UNIVERSITÀ
DEGLI STUDI
DI BERGAMO

University Of Bergamo

Doctoral Thesis

Regularized kernel-based learning for system identification

Author:
Matteo Scandella

Supervisor:
Prof. Fabio Previdi

Co-supervisors:
Prof. Simone Formentin
Prof. Mirko Mazzoleni

October 8, 2019

UNIVERSITY OF BERGAMO

Abstract

Department of Management, Information and Production Engineering

Doctor of Philosophy

Regularized kernel-based learning for system identification

by Matteo SCANDELLA

The problem of finding a good mathematical model of the phenomenon under analysis is a key topic in the control system community. In the past, this task was performed by experts of the field, but nowadays approaches that rely on experimental data and statistical learning techniques have seen an always increasing interest. For this reason, a lot of different learning techniques were adapted from the estimation of static relations performed by the statistical learning community to the identification of dynamical relations employed by control engineers. In recent times, kernel-based learning methods were employed for dynamical system modeling as part of this research trend. This thesis aims to further expand the knowledge about this important family of methods. In the first part, the theoretical foundation of this kind of techniques is presented in the necessary details. The second part contains the innovative contribution of the thesis. Firstly, it is shown that there exists more than one equivalent way to represent the identified model when dealing with kernel methods. Next, a new kernel approach for the identification of continuous-time model is proposed. Finally, the manifold regularization method in the case of dynamical systems identification is explored. Furthermore, a Bayesian perspective of the manifold regularization is provided. The thesis ends with a practical application of system identification using kernel methods in the field of nuclear physics.

Contents

List of Figures

List of Acronyms

AIC Akaike Information Criterion. 47, 162

ARMAX AutoRegressive Moving Average with an eXogenous variable. 2

ARX AutoRegressive with an eXogenous variable. 45–47

BIBO Bounded-Input Bounded-Output. 50, 55, 90

BIC Bayesian Information Criterion. 47

CHIMERA Charge Heavy Ions Mass and Energy Resolving Array. xi, 161, 163

CO_2 Carbon dioxide. 37

COD Complete Orthogonal Decomposition. 70, 75

CsI(Tl) Cesium Iodide crystals doped with Thallium. 163–165

DC Diagonal Correlated. ix, 51, 52, 56, 57, 86

dof degrees of freedom. 39–41, 138

FIR Finite Impulse Response. 3, 47, 130

FRF Frequency Response Function. 1, 2

GCV Generalized Cross-Validation. 3, 38, 41, 50, 137, 139, 142, 145, 158, 159

GP Gaussian process. ix, xi, 7, 24–29, 53, 146, 149, 162, 164, 167–170, 173, 177

IID Independent and Identically Distributed. 17, 20, 46, 54, 58, 66, 84, 130, 131, 145, 167

IIR Infinite Impulse Response. 47

k-NN k-Nearest Neighbors. 35

LCP Light Charged Particles. 161–163, 170, 174

LN1 Least Norm 1. x, 71–73, 104

LN2 Least Norm 2. x, 75, 77–79

LOOCV Leave One Out Cross-Validation. 39, 138

LPV Linear and Parameter-Varying. 2, 45

LTI Linear and Time-Invariant. x, 2–4, 45, 47, 50, 51, 54, 57, 84, 88, 89, 108, 111, 112, 120, 164, 165, 167, 169, 177

LTV Linear and Time-Varying. 2, 45

ML marginal likelihood. xi, 27, 38, 43, 53, 85, 142, 145, 149, 153–160, 168, 177

MSE Mean Square Error. 3

N4SID subspace state-space system identification. 162, 169, 170, 173

NARX Nonlinear AutoRegressive with an eXogenous variable. 57, 158

NFIR Nonlinear Finite Impulse Response. 3, 129, 132, 139, 141, 142

NMAE Normalized Mean Absolute Error. 142

NN neural network. 2, 3, 59, 172

OE Output Error. 2

pdf Probability Density Function. ix, xvi, 30, 41, 42, 85, 86

PEM Prediction Error Method. 1, 46, 47, 59, 162

RBF Radial Basis Functions. 11

RKHS Reproducing Kernel Hilbert space. 2, 4, 7–14, 16, 17, 19–21, 24, 37, 47, 54, 59, 60, 65, 66, 83, 85, 134, 177

RMS Root Mean Square. 25

SEM Simulation Error Method. 1, 59

SISO Single-Input Single-Output. 45, 165, 169

SNR Signal to Noise Ratio. 76, 106, 108, 112, 142

SRIVC Simple Refined Instrumental Variable for Continuous-time Output-Error models. x, 110, 112

SURE Stein's Unbiased Risk Estimate. 38

SVD Singular Value Decomposition. 169

WGN White Gaussian Noise. 106, 141, 158

YIC Young Information Criterion. 110

ZOH Zero Order Holder. 101

List of Notations

Important sets

- $\mathbb{N}$ is the set of all natural numbers;
- $\mathbb{Z}$ is the set of all integer numbers;
- $\mathbb{Q}$ is the set of all rational numbers;
- $\mathbb{Q}_+$ is the set of all strictly-positive natural number;
- $\mathbb{R}$ is the set of all real numbers;
- $\mathbb{R}_+$ is the set of all strictly-positive real number;
- $\mathbb{C}$ is the set of all complex numbers;
- l^p is the space of all p-summable sequence [108]:
  - l^1 is the space of sequences whose series is absolutely convergent;
  - l^2 is the space of square summable sequences;
  - l^∞ the space of bounded sequences;
- $L^p(\Omega, \mu)$ is the space of all p-inferable functions with domain Ω according to the measure μ [108]:
  - $L^2(\Omega, \mu)$ is the space of square-integrable functions;
  - $L^\infty(\Omega, \mu)$ is the space of bounded functions;

Mathematical constants

- $\pi = 3.141592653589793 \in \mathbb{R}$;
- $e = 2.718281828459046 \in \mathbb{R}$;
- j is the imaginary unit;

Vectors and matrices

Let $n, m \in \mathbb{N} \setminus \{0\}$.

- Generic scalars are indicated with a lower-case letter, e.g. a;
- Generic vectors are indicated with a lower-case bold letter, e.g. $\boldsymbol{v}$;
- Generic matrices are indicated with a upper-case bold letter, e.g. $\boldsymbol{A}$;

- Set of all real matrices with n rows and m columns: $\mathbb{R}^{n\times m}$;
- Identity matrix with n rows: $\boldsymbol{I}_n \in \mathbb{R}^{n\times n}$;
- Zero matrix with n rows and m columns $\mathbf{0}_{n\times m} \in \mathbb{R}^{n\times m}$;
- The transpose of the matrix $\boldsymbol{A}$ is $\boldsymbol{A}^\top$;
- The inverse of the invertible square matrix $\boldsymbol{A}$ is $\boldsymbol{A}^{-1}$;
- The determinant of the square matrix $\boldsymbol{A}$ is $\det \boldsymbol{A}$;
- The trace of a square matrix $\boldsymbol{A}$ is $\mathrm{Tr}\boldsymbol{A}$
- The rank of the matrix $\boldsymbol{A}$ is $\mathrm{rank}\boldsymbol{A}$;
- The vector $\boldsymbol{e}_{n,i} \in \mathbb{R}^{n\times 1}$ is a column vector with 1 at the i-th position and 0 in all the other positions;

Statistics

- Given a distribution p than $\mathrm{supp}\left(p\right)$ is the support of p;
- Given two independent random variables a and b then $a \perp b$
- The expected value of a random variable X is indicated as

$$\mathbb{E}\left[\varphi\left(X\right)\right] = \int \varphi\left(x\right) p\left(x\right)\, dx \tag{1}$$

  where φ is some function and p the pdf of the distribution of X;
- The variance of a random variable X is indicated as $\mathrm{Var}\left(X\right)$;
- The covariance between two random variables X and Y is indicated as $\mathrm{Cov}\left(X, Y\right)$;
- The normal distribution with mean $\boldsymbol{\mu} \in \mathbb{R}^{n\times 1}$ and variance $\boldsymbol{\Sigma} \in \mathbb{R}^{n\times n}$ is $\mathcal{N}\left(\boldsymbol{\mu}, \boldsymbol{\Sigma}\right)$;
- The pdf of the normal distribution with mean $\boldsymbol{\mu} \in \mathbb{R}^{n\times 1}$ and variance $\boldsymbol{\Sigma} \in \mathbb{R}^{n\times n}$ evaluated in $\boldsymbol{x} \in \mathbb{R}^{n\times 1}$ is $\mathcal{N}\left(\boldsymbol{x} \left| \boldsymbol{\mu}, \boldsymbol{\Sigma}\right.\right)$;
- The uniform distribution between $a \in \mathbb{R}$ and $b \in \mathbb{R}$ is $\mathcal{U}\left(a, b\right)$;
- The pdf of the uniform distribution between $a \in \mathbb{R}$ and $b \in \mathbb{R}$ evaluated in $x \in \mathbb{R}$ is $\mathcal{U}\left(x \left| a, b\right.\right)$;

Dynamical system theory

- Generic dynamical systems are indicated with a formal upper-case letter, e.g. $\mathscr{G}$;
- Transfer functions are indicated with an upper-case letter, e.g. G;
- The Laplace variable is indicated with $s \in \mathbb{C}$;
- The Laplace transform of $x\left(t\right)$ is indicated with $X\left(s\right) = \mathcal{L}\left[x\right]\left(s\right)$;
- The Laplace anti-transform of $X\left(s\right)$ is indicated with $x\left(t\right) = \mathcal{L}^{-1}\left[X\right]\left(t\right)$
- The Fourier transform of $x\left(t\right)$ is indicated with $X\left(\omega\right) = \mathcal{F}\left[x\right]\left(s\right)$;
- The Fourier anti-transform of $X\left(\omega\right)$ is indicated with $x\left(t\right) = \mathcal{F}^{-1}\left[X\right]\left(t\right)$;
- The convolution of two functions is indicated with $\left[a \star b\right]\left(t\right)$;

Other notations

- Generic sets are indicated with a calligraphic upper-case letter, e.g. $\mathcal{G}$;
- Generic functions are indicated with a lower case letter, e.g. g;
- The Beta function [88] is indicated with $B(a, b)$
- Given a function $g : \mathbb{R}^d \to \mathbb{R}$ than

  $$\nabla g(\boldsymbol{x}) = \begin{bmatrix} \frac{\partial}{\partial x_1} g(\boldsymbol{x}) & \cdots & \frac{\partial}{\partial x_d} g(\boldsymbol{x}) \end{bmatrix} \in \mathbb{R}^{1 \times d} \tag{2}$$

  is the gradient of g.
- $\overline{\mathcal{A}}$ is the closure of the set $\mathcal{A}$;
- Given $\mathcal{A} \subseteq \mathcal{B}$, where $\mathcal{B}$ is a vector space, then $\operatorname{span}(\mathcal{A})$ is the vector space containing all the finite linear combinations of the elements of $\mathcal{A}$;

Introduction

Context of the Thesis

Understanding the natural phenomena that happen around us is the ultimate aim of science. This knowledge can be expressed in various forms, but the more universal one employs mathematical models. These abstract entities are composed of mathematical objects, and they provide a simplified version of the natural phenomenon they are describing. For this reason, they are fundamental in modern science and engineering and they are employed in the most diverse fields. For this reason, the activities that aim at the construction of models are an important keystone of the current society.

In most cases, modeling is an activity that is performed mainly by experts of the phenomenon that could exploit their knowledge to provide a complex and accurate model. This approach is called *white-box modeling* and it is by far the more common way to approach the problem. However, this is an expensive procedure and, in most cases, the people that plan to use the model have different needs to the one that provides the model and the complexity of a white-box model is not always desirable. Therefore, in the last decades, a different modeling approach has seen increasing interest: *black-box modeling*. This rationale exploits statistical learning methods coupled with a set of observations taken from the phenomenon under study. Black-box modeling for static systems has seen an always increasing interest thanks to the explosion of computational power. The technique that are used for this aim can be divided in two categories based on the type of system that is under analysis. When the variables of a system depend only on the values of the other variables taken at the same time, then the system is called *static system*. The modeling of this type of systems is tackled by the machine learning community [17, 44, 125]. Vice versa, the system identification community [19, 72, 98, 117, 128] deals with the so-called *dynamical system*. In this type of model the variables depend also on the what happen in the past.

Most system identification methodologies rely on the Prediction Error Method (PEM) rationale [19, 72, 117]. Here, the identified model is selected from a certain set, called hypothesis set, as the one that minimizes the prediction error on the available dataset. An important alternative rationale that can be found in the literature is the Simulation Error Method (SEM) approach [20, 99, 100] where the selected model is the one that minimizes the simulation error on the available dataset. Other methodologies leverage some of the properties of a dynamical system to avoid the needs of an optimization technique, e.g. subspace methods [41, 89, 103] or non-parametric Frequency Response Function (FRF) estimation [68, 98]. In general, all the various rationales select the identified model from a certain hypothesis set. Therefore, the choice of the right set is a fundamental part of the identification procedure.

The hypothesis set determines the type of system we want to identify and its complexity. For example, the simplest and most studied class of systems is composed of Linear and Time-Invariant (LTI) systems [19, 46, 72, 95, 98, 128]. However, it is also possible to use more complex families such as Linear and Time-Varying (LTV) [29, 68, 69], Linear and Parameter-Varying (LPV) [8, 26, 93, 123, 127] or non-linear systems [28, 86, 97, 100, 133]. Understanding what is the best hypothesis set, for the application at hand, is a difficult task, but a fundamental one. In fact, the identified system should be the simplest one that reaches satisfactory performance for the application at hand.

Based on the type of hypothesis space the various algorithms can be divided into two categories: parametric [19, 46, 72, 100] and non-parametric [68, 95, 98]. In the first class, the hypothesis space has a known bijection with $\mathbb{R}^d$ where d is a certain natural number. In this case, the algorithm boils down to the selection of the best d-length vector that corresponds to the best model in the hypothesis set according to the used criterion. This vector is typically called the parameters vector. In these settings, among others, finds place methods that employ ARMAX [19, 72] or OE [47, 95] models, for linear systems, or neural network [28, 92], wavelets [133], or polynomials [99] for non-linear systems. Vice-versa, for the non-parametric methods this explicit parametrization is not possible. For example, in this category, are present various FRF estimators [68, 98] or the kernel methods [37, 51].

The focus of this Thesis is on the non-parametric kernel methods for system identification. These methods expand the theory of linear modeling from data allowing the use of an infinite amount of features to characterize the system behavior. Usually, kernel methods are endowed with a regularization term, such as Tikhonov regularization [111] or manifold regularization [11], that equips the method with a flexible way to tune the complexity of the estimated model and to deal with overfitting problems. In this framework, the estimation is recast into an optimization problem inside a, potentially, infinite-dimensional Reproducing Kernel Hilbert space (RKHS) [4, 109]. Thanks to the Representer theorem [11, 40, 111], such a problem boils down to finite-dimensional optimization, whose solution can be treated analytically, if the cost is quadratic. These kind of approaches have been successfully employed for dynamical models of various types, such as LTI [30, 95], LPV [123], LTV [68] or non-linear [97] systems.

New contributions of the Thesis

This thesis aims to expand the knowledge about kernel methods for the system identification problem. For this reason, the thesis contains four different new theoretical contributions and a practical application in the field of nuclear physics.

The first contribution deals with a typical case of many practical applications, where the kernel is truncated into a degenerate kernel due to limited numerical precision. As a consequence, the optimization problem is no longer strictly convex and infinite equivalent solutions exist. The main message that is conveyed is that such an apparent problem actually allows enforcing some additional desired properties on the estimated model. In particular, it is shown that this additional freedom can be used to: **(i)** select the solution that minimizes the number of features (thus reducing the computational requirements to perform predictions on new data); **(ii)** tackle the ill-conditioning of the manifold regularization for semi-supervised problems [11].

Most of the system-identification literature is based on discrete-time models [19, 72, 100] for the discrete nature of input/output measurements. However, continuous-time models are the most used for control and analysis purpose, and they are not constrained to a certain sampling frequency. For this reason, the second contribution introduces a novel black-box non-parametric kernel-based technique for the identification of continuous-time LTI systems based on the work of [95]. In particular, the focus will be on dataset taken from low exciting inputs such as the step signal because they are very common excitation that can be applied to almost every real system. Additionally, the proposed method does not require evenly sampled data but can also work with irregularly sampled one.

The third contribution investigates the kernel-based estimation of nonlinear dynamical systems via regularization using artificially augmented datasets. Such idea seems particularly promising in all applications where there is some prior knowledge about the system [10, 11, 27], but only a small amount of data are available as running new experiments is difficult or too costly, e.g. some biomedical systems like glucose dynamics [64] or complex industrial plants [135]. More specifically, the focus will be on Nonlinear Finite Impulse Response (NFIR) systems [132], in that they represent a wide range of applications [5, 80] and, for such models, augmented regressors can be generated without running new experiments on the systems. The author proposes a novel way to generate artificial data that can be employed by a manifold regularization term [11] to improve the estimation. This type of regularization penalizes the roughness of the unknown function alongside the manifold where the regressors are supposedly constrained. Typically, it is used to solve semi-supervised learning problems or when the regressor distribution contains some information on the system [11, 87]. In these settings, the manifold is approximated using a graph that links the regressors to their neighbor [11, 15, 115]. For this reason, it is, also, presented a novel approach for the selection of the graph topology that exploits the properties of the dynamical behavior of the system.

Employing the manifold regularization, however, introduces some hyper-parameters that have to be tuned. The role of hyperparameters selection is similar to that of model order determination in traditional parametric system identification, with the difference that now we are not restricted to choose from a discrete grid of values. Common methods for hyperparameters estimation consist in various cross-validation strategies such as Generalized Cross-Validation (GCV) [44, 82] or the maximum likelihood methods [82, 104]. Even though the maximum likelihood estimator has been shown to not converge to an "optimal" estimate in a specific Mean Square Error (MSE) sense [82] (as opposed to the GCV approach), it was observed how maximum likelihood can better balance the trade-off between data fit and model complexity. It is important to notice that the maximum likelihood approach is available only when the regularized problems admits also a Bayesian interpretation [31, 95, 97, 104]. For this reason, as a fourth contribution, it is shown a novel Bayesian interpretation of manifold regularization, that allows hyperparameters to be tuned using this proven methodology.

To end the thesis, a practical application of kernel methods in the context of nuclear physics is presented. To characterize the property of particles, physicists perform an experiment that allows measuring the energy decay after a collision of the particles with a target [1, 74]. Currently, the measured signal is analyzed by an expert that can manually classify the particle by looking at the signal shape [42]. Here, the author aim to automate this procedure using a sequence of system identification techniques and machine learning ones. In particular, the signal produced by the sensor is very similar to a truncated impulse response of a linear system. Therefore, a non-parametric kernel-based identification technique is used to identify a high-dimensional Finite Impulse Response (FIR) system that can be reduced to a simpler model using a model reduction technique. After that, a neural network is trained to classify the particles using the time constants of the identified model.

Thesis structure

The remainder of the Thesis is organized as follow:

- **Chapter 1** explains the basic concepts behind the learning methods that relies on RKHS for dynamical systems;
- **Chapter 2** continues the previous Chapter by explaining how the kernel methods can be employed for dynamical system identification;
- **Chapter 3** delves into the first new contribution of this Thesis by explaining how to leverage the degeneracy of the kernel;
- **Chapter 4** illustrates the proposed continuous-time LTI identification approach of the second contribution;
- **Chapter 5** explains how to employ the manifold regularization for non-linear dynamical systems;
- **Chapter 6** presents the Bayesian perspective when employing the manifold regularization;
- **Chapter 7** describes the application of the kernel methods for the classification of nuclear particles;

PART I

State of the art on kernel-based system identification

CHAPTER 1

Kernel-based learning methods for static models

This chapter briefly reviews the literature about kernel-based learning for the identification of a non-linear function. In particular, the following sections delve into the Reproducing Kernel Hilbert space (RKHS) and their application for non-linear regression. The Tikhonov and manifold regularizations will be introduced as methods to regulate the bias-variance trade-off. Furthermore, the semi-supervised regression using manifold learning and RKHS is going to be briefly discussed.

The same concepts are, then, reviewed from a Bayesian perspective using the so-called Gaussian process regression. This allows defining a way to tune the hyper-parameters of the method and it gives a different way to interpret them.

This chapter is organized as follow:

- Section 1.1 introduces the concept of Reproducing Kernel Hilbert space;
- Section 1.2 explains how to use RKHS to identify non-linear functions;
- Section 1.3 delves into the Bayesian perspective of the method explained in the previous sections;
- Section 1.4 introduces a different regularization method used for the semi-supervised learning;
- Section 1.5 contains an explanation on how to select the various hyper-parameters.

1.1 Reproducing Kernel Hilbert Spaces

This section lays the basis to the theory behind the Reproducing Kernel Hilbert space (RKHS). Since these spaces are a special case of Hilbert spaces, to follow this section, some background of functional analysis is needed. This knowledge can be found in a lot of different mathematical books [108, 110].

1.1.1 RKHS definition and basic properties

Definition 1.1 (RKHS). *Let $\mathcal{H}$ be an Hilbert space with inner product $\langle\cdot,\cdot\rangle_{\mathcal{H}}$ and norm $\|\cdot\|_{\mathcal{H}}$. The space $\mathcal{H}$ is called **RKHS** if and only if:*

a) *its elements are real functions that share the same domain $\mathcal{X}$*

$$u \in \mathcal{H} \to u : \mathcal{X} \to \mathbb{R} \tag{1.1}$$

b) *for each element $x \in \mathcal{X}$, the evaluator functional L_x, i.e.*

$$L_x : \mathcal{H} \to \mathbb{R} \tag{1.2}$$
$$u \to u\left(x\right) \tag{1.3}$$

is linear and continuous.

Remark 1.1. The evaluator functional L_x is always linear because we have

$$L_x\left(\alpha u + \beta v\right) = \left(\alpha u + \beta v\right)\left(x\right) \tag{1.4}$$
$$= \alpha u\left(x\right) + \beta v\left(x\right) \tag{1.5}$$
$$= \alpha L_x\left(u\right) + \beta L_x\left(v\right) \tag{1.6}$$

where $\alpha, \beta \in \mathbb{R}$, $x \in \mathcal{X}$ and $u, v \in \mathcal{H}$.

Now, Let x be an element of $\mathcal{X}$ and L_x be its evaluator functional. Since L_x is continuous and linear, we have that $L_x \in \mathcal{H}^*$, where $\mathcal{H}^*$ is the dual space [110] of $\mathcal{H}$. Therefore, thanks to the Riesz-Fréchet representation theorem [105, 110], there exists a function $r_x \in \mathcal{H}$ such that

$$L_x\left(u\right) = \langle r_x, u\rangle_{\mathcal{H}} \qquad \forall u \in \mathcal{H} \tag{1.7}$$

then, for the definition of L_x, we can write

$$u\left(x\right) = \langle r_x, u\rangle_{\mathcal{H}} \qquad \forall u \in \mathcal{H} \tag{1.8}$$

this important property, called *reproducing property*, allows evaluating all the functions inside the RKHS. The only requirement is to find a way to associate each element $x \in \mathcal{X}$ to its right function r_x.

Definition 1.2 (Representer function). *Let $x \in \mathcal{X}$, then the function $r_x \in \mathcal{H}$, such that $u\left(x\right) = \langle r_x, u\rangle_{\mathcal{H}}$ $\forall u \in \mathcal{H}$, is called **representer function** of x.*

These representer functions allow defining the entire RKHS as stated by the following theorem.

Theorem 1.1. *Let $\mathcal{H}$ be an RKHS containing functions with domain $\mathcal{X}$. Then*

$$\mathcal{H} = \text{span}\left\{r_x | x \in \mathcal{X}\right\} \tag{1.9}$$

Remark 1.2. From the definition of span, the relation (1.9) can be rewritten as

$$\mathcal{H} = \bigcup_{n=1}^{\infty} \left\{\sum_{i=1}^{n} c_i r_{x_i} \text{ s.t. } x_1, \ldots, x_n \in \mathcal{X} \text{ and } c_1, \ldots, c_n \in \mathbb{R}\right\} \tag{1.10}$$

therefore, given $u \in \mathcal{H}$ they exist $x_1, \ldots, x_n \in \mathcal{X}$ and $c_1, \ldots, c_n \in \mathbb{R}$ such that

$$u(z) = \sum_{i=1}^{n} c_i r_{x_i}(z) \tag{1.11}$$

where $z \in \mathcal{X}$ is a generic argument.

Now, it is possible to introduce the concept of *reproducing kernel.*

Definition 1.3 (Reproducing kernel)**.** *Let $\mathcal{H}$ be an RKHS containing functions with domain $\mathcal{X}$. Then the function*

$$k : \mathcal{X} \times \mathcal{X} \to \mathbb{R} \tag{1.12}$$

$$(x, y) \to \langle r_x, r_y \rangle_{\mathcal{H}} \tag{1.13}$$

is called ***reproducing kernel*** *of $\mathcal{H}$.*

The reproducing kernel k of an RKHS $\mathcal{H}$ is a fundamental object for two reasons: **(i)** it fully characterize its RKHS and **(ii)** it provides a way to easily define new RKHS. The first statement is due to the following theorem.

Theorem 1.2. *Let $\mathcal{H}_1$ and $\mathcal{H}_2$ be two RKHS. If they admit the same reproducing kernel k, then $\mathcal{H}_1 = \mathcal{H}_2$.*

The second statement is a direct consequence of the following theorem.

Theorem 1.3 (Moore-Aronszajn Theorem [4])**.** *Given a function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ such that:*

- *the function is symmetric*

$$k(x, y) = k(y, x) \qquad \forall x, y \in \mathcal{X} \tag{1.14}$$

- *the function is positive semi-definite*

$$\sum_{i=1}^{n} \sum_{j=1}^{n} c_i c_j k(x_i, x_j) \geq 0 \qquad \begin{array}{l} \forall n \in \mathbb{N} \setminus \{0\} \\ \forall \boldsymbol{c} = [c_1, \ldots c_n] \in \mathbb{R}^{1 \times n} \\ \forall \boldsymbol{x} = [x_1, \ldots x_n] \in \mathcal{X}^{1 \times n} \end{array} \tag{1.15}$$

then, there exists an RKHS $\mathcal{H}$ with reproducing kernel k.

This theorem tells us that a function to be a valid reproducing kernel has to be symmetric and positive semi-definite. Furthermore, every function with these two properties is a valid kernel of a certain RKHS. Therefore, to define an RKHS it is enough to find a function with these properties.

Remark 1.3. Given the kernel k, is always possible to find the various representer functions r_x of the same RKHS. In particular, the evaluation of the representer of $x \in \mathcal{X}$ in $y \in \mathcal{X}$ is

$$r_x(y) = \langle r_x, r_y \rangle = k(x, y) \tag{1.16}$$

Therefore

$$r_x = k(x, \cdot) \tag{1.17}$$

For this reason, the representer functions are often called *kernel slice.*

Remark 1.4. The second condition of Theorem 1.3 requires that the kernel function is positive semi-definite. This condition is equivalent to ask that the matrix $\boldsymbol{K} \in \mathbb{R}^{n\times n}$ whose element (i,j) is $k(x_i, x_j)$ is a positive semi-definite matrix $\forall \boldsymbol{x} = [x_1, \dots x_n] \in \mathcal{X}^{1\times n}$ and $\forall n \in \mathbb{N} \setminus \{0\}$.

To better understand these concepts, consider the following examples.

Example 1.1: Constant kernel

The constant kernel is the simplest kernel and it is defined as

$$k : \mathcal{X} \times \mathcal{X} \to \mathbb{R} \tag{1.18}$$

$$(x, y) = 1 \tag{1.19}$$

It is straightforward to see that the two condition of Theorem 1.3 are respected. Following (1.17), the representer of $x \in \mathcal{X}$ is

$$r_x = k(x, \cdot) = 1 \tag{1.20}$$

From Remark 1.1, the RKHS associated with the constant kernel is the span of a set containing constant functions. Therefore, the constant kernel defines the space of all constant functions.

Example 1.2: Linear kernel

Another very simple kernel is the linear kernel.

$$k : \mathbb{R}^{d\times 1} \times \mathbb{R}^{d\times 1} \to \mathbb{R} \tag{1.21}$$

$$(\boldsymbol{x}, \boldsymbol{y}) = \boldsymbol{x}^\top \boldsymbol{y} \tag{1.22}$$

where $d \in \mathbb{N} \setminus \{0\}$.
It is trivial to see that this kernel is symmetric and positive semi-definite. Therefore, the conditions of Theorem 1.3 are respected.
Following (1.17), the representer of $x \in \mathbb{R}^{d\times 1}$ is

$$r_x(z) = x^\top z \tag{1.23}$$

From Theorem 1.2, the RKHS associated with the linear kernel is the span of a set containing linear functions. Therefore, there exist $n \in \mathbb{N} \setminus \{0\}$, $\boldsymbol{x}_1, \dots, \boldsymbol{x}_n \in \mathbb{R}^{d\times 1}$ and $c_1, \dots, c_n \in \mathbb{R}$ such that the generic function $u \in \mathcal{H}$ evaluated in $\boldsymbol{z} \in \mathbb{R}^{d\times 1}$ can be written as

$$u(\boldsymbol{z}) = \sum_{i=1}^{n} c_i r_{\boldsymbol{x}_i}(\boldsymbol{z}) = \sum_{i=1}^{n} c_i \boldsymbol{x}_i^\top \boldsymbol{z} = \left(\sum_{i=1}^{n} c_i \boldsymbol{x}_i\right)^\top \boldsymbol{z} = \boldsymbol{w}^\top \boldsymbol{z} \tag{1.24}$$

Now, we can see that the generic function u is a linear function and therefore the RKHS $\mathcal{H}$, defined using the linear kernel, is the space of all linear function.

Example 1.3: Gaussian kernel

This kernel is one of the most utilized because it can be shown that its corresponding RKHS contains a good approximation for each square-integrable functions. The

Gaussian kernel, often called *squared-exponential kernel*, *heat kernel* or *RBF kernel*, is defined as follow:

$$k : \mathcal{X} \times \mathcal{X} \to \mathbb{R} \tag{1.25}$$

$$(x, y) = e^{-\frac{d(x-y)^2}{\sigma^2}} \tag{1.26}$$

where $\sigma \in \mathbb{R}_+$ is a positive constants often called *width* of the kernel and d is a valid distance defined on the set $\mathcal{X}$. For example, if $\mathcal{X} \in \mathbb{R}^d$, we can use the Euclidian distance, i.e.

$$d(\boldsymbol{x} - \boldsymbol{y}) = \|\boldsymbol{x} - \boldsymbol{y}\|_2 \tag{1.27}$$

Some in-depth analysis of this kernel can be found in the literature [119, 129] and some other properties will be discussed later in this chapter.

Example 1.4: Space of band limited function

Consider the set containing all the band-limited functions

$$\mathcal{H} = \left\{u \in L^2(\mathbb{R}) \text{ s.t. } \operatorname{supp}(\mathcal{F}[u]) \subset [-a, a]\right\} \tag{1.28}$$

where $a \in \mathbb{R}_+$ is the maximum frequency of the functions inside the set. It is possible to show that this, in fact, a Hilbert space with the inner product

$$\langle u, v \rangle_{\mathcal{H}} = \int_{-\infty}^{+\infty} u(x)\, v(x)\; dx \tag{1.29}$$

furthermore, it can be shown that the evaluator functional of the functions inside $\mathcal{H}$ is continuous.
For these reason, $\mathcal{H}$ is a valid RKHS. Its kernel is

$$k_{\mathcal{H}}(y, x) = \frac{\sin(a(y - x))}{\pi(y - x)} \tag{1.30}$$

therefore, for Remark 1.2, there exist $n \in \mathbb{N} \setminus \{0\}, t_1, \ldots, t_n \in \mathbb{R}$ and $c_1, \ldots, c_n \in \mathbb{R}$ such that the generic function $u \in \mathcal{H}$ evaluated in $t \in \mathcal{X}$ can be written as

$$u(t) = \sum_{i=1}^{n} c_i r_{t_i}(t) = \sum_{i=1}^{n} c_i \frac{\sin(a(t - t_i))}{\pi(t - t_i)} \tag{1.31}$$

Now, if we set $c_i = u(t_i)$ and $t_i = \frac{i}{2a}$, we can note that this formula correspond to the Whittaker–Shannon interpolation formula used to reconstruct band-limited signals from a set of samples taken with a sampling frequency of $2a$.

Example 1.5: Spline kernel

Another famous kernel is the spline kernel [131] that is defined as

$$k(x,y) = \int_0^1 G_q(a,x)\, G_q(b,x)\ dx \tag{1.32}$$

where $q \in \mathbb{N} \setminus \{0\}$ and

$$G_q(a,x) = \frac{1}{(q-1)!} \begin{cases} (a-x)^{q-1} & \text{if } a \geq x \\ 0 & \text{if } a < x \end{cases} \tag{1.33}$$

This kernel defines the following RKHS

$$\mathcal{H} = \left\{ u \in L_2([0,1]) \text{ s.t. } u^{(m)}(0), \forall m = 0, \dots q-1 \text{ and } u \text{ is continuous} \right\} \tag{1.34}$$

Therefore, it contains all the continuous functions that have the first $q-1$ derivative evaluated in 0 equal to 0.

1.1.2 Mercer theorem

Using the theorems shown before, it is possible to define an RKHS and to understand what kind of functions are contained in the space, but it is not easy to understand the norm $\|\cdot\|_{\mathcal{H}}$ and the inner product $\langle \cdot, \cdot \rangle_{\mathcal{H}}$ of the space. To do so, we will consider only the case where:

- $\mathcal{X}$ is a compact set with a probability distribution π defined on it;
- the kernel k is continuous on $\mathcal{X} \times \mathcal{X}$.

These are not hard restrictions. For example, all the kernels cited before respect these condition in this category (with an appropriate restricted domain if necessary).

In these settings, it is possible to prove the following important theorem.

