the stability region.

As it will be shown in Algorithm 11, the two parameters α and δ are iteratively adjusted until the estimated hyperparameters lead to a stable forward model which solves the constrained problem (5.15).

Note that when $\alpha \to 0$, $J(\bar{\rho}_\eta)$ gives no penalty for $\eta < \delta$ and an infinite penalty for $\eta \geq \delta$. Elaborating upon the intuition above, it is easy to prove that the solution of Problem 5.4.1 can be found by the algorithm described below:

Algorithm 11 Stabilization via Penalty Function

Init:
Compute η_0 through marginal likelihood maximization (Section 2.4.3),
Compute the predictor impulse response $\hat{w}^y_{\eta_0}$ using (5.16),
Compute $A_{\eta_0}(z)$ and $\bar{\rho}_{\eta_0}$ associated to $\hat{w}^y_{\eta_0}$,
Set $\alpha = 1$.
while $\bar{\rho}_{\eta_k} \geq 1$ **do**
    Set $\delta = \bar{\rho}_{\eta_k}(1 + \epsilon)$,
    Compute

$$\eta_k = \arg\min_\eta - \ln p_\eta(Y) + J(\bar{\rho}_\eta) \tag{5.18}$$

    and the associated $\bar{\rho}_{\eta_k}$,
    if $-\ln p_{\eta_k}(Y) + J(\bar{\rho}_{\eta_k}) = -\ln p_{\eta_{k-1}}(Y) + J(\bar{\rho}_{\eta_{k-1}})$ **then**
        $\alpha = \alpha - \Delta\alpha$, with $\Delta\alpha$ sufficiently small,
        $\delta = \delta - \Delta\delta$, with $\Delta\delta$ sufficiently small,
Set $\alpha = \epsilon$ and $\delta = 1$.
The solution of Problem 5.4.1 is given by:

$$\hat{\eta} = \arg\min_\eta - \ln p_\eta(Y) + J(\bar{\rho}_\eta) \tag{5.19}$$

$$\hat{w}^y_{\hat{\eta}} = \mathbb{E}_{\hat{\eta}}[w^y|Y], \quad \hat{w}^u_{\hat{\eta}} = \mathbb{E}_{\hat{\eta}}[w^u|Y] \tag{5.20}$$

In the remaining of the paper the model obtained by (5.6) using (5.20) will be called "ML + PF" model.

Remark 5.4.2. Notice that the iterative procedure which updates δ and α is needed because, in general, there is no guarantee that one can find an initial value of $\eta \in \Omega_S$.

Note also that the set Ω_S is always non-empty provided the hyperparameter η includes a scaling factor for the Kernel, i.e., a scalar variable which multiplies the Kernel. In fact, if this is the case, there exist values of η which leads to an estimator $\hat{w}^y = O_{n\times 1}$ which, in turn leads to stable $\hat{G}(z)$ and $\hat{H}(z)$.

5.5 Stabilization via a Full Bayes Sampling Approach

The third stabilization technique is based on a variant of the FBS paradigm. Consider the posterior distribution of the impulse responses w^y and w^u conditional to the data, it is known that some of the impulse responses in its support lead to a stable forward model and some others not.

"How can the domain of the posterior be restricted to only the impulse responses that satisfies Problem 5.2.1?".

Formally the "stable" posterior distribution is defined as

$$\begin{aligned} p_S(w^y, w^u|Y) &= \int p_S(w^y, w^u, \eta|Y)\, d\eta \\ &= \tfrac{1}{p(Y)} \int p(Y|w^y, w^u) p_S(w^y, w^u|\eta) p(\eta)\, d\eta \end{aligned} \tag{5.21}$$

where $p_S(w^y, w^u|\eta)$ is the "truncated" Gaussian prior

$$p_S(w^y, w^u|\eta) := \begin{cases} k_\eta p_\eta(w^y, w^u) & w^y : A(z) \text{ stable} \\ 0 & \text{otherwise} \end{cases} \tag{5.22}$$

which, a priori, excludes all impulse responses w^y which lead to unstable $A(z)$. Note that the constant k_η in (5.22) equals

$$k_\eta := \frac{1}{\int_{w^y \in \mathcal{W}} p(w^y, w^u|\eta)\, df\, dg}, \quad \text{with } \mathcal{W} := \{w^y | A(z) \text{ stable}\}$$

To the purpose of marginalizing over η we consider a non-informative prior[2] $p(\eta)$ on the hyperparameters η. Unfortunately, the "stable" conditional

$$p_S(w^y, w^u|Y, \eta) := \frac{p(Y|w^y, w^u) p_S(w^y, w^u|\eta)}{p_S(Y, \eta)}$$

is not Gaussian and, in addition, the integral in (5.21) cannot be computed in closed form. Hence, we need to rely on a sampling approximation.

Accordingly, Problem 5.2.1 can be formulated as follows.

Problem 5.5.1 (Reformulation)**.** *Employ a Full Bayes Sampling algorithm to obtain a sample form of the "stable" posterior distribution* (5.21)*, composed by the samples* w_i^y, w_i^u*,* $i = [1, \ldots, T]$*. Compute from these samples the estimates* $\hat{w}^y, \hat{w}^u$ *in* (5.5) *which leads to stable* $\hat{G}, \hat{H}$ *in* (5.6)*. This will be done computing sample minimum variance estimate as well as sample maximum a posteriori (MAP) estimate.*

[2] This may be a uniform distribution if the domain is compact.

As discussed in Section 2.4.1 a common manner to implement FBS is to adopt MCMC algorithms. In order to sample from the stable posterior (5.21) one can use a Metropolis-Hasting type of algorithm Gilks et al. (1995), described in Algorithm 12.

Algorithm 12 Metropolis Hastings sampling type

Init: Set (w_1^y, w_1^u) so that w_1^y corresponds to a stable $A(z)$.
Set the proposal distribution to $Q((\cdot,\cdot)|(w_1^y, w_1^u))$
for k = 1 **to** T **do**
  Sample from a proposal distribution $((\tilde{w}^y, \tilde{w}^u) \sim Q((\tilde{w}^y, \tilde{w}^u)|(w_i^y, w_i^u))$
  Set the acceptance probability

$$\alpha := \min\left(1, \frac{p_S(\tilde{w}^y, \tilde{w}^u|Y)Q((w_i^y, w_i^u)|(\tilde{w}^y, \tilde{w}^u))}{p_S(w_i^y, w_i^u|Y)Q((\tilde{w}^y, \tilde{w}^u)|(w_i^y, w_i^u))}\right) \tag{5.23}$$

  if $\alpha > u, \quad u \sim \mathcal{U}(0,1)$ **then**
    set $(w_{k+1}^y, w_{k+1}^u) = (\tilde{w}^y, \tilde{w}^u)$
  else
    set $(w_{k+1}^y, w_{k+1}^u) = (w_i^y, w_i^u)$.

Algorithm 12 relies on the possibility of evaluating the stable posteriori which is not available in closed form. Hence, two fundamental issues need to be addressed for this algorithm to be implementable, namely:

(i) Design the proposal density $Q((\tilde{w}^y, \tilde{w}^u)|(w_i^y, w_i^u))$

(ii) Compute the posterior, up to a constant multiplicative factor[3] $p_S(w^y, w^u|Y)$

Moreover, a preliminary step for both items (i) and (ii) is required:

(∘) the computation of a set of samples $\eta_i \sim p(\eta|Y)$ from the posterior of the hyperparameters, without accounting for the stability constraint.

These three issues are addressed in the following.

(∘) Hyperparameters posterior density $p(\eta|Y)$

The goal is to draw points from the posterior density of η given Y. By applying the Bayes' rule the posterior can be written as:

$$p(\eta|Y) = \frac{p_\eta(Y)p(\eta)}{p(Y)} \tag{5.24}$$

[3]This is because only ratios of posterior probabilities need be computed in (5.35).

where, as mentioned earlier on, $p(\eta)$ is assumed to be a non-informative prior distribution, and $p(Y)$ is the normalization constant. The marginal density $p_\eta(Y)$ of Y given η can be computed in a closed form, as discussed in Pillonetto et al. (2011a) and is given by

$$p_\eta(Y) = \exp\left(-\frac{1}{2}\ln(\det[2\pi\Sigma_Y(\eta)]) - \frac{1}{2}Y^\top\left(\Sigma_Y(\eta)\right)^{-1}Y\right) \tag{5.25}$$

where

$$\Sigma_Y(\eta) = \Phi_Y K_\eta \Phi_Y^\top + \Phi_U K_\eta \Phi_U^\top + \sigma^2 I_N \tag{5.26}$$

and $\sigma^2 := Var\{e(t)\}$ is the variance of the innovation process and Φ_Y, Φ_U are regressors matrix computed as (2.13) with $\phi_Y(t) = [-y(t-1)\ldots - y(t-n)]^\top \in \mathbb{R}^{n\times 1}$ and $\phi_U(t) = [u(t-n)]^\top \in \mathbb{R}^{n\times 1}$.

In order to obtain samples from (5.24) a Metropolis-Hasting algorithm as the one described in Algorithm 12 is implemented. As previously mentioned, the implementation of the sampling algorithm requires the proposal and the target in order to evaluate the acceptance rate. The proposal is a symmetric distribution $q_\eta(\cdot|\cdot)$ which describes a random walk in the hyperparameter space, whose mean is centred at the current value and its variance contains information about the local curvature of the target. To do so, let us define:

$$\underline{\eta} = \arg\min_\eta -\ln[p_\eta(Y)p(\eta)] \tag{5.27}$$

and

$$H = -\frac{d^2\ln[p_{\underline{\eta}}(Y)p(\underline{\eta})]}{d\eta d\eta^\top} \tag{5.28}$$

the Hessian matrix evaluated in $\underline{\eta}$.
Thus, the proposal distribution is defined as:

$$q_\eta(\cdot|\mu) = \mathcal{N}(\mu, \gamma H^{-1}) \tag{5.29}$$

where γ is a positive scalar chosen to obtain an acceptance probability in the MCMC algorithm around the 30% via a pilot analysis, see e.g. Roberts, Gelkman, and Gilks (1997).
In the cases where the covariance γH^{-1} results to be inefficient to explore the support, it can be exchanged by computing the sample covariance from the samples accepted during an additional pilot analysis (e.g. starting from a diagonal covariance, obtaining a certain amount of samples and computing the sample covariance of these samples).

The acceptance rate of the MCMC results to be:

$$\alpha_{\eta_i} = \min\left(1, \frac{p_{\eta_i}(Y)p(\eta_i)}{p_{\eta_{i-1}}(Y)p(\eta_{i-1})}\right) \tag{5.30}$$

which can be evaluated.

The MCMC to obtain hyperparameters distributed as a sampled form of (5.24) is now characterized. Points (i) and(ii) can be addressed.

(i) Proposal density

It is well known in the MCMC literature that an accurate choice of the proposal distribution may have a remarkable impact on the performance (burn in time) of the Markov Chain. Here, a data-driven proposal is obtained from the posterior distribution disregarding the stability constraint. The algorithm is based on the FBS approximation presented in Section 2.4.1 and here reported:

$$p(w^y, w^u|Y) = \int_\eta p(w^y, w^u|Y, \eta)p(\eta|Y)\, d\eta \simeq \frac{1}{T}\sum_{i=1}^{T} p_{\eta_i}(w^y, w^u|Y) \tag{5.31}$$

where η_i, $i = 1, .., T$ are the samples from $p(\eta|Y)$ built with the MCMC algorithm described in point ($\circ$) and

$$p_{\eta_i}(w^y, w^u|Y) \sim \mathcal{N}\left(\mu_{\eta_i}^{MAP}, \Sigma_{\eta_i}^{MAP}\right) \tag{5.32}$$

is the (Gaussian) posterior density of w^y, w^u when the hyperparameters are fixed equal to η_i. The posterior means and variance are, respectively:

$$\begin{aligned}
\mu_\eta^{MAP} &:= (\mathbb{E}_\eta[w^y|Y], \mathbb{E}_\eta[w^u|Y]) \\
&\quad \mathbb{E}_\eta[w^y|Y] = K_\eta \Phi_Y^\top \left(\Sigma_Y(\eta)\right)^{-1} y \\
&\quad \mathbb{E}_\eta[w^u|Y] = K_\eta \Phi_U^\top \left(\Sigma_Y(\eta)\right)^{-1} y \\
\Sigma_\eta^{MAP} &= \underline{K}_\eta - \underline{K}_\eta \begin{bmatrix} \Phi_Y^\top \\ \Phi_U^\top \end{bmatrix} \left(\Sigma_Y(\eta)\right)^{-1} \begin{bmatrix} \Phi_Y & \Phi_U \end{bmatrix} \underline{K}_\eta \\
&\quad \underline{K}_\eta := \begin{bmatrix} K_\eta & \underline{O} \\ \underline{O} & K_\eta \end{bmatrix}
\end{aligned} \tag{5.33}$$

and $\Sigma_Y(\eta)$ is defined in (5.26).

Summarizing, equations (5.33) allow to compute the terms inside the sum in (5.31) once the hyperparameters η_i are given. Consequently, in order to sample from the proposal density $p(w^y, w^u|Y)$ one can

1. Sample $\eta_i \sim p(\eta|Y)$ as described in point $(\circ)$,
2. Sample $(w^y, w^u) \sim p_{\eta_i}(w^y, w^u|Y)$ in (5.32).

(ii) Evalution of the stable posterior $p_S(w^y, w^u|Y)$

The stable posterior in equation (5.21) can be approximated as follows:

$$\begin{aligned} p_S(w^y, w^u|Y) &= \int p_S(w^y, w^u, \eta|Y)\, d\eta \\ &= \tfrac{1}{p(Y)} \int p(Y|w^y, w^u) p_S(w^y, w^u|\eta) p(\eta)\, d\eta \\ &= \tfrac{1}{p(Y)} \int p(Y|w^y, w^u) p_S(w^y, w^u|\eta) p(\eta) \tfrac{q(\eta)}{q(\eta)}\, d\eta \\ &\simeq \tfrac{1}{Tp(Y)} \sum_{i=1}^{T} \tfrac{p(Y|w^y, w^u) p_S(w^y, w^u|\eta_i) p(\eta_i)}{q(\eta_i)} \end{aligned} \tag{5.34}$$

where $\eta_i \sim q(\eta)$. Notice that the quantities $p(Y|w^y, w^u)$, $p_S(w^y, w^u|\eta)$ and $p(\eta)$ can be evaluated and the stable posterior $p_S(w^y, w^u|Y)$ can then be approximated (up to the irrelevant normalization constant $\frac{1}{Tp(Y)}$) from equation (5.34).

Thus, setting $q(\eta) := p(\eta|Y)$ and using the MCMC algorithm described in (i) to obtain samples from the posterior $p(\eta|Y)$, it is possible to sample from the stable posterior $p_S(w^y, w^u|Y)$ (5.21). Algorithm provides our solution to Problem 5.5.1.

Note that, from (5.37), an estimate of $G(z)$ is obtained directly. This is to guarantee that $G(z)$ is stable since the average $\sum_i \hat{G}_i(z)$ of BIBO stable function is BIBO stable. On the other hand, if one would have averaged[4] the w_i^y directly, there would be no guarantee that the average would lead to a stable $A(z)$ (and thus a stable model). Of course, if needed, an estimate of $W^y(z)$ can be obtained from (5.2) using $\hat{G}$ and $\hat{H}$ in (5.37) :

$$\hat{W}^u(z) := \hat{H}^{-1}(z)\hat{G}(z)$$
$$\hat{W}^y(z) := 1 - \hat{H}^{-1}(z)$$

[4]Recall that the average of stable polynomial is not necessarily a stable polynomial unless the degree is smaller than 3, see Gora (2007).

Algorithm 13 Stabilization via a Full Bayes Sampling Approach

Hyperparameters Sampling: a Metropolis-Hastings approach

Init: set $\eta_0 = \underline{\eta}$ using (5.27)
  Set the proposal distribution to $q_\eta(\cdot|\eta_0) \sim \mathcal{N}(\eta_0, \gamma H^{-1}))$
for k = 1 **to** $T + N_{burnIn}$ **do**
  Sample η from $q_\eta(\cdot|\eta_{i-1}) \sim \mathcal{N}(\eta_{i-1}, \gamma H^{-1}))$
  Set the acceptance probability

$$\alpha := \min\left(1, \frac{p_{\eta_i}(Y)p(\eta_i)}{p_{\eta_{i-1}}(Y)p(\eta_{i-1})}\right) \tag{5.35}$$

  if $\alpha > u, \quad u \sim \mathcal{U}(0,1)$ **then**
    set $\eta_i = \eta$,
  else
    set $\eta_i = \eta_{i-1}$
Retain the last T samples η_i which are (approximately) i.i.d. samples from $p(\eta|Y)$.

Predictor Impulse Response Estimate: an MCMC approach

Init: Consider the η_i with $i = [1, T]$ obtained above
  Compute $[w_0^y, w_0^u]$ from η_0 using (5.32)
for k = 1 **to** T **do**
  Compute μ_i^{MAP}, Σ_i^{MAP} with η fixed to η_i as in (5.33),
  Sample $(\tilde{w}^y, \tilde{w}^u) \sim \mathcal{N}(\mu_i^{MAP}, \Sigma_i^{MAP})$
  Compute the acceptance probability

$$\alpha := \min\left(1, \frac{p_S(\tilde{w}^y, \tilde{w}^u|Y)p(w_i^y, w_i^u|Y)}{p_S(w_i^y, w_i^u|Y)p(\tilde{w}^y, \tilde{w}^u|Y)}\right) \tag{5.36}$$

with $p_S(w^y, w^u|Y)$ and $p_S(w^y, w^u|Y)$ approximated as in (5.34) and (5.31),
  if $\alpha > u, \quad u \sim \mathcal{U}(0,1)$ **then**
    set $(w_i^y, w_i^u) = (\tilde{w}^y, \tilde{w}^u)$,
  else
    set $(w_i^y, w_i^u) = (w_{i-1}^y, w_{i-1}^u)$.
Samples (w_i^y, w_i^u) are i.i.d. samples from $p_S(w^y, w^u|Y)$ as requested by Problem 5.5.1.
Compute Minimum Variance and MAP estimators of the forward model (5.1):

- **Minimum Variance Estimate**: Compute the averages of the impulse responses g_i and h_i obtained from each sample (w_i^y, w_i^u) as in (5.6)

$$\hat{g} := \frac{1}{T}\sum_{i=1}^{T} g_i, \quad \hat{h} := \frac{1}{T}\sum_{i=1}^{T} h_i \tag{5.37}$$

  in the remaining of the paper the model obtained by (5.6) using (5.37) will be called "MCMC posterior mean" model.

- **Maximum a Posteriori Estimate**

$$\bar{w}^y, \bar{w}^u = \arg\max_{w_i^y, w_i^u} p_S(w^y, w^u|Y) \tag{5.38}$$

  in the remaining of the paper the model obtained by (5.6) using (5.38) will be called "MCMC MAP" model.

5.6 Simulations Results

The performances of the models obtained from the four[5] techniques described in Section 5.3, 5.4 and 5.5 are compared by means of a Monte Carlo experiment. Recall that the acronyms for the estimates associated to the stabilization technique are "LMI", "ML+PF", "MCMC Posterior Mean" and "MCMC MAP".

At each Monte Carlo run, the aim is to estimate stable ARMAX models from a training data set, generated by a marginally stable model, using a NPPEM approach equipped with the four stabilization techniques. By marginally stable model we mean that its poles are close to the complex unit circle.

A comparison on the performance is given in terms of both prediction quality of new data and of reconstruction of the "true" impulse response.

Generation of Marginally Stable Model

At each Monte Carlo run a 2^{nd}-order SISO ARMAX model, called M, is generated:

$$A(z)y(t) = kz^{-1}B(z)u(t) + C(z)e(t) \tag{5.39}$$

The two complex conjugate roots of the monic polynomial $A(z)$ are placed in $0.996 \cdot \exp(\pm j\frac{\pi}{3})$, $B(z)$ is a random polynomial whose roots are restricted to lie inside the circle of radius 0.9 and $C(z)$ has randomly roots with absolute value chosen in the interval $[0.65, 0.73]$ in order to ensure that the predictor impulse responses decay in no more then 30 steps. The roots of $C(z)$ have a restricted domain only for simulation purposes, but in principle they could be anywhere in the unit circle.

The system input $u(t)$ and the disturbance noise $e(t)$ are independent white noise with unit variance (for both identification and test data sets). The constant k is designed so that the signal-to-noise ratio of (5.39) equals 1. More specifically, let $y_u(t) := \frac{B(z)}{A(z)}u(t)$ and $y_e(t) := \frac{C(z)}{A(z)}e(t)$, then k is set to $k := \sqrt{\frac{var(y_e)}{var(y_u)}}$, where with the symbol $var(x)$ we mean the sample variance of the vector x.

Experiment description

A Monte Carlo study of 5000 runs is implemented. At each run, a model of the form of (5.39) is used to generate a training set of 400 samples and a test set of 1000 samples.

The predictor impulse responses w^y and w^u are estimated via an Empirical Bayes approach described in (2.4.4) , using the Stable Spline Kernel, Pillonetto and De Nicolao

[5]Recall that in 5.5 two estimators (MAP and Posterior Mean) are considered.

(2010), as a priori covariance matrix and determining the hyperparameters by marginal likelihood optimization (see Section 2.4.3).

The predictor impulse responses coefficients become negligible after 30 temporal lags for how model (5.39) has been designed. Hence, the predictor impulse responses truncation length is set to $n = 30$. The noise variance, σ^2, is computed via a low bias Least Squares identification method.

The system transfer functions in (5.6) obtained from the Empirical Bayes estimates w^y_{EB} and w^u_{EB} ended up to be unstable about 150 times out of the 5000 Monte Carlo runs. In these cases the stabilisation procedures have been applied. Thus, our Monte Carlo analysis is limited to these 150 data sets which resulted in unstable forward model estimates. All these 150 models became stable after applying any of our stabilisation techniques.

The *CVX toolbox*, Grant et al. (2006) based on *YALMIP*, has been used in Matlab to solve the convex optimization problem (5.13), with solver SeDuMi, Sturm (1999). Instead, the optimization problem (5.18) has been solved using the Matlab function `fminsearch.m`.

Performance

The experimental results are measured in terms of both capability of predicting new data and reconstruction of the true impulse responses. A comparison among the different techniques follow each performance index.

At each Monte Carlo run, after the predictor estimates $\hat{w}^y$ and $\hat{w}^u$, solutions of Problem 5.2.1 are obtained with the four stabilization techniques, the prediction quality of the one-step-ahead prediction $\hat{y}(t|t^-)$ is evaluated by means of the one-step-ahead coefficient of determination:

$$COD^{(i)} = 100\left(1 - \sqrt{\frac{\sum_{t=1}^{1000}\left(y_{test}(t) - \hat{y}^{(i)}(t|t^-)\right)^2}{\sum_{t=1}^{1000} y^2_{test}(t)}}\right) \tag{5.40}$$

where i indicates the i-th Monte Carlo run.

Figure 5.2 displays the boxplots of the $\{COD^{(i)}\}$ obtained through all the Monte Carlo runs by any of the four stabilization techniques as well as for the "true" model, M.

Index (5.40) obtained from the "true" model is used as a reference in the prediction performance and the comparison shows that all the models identified perform remarkably well, with no significant differences. This shows that the artefacts introduced by the stabilization techniques into the Bayesian estimates do not affect negatively the prediction

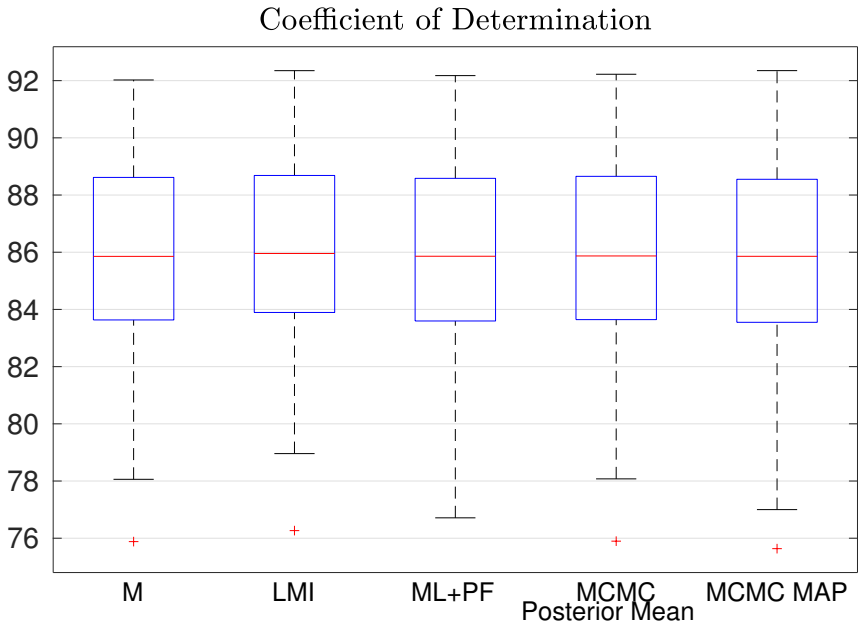

Figure 5.2: Monte Carlo results. Boxplots of the $\{COD^{(i)}\}$ obtained from the one-step ahead predictor of the "true" model, M, and from the one-step ahead predictors estimated with the four stabilization techniques.

performance.

The major interest of this work is in the quality of the impulse responses estimation of the forward model after the four stabilization techniques have been applied. In order to understand it better, previously, the dominant poles of the estimated forward models are analysed. In particular, we are interested in the absolute values of the dominant poles.

In Figure 5.3 the boxplots show the absolute values of the dominant poles of the stabilized models in the Monte Carlo experiment, the horizontal line in 0.996 indicates the one of the "true" models. Not surprisingly, all the estimation methods tend to place the poles closer to the unit circle than 0.996.

Loosely speaking, the Bayesian estimates were unstable, which means that the estimators were lead by the data to place the dominant pole outside the unit circle and the proposed stabilization techniques suggest solutions with the dominant pole placed in between the unstable and the true pole. In particular, "ML+PF" and "MCMC Posterior Mean" estimate the poles almost at the border of the unit circle, while "MCMC MAP" has the best approximation to the true dominant pole.

Finally, the estimated impulse responses can be compared to the true ones in terms

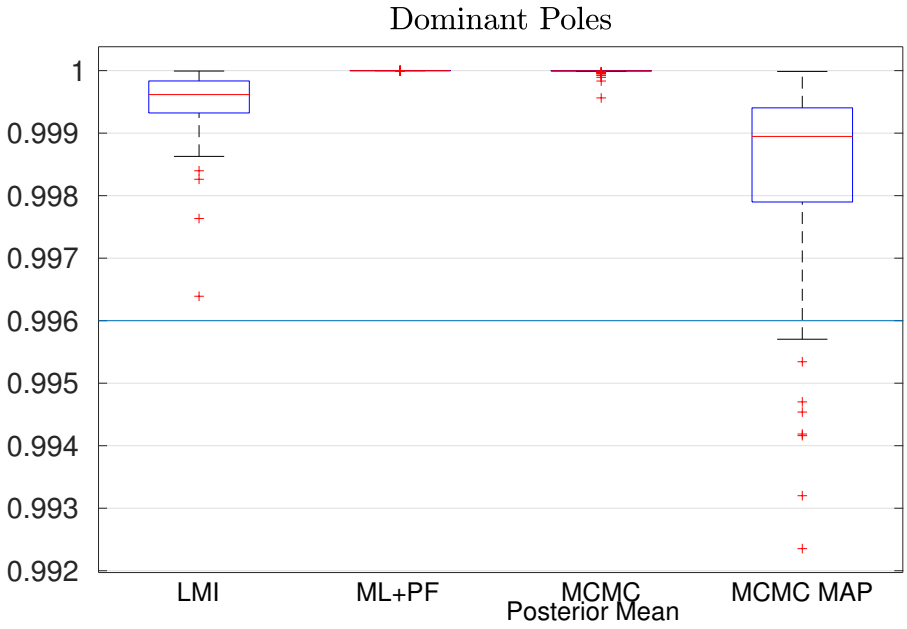

Figure 5.3: Monte Carlo results. Boxplots of the absolute value of the dominant poles of the identified models. The horizontal line represents the absolute value of the dominant pole of the true model.

.

of relative errors:

$$\mathrm{err}^{(i)} = \frac{1}{2}\sqrt{\frac{\sum_{t=1}^{1000}\left(g(k)-\hat{g}^{(i)}(k)\right)^2}{\sum_{t=1}^{1000} g^2(k)}} + \frac{1}{2}\sqrt{\frac{\sum_{t=1}^{1000}\left(h(k)-\hat{h}^{(i)}(k)\right)^2}{\sum_{t=1}^{1000} h^2(k)}} \tag{5.41}$$

where $\{\hat{g}^{(i)}\}$ and $\{\hat{h}^{(i)}\}$ are the estimators of the (truncated) impulse responses $\{g\}$ and $\{h\}$ of the true forward model (5.1).

Figure 5.4 reports the boxplots of the impulse responses relative error in the Monte Carlo experiment for the forward models estimated with the four stabilization techniques. The "MCMC Posterior Mean" estimator significantly outperforms all the others. The remaining three techniques achieve rather poor performance in the identification of the system and the cause can be researched in the estimation of the dominant modes. Indeed, a higher absolute value of the dominant pole corresponds to a slower decay rate of the impulse responses. The estimators place the dominant poles very close to the unit circle, see Figure 5.3, which results in long tailed impulse responses that in turn yield a high relative error.

The algorithm "MCMC Posterior Mean" deserves a separate discussion. In this case,

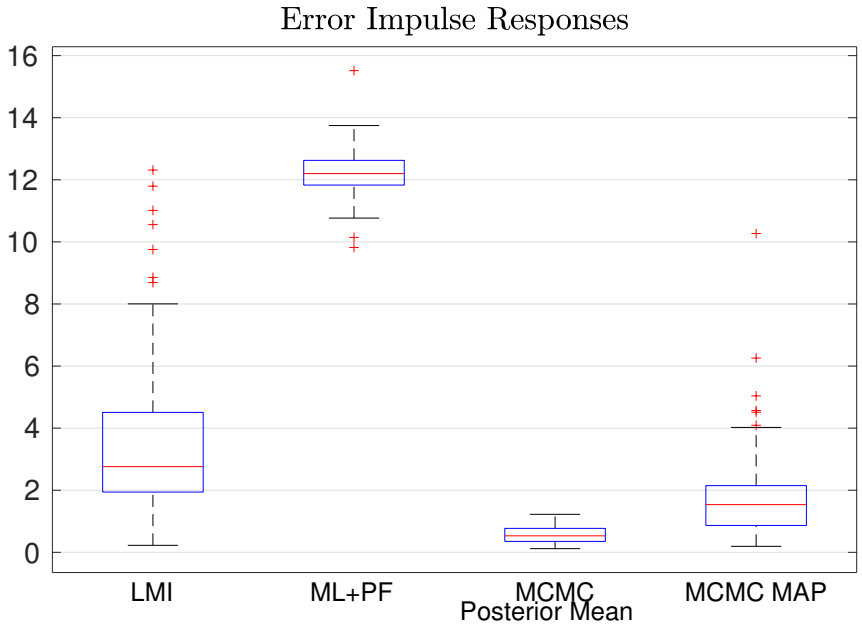

Figure 5.4: Monte Carlo results. Boxplots of the $\{\text{err}^{(i)}\}$. Quantity to qualify the reconstruction of the system impulse responses.

since the estimator $\hat{G}(z)$ is the average of all samples $\hat{G}^{(i)}(z)$ obtained by the MCMC algorithm, see (5.37), the dominant pole of $\hat{G}(z)$ is the slowest among the dominant poles of the $\hat{G}^{(i)}(z)$ with $i = [1, T]$. Indeed, in Figure 5.3 the "MCMC Posterior Mean" procedure estimates among the highest absolute values of the dominant pole w.r.t. the others procedures. Yet, the effect of these dominant modes in the average (5.37) is mitigated by the factor $1/T$, which have a shrinking effect into the impulse responses tails. In turn, the estimated impulse responses well approximate functions with a decay rate inferior to the one suggested by the dominant modes in Figure 5.3 and the relative error (5.41) for this technique performs satisfactorily, as it can be seen in Figure 5.4.

5.7 Conclusions

Four different techniques have been presented to face the problem of identifying a stable system using Gaussian regression based on the minimization of the predictor error. The Monte Carlo experiment shows that the proposed techniques perform remarkably well in the prediction of new data. The problem arises while reconstructing the system impulse responses due to the tendency of estimating the dominant modes too close to the border of the unit circle, thus inducing an excessively slow decay of the impulse responses. Only the model estimated with the so called "MCMC Posterior Mean" technique performs

satisfactorily in terms of impulse response fit.

Future works

The computational load required by the proposed solution might be problematic in some application. New techniques to overcome the stability problem of NPPEM approaches could be designed without the usage of an MCMC algorithm.

In the authors opinion, it would be of particular interest to formalize a new regularization problem which takes into account penalty terms both in the predictor and in the forward model impulse responses. Some ideas have been developed by the authors but should be further researched and therefore are omitted here.

6

Online semiparametric learning for inverse dynamics modeling

Humanoid robots are sophisticated platforms controlled by a mathematical model to relate to the actuator inputs to the interactions with the external world read by sensor feedbacks. This model is called the dynamic model of the robot.

Inverse dynamics models are one possibility to describe the dynamics of the robot. These models are very useful in robotic applications because they can be used in model-based control applications to improve the tracking performances leading to high accuracy and low control gains Craig (2005); Nguyen-Tuong et al. (2009), see the survey Nguyen-Tuong and Peters (2011b) for a comprehensive overview.

Typically, inverse dynamic models can be obtained from physics first principles, using the techniques of rigid body dynamics (RBD), Siciliano, Sciavicco, Villani, and Oriolo (2010). This approach results in a parametric model in which the values of physically meaningful parameters must be provided in order to complete the fixed structure of

the model. Building an inverse dynamics model from the first principles might be very demanding and, in most cases, out of reach and not suitable for online applications. For this reason the model can be achieved from data and framed as a parametric estimation problem, Hollerbach, Khalil, and Gautier (2008); Siciliano et al. (2010). The main advantage of the parametric approach is that in principle it provides a global relationship between the input (joint angles, velocities and accelerations) and the output (torques). However, the linear model does not capture nonlinearities in the data.