Theorem 1.4 (Mercer Theorem [109])**.** *Let k be a continuous valid kernel and $\mathcal{H}$ its corresponding RKHS. Then the operator $T : L_2(\mathcal{X}, \pi) \to L_2(\mathcal{X}, \pi)$ defined as*

$$T[u](x) = \int_{\mathcal{X}} k(x,y)\, u(y)\, d\pi(y) \tag{1.35}$$

has the following properties:

1. *the eigenfunctions $\{\varphi_i\}_i^\infty$ of T are an orthonormal base of $L_2(\mathcal{X}, \pi)$;*
2. *the eigenvalues $\{\sigma_i\}_i^\infty$ of T are all non-negative with finite multiplicity;*
3. *all the eigenfunctions $\{\varphi_i\}_i^\infty$ of T are elements of the RKHS $\mathcal{H}$;*
4. *the functions $\psi_i = \sigma_i \varphi_i$ compose a orthonormal base of $\mathcal{H}$ (by removing the ones with the corresponding eigenvalue equal to 0);*
5. *a function $u \in \mathcal{H}$ if and only if*

$$M(u) = \sum_{i=1}^{\infty} \frac{\langle u, \varphi_i \rangle_\pi^2}{\sigma_i^2} < +\infty \tag{1.36}$$

where $\langle \cdot, \cdot \rangle_\pi$ is the $L_2(\mathcal{X}, \pi)$ inner product and the 0 eigenvalues are removed from the sum;

6. *the induced norm of $u \in \mathcal{H}$ according to the RKHS $\mathcal{H}$ is*

$$\|u\|_{\mathcal{H}}^2 = M(u); \tag{1.37}$$

7. *the kernel k evaluated in $(x, y) \in \mathcal{X} \times \mathcal{X}$ can be written as*

$$k(x, y) = \sum_{i=1}^{\infty} \sigma_i^2 \varphi_i(x) \varphi_i(y) \tag{1.38}$$

$$= \sum_{i=1}^{\infty} \psi_i(x) \psi_i(y) \tag{1.39}$$

*and this series converges for every value of $(x, y) \in \mathcal{X} \times \mathcal{X}$. This formulation is called **Mercer expansion** of the kernel.*

This theorem provides a lot of information about the space $\mathcal{H}$. First of all, it shows a way to compute an orthonormal base of the space and, therefore, to understand its dimension. Furthermore, it provides a different condition to check if a function is inside the space and a way to analyze the behavior of the induced norm of the space $\mathcal{H}$.

Following the theorem, the dimension of the space is equal to the number of eigenvalues σ_i that are not zero. Based on this fact, it is possible to classify the RKHS spaces into two categories.

Definition 1.4 (Degenerate and non-degenerate kernels)**.** *A kernel k and its corresponding RKHS $\mathcal{H}$ are called **degenerate** if and only if there is only a finite number of eigenvalues σ_i, as defined in Theorem 1.4, that are strictly positive. Otherwise, they are called **non-degenerate**.*

Degenerate kernels have a finite dimension and their Mercer expansion (1.38) is a finite summation while for the non-degenerate kernels the mercer expansion is a convergent series. For example, the linear kernel, as described in Example 1.2, is degenerate with dimension d and the Gaussian kernel is non-degenerate with a not finite dimension.

Consider, now, the norm (1.37) of the function $u \in \mathcal{H}$. This term is a summation of ratios between

- the projection of the function u on the eigenfunction ψ_i;
- the square of the associated eigenvalue σ_i;

therefore a function has a large norm when the projections on the eigenfunctions, associated with a small eigenvalue, are significant. For this reason, the functions with large norm are the one that behaves more "similarly" to the eigenfunctions with small eigenvalues and vice versa the functions with a small norm are "similar" to the eigenfunctions with large eigenvalue. To better understand this concept consider the following example.

Example 1.6: Gaussian kernel eigenfunctions and eigenvalues

In general, it is not straightforward to compute the eigenfunctions and the eigenvalues of the Mercer expansion. For the Gaussian kernel there are some theoretical results [104, 119, 138].

Consider the case where $\mathcal{X} \subset \mathbb{R}$, π is a normal distribution with mean 0 and variance η^2, i.e. $\mathcal{N}\left(0, \eta^2\right)$, and the norm used is the Euclidian norm. Here, the kernel is

$$k\left(x, y\right) = e^{-\frac{(x-y)^2}{\sigma^2}} \tag{1.40}$$

where $\sigma \in \mathbb{R}$ is strictly positive.
It can be shown that the i-th eigenfunction and eigenvalue (ordered from the largest eigenvalue to the smallest) is:

$$\sigma_i^2 = \sqrt{\frac{2a}{A}}\left(\frac{b}{A}\right)^i \tag{1.41}$$

$$\psi_i\left(x\right) = e^{(a-c)x^2} H_i\left(\sqrt{2c}x\right) \tag{1.42}$$

where $a = 4^{-1}\eta^{-2}$, $b = \sigma^{-2}$, $c = \sqrt{a^2 + 2ab}$, $A = a+b+c$ and H_i is the Hermitian polynomial [53] of order i. The first four eigenfunctions can be seen in Figure 1.1 and the first twenty eigenvalues are reported in Figure 1.2.
From these plots, it is possible to note that the eigenfunctions become more oscillating when their corresponding eigenvalues decrease. For this reason, the norm associated with the Gaussian kernel increases when the function is more oscillating. In particular, it is possible to show that [75, 129]

$$\|u\|_{\mathcal{H}}^2 = \frac{1}{2\pi}\int_{-\infty}^{+\infty} \left|\mathcal{F}\left[u\right]\left(\omega\right)\right|^2 e^{\frac{\sigma^2\omega^2}{2}}\, d\omega \tag{1.43}$$

where it is possible to note that the norm becomes bigger when the function u contains large components at higher frequencies.

1.1.3 Defining new kernels

In the previous sections, it is explained how to analyze a kernel and its corresponding RKHS, but it is not shown how to define new and more complex kernels. In order to do so, consider the following theorems.

Theorem 1.5 (Sum of kernels [109]). *Let $k_1 : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ and $k_2 : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ be two valid kernels that define, respectively, the spaces $\mathcal{H}_1$ and $\mathcal{H}_2$ and $a, b \in \mathbb{R}$ be strictly positive real numbers. Then the function*

$$k : \mathcal{X} \times \mathcal{X} \to \mathbb{R} \tag{1.44}$$

$$\left(x, y\right) \to ak_1\left(x, y\right) + bk_2\left(x, y\right) \tag{1.45}$$

is a valid kernel and it defines the space

$$\mathcal{H} = \left\{u \text{ s.t. } \exists u_1 \in \mathcal{H}_1, u_2 \in \mathcal{H}_2 \text{ s.t. } u\left(x\right) = au_1\left(x\right) + bu_2\left(x\right), \forall x \in \mathcal{X}\right\} \tag{1.46}$$

Theorem 1.6 (Product of kernels [109]). *Let $k_1 : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ and $k_2 : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ be two valid kernels that defines, respectively, the spaces $\mathcal{H}_1$ and $\mathcal{H}_2$. Then the function*

$$k : \mathcal{X} \times \mathcal{X} \to \mathbb{R} \tag{1.47}$$

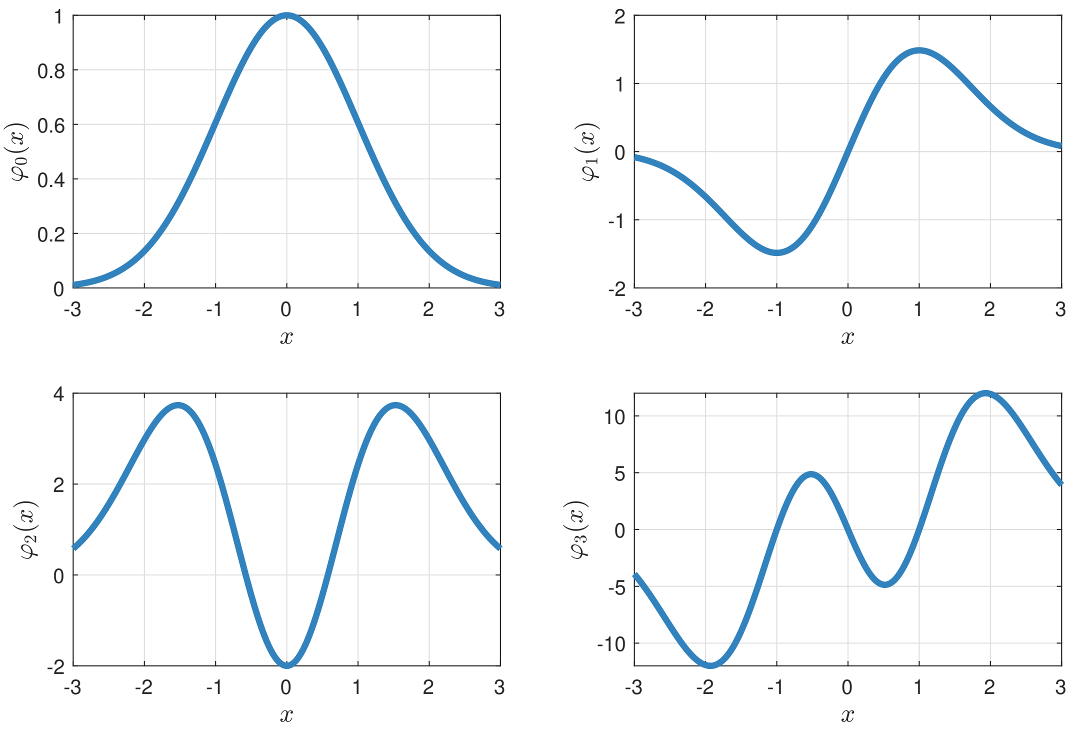

FIGURE 1.1: Plot of the first 4 eigenfunctions of the Gaussian kernel with $\eta^2 = \sigma^2 = 1$.

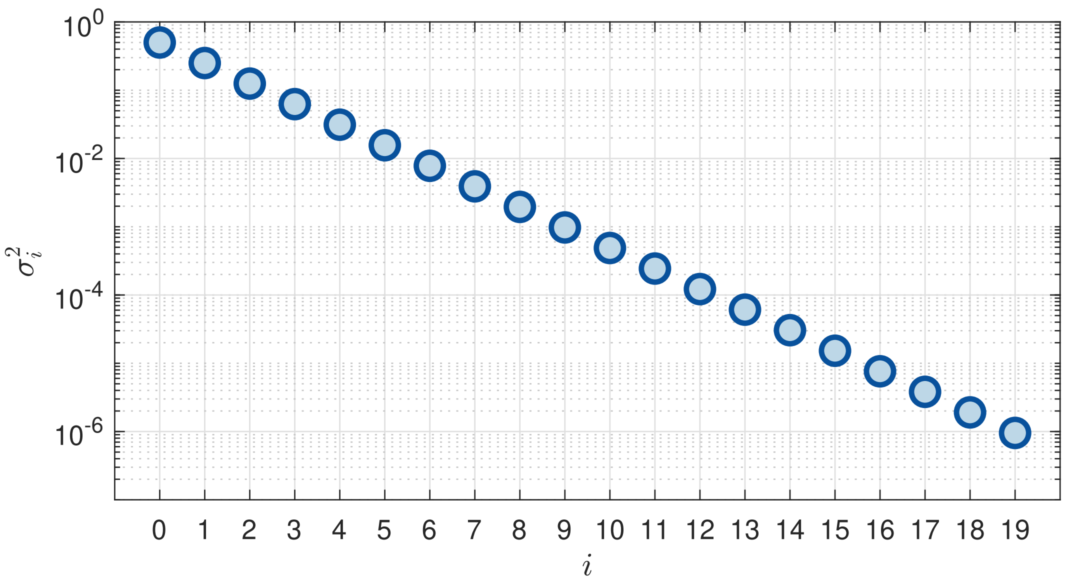

FIGURE 1.2: Plot of the first 20 eigenvalues of the Gaussian kernel with $\eta^2 = \sigma^2 = 1$.

$$(x, y) \rightarrow k_1(x, y) \cdot k_2(x, y) \tag{1.48}$$

is a valid kernel and it defines the space

$$\mathcal{H} = \{u \text{ s.t. } \exists u_1 \in \mathcal{H}_1, u_2 \in \mathcal{H}_2 \text{ s.t. } u(x) = u_1(x) \cdot u_2(x), \forall x \in \mathcal{X}\} \tag{1.49}$$

Remark 1.5. From Theorem 1.5, it is possible to see that if we stretch the kernel with a positive scalar, we obtain a new valid kernel.

These two theorems provide a way to combine different simple kernels in order to create a more complicated one. Consider the following examples.

Example 1.7: Space of linear affine functions

Consider the following kernel

$$k : \mathbb{R}^{d\times 1} \times \mathbb{R}^{d\times 1} \rightarrow \mathbb{R} \tag{1.50}$$

$$(\boldsymbol{x}, \boldsymbol{y}) = \boldsymbol{x}^\top \boldsymbol{y} + 1 \tag{1.51}$$

This kernel is the sum of a linear kernel and a constant kernel. Therefore, for Theorem 1.5, the associated space is

$$\mathcal{H} = \left\{u \text{ s.t. } \exists \boldsymbol{w} \in \mathbb{R}^{d\times 1}, c \in \mathbb{R} \text{ s.t. } u(\boldsymbol{x}) = \boldsymbol{w}^\top \boldsymbol{x} + c, \forall \boldsymbol{x} \in \mathcal{X}\right\} \tag{1.52}$$

because, as shown in Example 1.2, the linear kernel defines the space of linear functions and the constant kernel, as shown in Example 1.1, contains all the constant functions. Therefore, this space contains all the linear affine functions.

Example 1.8: Space of polynomial functions

Consider the following kernel

$$k : \mathbb{R}^{d\times 1} \times \mathbb{R}^{d\times 1} \rightarrow \mathbb{R} \tag{1.53}$$

$$(\boldsymbol{x}, \boldsymbol{y}) = \left(\boldsymbol{x}^\top \boldsymbol{y} + 1\right)^d \tag{1.54}$$

where $d \in \mathbb{N}$.
This kernel is the product of d valid kernel and therefore, for Theorem 1.6, is also a valid kernel. Furthermore, a product of d affine functions is a polynomial with degree d. For this reason, the RKHS associated with this kernel contains all the polynomial of degree d.

1.2 Non-linear regression using RKHS

In the previous section, a formal mathematical introduction on the RKHS and their properties is presented. This section delves into the application of these special spaces in the statistical learning theory. In particular, the focus will be on the regression because it is more useful for system identification.

1.2.1 Intuition and kernel trick

To understand how the RKHS can be useful for learning, consider the following linear regression model

$$\begin{aligned} y_i &= \breve{g}(\boldsymbol{x}_i) + e_i \\ &= \boldsymbol{\gamma}(\boldsymbol{x}_i)^\top \breve{\boldsymbol{\vartheta}} + e_i \qquad\qquad i = 1, \dots, n \\ &= \sum_{j=1}^{n_\vartheta} \breve{\vartheta}_j \gamma_j(\boldsymbol{x}_i) + e_i \end{aligned} \tag{1.55}$$

where

- $n \in \mathbb{N}$ is length of the dataset;
- $n_\vartheta \in \mathbb{N}$ is the number of parameters;
- $\boldsymbol{x}_i \in \mathcal{X} \subseteq \mathbb{R}^{n_x \times 1}$, with $i = 1, \dots, n$, are the regressors;
- $y_i \in \mathbb{R}$, with $i = 1, \dots, n$, are the measured outputs;
- $e_i \in \mathbb{R}$, with $i = 1, \dots, n$, are IID gaussian distributed noises, i.e. $e_i \sim \mathcal{N}(0, \eta^2)$;
- $\breve{\boldsymbol{\vartheta}} = \left[\breve{\vartheta}_1, \dots, \breve{\vartheta}_{n_\vartheta}\right]^\top \in \mathbb{R}^{n_\vartheta \times 1}$ is a vector composed by the unknown parameters;
- $\boldsymbol{\gamma}(\boldsymbol{x}) = [\gamma_1(\boldsymbol{x}), \dots, \gamma_{n_\vartheta}(\boldsymbol{x})]^\top \in \mathbb{R}^{n_\vartheta \times 1}$ is the function, often called *feature map*, that maps the regressors in the features space;

In this regression model, the aim is to find the function $g : \mathbb{R}^{n_x \times 1} \to \mathbb{R}$ that is inside the hypothesis set

$$\mathcal{H} = \text{span}\{\gamma_1, \dots, \gamma_{n_\vartheta}\} \tag{1.56}$$

$$= \left\{u \text{ s.t. } \exists \boldsymbol{\vartheta} \in \mathbb{R}^{n_\vartheta \times 1} \text{ s.t. } u(\boldsymbol{x}) = \boldsymbol{\gamma}(\boldsymbol{x})^\top \boldsymbol{\vartheta}, \forall \boldsymbol{x} \in \mathcal{X}\right\} \tag{1.57}$$

that better explains the phenomena at hand.

Following the reasoning behind the standard ridge regression [17, 44], we can estimate the parameters $\breve{\boldsymbol{\vartheta}}$ by minimizing the cost function

$$\hat{\boldsymbol{\vartheta}} = \underset{\boldsymbol{\vartheta} \in \mathbb{R}^{n_\vartheta \times 1}}{\arg\min} \{J(\boldsymbol{\vartheta})\} \tag{1.58}$$

$$J(\boldsymbol{\vartheta}) = \sum_{i=1}^{n} \left(y_i - \sum_{j=1}^{n_\vartheta} \vartheta_j \gamma_j(\boldsymbol{x}_i)\right)^2 + \tau \sum_{j=1}^{n_\vartheta} \vartheta_j^2 \tag{1.59}$$

where $\tau \in \mathbb{R}_+$ is the ridge regularization strength. It is well known that the minimizer of this cost function can be computed analytically [17, 44], in particular

$$\hat{\boldsymbol{\vartheta}} = \left(\boldsymbol{\Gamma}\boldsymbol{\Gamma}^\top + \tau \boldsymbol{I}_{n_\vartheta}\right)^{-1} \boldsymbol{\Gamma} \boldsymbol{y}^\top \tag{1.60}$$

where

$$\boldsymbol{\Gamma} = \begin{bmatrix} \boldsymbol{\gamma}(\boldsymbol{x}_1) & \cdots & \boldsymbol{\gamma}(\boldsymbol{x}_n) \end{bmatrix} \in \mathbb{R}^{n_\vartheta \times n} \tag{1.61}$$

$$\boldsymbol{y} = \begin{bmatrix} y_1 & \cdots & y_n \end{bmatrix} \in \mathbb{R}^{1 \times n} \tag{1.62}$$

In the end, the estimated function evaluated on a test regressor $\boldsymbol{x}^* \in \mathcal{X}$ is

$$\hat{y}^* = \boldsymbol{\gamma}\left(\boldsymbol{x}^*\right)^\top \hat{\boldsymbol{\vartheta}} \tag{1.63}$$

$$= \boldsymbol{\gamma}\left(\boldsymbol{x}^*\right)^\top \left(\boldsymbol{\Gamma}\boldsymbol{\Gamma}^\top + \tau \boldsymbol{I}_{n_\vartheta}\right)^{-1} \boldsymbol{\Gamma}\boldsymbol{y}^\top \tag{1.64}$$

This classical formulation is called *primal perspective*, but it is possible to solve this optimization problem in another way that is called *dual perspective*. This formulation is obtained by noting that the optimization problem (1.58) can be written as

$$J\left(\boldsymbol{e}, \boldsymbol{\vartheta}\right) = \sum_{i=1}^{n} e_i^2 + \tau \sum_{j=1}^{n_\vartheta} \vartheta_j^2 \tag{1.65}$$

$$\text{s.t } y_i = \sum_{j=1}^{n_\vartheta} \vartheta_j \gamma_j\left(\boldsymbol{x}_i\right) + e_i \qquad i = 1, \dots, n \tag{1.66}$$

where $\boldsymbol{e} = \left[e_1, \dots, e_n\right] \in \mathbb{R}^{1\times n}$. This is a constrained optimization problem that can be solved using the Lagrange multipliers [16]. In particular, we obtain the new cost function

$$J^*\left(\boldsymbol{e}, \boldsymbol{\vartheta}, \boldsymbol{\alpha}\right) = \sum_{i=1}^{n} e_i^2 + \tau \sum_{j=1}^{n_\vartheta} \vartheta_j^2 + \sum_{i=1}^{n} \alpha_i \left(y_i - \sum_{j=1}^{n_\vartheta} \vartheta_j \gamma_j\left(\boldsymbol{x}_i\right) - e_i\right) \tag{1.67}$$

where $\boldsymbol{\alpha} = \left[\alpha_1, \dots, \alpha_n\right]^\top \in \mathbb{R}^{n\times 1}$ are the Lagrange multipliers. Now, the optimality conditions are

$$\begin{cases} \dfrac{\partial}{\partial e_h} J^*\left(\boldsymbol{e}, \boldsymbol{\vartheta}, \boldsymbol{\alpha}\right) = 0 & h = 1, \dots, n \\ \dfrac{\partial}{\partial \alpha_h} J^*\left(\boldsymbol{e}, \boldsymbol{\vartheta}, \boldsymbol{\alpha}\right) = 0 & h = 1, \dots, n \\ \dfrac{\partial}{\partial \vartheta_h} J^*\left(\boldsymbol{e}, \boldsymbol{\vartheta}, \boldsymbol{\alpha}\right) = 0 & h = 1, \dots, n_\vartheta \end{cases} \tag{1.68}$$

by solving the partial derivatives, we obtain:

$$\begin{cases} \dfrac{\alpha_h}{2} = e_h & h = 1, \dots, n \\ y_h = \sum_{j=1}^{n_\vartheta} \vartheta_j \gamma_j\left(\boldsymbol{x}_h\right) + e_h & h = 1, \dots, n \\ \vartheta_h = \dfrac{1}{2\tau} \sum_{i=1}^{n} \alpha_i \gamma_h\left(\boldsymbol{x}_i\right) & h = 1, \dots, n_\vartheta \end{cases} \tag{1.69}$$

Now, it is possible to rewrite these three equations with only the Lagrange multipliers as variables. In particular, we can take the second equation and substitute e_h with the definition of the first equation and ϑ_j with the definition of the third equation.

$$y_i = \sum_{j=1}^{n_\vartheta} \left(\frac{1}{2\tau} \sum_{h=1}^{n} \alpha_h \gamma_j\left(\boldsymbol{x}_h\right)\right) \gamma_j\left(\boldsymbol{x}_i\right) + \frac{\alpha_i}{2} \qquad i = 1, \dots, n \tag{1.70}$$

now, with some mathematical steps, we can write

$$2y_i = \frac{1}{\tau} \sum_{h=1}^{n} \alpha_h \sum_{j=1}^{n_\vartheta} \gamma_j\left(\boldsymbol{x}_h\right) \gamma_j\left(\boldsymbol{x}_i\right) + \alpha_i \qquad i = 1, \dots, n. \tag{1.71}$$

This is a linear system with n equations and n unknown variables (the Lagrange multipliers). In matrix form, this system can be written as

$$\left(\boldsymbol{\Gamma}^\top\boldsymbol{\Gamma} + \tau\boldsymbol{I}_n\right)\hat{\boldsymbol{\alpha}} = 2\tau\boldsymbol{y}^\top \tag{1.72}$$

$$\hat{\boldsymbol{\alpha}} = 2\tau\left(\boldsymbol{\Gamma}^\top\boldsymbol{\Gamma} + \tau\boldsymbol{I}_n\right)^{-1}\boldsymbol{y}^\top \tag{1.73}$$

from $\boldsymbol{\alpha}$ we can compute the coefficients $\hat{\boldsymbol{\vartheta}}$ by using the third equation of (1.69). Therefore

$$\hat{\boldsymbol{\vartheta}} = \frac{1}{2\tau}\boldsymbol{\Gamma}\hat{\boldsymbol{\alpha}} \tag{1.74}$$

$$= \frac{1}{\cancel{2\tau}}\boldsymbol{\Gamma}\cdot\cancel{2\tau}\left(\boldsymbol{\Gamma}^\top\boldsymbol{\Gamma} + \tau\boldsymbol{I}_n\right)^{-1}\boldsymbol{y}^\top \tag{1.75}$$

$$= \boldsymbol{\Gamma}\left(\boldsymbol{\Gamma}^\top\boldsymbol{\Gamma} + \tau\boldsymbol{I}_n\right)^{-1}\boldsymbol{y}^\top \tag{1.76}$$

It can be shown that the parameters obtained in this way are equals to the one obtained with the primal perspective. Therefore, the evaluation of the estimated function on a test regressor $\boldsymbol{x}^* \in \mathcal{X}$

$$\hat{y}^* = \boldsymbol{\gamma}\left(\boldsymbol{x}^*\right)^\top\hat{\boldsymbol{\vartheta}} \tag{1.77}$$

$$= \boldsymbol{\gamma}\left(\boldsymbol{x}^*\right)^\top\boldsymbol{\Gamma}\left(\boldsymbol{\Gamma}^\top\boldsymbol{\Gamma} + \tau\boldsymbol{I}_n\right)^{-1}\boldsymbol{y}^\top \tag{1.78}$$

Remark 1.6. In the dual perspective, the number of unknown parameters to compute is equal to n, while in the primal formulation the number of unknown is n_ϑ. Therefore, when $n \geq n_\vartheta$ using the primal formulation is more convenient, vice versa when $n < n_\vartheta$ the dual formulation decreases the dimension of the linear system to solve.

Now, we can see that the (i, j) element of the matrix $\boldsymbol{K} = \boldsymbol{\Gamma}^\top\boldsymbol{\Gamma} \in \mathbb{R}^{n\times n}$ is

$$\boldsymbol{K}_{i,j} = \sum_{h=1}^{n_\vartheta}\gamma_h\left(\boldsymbol{x}_i\right)\gamma_h\left(\boldsymbol{x}_j\right) \tag{1.79}$$

$$= \boldsymbol{\gamma}\left(\boldsymbol{x}_i\right)^\top\boldsymbol{\gamma}\left(\boldsymbol{x}_j\right) \tag{1.80}$$

$$= k\left(\boldsymbol{x}_i, \boldsymbol{x}_j\right) \tag{1.81}$$

for construction, the function $k : \mathcal{X}\times\mathcal{X}\to\mathbb{R}$ is symmetric and positive semi-definite. For this reason, k is a valid kernel and therefore it defines a RKHS $\mathcal{H}$. Since we known that

$$k\left(\boldsymbol{x}_i, \boldsymbol{x}_j\right) = \sum_{h=1}^{n_\vartheta}\gamma_h\left(\boldsymbol{x}_i\right)\gamma_h\left(\boldsymbol{x}_j\right), \tag{1.82}$$

then this kernel is degenerate and it has dimension n_ϑ.

Remark 1.7. It is possible to note that, to compute $\hat{y}^*$, it is only necessary to know how to compute the kernel k. In particular, we can write

$$\hat{y}^* = \boldsymbol{\gamma}\left(\boldsymbol{x}^*\right)^\top\boldsymbol{\Gamma}\left(\boldsymbol{\Gamma}^\top\boldsymbol{\Gamma} + \tau\boldsymbol{I}_n\right)^{-1}\boldsymbol{y}^\top \tag{1.83}$$

$$= \boldsymbol{k}^*\left(\boldsymbol{x}^*\right)^\top\left(\boldsymbol{K} + \tau\boldsymbol{I}_n\right)^{-1}\boldsymbol{y}^\top \tag{1.84}$$

where

$$\boldsymbol{k}^*\left(\boldsymbol{x}^*\right)^\top = \boldsymbol{\gamma}\left(\boldsymbol{x}^*\right)^\top \boldsymbol{\Gamma} \tag{1.85}$$

$$= \left[\begin{array}{ccc} \boldsymbol{\gamma}\left(\boldsymbol{x}^*\right)^\top \boldsymbol{\gamma}\left(\boldsymbol{x}_1\right) & \cdots & \boldsymbol{\gamma}\left(\boldsymbol{x}^*\right)^\top \boldsymbol{\gamma}\left(\boldsymbol{x}_n\right) \end{array}\right] \tag{1.86}$$

$$= \left[\begin{array}{ccc} k\left(\boldsymbol{x}^*, \boldsymbol{x}_1\right) & \cdots & k\left(\boldsymbol{x}^*, \boldsymbol{x}_n\right) \end{array}\right] \in \mathbb{R}^{1\times n} \tag{1.87}$$

The kernel trick consist of substituting this degenerate kernel with a non-degenerate one. This is possible because we do not actually need to compute the features map function, as shown in Remark 1.7.

Thanks to the Mercer theorem (see Theorem 1.4), we known that

$$k\left(\boldsymbol{x}_i, \boldsymbol{x}_j\right) = \sum_{h=1}^{\infty} \psi_h\left(\boldsymbol{x}_i\right) \psi_h\left(\boldsymbol{x}_j\right) \tag{1.88}$$

and therefore, the sum (1.82) is expanded to, potentially, infinity. This results in a linear regression with an infinite amount of features where the hypothesis space (1.56), defined as the span of the features, is the RKHS associated with the kernel.

For this reason, using the kernel trick allows extending the theory behind linear regression to non-linear models where the hypothesis space is given by an RKHS that, as shown in Section 1.1, can contain a large variety of non-linear functions. Furthermore, it is possible to tune the kernel in such a way that the hypothesis space contains the functions that are more suitable in the application at hand. Therefore, it provides a straightforward way to incorporate prior knowledge in the regression algorithm.

1.2.2 Tikhonov regularization

The kernel trick is an intuitive way to understand how the RKHS can be used in learning theory. This result can be formalized more directly, called *Tikhonov regularization* [111, 131], that provides additional insight on the behavior of the method.

Consider the dataset

$$\mathcal{D} = \left\{\left(\boldsymbol{x}_i, y_i\right) | 1 \leq i \leq n\right\}, \tag{1.89}$$

sampled from the generic probabilistic model

$$y_i = \breve{g}\left(\boldsymbol{x}_i\right) + e_i \tag{1.90}$$

where e_i are IID noises with variance β^2, $\boldsymbol{x}_i \in \mathcal{X} \subseteq \mathbb{R}^{n_x \times 1}$ are the regressors, $y_i \in \mathbb{R}$ denote the measurements and $\breve{g}$ is an unknown function. To make the notation more compact, we define the vectors:

$$\boldsymbol{y} = \left[\begin{array}{ccc} y_1 & \cdots & y_n \end{array}\right] \in \mathbb{R}^{1\times n}, \tag{1.91}$$

$$\breve{\boldsymbol{g}} = \left[\begin{array}{ccc} \breve{g}\left(\boldsymbol{x}_1\right) & \cdots & \breve{g}\left(\boldsymbol{x}_n\right) \end{array}\right] \in \mathbb{R}^{1\times n}, \tag{1.92}$$

$$\boldsymbol{e} = \left[\begin{array}{ccc} e_1 & \cdots & e_n \end{array}\right] \in \mathbb{R}^{1\times n} \tag{1.93}$$

and rewrite (1.90) as:

$$\boldsymbol{y} = \breve{\boldsymbol{g}} + \boldsymbol{e}, \tag{1.94}$$

In order to estimate the function $\breve{g}$, instead of an hypothesis space composed by a span of a finite number of features as in (1.56), we suppose that the function $\breve{g}$ belongs to an RKHS $\mathcal{H}$ with kernel k. Using the normal least square method, we obtain the estimation

$$\hat{g} = \underset{g\in\mathcal{H}}{\arg\min} \left\{ \sum_{i=1}^{n} \left(y_i - g\left(\boldsymbol{x}_i\right)\right)^2 \right\} \tag{1.95}$$

$$= \underset{g\in\mathcal{H}}{\arg\min} \left\{ \|\boldsymbol{y} - \boldsymbol{g}\|_2^2 \right\} \tag{1.96}$$

In practice, this cost function cannot be used because the hypothesis space $\mathcal{H}$ can be very large, potentially infinite dimensional. This cause the estimate $\hat{g}$ to heavily overfit the training dataset (1.89). The natural solution of this problem is the regularization, but, in this case, we have to impose a penalty on the function itself because the function is not parametrized with a finite number of parameters.

Since $\mathcal{H}$ is an Hilbert space, it is equipped with a norm whose meaning depends on the type of space. For the Mercer theorem (see Theorem 1.4), we know that this norm is larger for functions that are similar to the eigenfunctions associated a with small eigenvalue. Therefore, it is possible to tune the kernel to obtain a norm that is large in correspondence of functions with not-wanted behavior. For example, in Example 1.6 the Gaussian kernel defines a norm that is larger for functions with significant components at higher frequencies.

For this reason, it is possible to regularize the cost function with the norm of the RKHS.

$$\hat{g} = \underset{g\in\mathcal{H}}{\arg\min} \left\{ \|\boldsymbol{y} - \boldsymbol{g}\|_2^2 + \tau \left\|g\right\|_{\mathcal{H}}^2 \right\} \tag{1.97}$$

This is, potentially, an infinite dimensional optimization problem that is not straightforward to solve. However, it is known that the solution of this optimization problem exists and it is unique [130].

Using the properties of the RKHS $\mathcal{H}$, it is possible to prove the following important theorem.

Theorem 1.7 (Representer theorem [40, 111])**.** *Let $\hat{g}$ be as in* (1.97)*. Then, there exists $\boldsymbol{c} \in \mathbb{R}^{n\times 1}$ such that*

$$\hat{g} = \sum_{i=1}^{n} c_i r_{\boldsymbol{x}_i} \tag{1.98}$$

where $r_{\boldsymbol{x}} \in \mathcal{H}$ is the representer of $\boldsymbol{x} \in \mathcal{X}$, as defined in Definition 1.2.

Thanks to this theorem, the optimization problem (1.97) boils down to a finite dimensional optimization problem where we need to estimate only the vector $\hat{\boldsymbol{c}} \in \mathbb{R}^{n\times 1}$.

Remark 1.8. The representer theorem utilizes the properties of the RKHS to find a finite dimensional subspace of $\mathcal{H}$ that contains the function that minimize the cost function. Furthermore, it provides a base of this subspace $\{r_{\boldsymbol{x}_1}, \dots, r_{\boldsymbol{x}_n}\}$.