Alternatively, the dynamical model can be obtained from experimental data using machine learning techniques, resulting in a nonparametric model. With respect to the parametric approach, the nonparametric modeling has the advantage of not requiring any unrealistic assumption on the physical system, such as rigidity of the links or a simplistic modeling of the friction; indeed, it can model the dynamics by extrapolating the input-output relationship directly from the available data.

In order to exploit the advantages of both estimation techniques, semiparametric models have been recently introduced as a combination of RBD and nonparametric models, as for instance in Nguyen-Tuong and Peters (2010); Wu and Movellan (2012). Two main alternatives are possible. The first one is to embed the rigid body dynamics as a "mean" in the nonparametric part. The second one is to incorporate the rigid body dynamics in the kernel function.

An important aspect in the inverse dynamics learning is the variation of the mechanical properties caused by the change of the tasks. It is then necessary to update the model online. In this framework, it is important that the online algorithm is able to take advantage of the knowledge already acquired from previously available data, thus speeding up the learning process. This concept is often called transfer learning Pan and Yang (2010); Bocsi, Csató, and Peters (2013). Several online learning algorithms have been proposed in the literature and the interest reader is referred to Section 2.4.5 for an overview. In addition, we mention the semiparametric algorithms based on the locally weighted projection approach, De la Cruz, Kulic, Owen, Calisgan, and Croft (2012), where the parametric approach is used to initialize the nonparametric algorithm and Camoriano, Traversaro, Rosasco, Metta, and Nori (2016), where a semiparametric online algorithm, based on the approximation of the kernel function using the so called "random features", Rahimi and Recht (2007); Quinonero-Candela and Rasmussen (2005), has been proposed.

Contributions

The main contributions of this research are the following.

1. Various semiparametric models proposed in the literature Nguyen-Tuong and Peters (2010); Wu and Movellan (2012); Camoriano et al. (2016) are shown to be sub-cases of a unique general model, and an online algorithm is provided for this general model, exploiting the random features approximation.

2. The online algorithm is used to compare the various modeling approaches (parametric, nonparametric, semiparametric) for estimating the inverse dynamics of one arm of the iCub humanoid robot, using real data. In doing so, two different approaches for estimating the hyperparameters (the parameters in the nonparametric approach), namely the marginal likelihood maximization and the cross validation, are compared. These approaches are discussed in Section 2.4.3.

3. Joint positions, velocities and accelerations are the input locations suggested by the physics in order to describe the inverse dynamics model. However, joint velocities and accelerations cannot be measured in the majority of the experiments and are approximated by numerical differentiation of the joint positions. This brings significant numerical errors into the model. We propose to replace joint velocities and accelerations by linear combination of past temporal lags of the joint positions, thus learning the weight of the linear combination directly from data.

The chapter is organized as follows. In Section 6.1 the problem of inverse dynamic estimation is formalized. In Section 6.2 parametric, nonparametric and semiparametric models are introduced while in Section 6.3 these models are approximated to linear models for which an online algorithm is available. Section 6.4 introduces learning methods to avoid the numerical errors that might arise in computing the derivatives of the joint positions. In Section 6.5 the different online algorithms are tested in the inverse dynamics estimation of the robotic platform iCub. Finally, in Section 6.6 conclusions and future works are drawn.

6.1 Problem Statement

Background

The dynamics of a robot (Siciliano et al., 2010) with n_{dof} degrees of freedom (DOF), or more in general of a dynamical system (Taylor, 2005), can be expressed as

$$\tau(s) = M(q(s))\ddot{q}(s) + h(q(s), \dot{q}(s)) \,, \tag{6.1}$$

where $\tau(s) \in \mathbb{R}^{n_{\text{dof}}}$ is the torque applied to the robot joints, $q(s), \dot{q}(s), \ddot{q}(s) \in \mathbb{R}^{n_{\text{dof}}}$ are the joint positions, velocities and accelerations, $M(q(s)) \in \mathbb{R}^{n_{\text{dof}} \times n_{\text{dof}}}$ represents the generalized inertia matrix and $h(q(s), \dot{q}(s)) \in \mathbb{R}^{n_{\text{dof}}}$ accounts for the modeled contributions of Coriolis, centrifugal, centripetal and gravitational and viscous, static and Coulomb frictions, at time $s \in \mathbb{Z}$.

Starting from the laws of physics it would in principle be possible to write a (direct) dynamical model which, having as inputs the torques acting on the robot joints, outputs the (sampled) trajectory of the free coordinates (joint angles) $q(s)$, $s \in \mathbb{Z}$. This is the so called "direct dynamics".

However, for the purpose of control design, it is of interest to know which torques should be applied in order to obtain a certain trajectory $q(s)$. This is the purpose of the inverse dynamics modeling: *finding a model which, having joint trajectories as inputs, outputs the applied torques.*

A common assumption for simplifying the modeling exercise is to measure not only the joint positions $q(s)$ but also joint velocities $\dot{q}(s)$ and joint accelerations $\ddot{q}(s)$ even when it is not possible for the robot to sense them. For this reason, velocities and accelerations are computed by numerical differentiation. Clearly, this can be a rather crude approximation. However, this assumption simplifies considerably the modeling exercise because, given $q(s), \dot{q}(s), \ddot{q}(s)$, the inverse dynamics model is, in principle, linear (as it will be shown in model (6.6)).

Inverse Dynamics Estimation

Consider the discrete nonlinear time invariant causal model

$$\mathcal{M}: \quad y(s) = f^*(x(s)) + e(s) \quad s \in \mathbb{Z} \tag{6.2}$$

where $y(s), x(s) \in \mathbb{R}^{n_{\text{dof}}}$ are the output and input measures at time s and $e(s)$ is a zero mean white Gaussian noise with unknown variance $\sigma^2 I_{n_{\text{dof}}}$. The inputs in this Chapter are denoted with the symbol x instead of u for connection with the machine learning literature and they will be called input locations, see e.g. Rasmussen and Williams (2006).

Model (6.2) can be considered for solving the problem of inverse dynamics estimation.

The problem of learning the inverse dynamics becomes the problem of *estimating* the model $\mathcal{M}$ (i.e. the function f^*) starting from a finite set of measured data samples $\{y(s), x(s)\}_{s=1}^{N}$. Furthermore, we are interested in performing the inverse dynamics estimation in an online framework. Suppose that the robot is currently performing a

trajectory and at each sampling interval a new pair of data becomes available, namely, the desired next point in the trajectory $x^d(s)$ and the torques applied to the joints at the previous time instant, $y(s-1)$, read by the sensors. Based on these informations we are interested in updating the inverse dynamics model within the sampling time interval.

In the first part of the treatise, "standard" quantities are considered as inputs and outputs: the outputs $y(s)$ are the torques applied to the n_{dof} joints of the robot at time s and $x(s) = [q(s)^\top \ \dot{q}(s)^\top \ \ddot{q}(s)^\top]^\top \in \mathbb{R}^m$, with $m = 3n_{\text{dof}}$, denotes the vector "input locations" obtained by stacking positions, velocities and accelerations of all the n_{dof} joints of the robot. However, as mentioned in the Background part, we consider that the joint velocities and accelerations cannot be sensed from the robot and they have been computed through pre-processing filters. It is well known that applying numerical differentiation operations to noisy data lead to an increasing amount of error in the derivative signals. Consequently, joint velocities and accelerations might not be meaningful data. We analyze the possibility of considering only the joint positions and learn some quantities from the measurable data to replace the numerical derivatives and accelerations. These quantities will be called features and they will be treated in Section 6.4 and in the experiments, in the other sections the input locations will be considered as $x(s)$.

The motivations and the challenges of tackling this identification problem are both from a theoretical point of view, because there is a nonlinear possibly very complicated function that has to be estimated from physically meaningful data and from a practical point of view, because the inverse dynamics model can then be used for robot motion control, see Figure 6.1.

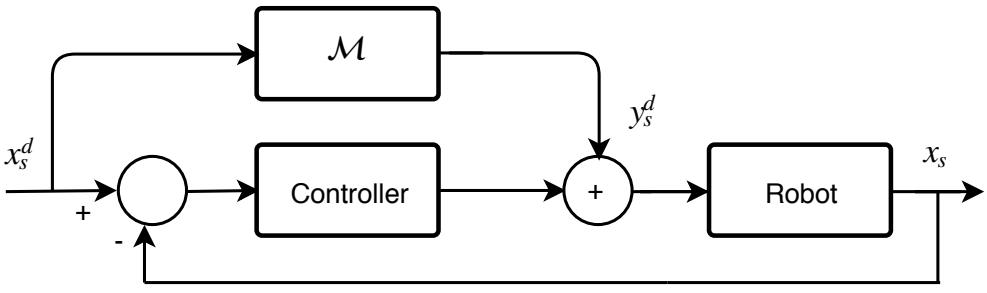

Figure 6.1: Schematic for robot motion control.

More precisely, inverse dynamics is exploited to determine the feedforward joint torques $y(s)^d$ which should be applied to follow a desired trajectory $x(s)^d$, while employing a feedback controller in order to stabilize the system. Clearly, the more accurate the inverse dynamics model $\mathcal{M}$, the more accurate the motion control is. In this work, several approaches, depending on how the function $f^*(\cdot)$ in (6.2) is modelled, are considered.

6.2 Semiparametric Models

The semiparametric models are a combination of models that contain physical knowledge of the true system (the parametric part) and of data-driven models (the nonparametric part).

In the following, several approaches to combine the global knowledge of the RBD model and the flexibility of the NPPEM approaches are described. Moreover, it is shown how these semiparametric models (and also parametric and nonparametric as particular cases of them) can be recast in the Gaussian regression framework.

Specifically, consider $f : \mathbb{R}^m \to \mathbb{R}^{n_{\text{dof}}}$ as the hypothesis we make to describe the true system (6.2):

$$y(s) = f(x(s)) + e(s) \tag{6.3}$$

where $e(s)$ is a zero mean Gaussian noise with covariance $\sigma^2 I_{n_{\text{dof}}}$ and f is a Gaussian process

$$f(\cdot) \sim \mathcal{N}\left(\mu(\cdot), \widetilde{K}(\cdot,\cdot)\right) \tag{6.4}$$

with $\mu(\cdot) := \mathbb{E}[f(\cdot)]$ the mean function and $\widetilde{K}(\cdot,\cdot) := cov\left(g(\cdot), g(\cdot)\right)$ the positive definite covariance matrix, also called kernel. See Section 2.4 for more details.

We shall show how all the semiparametric models can be described by appropriate choice of the mean and of the covariance function of (6.4).

Linear Parametric Model

The rigid body dynamics (RBD) of a robot is described by the equation

$$y(s) = M(q(s))\ddot{q}(s) + C(q(s), \dot{q}(s))\ddot{q}(s) + G(q(s)) \tag{6.5}$$

where $M(q(s))$ is the inertia matrix of the robot, $C(q(s), \dot{q}(s))$ the Coriolis and centripetal forces and $G(q(s))$ the gravity forces, Siciliano et al. (2010). The terms on the right hand side of (6.5) can be rewritten as $\psi^\top(x(s))\pi$ which is linear in the robot (base) inertial parameters $\pi \in \mathbb{R}^p$ and where $\psi(x(s)) : \mathbb{R}^m \to \mathbb{R}^{p \times n_{\text{dof}}}$ is the known RBD regressor which is a nonlinear function of the joint trajectories described by $x(s)$. Therefore, the RBD model becomes

$$y(s) = \psi^\top(x(s))\pi + e(s) \tag{6.6}$$

where $e(s)$ is a zero-mean Gaussian noise with covariance matrix $\sigma^2 I_{n_{\text{dof}}}$ and it includes

the nonlinearities of the robot that are not modeled in the rigid body dynamics (e.g. actuator dynamics, frictions, etc.).

A known issues of this model (see e.g., Hollerbach et al. (2008)) is that the problem of determining π from measured data $y(s)$ is usually ill posed and the matrix $\psi(x(s))$ is rank deficient. Possible solutions would be to design either efficient experiments to collect data sufficiently rich to excite all the modes of the system or dedicated experiments which are good to estimate parameters separately. Another possible solution is to add a regularization term in the estimation problem, which is equivalent to modeling π as a Gaussian random vector in a Bayesian view. The last option is here considered and the following prior distribution is assigned to π :

$$\pi \sim \mathcal{N}(0_p, \gamma^2 I_p) \tag{6.7}$$

Consequently, model (6.6) has a priori Gaussian distribution such as (6.4) with

$$\begin{aligned} \mu(x) &= 0_p \\ \widetilde{K}(x(t), x(s)) &= \gamma \psi^\top(x(t)) \psi(x(s)). \end{aligned}$$

The covariance $\gamma\psi^\top(x(t))\psi(x(s))$ shall be also called *RBD kernel* or *parametric kernel*.

Nonparametric Model

The robot inverse dynamics is modeled postulating

$$y(s) = g(x(s)) + e(s) \tag{6.8}$$

where $g(x(s))$ is a zero mean vector valued (taking values in $\mathbb{R}^{n_{\text{dof}}}$) Gaussian random process indexed in $\mathbb{R}^m$, with covariance function

$$\mathbb{E}[g(x(t))g(x(s))^\top] = \rho^2 K(x(t), x(s)) I_{n_{\text{dof}}}. \tag{6.9}$$

The hyperparameter ρ^2 plays the role of scaling factor of the kernel function K which shall be called *nonparametric kernel* and has to be chosen by the user.

In this case, there is a straightforward association with (6.4), i.e., $f(x) \sim g(x)$:

$$\mu(x) = 0_p \tag{6.10}$$
$$\widetilde{K}(x(t), x(s)) = \rho^2 K(x(t), x(s)) I_{n_{\text{dof}}}. \tag{6.11}$$

Semiparametric model with RBD mean

Taking in consideration the global property of the parametric model and the flexibility of the data-driven nonparametric models the first idea that might come up to combine the two is to use the former as a mean term and the latter as covariance function.

This is equivalent to model the inverse dynamics as the Gaussian process in (6.4) setting

$$\begin{aligned} \mu(x(s)) &= \mathbb{E}[f(x(s))] = \psi^\top(x(s))\pi \\ \widetilde{K}(x(t), x(s)) &= cov\left(f(x(t)), f(x(s))\right) = \rho^2 K(x(t), x(s)) I_{n_{\text{dof}}} \end{aligned} \tag{6.12}$$

where $\mu(x(s))$ corresponds to the RBD model (6.6), π is the vector of inertial parameters that has to be estimated from the data and $\psi(x(s))$ is the RBD regressor and K is the same as the nonparametric kernel in (6.9).

Semiparametric model with RBD kernel

An alternative possibility for combining the parametric and nonparametric models in model (6.8) is to incorporate the RBD structure in the kernel, Nguyen-Tuong and Peters (2010). This means that the physical knowledge of the system accounts for its variability instead of accounting as a mean term.

Therefore, the inverse dynamics model in (6.4) becomes a random process with zero mean and covariance function

$$\widetilde{K}(x(t)), x(s)) = \mathbb{E}[f(x(t))f(x(s))^\top] = \gamma^2 \psi(x(t))^\top \psi(x(s)) + \rho^2 K(x(t), x(s)) I_{n_{\text{dof}}} \tag{6.13}$$

where the first term $\psi(x(s))$ is the RBD regressor and the second term, K, is again the nonparametric kernel as in (6.9). As before, $e(s)$ is white noise with covariance matrix $\sigma^2 I_{n_{\text{dof}}}$.

6.3 Model Approximation to Regularized Least Squares

The problem of the Gaussian regression framework when applied to estimation or prediction problems is that the estimator depends on the number of data making this

approach intractable in big-data or online applications.

For this reason, it is proposed to approximate the kernel using the so called random features (in Section 6.3.1) which transform the semiparametric models into equivalent linear ones (in Section 6.3.2) and allows a well known recursive update suitable for online settings, described in Section 6.3.3.

6.3.1 Kernel Approximation in Random Features

In robotics, a typical choice is the Gaussian kernel, Gijsberts and Metta (2011); Nguyen-Tuong and Peters (2010); Wu and Movellan (2012); Rasmussen and Williams (2006),

$$K_G(x(t), x(s)) = e^{-\frac{\|x(t)-x(s)\|^2}{2\tau}} \tag{6.14}$$

where τ is the kernel width[1] and it represents the metric to correlate the input locations $x(t)$ and $x(s)$.

The Gaussian kernel has an interesting formulation in terms of Fourier transform. Indeed, as a positive definite real function the kernel is the Fourier transform of a probability density function, Rahimi and Recht (2007). Accordingly, the Gaussian kernel becomes

$$K_G(x(t), x(s)) = \int_{\mathbb{R}^m} p(\omega) e^{i\frac{\omega^\top (x(t)-x(s))}{\tau}} \mathrm{d}\omega \tag{6.15}$$

where $p(\omega)$ is the probability density defined as

$$p(\omega) = \frac{1}{(\sqrt{2\pi})^m} e^{-\frac{\|\omega\|^2}{2}}. \tag{6.16}$$

The integral in (6.15) can be written as an infinite summation of appropriate terms, therefore $K_G(x(t), x(s))$ can be approximated by the sample mean of $e^{i\frac{\omega_k^\top (x(t)-x(s))}{\tau}}$, $k = 1, .., d$ provided $w_k \sim p(\omega)$, that is:

$$K_G(x(t), x(s)) \approx \frac{1}{d} \sum_{k=1}^{d} e^{i\frac{\omega_k^\top (x(t)-x(s))}{\tau}} = \phi(x(t))^\top \phi(x(s)) \tag{6.17}$$

where the basis functions $\phi(x) \in \mathbb{R}^{2d}$ are

$$\phi(x) = \frac{1}{\sqrt{d}} \left[\cos\left(\frac{\omega_1^\top x}{\tau}\right) \quad \dots \quad \cos\left(\frac{\omega_d^\top x}{\tau}\right) \quad \sin\left(\frac{\omega_1^\top x}{\tau}\right) \quad \dots \quad \sin\left(\frac{\omega_d^\top x}{\tau}\right) \right]^\top. \tag{6.18}$$

[1] Therefore, to be precise the function K as well as its approximation (6.17) depends on the parameter τ which will be estimated from data. For simplicity of exposition this dependence is not made explicit in the notation.

This approximation has been introduced in Rahimi and Recht (2007).

6.3.2 Approximated Models

In the following it is shown how the semiparametric models described in Section 6.2 result after applying the approximation in random features to their nonparametric kernels.

Consider a nonparametric Gaussian model with a generic mean function $m(\cdot)$ and covariance function proportional to the Gaussian kernel (6.14):

$$y(s) = f(x(s)) + e(s), \quad f(\cdot) \sim \mathcal{N}\left(m(\cdot), \rho^2 K_G(\cdot,\cdot)\right) \tag{6.19}$$

by approximating K_G with the kernel approximation (6.17) the model (6.19) is equivalent (up to the approximation error) to model

$$y(s) = m(x(s)) + \phi(x(s))^\top \alpha + e(s), \quad \alpha \sim \mathcal{N}\left(0_{2d}, \rho^2 I_{2d}\right) \tag{6.20}$$

where $\phi(x(s))$ is a basis function defined in (6.18) and $\alpha \in \mathbb{R}^{2d}$ is a zero mean random Gaussian vector with covariance function proportional to the identity matrix. Recall that d is the number of the basis functions summed to compute the kernel in (6.17), consequently, its value has to be chosen according to a trade-off between model and computational complexity. We underline that a peculiarity of model (6.20) is that the regressor $\phi(x)$ is depending on the width of the Gaussian Kernel τ which has to be identified from the data.

In the following, the Gaussian kernel (6.14) is postulated for the nonparametric kernel of all the semiparametric models described in Section 6.2. It will be shown for each of them, i.e., (6.6), (6.26), (6.30) and (6.39), that when the approximation in random features (6.17) is used, they can ultimately be written in the form:

$$y(s) = \mu(x(s)) + \varphi(x(s))^\top \theta + e(s) \tag{6.21}$$

for a suitable choice of the mean function $\mu \in \mathbb{R}^{n_{\text{dof}}}$, of the regressor vector $\varphi(x(s))^\top \in \mathbb{R}^{n_{\text{dof}} \times c}$ and of the random vector $\theta \in \mathbb{R}^c$ which is modeled as a zero mean Gaussian vector with a suitable covariance matrix Σ_0. $e(s)$ is white noise with covariance matrix $\sigma^2 I_{n_{\text{dof}}}$.

Recalling the equivalence between the models (6.19) and (6.20), it is only a matter of substitution to obtain the inverse dynamics models as the linear approximation in (6.21).

Linear Parametric Model

Notice that the parametric model in (6.6) does not depend on any nonparametric kernel and it is already in the form of (6.21) therefore there is no need to approximate it. This model is denoted by "P". The analogy with model (6.21) is given by

$$\mu(x(s)) = 0_{n_{\text{dof}}} \in \mathbb{R}^{n_{\text{dof}}} \tag{6.22}$$
$$\varphi(x(s))^\top = \psi^\top(x(s)) \in \mathbb{R}^{n_{\text{dof}} \times p} \tag{6.23}$$
$$\theta = \pi \in \mathbb{R}^p. \tag{6.24}$$

Approximated Nonparametric Model

Applying the kernel approximation (6.17) to (6.9) is equivalent to model $g(x)$ in the form

$$g(x) = (\phi(x)^\top \otimes I_{n_{\text{dof}}})\alpha \tag{6.25}$$

where α is a zero mean Gaussian vector with variance $\rho^2 I_{2dn_{\text{dof}}}$. Therefore, the nonparametric model of the robot inverse dynamics (6.8) can be approximated by

$$y(s) = (\phi(x(s))^\top \otimes I_{n_{\text{dof}}})\alpha + e(s). \tag{6.26}$$

This model is denoted by "NP". The analogy with model (6.21) is given by

$$\mu(x(s)) = 0_{n_{\text{dof}}} \in \mathbb{R}^{n_{\text{dof}}} \tag{6.27}$$
$$\varphi(x(s))^\top = \phi(x(s))^\top \otimes I_{n_{\text{dof}}} \in \mathbb{R}^{n_{\text{dof}} \times 2dn_{\text{dof}}} \tag{6.28}$$
$$\theta = \alpha \in \mathbb{R}^{2dn_{\text{dof}}}. \tag{6.29}$$

Approximated Semiparametric Model with RBD Mean

Approximating, as above, the kernel K in (6.12) with the random features (6.17), the semiparametric model of the inverse dynamics takes the form

$$y(s) = \psi(x(s))^\top \pi + (\phi(x(s))^\top \otimes I_{n_{\text{dof}}})\alpha + e(s) \tag{6.30}$$

where α is a random vector with zero mean and covariance matrix $\rho^2 I_{2dn_{\text{dof}}}$. As before, $e(s)$ is white noise with covariance matrix $\sigma^2 I_{n_{\text{dof}}}$.

At this point two alternatives are possible. The first and most principled one is to treat π as an unknown parameter which has to be estimated e.g., it can be considered as a hyperparameter. In this case model (6.30) is denoted by "SP". The analogy with model (6.21) is given by

$$\mu(x(s)) = \psi^\top(x(s))\pi \in \mathbb{R}^{n_{\text{dof}}} \tag{6.31}$$
$$\varphi(x(s))^\top = \phi(x(s))^\top \otimes I_{n_{\text{dof}}} \in \mathbb{R}^{n_{\text{dof}} \times 2dn_{\text{dof}}} \tag{6.32}$$
$$\theta = \alpha \in \mathbb{R}^{2dn_{\text{dof}}}. \tag{6.33}$$

A suboptimal alternative but often applied in the literature is to assume that π is known, here denoted by $\hat{\pi}$, possibly estimated using some preliminary experiment as in Nguyen-Tuong and Peters (2010) or estimated via Least Squares or it can be given from some expert knowledge. Hence, only the residual vector is to be model, i.e.,

$$\tilde{y}(s) := y(s) - \psi(x(s))^\top \hat{\pi} = (\phi^\top(x(s)) \otimes I_{n_{\text{dof}}})\alpha + e(s). \tag{6.34}$$

This strategy is denoted by "SP2" and it is followed, for instance, in Camoriano et al. (2016), where the vector $\hat{\pi}$ is obtained solving in the least squares sense the regression model (6.6). The analogy with model (6.21) is given by

$$\mu(x(s)) = \psi(x(s))^\top \hat{\pi} \in \mathbb{R}^{n_{\text{dof}}} \tag{6.35}$$
$$\varphi(x(s))^\top = \phi(x(s))^\top \otimes I_{n_{\text{dof}}} \in \mathbb{R}^{n_{\text{dof}} \times 2dn_{\text{dof}}} \tag{6.36}$$
$$\theta = \alpha \in \mathbb{R}^{2dn_{\text{dof}}}. \tag{6.37}$$

Approximated Semiparametric Model with RBD Kernel

Approximating, as above, the kernel K in (6.13) with the random features (6.17), it turns that

$$\mathbb{E}[g(x(t))g(x(s))^\top] = \gamma^2 \psi(x(t))^\top \psi(x(s)) + \rho^2 \phi(x(t))^\top \phi(x(s)) I_{n_{\text{dof}}}. \tag{6.38}$$

Accordingly, the approximated semi-parametric model of the inverse dynamics with RBD kernel is:

$$y(s) = \begin{bmatrix} \psi^\top(x(s)) & \phi^\top(x(s)) \otimes I_{n_{\text{dof}}} \end{bmatrix} \begin{bmatrix} \pi \\ \alpha \end{bmatrix} + e(s) \tag{6.39}$$

where $[\,\pi^\top \ \alpha^\top\,]^\top \in \mathbb{R}^{p+2dn_{\text{dof}}}$ is a zero mean Gaussian random vector with covariance matrix $\text{blkdiag}(\gamma^2 I_p, \rho^2 I_{2dn_{\text{dof}}})$.

The analogy with model (6.21) is given by

$$\mu(x(s)) = 0_{n_{\text{dof}}} \in \mathbb{R}^{n_{\text{dof}}} \tag{6.40}$$

$$\varphi(x(s))^\top = \left[\psi^\top(x(s))\phi^\top(x(s)) \otimes I_{n_{\text{dof}}}\right] \in \mathbb{R}^{n_{\text{dof}} \times p+2dn_{\text{dof}}} \tag{6.41}$$

$$\theta = \begin{bmatrix} \pi \\ \alpha \end{bmatrix} \in \mathbb{R}^{p+2dn_{\text{dof}}}. \tag{6.42}$$

The semiparametric model with RBD kernel described in this Section is connected, under the Bayesian framework, with the RBD mean model. This analogy is stated in Proposition 6.3.1 and beyond being interesting from a theoretical point of view it can be also useful for the purpose of computing the estimator for model (6.30).

Proposition 6.3.1. *The estimate of model* (6.30) *is equivalent to the estimate of model* (6.39) *when* $\gamma \to \infty$ *and the parameters* (ρ, σ, τ) *are fixed.*

Proof. We shall now show that the semi-parametric model with RBD mean (6.30) can be obtained as a limiting case (as $\gamma \to \infty$) of model (6.39). To do so, let us assume that the vector parameter π in (6.30) is a zero mean Gaussian random vector with covariance $\gamma^2 I$, independent of α in (6.30). This implies that, conditionally on $x(s)$, $\psi^\top(x(s))\pi$ is a zero mean Gaussian with covariance matrix $\gamma^2\psi^\top(x(s))\psi(x(s))$ and it is independent of α.

Therefore $f(x(s)) := \psi^\top(x(s))\pi + (\phi^\top(x(s)) \otimes I_{n_{\text{dof}}})\alpha$ is zero mean Gaussian with covariance

$$\mathbb{E}[f(x(t))f(x(s))^\top] = \gamma^2\psi(x(t))^\top\psi(x(s)) + \rho^2(\phi(x(t))^\top\phi(x(s)) \otimes I_{n_{\text{dof}}}),$$

equal to (6.38).

A well known connection between Bayes and Fisher (i.e. assuming the parameter π is an unknown but fixed quantity) estimators, is that the latter can be obtained as a limiting case of the former when:

- the parameter π is modeled as a zero mean Gaussian vector with variance $\gamma^2 I$

- the variance of the prior distribution of π is let go to infinity by letting $\gamma^2 \to \infty$

To make this formal, let us stack the available data $y(s)$, $s = [1, N]$ in the vector Y and stack correspondingly the regressors $\psi^\top(x(s))$ and $\phi^\top(x(s)) \otimes I$ in the matrices Ψ and Φ respectively, so that models (6.30) and (6.39) can be written as

$$Y = \Psi\pi + \Phi\alpha + E \tag{6.43}$$

where E is defined with the same rule as Y. The minimum variance estimators of π and α under (6.38) are thus given by:

$$\begin{aligned} \hat{\pi} &= cov(\pi, Y) Var^{-1}\{Y\} Y \\ &= \gamma^2 \Psi^\top \left(\gamma^2 \Psi\Psi^\top + \rho^2 \Phi\Phi^2 + \sigma^2 I\right)^{-1} Y \\ \hat{\alpha} &= Cov(\alpha, Y) var^{-1}\{Y\} Y \\ &= \rho^2 \Phi^\top \left(\gamma^2 \Psi\Psi^\top + \rho^2 \Phi\Phi^2 + \sigma^2 I\right)^{-1} Y. \end{aligned} \tag{6.44}$$

Defining $R := \rho^2 \Phi\Phi^\top + \sigma^2 I$ and using the matrix inversion lemma we have:

$$\begin{aligned} &\left(\gamma^2 \Psi\Psi^\top + \rho^2 \Phi\Phi^\top + \sigma^2 I\right)^{-1} = \left(\gamma^2 \Psi\Psi^\top + R\right)^{-1} = \\ &= R^{-1} - R^{-1}\Psi \left(\Psi^\top R^{-1}\Psi + \gamma^{-2} I\right)^{-1} \Psi^\top R^{-1} \end{aligned}$$

so that, from (6.44)

$$\begin{aligned} \hat{\pi} &= \gamma^2 \left(I - \Psi^\top R^{-1}\Psi \left(\Psi^\top R^{-1}\Psi + \gamma^{-2} I\right)^{-1}\right) \Psi^\top R^{-1} Y \\ &= \left(\Psi^\top R^{-1}\Psi + \gamma^{-2} I\right)^{-1} \Psi^\top R^{-1} Y \end{aligned}$$

and, similarly

$$\begin{aligned} \hat{\alpha} &= \rho^2 \Phi^\top \left(I - R^{-1}\Psi \left(\Psi^\top R^{-1}\Psi + \gamma^{-2} I\right)^{-1} \Psi^\top\right) R^{-1} Y \\ &= \rho^2 \Phi^\top R^{-1} [Y - \Psi\hat{\pi}] \end{aligned}$$

Clearly, as $\gamma \to \infty$, we have that $\hat{\pi}$ converges to the *weighted* least squares estimate

$$\hat{\pi}_{WLS} = \left(\Psi^\top R^{-1}\Psi\right)^{-1} \Psi^\top R^{-1} Y \tag{6.45}$$

and $\hat{\alpha}$ converges to

$$\hat{\alpha} = \rho^2 \Phi^\top R^{-1} [Y - \Psi\hat{\pi}_{WLS}].$$

On the other hand the marginal likelihood function for model (6.43), under (6.12) i.e.

when π is considered as an unknown parameter, has the form:

$$\begin{aligned} L_m(Y) &= -2\log(p(Y)) \\ &= \log(\det(R)) + (Y - \Psi\pi)^\top R^{-1}(Y - \Psi\pi). \end{aligned}$$

When (ρ, σ^2, τ) are kept fixed, minimization w.r.t. π can be performed in closed form, and yields exactly the weighted least squares solution (6.45). Note however that, even for $\gamma \to \infty$, the marginal likelihoods of Y given the hyperparameters (ρ, σ^2, τ) computed using models (6.12) and (6.38) are different. In fact, if (6.12) is postulated, and π is solved for as above, one obtains the *profile* marginal log-likelihood $\hat{L}_m(Y) := L_m(Y)_{|\pi=\hat{\pi}_{WLS}}$

$$\hat{L}_m(Y) = \log(\det(R)) + (Y - \Psi\hat{\pi}_{WLS})^\top R^{-1}(Y - \Psi\hat{\pi}_{WLS}) \tag{6.46}$$

where the hyperparameters (ρ, σ^2, τ) which are hidden in the definition of $R = \rho^2\Phi\Phi^\top + \sigma^2 I$.

Instead, if (6.38) is postulated, the marginal log-likelihood takes the form

$$L_K(Y) = \log(\det(\gamma^2\Psi\Psi^\top + R)) + Y^\top(\gamma^2\Psi\Psi^\top + R)^{-1}Y.$$

Using, as above, the matrix inversion Lemma on $(\gamma^2\Psi\Psi^\top + R)$, Sylvester determinant identity and defining $\hat{\pi} := (\Psi^\top R^{-1}\Psi + \gamma^{-2}I)^{-1}\Psi R^{-1}Y$, we obtain

$$\begin{aligned} L_K(Y) &= \log(\det(R)) + \log(\det(I + \gamma^2\Psi^\top R^{-1}\Psi)) \\ &\quad +(Y - \Psi\hat{\pi})^\top R^{-1}(Y - \Psi\hat{\pi}) \\ &\quad +\hat{\pi}^\top\Psi^\top R^{-1}(Y - \Psi\hat{\pi}). \end{aligned}$$

As $\gamma \to \infty$ we have that $\hat{\pi} \to \hat{\pi}_{WLS}$ and $\hat{\pi}^\top\Psi^\top R^{-1}(Y - \Psi\hat{\pi}) \to 0$ so that

$$\begin{aligned} L_K(Y) &\simeq \log(\det(R)) + \log(\det(I + \gamma^2\Psi^\top R^{-1}\Psi)) \\ &\quad +(Y - \Psi\hat{\pi}_{WLS})^\top R^{-1}(Y - \Psi\hat{\pi}_{WLS}). \end{aligned} \tag{6.47}$$

The second term $\log(\det(I + \gamma^2\Psi^\top R^{-1}\Psi))$ can be manipulated as follows:

$$\begin{aligned} \log(\det(I + \gamma^2\Psi^\top R^{-1}\Psi)) &= \log(\det(\gamma^2 I)) + \log(\det(\gamma^{-2}I + \Psi^\top R^{-1}\Psi)) \\ &\simeq \log(\det(\gamma^2 I)) + \log(\det(\Psi^\top R^{-1}\Psi)) \end{aligned}$$

where the last approximation hold when $\gamma \to \infty$. Inserting the last expression in

$L_K(Y)$ we obtain that, up to the constant $\log(\det(\gamma^2 I))$ which is not a function of ρ, σ^2, τ,

$$\begin{aligned} L_K(Y) &\simeq \log(\det(R)) + \log(\det(\Psi^\top R^{-1}\Psi)) + (Y - \Psi\hat{\pi}_{WLS})^\top R^{-1}(Y - \Psi\hat{\pi}_{WLS}) \\ &\simeq L_m(Y) + \log(\det(\Psi^\top R^{-1}\Psi)) \end{aligned} \tag{6.48}$$

where the last equation shows that the two log likelihoods differ for a nontrivial term which have an influence on the location of their minima. ■

6.3.3 Online Learning

The problem of estimating model (6.3) from noise data $y(s)$ with $s = [1, N]$ is well known in the literature and has been detailed in Section 2.4.