Thank to this theorem, the norm becomes

$$\|\boldsymbol{g}\|_{\mathcal{H}}^2 = \left\| \sum_{i=1}^{n} c_i r_{\boldsymbol{x}_i} \right\|_{\mathcal{H}}^2 \tag{1.99}$$

$$= \left\langle \sum_{i=1}^{n} c_i r_{\boldsymbol{x}_i}, \sum_{j=1}^{n} c_j r_{\boldsymbol{x}_j} \right\rangle_{\mathcal{H}} \tag{1.100}$$

$$= \sum_{i=1}^{n}\sum_{j=1}^{n} c_i c_j \left\langle r_{\boldsymbol{x}_i}, r_{\boldsymbol{x}_j} \right\rangle_{\mathcal{H}} \tag{1.101}$$

$$= \sum_{i=1}^{n}\sum_{j=1}^{n} c_i c_j k\left(\boldsymbol{x}_i, \boldsymbol{x}_j\right) \tag{1.102}$$

$$= \boldsymbol{c}^\top \boldsymbol{K} \boldsymbol{c} \tag{1.103}$$

where $\boldsymbol{K} \in \mathbb{R}^{n\times n}$ is a matrix whose element (i,j) is $k\left(\boldsymbol{x}_i, \boldsymbol{x}_j\right)$ and it is, usually, called *Kernel matrix*. Since

$$\boldsymbol{g} = \left[\begin{array}{ccc} g\left(x_1\right) & \cdots & g\left(x_n\right) \end{array}\right] \tag{1.104}$$

$$= \left[\begin{array}{ccc} \sum_{i=1}^{n} c_i r_{\boldsymbol{x}_i}\left(x_1\right) & \cdots & \sum_{i=1}^{n} c_i r_{\boldsymbol{x}_i}\left(x_n\right) \end{array}\right] \tag{1.105}$$

$$= \left[\begin{array}{ccc} \sum_{i=1}^{n} c_i k\left(\boldsymbol{x}_i, \boldsymbol{x}_1\right) & \cdots & \sum_{i=1}^{n} c_i k\left(\boldsymbol{x}_i, \boldsymbol{x}_1\right) \end{array}\right] \tag{1.106}$$

$$= \left[\begin{array}{ccc} \boldsymbol{c}^\top \boldsymbol{K} \boldsymbol{e}_{n,1} & \cdots & \boldsymbol{c}^\top \boldsymbol{K} \boldsymbol{e}_{n,n} \end{array}\right] \tag{1.107}$$

$$= \boldsymbol{c}^\top \boldsymbol{K} \left[\begin{array}{ccc} \boldsymbol{e}_{n,1} & \cdots & \boldsymbol{e}_{n,n} \end{array}\right] \tag{1.108}$$

$$= \boldsymbol{c}^\top \boldsymbol{K} \boldsymbol{I}_n \tag{1.109}$$

$$= \boldsymbol{c}^\top \boldsymbol{K} \tag{1.110}$$

the loss term becomes

$$\|\boldsymbol{y} - \boldsymbol{g}\|_2^2 = \left\|\boldsymbol{y} - \boldsymbol{c}^\top \boldsymbol{K}\right\|_2^2 \tag{1.111}$$

therefore, the optimization problem becomes:

$$\hat{g} = \sum_{i=1}^{n} \hat{c}_i r_{\boldsymbol{x}_i} \tag{1.112}$$

$$\hat{\boldsymbol{c}} = \left[\begin{array}{ccc} c_1 & \cdots & c_n \end{array}\right]^\top = \underset{\boldsymbol{c}\in\mathbb{R}^{n\times 1}}{\arg\min} \left\{ \left\|\boldsymbol{y} - \boldsymbol{c}^\top \boldsymbol{K}\right\|_2^2 + \tau \boldsymbol{c}^\top \boldsymbol{K} \boldsymbol{c} \right\} \tag{1.113}$$

This is a normal quadratic optimization problem that can be treated analytically. In particular, it is straightforward to show that the coefficients vector $\hat{\boldsymbol{c}} \in \mathbb{R}^{n\times 1}$ is the solution of the following linear system

$$\boldsymbol{K}\left(\boldsymbol{K} + \tau \boldsymbol{I}_n\right) \hat{\boldsymbol{c}} = \boldsymbol{K} \boldsymbol{y}^\top \tag{1.114}$$

If the kernel is non-degenerate, then the matrix $\boldsymbol{K}$ is positive definite and therefore invertible. In this case, we can simplify the matrix $\boldsymbol{K}$ on both sides

$$\left(\boldsymbol{K} + \tau \boldsymbol{I}_n\right) \hat{\boldsymbol{c}} = \boldsymbol{y}^\top \tag{1.115}$$

$$\hat{\boldsymbol{c}} = \left(\boldsymbol{K} + \tau \boldsymbol{I}_n\right)^{-1} \boldsymbol{y}^\top \tag{1.116}$$

Then, given a test regressor $\boldsymbol{x}^*$, the output estimation is

$$\hat{y}^* = \hat{g}\left(\boldsymbol{x}^*\right) = \sum_{i=1}^{n} \hat{c}_i r_{\boldsymbol{x}_i}\left(\boldsymbol{x}^*\right) \tag{1.117}$$

$$= \sum_{i=1}^{n} \hat{c}_i \left\langle r_{\boldsymbol{x}_i}, r_{\boldsymbol{x}^*} \right\rangle_{\mathcal{H}} \tag{1.118}$$

$$= \sum_{i=1}^{n} \hat{c}_i k\left(\boldsymbol{x}_i, \boldsymbol{x}^*\right) \tag{1.119}$$

$$= \boldsymbol{k}^*\left(\boldsymbol{x}^*\right)^\top \hat{\boldsymbol{c}} \tag{1.120}$$

$$= \boldsymbol{k}^*\left(\boldsymbol{x}^*\right)^\top \left(\boldsymbol{K} + \tau \boldsymbol{I}_n\right)^{-1} \boldsymbol{y}^\top \tag{1.121}$$

Here, it is possible to note that this formulation is equivalent to the one obtained with the kernel trick, see equation (1.84).

However, this perspective provides more insight into the method. First of all, we know that the identified function is unique and it exists [130]. Additionally, the Mercer theorem (see Theorem 1.4) sheds some light on what the regularization achieves. In particular, we know that the Tikhonov regularizer penalizes functions that are similar to the eigenfunctions of the kernel that are associated with a small eigenvalue, as explained before.

1.3 Gaussian process regression

It is well known that the ridge regression has a Bayesian perspective where, instead of assuming that the parameters vector $\boldsymbol{\vartheta}$ is an element of a hypothesis set, it is assumed that $\boldsymbol{\vartheta}$ is a random variable with a certain distribution called *prior*. In particular, considering the following probabilistic model

$$p\left(\boldsymbol{y} \,|\boldsymbol{X}, \boldsymbol{\vartheta}\right) = \mathcal{N}\left(\boldsymbol{y}^\top \middle| \boldsymbol{\Gamma}^\top \boldsymbol{\vartheta}, \beta^2 \boldsymbol{I}_n\right) \qquad \text{Likelihood distribution} \tag{1.122}$$

$$p\left(\boldsymbol{\vartheta} \,|\boldsymbol{X}\right) = \mathcal{N}\left(\boldsymbol{\vartheta} \,\middle|\boldsymbol{0}_{n_\vartheta \times 1}, \eta^2 \boldsymbol{I}_{n_\vartheta}\right) \qquad \text{Prior distribution} \tag{1.123}$$

where $\boldsymbol{\Gamma} \in \mathbb{R}^{n_\vartheta \times n}$, $\boldsymbol{\vartheta} \in \mathbb{R}^{n_\vartheta \times 1}$, $\boldsymbol{y} \in \mathbb{R}^{1 \times n}$ are defined as in Section 1.2.1, β and η are strictly positive real numbers and

$$\boldsymbol{X} = \begin{bmatrix} \boldsymbol{x}_1 & \cdots & \boldsymbol{x}_n \end{bmatrix} \in \mathbb{R}^{n_x \times n}. \tag{1.124}$$

Using the conjugacy properties of the Normal distribution [17], it is straightforward to show that the posterior distribution is

$$p\left(\boldsymbol{\vartheta} \,|\boldsymbol{X}, \boldsymbol{y}\right) = \frac{p\left(\boldsymbol{y} \,|\boldsymbol{X}, \boldsymbol{\vartheta}\right) \cdot p\left(\boldsymbol{\vartheta} \,|\boldsymbol{X}\right)}{p\left(\boldsymbol{y} \,|\boldsymbol{X}\right)} \tag{1.125}$$

$$= \mathcal{N}\left(\boldsymbol{\vartheta} \middle| \hat{\boldsymbol{\vartheta}}, \boldsymbol{\Sigma}_{\vartheta|y}\right) \tag{1.126}$$

where

$$\hat{\boldsymbol{\vartheta}} = \left(\boldsymbol{\Gamma}\boldsymbol{\Gamma}^\top + \frac{\beta^2}{\eta^2} \boldsymbol{I}_{n_\vartheta}\right)^{-1} \boldsymbol{\Gamma} \boldsymbol{y}^\top \tag{1.127}$$

$$\boldsymbol{\Sigma}_{\vartheta|y} = \beta^2 \left(\boldsymbol{\Gamma}\boldsymbol{\Gamma}^\top + \frac{\beta^2}{\eta^2} \boldsymbol{I}_{n_\vartheta}\right)^{-1} \tag{1.128}$$

Additionally, given a test regressor $\boldsymbol{x}^*$, the estimated output is a random variable whose distribution is called *prediction distribution*. In particular, we have:

$$p\left(y^* \left| \boldsymbol{X}, \boldsymbol{y}, \boldsymbol{x}^* \right.\right) = \int p\left(y^* \left| \boldsymbol{x}^*, \boldsymbol{\vartheta} \right.\right) p\left(\boldsymbol{\vartheta} \left| \boldsymbol{X}, \boldsymbol{y} \right.\right) \, d\boldsymbol{\vartheta} \tag{1.129}$$

$$= \mathcal{N}\left(y^* \left| \hat{y}^*, \hat{\sigma}^2_{y^*} \right.\right) \tag{1.130}$$

where

$$\hat{y}^* = \boldsymbol{\gamma}\left(\boldsymbol{x}^*\right)^\top \hat{\boldsymbol{\vartheta}} \tag{1.131}$$

$$\hat{\sigma}^2_{y^*} = \beta^2 + \boldsymbol{\gamma}\left(\boldsymbol{x}^*\right)^\top \boldsymbol{\Sigma}_{\vartheta|y} \boldsymbol{\gamma}\left(\boldsymbol{x}^*\right) \tag{1.132}$$

Here, it is possible to see that the mean of the posterior correspond to the ridge regression estimation, see equation (1.60), with $\tau = \frac{\beta^2}{\eta^2}$. For this reason, this approach can be considered as a different way to reach the same method as the ridge regression. However, it provides new insight on the meaning of the parameter τ and it provides the variance of the estimation that can be used to define confidence interval on the estimation.

1.3.1 Gaussian process definition

In the previous sections, we saw that the kernel regression can be seen as an extension of the ridge regression to the infinite-dimensional case. Therefore, it is legit to ask if there exists a Bayesian perspective even for the kernel regression. The answer is positive and it is provided by the so-called *Gaussian process regression* [104].

In the kernel methods, the unknown is not a finite-dimensional vector, but a function. Therefore, it is necessary to use a statistical distribution of functions defined on the RKHS $\mathcal{H}$. This distribution is called *Gaussian process* and it is defined as follow.

Definition 1.5 (Gaussian process (GP))**.** *A function $u : \mathcal{X} \to \mathbb{R}$ is distributed according to a **Gaussian process** $\mathcal{GP}$ with mean $\rho : \mathcal{X} \to \mathbb{R}$ and covariance $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ if and only if for every finite subset $\overline{\mathcal{X}} = \{x_1, \cdots, x_n\}$ of $\mathcal{X}$, we have*

$$\begin{bmatrix} u\left(x_1\right) \\ \vdots \\ u\left(x_m\right) \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \rho\left(x_1\right) \\ \vdots \\ \rho\left(x_m\right) \end{bmatrix}, \begin{bmatrix} k\left(x_1, x_1\right) & \cdots & k\left(x_m, x_a\right) \\ \vdots & \ddots & \vdots \\ k\left(x_1, x_m\right) & \cdots & k\left(x_m, x_m\right) \end{bmatrix} \right) \tag{1.133}$$

and it is indicated as

$$u \sim \mathcal{GP}\left(\rho, k\right) \tag{1.134}$$

Remark 1.9. Since the variance of the normal distribution has to be symmetric and positive semi-definite matrix, the covariance function k has to a valid kernel. For this reason, the various kernel examples provided in Section 1.1 are all valid covariance functions for a Gaussian process.

Example 1.9: Gaussian process with Gaussian variance

To better understand the behavior of the functions sampled from a Gaussian process, consider the case where the covariance function is a Gaussian kernel.

$$k : \mathbb{R} \times \mathbb{R} \to \mathbb{R} \tag{1.135}$$

$$(x, y) \to e^{-\frac{(x-y)^2}{\sigma^2}} \tag{1.136}$$

then in Figure 1.3, it is possible to see some functions extracted by this distribution with three different value of σ and $\rho(x) = 0, \forall x \in \mathbb{R}$. Here, it is possible to note the effect of σ on the distribution. When σ is small, the function k tends to become similar to the Kronecker delta [88] and therefore the covariance matrix becomes similar to an identity matrix. This generates high varying functions because the samples are uncorrelated with each other. Vice versa, with larger σ the covariance matrix tends to become full of ones and therefore the samples are all heavily correlated with each other. For this reason, the sampled functions become smoother.
In Figure 1.4, it is possible to see the effect of different mean functions ρ on the distribution. In particular, we consider the following cases

$$\rho_1(x) = 10 \tag{1.137}$$

$$\rho_2(x) = 2\sin(3x) \tag{1.138}$$

$$\rho_3(x) = (x-2)^2 \tag{1.139}$$

From these plots, it is clear that the mean function moves the "baseline" of the sampled functions.

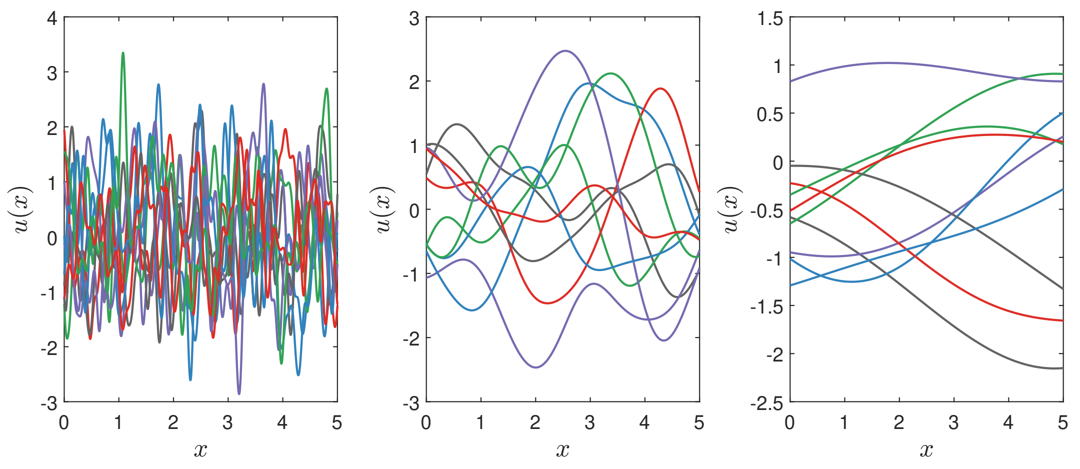

FIGURE 1.3: Plot of 10 functions taken from a Gaussian process with a Gaussian kernel and 0 mean with different values of σ. From left to right: $\sigma = 0.1$, $\sigma = 1$ and $\sigma = 5$.

Remark 1.10. From Theorem 1.5, we know that multiplying a kernel with a positive scalar allows defining a new valid kernel. If this kernel is used as a covariance function of a Gaussian process the effect is to increase the RMS of the sampled functions.

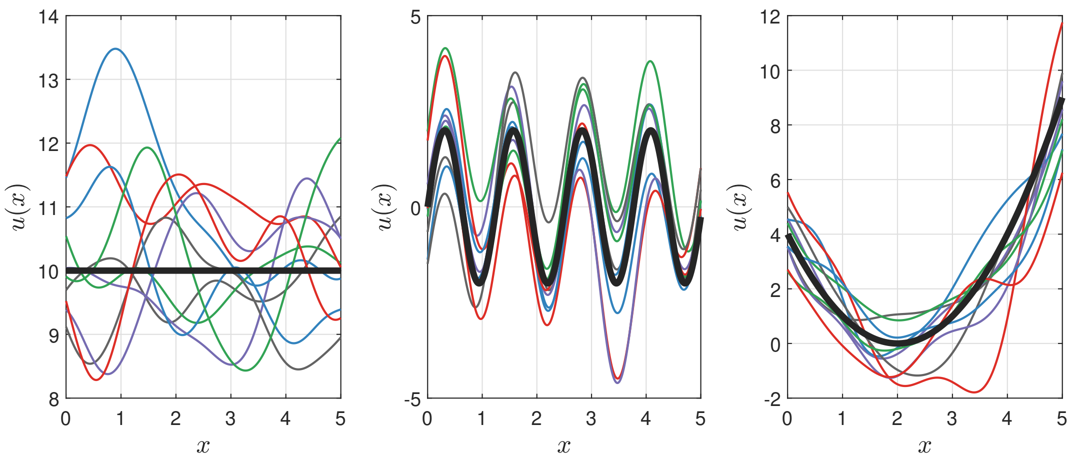

FIGURE 1.4: Plot of 10 functions taken from a Gaussian process (colored lines) with a Gaussian kernel with $\sigma = 1$ and different mean function ρ. The black line is the mean function. From left to right: $\rho = \rho_1$, $\rho = \rho_2$ and $\rho = \rho_3$.

1.3.2 Bayesian perspective of the Tikhonov regression

With the knowledge of a Gaussian process, it is possible to find the Bayesian perspective of the Tikhonov regularization presented in Section 1.2. Recalling the probabilistic model (1.90), we can define the likelihood distribution

$$p\left(\boldsymbol{y}\,|\boldsymbol{X}, g\right) = \mathcal{N}\left(\boldsymbol{y}^\top \middle| \boldsymbol{g}^\top, \beta^2 \boldsymbol{I}_n\right) \tag{1.140}$$

that can be complemented with a Gaussian process prior on the unknown function

$$g \sim \mathcal{GP}\left(0_{\mathcal{X}}, k\right) \tag{1.141}$$

where $0_{\mathcal{X}}$ is the function that returns 0 for every regressor inside $\mathcal{X}$ and $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is a valid kernel, i.e. its symmetric and positive semi-definite.

Now, let $\boldsymbol{x}^* \in \mathcal{X}$ be a new regressor, $g^* = g\left(\boldsymbol{x}^*\right) \in \mathbb{R}$ and y^* be the unknown measurement associated with the regressor $\boldsymbol{x}^*$. Assuming that this new data point y^* is sampled independently from the training dataset $\boldsymbol{y}$, we have

$$p\left(\boldsymbol{y}, y^*\,|\boldsymbol{g}, g^*, \boldsymbol{X}, \boldsymbol{x}^*\right) = \mathcal{N}\left(\begin{bmatrix} \boldsymbol{y}^\top \\ y^* \end{bmatrix} \middle\| \begin{bmatrix} \boldsymbol{g} \\ g^* \end{bmatrix}, \begin{bmatrix} \beta^2 \boldsymbol{I}_n & 0 \\ 0 & \beta^2 \end{bmatrix}\right) \tag{1.142}$$

Then we can note that, for the definition of Gaussian process, we have

$$p\left(\boldsymbol{g}, g^*\,|\boldsymbol{X}, \boldsymbol{x}^*\right) = \mathcal{N}\left(\begin{bmatrix} \boldsymbol{g}^\top \\ g^* \end{bmatrix} \middle\| \begin{bmatrix} \boldsymbol{0}_{n\times 1} \\ 0 \end{bmatrix}, \begin{bmatrix} \boldsymbol{K} & \boldsymbol{k}^*\left(\boldsymbol{x}^*\right) \\ \boldsymbol{k}^*\left(\boldsymbol{x}^*\right)^\top & k\left(\boldsymbol{x}^*, \boldsymbol{x}^*\right) \end{bmatrix}\right) \tag{1.143}$$

where $\boldsymbol{K} \in \mathbb{R}^{n\times n}$ is the kernel matrix and $\boldsymbol{k}^*\left(\boldsymbol{x}^*\right) = \left[k\left(\boldsymbol{x}^*, \boldsymbol{x}_1\right), \ldots, k\left(\boldsymbol{x}^*, \boldsymbol{x}_n\right)\right]^\top \in \mathbb{R}^{n\times 1}$.

Using these two expressions and the Normal distribution conjugacy properties [17], it is possible to compute the marginal likelihood

$$p\left(\boldsymbol{y}, y^* \left| \boldsymbol{X}, \boldsymbol{x}^* \right.\right) = \int p\left(\boldsymbol{y}, y^* \left| \boldsymbol{g}, g^*, \boldsymbol{X}, \boldsymbol{x}^* \right.\right) p\left(\boldsymbol{g}, g^* \left| \boldsymbol{X}, \boldsymbol{x}^* \right.\right) \, d\boldsymbol{g} dg^* \tag{1.144}$$

$$= \mathcal{N}\left(\begin{bmatrix} \boldsymbol{y}^\top \\ y^* \end{bmatrix} \middle| \begin{bmatrix} \boldsymbol{0}_{n\times 1} \\ 0 \end{bmatrix}, \begin{bmatrix} \boldsymbol{K} + \beta^2 \boldsymbol{I}_n & \boldsymbol{k}^*\left(\boldsymbol{x}^*\right) \\ \boldsymbol{k}^*\left(\boldsymbol{x}^*\right)^\top & k\left(\boldsymbol{x}^*, \boldsymbol{x}^*\right) + \beta^2 \end{bmatrix} \right) \tag{1.145}$$

In the end, we can compute the prediction distribution as

$$p\left(y^* \left| \boldsymbol{X}, \boldsymbol{x}^*, \boldsymbol{y} \right.\right) = \frac{p\left(\boldsymbol{y}, y^* \left| \boldsymbol{X}, \boldsymbol{x}^* \right.\right)}{p\left(\boldsymbol{y} \left| \boldsymbol{X}, \boldsymbol{x}^* \right.\right)} \tag{1.146}$$

$$= \mathcal{N}\left(y^* | \hat{y}^*, \hat{\sigma}^2_{y^*}\right) \tag{1.147}$$

where

$$\hat{y}^* = \boldsymbol{k}^*\left(\boldsymbol{x}^*\right)^\top \left(\boldsymbol{K} + \beta^2 \boldsymbol{I}_n\right)^{-1} \boldsymbol{y}^\top \tag{1.148}$$

$$\hat{\sigma}^2_{y^*} = \beta^2 + k\left(\boldsymbol{x}^*, \boldsymbol{x}^*\right) - \boldsymbol{k}^*\left(\boldsymbol{x}^*\right)^\top \left(\boldsymbol{K} + \beta^2 \boldsymbol{I}_n\right)^{-1} \boldsymbol{k}^*\left(\boldsymbol{x}^*\right) \tag{1.149}$$

Here, we can note that the expected value of the predictive distribution is equal to the one obtained using the Tikhonov regularization, see equation (1.121), or the kernel trick, see equation (1.77), where $\tau = \beta^2$.

This new perspective on the method provides two new important insight

- The regularization strength τ corresponds to the measurement noise variance. This is intuitive because larger noises correspond to less reliable data and therefore the regularization is more important.
- It provides the variance of the estimation and therefore a way to compute the confidence interval.

To better understand the usefulness of this new interpretation, let us consider the following example.

Example 1.10: Gaussian process application

Consider the following function

$$\breve{g} : \mathbb{R} \rightarrow \mathbb{R} \tag{1.150}$$

$$x \rightarrow \sin\left(2x\right) + 4x \cos\left(x\right) \tag{1.151}$$

and the following small noiseless dataset

$$\boldsymbol{x} = \begin{bmatrix} 5.96 & 7.37 & 3.62 & 5.60 & 9.04 \end{bmatrix} \in \mathbb{R}^{1\times 5} \tag{1.152}$$

$$\boldsymbol{y} = \breve{\boldsymbol{g}} \in \mathbb{R}^{1\times 5} \tag{1.153}$$

Now, we can impose a Gaussian process prior on the unknown function, $g \sim \mathcal{GP}(0, k)$, where k is

$$k(x, y) = e^{-(x-y)^2}. \tag{1.154}$$

The posterior, obtained as shown before, is shown in Figure 1.5. In this graph, as expected, we can see that the uncertainty of the estimation increases with the distance to the available regressors. Furthermore, since the dataset used is noiseless, in the proximity of the regressors the uncertainty is close to 0. If we add a small amount of noise ($\beta^2 = 0.05$), we obtain a similar result, as shown in Figure 1.6, where the major difference is the presence of some uncertainty even in the proximity of the regressors because there is some uncertainty due to the measurement noise.

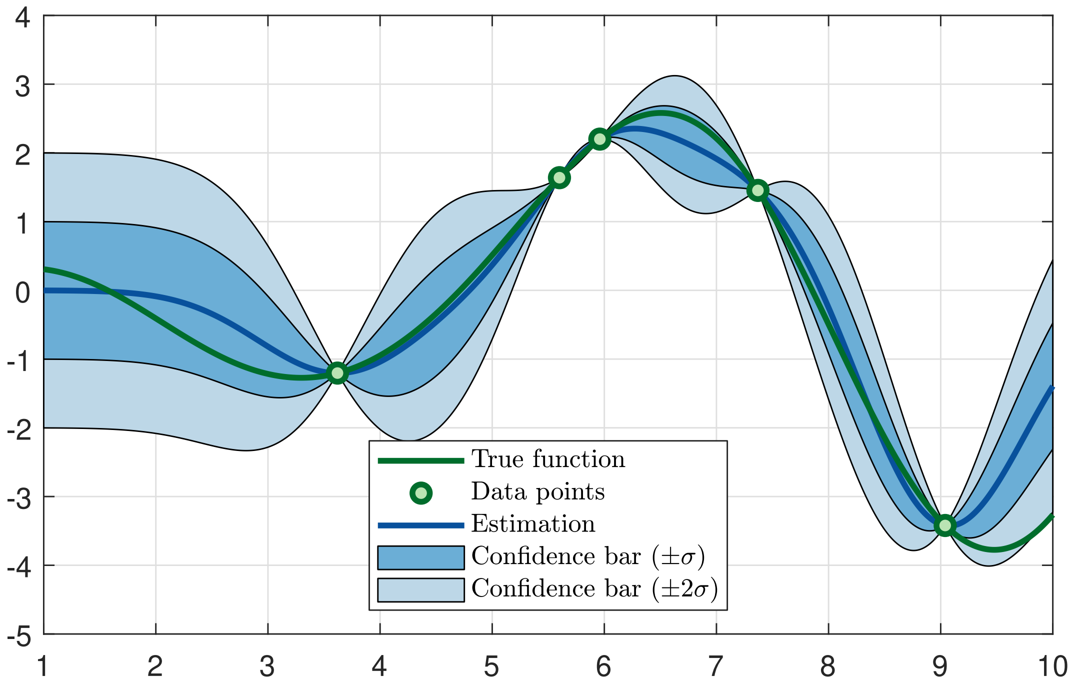

FIGURE 1.5: Plot of the posterior distribution (blue line and light-blue colored bands) of a Gaussian process regression without noise in comparison with the true function (green line). The points $(x, y) \in \mathcal{D}$ are shown in green circles.

1.4 Manifold regularization and semi-supervised learning

In the previous sections, the Tikhonov regularization is explored in three different perspectives (kernel trick, Tikhonov regularization, and Gaussian process regression), but it is possible to define other types of regularization as well. In particular, we can define other penalty terms based on some different unwanted behavior of the estimated function. This can be useful sometimes because in the standard Tikhonov regularization the kernel defines both the hypothesis space and the properties of the penalty term, as shown in the sections before. Sometimes, it is desirable to use a certain space $\mathcal{H}$, but not the squared norm $\|\cdot\|_{\mathcal{H}}^2$ as a penalty term.

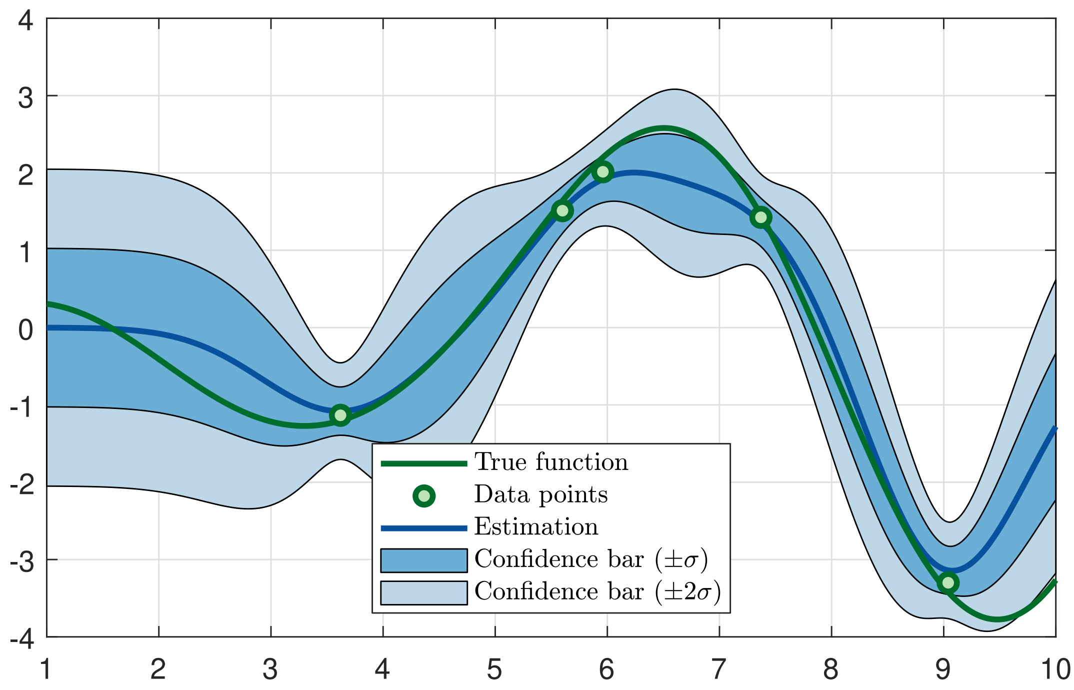

FIGURE 1.6: Plot of the posterior distribution (blue line and light-blue colored bands) of a Gaussian process regression with a small noise ($\beta^2 = 0.05$) in comparison with the true function (green line). The points $(x, y) \in \mathcal{D}$ are shown in green circles.

In this section, we will delve into one possible way to impose a different kind of penalization term that is based on the assumption that the regressor distribution holds some information on the system under exam.

1.4.1 Manifold regularization

In the standard learning paradigm the dataset is taken in the following way:

1. the regressors are sampled from a certain distribution:

$$x_i \sim p_x \qquad\qquad i = 1, \dots, n \tag{1.155}$$

2. the outputs measurements are sampled from a certain conditional distribution $p_{x|y}$

$$y_i \sim p_{y|x=x_i} \qquad\qquad i = 1, \dots, n \tag{1.156}$$

This dataset is then used to find a good approximation of the conditional distribution $p_{y|x=\boldsymbol{x}^*}$ where $\boldsymbol{x}^* \in \mathcal{X}$ is a generic regressor. For this reason, usually, the marginal distribution p_x is ignored because it does not contain any useful information about the conditional distribution $p_{y|x=\boldsymbol{x}^*}$. However, in some cases, this is not true and it is possible to extract some useful information from the marginal distribution p_x. To understand this concept, consider the following example.

Example 1.11: Example of the usefulness of the marginal distribution p_x

Consider a classification problem with a dataset that contains only one point for each of the two different classes. Without additional information, the most intuitive classifier is the linear classifier represented in the left graph of Figure 1.7.
However, in the right part, we can see that, considering the distribution of the regressors p_x, the most intuitive classifier is different. This is because, in the second plot, we consider the distance alongside the intrinsic geometry of the distribution. In other words, to find the classifier, we have used the assumption that the labels do not change along the same regions of the regressors distribution.

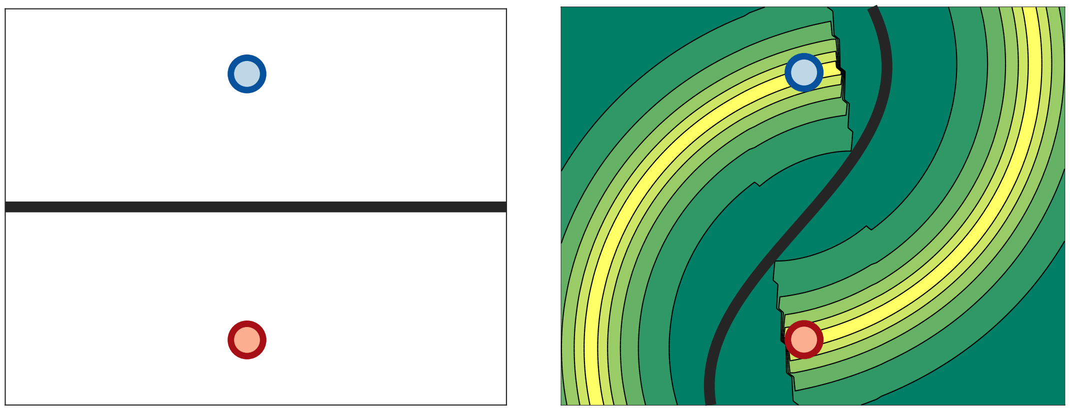

FIGURE 1.7: Plot of the most intuitive classifiers (black lines) with the two points of Example 1.11. In the left graph, the knowledge of the regressors distribution is unknown. Instead, in the right graph, the background color represents the pdf regressors distribution (yellow corresponds to a higher pdf values) and the most intuitive classifier is different.

In this example, we have shown that the knowledge about the regressors distribution p_x can be useful in order to learn the conditional. However, this was possible only because we have introduced an assumption on the conditional distribution $p_{y|x=\boldsymbol{x}^*}$. For this reason, it is necessary to consider the following assumption.

Assumption 1.1. *The conditional distribution $p_{y|x=\boldsymbol{x}^*}$ varies smoothly alongside $\boldsymbol{x}^*$ and its intrinsic geometry p_x.*

In the regression case treated in previous sections, the conditional distribution $p_{y|x=\boldsymbol{x}^*}$ is defined according to the simple model (see Equation (1.90))

$$y^* = \breve{g}\left(\boldsymbol{x}^*\right) + e^* \tag{1.157}$$

where $\breve{g} : \mathcal{X} \to \mathbb{R}$ is the function that we want to identify and e^* is a stochastic additive noise that is assumed to be independent from the regressor $\boldsymbol{x}^*$. Therefore, the variance of the conditional distribution $p_{y|x=\boldsymbol{x}^*}$ is equal to the variance of e^* that does not change with $\boldsymbol{x}^*$. However, this is not the case for the mean because it is equal to $\breve{g}\left(\boldsymbol{x}^*\right)$. For this reason, Assumption 1.1 can be seen as a condition on the unknown function $\breve{g}$. In particular, the assumption is respected if and only if the function $\breve{g}$ behaves smoothly alongside the intrinsic geometry of p_x.

For this reason, given a generic function $g : \mathcal{X} \to \mathbb{R}$, we can define a new term $\|g\|_{\mathcal{I}}^2$ that is higher when the function g is not smooth alongside p_x. This term, called *intrinsic regularizer*, can be added to the Tikhonov cost function (1.97) as a second penalty term to constrain the identified function $\hat{g}$ to respect, to some degree, Assumption 1.1. Therefore, the cost function becomes:

$$\hat{g} = \arg\min_{g \in \mathcal{H}} \left\{ \|\boldsymbol{y} - \boldsymbol{g}\|_2^2 + \tau \|g\|_{\mathcal{H}}^2 + \mu \|g\|_{\mathcal{I}}^2 \right\} \tag{1.158}$$

where $\mu \in \mathbb{R}_+$ is the strength of the new regularization term.

The choice of the intrinsic regularizer is not trivial, but there are some valid choices in the literature [11]. However, in this thesis, the focus will be on one of these possible choices. Let us assume that the support $\mathcal{X}$ of p_x is a compact submanifold, then a possible intrinsic regularizer is

$$\|g\|_{\mathcal{I}}^2 = \int_{\mathcal{X}} \|\nabla g(\boldsymbol{x})\|^2 p_x(\boldsymbol{x})\, d\boldsymbol{x} = \int_{\mathcal{X}} g(\boldsymbol{x})\, \Delta g(\boldsymbol{x})\, p_x(\boldsymbol{x})\, d\boldsymbol{x} \tag{1.159}$$

where ∇g is the gradient vector of g, Δ is the Laplace-Beltrami operator along the manifold $\mathcal{X}$ and, with a slight abuse of notation, $p_x(\boldsymbol{x})$ is the pdf of the regressors distribution evaluated on $\boldsymbol{x}$. This term penalizes functions with a high gradient with a weight that depends on the pdf p_x. Therefore, it penalizes functions that have high variation in the high probability regions of the regressors space, but it allows large swings of the function in the other regions. In other words, this intrinsic regularizer promotes functions that are smooth locally in the high-density regions of the regressors space even if they are not smooth globally in all the domain $\mathcal{X}$.

Remark 1.11. Since this type of intrinsic regularizer is defined on a manifold, this method is often called *manifold regularization.*

Unfortunately, this term is not computable in practice, because the distribution p_x is usually unknown. For this reason, it is necessary to find a way to approximate it by using the available regressors.

Remark 1.12. Since $\|g\|_{\mathcal{I}}^2$ depends only on p_x, only the regressors $x_i \sim p_x$ holds some useful information that can be exploited for the approximation. Therefore, for this purpose, the measurements of the output y_i are useless.

In order to approximate this regularizer, it is necessary to define the regressors graph.

Definition 1.6 (Regressors graph)**.** *Given a finite set of regressors $\mathcal{D} = \{x_1, \ldots, x_n\} \subset \mathcal{X}$, a* ***regressors graph*** *is a weighted graph with the following properties:*

- *each vertex of the graph is associated with a regressor of the set $\mathcal{D}$;*
- *the weight $w_{i,j} \geq 0$ of the edge between the vertices i and j represents the degree of proximity between x_i and x_j in the intrinsic geometry of p_x;*
- *if the edge between the vertices i and j is missing then x_i and x_j are considered not neighbors in the intrinsic geometry of p_x.*

Remark 1.13. How to create such graph is left for the next subsection (see Section 1.4.2).

Now, it can be shown [35, 58] that, given a dataset $\mathcal{D} = \{x_1, \ldots, x_n\}$ and a regressors graph with certain properties (see Section 1.4.2), the regularization term (1.159) can be approximated as

$$\|g\|_{\mathcal{I}}^2 \simeq \sum_{i=1}^{n} \sum_{j=1}^{n} w_{i,j} \left(g(\boldsymbol{x}_i) - g(\boldsymbol{x}_j)\right)^2 = \boldsymbol{g} \boldsymbol{L} \boldsymbol{g}^\top \tag{1.160}$$

where $\boldsymbol{g} = [g(x_i), \dots, g(x_j)]$ and $\boldsymbol{L} \in \mathbb{R}^{n \times n}$ is the Laplacian of a regressors graph i.e.

$$\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{W} \tag{1.161}$$

where $\boldsymbol{W} \in \mathbb{R}^{n \times n}$ is the weighted adjacency matrix of the regressors graph, i.e. the element (i, j) of $\boldsymbol{W}$ is the weight $w_{i,j}$, and $\boldsymbol{D} \in \mathbb{R}^{n \times n}$ is a diagonal matrix whose i-th diagonal element is

$$d_{i,i} = \sum_{j=1}^{n} w_{i,j} \tag{1.162}$$

Example 1.12: Importance of the right graph selection

Definition 1.6 does not specify how to construct such a graph and what "degree of proximity" actually means. These details are left for the Section 1.4.2, however, it is possible to show the effect of the graph selection on the intrinsic regularizer on a toy example.
Consider the following small regressors set

$$\mathcal{D} = \{-3, -2, -1, 0, 1, 2, 3\} \subseteq \mathbb{R} \tag{1.163}$$

and the two regressors graphs shown in Figure 1.8, where, for simplicity, the weights are 1 or 0. The first graph connects the regressors with their respective neighbors, while the second one connects each regressor with 0 and with the one with the opposite sign. The Laplacian matrix defined using the first graph is $\boldsymbol{L}_1 \in \mathbb{R}^{n \times n}$ and the one defined using the second graph is $\boldsymbol{L}_2 \in \mathbb{R}^{n \times n}$
Now, consider the two functions shown in Figure 1.9. The first function y_1 is smooth along the real axis, while the second one y_2 has an erratic behavior.
The value of the intrinsic regularizer of each function evaluated on each graph is

$$\text{First graph:} \qquad \boldsymbol{y}_1 \boldsymbol{L}_1 \boldsymbol{y}_1^\top = 70 \qquad \boldsymbol{y}_2 \boldsymbol{L}_1 \boldsymbol{y}_2^\top = 390 \tag{1.164}$$

$$\text{Second graph:} \qquad \boldsymbol{y}_1 \boldsymbol{L}_2 \boldsymbol{y}_1^\top = 588 \qquad \boldsymbol{y}_2 \boldsymbol{L}_2 \boldsymbol{y}_2^\top = 196 \tag{1.165}$$

where

$$\boldsymbol{y}_1 = [y_1(-3), y_1(-2), y_1(-1), y_1(0), y_1(1), y_1(2), y_1(3)] \in \mathbb{R}^{1 \times 7} \tag{1.166}$$

$$\boldsymbol{y}_2 = [y_2(-3), y_2(-2), y_2(-1), y_2(0), y_2(1), y_2(2), y_2(3)] \in \mathbb{R}^{1 \times 7} \tag{1.167}$$

According to the first graph, the smoothest function is $\boldsymbol{y}_1$, while according to the second on $\boldsymbol{y}_2$ is smoother. This is because the graph changes what regressor is considered close to a second regressor and therefore what function is considered smooth. For this reason, the selection of the right graph is really important for the definition of the regularizer.

The matrix $\boldsymbol{L}$ is the Laplacian of the regressors graph and, for its construction, is always symmetric and positive semi-definite [34]. In particular, the number of 0 eigenvalues is equal to the number of completely connected part of the regressors graph [34]. For this reason, there is always at least an eigenvalue equal to 0 and therefore the matrix $\boldsymbol{L}$ is always singular.

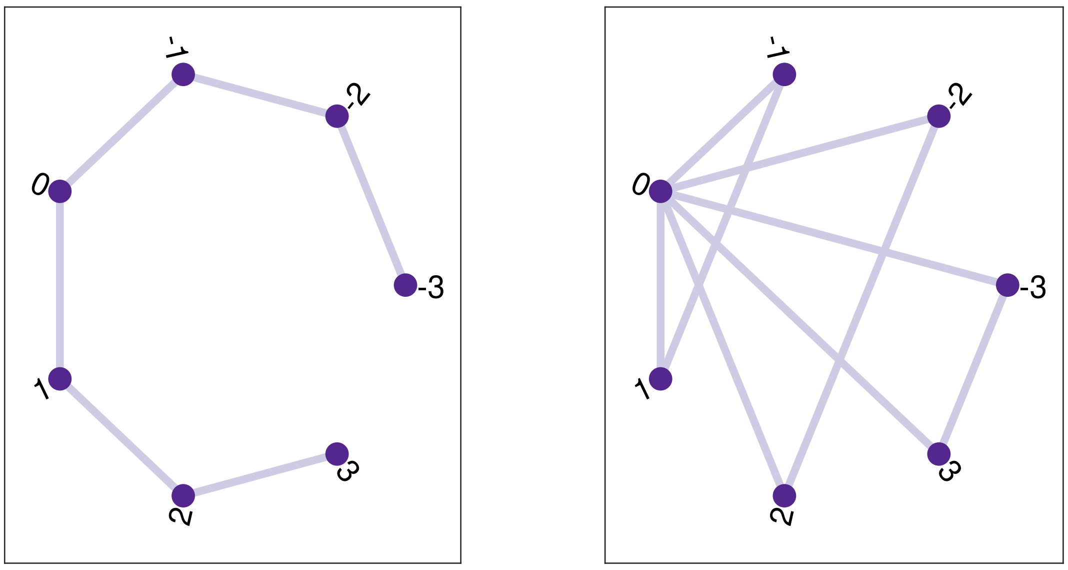

FIGURE 1.8: Plot of the two regressors graphs used in Example 1.12. If the edge is not drawn then its associated weight is 0, otherwise it is 1.

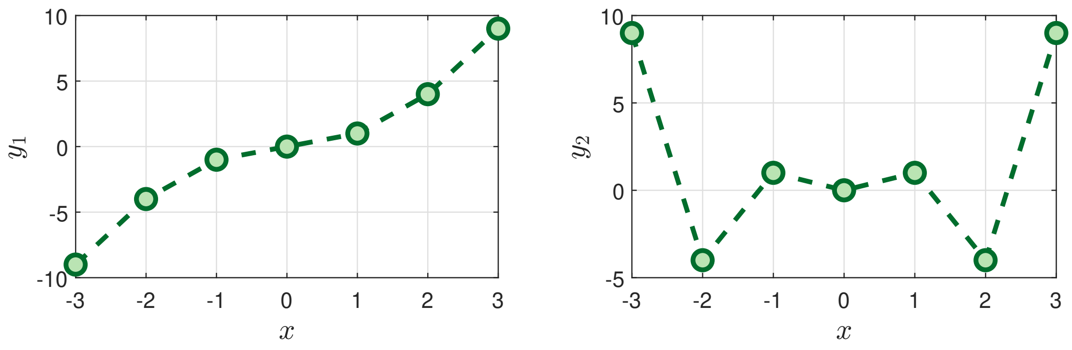

FIGURE 1.9: Plot of the two functions used in Example 1.12.

Using this approximation of the intrinsic regularizer, the cost function (1.158) becomes

$$\hat{g} = \arg\min_{g\in\mathcal{H}} \left\{ \|\boldsymbol{y} - \boldsymbol{g}\|_2^2 + \tau \|g\|_{\mathcal{H}}^2 + \mu \boldsymbol{g}\boldsymbol{L}\boldsymbol{g}^\top \right\} \tag{1.168}$$

For this cost function, the classic Representer theorem, reported as Theorem 1.7, cannot be applied. However, it is possible to show that the Representer theorem can be extended to this case [11, 40, 111].

Theorem 1.8 (Representer Theorem with Laplacian intrinsic regularizer)**.** *Let $\hat{g}$ be as in* (1.168)*. Then, there exists $\boldsymbol{c} \in \mathbb{R}^{n\times 1}$ such that*

$$\hat{g} = \sum_{i=1}^{n} c_i r_{\boldsymbol{x}_i} \tag{1.169}$$

where $r_{\boldsymbol{x}} \in \mathcal{H}$ is the representer of $\boldsymbol{x} \in \mathcal{X}$, as defined in Definition 1.2.

Therefore, the estimated function $\hat{g}$ can be written in the same way as for the classical Tikhonov regression explained in the previous sections. In these settings, since $\boldsymbol{g} = \boldsymbol{c}^\top \boldsymbol{K}$

(as shown in equation (1.110)) the intrinsic regularizer becomes

$$\boldsymbol{g}\boldsymbol{L}\boldsymbol{g}^\top = \boldsymbol{c}^\top \boldsymbol{K}\boldsymbol{L}\boldsymbol{K}\boldsymbol{c} \tag{1.170}$$

therefore the optimization problem can be rewritten as

$$\hat{g} = \sum_{i=1}^{n} \hat{c}_i r_{\boldsymbol{x}_i} \tag{1.171}$$

$$\hat{\boldsymbol{c}} = \begin{bmatrix} \hat{c}_1 & \cdots & \hat{c}_n \end{bmatrix}^\top = \underset{\boldsymbol{c}\in\mathbb{R}^{n\times 1}}{\arg\min} \left\{ \left\| \boldsymbol{y} - \boldsymbol{c}^\top \boldsymbol{K} \right\|_2^2 + \tau \boldsymbol{c}^\top \boldsymbol{K}\boldsymbol{c} + \mu \boldsymbol{c}^\top \boldsymbol{K}\boldsymbol{L}\boldsymbol{K}\boldsymbol{c} \right\} \tag{1.172}$$

This is a quadratic cost function whose minimizer can be found analytically. In particular, we obtain that the minimizer can be found by solving the linear system:

$$\boldsymbol{K}\left(\boldsymbol{K} + \tau \boldsymbol{I}_n + \mu \boldsymbol{L}\boldsymbol{K}\right)\hat{\boldsymbol{c}} = \boldsymbol{K}\boldsymbol{y}^\top \tag{1.173}$$

If the kernel is non-degenerate, then the matrix $\boldsymbol{K}$ is positive definite and therefore invertible. In this case, we can simplify the matrix $\boldsymbol{K}$ on both sides

$$\left(\boldsymbol{K} + \tau \boldsymbol{I}_n + \mu \boldsymbol{L}\boldsymbol{K}\right)\hat{\boldsymbol{c}} = \boldsymbol{y}^\top \tag{1.174}$$

$$\hat{\boldsymbol{c}} = \left(\boldsymbol{K} + \tau \boldsymbol{I}_n + \mu \boldsymbol{L}\boldsymbol{K}\right)^{-1} \boldsymbol{y}^\top \tag{1.175}$$

Remark 1.14. Even with the new regularization term, the classical Tikhonov regularizer cannot be removed because, for its construction, the linear system 1.175 is ill-posed unless τ is large [11]. This problem will be tackled in Section 3.4 as one of the new contributions of this thesis.

1.4.2 Graph selection methods

Let us now focus on the selection of the right regressors graph. In the literature, there are a lot of different ways to select this graph. In particular, there are two schools of thought:

- the first one employs complete graph with different edge weights [10, 13, 14, 36];
- the second one uses non-complete graph with all the edges with the same weight [15, 23, 49];

For both types of graph, there are theoretical results that show that the estimator $\boldsymbol{g}\boldsymbol{L}\boldsymbol{g}^\top$, explained in equation (1.160), converges to the desired regularizer (1.159) for a large amount of data.

In this section some basic methods to construct the graph are reported, however for more details refer to the literature, among others [10, 13, 14, 15, 23, 36, 49].

In [10, 11], they propose to use a complete graph with gaussian weights

$$w_{i,j} = e^{-\frac{\text{dist}(\boldsymbol{x}_i,\boldsymbol{x}_j)^2}{\sigma}} \tag{1.176}$$

where $\sigma \in \mathbb{R}_+$ and $\text{dist}\left(\boldsymbol{x}_i, \boldsymbol{x}_j\right)$ is a valid distance between two regressors. An evolution of this method is presented in [13, 14] where the weights are computed with more complex formulations.

In the second school of thought, the two basic techniques are: the fixed ε-ball and the k-NN. In the former method, a regressor $\boldsymbol{x_i}$ is connected to all the regressors such that

$$\operatorname{dist}\left(\boldsymbol{x}_i, \boldsymbol{x}_j\right)^2 \leq \varepsilon \tag{1.177}$$

where $\varepsilon \in \mathbb{R}_+$. Instead, the k-NN, that stands for k-Nearest Neighbors, connect the regressor $\boldsymbol{x_i}$ with the closest $k \in \mathbb{N} \setminus \{0\}$ other regressors in $\mathcal{D}$ according to some distance. The fixed ε-ball works best when the points are uniformly distributed in the manifold, while the k-NN tends to adapt to different points densities. In the literature, there are also some evolutions of these methods [15].

1.4.3 Semi-supervised identification

Classical statistical learning regression methods can be divided into two categories based on the type of data available

Supervised methods where the available dataset contains an output measurement for each regressor, i.e.

$$\mathcal{D} = \left\{\left(\boldsymbol{x}_i, y_i\right) | i = 1, \ldots, n\right\} \tag{1.178}$$

where n is the number of available data and y_i is the output measurement associated with the regressor $\boldsymbol{x}_i$.

Semi-supervised methods where not all the regressors in the available dataset have an associated measurement. In other words, there are two different datasets that can be used

$$\mathcal{D}_s = \left\{\left(\boldsymbol{x}_i, y_i\right) | i = 1, \ldots, n_s\right\} \qquad \text{Supervised dataset} \tag{1.179}$$

$$\mathcal{D}_u = \left\{\boldsymbol{x}_i | i = n_s + 1, \ldots, n = n_s + n_u\right\} \qquad \text{Unsupervised dataset} \tag{1.180}$$

the first one is a normal dataset that contains the regressors and their associated output measurements, while the second one contains only the regressors.

Until now, in this documents, the focus was on the supervised regression using kernel methods, however, in this section, a brief introduction on how to use kernel methods to do semi-supervised regression is presented. The intuition comes from Remark 1.12 where it is explained that only the regressors are needed to approximate the intrinsic regularizer. For this reason, the second dataset can be used to improve the estimation of the regularizer by providing additional information on the intrinsic geometry of the regressors. Therefore, in order to apply the semi-supervised regression in this way, it is necessary that Assumption 1.1 holds.

To enter in more details, let us define the following vectors

$$\boldsymbol{g}_s = \begin{bmatrix} g\left(x_1\right) & \cdots & g\left(x_{n_s}\right) \end{bmatrix} \in \mathbb{R}^{1 \times n_s} \tag{1.181}$$

$$\boldsymbol{g}_{su} = \begin{bmatrix} g\left(x_1\right) & \cdots & g\left(x_{n_s}\right) & g\left(x_{n_s+1}\right) & \cdots & g\left(x_{n_s+n_u}\right) \end{bmatrix} \in \mathbb{R}^{1 \times n} \tag{1.182}$$

$$\boldsymbol{y}_s = \begin{bmatrix} y_1 & \cdots & y_{n_s} \end{bmatrix} \in \mathbb{R}^{1 \times n_s} \tag{1.183}$$

where the regressors and the output are taken from the datasets (1.179) and (1.180).

Now, to better approximate the intrinsic geometry, we can define a new extended regressors graph that uses the regressors from both datasets. In this way, following the reasoning of the previous section, the intrinsic regularizer can be approximated as

$$\|g\|_{\mathcal{I}}^2 = \int_{\mathcal{X}} \|\nabla g(x)\| \, p_x(x) \; dx \simeq \boldsymbol{g}_{su} \boldsymbol{L} \boldsymbol{g}_{su}^\top \tag{1.184}$$

where $\boldsymbol{L} \in \mathbb{R}^{n \times n}$ is the Laplacian matrix of the extended regressors graph.

Remark 1.15. Since we are using more regressors this approximation is more accurate than the one that can be obtained using only the supervised regressors.

Using this improved approximation the cost function (1.168) that use both type of regularization becomes

$$.\hat{g} = \arg\min_{g \in \mathcal{H}} \left\{ \|\boldsymbol{y}_s - \boldsymbol{g}_s\|_2^2 + \tau \|g\|_{\mathcal{H}}^2 + \mu \boldsymbol{g}_{su} \boldsymbol{L} \boldsymbol{g}_{su}^\top \right\} \tag{1.185}$$

In order to compute $\hat{g}$, we need to generalize the representer theorem to this new cost function. This new theorem can be found in [11].

Theorem 1.9 (Representer Theorem for semi-supervised regressions [11])**.** *Let $\hat{g}$ be as in* (1.185)*. Then, exists $\boldsymbol{c} \in \mathbb{R}^{n \times 1}$ such that*

$$\hat{g} = \sum_{i=1}^{n} c_i r_{\boldsymbol{x}_i} \tag{1.186}$$

where $r_{\boldsymbol{x}} \in \mathcal{H}$ is the representer of $\boldsymbol{x} \in \mathcal{X}$, as defined in Definition 1.2.

This theorem is a generalization of the other Representer theorems, see Theorem 1.7 and Theorem 1.8.

In this settings, the optimization problem boils down to a finite dimensional one. In particular, following the same reasoning used for the supervised case, we obtain

$$\hat{g} = \sum_{i=1}^{n_u + n_s} \hat{c}_i r_{\boldsymbol{x}_i} \tag{1.187}$$

$$\hat{\boldsymbol{c}} = \arg\min_{\boldsymbol{c} \in \mathbb{R}^{n \times 1}} \left\{ \left\| \boldsymbol{y} - \boldsymbol{c}^\top \boldsymbol{P} \boldsymbol{K} \right\|_2^2 + \tau \boldsymbol{c}^\top \boldsymbol{K} \boldsymbol{c} + \mu \boldsymbol{c}^\top \boldsymbol{K} \boldsymbol{L} \boldsymbol{K} \boldsymbol{c} \right\} \tag{1.188}$$

where $\boldsymbol{K} \in \mathbb{R}^{n \times n}$ is the kernel matrix computed using all the regressors, from both datasets, and

$$\boldsymbol{y} = \begin{bmatrix} \boldsymbol{y}_s & \boldsymbol{0}_{1 \times n_u} \end{bmatrix} \in \mathbb{R}^{1 \times n} \tag{1.189}$$

$$\boldsymbol{P} = \begin{bmatrix} \boldsymbol{I}_{n_s} & \boldsymbol{0}_{n_s \times n_u} \\ \boldsymbol{0}_{n_u \times n_s} & \boldsymbol{0}_{n_u \times n_u} \end{bmatrix} \in \mathbb{R}^{n \times n} \tag{1.190}$$

is a matrix that selects the part of $\boldsymbol{K}$ that is needed to compute the loss term of the cost function. This is a quadratic optimization problem whose minimizer can be computed as a solution of the linear system

$$\boldsymbol{K} \left(\boldsymbol{P} \boldsymbol{K} + \tau \boldsymbol{I}_n + \mu \boldsymbol{L} \boldsymbol{K} \right) \hat{\boldsymbol{c}} = \boldsymbol{K} \boldsymbol{y}^\top \tag{1.191}$$

as usual, if $\boldsymbol{K}$ is invertible, we can simplify $\boldsymbol{K}$ on both side of the equation

$$(\boldsymbol{P}\boldsymbol{K} + \tau \boldsymbol{I}_n + \mu \boldsymbol{L}\boldsymbol{K})\,\hat{\boldsymbol{c}} = \boldsymbol{y}^\top \tag{1.192}$$

$$\hat{\boldsymbol{c}} = (\boldsymbol{P}\boldsymbol{K} + \tau \boldsymbol{I}_n + \mu \boldsymbol{L}\boldsymbol{K})^{-1}\,\boldsymbol{y}^\top \tag{1.193}$$

This linear system is often ill-conditioned because:

- the term $\boldsymbol{P}\boldsymbol{K}$ is a matrix with n_u columns equal to $\mathbf{0}_{n\times 1}$ and therefore is rank-deficient;
- the regularizer $\boldsymbol{L}\boldsymbol{K}$ is also rank deficient because $\boldsymbol{L}$ has at least one null eigenvalue [34].

therefore, the second term is the only full-rank matrix between the three of them. for this reason, τ has to be large enough to make this system well-conditioned and it cannot be omitted. A possible approach to deal with the numerical problems, that are caused by the ill-conditioning of this matrix, is proposed in Section 3.4 as a new contribution of this thesis.

1.5 Hyper-parameters selection

In this chapter, we have discussed how to identify a non-linear function given a set of data using kernel methods in both supervised and semi-supervised settings. These methods require the selection of some hyper-parameters:

- the Tikhonov regularization strength τ;
- the kernel hyper-parameters $\boldsymbol{\psi}$ that determines the hypothesis space and properties of the Tikhonov regularizer.

Furthermore, if the manifold regularization is employed, it is also necessary to tune

- the manifold regularization strength μ;
- the eventual hyper-parameters $\boldsymbol{\rho}$ needed to construct the regressors graph.

In the remainder of this section the vector that contains all the hyper-parameters is called $\boldsymbol{\zeta}$, i.e.

$$\boldsymbol{\zeta} = \begin{bmatrix} \mu & \tau & \boldsymbol{\psi} & \boldsymbol{\rho} \end{bmatrix}^\top \in \mathbb{R}^{n_\zeta \times 1} \tag{1.194}$$

where n_ζ is the number of hyper-parameters.

The number of hyper-parameters n_ζ can become very large in particular when a complex combined kernel is used. In fact, recalling Theorem 1.5 and Theorem 1.6, it is possible to combine a lot of different kernels in order to shape the RKHS to our needs.

For example, in [104, Section 5.4] they try to model the Carbon dioxide (CO_2) concentration as a function of time using the dataset [60]. To do so, they proposed a combined kernel with 9 different hyper-parameters. Another important example, that will be discussed in more details in Section 2.3, is the kernel proposed in [97] for non-linear system identification. This kernel has 10 different hyper-parameters.

For this reason, the tuning of the hyper-parameters is not trivial. The more convenient way to solve this problem is to use the prior knowledge available on the system. In particular, these hyper-parameters impact the optimization procedure in an interpretable way and therefore they can be set exploiting the available information. For example, in section 1.3, we have shown that τ is equal to the measurements noise. Therefore, if this information is known, it is possible to set τ beforehand. Also, a lot of the common kernels hyper-parameters have a practical interpretation. For example, the hyper-parameters a of the band-limited kernel (see example 1.4 for more details) imposes a limit in the frequency domain of the identified function.

However, prior knowledge is not always enough for the fine-tuning of the hyper-parameters and some data-driven methods are necessary. Furthermore, sometimes the prior information can be wrong and therefore it is not advisable to put strict constraints on the hyper-parameters that can exclude possible explanation of the data.

For this reason, in the literature, there are a lot of different studies on this topic [104, 121]. This problem is often treated as a complexity tuning problem because the hyper-parameters determines the the complexity of the hypothesis set. Therefore, they have to be tuned by leveraging the bias-variance trade-off [17, 44]. The three most common ways to tune them are: cross-validation, Generalized Cross-Validation (GCV), and marginal likelihood (ML) optimization. However, in the literature there are other methods such as the Stein's Unbiased Risk Estimate (SURE) [118].

1.5.1 Cross-validation

The cross-validation [17, 44] is a technique widely used for the hyper-parameters selection and the tuning of the complexity of the identified object in many different statistical learning problems.

The basic idea is to divide the available dataset in two disjointed parts:

- the first part $\mathcal{D}_T$, called *training dataset*, is used to identify the model;
- the second part $\mathcal{D}_V$, called *validation dataset*, is used to estimate the out-of-sample performance of the identified model, i.e. the performance on data points that are not employed for the identification;

Then, the selected hyper-parameters are the one that maximize the performance of the estimated out-of-sample performance. In mathematical form, we can write

$$\hat{\boldsymbol{\zeta}} = \underset{\boldsymbol{\zeta} \in \mathbb{R}^{n_\zeta \times 1}}{\arg\min} \left\{ \sum_{\boldsymbol{x} \in \mathcal{D}_V} L\left(y_i, \hat{g}^{\boldsymbol{\zeta}}_{\mathcal{D}_T}(\boldsymbol{x}_i) \right) \right\} \tag{1.195}$$

where $L : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ is a loss function that is small when the two arguments are similar and $\hat{g}^{\boldsymbol{\zeta}}_{\mathcal{D}_T}$ is the estimated function using only the training dataset $\mathcal{D}_T$ and employing the hyper-parameters $\boldsymbol{\zeta}$.

The drawback of this procedure, called *hold-out cross-validation*, is that only a part of the dataset is actually used for training purpose. To solve this problem, usually, the so-called *k-fold cross-validation* is used instead. Here, the dataset is divided in $k \in \mathbb{N} \setminus \{0\}$ disjointed parts $\mathcal{D}_1, \ldots, \mathcal{D}_k$, called *folds*, with approximately the same size. Then $k-1$ parts are used for training and the left-out part is used as validation dataset to estimate the performance. This procedure is then repeated k times with all the possible different validation dataset. The hyper-parameters selected are the one the maximize the mean performance of the different

k trials. In mathematical form, this is equivalent to

$$\hat{\boldsymbol{\zeta}} = \underset{\boldsymbol{\zeta} \in \mathbb{R}^{n_\zeta \times 1}}{\arg\min} \left\{ \frac{1}{k} \sum_{i=1}^{k} \sum_{\boldsymbol{x} \in \mathcal{D}_i} L\left(y_i, \hat{g}^{\boldsymbol{\zeta}}_{\mathcal{D} \setminus \mathcal{D}_i} (\boldsymbol{x}_i) \right) \right\} \tag{1.196}$$

where $\hat{g}^{\boldsymbol{\zeta}}_{\mathcal{D} \setminus \mathcal{D}_i}$ is the estimated function using the data from all the dataset excluding the fold $\mathcal{D}_i$ and employing the hyper-parameters $\boldsymbol{\zeta}$.

In the k-fold cross-validation, all the dataset is used for training and all cases appear as validation cases at least one time. However, it is necessary to train the model k times. It can be shown that smaller k provides a biased estimation of the out-of-sample performance with small variance, vice-versa larger k provides less biased estimation with more variance. For more details about the theoretical properties of this method, refer to chapter 7 of [44]. In practical application, the number of folds used is usually between 3 and 10.

A special case of the k-fold cross-validation is the Leave One Out Cross-Validation (LOOCV) where the number of folds is the same as the number of data. Here, the training procedure is executed on $n-1$ regressors and validated on a single regressor. Therefore

$$\hat{\boldsymbol{\zeta}}_{loocv} = \underset{\boldsymbol{\zeta} \in \mathbb{R}^{n_\zeta \times 1}}{\arg\min} \left\{ \frac{1}{n} \sum_{i=1}^{n} L\left(y_i, \hat{g}^{\boldsymbol{\zeta}}_{\mathcal{D} \setminus \{\boldsymbol{x}_i\}} (\boldsymbol{x}_i) \right) \right\} \tag{1.197}$$

where $\hat{g}^{\boldsymbol{\zeta}}_{\mathcal{D} \setminus \{\boldsymbol{x}_i\}}$ is the estimated function using the data from all the dataset excluding the regressor $\boldsymbol{x}_i$ and the hyper-parameters $\boldsymbol{\zeta}$.

It can be shown that the LOOCV provides an approximately unbias estimation but with large variance. However, this type of cross-validation has a significant computational burden since it requires the training of n different models. However, if the manifold regularization is not used, i.e. $\mu = 0$, it is possible to implement the LOOCV in a very efficient manner that has a $O\left(n^3\right)$ computational complexity [104].

1.5.2 Generalized cross-validation

The various cross-validation methods shown before require to split the dataset. When the dataset is small this is not always possible. In these cases, it is necessary to develop a way to tune them without training the model on a smaller training dataset. The *generalized cross-validation* [44, 131] is a way to compute an approximation of the LOOCV that requires to train the model only on the complete dataset one time.

To apply this method is necessary to introduce the concept of degrees of freedom (dof) of the method. To understand this concept consider a classic linear regression problem with d parameters. In this case, the number of parameters of the method is given by the actual number of parameters used d, and the space of possible solutions has dimension d. When we employ some sort of regularization some of the previously possible solutions becomes too “big” to be accepted as such. Therefore the estimated model is constrained to a smaller set. This result in a decrease of degrees of freedom. For this reason, the degrees of freedom is a way to assess the complexity of the hypothesis set of the method.

More formally consider the following definition.

Definition 1.7 (degrees of freedom (dof) [44]). *Let $\hat{g}$ be the estimated function obtained using the dataset*

$$\mathcal{D} = \{(\boldsymbol{x}_i, y_i) | \, i = 1, \dots, n\} \tag{1.198}$$

taken from the probabilistic model $y_i = g(\boldsymbol{x}_i) + e_i$ *where* $\mathrm{Var}(e_i) = \beta^2$ *. Then the degrees of freedom (dof) are defined as*

$$\mathrm{dof}(\hat{g}) = \frac{1}{\eta^2} \sum_{i=1}^{n} \mathrm{Cov}(\hat{g}(\boldsymbol{x}_i), y_i) \tag{1.199}$$

Remark 1.16. To understand the intuition behind this definition, consider the fact that when the degrees of freedom are higher the estimated outputs will be similar to measured ones and therefore the covariance is large.