The minimum variance linear estimator of f given N input-output data pair $\{y(s), x(s)\}$ is the solution of the Tikhonov regularization problem

$$\hat{f}_N = \underset{f\in\mathbb{R}^t}{\operatorname{argmin}} \frac{1}{\sigma^2}\sum_{s=1}^{N} \|y(s) - f(x(s))\|^2 + \frac{1}{\rho^2}\|f\|^2_{\widetilde{K}} \tag{6.49}$$

where $\|f\|^2_{\widetilde{K}} = f^\top \widetilde{K}^{-1} f$. By the Representer Theorem and considering w.l.o.g. the kernel $\widetilde{K} = \rho^2 K_G$ the solution of (6.49) is equivalent to

$$\hat{f}_N(x) = \rho^2 \sum_{s=1}^{N} a(s) K_G(x(s), x) \tag{6.50}$$

where $a(s) \in \mathbb{R}^{n_{\text{dof}}}$. Unfortunately, the number of parameters $a(s)$ is depending on the number of data N, making the estimator intractable for an online (recursive) solution. To overcome this limitation, it turns helpful for the approximation in random features (6.17) of K_G. Indeed, it has been shown that under this approximation model (6.3) can be approximated as the model (6.21) linear in the parameters θ. Recall that model (6.21) depends on the other parameters Σ_0, τ and σ^2. At the moment, we assume that these parameters are known. How to estimate them is a crucial point and will be explained at the end of this Section (see "Hyperparameters Tuning"). Thus, the vector $\theta \sim \mathcal{N}(0_c, \Sigma_0)$ completely specifies the inverse dynamics model and, as such, our learning problem has been reduced to estimating the vector θ in (6.21).

Therefore, the minimum variance linear estimator of θ given N input-output data pair $\{y(s), x(s)\}$ is the solution of the Tikhonov regularization problem:

$$\hat{\theta}_N = \underset{\theta\in\mathbb{R}^p}{\operatorname{argmin}} \frac{1}{\sigma^2}\sum_{s=1}^{N} \|y(s) - \varphi(x(s))^\top\theta\|^2 + \|\theta\|^2_{\Sigma_0}. \tag{6.51}$$

This coincides with the so called Regularized Least Squares problem and its optimal solution can be computed recursively through the well known Recursive Least Squares algorithm, see e.g (Ljung, 1999, Chapter 11). At a certain time instant t problem (6.51) admists the solution

$$\hat{\theta}_t = \hat{\theta}_{t-1} + L_t(y_t - \varphi(x_t)^\top \hat{\theta}_{t-1}) \tag{6.52}$$

$$L_t = \frac{P_{t-1}\varphi(x_t)}{1 + \varphi(x_t)^\top P_{t-1}\varphi(x_t)} \tag{6.53}$$

$$P_t = P_{t-1} - \frac{P_{t-1}\varphi(x_t)\varphi(x_t)^\top P_{t-1}}{1 + \varphi(x_t)^\top P_{t-1}\varphi(x_t)} \tag{6.54}$$

with initial conditions e.g.,

$$\hat{\theta}_0 = 0_c, \quad P_0 = I_c. \tag{6.55}$$

Recursion (6.52)-(6.55) can be used to update the inverse dynamics model (6.21) when new data become available.

In practice the Recursive Least Squares algorithm can be implemented taking advantage of the square-root algorithms, e.g. propagating Cholesky factors L_t of P_t rather than P_t as in (6.54). We have actually used the Cholesky-based update, see Björck (1996), which has better numerical properties. The computational complexity of each update is $O(c^2)$.

Hyperparameters Tuning

All the models presented in Section 6.2 depend on one or more parameters, called hyperparameters, which describe the prior model. For instance, the hyperparameters in model (6.30), used in the semiparametric learning with RBD mean, are $\eta := (\pi, \rho^2, \tau^2, \sigma^2)$ while those in model (6.39), used in the semiparametric learning with RBD kernel, are $\eta := (\gamma^2, \rho^2, \tau^2, \sigma^2)$. These hyperparameters are not known and need to be estimated from the data. In Section 2.4.3 two different approaches are outlined to address this problem, namely the marginal likelihood maximization and the validation set or cross validation. The *validation set* was described with several options while here the considered version is specified.

The batch of data used for the identification is split in two data sets: the training set and the validation set. A set of candidate hyperparameters is specified and denoted as

Ω. The inverse dynamics model $\mathcal{M}_\eta$ is estimated for each $\eta \in \Omega$ using the training set. Then, for any $\mathcal{M}_\eta$ the mean squared error $\mathrm{MSE}(\eta)$ is computed using the validation set. Hence, the $\mathrm{MSE}(\eta)$ provides an estimate of the error rate. According to the validation set approach, (James et al., 2013, Chapter 6), the optimal hyperparameters are given by solving

$$\hat{\eta} = \underset{\eta \in \Omega}{\operatorname{argmin}} \ \mathrm{MSE}(\eta). \tag{6.56}$$

6.4 Derivative-free Model

In this section the input locations $x(s)$ are a subject of study. Until now it has been considered that the input locations are composed by the joint positions, joint velocities and joint accelerations, i.e., $x(s) = [q(s)^\top \ \dot{q}(s)^\top \ \ddot{q}(s)^\top]^\top \in \mathbb{R}^m$, with $m = 3n_{\mathrm{dof}}$. As discussed in Section 6.1 the joint velocities and joint accelerations cannot be measured from the robot but are obtained through numerical differentiation of the joint positions leading to possible significant numerical errors. This is a very well known problem and highly discussed, see e.g., Siciliano et al. (2010); Hollerbach et al. (2008); Kozlowski (2012); Craig (2005); Nguyen-Tuong and Peters (2011b). Common solutions are to resort to ad-hoc filter design or to not take into account the noisy derivatives.

Some qualitative considerations on choosing these input locations are drawn.

- The use of joint positions, velocities and accelerations as input locations is justified by the physics because the RBD model is a second order dynamical system w.r.t. the joint positions. The concept of using these quantities is therefore linked to a parametric model and in nonparametric modeling, one can in principle rely on different input quantities.

- The problems related to numerical differentiation can be partially addressed by specific filter design. However, this requires the knowledge of the user how to do it, ad-hoc tuning of the parameters and moreover, there is not a direct way to evaluate how well the filtered signal results w.r.t. the unknown "true" one.

The issues listed above are only meant to be some discussion points that inspired the following methodology.

Given the measured data joint torques $y(s)$ and joint positions $q(s)$ with $s = [1, N]$, we want to learn the inverse dynamics as a function of a vector of input locations $\tilde{x}(s)$ which depends on the past joint positions $q(s^-) := [q(s)^\top, q(s-1)^\top, \ldots, q(s-M)^\top]^\top \in \mathbb{R}^{(M+1)n_{\mathrm{dof}}}$. This means that the inverse dynamics model becomes

$$y(s) = f(\tilde{x}(s)) + e(s), \quad s = [1, N]$$

where $f : \mathbb{R}^{\widetilde{m}} \to \mathbb{R}^{n_{\text{dof}}}$ is considered to be the nonparametric model[2] (6.8) and $\tilde{x} : \mathbb{R}^{(M+1)n_{\text{dof}}} \to \mathbb{R}^{\widetilde{m}}$ is the input location map that takes as input the vector $q(s^-)$. Specifically, $\tilde{x}(s)$ is linearly dependent on $q(s^-)$, i.e.,

$$\tilde{x}(s) = R\, q(s^-) \tag{6.57}$$

where $R \in \mathbb{R}^{\widetilde{m} \times M n_{\text{dof}}}$ is a matrix of parameters that will be estimated as hyperparameters. Notice that $\widetilde{m} = k n_{\text{dof}}$ and k is the number of quantities that compose the input vector, here called *features*. For example, in the standard case where the input locations are joint positions, velocities and accelerations the input vector is composed by $k = 3$ features, indeed, $\widetilde{m} = m = 3n_{\text{dof}}$. In the following, all the degrees of freedom (joints) will have the same features.

The number of features k, the number of past temporal lags M as well as the the structure of R have to be defined. In the following two possibilities are proposed.

Features resembling joint velocities and accelerations

The first idea is based on the belief that joint positions, velocities and accelerations are the correct features and the problem relies on the numerical differentiation operations. Hence, in this case $k = 3$ features are considered.

We assume that the joint velocities and accelerations are computed by a 1st order backward difference, $B_1(z)$, and by a 2nd order backward difference, $B_2(z)$, respectively. In addition both are filtered by a first order low pass filter, that is

$$\begin{aligned}
\dot{q}(s) &= B_1(z)F_1(z)q(s) = \frac{1 - z^{-1}}{T_s}\frac{z}{z - \beta_1}q(s) \\
\ddot{q}(s) &= B_2(z)F_2(z)q(s) = \frac{1 - 2z^{-1} + z^{-2}}{T_s^2}\frac{z}{z - \beta_2}q(s)
\end{aligned}$$

where $T_s > 0$ is the sampling time and $0 < \beta_1,\ \beta_2 < 1$ represent the poles of the filters.

We resort to a partial fraction decomposition to rewrite the above expressions as a function of $q(s^-)$, that is:

[2] The extension to the semiparametric models is straightforward from a mathematical point of view. However, the RBD component in semiparametric models requires that the input locations should be consistent with the physical laws. More studies on that are needed and have not been done yet.

$$
\begin{aligned}
\dot{q}(s) &= \alpha_1(1-z^{-1})\frac{z}{z-\beta_1}q(s) = \alpha_1\frac{z-1}{z-\beta_1}q(s) \\
&= \alpha_1 q(s) + \sum_{t=1}^{M}\alpha_1\beta_1^{t-1}(\beta_1-1)q(s-t) \\
\ddot{q}(s) &= \alpha_2(1-2z^{-1}+z^{-2})\frac{z}{z-\beta_2}q(s) = \alpha_2\frac{z-2+z^{-1}}{z-\beta_2}q(s) \\
&= \alpha_2 q(s) + \alpha_2(\beta_2-2)q(s-1) + \sum_{t=2}^{M}\alpha_2\beta_2^{t-2}(\beta_2^2-2\beta_2+1)q(s-t)
\end{aligned}
\tag{6.58}
$$

where $\alpha_1 = 1/T_c$ and $\alpha_2 = 1/T_c^2$. Accordingly, the vector of the input locations becomes

$$
\begin{aligned}
&\tilde{x}(s) = R\; q(s^-) \\
&:= \begin{bmatrix}
I_{n_{\text{dof}}} & 0 & \dots & \dots & 0 \\
\alpha_1 I_{n_{\text{dof}}} & \alpha_1(\beta_1-1)I_{n_{\text{dof}}} & \dots & \alpha_1\beta_1^{t-1}(\beta_1-1)I_{n_{\text{dof}}} & \dots \\
\alpha_2 & \alpha_2(\beta_2-2)I_{n_{\text{dof}}} & \dots & \alpha_2\beta_2^{t-2}(\beta_2^2-2\beta_2+1)I_{n_{\text{dof}}} & \dots
\end{bmatrix}
\begin{bmatrix} q(s) \\ q(s-1) \\ \vdots \\ q(s-M) \end{bmatrix}
\end{aligned}
\tag{6.59}
$$

The hyperparameters that have to be estimated in order to characterise R are $\eta = [\alpha_1, \beta_1, \alpha_2, \beta_2]$.

This vector of input locations is denoted with the abbreviation "FVA" since its Features resemble the structure of the joint Velocities and Accelerations.

A nice property of this characterization is that the number of hyperparameters is low and it is independent on the length of the past temporal lags M, which can be arbitrarily chosen.

Structure-free features

An alternative is to consider the features "free" of any structure and let them be estimated from the data. This means

$$\tilde{x}(s) = R\, q(s^-) := \begin{bmatrix} r_1^\top \otimes I_{n_{\text{dof}}} \\ \vdots \\ r_k^\top I_{n_{\text{dof}}} \end{bmatrix} \begin{bmatrix} q(s) \\ q(s-1) \\ \vdots \\ q(s-M) \end{bmatrix} \tag{6.60}$$

where $r_i^\top \in \mathbb{R}^{1\times M+1}$ represent the i-th vector of hyperparameters. This input locations vector will be denoted with the abbreviation "FSF" since its Features are Structure-Free. However, the joint positions are usually measured with accurate sensors and it makes sense to consider them as a feature; accordingly, the first row of hyperparameters is considered known and set to $r_1^\top := [1, 0, \ldots, 0]$.

Notice, that the input locations vector "FSF" includes "FVA" in (6.59) and all the possible linear and causal numerical differentiation operations. The price of this generality is that a large number of hyperparameters has to be estimated, i.e., $(k-1)(M+1)$. As it is well known in optimization this might lead to local minima problems. In order to overcome this issue one might resort to regularization techniques on the hyperparameters or to set appropriate initial conditions.

6.5 Simulations Results

An interesting aspect of this research has been the possibility to work with real data collected from the humanoid robot iCub, Metta, Natale, Nori, Sandini, Vernon, Fadiga, Von Hofsten, Rosander, Lopes, Santos-Victor, et al. (2010), shown in Figure 6.2.

The data have been kindly provided by the authors of Camoriano et al. (2016).

iCub is a full-body humanoid robot with 53 degrees of freedom. The aim of the experiments is to test the models of Section 6.3.2 for learning online the inverse dynamics of one iCub's arm, while the others degree of freedom are fixed.

The inputs $q(s)$ are the angular positions of the 3 degrees of freedom (dof) shoulder joints and of the 1-dof elbow joint. Joint positions have been differentiated to obtain joint velocities and accelerations by the authors of Camoriano et al. (2016) using a standard module of the open source iCub project (the adaptWinPolyEstimator[3] module) which is based on the work Janabi-Sharifi, Hayward, and Chen (2000). The input locations

[3] Available at *http : //wiki.icub.org/brain/adaptWinPolyEstimator_8cpp_source.html*

Figure 6.2: Humanoid Robot iCub.

vector $x(s)$ is the stack of the joint positions, velocities and accelerations obtained in this way, while the input locations vectors $\tilde{x}(s)$ considered in Section 6.4 are obtained starting from the $q(s)$.

The outputs $y(s)$ are the 3 force and 3 torque components measured by the six-axes force/torque (F/T) sensor embedded in the shoulder of the iCub arm, see Figure 6.3.

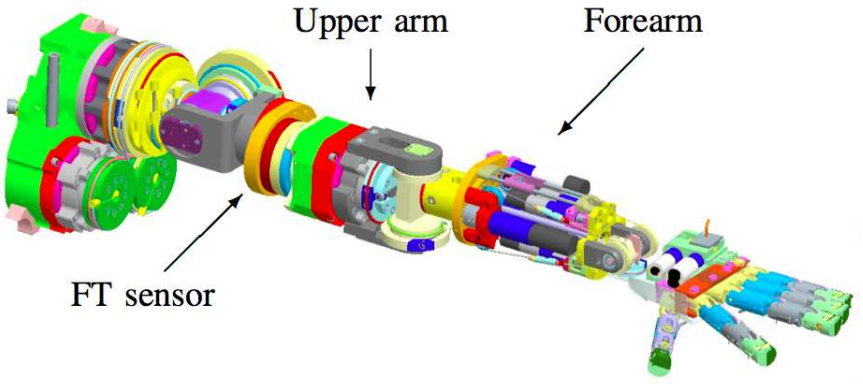

Figure 6.3: iCub's arm.

Notice that the measured forces/torques are not the applied joint forces and torques and, as such, the model we learn is not, strictly speaking, the inverse dynamics model. Yet, as explained in Ivaldi, Fumagalli, Randazzo, Nori, Metta, and Sandini (2011), the

feedforward joint torques can be determined from components (forces and torques) of $y(s)$. Indeed, such a model has been used in the literature as a benchmark for the inverse dynamics learning, Gijsberts and Metta (2011), Camoriano et al. (2016) .

Datasets

We consider the 2 datasets used in Camoriano et al. (2016), corresponding to different trajectories of the end-effector. In the first one the end-effector tracks circles in the XY plane of radius $10cm$ at an approximative speed of $6m/s$; in the second one, the end-effector tracks similar circles but in the XZ plane (the Z axis corresponds to the vertical direction, parallel to the gravity force). The two circles are tracked using the Cartesian controller proposed in Pattacini, Nori, Natale, Metta, and Sandini (2010). Each dataset contains approximately 8 minutes of data collected at a sampling rate of $20Hz$, for a total of 10000 points per dataset. One single circle is completed by the robot in about 1.25 seconds which corresponds to 25 points.

Models

The models described in Sections 6.3.2 and 6.4 are considered, endowed with the marginal likelihood approach (ML) for the estimation of the hyperparameters, as well as the validation based methods[4] (VS) discussed in Camoriano et al. (2016). For ease of exposition the following shorthands are used:

- P: the parametric model (6.6).
- *NP-ML*: the approximated nonparametric model (6.26); hyperparameters estimated with ML.
- *SP-ML*: the semi-parametric model with RBD mean (6.30); hyperparameters estimated with ML.
- *SP2-ML*: the semi-parametric model with RBD mean (6.34), in which the mean is computed via least squares as in Camoriano et al. (2016) and then the nonparametric model is applied to the residuals; hyperparameters estimated with ML.
- *SPK-ML*: the semi-parametric model with RBD kernel (6.39); hyperparameters estimated with ML.