In the supervised kernel methods explained before, we have that

$$\left(\hat{\boldsymbol{g}}^{\boldsymbol{\zeta}}\right)^\top = \boldsymbol{K}\hat{\boldsymbol{c}} \tag{1.200}$$

$$\hat{\boldsymbol{c}} = (\boldsymbol{K} + \tau \boldsymbol{I}_n + \mu \boldsymbol{L}\boldsymbol{K})^{-1} \boldsymbol{y}^\top \tag{1.201}$$

where $\hat{\boldsymbol{g}}^{\boldsymbol{\zeta}} = \left[\hat{g}^{\boldsymbol{\zeta}}(\boldsymbol{x}_1), \dots, \hat{g}^{\boldsymbol{\zeta}}(\boldsymbol{x}_n)\right] \in \mathbb{R}^{1\times n}$ and $\hat{g}^{\boldsymbol{\zeta}}$ is the estimated function employing the hyperparameters $\boldsymbol{\zeta}$. Therefore

$$\left(\hat{\boldsymbol{g}}^{\boldsymbol{\zeta}}\right)^\top = \boldsymbol{K}(\boldsymbol{K} + \tau \boldsymbol{I}_n + \mu \boldsymbol{L}\boldsymbol{K})^{-1} \boldsymbol{y}^\top \tag{1.202}$$

$$= \boldsymbol{S}(\boldsymbol{\zeta})\, \boldsymbol{y}^\top \tag{1.203}$$

where $\boldsymbol{S}(\boldsymbol{\zeta}) = \boldsymbol{K}(\boldsymbol{K} + \tau \boldsymbol{I}_n + \mu \boldsymbol{L}\boldsymbol{K})^{-1} \in \mathbb{R}^{n\times n}$. It can be shown [44] that when the estimated output can be compute using a linear transformation on the measurements the degrees of freedom are:

$$\mathrm{dof}\left(\hat{g}^{\boldsymbol{\zeta}}\right) = \mathrm{Tr}\left[\boldsymbol{S}(\boldsymbol{\zeta})\right] \tag{1.204}$$

$$= \mathrm{Tr}\left[\boldsymbol{K}(\boldsymbol{K} + \tau \boldsymbol{I}_n + \mu \boldsymbol{L}\boldsymbol{K})^{-1}\right] \tag{1.205}$$

Since the degrees of freedomm provides a way to assess the complexity of the estimated model, we can select the hyper-parameters by finding the right trade-off between the performance on the dataset and the number of degrees of freedom. This is achieved by the generalized cross-validation. In particular, we have [44]

$$\hat{\boldsymbol{\zeta}}_{gcv} = \arg\min_{\boldsymbol{\zeta}\in\mathbb{R}^{n_\zeta \times 1}} \left\{ \frac{1}{n} \sum_{i=1}^{n} \left(\frac{y_i - \hat{g}^{\boldsymbol{\zeta}}(\boldsymbol{x}_i)}{1 - \frac{\mathrm{dof}\left(\hat{g}^{\boldsymbol{\zeta}}\right)}{n}} \right)^2 \right\} \tag{1.206}$$

The numerator of the cost function decreases with the performance of the estimation on the dataset while the denominator decreases with the number of degrees of freedom. Therefore, the minimizer $\hat{\boldsymbol{\zeta}}_{gcv}$ provides a trade-off between the complexity of the model and the performance on the training dataset.

For numerical reasons, it is convenient to minimize the natural logarithm of the GCV cost function. Therefore:

$$\hat{\boldsymbol{\zeta}}_{gcv} = \underset{\boldsymbol{\zeta}\in\mathbb{R}^{n_\zeta \times 1}}{\arg\min} \left\{ \log \left[\frac{1}{n}\sum_{i=1}^{n} \left(\frac{y_i - \hat{g}^{\zeta}(\boldsymbol{x}_i)}{1 - \dfrac{\mathrm{dof}\left(\hat{g}^{\zeta}\right)}{n}} \right)^2 \right] \right\} \tag{1.207}$$

$$= \underset{\boldsymbol{\zeta}\in\mathbb{R}^{n_\zeta \times 1}}{\arg\min} \left\{ \log \left[\frac{n^2}{n}\sum_{i=1}^{n} \left(\frac{y_i - \hat{g}^{\zeta}(\boldsymbol{x}_i)}{n - \mathrm{dof}\left(\hat{g}^{\zeta}\right)} \right)^2 \right] \right\} \tag{1.208}$$

$$= \underset{\boldsymbol{\zeta}\in\mathbb{R}^{n_\zeta \times 1}}{\arg\min} \left\{ \log \left[\frac{\sum_{i=1}^{n} \left(y_i - \hat{g}^{\zeta}(\boldsymbol{x}_i)\right)^2}{\left(n - \mathrm{dof}\left(\hat{g}^{\zeta}\right)\right)^2} \right] \right\} \tag{1.209}$$

$$= \underset{\boldsymbol{\zeta}\in\mathbb{R}^{n_\zeta \times 1}}{\arg\min} \left\{ \log \left[\|\boldsymbol{y} - \hat{\boldsymbol{g}}\|^2 \right] - 2\log\left[n - \mathrm{dof}\left(\hat{g}^{\zeta}\right) \right] \right\} \tag{1.210}$$

$$= \underset{\boldsymbol{\zeta}\in\mathbb{R}^{n_\zeta \times 1}}{\arg\min} \left\{ \log \|\boldsymbol{y} - \hat{\boldsymbol{g}}\| - \log\left[n - \mathrm{dof}\left(\hat{g}^{\zeta}\right) \right] \right\} \tag{1.211}$$

Furthermore, recalling that $\hat{\boldsymbol{g}}^\top = \boldsymbol{S}(\boldsymbol{\zeta})\,\boldsymbol{y}^\top$ and $\mathrm{dof}\left(\hat{g}^{\zeta}\right) = \mathrm{Tr}\left[\boldsymbol{S}(\boldsymbol{\zeta})\right]$, we obtain:

$$\hat{\boldsymbol{\zeta}}_{gcv} = \underset{\boldsymbol{\zeta}\in\mathbb{R}^{n_\zeta \times 1}}{\arg\min} \left\{ \log \left\| \left(\boldsymbol{I}_n - \boldsymbol{S}(\boldsymbol{\zeta})\right)\boldsymbol{y}^\top \right\| - \log\left[n - \mathrm{Tr}\left[\boldsymbol{S}(\boldsymbol{\zeta})\right]\right] \right\} \tag{1.212}$$

therefore, it is only necessary to compute the matrix $\boldsymbol{S}(\boldsymbol{\zeta})$ in order to evaluate the cost function.

1.5.3 Marginal likelihood optimization

In the learning theory, a typical approach for parameter estimation is the maximization of the likelihood function. This function is the pdf of the conditional distribution of the available measurements given a certain set of parameters. The same reasoning can be extended to the hyper-parameters if it is possible to compute the distribution $p\left(\boldsymbol{y}\,|\boldsymbol{X},\boldsymbol{\zeta}\right)$.

In section 1.3, it is shown that

$$p\left(\boldsymbol{y}\,|\boldsymbol{g},\boldsymbol{X},\boldsymbol{\zeta}\right) = \mathcal{N}\left(\boldsymbol{y}^\top \middle| \boldsymbol{g}^\top, \beta^2\boldsymbol{I}_n\right) \tag{1.213}$$

$$p\left(\boldsymbol{g}\,|\boldsymbol{X},\boldsymbol{\zeta}\right) = \mathcal{N}\left(\boldsymbol{y}^\top \,|\boldsymbol{0}_{n\times 1}, \beta^2\boldsymbol{I}_n\right) \tag{1.214}$$

therefore, using the conjugacy relations of the normal distribution [17], we have:

$$p\left(\boldsymbol{y}\,|\boldsymbol{X},\boldsymbol{\zeta}\right) = \int p\left(\boldsymbol{y}\,|\boldsymbol{g},\boldsymbol{X},\boldsymbol{\zeta}\right) p\left(\boldsymbol{g}\,|\boldsymbol{X},\boldsymbol{\zeta}\right)\, d\boldsymbol{g} \tag{1.215}$$

$$= \mathcal{N}\left(\boldsymbol{y}^\top \,|\boldsymbol{0}_{n\times 1}, \boldsymbol{K} + \beta^2\boldsymbol{I}_n\right) \tag{1.216}$$

now, it is possible to select the hyper-parameters the maximize this pdf.

$$\hat{\boldsymbol{\zeta}}_{mml} = \underset{\boldsymbol{\zeta}\in\mathbb{R}^{n_\zeta \times 1}}{\arg\max} \left\{ \mathcal{N}\left(\boldsymbol{y}^\top \,|\boldsymbol{0}_{n\times 1}, \boldsymbol{K} + \beta^2\boldsymbol{I}_n\right) \right\} \tag{1.217}$$

As usual, to compute this minimizer it is convenient to minimize the negative logarithm of the pdf to remove the exponential of the normal distribution pdf.

$$\begin{aligned}
\hat{\zeta}_{mml} &= \underset{\zeta \in \mathbb{R}^{n_\zeta \times 1}}{\arg\min} \left\{ -\log \left[\mathcal{N}\left(\boldsymbol{y}^\top \left| \boldsymbol{0}_{n\times 1}, \boldsymbol{K} + \beta^2 \boldsymbol{I}_n \right.\right) \right] \right\} && (1.218)\\
&= \underset{\zeta \in \mathbb{R}^{n_\zeta \times 1}}{\arg\min} \left\{ -\log \left[\frac{\exp\left[-\frac{1}{2} \boldsymbol{y} \left(\boldsymbol{K} + \beta^2 \boldsymbol{I}_n \right)^{-1} \boldsymbol{y}^\top \right]}{\sqrt{2^n \pi^n \det\left(\boldsymbol{K} + \beta^2 \boldsymbol{I}_n \right)}} \right] \right\} && (1.219)\\
&= \underset{\zeta \in \mathbb{R}^{n_\zeta \times 1}}{\arg\min} \left\{ \frac{1}{2} \boldsymbol{y} \left(\boldsymbol{K} + \beta^2 \boldsymbol{I}_n \right)^{-1} \boldsymbol{y}^\top + \log\left[\sqrt{2^n \pi^n \det\left(\boldsymbol{K} + \beta^2 \boldsymbol{I}_n \right)} \right] \right\} && (1.220)\\
&= \underset{\zeta \in \mathbb{R}^{n_\zeta \times 1}}{\arg\min} \left\{ \frac{1}{2} \boldsymbol{y}\hat{\boldsymbol{c}} + \frac{1}{2} \log\det\left(\boldsymbol{K} + \beta^2 \boldsymbol{I}_n \right) + \frac{n}{2} \log\left(2\pi \right) \right\} && (1.221)\\
&= \underset{\zeta \in \mathbb{R}^{n_\zeta \times 1}}{\arg\min} \left\{ \boldsymbol{y}\hat{\boldsymbol{c}} + \log\det\left(\boldsymbol{K} + \beta^2 \boldsymbol{I}_n \right) \right\} && (1.222)
\end{aligned}$$

This cost function requires the computation of the determinant of the square matrix $\boldsymbol{K} + \beta^2 \boldsymbol{I}_n$. Since this matrix is square a positive definite, it is possible to employ the Cholesky decomposition [52] in order to obtain the lower rectangular matrix $\boldsymbol{Q} \in \mathbb{R}^{n \times n}$ with strictly positive diagonal elements such that

$$\boldsymbol{K} + \beta^2 \boldsymbol{I}_n = \boldsymbol{Q}\boldsymbol{Q}^\top \tag{1.223}$$

with this decomposition, it is possible to

- compute the vector $\hat{\boldsymbol{c}}$ by solving two triangular systems [52]

$$\begin{aligned}
\boldsymbol{Q}\boldsymbol{z} &= \boldsymbol{y}^\top && (1.224)\\
\boldsymbol{Q}^\top \hat{\boldsymbol{c}} &= \boldsymbol{z} && (1.225)
\end{aligned}$$

- compute the determinant of $\boldsymbol{K} + \beta^2 \boldsymbol{I}_n$ with a simple multiplication

$$\begin{aligned}
\det\left(\boldsymbol{K} + \beta^2 \boldsymbol{I}_n \right) &= \det\left(\boldsymbol{Q}\boldsymbol{Q}^\top \right) && (1.226)\\
&= \det\left(\boldsymbol{Q} \right)^2 && (1.227)\\
&= \prod_{i=1}^{n} \boldsymbol{Q}_{i,i}^2 && (1.228)
\end{aligned}$$

  where $\boldsymbol{Q}_{i,i}$ is the i-th element on the diagonal of $\boldsymbol{Q}$.

Then, the cost function becomes

$$\begin{aligned}
\hat{\zeta}_{mml} &= \underset{\zeta \in \mathbb{R}^{n_\zeta \times 1}}{\arg\min} \left\{ \boldsymbol{y}\hat{\boldsymbol{c}} + \log\det\left(\prod_{i=1}^{n} \boldsymbol{Q}_{i,i}^2 \right) \right\} && (1.229)\\
&= \underset{\zeta \in \mathbb{R}^{n_\zeta \times 1}}{\arg\min} \left\{ \boldsymbol{y}\hat{\boldsymbol{c}} + 2\sum_{i=1}^{n} \log \boldsymbol{Q}_{i,i} \right\} && (1.230)
\end{aligned}$$

This cost function is composed of two terms: the first decreases with the performance on the training dataset of the estimated model while the second one is a penalization term on the more complex model. For additional theoretical and computational details see [17, 104].

Remark 1.17. This method can be used only when the manifold regularization is not used, i.e. $\mu = 0$. If $\mu > 0$, the Bayesian perspective, explained in Section 1.3, does not work and therefore it is not possible to write the marginal likelihood $p\left(\boldsymbol{y} \,|\boldsymbol{X}, \boldsymbol{\zeta}\,\right)$. This problem will be tackled in Chapter 6 as a new contribution of this thesis.

CHAPTER 2

Kernel-based methods for dynamic system identification

This chapter overviews how kernel-methods can be used in system identification where the relation between inputs and outputs is dynamic and not static. In particular, it is shown how to employ kernel-based approaches for non-parametric system identification for different classes of systems. Both linear and non-linear system will be explored.

This chapter is organized as follow:

- Section 2.1 explains how to use kernel methods for discrete linear systems;
- Section 2.2 illustrates how to use kernel methods for continuous linear systems;
- Section 2.3 delves into the identification of discrete non-linear system using kernel-methods;

In the literature, kernel methods are used for other classes of systems, such as LPV [37, 51, 107] or LTV [68]. However, these methods are outside the focus of this thesis and they will not be treated.

2.1 Discrete-time linear system identification

Single-Input Single-Output (SISO) Linear and Time-Invariant (LTI) discrete system identification is the most studied argument in the system identification community. This kind of systems are simple, they have a large number of well-known properties and they can synthesize a large number of common phenomena. Furthermore, algorithms for this kind of model are well-known [72, 98], implemented in a lot of different libraries [63, 71] and with strong theoretical guarantees on the stability of the estimated model.

2.1.1 Parametric system identification

A common family of discrete Single-Input Single-Output (SISO) Linear and Time-Invariant (LTI) systems, that is commonly used in system identification, is the AutoRegressive with an eXogenous variable (ARX) family [19]. In this kind of model the samples of the output can be computed using the recursive equation

$$y\left(t_i\right)=\sum_{j=1}^{n_y} \breve{a}_j y\left(t_{i-j}\right)+\sum_{j=1}^{n_u} \breve{b}_j u\left(t_{i-j}\right)+e\left(t_i\right) \tag{2.1}$$

where

- $u : \mathbb{R} \rightarrow \mathbb{R}$ is the input signal (often called exogenous variable or excitation signal);
- $y : \mathbb{R} \rightarrow \mathbb{R}$ is the output signal;
- $t_i = i \cdot T_s$, with $i \in \mathbb{Z}$, are the time instants selected by the sampling process and $T_s \in \mathbb{R}_+$ is the sampling period;
- n_y is the number of autoregressive coefficients;
- n_u is the number of exogenous coefficients;
- $\breve{a}_1, \ldots, \breve{a}_{n_y}$ are the coefficients of the autoregressive part;
- $\breve{b}_1, \ldots, \breve{b}_{n_u}$ are the coefficients of the exogenous part;
- $e : \mathbb{R} \rightarrow \mathbb{R}$ is the noise term.

The samples of the noise are considered IID, the input function is considered known in all the domain and the sampling period is considered to be known.

For compactness sake, the i-th sample of the input, output and noise signal are indicated, respectively, with $u_i = u\left(t_i\right)$, $y_i = y\left(t_i\right)$ and $e_i = e\left(t_i\right)$. Then, the recursive equation (2.1) can be rewritten as

$$y_i=\boldsymbol{x}_i^{\top} \breve{\boldsymbol{\vartheta}}+e_i \tag{2.2}$$

where $n_\vartheta = n_u + n_y$ and

$$\breve{\boldsymbol{\vartheta}}=\left[\begin{array}{cccccc}\breve{a}_1 & \cdots & \breve{a}_{n_u} & \breve{b}_1 & \cdots & \breve{b}_{n_y}\end{array}\right]^{\top} \in \mathbb{R}^{n_\vartheta \times 1} \tag{2.3}$$

$$\boldsymbol{x}_i=\left[\begin{array}{cccccc}y_{i-1} & \cdots & y_{i-n_y} & u_{i-1} & \cdots & u_{i-n_u}\end{array}\right]^{\top} \in \mathbb{R}^{n_\vartheta \times 1} \tag{2.4}$$

are, respectively, the parameters vectors and the t-th regressors.

Now, the objective is to identify the unknown coefficients $\breve{\boldsymbol{\vartheta}}$ with the first n output samples, i.e.

$$\mathcal{D}=\left\{y_i \mid i=1, \ldots, n\right\}. \tag{2.5}$$

The most common used rationale is Prediction Error Method (PEM) [19, 72]. Here, the estimation is obtained by selecting the model that minimizes the one-step prediction error on the available data. This results in the optimization problem:

$$\widehat{\boldsymbol{\vartheta}}=\underset{\boldsymbol{\vartheta} \in \mathbb{R}^{n_\vartheta \times 1}}{\arg \min }\left\{\sum_{i=1}^n\left(y_i-\hat{y}_{i \mid i-1}^{\boldsymbol{\vartheta}}\right)^2\right\} \tag{2.6}$$

where $\hat{y}_{i \mid i-1}^{\boldsymbol{\vartheta}}$ is the one-step predictor of the model defined using the parameters $\boldsymbol{\vartheta}$. For the ARX family, it is possible to show [72] that the optimal predictor is

$$\hat{y}_{i \mid i-1}^{\boldsymbol{\vartheta}}=\boldsymbol{x}_i^{\top} \boldsymbol{\vartheta} \tag{2.7}$$

therefore

$$\widehat{\boldsymbol{\vartheta}} = \underset{\boldsymbol{\vartheta} \in \mathbb{R}^{n_\vartheta \times 1}}{\arg\min} \left\{ \sum_{i=1}^{n} \left(y_i - \boldsymbol{x}_i^\top \boldsymbol{\vartheta} \right)^2 \right\} \tag{2.8}$$

Remark 2.1. To compute the predictor at the i-th time instant, it is necessary to know the measurements of y for some past time instants. For this reason, the summation in the optimization problem has to be restricted to the cases where the predictor can be computed with the data at hand. Otherwise, it is possible to make some assumption on the state of the system output before the start of the experiment.

This estimator requires the knowledge of n_a and n_b. Usually, this is not the case and they need to be estimated from the dataset. In the literature, this problem is treated as an hyper-parameters selection problem. Therefore, the cross-validation is the most common solution [19, 72]. If the dataset is not large enough to be divided then some sort of penalization on the number of parameters n_ϑ is used, such as Akaike Information Criterion (AIC) [2, 72], Bayesian Information Criterion (BIC) [72, 106, 112] or other similar penalizations [19, 72].

2.1.2 Non-parametric system identification

In recent time, the trend moved to the possibility to estimate the model without using the knowledge on the number of parameters. These methods are called *non-parametric* and they are, usually, kernel methods. The main idea is to use a large number of parameters, potentially an infinite amount, and a regularization term that penalizes overly complex models thanks to RKHS properties.

To do so, consider that the output of a generic discrete LTI model can be computed as

$$y_i = \sum_{\xi=0}^{m} \breve{g}(\xi)\, u_{i-\xi} + e_i \tag{2.9}$$

where $m \in \mathbb{N} \cup \{+\infty\}$, $\breve{g} : \mathbb{N} \to \mathbb{R}$ is the impulse response of the system and e_t is a white noise. Here, the unknown parameters are the, potentially, infinite samples of the impulse response $\{\breve{g}(i)\}_{i=0}^{m}$.

Remark 2.2. It can be shown that this representation is equivalent to the ARX model [128], presented in equation (2.1). In particular, if m is finite, the system is a Finite Impulse Response (FIR) because all the impulse response samples after the time-instants m are 0. Vice versa, when $m = +\infty$ the system is a Infinite Impulse Response (IIR) like the ARX, with $n_a > 0$, model considered before.

To identify the impulse response functions $\breve{g} : \mathbb{N} \to \mathbb{R}$, we can assume that this function is a member of a certain RKHS $\mathcal{H}$ with kernel $k : \mathbb{N} \times \mathbb{N} \to \mathbb{R}$ because the function that we want to identify has $\mathbb{N}$ as domain. Then using the PEM rationale with the Tikhonov regularizer, we obtain

$$\widehat{g} = \underset{g \in \mathcal{H}}{\arg\min} \left\{ \sum_{i=1}^{n} \left(y_i - \hat{y}_{i|i-1}^{g} \right)^2 + \tau \|g\|_{\mathcal{H}}^2 \right\} \tag{2.10}$$

$$= \underset{b \in \mathcal{H}}{\arg\min} \left\{ \sum_{i=1}^{n} \left(y_i - \sum_{\xi=0}^{m} g(\xi)\, u_{i-\xi} \right)^2 + \tau \|g\|_{\mathcal{H}}^2 \right\} \tag{2.11}$$

where $\hat{y}^{g}_{i|i-1} = \sum_{\xi=0}^{m} g(\xi)\, u_{i-\xi}$ is the one-step predictor of the model defined using the impulse response g.

This cost function is different from the one of the classical Tikhonov regularization because, in the loss term, the output measurements are not compared with the evaluation of the unknown function, but with a functional evaluated on the function. In particular, defining the functionals:

$$\begin{aligned} p_i &: \mathcal{H} \to \mathbb{R} \\ g &\to \sum_{\xi=0}^{m} g(\xi)\, u_{t-\xi} \end{aligned} \qquad i = 1, \dots, n \tag{2.12}$$

the cost function becomes

$$\widehat{g} = \arg\min_{g \in \mathcal{H}} \left\{ \sum_{i=1}^{n} \left(y_i - p_i(g)\right)^2 + \tau \left\|g\right\|_{\mathcal{H}}^2 \right\} \tag{2.13}$$

Remark 2.3. It is trivial to show that the functionals p_t, with $t \in \mathbb{N}$, are linear. Therefore

$$p_i(\alpha g_1 + \beta g_2) = \alpha p_i(g_1) + \beta p_i(g_2) \qquad \begin{aligned} &\forall i \in \mathbb{N} \\ &\forall g_1, g_2 \in \mathcal{H} \\ &\forall \alpha, \beta \in \mathbb{R} \end{aligned} \tag{2.14}$$

For this cost function, it is not possible to use the classical representer theorem, see Theorem 1.7. However, there is a more general representer theorem [32, 40, 95] that can be applied to this cost function. In this case, the minimizer can be written in the form

$$g(i) = \sum_{j=1}^{n} c_j p_j(r_i) \tag{2.15}$$

$$= \sum_{j=1}^{n} c_j \sum_{\xi=0}^{m} r_i(\xi)\, u_{j-\xi} \tag{2.16}$$

$$= \sum_{j=1}^{n} c_j \sum_{\xi=0}^{m} u_{j-\xi} k(\xi, i) \tag{2.17}$$

therefore, the optimization problem can be reduced to a n dimensional problem that search the optimal parameters $\hat{\boldsymbol{c}} = [\hat{c}_1, \dots, \hat{c}_n]^\top \in \mathbb{R}^{n \times 1}$. Alternatively, it is also possible to write

$$g = \sum_{j=1}^{n} c_j \sum_{\xi=0}^{m} u_{j-\xi} r_\xi \tag{2.18}$$

$$= \sum_{j=1}^{n} \sum_{\xi=0}^{m} c_j u_{j-\xi} \cdot r_\xi \tag{2.19}$$

where we can see that the estimated function is a weighted sum of representer functions of the space $\mathcal{H}$. In these settings, employing the linearity of the functionals p_t, the norm becomes

$$\|g\|_{\mathcal{H}}^2 = \langle g, g \rangle_{\mathcal{H}} \tag{2.20}$$

$$= \left\langle \sum_{j=1}^{n}\sum_{\xi=0}^{m} c_j u_{j-\xi}\cdot r_\xi, \sum_{h=1}^{n}\sum_{\psi=0}^{m} c_h u_{h-\psi}\cdot r_\psi \right\rangle_{\mathcal{H}} \tag{2.21}$$

$$= \sum_{j=1}^{n}\sum_{h=1}^{n} c_j c_h \sum_{\xi=0}^{m}\sum_{\psi=0}^{m} u_{j-\xi}u_{h-\psi}\left\langle r_\xi, r_\psi\right\rangle_{\mathcal{H}} \tag{2.22}$$

$$= \sum_{j=1}^{n}\sum_{h=1}^{n} c_j c_h \sum_{\xi=0}^{m}\sum_{\psi=0}^{m} u_{j-\xi}u_{h-\psi} k\left(\xi,\psi\right) \tag{2.23}$$

$$= \boldsymbol{c}^\top \boldsymbol{O}\boldsymbol{c} \tag{2.24}$$

where the matrix $\boldsymbol{O}\in\mathbb{R}^{n\times n}$ is a symmetric and positive semi-definite matrix whose (i,j) element is

$$o\left(j,h\right) = \sum_{\xi=0}^{m}\sum_{\psi=0}^{m} u_{j-\xi}u_{h-\psi}k\left(\xi,\psi\right) \tag{2.25}$$

Furthermore, we can see that the loss term becomes

$$\sum_{i=1}^{n}\left(y_i - p_i\left(g\right)\right)^2 = \sum_{i=1}^{n}\left(y_i - p_i\left(\sum_{j=1}^{n}\sum_{\xi=0}^{m} c_j u_{j-\xi}\cdot r_\xi\right)\right)^2 \tag{2.26}$$

$$= \sum_{i=1}^{n}\left(y_i - \sum_{j=1}^{n}\sum_{\xi=0}^{m} c_j u_{j-\xi}\cdot p_i\left(r_\xi\right)\right)^2 \tag{2.27}$$

$$= \sum_{i=1}^{n}\left(y_i - \sum_{j=1}^{n}\sum_{\xi=0}^{m} c_j u_{j-\xi}\cdot\left(\sum_{\psi=0}^{m} r_\xi\left(\psi\right)u_{i-\psi}\right)\right)^2 \tag{2.28}$$

$$= \sum_{i=1}^{n}\left(y_i - \sum_{j=1}^{n} c_j \sum_{\xi=0}^{m}\sum_{\psi=0}^{m} u_{j-\xi}u_{i-\psi}k\left(\xi,\psi\right)\right)^2 \tag{2.29}$$

$$= \sum_{i=1}^{n}\left(y_i - \sum_{j=1}^{n} c_j o\left(j,i\right)\right)^2 \tag{2.30}$$

$$= \left\|\boldsymbol{y} - \boldsymbol{c}^\top\boldsymbol{O}\right\|_2^2 \tag{2.31}$$

obtaining the optimization problem

$$\hat{g} = \sum_{j=1}^{n}\sum_{\xi=0}^{m} c_j u_{j-\xi}\cdot r_\xi \tag{2.32}$$

$$\hat{\boldsymbol{c}} = \underset{\boldsymbol{c}\in\mathbb{R}^{n\times 1}}{\arg\min}\left\{\left\|\boldsymbol{y} - \boldsymbol{c}^\top\boldsymbol{O}\right\|_2^2 + \tau\boldsymbol{c}^\top\boldsymbol{O}\boldsymbol{c}\right\} \tag{2.33}$$

This is a quadratic optimization problem whose optimizer can be computed analytically. In particular, the vector $\hat{\boldsymbol{c}}$ can be computed by solving the linear system

$$\boldsymbol{O}\left(\boldsymbol{O} + \tau\boldsymbol{I}_n\right)\hat{\boldsymbol{c}} = \boldsymbol{O}\boldsymbol{y}^\top \tag{2.34}$$

Therefore, if $\boldsymbol{O}$ is a full rank matrix, we can write

$$\left(\boldsymbol{O}+\tau \boldsymbol{I}_n\right) \hat{\boldsymbol{c}}=\boldsymbol{y}^\top \tag{2.35}$$

$$\hat{\boldsymbol{c}}=\left(\boldsymbol{O}+\tau \boldsymbol{I}_n\right)^{-1} \boldsymbol{y}^\top \tag{2.36}$$

Remark 2.4. The hyperparameters can tuned as indicated in Section 1.5. It is also possible to show that the proposed methods have good asymptotic properties [83, 94]. In particular there are strong results for the GCV [81] and the cross-validation [82] methodologies.

2.1.3 Kernel selection

For the before-mentioned method to work, it is necessary to select the right kernel. Is it important to note that the classical kernels, shown in Section 1.1, are not suitable in this contest. An important restriction on the kernel choice is the computability of the estimated impulse response. In particular, the estimated function is composed by a weighted sum of the functional $p_i\left(r_j\right)$, with $j \in \mathbb{N}$ and $i=1,\ldots,n$, that can be an infinite series. Therefore, these series have to converge to a finite number.

This imposes an important restriction on the suitable kernels that excludes a lot of classical kernels such as the linear kernel, polynomial kernel or gaussian kernel. It can be shown [39] that a necessary condition for the convergence is that

$$\lim_{i \to +\infty} k\left(i,i\right)=0 \tag{2.37}$$

that exclude all the stationary kernels, i.e. the kernels whose evaluation depends only on the difference between the arguments. Furthermore, it is necessary to select a kernel that defines a space that contains only functions that correspond to the impulse response of a stable LTI. In this way, it is possible to guarantee the stability of the identified system. Kernels with these two properties are called *stable kernel* [30, 33, 95]. More formally, we define:

Definition 2.1 (Discrete-time stable kernel [30, 33, 95])**.** *A symmetric and positive semi-definite kernel* $k : \mathbb{N} \times \mathbb{N} \to \mathbb{R}$ *that defines the space* $\mathcal{H}$ *is called* ***stable kernel*** *if and only if* $\mathcal{H} \subseteq l^1$.

Remark 2.5. The before-mentioned definition derives from the fact that an LTI system with impulse response g is Bounded-Input Bounded-Output (BIBO) stable if and only $g \in l^1$. Therefore, if it is imposed that the space $\mathcal{H}$ contains only functions in l^1, it is guaranteed that the identified system is BIBO stable.

To verify if a kernel is stable consider the following two theorems

Theorem 2.1 ([39, 95])**.** *A symmetric and positive semi-definite function* $k : \mathbb{N} \times \mathbb{N} \to \mathbb{R}$ *is a stable kernel if and only if:*

$$\sum_{\xi=1}^{\infty} \left| \sum_{\psi=1}^{\infty} k\left(\xi,\psi\right) a\left(\psi\right) \right| < \infty \qquad \forall a \in l^\infty \tag{2.38}$$

Theorem 2.2 ([39, 95]). *A symmetric and positive semi-definite function $k : \mathbb{N} \times \mathbb{N} \to \mathbb{R}$ is a stable kernel if:*

$$\sum_{\xi=1}^{\infty} \left| \sum_{\psi=1}^{\infty} k(\xi, \psi) \right| < \infty \tag{2.39}$$

The first theorem provides a sufficient and necessary condition, but it is not easily verifiable, while the second one defines only a sufficient condition that can be easily verified.

Remark 2.6. In recent time, it was shown that the stable spline condition is only sufficient, but it is not necessary [18]. In other words, there exist at least a kernel that cannot be considered stable, according to Definition 2.1, but it defines a space that contains only impulse response that corresponds to stable LTI systems.

Definition 2.1 allows understanding if a kernel can be used in these settings, but it does not provide a way to choose the right kernel for the application at hand. In particular, given a non-stable kernel k, it is always possible to make it stable by truncation. In fact, it is straightforward to see that the kernel

$$\tilde{k}(\xi, \psi) = \begin{cases} k(\xi, \psi) & \text{if } \xi \le T \wedge \psi \le T \\ 0 & \text{otherwise} \end{cases} \tag{2.40}$$

where $T \in \mathbb{N}$, is always stable. However, this small trick does not improve the performance of the method significantly, as shown in [95].

It is possible to show [32, 95] that the the most convenient kernel for this application is

$$\overline{k}(\xi, \psi) = \breve{g}(\xi)\, \breve{g}(\psi) \qquad \forall \xi, \psi \in \mathbb{N} \tag{2.41}$$

where $\breve{g}$ is the true impulse response of the system. This kernel is usually called *optimal kernel*[I]. For obvious reason, the optimal kernel is not usable in practice. However, it is possible to select a kernel that mimics this behavior as much as possible. It is possible to see that the representer functions of the optimal kernel are

$$\overline{r}_{\xi} = \breve{g}(\xi) \cdot \breve{g} \qquad \xi \in \mathbb{N} \tag{2.42}$$

Therefore, the representer functions of the optimal kernel correspond to the true impulse response of the system. For this reason, we need a kernel whose representer functions correspond to a valid impulse response of an LTI stable system. The two most used kernels for this application are reported in the following examples.

Example 2.1: Diagonal correlated kernel [32]

The Diagonal Correlated (DC) kernel is a kernel that was recently introduced as a suitable kernel for LTI system identification and it is defined as

$$k_{DC}(a, b) = \lambda \sqrt{\alpha^{a+b}} \beta^{|a-b|} \tag{2.43}$$

where $\lambda \in \mathbb{R}_+$, $\alpha \in \mathbb{R}_+$ and $\beta \in [-1, 1]$. It is straightforward to see that this kernel is a stable kernel for Theorem 2.2.
The value of λ determines the amplitude of the representer functions and therefore it is connected to the static gain of the system. Higher λ corresponds to larger static

[I]for a more formal definition of optimality in this context refer to [32, 95].

gains. The other two parameters α and β provides a way to tune the shape of the response. In Figure 2.1, the representer functions of a DC kernel with different α and β are reported. Qualitatively, α controls how much oscillations there are in the impulse response and β the decay rate.