[4] Using validation based methods can become infeasible when the number of hyperparameters is large; therefore we have not applied the validation set approach to four models: first, the semiparametric model with RBD mean when the mean is to be considered as an hyperparameter, second, the semiparametric model with RBD kernel which has the extra parameter γ and finally the two derivative-free models.

- *NP-VS*: the nonparametric model (6.26); hyperparameters estimated with VS.
- *SP2-VS*: the semi-parametric model with RBD mean (6.34), in which the mean is computed via least squares; hyperparameters estimated with VS.
- *NP-ML-FVA*: the nonparametric model (6.26); input locations "FVA"; hyperparameters estimated with ML.
- *NP-ML-FSF*: the nonparametric model (6.26); input locations "FSF'; hyperparameters estimated with ML.

Experiment

The proposed algorithms have been implemented using Matlab. The RBD regressor ψ for the right arm of iCub has been computed using the library iDynTree, Nori, Traversaro, Eljaik, Romano, Del Prete, and Pucci (2015). The Marginal Likelihood has been optimized using the Matlab `fminsearch.m` function. The recursive least square algorithms have been implemented using GURLS library, Tacchetti, Mallapragada, Santoro, and Rosasco (2013). The results of all validation based methods are obtained using code which has been kindly provided by the authors of Camoriano et al. (2016).

For each model as above, the following online learning scenario is considered (with reference to the general model structure (6.21)):

- Initialization: The first 1000 points in XY-dataset are used to estimate the hyperparameters with the two techniques considered, say $\hat{\eta}_{ML}$ and $\hat{\eta}_{VS}$, as well as to compute an initial estimate of parameter θ, say $\hat{\theta}_0$.
- Stage 1: The remaining 9000 points of the XY-dataset are used to update online parameter θ using the recursive least-squares algorithm, thus obtaining $\hat{\theta}_t$, $t = 1, \ldots, 9000$. Let us call $\hat{\theta}^1 = \hat{\theta}_{9000}$ the estimator that has seen all the data of the XY-dataset.
- Stage 2: The XZ-dataset is split in 5 sequential subsets of 2000 points each (approximately 100 seconds). In these subsets the performance of the online estimators are evaluated. For each subset, the parameter θ is independently always initialized with $\hat{\theta}^1$. Then, $\hat{\theta}^1$ is updated with the recursive least-squares algorithm for all the data in the subset.

It is underlined that in Stage 2 the initial model has been computed from the XY-dataset, which corresponds to a different motion with respect to XZ-dataset. There-

fore, the evaluation of the performance in Stage 2 allows us to verify the property of generalization to unseen data of the considered models.

Performance

This section is divided into two: in the first part the models described in Section 6.3.2 are compared while in the second one the nonparametric models of Section 6.4 are compared.

The goal of the online algorithm is that the model estimated in the second dataset quickly captures the new information gathered from the XZ-dataset, adapting to the new task. For instance, in model predictive control the quality of the control depends on the prediction capability of the model over a prescribed horizon, Maciejowski (2002). In order to measure this ability we consider the following index:

$$\varepsilon_t^{(i)} = \frac{\sum_{s=1}^{T}(y_{t+s}^{(i)} - \hat{y}_{t+s|t}^{(i)})^2}{\sum_{s=1}^{T}(y_{t+s}^{(i)})^2} \tag{6.61}$$

where $\hat{y}_{t+s|t}^{(i)}$ is the estimate of the output $y_{t+s}^{(i)}$ at time $t+s$ using the model estimated with data up to time t. Therefore, $\varepsilon_t^{(i)}$ represents the relative squared prediction error over the horizon $[t+1,\dots,t+T]$ at time t. Let ε_t^F and ε_t^T be the average value of $\varepsilon_t^{(i)}$ for the 3 forces and the 3 torques, respectively.

In Figure 6.4 we show ε_t^F and ε_t^T, averaged over the 5 subsets, with $T = 25$ (1.25 seconds), i.e. with the end-effector completing one circle during the prediction horizon.

Clearly, the parametric algorithm P exhibits a poor performance because it describes only crude idealizations of the actual dynamics. The algorithms based on the VS approach perform significantly worse in the first 60 seconds than those based on the ML approach. This result is not unexpected because the ML approach represents a robust way to estimate hyperparamters, Pillonetto and Chiuso (2015). The models with the best performance are SP-ML and SPK-ML because they combine the benefit of the parametric and the nonparametric approach. Although also SP2-ML exploits this benefit, it provides a slightly worse performance especially in the estimation of the torques. This is probably due to the fact that the first (least squares) step, i.e. estimation of the linear model, is a subject to strong bias deriving from the unmodeled dynamics. Instead, a sound approach is followed by SP-ML and SPK-ML in which the estimation of the hyperparameters is performed jointly, avoiding such bias. In the steady state all these methods, with the exception of P, provide similar performance; yet the two semi-parametric models (SP-ML and SPK-ML) perform better in terms of both average and standard deviation, as clearly shown in Figure 6.5 which reports the boxplots of ε_t^F and ε_t^T in "steady state", i.e. after

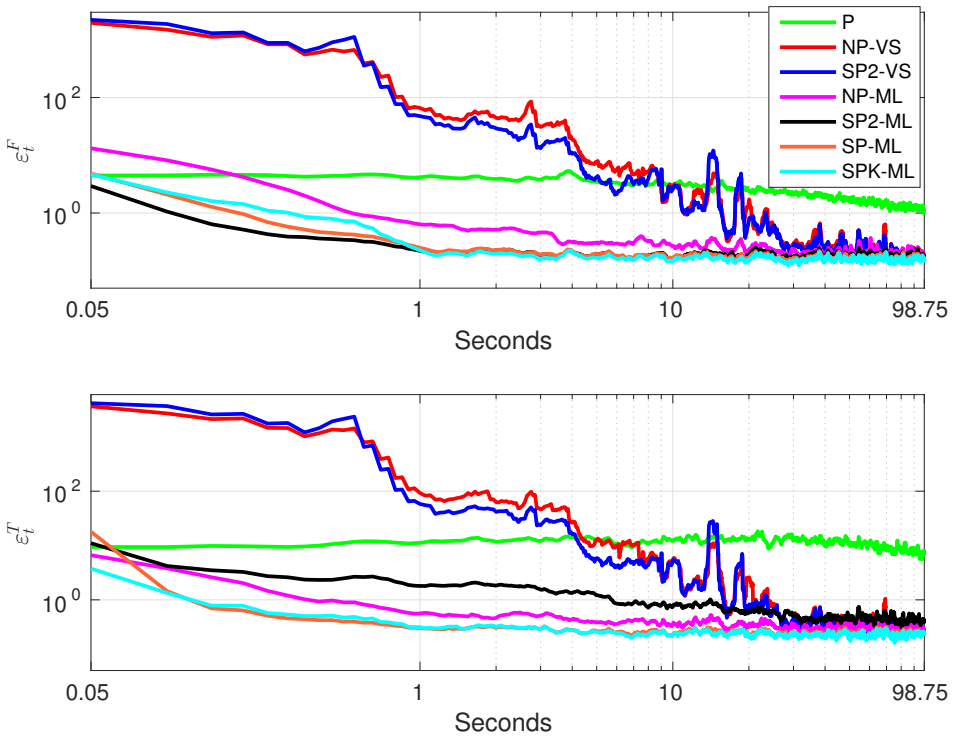

Figure 6.4: Average (over the 5 subsets of c.a 100 seconds each) of the relative squared prediction errors ε_t^F and ε_t^T, computed with $T = 25$ corresponding to a horizon of 1.25 seconds.

the first 30 seconds which is considered to be transient (see Figure 6.4).

The comparison of the models in Section 6.3.2 is concluded. The focus now is in the evaluation of the nonparametric derivative-free models described in Section 6.4, namely NP-ML-FVA and NP-ML-FSF. Their hyperparameters are estimated through maximization of the marginal likelihood, therefore the comparison is done w.r.t. the analogous model with "standard" input locations NP-ML, which also outperforms NP-VS.

The derivative-free models require to set two parameters: first, the number of past temporal lags M, considered by the features, see (6.59) and (6.60) and second, the number of features k. The former, after some empirical experiments, has been chosen $M = 10$ and the latter, is fixed to $k = 3$ in the input locations FVA by definition and for a fair comparison, the same value has been chosen for the input locations FSF.

The prediction capabilities are again measured by the average values for the 3 forces, ε_t^F, and for the 3 torques, ε_t^T, of the relative squared prediction error in (6.61). Therefore, the following figures are analogous to the Figures 6.4 and 6.5 for the three nonparametric models.

The average values over the 5 subsets of ε_t^F and ε_t^T are illustrated in Figure 6.6, with $T = 25$ (1.25 seconds), i.e. with the end-effector completing one circle during the

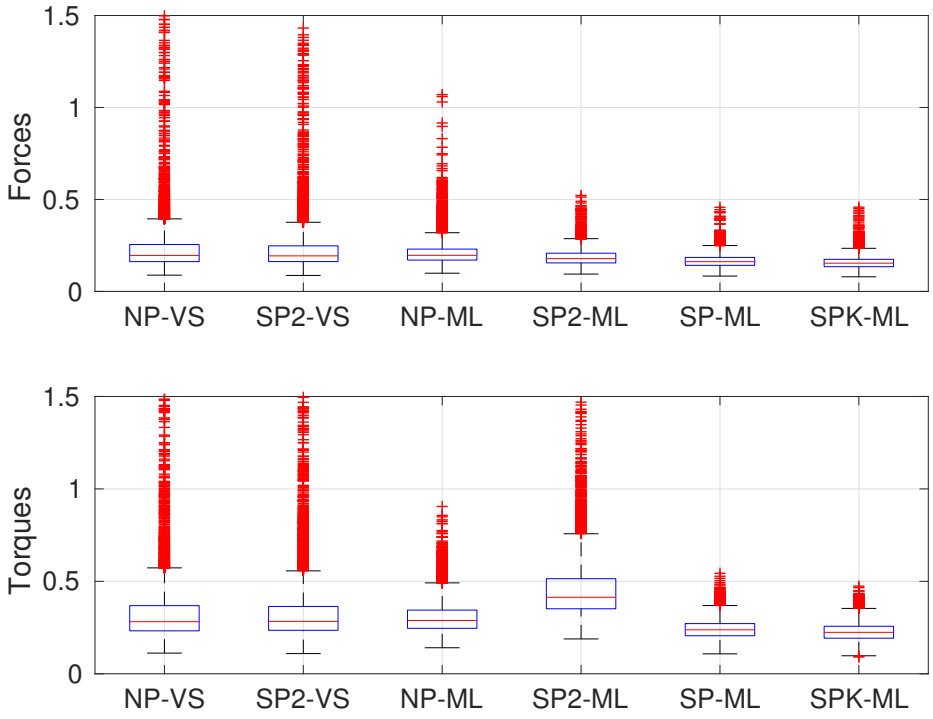

Figure 6.5: Boxplots of the steady state (i.e. after 30 seconds, see Figure 6.4) relative squared prediction errors ε_t^F and ε_t^T, computed with $T = 25$ corresponding to a horizon of 1.25 seconds.

prediction horizon.

Both nonparametric derivative-free models are high-performing and a significant improvement in terms of transitory as well as steady-state can be seen w.r.t. the NP-ML model.

At time instant $t = 0.05$, which means when the estimator obtained from the XY-dataset, $\hat{\theta}^1$, makes predictions in a different trajectory never seen before, the NP-ML estimator performs 10 times worse in the forces and 6 times worse in the torques.

As before, the behaviour in steady state is analysed by boxplots of ε_t^F and of ε_t^T after the first 30 seconds of simulations. The results are shown in Figure 6.7.

The derivative-free models significantly outperform NP-ML in terms of both median and distribution.

The difference between NP-ML-FSF and NP-ML-FVA is less notable, however the former slightly outperforms the latter in terms of transitory as well as steady state performance, resulting to be the preferable method. However, it is remarkable that NP-ML-FVA outperforms NP-ML; indeed, FVA resembles the most basic backwards differentiation with a simple 1st order low pass (see (6.58)) and is characterised by a few hyperparameters.

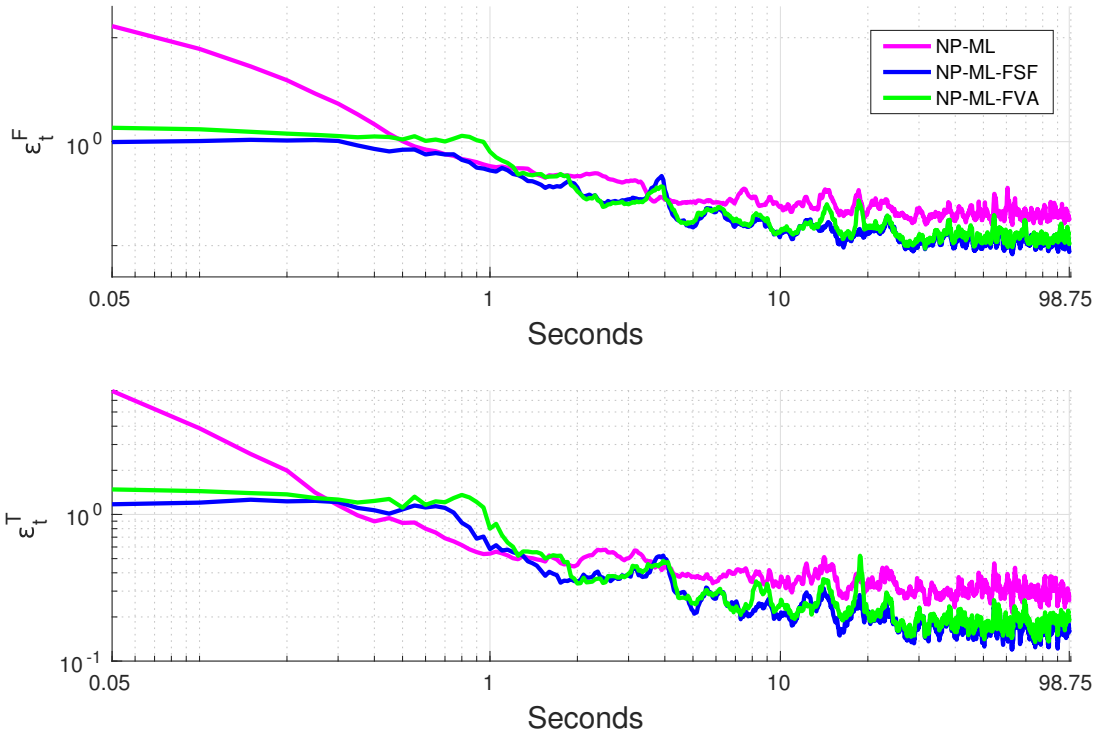

Figure 6.6: Average (over the 5 subsets of c.a 100 seconds each) of the relative squared prediction errors ε_t^F and ε_t^T, computed with $T = 25$ corresponding to a horizon of 1.25 seconds.

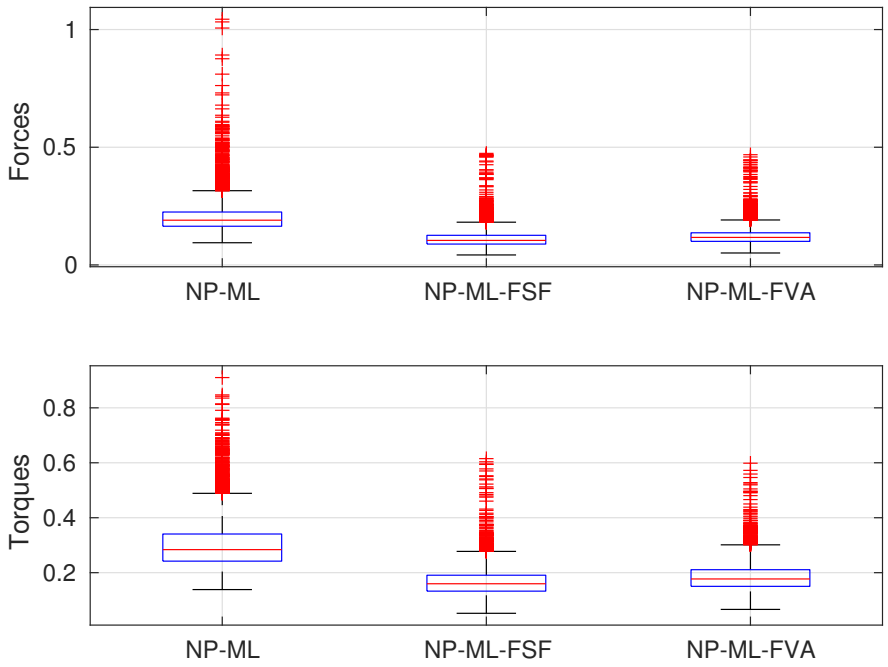

Figure 6.7: Boxplots of the steady state (i.e. after 30 seconds, see Figure 6.6) relative squared prediction errors ε_t^F and ε_t^T, computed with $T = 25$ corresponding to a horizon of 1.25 seconds.