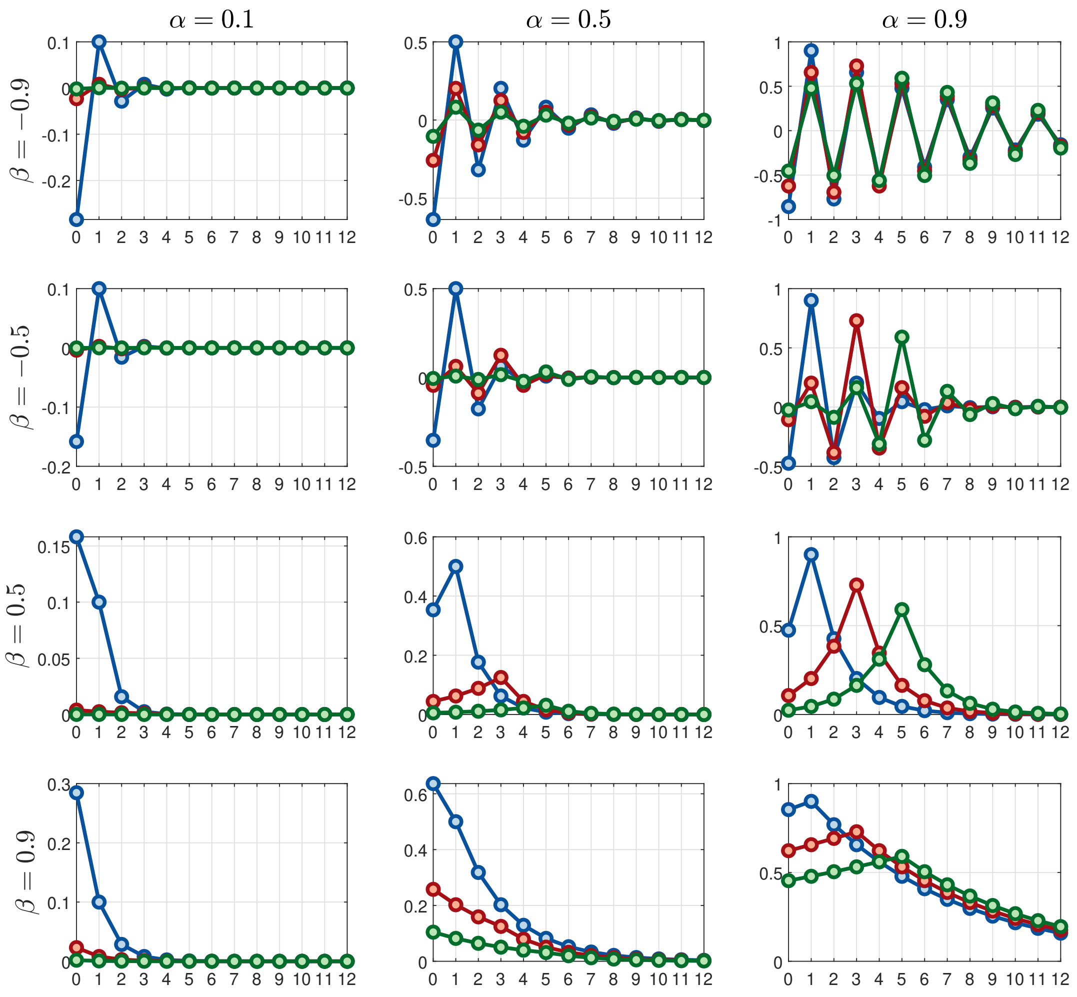

FIGURE 2.1: Plot of three representer functions of the discrete DC kernel with different values of α and β. The parameter λ is set to 1.

Example 2.2: Stable-spline kernel [96]

The stable-splines are a family of kernels introduced in [96] as a way to make the classic spline kernel, explained in Example 1.5, a stable kernel. The stable-spline of order $q \in \mathbb{N} \setminus \{0\}$ is defined as

$$k_q(a,b) = \lambda s_q\left(e^{-\beta a}, e^{-\beta b}\right) \tag{2.44}$$

where s_q is the kernel of the spline of order q (see Example 1.5), $\lambda \in \mathbb{R}_+$ and $\beta \in \mathbb{R}_+$. For example, the first two stable-splines can be computed as

$$k_1(a,b) = \lambda e^{-\beta \max(a,b)} \tag{2.45}$$

$$k_2(a,b) = \lambda \left(\frac{e^{-\beta(a+b+\max(a,b))}}{2} - \frac{e^{-3\beta\max(a,b)}}{6} \right) \tag{2.46}$$

In Chapter 4, as a new contribution of this thesis, a general formula that can be used to compute spline of a given order is provided.
As in the DC kernel, the value of λ determines the amplitude of the representer functions and the static gain of the system. Therefore, higher λ corresponds to larger static gains. The other hyper-parameter β can be used to tune the decay rate over time of the representer functions of the kernel. The effect of β can be seen in Figure 2.2.

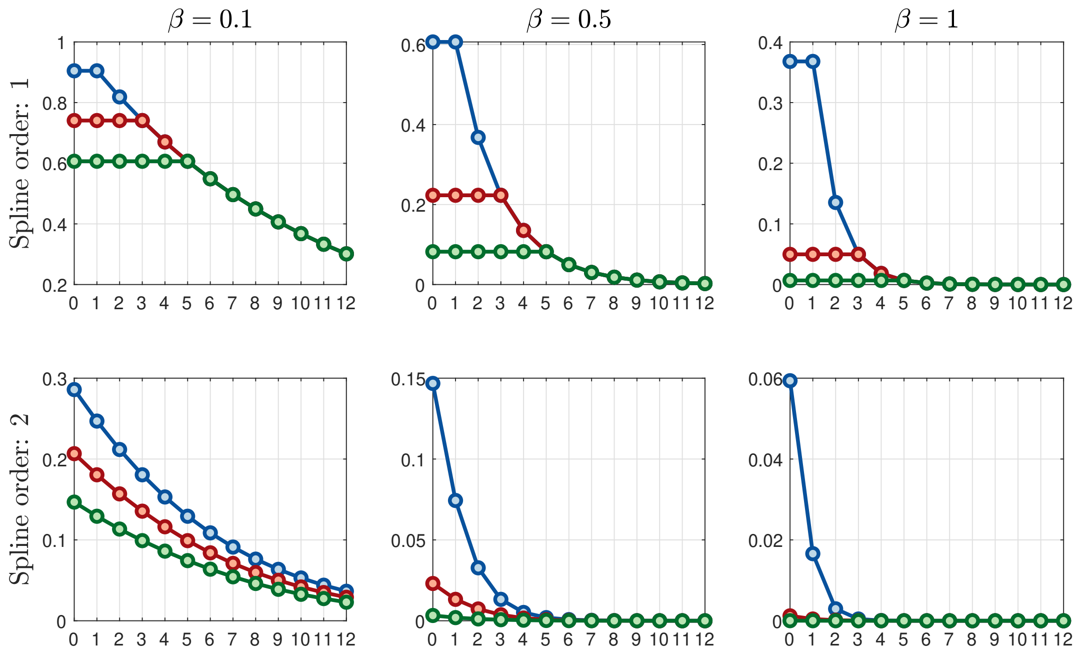

FIGURE 2.2: Plot of three representer functions of the discrete stable-spline kernel with different values of β. The parameter λ is set to 1.

Even if these two kernels are the most used, in the literature there are some new results. In particular, in [30] the author analyzes the problem of the kernel selection in details and he introduces some interesting methodology to tailor the kernel for the application at hand.

2.1.4 Bayesian interpretation

In Section 1.3, we have seen that the classical Tikhonov regularization can be seen from a Bayesian point of view. Furthermore in Section 1.5.3, we have seen that the Bayesian perspective provides a way to tune the kernel-parameters. For this reason, it is important to ask if there exists a Bayesian perspective even for the dynamical case.

The answers in positive as shown in [95]. In particular, it is possible to show that by imposing a Gaussian process prior, with zero mean and variance function k, on the unknown impulse response we obtain a posterior whose mean is equal to the impulse response estimated with the previously explained method. Furthermore, it is possible to show that the marginal likelihood is

$$p(\boldsymbol{y}\,|u,\boldsymbol{\zeta}) = \mathcal{N}\left(\boldsymbol{y}^\top\,|\mathbf{0}_{n\times1}, \boldsymbol{O} + \tau\boldsymbol{I}_n\right) \tag{2.47}$$

where $\boldsymbol{\zeta}$ is the hyper-parameters vector. Therefore, following the reasoning presented in Section 1.5.3, we can select the hyper-parameters by solving the optimization problem

$$\hat{\boldsymbol{\zeta}}_{mml} = \underset{\boldsymbol{\zeta} \in \mathbb{R}^{n_\zeta \times 1}}{\arg\min} \left\{ \boldsymbol{y}\hat{\boldsymbol{c}} + 2 \sum_{i=1}^{n} \log \boldsymbol{Q}_{i,i} \right\} \tag{2.48}$$

where $\boldsymbol{Q} \in \mathbb{R}^{n \times n}$ is the Cholesky decomposition [52] of the matrix $\boldsymbol{O} + \tau \boldsymbol{I}_n$, i.e.

$$\boldsymbol{O} + \tau \boldsymbol{I}_n = \boldsymbol{Q}\boldsymbol{Q}^\top \tag{2.49}$$

2.2 Continuous-time linear system identification

The concepts explained in the previous section about the identification of discrete-time LTI systems can be extended to the identification of continuous-time LTI systems. Following the same reasoning used for the discrete-time systems, we consider that the output of a continuous-time LTI system can be computed as

$$y(t) = \int_0^{+\infty} \breve{g}(\xi)\, u(t - \xi)\, d\xi \tag{2.50}$$

where $\breve{g} : \mathbb{R}_+ \to \mathbb{R}$ is the impulse response of the system.

2.2.1 Non-parametric system identification

The aim is to identify the impulse response of the system using the following dataset

$$\mathcal{D} = \{(t_i, y_i) \,|\, i = 1, \dots, n\} \tag{2.51}$$

where the outputs y_i, with $i = 1, \dots, n$, are taken according to the following probabilistic model

$$y_i = \int_0^{+\infty} \breve{g}(\xi)\, u(t_i - \xi)\, d\xi + e_i \qquad\qquad i = 1, \dots, n \tag{2.52}$$

where e_i, with $i = 1, \dots, n$, are IID output-error noises and $u : \mathbb{R}_+ \to \mathbb{R}$ is the input excitations used during the experiment. Here, we will assume that the excitation signal u is known.

As for the discrete case, the main idea is to assume that the impulse response $\breve{g}$ is an element of a RKHS $\mathcal{H}$ with kernel $k : \mathbb{R}_+ \times \mathbb{R}_+ \to \mathbb{R}$. Then

$$\hat{g} = \underset{g \in \mathcal{H}}{\arg\min} \left\{ \sum_{i=1}^{n} \left(y_i - \int_0^{+\infty} \breve{g}(\xi)\, u(t_i - \xi)\, d\xi \right)^2 + \tau \left\| g \right\|_k^2 \right\} \tag{2.53}$$

defining the functionals

$$\begin{aligned} q_t &: \mathcal{H} \to \mathbb{R} \\ g &\to \int_0^{+\infty} g(\xi)\, u(t_i - \xi)\, d\xi\, d\xi \end{aligned} \qquad t = 1, \dots, n \tag{2.54}$$

we can write

$$\widehat{g} = \arg\min_{g \in \mathcal{H}} \left\{ \sum_{t=1}^{n} (y_i - q_t(g))^2 + \tau \|g\|_{\mathcal{H}}^2 \right\} \tag{2.55}$$

This optimization problem is very similar to the one obtained for discrete systems. The only difference is the functionals definition. However, these functionals are still linear and therefore all the steps reported for the discrete-time case are still valid. The only difference is the integration instead of the summation.

For this reason, the estimated impulse response is:

$$\hat{g}(t) = \sum_{i=1}^{n} \hat{c}_i q_i(r_t) \tag{2.56}$$

$$= \sum_{i=1}^{n} \hat{c}_i \int_0^{+\infty} r_t(\xi)\, u(t_i - \xi)\, d\xi \tag{2.57}$$

$$= \sum_{i=1}^{n} \hat{c}_i \int_0^{+\infty} k(t, \xi)\, u(t_i - \xi)\, d\xi \tag{2.58}$$

where the coefficient $\hat{\boldsymbol{c}} = [\hat{c}_1, \dots, \hat{c}_n]$ can be computed by solving the linear system

$$(\boldsymbol{O} + \tau \boldsymbol{I}_n)\, \hat{\boldsymbol{c}} = \boldsymbol{y}^\top \tag{2.59}$$

$$\hat{\boldsymbol{c}} = (\boldsymbol{O} + \tau \boldsymbol{I}_n)^{-1} \boldsymbol{y}^\top \tag{2.60}$$

where

$$\boldsymbol{y} = \left[\begin{array}{ccc} y_1 & \cdots & y_n \end{array} \right] \in \mathbb{R}^{1 \times n} \tag{2.61}$$

and $\boldsymbol{O} \in \mathbb{R}^{n \times n}$ is the matrix whose (i, j) element is

$$o(i, j) = \int_0^{+\infty} \int_0^{+\infty} u(i - \xi)\, u(j - \psi)\, k(\xi, \psi)\, d\xi\, d\psi \tag{2.62}$$

2.2.2 Kernel selection

The main difference between the continuous-time approach and the discrete-time one is the kernel. In the latter, the kernel define a space that contains sequence while the one used for the continuous-time approach has to contain functions. However, the reasoning used for the discrete-time approach holds even in the continuous-time case.

In particular, we want the kernel to defines a space that contains only functions that correspond to impulse response of BIBO stable systems and it is necessary that the functionals $p_i(r_t)$, with $t \in \mathbb{R}_+$ and $i = 1, \dots, n$, to converge. A continuous kernel that has these properties is called *stable kernel*. This concept can be formalized in the following definition.

Definition 2.2 (Continuous-time stable kernel [30, 33, 95]). *A symmetric and positive semi-definite kernel $k : \mathbb{R}_+ \times \mathbb{R}_+ \to \mathbb{R}$ that define the space $\mathcal{H}$ is called stable kernel if and only if $\mathcal{H} \subseteq L^1$.*

Then, to check if a kernel is stable, it is possible to use the following two theorems.

Theorem 2.3 ([39, 95]). *A symmetric and positive semi-definite function $k : \mathbb{R}_+ \times \mathbb{R}_+ \to \mathbb{R}$ is a stable kernel if and only if:*

$$\int_0^{+\infty} \left| \int_0^{+\infty} k\left(\xi, \psi\right) a\left(\psi\right) \, d\psi \right| d\xi < \infty \qquad \forall a \in L^\infty \tag{2.63}$$

Theorem 2.4 ([39, 95]). *A symmetric and positive semi-definite function $k : \mathbb{R}_+ \times \mathbb{R}_+ \to \mathbb{R}$ is a stable kernel if:*

$$\int_0^{+\infty} \left| \int_0^{+\infty} k\left(\xi, \psi\right) \, d\psi \right| d\xi < \infty \tag{2.64}$$

As in the discrete case, the first theorem provides a sufficient and necessary condition, but it is not easily verifiable, while the second one defines only a sufficient condition that can be easily verified.

Another similarity with the discrete case is the optimal kernel. It can be show [32, 95] that the optimal kernel in the continuous-time settings is

$$\overline{k}\left(\xi, \psi\right) = \breve{g}\left(\xi\right) \breve{g}\left(\psi\right) \tag{2.65}$$

where $\breve{g}$ is the true impulse response of the system under analysis.

Therefore, we need a kernel that defines a space with representer functions similar to the true impulse response of the system. This is the same reasoning used to define the various kernels for the discrete case. In fact, it is possible to employ the same kernels as in the discrete case by enlarging the domain from $\mathbb{N}$ to $\mathbb{R}_+$. For this reason the most popular kernels are

- The continuous DC kernel that is defined as

$$k_{DC}\left(a, b\right) = \lambda \sqrt{\alpha^{a+b}} \beta^{|a-b|} \tag{2.66}$$

  where $\lambda \in \mathbb{R}_+$, $\alpha \in \mathbb{R}_+$ and $\beta \in [0, 1]$. Three representer functions of this kernel are reported in Figure 2.3. For more details, see Example 2.1.

- The continuous stable-spline kernel that is defined as

$$k_q\left(a, b\right) = \lambda s_q \left(e^{-\beta a}, e^{-\beta b}\right) \tag{2.67}$$

  where $q \in \mathbb{N} \setminus \{0\}$ is the spline order, s_q is the spline kernel of order q, $\lambda \in \mathbb{R}_+$ and $\beta \in [0, 1]$. Three representer functions of this kernel are reported in Figure 2.4. For more details, see Example 2.2.

Remark 2.7. The continuous DC kernel is well defined only when $\beta > 0$ because otherwise the term $\beta^{|t_i - t_j|}$ is a complex number when $|t_i - t_j| \notin \mathbb{Z}$.

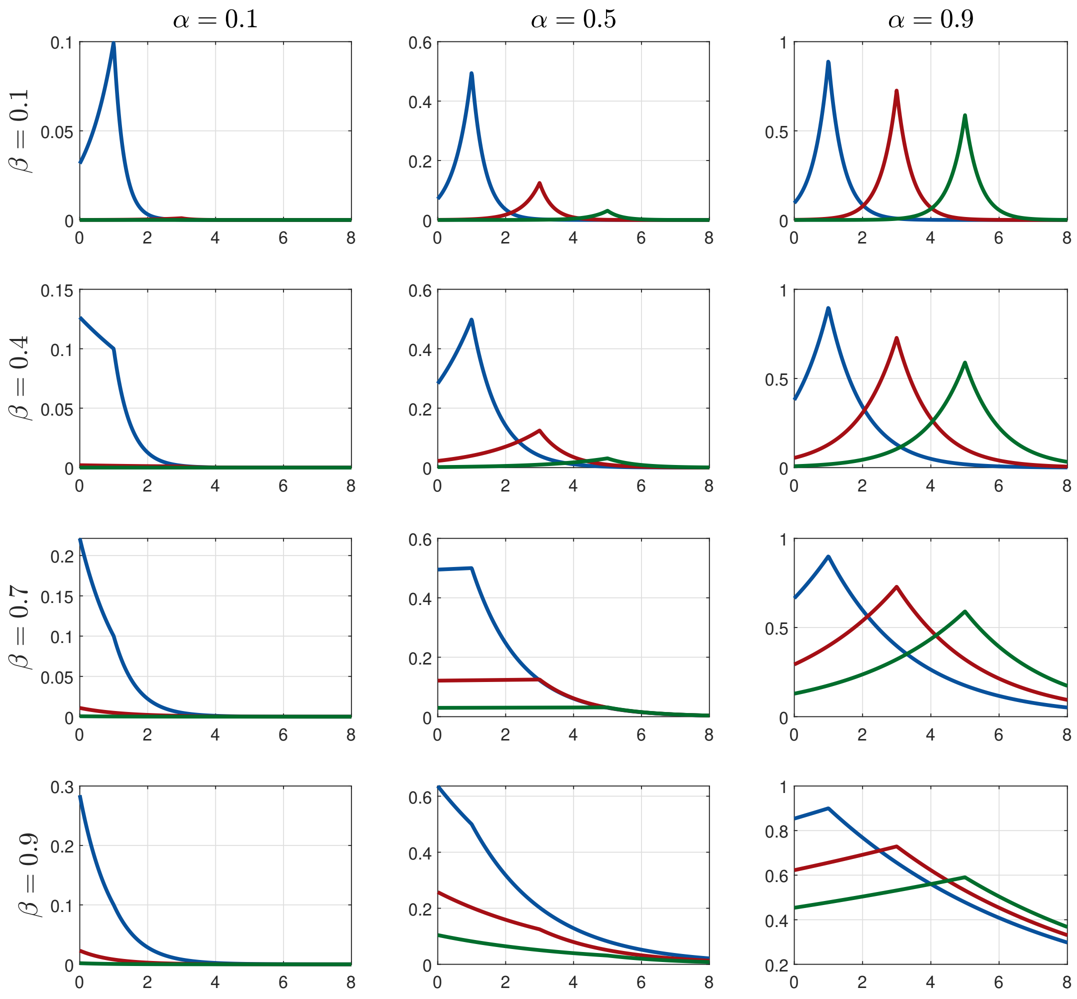

FIGURE 2.3: Plot of three representer functions of the continuous DC kernel with different values of α and β. The parameter λ is set to 1.

Remark 2.8. The Bayesian interpretation is analogue to the one explained for the discrete case in Section 2.1.4. For this reason, the reader can refer to that section or to the literature [32].

2.3 Discrete-time non-linear system identification

In the previous sections, it is illustrated how the system identification community has adapted the kernel methods to the estimation of the impulse response of an LTI system to develop a true black-box algorithm that does not require the knowledge of the system basis structure. In the last decade, kernel methods were employed also for non-linear systems [6, 86, 97].

In these works, the focus is on the identification of Nonlinear AutoRegressive with an eXogenous variable (NARX) models. In this type of models, the output is computed as a non-linear function of the input and output of the model taken at previous time-instants. In mathematical form, a NARX system can be written as

$$y\left(t_i\right) = \breve{g}\left(u\left(t_i - 1\right), \cdots, u\left(t_{i-n_u}\right), y\left(t_i - 1\right), \cdots, y\left(t_{i-n_y}\right)\right) + e\left(t_i\right) \tag{2.68}$$

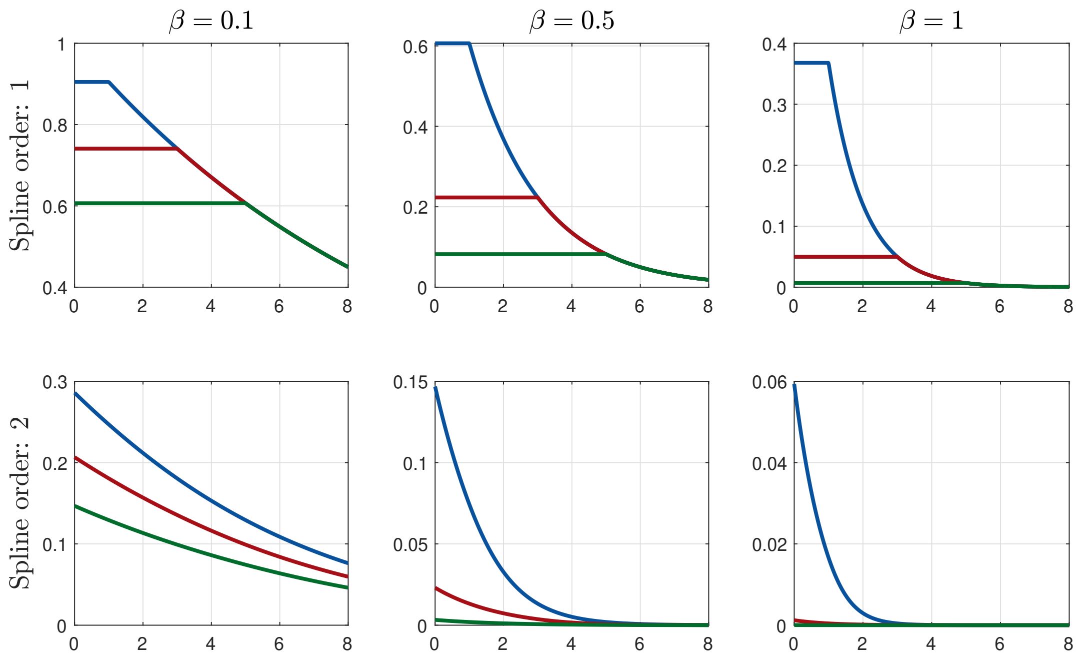

FIGURE 2.4: Plot of three representer functions of the continuous stable-spline kernel with different values of β. The parameter λ is set to 1.

where

- $u : \mathbb{R} \to \mathbb{R}$ is the input signal;
- $y : \mathbb{R} \to \mathbb{R}$ is the output signal;
- $t_i = i \cdot T_s$, with $i \in \mathbb{Z}$, are the time instants selected by the sampling process and $T_s \in \mathbb{R}_+$ is the sampling period;
- n_y is the autoregressive order;
- n_u is the exogenous order;
- $\breve{g} : \mathbb{R}^{n_x \times 1} \to \mathbb{R}$, with $n_x = n_u + n_y$, is a function that describes the model behavior;
- $e : \mathbb{R} \to \mathbb{R}$ is the noise term.

The samples of the noise are considered IID and the sampling period is considered to be known. For compactness sake, as in Section 2.1, the i-th sample of the input, output and noise signal are indicated, respectively, with $u_i = u\left(t_i\right)$, $y_i = y\left(t_i\right)$ and $e_i = e\left(t_i\right)$. Now, the recursive equation 2.68 can be written as

$$y_i = \breve{g}\left(\boldsymbol{x}_i\right) + e_i \tag{2.69}$$

where

$$\boldsymbol{x}_i = \begin{bmatrix} u_{i-1} & \cdots & u_{i-n_u} & y_{i-1} & \cdots & y_{i-n_y} \end{bmatrix}^\top \in \mathbb{R}^{n_x \times 1} \tag{2.70}$$

where $\boldsymbol{x}_i \in \mathbb{R}^{n_x \times 1}$ is the i-th regressor and $n_x = n_u + n_y$ is the regressor length. The function $\breve{g}$ characterize the behavior of the system and it is considered unknown in the identification problem. The orders n_u and n_y are typically unknown, however, let us first consider the case where these values are known.

Suppose, now, to have the first n input-output couples, i.e.

$$\mathcal{D} = \{(u_i, y_i) \,|i = 1, \dots, n\} \tag{2.71}$$

and to want to employ the PEM approach for the identification of the model $\breve{g}$. Then the estimation $\hat{g}$ is obtained by solving the optimization problem

$$\hat{g} = \arg\min_{g \in \mathcal{H}} \left\{ \sum_{i=1}^{n} (y_i - g(\boldsymbol{x}_i))^2 \right\} \tag{2.72}$$

where $\mathcal{H}$ is a certain hypothesis set.

Remark 2.9. To compute the predictor at the i-th time instant, it is necessary to know the measurements of y for some past time instants. For this reason, the summation in the optimization problem has to be restricted to the cases where the predictor can be computed with the data at hand. Otherwise, it is possible to make some assumption on the state of the system output before the start of the experiment.

This optimization problem strongly depends on the type of hypothesis set used. A common approach is to parametrize the function g with a finite number of parameters in order to obtain a finite-dimensional optimization problem. This can be achieved by using, for example, wavelets [133] or neural network [28, 92]. However, these approaches create non-convex cost functions that are difficult to minimize efficiently. To solve this problem, some researchers propose to use a linear-in-the-parameters parametrization [99]. However, this approach requires a large number of parameters and very complex models. To solve this problem, these linear-in-the-parameters parametrizations are often equipped with a LASSO regularizer [20]. Some researchers have also advocated for the use of the Simulation Error Method (SEM) [20, 99, 100] approach instead of the PEM one in order to obtain a more robust non-linear model. However, in this thesis, the focus will be on the PEM approach because it provides simpler cost functions that allow a more efficient minimization procedure.

2.3.1 Kernel method

More recently, the idea to use an RKHS as a hypothesis set was explored [6, 86, 97]. In these settings, the idea is to use a large amount, potentially infinite, of features and to resolve the optimization problem using the representer theorem. Following this rationale, let us assume that the hypothesis space $\mathcal{H}$ is an RKHS with kernel k and that the cost function becomes

$$\hat{g} = \arg\min_{g \in \mathcal{H}} \left\{ \sum_{i=1}^{n} (y_i - g(\boldsymbol{x}_i))^2 + \tau \|g\|_{\mathcal{H}}^2 \right\} \tag{2.73}$$

where the Tikhonov regularizer is added in order to tune the complexity of the solution.

Then, it is possible to apply the standard representer theorem reported in Theorem 1.7 in order to boil down the number of parameters to a finite number. In particular, the optimizer $\hat{g}$ can be written in the form

$$\hat{g} = \sum_{i=1}^{n} \hat{c}_i r_{\boldsymbol{x}_i} \tag{2.74}$$

where $r_{\boldsymbol{x}}$ is the representer of the regressor $\boldsymbol{x}$, as defined in 1.2. Following the same reasoning as in the static case described in Section 1.2.2, we obtain that

$$(\boldsymbol{K} + \tau \boldsymbol{I}_n)\, \hat{\boldsymbol{c}} = \boldsymbol{y}^\top \tag{2.75}$$

$$\hat{\boldsymbol{c}} = \left(\boldsymbol{K} + \tau \boldsymbol{I}_n\right)^{-1} \boldsymbol{y}^\top \tag{2.76}$$

where

$$\boldsymbol{y} = \begin{bmatrix} y_1 & \cdots & y_n \end{bmatrix} \in \mathbb{R}^{1\times n} \tag{2.77}$$

$$\boldsymbol{K} = \begin{bmatrix} k\left(\boldsymbol{x}_1, \boldsymbol{x}_1\right) & \cdots & k\left(\boldsymbol{x}_1, \boldsymbol{x}_n\right) \\ \vdots & \ddots & \vdots \\ k\left(\boldsymbol{x}_n, \boldsymbol{x}_1\right) & \cdots & k\left(\boldsymbol{x}_n, \boldsymbol{x}_n\right) \end{bmatrix} \in \mathbb{R}^{n\times n} \tag{2.78}$$

2.3.2 Kernel and order selection

As in the linear case, the choice of the right kernel is key in the performance of this method. However, assessing the stability of a non-linear system is difficult and the problem of defining a kernel that guarantees some sort of stability on the system is, according to the author knowledge, still unsolved. However, it is possible to encode in the kernel some of the important properties that the function g has to have. In particular, it is known that a stable dynamical system has a fading memory. In details, the dependency of the output y_i on the input-output samples (u_j, y_j) decreases as $|i - j|$ increases.

The simplest way to achieve this fading memory is to tune the memory of the system by modifying the orders n_u and n_y. In this case, the output will strongly depend on the closest n_u input samples and to the closest n_y output samples while it will not depends on the other samples. Here, the orders n_u and n_y are treated as an hyper-parameters of the kernel that has to be tuned.

In these settings, it is necessary to use a kernel that can works with different regressor lengths because they are an hyper-parameters. For this reason, a reasonable choice is the Gaussian kernel

$$k\left(\boldsymbol{x}_a, \boldsymbol{x}_b\right) = \lambda_{nl} e^{-\frac{\|\boldsymbol{x}_a - \boldsymbol{x}_b\|_2^2}{\sigma^2}} \tag{2.79}$$

where $\sigma > 0$ and $\lambda_{nl} > 0$ are two hyper-parameters to tune. In this kernel, the two parameters can have arbitrary length because the 2-norm is defined for every finite regressors length. Usually, this kernel is used in combination with the linear kernel because it is known that the RKHS defined by the Gaussian kernel does not contain linear functions. Therefore, it is convenient to enrich the kernel with a linear one

$$k\left(\boldsymbol{x}_a, \boldsymbol{x}_b\right) = \lambda_{nl} e^{-\frac{\|\boldsymbol{x}_a - \boldsymbol{x}_b\|_2^2}{\sigma^2}} + \lambda_l \boldsymbol{x}_a^\top \boldsymbol{x}_b + \lambda_c \tag{2.80}$$

where $\lambda_{nl} > 0$, $\lambda_l > 0$ and $\lambda_c > 0$ are the strength of, respectively, the Gaussian part, the linear part and constant component.

A second approach is to define a kernel that works on very long regressors, i.e. with large n_u and n_y, but that weights each sample differently based on its position inside the regressor itself. In this way, it is possible to increase the importance of the closest samples and decrease the one for the furthest away measurements. Ideally, we would want to use an infinite long regressor with a dependency that decreases exponentially to zero with the time difference. This approach was explored in [97] where the author defines and characterize a new kernel that employs this idea. This kernel is defined for the case where $n_u = n_y = m$

and it can be computed as:

$$k\left(\boldsymbol{x}_a, \boldsymbol{x}_b\right) = \lambda_{nl} \sum_{t=1}^{m-p+1} e^{-\beta_{nl} t} e^{-\frac{d_t(a,b)}{\sigma^2}} \tag{2.81}$$

where

$$d_i\left(a, b\right) = \sum_{j=0}^{p-1} \left(u_{a-i-j} - u_{b-i-j}\right)^2 + \left(y_{a-i-j} - y_{b-i-j}\right)^2 \tag{2.82}$$

while $\lambda_{nl} \in \mathbb{R}_+$, $\beta_{nl} \in \mathbb{R}_+$, $\sigma \in \mathbb{R}_+$ and $1 \leq p \leq m$ are hyper-parameters to tune.

To understand how this kernel behaves, consider first the simpler case when $p = 1$. Here, the kernel boils down to

$$k\left(\boldsymbol{x}_a, \boldsymbol{x}_b\right) = \lambda_{nl} \sum_{i=1}^{m} e^{-\beta_{nl} i} e^{-\frac{\left(u_{a-i} - u_{b-i}\right)^2 + \left(y_{a-i} - y_{b-i}\right)^2}{\sigma^2}} \tag{2.83}$$

now, it is clear that this kernel is a weighted sum of Gaussian kernels. Using the sum of kernel theorem, reported in Theorem 1.5, we can assess that the space $\mathcal{H}$ defined by k contains functions composed by a weighted sum. In particular, if $g \in \mathcal{H}$ than

$$g\left(\boldsymbol{x}_a\right) = \lambda_{nl} \sum_{i=1}^{m} e^{-\beta_{nl} i} g_i\left(u_{a-i}, y_{a-i}\right) \tag{2.84}$$

where g_i, with $i = 1, \ldots, m$, are functions that belongs to the space defined by a Gaussian kernel with width σ. Therefore, the input-output samples closest in time with x are weighted more than the one taken further in time. Additionally, these weights decrease exponentially with time creating the desired fading memory.

However, with $p = 1$, there is no non-linear relation between samples taken at different time instants. To solve this problem we can increase the parameter p that controls the number of samples fed to the Gaussian kernel. In particular, if the function g is an element of the space defined by k then

$$g\left(\boldsymbol{x}_a\right) = \lambda_{nl} \sum_{i=1}^{m} e^{-\beta_{nl} i} g_i\left(u_{a-i}, \ldots, u_{a-i-p+1}, y_{a-i}, \ldots, y_{a-i-p+1}\right) \tag{2.85}$$

Therefore, the same reasoning employed when $p = 1$ still holds, but now there are non-linear relations between samples taken at different time instants.