As a final remark, the nonparametric derivative-free models achieved prediction performances that neither the semiparametric models SPK-ML nor SP-ML could, as it can be seen when comparing Figure 6.4 with Figure 6.6 and Figure 6.5 with Figure 6.7. The prediction error is lower in the derivative-free models in terms of both transitory and steady-state.

These results give an empirical evidence that learning the features from the input location is a rather promising direction.

NP-ML-FSF resulted to be the most high-performing technique and, as a last comparison, it is of interest to examine its performance as the number of features changes. The choice of $k = 3$ results from the physics[5] that suggests three specific features: the joint positions, velocities and accelerations. In terms of differential equations this means that the RBD is a second order model of the joint positions. As already discussed the choice of the number of features is arbitrary in the nonparametric model; consequently one possible question that could arise is if the physical laws are right in yielding a second order differential equation for the Rigid Body Dynamical model. The question is clearly too ambitious and we do not claim to have an answer to it. However, the following results give some empirical evidence on the question.

The model NP-ML-FSF is compared for different number of features, i.e., $k = 2, 3, 4$. The way the comparison is carried out is the usual one: Figure 6.8 represents the average over the 5 subsets of the prediction error (6.61) for the forces and the torques (analogously to Figures 6.4, 6.6) and the steady state of ε_t^F and ε_t^T are analyzed in Figure 6.9 (analogously to Figures 6.5 and 6.7).

From the Figures 6.8 and 6.9 it can be observed that when only 2 features in the input locations are considered the performance deteriorates, which means that input space is not rich enough to describe the dynamics of the robot. Instead, in the cases of $k = 3$ or $k = 4$ the performance are analogous, which means that adding a fourth feature does not provide any helpful information in the input space to describe the dynamics. Notice that also the case of $k = 5$ leads to similar results and therefore is omitted here.

Concluding, this experiment shows that the appropriate number of features required to describe the dynamics of the system is 3, which confirms that the RBD model should be a set of second order differential equations w.r.t. the joint positions.

[5]The choice $k = 3$ is set for a "fair comparison" with the method FVA, which indeed tries to resemble the physics laws

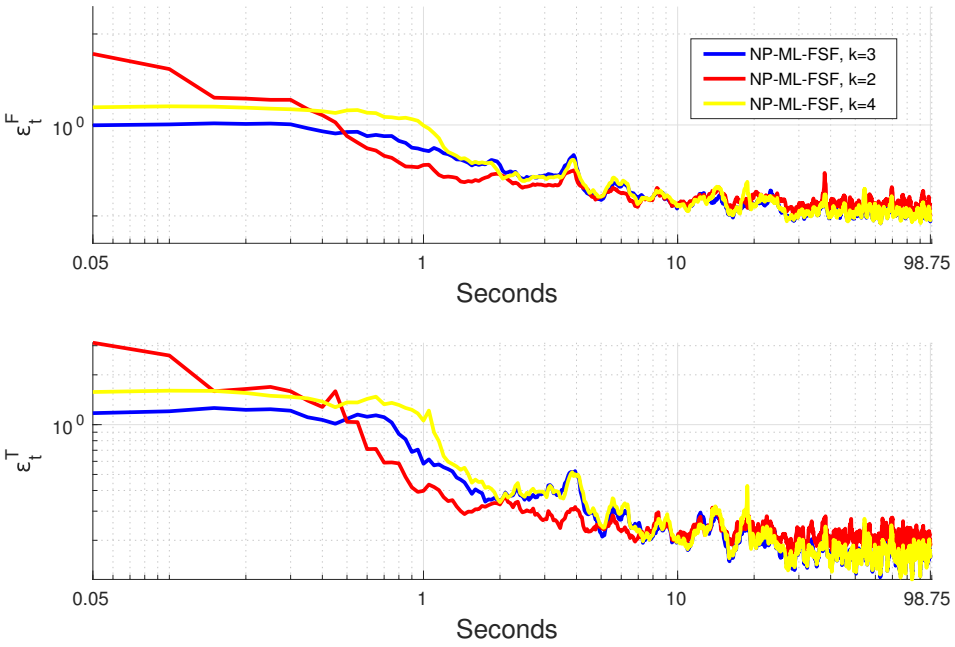

Figure 6.8: Average (over the 5 subsets of c.a 100 seconds each) of the relative squared prediction errors ε_t^F and ε_t^T, computed with $T = 25$ corresponding to a horizon of 1.25 seconds.

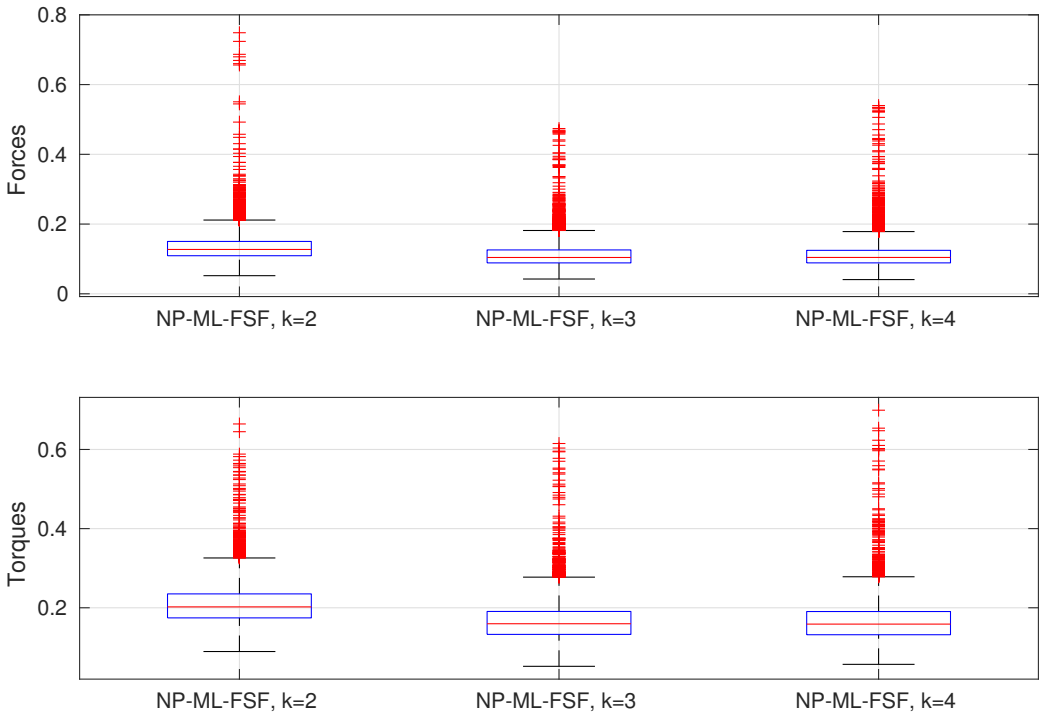

Figure 6.9: Boxplots of the steady state (i.e. after 30 seconds, see Figure 6.8) relative squared prediction errors ε_t^F and ε_t^T, computed with $T = 25$ corresponding to a horizon of 1.25 seconds.

6.6 Conclusions

In this Chapter, several algorithms used for online learning of the robot inverse dynamics are placed in a common framework. Such algorithms are classified according to the following: the considered model (parametric, nonparametric, semiparametric with RBD mean and semi-parametric with RBD kernel), the way the hyperparameters are estimated (VS approach and ML approach) and the choice of the input locations (derivative-free features and joint velocities and accelerations).

Those algorithms are applied to the online leaning of the inverse dynamics of one iCub's arm. The results showed the superiority of the ML approach to estimate the hyperparameters and also that semiparametric models outperform the others, providing the same input locations. The last result confirms the advantage of combining parametric and nonparametric approaches together.

The introduction of the derivative-free models yields outstanding results: the nonparametric models equipped with the procedure to learn the features of the input locations outperform the companion nonparametric model and the semiparametric models with input locations given by joint positions velocities and acceleration.

Future works

Several extensions are possible:

- Semiparametric models appear to be an effective resource to estimate the dynamics of physical systems since they combine the qualities of both parametric and nonparametric models. Nevertheless, it is not clear yet, what the best way to combine the two is. Furthermore, in several applications one might be interested in the real physical value of the parameters. This means that it should be granted to the parametric component of the semiparametric models to describe as much information as possible from the data. Indeed, in the current methods, it plausible that the nonparametric component explains informations that could be in principle described by the parametric model. This result is not addressed in the current research but it would be an important turning point in further researches.

- in the proposed online algorithms tuning the hyperparameters is a fundamental step which lead to very different performance. However, in this work the hyperparameters are retained fixed to the initial estimate. This is done in order to achieve high computational performance in the update of the estimator, $O(c^2)$ if c is the dimension of the estimator. However, in Chapter 4 it is shown that the online update of the hyperparameters yield great advantages in the estimation

performance (in particular when dealing with time-varying systems). The extension of the online techniques proposed in Chapter 4 to the problem of estimating robotic inverse dynamics could bring important advantages for the real-world applications.

- On the one hand, the introduction of the nonparametric derivative-free models seems to be a promising direction in the estimation of the inverse dynamic models. On the other hand, semiparametric models outperform the nonparametric ones. Consequently, semiparametric derivative-free models are an appealing extension.

References

Andrieu C., Doucet A., and Holenstein R. Particle markov chain monte carlo methods. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 72(3):269–342, 2010.

Aravkin A., Burke J., Chiuso A., and Pillonetto G. On the estimation of hyperparameters for empirical bayes estimators: Maximum marginal likelihood vs minimum mse. *Proc. of SYSID 2012*, 2012.

Aronszajn N. Theory of reproducing kernels. *Trans. of the American Mathematical Society*, 68:337–404, 1950.

Åström K.-J. and Bohlin T. *Numerical Identification of Linear Dynamic Systems from Normal Operating Records*, pages 96–111. Springer US, Boston, MA, 1966. ISBN 978-1-4899-6289-8. URL http://dx.doi.org/10.1007/978-1-4899-6289-8_12.

Bachnas A., Tóth R., Ludlage J., and Mesbah A. A review on data-driven linear parameter-varying modeling approaches: A high-purity distillation column case study. *Journal of Process Control*, 24(4):272–285, 2014.

Bai E.-W., Tempo R., Liu Y., and others . Identification of iir nonlinear systems without prior structural information. *IEEE transactions on automatic control*, 52(3): 442–453, 2007.

Banbura M., Giannone D., and Reichlin L. Large Bayesian VARs. *Journal of Applied Econometrics*, 25(1):71–92, 2010.

Barzilai J. and Borwein J. M. Two-point step size gradient methods. *IMA Journal of Numerical Analysis*, 8(1):141–148, 1988.

Bertsekas D. *Nonlinear Programming*. Athena Scientific, 1995. ISBN 9781886529144. URL https://books.google.it/books?id=QeweAQAAIAAJ.

Billings S. Identification of nonlinear systems¿ a survey. In *IEE Proceedings D-Control Theory and Applications*, volume 127, pages 272–285. IET, 1980.

Billings S. A. *Nonlinear system identification: NARMAX methods in the time, frequency, and spatio-temporal domains*. John Wiley & Sons, 2013.

Bishop C. M. *Pattern Recognition and Machine Learning*. Springer, 2006.

Bittanti S., Bolzern P., and Campi M. Convergence and exponential convergence of identification algorithms with directional forgetting factor. *Automatica*, 26(5):929–932, 1990.

Björck A. *Numerical Methods for Least Squares Problems*. Society for Industrial and Applied Mathematics, 1996. URL http://epubs.siam.org/doi/abs/10.1137/1.9781611971484.

Bocsi B., Csató L., and Peters J. Alignment-based transfer learning for robot models. In *The 2013 International Joint Conference on Neural Networks (IJCNN)*, pages 1–7, 2013.

Bonettini S., Chiuso A., and Prato M. A scaled gradient projection methods for Bayesian learning in dynamical systems. *SIAM Journal on Scientific Computing*, page in press, 2015.

Bonettini S. and Prato M. Nonnegative image reconstruction from sparse fourier data: a new deconvolution algorithm. *Inverse Problems*, 26(9):095001, 2010. URL http://stacks.iop.org/0266-5611/26/i=9/a=095001.

Bonettini S., Zanella R., and Zanni L. A scaled gradient projection method for constrained image deblurring. *Inverse Problems*, 25(1):015002, 2009. URL http://stacks.iop.org/0266-5611/25/i=1/a=015002.

Bonettini S. Inexact block coordinate descent methods with application to non-negative matrix factorization. *IMA Journal of Numerical Analysis*, 2011. URL http://imajna.oxfordjournals.org/content/early/2011/01/12/imanum.drq024.abstract.

Bottegal G., Aravkin A. Y., Hjalmarsson H., and Pillonetto G. Robust EM kernel-based methods for linear system identification. *CoRR*, abs/1411.5915, 2014. URL http://arxiv.org/abs/1411.5915.

Box G. E., Jenkins G. M., Reinsel G. C., and Ljung G. M. *Time series analysis: forecasting and control*. John Wiley & Sons, 2015.

Brockwell P. J. and Davis R. A. *Time series: theory and methods*. Springer Science & Business Media, 2013.

Calafiore G. Approximation of n-dimensional data using spherical and ellipsoidal primitives. *IEEE Transaction on System, Mand, And Cybernetics*, 32, March 2002.

Camoriano R., Traversaro S., Rosasco L., Metta G., and Nori F. Incremental semiparametric inverse dynamics learning. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 544–550, May 2016.

Campi M. C. and Weyer E. Guaranteed non-asymptotic confidence regions in system identification. *Automatica*, 41(10):1751–1764, 2005.

Campi M. C. and Weyer E. Finite sample properties of system identification methods. *Automatic Control, IEEE Transactions on*, 47(8):1329–1334, 2002.

Candes E. J., Wakin M. B., and Boyd S. P. Enhancing sparsity by reweighted l1 minimization. *Journal of Fourier analysis and applications*, 14(5-6):877–905, 2008.

Carli F., Chen T., Chiuso A., Ljung L., and Pillonetto G. On the estimation of hyperparameters for bayesian system identification with exponentially decaying kernels. In *2012 IEEE 51st IEEE Conference on Decision and Control (CDC)*, pages 5260–5265. IEEE, 2012.

Carlin B. P. and Louis T. A. Bayes and empirical bayes methods for data analysis. *Statistics and Computing*, 7(2):153–154, 1997.

Chartrand R. and Yin W. Iteratively reweighted algorithms for compressive sensing. In *Acoustics, speech and signal processing, 2008. ICASSP 2008. IEEE international conference on*, pages 3869–3872. IEEE, 2008.

Chen G. Stability of nonlinear systems. *Encyclopedia of RF and Microwave Engineering*, 2004.

Chen T., Andersen M., Ljung L., Chiuso A., and Pillonetto G. System identification via sparse multiple kernel-based regularization using sequential convex optimization techniques. *IEEE Transactions on Automatic Control*, 59(11):2933–2945, 2014.

Chen T., Ohlsson H., and Ljung L. On the estimation of transfer functions, regularizations and Gaussian processes - revisited. *Automatica*, 48(8):1525–1535, 2012.

Chen T. and Ljung L. Implementation of algorithms for tuning parameters in regularized least squares problems in system identification. *Automatica*, 49(7):2213–2220, 2013.

Chilali M. and Gahinet P. H-infinity design with pole placement constraints:an lmi approach. *IEEE Transactions on Automatic Control*, 41:358–367, 1996.

Chiuso A. The role of vector autoregressive modeling in predictor based subspace identification. *Automatica*, 43:1034–1048, 2007. ISSN 0005-1098.

Chiuso A. Regularization and bayesian learning in dynamical systems: Past, present and future. *Annual Reviews in Control*, 41:24–38, 2016.

Craig J. J. *Introduction to robotics: mechanics and control*, volume 3. Pearson Prentice Hall Upper Saddle River, 2005.

Cressie N. *Statistics for spatial data*. John Wiley & Sons, 2015.

Csáji B. C., Campi M. C., and Weyer E. Sign-perturbed sums: A new system identification approach for constructing exact non-asymptotic confidence regions in linear regression models. *Signal Processing, IEEE Transactions on*, 63(1):169–181, 2015.

Csató L. and Opper M. Sparse on-line gaussian processes. *Neural Comput.*, 14 (3):641–668, March 2002. ISSN 0899-7667. URL http://dx.doi.org/10.1162/089976602317250933.

Daley R. *Atmospheric data analysis*. Cambridge university press, 1993.

Dasgupta S. and Huang Y. F. Asymptotically convergent modified recursive least-squares with data-dependent updating and forgetting factor for systems with bounded noise. *Information Theory, IEEE Transactions on*, 33(3):383–392, 1987.

Davidon W. C. Variable metric method for minimization. *SIAM Journal on Optimization*, 1(1):1–17, 1991. URL http://dx.doi.org/10.1137/0801001.

De la Cruz J. S., Kulic D., Owen W. S., Calisgan E., and Croft E. A. On-line dynamic model learning for manipulator control. In *SyRoCo*, pages 869–874, 2012.

De Nicolao G., Ferrari-Trecate G., and Lecchini A. MAXENT priors for stochastic filtering problems. In *Mathematical Theory of Networks and Systems*, Padova, Italy, July 1998. URL http://control.ee.ethz.ch/index.cgi?page=publications;action=details;id=1779.

Efron B. The estimation of prediction error. *Journal of the American Statistical Association*, 99(467):619–632, 2004. URL http://dx.doi.org/10.1198/016214504000000692.