This kernel can model the non-linearities of the system, but it fails to see the eventual linear components [97]. Therefore, it is useful to add a second kernel that can model the linearities of the system. Since we are employing a large regressor, it is convenient to add the exponentially decaying weights also to the linear part in order to consider the fading memory of the system. For this reason, the kernel becomes:

$$\begin{aligned} k\left(\boldsymbol{x}_a, \boldsymbol{x}_b\right) = \lambda_{nl} & \sum_{i=1}^{m-p+1} e^{-\beta_{nl} i} e^{-\frac{d_t(a,b)}{\sigma^2}} + \\ & + \lambda_{lu} \sum_{i=1}^{m} e^{-\beta_{lu} i} u_{a-t} u_{b-t} + \\ & + \lambda_{ly} \sum_{i=1}^{m} e^{-\beta_{ly} i} y_{a-t} y_{b-t} \end{aligned} \tag{2.86}$$

where $\lambda_{nl} \in \mathbb{R}_+$, $\lambda_{lu} \in \mathbb{R}_+$ and $\lambda_{ly} \in \mathbb{R}_+$ are, respectively, the strength of the non-linear part, the linearity with respect of the past inputs and the linearities with respect of the past outputs while $\beta_{nl} \in \mathbb{R}_+$, $\beta_{lu} \in \mathbb{R}_+$ and $\beta_{ly} \in \mathbb{R}_+$ are, respectively, the rate of the exponentially decaying weights of the non-linear part, the linearity with respect of the past inputs and the linearities with respect of the past outputs.

For a more formal characterization of this kernel refer to [97].

PART II

Contributions and new research

CHAPTER 3

Computational remarks for the implementation of kernel methods

As seen in Section 1.2.2, the solution of the kernel-based regression problem is carried out by solving a linear system. However, in many practical applications, the matrix that has to be inverted is singular for computational reasons. In this Chapter, this behavior is explored in detail.

In particular, it is shown that there infinite many possible solutions of the linear system and that they all correspond to the same estimated function. For this reason, this apparent problem becomes an opportunity to select a coefficient vector that has additional useful properties not strictly coupled with the out-of-sample performance of the estimation. For example, we will show an algorithm that selects the solution that minimizes the computational complexity of the estimated model, i.e. the one that has the least number of non-zero elements.

Additionally, it is possible to tackle the problems that arise from the ill-conditioning of the semi-supervised learning using the manifold regularizer with a small to none Tikhonov regularization (see Section 1.4 for more details). In particular, it is proposed an algorithm that selects one of the equivalent solutions reliably even in these conditions.

The remainder of the Chapter is organized as follow:

- Section 3.1 briefly recalls the RKHS regression problem with the Tikhonov and the manifold regularizers and the singularity of the matrix, that has to be inverted, is shown and motivated;
- in Section 3.2 a detailed analysis of that RKHS regression with a singular kernel matrix is presented;
- in Section 3.3 the algorithm for the selection of the solution that minimizes the computational complexity of the estimated model is described and analyzed;
- Section 3.4 describes the algorithm that deals with the ill-conditioning of the semi-supervised manifold regression;
- Section 3.5 ends the chapter with some concluding remarks;
- Section 4.10 contains the proofs of all the presented theorems.

3.1 Background and motivation

Consider the dataset

$$\mathcal{D} = \{(\boldsymbol{x}_i, y_i) \,|1 \leq i \leq n\}, \tag{3.1}$$

sampled from the generic probabilistic model

$$y_i = g(\boldsymbol{x}_i) + e_i \tag{3.2}$$

where e_i are IID noises with variance β^2, $\boldsymbol{x}_i \in \mathcal{X} \subseteq \mathbb{R}^{n_x \times 1}$ are the regressors, $y_i \in \mathbb{R}$ denote the measurements and g is an unknown function.

Remark 3.1. This model corresponds to the one described in Section 1.2.2 for the identification of a static model. However, this formulation is general enough to comprehend the dynamical system case. In particular, when $\boldsymbol{x}$ is composed by the past input-output samples this formulation is equivalent at the one used in Section 2.1.4 for the non-linear system identification. Furthermore, it is trivial to change $g(\boldsymbol{x}_i)$ with the functional used for the identification of the impulse-response of a linear system, as shown in Section 2.1 and Section 2.2.

To make the notation more compact, we define the vectors:

$$\boldsymbol{y} = \begin{bmatrix} y_1 & \cdots & y_n \end{bmatrix} \in \mathbb{R}^{1\times n}, \tag{3.3}$$

$$\boldsymbol{g} = \begin{bmatrix} g(\boldsymbol{x}_1) & \cdots & g(\boldsymbol{x}_n) \end{bmatrix} \in \mathbb{R}^{1\times n}, \tag{3.4}$$

$$\boldsymbol{e} = \begin{bmatrix} e_1 & \cdots & e_n \end{bmatrix} \in \mathbb{R}^{1\times n} \tag{3.5}$$

and rewrite (3.2) as:

$$\boldsymbol{y} = \boldsymbol{g} + \boldsymbol{e}, \tag{3.6}$$

We consider the problem of finding from data an estimator $\hat{g}$ of the function g, by minimizing a regularized fitting cost, i.e.:

$$\hat{g} = \arg\min_{g\in\mathcal{H}} \{J(g)\} \tag{3.7}$$

$$J(g) = \|\boldsymbol{y} - \boldsymbol{g}\|_2^2 + \tau \|g\|_{\mathcal{H}}^2 + \mu \boldsymbol{g}\boldsymbol{M}\boldsymbol{g}^\top \tag{3.8}$$

where $\mathcal{H}$ is an RKHS [4, 109] with kernel $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ (see Section 1.1 for more details), τ and μ are non-negative scalars that define the strength of the regularizers, $\|\cdot\|_{\mathcal{H}}$ is the induced norm of the RKHS $\mathcal{H}$, and $\boldsymbol{M} \in \mathbb{R}^{n\times n}$ is symmetric and positive semi-definite.

The cost function J is composed by three terms:

- The first one is a quadratic loss;
- The second one is the classical Tikhonov regularizer [111, 131], as explained in Section 1.2.1;
- The third one is a intrinsic regularizer [10, 11] that employ a graph-based solution [10, 13, 14, 15, 23, 36, 49], as explained in Section 1.4.

Remark 3.2. More generally, in the semi-supervised case (see Section 1.4.3), the matrix $\boldsymbol{M}$ has dimension $r > n$ and the regularization term is $\bar{\boldsymbol{g}}\boldsymbol{M}\bar{\boldsymbol{g}}^\top$ where $\bar{\boldsymbol{g}} \in \mathbb{R}^{1\times r}$ is built by evaluating g in correspondence of a set of regressors with cardinality r. Here, we will keep the dimension of $\boldsymbol{M}$ equal to n (supervised case) to keep the mathematical notation compact. With similar reasoning, it is possible to generalize the discussion to the more generic case. More details are given in Section 3.4.

The Representer theorem [11, 40, 111] states that the minimizer of $J(g)$ can be written in the form:

$$\hat{g}(\boldsymbol{x}) = \hat{\boldsymbol{c}}^\top \cdot \boldsymbol{k}^*(\boldsymbol{x}), \tag{3.9}$$

where $\boldsymbol{k}^* : \mathcal{X} \to \mathbb{R}^{n\times 1}$ is a function such that

$$\boldsymbol{k}^*(\boldsymbol{x}) = \begin{bmatrix} k(\boldsymbol{x}_1, \boldsymbol{x}) & \cdots & k(\boldsymbol{x}_n, \boldsymbol{x}) \end{bmatrix}^\top \in \mathbb{R}^{n\times 1} \tag{3.10}$$

and the coefficients vector $\hat{\boldsymbol{c}} \in \mathbb{R}^{n\times 1}$ can be found by minimizing (3.8), which can be rewritten as a function of $\boldsymbol{c}$ as:

$$J_c(\boldsymbol{c}) = \left\| \boldsymbol{y}^\top - \boldsymbol{K}\boldsymbol{c} \right\|_2^2 + \boldsymbol{c}^\top (\tau\boldsymbol{K} + \mu\boldsymbol{K}\boldsymbol{M}\boldsymbol{K})\,\boldsymbol{c} \tag{3.11}$$

$$= \boldsymbol{c}^\top\boldsymbol{B}\boldsymbol{c} - 2\boldsymbol{c}^\top\boldsymbol{b} + \boldsymbol{y}\boldsymbol{y}^\top. \tag{3.12}$$

In the above equations, $\boldsymbol{K}$ is the kernel matrix whose (i, j) entry is $k(\boldsymbol{x}_i, \boldsymbol{x}_j)$ and

$$\boldsymbol{B} = \boldsymbol{K}\boldsymbol{A} \in \mathbb{R}^{n\times n} \tag{3.13}$$

$$\boldsymbol{A} = \boldsymbol{K} + \tau\boldsymbol{I}_n + \mu\boldsymbol{M}\boldsymbol{K} \in \mathbb{R}^{n\times n} \tag{3.14}$$

$$\boldsymbol{b} = \boldsymbol{K}\boldsymbol{y}^\top \in \mathbb{R}^{n\times 1} \tag{3.15}$$

Remark 3.3. Note that $\boldsymbol{B}$ is symmetric and positive semi-definite because it is defined as a multiplication of symmetric positive semi-definite matrices.

Since J_c is a quadratic function, its stationary points can be computed analytically as the solution of

$$\boldsymbol{B}\boldsymbol{c} = \boldsymbol{b}. \tag{3.16}$$

This is a linear system with n variables and n equations, thus it has a unique solution if and only if $\boldsymbol{B}$ is non-singular.

Proposition 3.1. *The rank of the matrix $\boldsymbol{B}$ is equal to the rank of the matrix $\boldsymbol{K}$ for every non-negative values of τ and μ.*

Proof. See Section 3.6 on page 79. ■

In order to have a full-rank $\boldsymbol{B}$ and a unique solution of (3.16), the kernel matrix $\boldsymbol{K}$ must be non-singular. If k is a non-degenerate kernel, this is always the case, because the eigenvalues of the matrix $\boldsymbol{K}$ are an approximation of the eigenvalues of the kernel function k [104]. In practice, however, this may not be the case. In fact, the eigenvalues of the kernel function k tend to zero with a rate that depends on the kernel shape and the regressors distribution [104]. Therefore, since the kernel matrix $\boldsymbol{K}$ is computed with limited machine precision, the lowest eigenvalues may become practically zero. This situation is illustrated in Example 3.1, which further motivates the presented work.

Example 3.1: Example with a Gaussian kernel

Consider the function $g : [-5, 5] \to \mathbb{R}$ defined as:

$$g(x) = 0.5\cos(3x) + 0.3x^2\sin(x) + x + 0.2x^2. \tag{3.17}$$

The available dataset $\mathcal{D} = \{(\boldsymbol{x}_i, y_i) \,|1 \leq i \leq n\}$ is obtained from (3.2) using $x_i \sim \mathcal{U}(-5, 5)$ and $e_i \sim \mathcal{N}(0, 1)$. The problem is tackled using the Gaussian kernel

$$k(a, b) = \exp[-(a-b)^2] \tag{3.18}$$

and with $\boldsymbol{M}$ equal to the Laplacian matrix, constructed as in [11], with a fully connected regressor graph that connects the nodes i and j with a edge weighted

$$w_{i,j} = \exp\left[-0.01n^2 (x_i - x_j)^2\right]. \tag{3.19}$$

The regularization strengths are set to $\tau = \mu = 0.05$.
In Figure 3.1 (left plot), the median of the first 80 eigenvalues of $\boldsymbol{K}$ over 100 Monte Carlo runs (with different regressors) are plotted for the case of $n = 500$. At first sight, it is clear that many of them become practically zero due to numerical precision. As a consequence, the rank of $\boldsymbol{K}$ becomes lower than n, as shown in Figure 3.1 (right plot). In the same figures, both single and double precision computations are plotted, to show that the above problem is actually due to numerical issues and is more evident when more limiting numerical accuracies are employed.

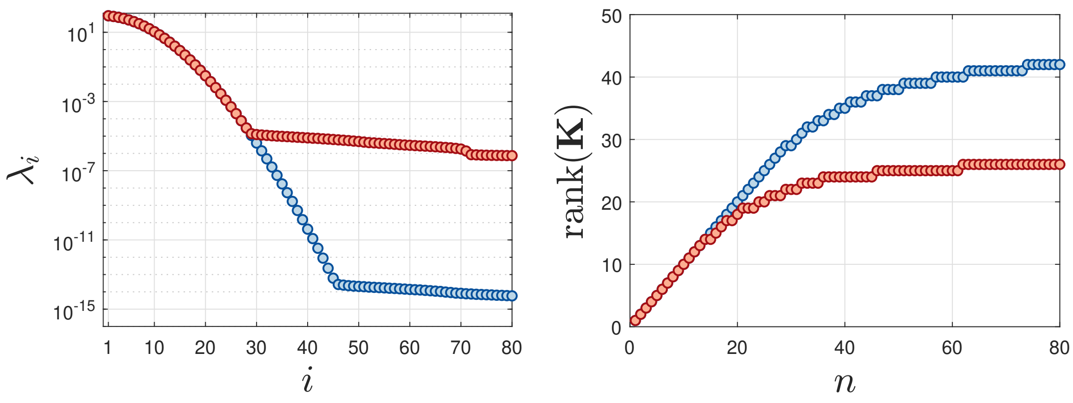

FIGURE 3.1: Median eigenvalues of $\boldsymbol{K}$ over 100 Monte Carlo runs using different regressors (left) and the corresponding median rank of $\boldsymbol{K}$ (right) for $n = 500$ in Example 3.1. Red circles: single precision, blue circles: double precision.

This phenomenon can be explained considering that, when the number of data n increases, the Representer theorem enlarges the number of features. However, they tend to become very similar to each other and, therefore, practically redundant. This problem could be attenuated with better precision that, however, cannot be selected arbitrarily in real-world applications.

3.2 Kernel-based learning with a singular kernel matrix

Consider the generic kernel $\tilde{k}$ with the Mercer expansion (see Section 1.1.2 for more details):

$$\tilde{k}(\boldsymbol{a}, \boldsymbol{b}) = \sum_{i=1}^{\infty} \sigma_i^2 \varphi_i(\boldsymbol{a}) \varphi_i(\boldsymbol{b}), \tag{3.20}$$

where σ_i^2 are the eigenvalues of the kernel and $\varphi_i \in \mathcal{H}_{\tilde{k}}$ are its eigenfunctions. Following the observations of the previous section, we will now consider the case where only the first m eigenvalues are different from zero. The actual kernel k has the truncated Mercer expansion:

$$k(\boldsymbol{a}, \boldsymbol{b}) = \sum_{i=1}^{m} \sigma_i^2 \varphi_i(\boldsymbol{a}) \varphi_i(\boldsymbol{b}) . \tag{3.21}$$

This is a degenerate kernel with m non-zero eigenvalues. The value of m depends on the numerical precision, the kernel type, and the regressors distribution. Assuming (reasonably) that $m < n$, by using k instead of $\tilde{k}$, the kernel matrix $\boldsymbol{K}$ has rank m and it is singular.

Since $\boldsymbol{K}$ is symmetric, $\boldsymbol{K} = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^\top$, where $\boldsymbol{\Lambda} \in \mathbb{R}^{m \times m}$ is a diagonal matrix built with the strictly positive eigenvalues of $\boldsymbol{K}$ and the columns of $\boldsymbol{U} \in \mathbb{R}^{n \times m}$ are the corresponding eigenvectors [52]. Recall that $\boldsymbol{U}^\top\boldsymbol{U} = \boldsymbol{I}_m$, $\boldsymbol{\Lambda}$ is invertible and $\operatorname{rank}(\boldsymbol{U}) = m$. The following theorem holds.

Theorem 3.1 (Degenerate solution)**.** *Let*

$$\mathcal{S}_c = \left\{\boldsymbol{c} \in \mathbb{R}^{n \times 1} \text{ s.t. } \boldsymbol{B}\boldsymbol{c} = \boldsymbol{b}\right\} \tag{3.22}$$

be the solution set of the linear system (3.16) *and* $\mathcal{N}\left(\boldsymbol{U}^\top\right)$ *be the null space of* $\boldsymbol{U}^\top$. *Then:*

a) $\mathcal{S}_c \neq \varnothing$, *i.e. the linear system* (3.16) *is consistent;*

b) $\mathcal{S}_c$ *is an affine space with dimension* $n - m$;

c) *given* $\boldsymbol{c}_1, \boldsymbol{c}_2 \in \mathcal{S}_c$, $\boldsymbol{c}_1 - \boldsymbol{c}_2 \in \mathcal{N}\left(U^\top\right)$;

d) *given* $\boldsymbol{c} \in \mathcal{S}_c$, $\boldsymbol{c}$ *is a global minimum of* J_c.

Proof. See Section 3.6 on page 80. ■

Theorem 3.1 allows us to better understand what happens when the kernel matrix becomes rank deficient. Specifically, the cost function J_c becomes a degenerate paraboloid with infinite minima lying on a subspace of dimension $n - m$, as it is shown in point **b)**, and $\mathcal{S}_c$ becomes the set containing all the global minima of J_c, as it is shown in point **d)**.

Remark 3.4. When $n = m$, it is possible to note that this theorem still holds and that the solution space $\mathcal{S}_c$ has dimension $n - m = 0$. Therefore, in this particular case, the solution is unique. This is the case that is normally considered in the literature [17, 44, 104] where the matrix $\boldsymbol{K}$ is considered full-rank and therefore invertible.

Recalling that the estimated function $\hat{g}$ can be written as in (3.9) for the Representer theorem, it is possible to assign an estimated function to each element of the solution set $\mathcal{S}_c$. Given the solution $\boldsymbol{c} \in \mathcal{S}_c$ the associated estimated function will be denoted as $\hat{g}_{\boldsymbol{c}}$.

Theorem 3.2 (Equivalent solutions)**.** *Let* $\boldsymbol{c}_1, \boldsymbol{c}_2 \in \mathcal{S}_c$ *and* $\boldsymbol{x} \in \mathcal{X}$. *Then* $\hat{g}_{\boldsymbol{c}_1}(\boldsymbol{x}) = \hat{g}_{\boldsymbol{c}_2}(\boldsymbol{x})$.

Proof. See Section 3.6 on page 82. ■

From the above theorem, it turns out that different solutions can be equivalently selected. Typically, the matrix $\boldsymbol{K}$ is considered invertible, i.e. $n = m$. In this case, the solution is unique and can be computed by solving the linear system

$$\boldsymbol{A}\boldsymbol{c}_T = \boldsymbol{y}^\top \tag{3.23}$$

$$\boldsymbol{c}_T = \boldsymbol{A}^{-1}\boldsymbol{y}^\top \tag{3.24}$$

therefore, the most common way to compute the solution vector is to solve the linear system (3.23) [17, 44, 104], usually, using the Cholesky decomposition of $\boldsymbol{A}$ [52, 104]. This is a valid approach, even when $\boldsymbol{K}$ is singular because

$$\boldsymbol{B}\boldsymbol{c}_T = \boldsymbol{K}\boldsymbol{A}\boldsymbol{A}^{-1}\boldsymbol{y}^\top = \boldsymbol{B}\boldsymbol{y}^\top = \boldsymbol{b}, \tag{3.25}$$

This solution $\boldsymbol{c}_T = \boldsymbol{A}^{-1}\boldsymbol{y}^\top$, that will be called *trivial solution* from now on, does not have any special property, but it is straightforward to compute. Therefore, if the computational time needed to find the solution is the most important aspect for the considered application, solving the linear system (3.23) to find the trivial solution remains a suitable approach even when $\boldsymbol{K}$ is not-invertible.

However, this solution exists only when the matrix $\boldsymbol{A}$ is invertible and when dealing with the manifold regularization this is not always the case. Furthermore, when the matrix $\boldsymbol{A}$ is ill-conditioned, like in the case where the manifold regularizer is the predominant one [11], this approach is ill-conditioned.

In the next two sections, two non-trivial solutions with different properties are described.

3.3 A sparse equivalent solution

In general, kernel methods produce an estimated function that can be written as the sum of n features, as shown in (3.9) or in more details in section 1.2.2. With large n, this kind of models can be computationally expensive and they require the entire training dataset to be stored.

To solve this problem, consider that (3.9) can be rewritten as

$$\hat{g}_{\boldsymbol{c}}(\boldsymbol{x}) = \boldsymbol{c}^\top \boldsymbol{k}_*(\boldsymbol{x}) = \sum_{i \in \mathcal{I}} c_i k(\boldsymbol{x}_i, \boldsymbol{x}) \tag{3.26}$$

where $c_i \in \mathbb{R}$ is the i-th element of the vector $\boldsymbol{c}$ and $\mathcal{I} = \{1 \leq i \leq n \text{ s.t. } c_i \neq 0\}$ is the set of the indexes where the vector $\boldsymbol{c}$ is non-zero. It is possible to exploit the additional freedom coming from the singularity of $\boldsymbol{K}$ to force many entries of $\boldsymbol{c}$ to zero and decrease the computational complexity of the estimated model. Furthermore, it is necessary to store only the part of the training dataset with the indexes inside the set $\mathcal{I}$.

To tackle this problem consider the Complete Orthogonal Decomposition (COD) [52] of the matrix $\boldsymbol{B}$. This decomposition searches the quadruple $(\boldsymbol{Q}, \boldsymbol{R}, \boldsymbol{H}, r)$ such that:

a) $r = \operatorname{rank}(\boldsymbol{B})$;

b) $\boldsymbol{Q}, \boldsymbol{H} \in \mathbb{R}^{n \times r}$ are orthogonal matrices, i.e. $\boldsymbol{Q}^\top\boldsymbol{Q} = \boldsymbol{I}_r$ and $\boldsymbol{H}^\top\boldsymbol{H} = \boldsymbol{I}_r$;

c) $\boldsymbol{R} \in \mathbb{R}^{r \times r}$ is a upper triangular matrix;

d) $\boldsymbol{B} = \boldsymbol{Q}\boldsymbol{R}\boldsymbol{H}^\top$;

The algorithm that computes this decomposition requires the tuning of the tolerance parameter ε, so that all the eigenvalues smaller of ε are considered to 0. This threshold ε can be tuned using the characterization of the quantization noise level of the elements of the matrix [85] (i.e., machine precision). A common criterion for the selection of this threshold is [52]

$$\varepsilon = \delta\left(\|\boldsymbol{B}\|_\infty\right) \tag{3.27}$$

where δ is a function that returns the positive distance between its argument and the next larger floating point number with the same precision and $\|\boldsymbol{B}\|_\infty$ is the matrix ∞-norm of $\boldsymbol{B}$.

Remark 3.5. According to Proposition 3.1, we have that the rank r of $\boldsymbol{B}$ is equal to the rank m of $\boldsymbol{K}$.

Using this decomposition, the linear system (3.16) can be rewritten as

$$\boldsymbol{B}\boldsymbol{c} = \boldsymbol{b} \tag{3.28}$$

$$\boldsymbol{Q}\boldsymbol{R}\boldsymbol{H}^\top \boldsymbol{c} = \boldsymbol{b} \tag{3.29}$$

$$\boldsymbol{Q}^\top\boldsymbol{Q}\boldsymbol{R}\boldsymbol{H}^\top \boldsymbol{c} = \boldsymbol{Q}^\top\boldsymbol{b} \tag{3.30}$$

$$\boldsymbol{R}\boldsymbol{H}^\top \boldsymbol{c} = \boldsymbol{Q}^\top\boldsymbol{b} \tag{3.31}$$

this is a linear system with m equations and n variables and, therefore, it is underdetermined. Now, the objective is to find the solution with the larger number of 0 elements. In theory, this is achieved by solving the optimization problem

$$\boldsymbol{c}_{ln0} = \arg\min_{\boldsymbol{c}\in\mathbb{R}^{n\times 1}} \|\boldsymbol{c}\|_0 \tag{3.32}$$

$$\text{s.t. } \boldsymbol{R}\boldsymbol{H}^\top \boldsymbol{c} = \boldsymbol{Q}^\top\boldsymbol{b} \tag{3.33}$$

Remark 3.6. It is straightforward to note that a solution with at least $n-m$ entries equal to 0 always exists [21, 65].

It is a well-known fact that is not possible to compute $\boldsymbol{c}_{ln0}$ in polynomial time [21, 65, 84]. For this reason, in the literature there different approaches that try to compute a good approximation of $\boldsymbol{c}_{ln0}$. The most common ones are derivations of the minimization of the l_1 norm [22, 66]. Some alternatives relies on the greedy rationale [122, 124]. All these approaches can be used to solve the problem at hand, however, in this document, the classic l_1 minimization [65] is used.

$$\begin{aligned} \boldsymbol{c}_{ln1} &= \arg\min_{\boldsymbol{c}\in\mathbb{R}^{n\times 1}} \|\boldsymbol{c}\|_1 \\ &\text{s.t. } \boldsymbol{R}\boldsymbol{H}^\top \boldsymbol{c} = \boldsymbol{Q}^\top\boldsymbol{b} \end{aligned} \tag{3.34}$$

This is a linearly constrained convex optimization problem [21] that can be solved using an appropriate solver. In this document, YALMIP [73] equipped with CPLEX was used. This solution will be called *Least Norm 1 (LN1) solution* from now on.

Remark 3.7. The reasoning behind the optimization problem (3.34) is analogue to the one behind the LASSO regularization [17, 44]. However, the objective is different: here, we want to solve an underdetermined linear system of equations while the LASSO regularization is a regularizer that can be used to impose sparsity on the estimation of a large linear regression problem.

Example 3.2: Example of computation of the LN1 solution

Consider again the problem treated in Example 3.1. Since the rank of the matrix $\boldsymbol{K}$ may be small with respect of the number of data n, it is possible to find a solution $\boldsymbol{c}_{ln1}$ with several zero entries. This is confirmed by the results in Figure 3.2, illustrating the number of non-zero elements over 10^3 runs. The figure also shows that such a

number does not change significantly with n. This is reasonable, as the complexity of the estimated model should depend only on the system nature and not on the amount of data available.
In Figure 3.3, the computational efficiency of the LN1 solution $\boldsymbol{c}_{ln1}$ is plotted, as compared to the trivial one $\boldsymbol{c}_T$, when used for prediction on new points. As expected, the LN1 solution is much more efficient. This is true especially for high n, because the complexity of the LN1 model does not increase with the number of data, while the trivial model increases the computational time significantly. However, it is important to note that the solution LN1 is slower to compute especially with large datasets because it requires to solve the optimization problem (3.34), as shown in Figure 3.4.

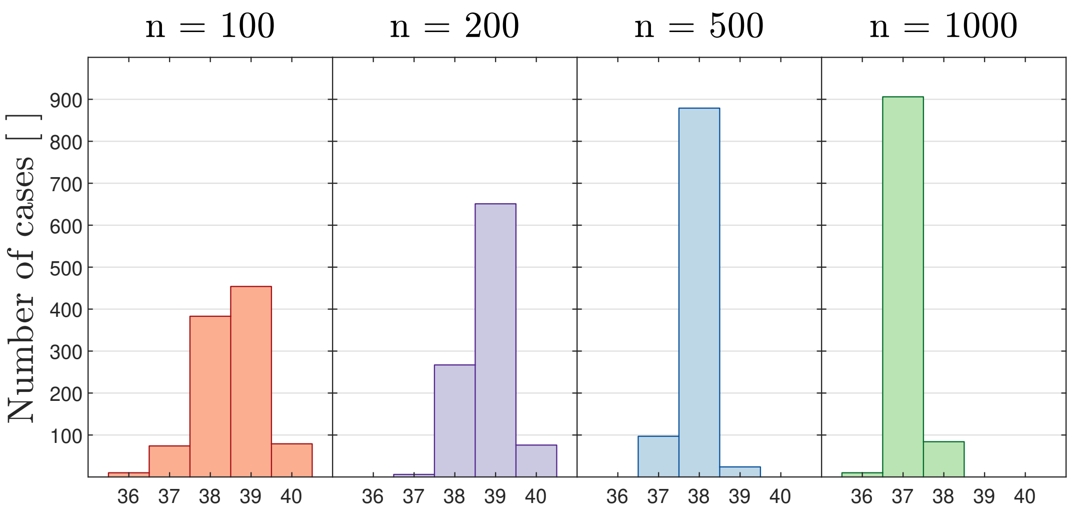

FIGURE 3.2: Number of non-zero entries in the solution $\boldsymbol{c}_{ln1}$ for 4 values of n and 10^3 realizations of the noise.

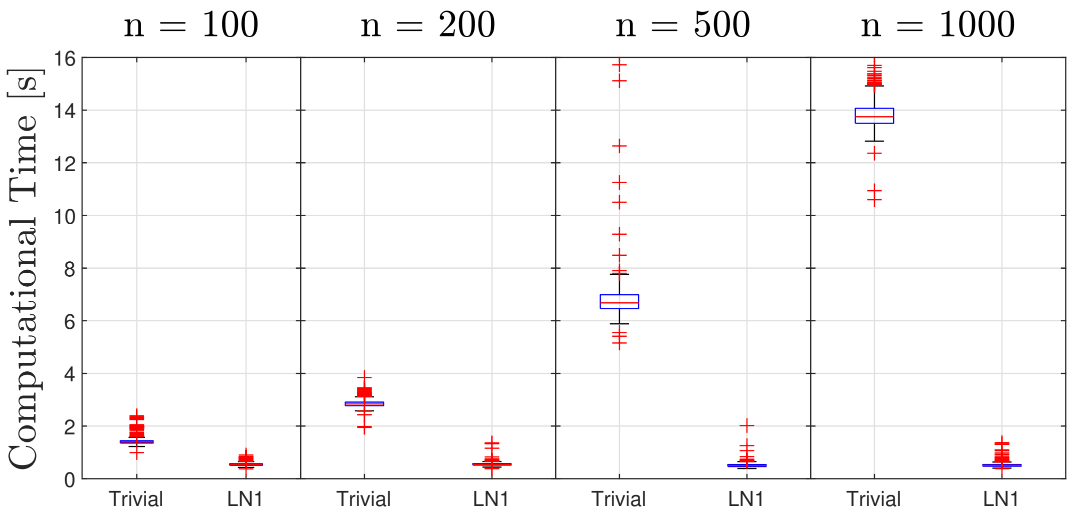

FIGURE 3.3: Computational time of the model on 5000 different validation points for the trivial solution $\boldsymbol{c}_T$ and the LN1 solution $\boldsymbol{c}_{ln1}$.

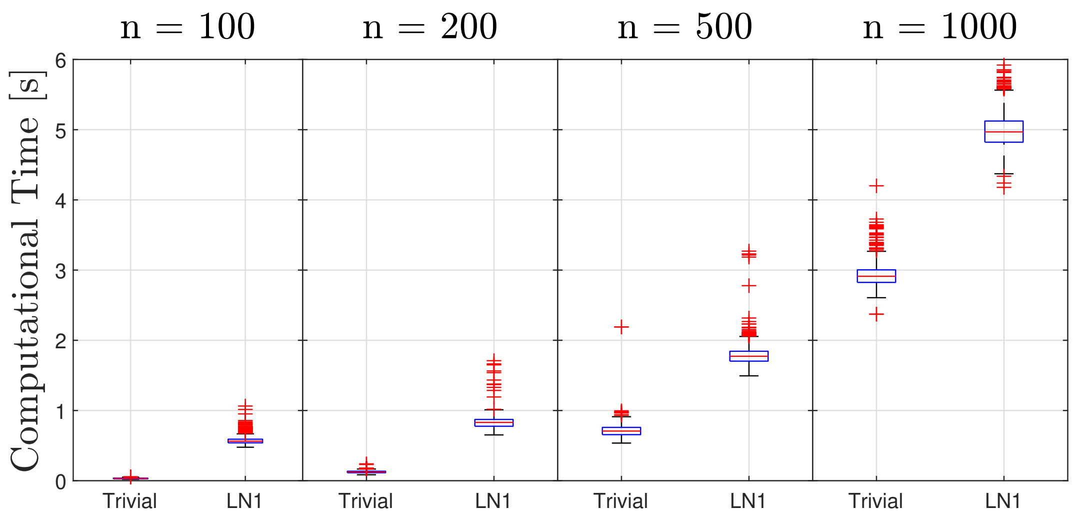

FIGURE 3.4: Computational time needed to compute the trivial and the LN1 solution for 10^3 different datasets.

3.4 A WELL-CONDITIONED SOLUTION FOR SEMI-SUPERVISED REGRESSION

In this section, we will consider again the case of degenerate kernels to select the solution most suited for semi-supervised regression.

In this setting, the measurements vector $\boldsymbol{y}$ is only partially known. In particular, we will assume that only $n_s \leq n$ regressors have a known associated measurement. Following the rationale described in Section 1.4.3 and in [11], the semi-supervised solution is given by:

$$\hat{g} = \arg\min_{g \in \mathcal{H}_k} \left\{ \overline{J}(g) \right\} \tag{3.35}$$

$$\overline{J}(g) = \|\boldsymbol{y}_s - \boldsymbol{g}_s\|_2^2 + \tau \|g\|_k^2 + \mu \boldsymbol{g}\boldsymbol{M}\boldsymbol{g}^\top \tag{3.36}$$

where $\boldsymbol{y}_s$ and $\boldsymbol{g}_s$ are, respectively, the parts of $\boldsymbol{y}$ and $\boldsymbol{g}$ associated with known measurements and $\boldsymbol{M} \in \mathbb{R}^{n\times n}$ is a positive semi-definite symmetric matrix [10, 11]. In particular, the third term in (3.36) is called *manifold regularization term* and penalizes the functions that are not smooth alongside the intrinsic regressors structure, as described in Section 1.4.3.