Efron B. Bayesians, frequentists, and scientists. *Journal of the American Statistical Association*, 100(469):1–5, 2005.

Efron B. and Tibshirani R. J. *An introduction to the bootstrap*. CRC press, 1994.

Garatti S., Campi M. C., and Bittanti S. Assessing the quality of identified models through the asymptotic theory-when is the result reliable? *Automatica*, 40(8): 1319–1332, August 2004. ISSN 0005-1098.

Gelman A., Roberts G., and Gilks W. *Efficient Metropolis jumping rules.*, volume Bayesian Statistics, chapter V. Oxford University Press, 1996.

Gijsberts A. and Metta G. Incremental learning of robot dynamics using random features. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 951–956, 2011.

Gilks W., Richardson S., and Spiegelhalter D. *Markov Chain Monte Carlo in Practice*. Chapman & Hall/CRC Interdisciplinary Statistics. Taylor & Francis, 1995. ISBN 9780412055515.

Goodwin G., Gevers M., and Ninness B. Quantifying the error in estimated transfer functions with application to model order selection. *IEEE Transactions on Automatic Control*, 37(7):913–928, 1992.

Gora M. Stability of the convex combination of polynomials. *Control and Cybernetics*, 36, 2007.

Grant M., Boyd S., and Ye Y. Disciplined convex programming. In **Liberti L. and Maculan N.**, editors, *Global Optimization: from Theory to Implementation, Nonconvex Optimization and Its Applications*, pages 155–210. Springer, New York, 2006.

Guo L. Estimating time-varying parameters by the kalman filter based algorithm: stability and convergence. *Automatic Control, IEEE Transactions on*, 35(2):141–147, 1990.

Guo L., Ljung L., and Priouret P. Performance analysis of the forgetting factor rls algorithm. *International journal of adaptive control and signal processing*, 7(6): 525–538, 1993.

Haario H., Saksman E., and Tamminen J. An adaptive Metropolis algorithm. *Bernoulli*, 7, 2001.

Hannan E. and Deistler M. *The Statistical Theory of Linear Systems*. Wiley, 1988.

Hartikainen J. and Särkkä S. Kalman filtering and smoothing solutions to temporal gaussian process regression models. In *2010 IEEE International Workshop on Machine Learning for Signal Processing*, pages 379–384, Aug 2010.

Hastie T., Tibshirani R., and Friedman J. *The elements of statistical learning*. Springer series in statistics Springer, Berlin, 2008.

Hollerbach J., Khalil W., and Gautier M. Model identification. In *Springer Handbook of Robotics*, pages 321–344. Springer, 2008.

Huber M. F. Recursive gaussian process: On-line regression and learning. *Pattern Recognition Letters*, 45:85 – 91, 2014. ISSN 0167-8655. URL http://www.sciencedirect.com/science/article/pii/S0167865514000786.

Hunt K. J., Sbarbaro D., Żbikowski R., and Gawthrop P. J. Neural networks for control systems?a survey. *Automatica*, 28:1083–1112, 1992.

Ikeara S. A method of wiener in a nonlinear circuit. *Technical report, MIT*, 1951.

Ivaldi S., Fumagalli M., Randazzo M., Nori F., Metta G., and Sandini G. Computing robot internal/external wrenches by means of inertial, tactile and f/t sensors: theory and implementation on the icub. In *Humanoid Robots (Humanoids), 2011 11th IEEE-RAS International Conference on*, pages 521–528, 2011.

J. E. Dennis J. and Moré J. J. Quasi-newton methods, motivation and theory. *SIAM Review*, 19(1):46–89, 1977. URL http://dx.doi.org/10.1137/1019005.

James G., Witten D., Hastie T., and Tibshirani R. *An introduction to statistical learning*, volume 112. Springer, 2013.

Janabi-Sharifi F., Hayward V., and Chen C.-S. Discrete-time adaptive windowing for velocity estimation. *IEEE Transactions on control systems technology*, 8(6):1003–1009, 2000.

Jaynes E. T. and Kempthorne O. Confidence intervals vs bayesian intervals. In *Foundations of probability theory, statistical inference, and statistical theories of science*, pages 175–257. Springer, 1976.

Katayama T. *Subspace Methods for System Identification*. Communications and Control Engineering. Springer London, 2006. ISBN 9781846281587. URL https://books.google.it/books?id=Ge_HTdCBtZAC.

Kimeldorf G. and Wahba G. Some results on tchebycheffian spline functions. *Journal of Mathematical Analysis and Applications*, 33(1):82 – 95, 1971. ISSN 0022-247X. URL http://www.sciencedirect.com/science/article/pii/0022247X71901843.

Kozlowski K. R. *Modelling and identification in robotics*. Springer Science & Business Media, 2012.

Kubat M. Neural networks: a comprehensive foundation by simon haykin, macmillan, 1994, isbn 0-02-352781-7., 1999.

Lakshmikantham V., Leela S., and Martynyuk A. A. *Practical stability of nonlinear systems*. World Scientific, 1990.

Lawrence N., Seeger M., and Herbrich R. Fast sparse gaussian process methods: The informative vector machine. In *Proceedings of the 15th International Conference on Neural Information Processing Systems*, NIPS'02, pages 625–632, Cambridge, MA, USA, 2002. MIT Press. URL http://dl.acm.org/citation.cfm?id=2968618.2968696.

Leeb H. and Potscher B. Model selection and inference: Facts and fiction. *Econometric Theory*, 21(01):21–59, 2005.

Ljung L. *System Identification - Theory for the User*. Prentice-Hall, Upper Saddle River, N.J., 2nd edition, 1999.

Ljung L. . and Söderström T. *Theory and Practice of Recursive Identificationn*. Signal Processing, Optimization, and Control. The MIT Press, 1983.

Lozano L. Convergence analysis of recursive identification algorithms with forgetting factor. *Automatica*, 19(1):95–97, 1983.

Maciejowski J. M. *Predictive control: with constraints*. Pearson education, 2002.

Magni P., Bellazzi R., and Nicolao G. D. Bayesian function learning using mcmc methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(12): 1319–1331, Dec 1998. ISSN 0162-8828.

Matheron G. The intrinsic random functions and their applications. *Advances in applied probability*, pages 439–468, 1973.

Metta G., Natale L., Nori F., Sandini G., Vernon D., Fadiga L., Von Hofsten C., Rosander K., Lopes M., Santos-Victor J., and others . The icub humanoid robot: An open-systems platform for research in cognitive development. *Neural Networks*, 23(8):1125–1134, 2010.

Miller D. N. and de Callafon R. A. Subspace identification with eigenvalue constraints. *Automatica*, 49:2468–2473, 2013.

Nelles O. *Nonlinear system identification: from classical approaches to neural networks and fuzzy models*. Springer Science & Business Media, 2013.

Ng T., Goodwin G., and Andersson B. Identifiability conditions for linear multivariable systems operating under feedback. *Automatica*, 13:477–485, 1977.

Nguyen-Tuong D. and Peters J. Using model knowledge for learning inverse dynamics. In *IEEE International Conference on Robotics and Automation*, 2010.

Nguyen-Tuong D. and Peters J. Incremental online sparsification for model learning in real-time robot control. *Neurocomputing*, 74(11):1859–1867, 2011a.

Nguyen-Tuong D. and Peters J. Model learning for robot control: a survey. *Cognitive Processing*, 12(4):319–340, 2011b.

Nguyen-Tuong D., Seeger M., and Peters J. Model learning with local gaussian process regression. *Advanced Robotics*, 23(15):2015–2034, 2009.

Ninness B. and Henriksen S. Bayesian system identification via markov chain monte carlo techniques. *Automatica*, 46(1):40–51, 2010.

Nocedal J. and Wright S. J. *Numerical Optimization, second edition.* World Scientific, 2006.

Nori F., Traversaro S., Eljaik J., Romano F., Del Prete A., and Pucci D. icub whole-body control through force regulation on rigid non-coplanar contacts. *Frontiers in Robotics and AI*, page 18, 2015.

O'Hagan A. and Kingman J. Curve fitting and optimal design for prediction. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 1–42, 1978.

Overschee P. V. and Moor B. D. A unifying theorem for three subspace system identification algorithms. *Automatica*, 31(12):1853 – 1864, 1995. ISSN 0005-1098. URL http://www.sciencedirect.com/science/article/pii/0005109895000720.

Pan S. J. and Yang Q. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359, 2010.

Pattacini U., Nori F., Natale L., Metta G., and Sandini G. An experimental evaluation of a novel minimum-jerk cartesian controller for humanoid robots. *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1668–1674, 2010.

Pillonetto G. and Chiuso A. A bayesian learning approach to linear system identification with missing data. In *Proceedings of the 48th IEEE International Conference on Decision and Control 2009 Shangai China*, 2009.

Pillonetto G., Chiuso A., and Nicolao G. D. Prediction error identification of linear systems: a nonparametric Gaussian regression approach. *Automatica*, pages 291–305, 2011a.

Pillonetto G. and De Nicolao G. A new kernel-based approach for linear system identification. *Automatica*, 46(1):81–93, 2010.

Pillonetto G., Dinuzzo F., Chen T., Nicolao G. D., and Ljung L. Kernel methods in system identification, machine learning and function estimation: a survey. *Automatica*, 50(3):657–682, 2014.

Pillonetto G., Quang M., and Chiuso A. A new kernel-based approach for nonlinear system identification. *IEEE Transactions on Automatic Control [accepted]*, 2011b.

Pillonetto G. and Bell B. M. Bayes and empirical bayes semi-blind deconvolution using eigenfunctions of a prior covariance. *Automatica*, 43(10):1698 – 1712, 2007. ISSN 0005-1098. URL http://www.sciencedirect.com/science/article/pii/S0005109807001999.

Pillonetto G. and Chiuso A. Tuning complexity in regularized kernel-based regression and linear system identification: The robustness of the marginal likelihood estimator. *Automatica*, 58:106 – 117, 2015.

Pintelon R. and Schoukens J. *System identification: a frequency domain approach*. John Wiley & Sons, 2012.

Prando G., Romeres D., Pillonetto G., and Chiuso A. Classical vs. bayesian methods for linear system identification: point estimators and confidence sets. In *Proc. of ECC*, 2016a.

Prando G., Romeres D., and Chiuso A. Online identification of time-varying systems: a bayesian approach. In *Proc. of IEEE CDC*, 2016b.

Prüher J. and Simandl M. Gaussian process based recursive system identification. *Journal of Physics: Conference Series*, 570(1), 2014. URL http://stacks.iop.org/1742-6596/570/i=1/a=012002.

Qin S. J. An overview of subspace identification. *Computers & Chemical Engineering*, 30 (10–12):1502 – 1513, 2006. ISSN 0098-1354. URL http://www.sciencedirect.com/science/article/pii/S009813540600158X. Papers form Chemical Process Control {VIICPC} {VIISeventh} international conference in the Series.

Quinonero-Candela J. and Rasmussen C. E. A unifying view of sparse approximate gaussian process regression. *The Journal of Machine Learning Research*, 6:1939–1959, 2005.

Raftery A. and Lewis S. One long run with diagnostics: Implementation strategies for Markov chain Monte Carlo. *Statistical Science*, 7:493–497, 1992.

Rahimi A. and Recht B. Random features for large-scale kernel machines. In *Advances in neural information processing systems*, pages 1177–1184, 2007.

Ranganathan A., Yang M. H., and Ho J. Online sparse gaussian process regression and its applications. *IEEE Transactions on Image Processing*, 20(2):391–404, Feb 2011. ISSN 1057-7149.

Rasmussen C. and Williams C. *Gaussian Processes for Machine Learning*. The MIT Press, 2006.

Ripley B. D. *Spatial statistics*, volume 575. John Wiley & Sons, 2005.

Robbins H. The empirical bayes approach to statistical decision problems. In *Herbert Robbins Selected Papers*, pages 49–68. Springer, 1958.

Roberts G., Gelkman A., and Gilks W. *Weak convergence and optimal scaling of random walk Metropolis algorithms*, volume 7. Ann. Appl. Prob, 1997.

Romeres D., Prando G., Pillonetto G., and Chiuso A. On-line bayesian system identification. In *Proc. of ECC 2016*, 2016a.

Romeres D., Pillonetto G., and Chiuso A. Identification of stable models via nonparametric prediction error methods. In *Control Conference (ECC), 2015 European*, pages 2044–2049. IEEE, 2015.

Romeres D., Prando G., Pillonetto G., and Chiuso A. Online bayesian system identification. In *Proc. of ECC*, 2016b.

Romeres D., Zorzi M., and Chiuso A. Online semi-parametric learning for inverse dynamics modeling. In *Proc. of IEEE CDC*, 2016c.

Rugh W. J. and Shamma J. S. Research on gain scheduling. *Automatica*, 36(10): 1401–1425, 2000.

Sacks J., Welch W. J., Mitchell T. J., and Wynn H. P. Design and analysis of computer experiments. *Statistical science*, pages 409–423, 1989.

Saitoh S. *Theory of reproducing kernels and its applications*. Pitman research notes in mathematics series. Longman Scientific & Technical, 1988. ISBN 9780582035645. URL https://books.google.it/books?id=YzxPAQAAIAAJ.

Santner T. J., Williams B. J., and Notz W. I. *The design and analysis of computer experiments*. Springer Science & Business Media, 2013.

Scholkopf B. and Smola A. J. *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press, 2001.

Siciliano B., Sciavicco L., Villani L., and Oriolo G. *Robotics: modelling, planning and control*. Springer Science & Business Media, 2010.

Smola A. J. and Bartlett P. L. Sparse greedy gaussian process regression. In **Leen T. K., Dietterich T. G., and Tresp V.**, editors, *Advances in Neural Information Processing Systems 13*, pages 619–625. MIT Press, 2001. URL http://papers.nips.cc/paper/1880-sparse-greedy-gaussian-process-regression.pdf.

Snelson E. and Ghahramani Z. Sparse gaussian processes using pseudo-inputs. In *Advances in Neural Information Processing Systems*, pages 1257–1264. MIT press, 2006.

Söderström T., Ljung L., and Gustafsson I. Identifiability conditions for linear multivariable systems operating under feedback. *IEEE Trans. on Aut. Contr.*, 21: 837–840, 1976.

Söderström T. and Stoica P. *System Identification*. Prentice-Hall, 1989.

Sturm J. F. Using sedumi 1.02, a matlab toolbox for optimization over symmetric cones. *Optimization Methods and Software*, 11(1-4):625–653, 1999. URL http://dx.doi.org/10.1080/10556789908805766.

Tacchetti A., Mallapragada P., Santoro M., and Rosasco R. Gurls: A least squares library for supervised learning. *Journal of Machine Learning Research*, 14: 3201–3205, 2013.

Taylor J. *Classical Mechanics*. University Science Books, 2005. ISBN 9781891389221. URL https://books.google.de/books?id=P1kCtNr-pJsC.

Tikhonov A. and Arsenin V. *Solutions of Ill-Posed Problems*. Washington, D.C.: Winston/Wiley, 1977.

Tipping M. E. Sparse bayesian learning and the relevance vector machine. *Journal of Machine Learning Research*, 1:211–244, 2001.

Tresp V. A bayesian committee machine. *Neural Comput.*, 12(11):2719–2741, November 2000. ISSN 0899-7667. URL http://dx.doi.org/10.1162/089976600300014908.

Tsatsanis M. K. and Giannakis G. B. Time-varying system identification and model validation using wavelets. *Signal Processing, IEEE Transactions on*, 41(12):3512–3523, 1993.

Viberg M. Subspace-based methods for the identification of linear time-invariant systems. *Automatica*, 31(12):1835–1851, December 1995. ISSN 0005-1098. URL http://dx.doi.org/10.1016/0005-1098(95)00107-5.

Volterra V. *Theory of functionals and of integral and integro-differential equations*. Courier Corporation, 2005.

Wahba G. *Spline models for observational data*, volume 59. Siam, 1990.

Wahlberg B. Estimation of autoregressive moving-average models via high order autoregressive approximations. *Journal of Time Series Analysis*, 10(3):283–299, 1989.

Weyer E., Williamson R. C., and Mareels I. M. Y. Finite sample properties of linear model identification. *IEEE Transactions on Automatic Control*, 44:1370–1383, 1999.

Wiener N. *Extrapolation, interpolation, and smoothing of stationary time series*, volume 2. MIT press Cambridge, 1949.

Williams C. and Seeger M. Using the nyström method to speed up kernel machines. In *Advances in Neural Information Processing Systems 13*, pages 682–688. MIT Press, 2001.

Williams C. K. and Rasmussen C. E. Gaussian processes for regression. *Advances in Neural Information Processing Systems*, 1996.

Wipf D. P. and Nagarajan S. S. Iterative reweighted l1 and l2 methods for finding sparse solutions. *J. Sel. Topics Signal Processing*, 4(2):317–329, 2010.

Wu T. and Movellan J. Semi-parametric gaussian process for robot system identification. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 725–731, 2012.

Yao Y., Rosasco L., and Caponnetto A. On early stopping in gradient descent learning. *Constructive Approximation*, 26(2):289–315, 2007. ISSN 0176-4276. URL http://dx.doi.org/10.1007/s00365-006-0663-2.

Zadeh L. On the identification problem. *IRE Transactions on Circuit Theory*, 3(4): 277–281, 1956.

Zoubir A. M. and Boashash B. The bootstrap and its application in signal processing. *IEEE signal processing magazine*, 15(1):56–76, 1998.