The Representer theorem holds also for the cost function (3.36) [11]. Therefore, $\hat{g}$ can be written as:

$$\hat{g}(\boldsymbol{x}) = \hat{\boldsymbol{c}}^\top \boldsymbol{k}_*(\boldsymbol{x}) \tag{3.37}$$

where $\hat{\boldsymbol{c}}$ can be found by minimizing the cost function:

$$\begin{aligned} \overline{J}_c(\boldsymbol{c}) &= \left\| \boldsymbol{y}_s^\top - \boldsymbol{P}\boldsymbol{K}\boldsymbol{c} \right\|_2^2 + \boldsymbol{c}^\top (\tau \boldsymbol{K} + \mu \boldsymbol{K}\boldsymbol{M}\boldsymbol{K})\, \boldsymbol{c} & (3.38)\\ &= \boldsymbol{c}^\top \overline{\boldsymbol{B}}\boldsymbol{c} - 2\boldsymbol{c}^\top \overline{\boldsymbol{b}} + \boldsymbol{y}\boldsymbol{y}^\top, & (3.39) \end{aligned}$$

where:

$$\overline{\boldsymbol{B}} = \boldsymbol{K}\boldsymbol{A} \in \mathbb{R}^{n\times n}, \tag{3.40}$$

$$\overline{\boldsymbol{A}} = \boldsymbol{P}\boldsymbol{K} + \tau \boldsymbol{I}_n + \mu \boldsymbol{M}\boldsymbol{K} \in \mathbb{R}^{n\times n}, \tag{3.41}$$

$$\overline{\boldsymbol{b}} = \boldsymbol{K} \begin{bmatrix} \boldsymbol{y}_s^\top \\ \boldsymbol{0}_{(n-n_s)\times 1} \end{bmatrix} \in \mathbb{R}^{n\times 1}, \tag{3.42}$$

$$\boldsymbol{P} = \begin{bmatrix} \boldsymbol{I}_{n_s} & \boldsymbol{0}_{n_s\times(n-n_s)} \\ \boldsymbol{0}_{(n-n_s)\times 1} & \boldsymbol{0}_{(n-n_s)\times(n-n_s)} \end{bmatrix} \in \mathbb{R}^{n\times n}. \tag{3.43}$$

Since $\overline{J}_c$ is a quadratic function, its stationary points can be computed analytically by solving the linear system:

$$\overline{\boldsymbol{B}}\boldsymbol{c} = \overline{\boldsymbol{b}}. \tag{3.44}$$

This a generalization of the problem treated in previous sections. In particular, with $n_s = n$ the matrix $\boldsymbol{P}$ becomes an identity matrix and (3.44) becomes equal to (3.16). For this reason Theorem 3.1 and Theorem 3.2 needs to be generalized at the semi-supervised case. This is achieved by the following theorems.

Theorem 3.3 (Degenerate solution - general case)**.** *Let*

$$\overline{\mathcal{S}}_c = \left\{\boldsymbol{c} \in \mathbb{R}^{n\times 1} | \overline{\boldsymbol{B}}\boldsymbol{c} = \overline{\boldsymbol{b}}\right\} \tag{3.45}$$

be the solution set of the linear system (3.44) *and* $\mathcal{N}\left(\boldsymbol{U}^\top\right)$ *be the null space of* $\boldsymbol{U}^\top$. *If* $\tau > 0$, *then*

a) $\overline{\mathcal{S}}_c \neq \varnothing$, *i.e. the linear system* (3.44) *is consistent;*

b) $\overline{\mathcal{S}}_c$ *is an affine space with dimension* $n - m$*;*

c) *given* $\boldsymbol{c}_1, \boldsymbol{c}_2 \in \overline{\mathcal{S}}_c$, $\boldsymbol{c}_1 - \boldsymbol{c}_2 \in \mathcal{N}\left(U^\top\right)$*;*

d) *given* $\boldsymbol{c} \in \overline{\mathcal{S}}_c$ *is a global minimum of* $\overline{J}_c$.

Proof. See Section 3.6 on page 81. ■

Theorem 3.4 (Equivalent solutions - general case)**.** *Let* $\boldsymbol{c}_1, \boldsymbol{c}_2 \in \overline{\mathcal{S}}_c$ *and* $x \in \mathcal{X}$. *If* $\tau > 0$, *then* $\hat{g}_{\boldsymbol{c}_1}(\boldsymbol{x}) = \hat{g}_{\boldsymbol{c}_2}(\boldsymbol{x})$.

Proof. See Section 3.6 on page 82. ■

Then, also in the semi-supervised framework, if the kernel is degenerate, there are infinite solutions with equivalent out-of-sample performance.

Now notice that, when τ is small, we have

$$\begin{aligned} \operatorname{rank}\left(\overline{\boldsymbol{A}}\right) &= \operatorname{rank}\left(\boldsymbol{P}\boldsymbol{K} + \mu\boldsymbol{M}\boldsymbol{K}\right) & (3.46)\\ &= \operatorname{rank}\left(\left(\boldsymbol{P}\boldsymbol{K} + \mu\boldsymbol{M}\right)\boldsymbol{K}\right) & (3.47)\\ &\leq \operatorname{rank}\left(\boldsymbol{K}\right). & (3.48) \end{aligned}$$

Therefore, $\overline{\boldsymbol{A}}$ may be singular and the trivial solution $\boldsymbol{c}_T = \overline{\boldsymbol{A}}^{-1}\overline{\boldsymbol{b}}$ may be ill-conditioned.

In this manuscript, we enhance the numerical conditioning of the solutions by relying on numerical algebra techniques. More specifically, we propose to replace the trivial solution with the Least Norm 2 (LN2) solution defined as:

$$\boldsymbol{c}_{ln2} = \underset{\boldsymbol{c}\in\mathbb{R}^{n\times 1}}{\arg\min}\left\{\|\boldsymbol{c}\|_2\right\} \tag{3.49}$$

$$\text{s.t. } \overline{\boldsymbol{B}}\boldsymbol{c} = \overline{\boldsymbol{b}} \tag{3.50}$$

This solution can be easily computed using the Complete Orthogonal Decomposition (COD) [52] of the matrix $\overline{\boldsymbol{B}}$. As shown in Section 3.3, we can rewrite the linear system (3.44) as

$$\overline{\boldsymbol{R}}\,\overline{\boldsymbol{H}}^\top \boldsymbol{c} = \overline{\boldsymbol{Q}}^\top \overline{\boldsymbol{b}} \tag{3.51}$$

where $\overline{\boldsymbol{R}} \in \mathbb{R}^{m\times m}$, $\overline{\boldsymbol{H}} \in \mathbb{R}^{n\times m}$ and $\overline{\boldsymbol{Q}} \in \mathbb{R}^{n\times m}$ are the three matrices described in Section 3.3 obtained using the Complete Orthogonal Decomposition (COD) on the matrix $\overline{\boldsymbol{B}}$. Then, it is possible to obtain the LN2 solution by following Algorithm 3.1.

Algorithm 3.1: Computation of the LN2 solution

```
Input: The matrix $\overline{\boldsymbol{B}}$
Input: The vector $\overline{\boldsymbol{b}}$
Input: The threshold $\varepsilon$, tuned as described in the previous section
1 Compute the quadruple $\left(\overline{\boldsymbol{Q}}, \overline{\boldsymbol{R}}, \overline{\boldsymbol{H}}, r\right)$ as the Complete Orthogonal Decomposition
    (COD) of $\overline{\boldsymbol{B}}$ using the tolerance $\varepsilon$ to determine the rank $r$ of the matrix $\overline{\boldsymbol{B}}$
2 Compute $\overline{\boldsymbol{c}} = \overline{\boldsymbol{Q}}^\top \overline{\boldsymbol{b}}$
3 Solve for $\overline{\boldsymbol{d}}$ the upper triangular linear system $\overline{\boldsymbol{R}}\,\overline{\boldsymbol{d}} = \overline{\boldsymbol{c}}$
4 Compute the LN2 solution as $\boldsymbol{c}_{ln2} = \overline{\boldsymbol{H}}\,\overline{\boldsymbol{d}}$
Output: The vector $\boldsymbol{c}_{ln2}$
```

The following illustrative example illustrates the effectiveness of the proposed LN2 solution as compared to the trivial one.

Example 3.3: Semi-supervised regression using the LN2 solution

Consider the estimation of the function $g : \mathbb{R}^2 \to \mathbb{R}$ such that

$$\begin{aligned} g\left(x_1, x_2\right) = 3c_1\left(x_1, x_2\right) e^{-2\left|-1+\sqrt{x_1^2+\left(x_2-\frac{1}{5}\right)^2}\right|} + \\ - 2c_2\left(x_1, x_2\right) e^{-\frac{10}{7}\left|-1+\sqrt{\left(x_1-1\right)^2+x_2^2}\right|} \end{aligned} \tag{3.52}$$

where:

$$c_1\left(x_1, x_2\right) = \begin{cases} 1 & -\frac{\pi}{5} \le \operatorname{arctan2}\left(x_2, x_1\right) \le \frac{6\pi}{5} \\ 0 & \text{otherwise} \end{cases} \tag{3.53}$$

$$c_2\left(x_1, x_2\right) = \begin{cases} 1 & \frac{4\pi}{5} \le \operatorname{arctan2}\left(x_2, x_1\right) \le \frac{11\pi}{5} \\ 0 & \text{otherwise} \end{cases} \tag{3.54}$$

using a dataset

$$\mathcal{D} = \left\{\left(\boldsymbol{x}_i, y_i\right) | 1 \le i \le n\right\}. \tag{3.55}$$

The regressors $\boldsymbol{x}_i$ are equal to:

$$x_{i,1} = 1 - p_i + \frac{(2p_i - 1)(10 + a_i)}{10} \cos\left((1 - 7b_i)\frac{\pi}{5}\right) \tag{3.56}$$

$$x_{i,2} = \frac{p_i}{5} - \frac{(2p_i - 1)(10 + a_i)}{10} \sin\left((1 - 7b_i)\frac{\pi}{5}\right) \tag{3.57}$$

where a_i, b_i and p_i are random variables distributed as

$$a_i \sim \mathcal{N}(0, 1) \tag{3.58}$$

$$b_i \sim \mathcal{U}(0, 1) \tag{3.59}$$

$$p_i \sim \text{Bernoulli}\left(\frac{1}{2}\right) \tag{3.60}$$

then the output are sampled according the model described in Equation (3.2) with noises e_i distributed as a normal distribution with 0 mean and variance η^2 chosen in order to have SNR equal to 1000.
The dataset contains $n = 500$ elements, but only $n_s = 12$ of them are supervised. In Figure 3.5, it is possible to see the function g and sample dataset generated according to the distribution. Notice that the considered system has a particular regressors distribution π with two distinct regions where the unknown function is continuous. For this reason, manifold regularization can be employed to enforce smoothness along the regions by using the unsupervised regressors.
The problem is tackled using the Gaussian kernel

$$k(\boldsymbol{a}, \boldsymbol{b}) = e^{-\beta\|\boldsymbol{a}-\boldsymbol{b}\|_2^2} \tag{3.61}$$

and $\boldsymbol{M}$ is equal to the Laplacian matrix, constructed as explained in [11], with a fully connected regressor graph that connects the nodes i and j with a edge weighted

$$w_{i,j} = e^{-\frac{2500}{9}\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2} \tag{3.62}$$

The hyper-parameters (β, τ, μ) are chosen via 3-fold cross-validation [44, 104], refer to Section 1.5.1. The performance of the estimated functions is tested on a noiseless dataset of 10^5 samples using the fit index

$$Fit = 1 - \frac{\sqrt{\sum_{i=1}^{10000} (\tilde{y}_i - \hat{g}(\tilde{\boldsymbol{x}}_i))^2}}{\sqrt{\sum_{i=1}^{10000} \left(\tilde{y}_i - \sum_{i=1}^{10000} \tilde{y}_i\right)^2}} \tag{3.63}$$

where $\hat{g}$ is the estimated function and

$$\mathcal{D}_V = \left\{ (\boldsymbol{x}_i, y_i) \middle| \, 1 \le i \le 10^5 \right\} \tag{3.64}$$

is the validation dataset.
To assess the statistical properties of the method, a Monte Carlo simulation with 100 different realizations of the noise is performed. In Figure 3.6, the performance of the solution generated with Algorithm 3.1 with threshold $\varepsilon = \delta\left(\|\boldsymbol{B}\|_\infty\right)$ is compared with the one obtained using the trivial solution. The proposed approach clearly

outperforms the one proposed in the literature. This is due to the fact that the optimization procedure that searches the hyper-parameters converges to different (better conditioned) combinations of (β, τ, μ).
The slight increase of performance can be further assessed by looking at the residue

$$\text{res}(\boldsymbol{c}) = \left\| \overline{\boldsymbol{B}}\boldsymbol{c} - \overline{\boldsymbol{b}} \right\|_2 \tag{3.65}$$

reported in Figure 3.7.
Finally, it turns out that the optimization procedure to find the hyperparameters for the LN2 solution is much faster, see Figure 3.8. The latter observation is only empirical and will be the object of further analysis. However, it definitely encourages additional research on this topic.

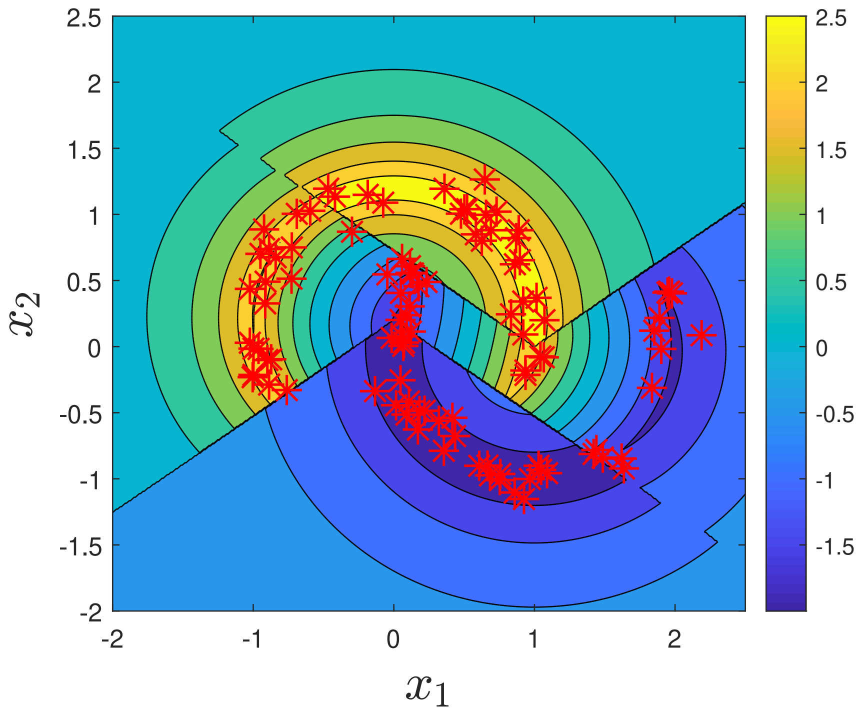

FIGURE 3.5: Function g of Example 3.3 (background color) and 100 regressor samples (red asterisks).

3.5 Chapter concluding remarks

In many practical applications, kernel-based learning problems have to be solved by relying on rank-deficient kernel matrices. This fact is usually ignored because it is possible to find a solution even if the kernel matrix is very ill-conditioned.

This work delves into the reasons behind this fact and analyzes the opportunities that arise from this apparent issue. In particular, it is shown that there are multiple equivalent solutions in term of out-of-sample performance and that it is possible to select one of them to optimize some additional criteria. In particular, in this manuscript, we discuss the possibility of computing an equivalent sparse solution and a numerically better-conditioned solution for semi-supervised regression.

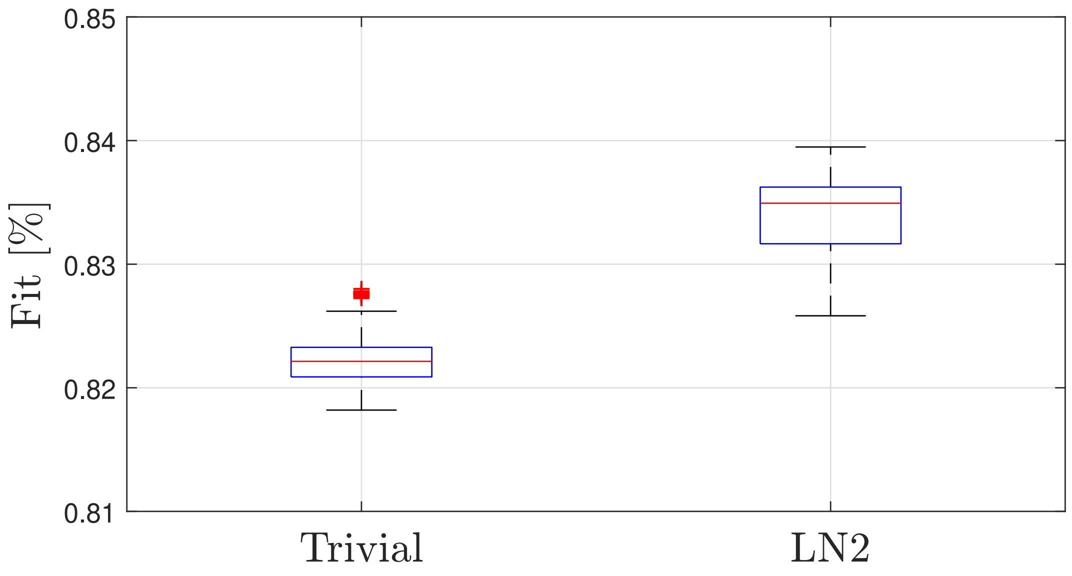

FIGURE 3.6: Fit of trivial and LN2 solutions for Example 3.3.

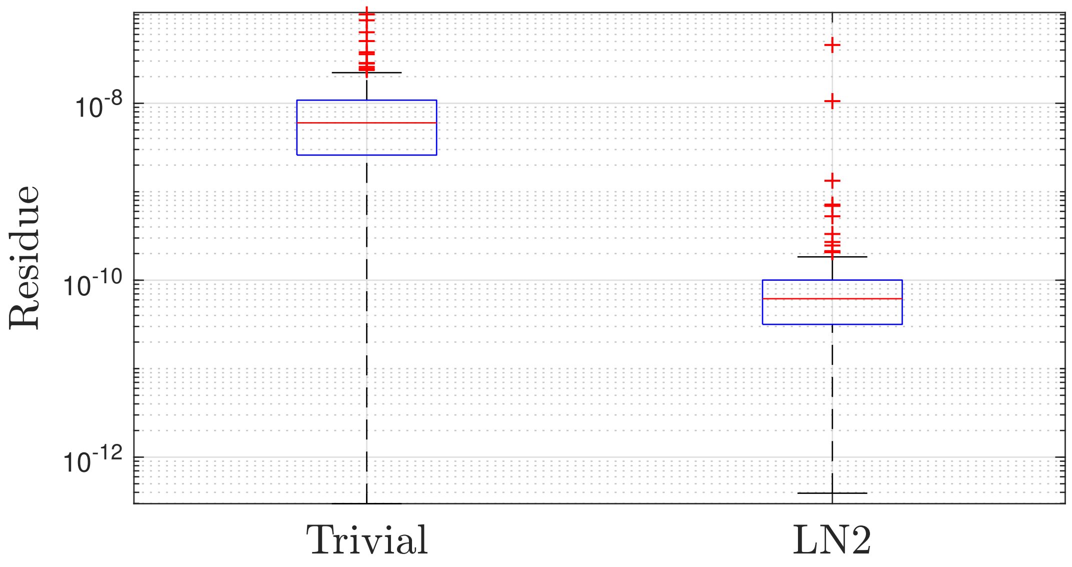

FIGURE 3.7: Residue (3.65), of the trivial and LN2 solutions for Example 3.3.

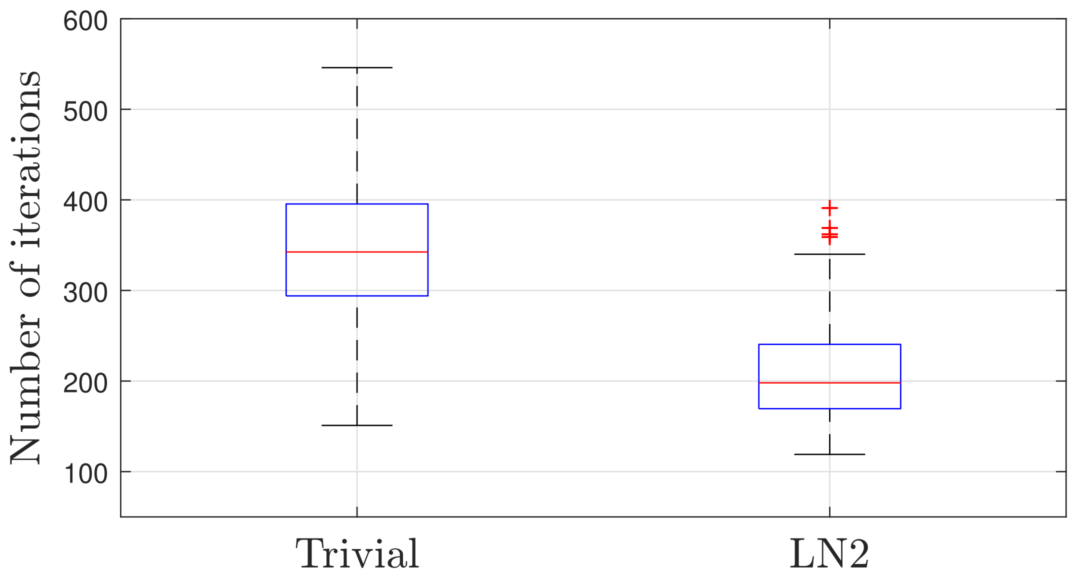

FIGURE 3.8: Number of iterations needed to converge to the optimal hyper-parameters using the trivial and the LN2 solutions for Example 3.3.

In future research, other selection criteria will be dealt with to enforce specific properties of the solutions without decreasing the out-of-sample performance. Further work will be devoted to the optimization of the hyperparameters in the degenerate case.

3.6 Proofs

The proofs of all the theorems presented in this chapter are reported in this section.

Proof of Proposition 3.1. To prove this statement let us consider first the case when $\tau > 0$. Here, the matrix $\boldsymbol{A}$ is full-rank because its eigenvalues have to be greater than the one of $\tau \boldsymbol{I}_n$ and therefore they have to be greater than τ. Therefore, the matrix $\boldsymbol{B}$ is the multiplication between a full-rank matrix and a second matrix. Therefore rank $(\boldsymbol{B}) =$ rank $(\boldsymbol{K})$.

In the case where $\tau = 0$, we have

$$\begin{aligned} \boldsymbol{B} &= \boldsymbol{K}\boldsymbol{A} && (3.66)\\ &= \boldsymbol{K}\left(\boldsymbol{K} + \mu \boldsymbol{M}\boldsymbol{K}\right) && (3.67)\\ &= \boldsymbol{K}\left(\boldsymbol{I}_n + \mu \boldsymbol{M}\right)\boldsymbol{K} && (3.68) \end{aligned}$$

since the matrix $\boldsymbol{I}_n + \mu \boldsymbol{M}$ is positive definite, we can employ the Cholesky decomposition [52] $\boldsymbol{I}_n + \mu \boldsymbol{M} = \boldsymbol{L}\boldsymbol{L}^\top$ in order to write

$$\begin{aligned} \boldsymbol{B} &= \boldsymbol{K}\boldsymbol{L}\boldsymbol{L}^\top \boldsymbol{K} && (3.69)\\ &= \left(\boldsymbol{K}\boldsymbol{L}\right)\left(\boldsymbol{K}\boldsymbol{L}\right)^\top && (3.70) \end{aligned}$$

therefore, the rank of $\boldsymbol{B}$ is equal to the rank of $\boldsymbol{K}\boldsymbol{L}$, where $\boldsymbol{L}$ is a full rank matrix. Therefore, $\boldsymbol{B}$ is full-rank. ■

Proof of Theorem 3.1. Using the eigen-decomposition [52] of the matrix $\boldsymbol{K} = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^\top$, the matrix $\boldsymbol{B}$ can be rewritten as:

$$
\begin{aligned}
\boldsymbol{B} &= \boldsymbol{K}\left(\boldsymbol{K} + \tau \boldsymbol{I}_n + \mu \boldsymbol{M}\boldsymbol{K}\right) && (3.71)\\
&= \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^\top \left(\boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^\top + \tau \boldsymbol{I}_n + \mu \boldsymbol{M}\boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^\top\right) && (3.72)\\
&= \boldsymbol{U}\boldsymbol{\Lambda} \left(\boldsymbol{U}^\top\boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^\top + \tau \boldsymbol{U}^\top + \mu \boldsymbol{U}^\top\boldsymbol{M}\boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^\top\right) && (3.73)\\
&= \boldsymbol{U}\boldsymbol{\Lambda} \left(\boldsymbol{\Lambda} + \tau \boldsymbol{I}_m + \mu \boldsymbol{U}^\top\boldsymbol{M}\boldsymbol{U}\boldsymbol{\Lambda}\right) \boldsymbol{U}^\top && (3.74)\\
&= \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{V}\boldsymbol{U}^\top, && (3.75)
\end{aligned}
$$

where $\boldsymbol{V} = \boldsymbol{\Lambda} + \tau \boldsymbol{I}_m + \mu \boldsymbol{U}^\top\boldsymbol{M}\boldsymbol{U}\boldsymbol{\Lambda} \in \mathbb{R}^{m\times m}$. The eigenvalues of V have to be greater or equal to the smallest eigenvalues of $\boldsymbol{\Lambda}$, see Theorem 8.1.5 in [52]. Since $\boldsymbol{\Lambda}$ is a diagonal matrix whose diagonal elements are the strictly positive eigenvalues of $\boldsymbol{K}$, $\boldsymbol{V}$ has only strictly positive eigenvalues and, therefore, it is invertible for every non-negative value of τ and μ.

From this fact and the eigen-decomposition of $\boldsymbol{K}$, it is possible to rewrite the linear system (3.16) as

$$
\begin{aligned}
\boldsymbol{B}\boldsymbol{c} &= \boldsymbol{b} && (3.76)\\
\boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{V}\boldsymbol{U}^\top\boldsymbol{c} &= \boldsymbol{K}\boldsymbol{y}^\top && (3.77)\\
\boldsymbol{U}^\top\boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{V}\boldsymbol{U}^\top\boldsymbol{c} &= \boldsymbol{U}^\top\boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^\top\boldsymbol{y}^\top && (3.78)\\
\boldsymbol{\Lambda}\boldsymbol{V}\boldsymbol{U}^\top\boldsymbol{c} &= \boldsymbol{\Lambda}\boldsymbol{U}^\top\boldsymbol{y}^\top && (3.79)\\
\boldsymbol{U}^\top\boldsymbol{c} &= \left(\boldsymbol{\Lambda}\boldsymbol{V}\right)^{-1}\boldsymbol{\Lambda}\boldsymbol{U}^\top\boldsymbol{y}^\top && (3.80)\\
\boldsymbol{U}^\top\boldsymbol{c} &= \boldsymbol{V}^{-1}\boldsymbol{U}^\top\boldsymbol{y}^\top && (3.81)
\end{aligned}
$$

This is a rectangular linear system with n variables and m equations that has the same solutions set $\mathcal{S}_c$ as the one of (3.16). Since $\operatorname{rank}\left(\boldsymbol{U}^\top\right) = m$, for the Rouché-Capelli theorem [113], point **a)** of the theorem is proven because the system (3.16) is consistent. The same theorem states that the solution set $\mathcal{S}_c$ is an affine space with dimension equal to the number of variables minus the rank of $\boldsymbol{U}^\top$. Therefore its dimension is $n - m$, proving point **b)** of the theorem.

To prove point **c)**, let $\boldsymbol{c}_1, \boldsymbol{c}_2 \in \mathcal{S}_c$. It holds that:

$$
\begin{aligned}
\boldsymbol{U}^\top\left(\boldsymbol{c}_1 - \boldsymbol{c}_2\right) &= \boldsymbol{U}^\top\boldsymbol{c}_1 - \boldsymbol{U}^\top\boldsymbol{c}_2 && (3.82)\\
&= \boldsymbol{V}^{-1}\boldsymbol{U}^\top\boldsymbol{y}^\top - \boldsymbol{V}^{-1}\boldsymbol{U}^\top\boldsymbol{y}^\top && (3.83)\\
&= \boldsymbol{0}_{m\times 1} && (3.84)
\end{aligned}
$$

Then, $\boldsymbol{c}_1 - \boldsymbol{c}_2 \in \mathcal{N}\left(\boldsymbol{U}^\top\right)$. Before proving point **d)**, we need the following Lemma.

Lemma 3.1. *Let $\mathcal{N}\left(\boldsymbol{B}\right)$ be the null space of the matrix $\boldsymbol{B}$. Then $\mathcal{N}\left(\boldsymbol{U}^\top\right) \subseteq \mathcal{N}\left(\boldsymbol{B}\right)$.*

The proof of the Lemma is straightforward. Let $\boldsymbol{x} \in \mathcal{N}\left(\boldsymbol{U}^\top\right)$. Then,

$$
\boldsymbol{B}\boldsymbol{x} = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{V}\underbrace{\boldsymbol{U}^\top\boldsymbol{x}}_{\boldsymbol{0}_{m\times 1}} = \boldsymbol{0}_{n\times 1}. \qquad (3.85)
$$

Therefore, $\boldsymbol{x} \in \mathcal{N}\left(\boldsymbol{B}\right)$ and $\mathcal{N}\left(\boldsymbol{U}^\top\right) \subseteq \mathcal{N}\left(\boldsymbol{B}\right)$.

Now, to start with the proof of point **d)**, let us first note that the Hessian matrix of J_c is equal to $\boldsymbol{B}$, $\forall \boldsymbol{c} \in \mathbb{R}^{n\times 1} \supseteq \mathcal{S}_c$, as the cost function is quadratic. Then, all the points inside $\mathcal{S}_c$ share the same positive semi-define Hessian matrix and are local minima. Consider now two minima $\boldsymbol{c}_1, \boldsymbol{c}_2 \in \mathcal{S}_c$ and their difference $\boldsymbol{w} = \boldsymbol{c}_1 - \boldsymbol{c}_2 \in \mathcal{N}\left(\boldsymbol{U}^\top\right)$.

Evaluating the cost function in $\boldsymbol{c}_1 = \boldsymbol{c}_2 + \boldsymbol{w}$, we obtain:

$$\begin{aligned} J_c\left(\boldsymbol{c}_1\right) &= \boldsymbol{c}_1^\top \boldsymbol{B} \boldsymbol{c}_1 - 2\boldsymbol{c}_1^\top \boldsymbol{b} + \boldsymbol{y}\boldsymbol{y}^\top && (3.86)\\ &= \left(\boldsymbol{c}_2 + \boldsymbol{w}\right)^\top \boldsymbol{B}\left(\boldsymbol{c}_2 + \boldsymbol{w}\right) - 2\left(\boldsymbol{c}_2 + \boldsymbol{w}\right)^\top \boldsymbol{b} + \boldsymbol{y}\boldsymbol{y}^\top && (3.87)\\ &= \boldsymbol{c}_2^\top \boldsymbol{B}\boldsymbol{c}_2 + \boldsymbol{w}^\top \boldsymbol{B}\boldsymbol{c}_2 + \boldsymbol{c}_2^\top \boldsymbol{B}\boldsymbol{w} + \boldsymbol{w}^\top \boldsymbol{B}\boldsymbol{w} - 2\boldsymbol{c}_2^\top \boldsymbol{b} - 2\boldsymbol{w}^\top \boldsymbol{b} + \boldsymbol{y}\boldsymbol{y}^\top. && (3.88) \end{aligned}$$

Since $\boldsymbol{w} \in \mathcal{N}\left(\boldsymbol{U}^\top\right) \subseteq \mathcal{N}\left(\boldsymbol{B}\right)$, as shown in Lemma 3.1, all the terms that contains $\boldsymbol{B}\boldsymbol{w}$, or its transpose, are zero, obtaining:

$$\begin{aligned} J_c\left(\boldsymbol{c}_1\right) &= \underbrace{\boldsymbol{c}_2 \boldsymbol{B}\boldsymbol{c}_2 - 2\boldsymbol{c}_2^\top \boldsymbol{b} + \boldsymbol{y}\boldsymbol{y}^\top}_{J_c(\boldsymbol{c}_2)} - 2\boldsymbol{w}^\top \boldsymbol{b} && (3.89)\\ &= J_c\left(\boldsymbol{c}_2\right) - 2\boldsymbol{w}^\top \boldsymbol{b}. && (3.90) \end{aligned}$$

Furthermore, it can be noted that:

$$2\boldsymbol{w}^\top \boldsymbol{b} = 2\boldsymbol{w}^\top \boldsymbol{K}\boldsymbol{y}^\top = 2\underbrace{\boldsymbol{w}^\top \boldsymbol{U}}_{\boldsymbol{0}_{1\times m}} \boldsymbol{\Lambda}\boldsymbol{U}^\top \boldsymbol{y}^\top = 0 \qquad (3.91)$$

obtaining $J_c\left(\boldsymbol{c}_1\right) = J_c\left(\boldsymbol{c}_2\right)$. Since all the local minima share the same cost function value and J_c is quadratic, they are all global minima. ■

Proof of Theorem 3.3. Using the eigen-decomposition $\boldsymbol{K} = \boldsymbol{U}\Lambda\boldsymbol{U}^\top$, $\overline{\boldsymbol{B}}$ can be rewritten as:

$$\begin{aligned} \overline{\boldsymbol{B}} &= \boldsymbol{K}\left(\boldsymbol{P}\boldsymbol{K} + \tau \boldsymbol{I}_n + \mu \boldsymbol{M}\boldsymbol{K}\right) && (3.92)\\ &= \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^\top \left(\boldsymbol{P}\boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^\top + \tau \boldsymbol{I}_n + \mu \boldsymbol{M}\boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^\top\right) && (3.93)\\ &= \boldsymbol{U}\boldsymbol{\Lambda}\left(\boldsymbol{U}^\top \boldsymbol{P}\boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^\top + \tau \boldsymbol{U}^\top \cdot + \mu \boldsymbol{U}^\top \boldsymbol{M}\boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^\top\right) && (3.94)\\ &= \boldsymbol{U}\boldsymbol{\Lambda}\left(\boldsymbol{U}^\top \boldsymbol{P}\boldsymbol{U}\boldsymbol{\Lambda} + \tau I_m + \mu \boldsymbol{U}^\top \boldsymbol{M}\boldsymbol{U}\boldsymbol{\Lambda}\right)\boldsymbol{U}^\top && (3.95)\\ &= \boldsymbol{U}\boldsymbol{\Lambda}\left(\boldsymbol{U}^\top \left(\boldsymbol{P} + \mu \boldsymbol{M}\right)\boldsymbol{U}\boldsymbol{\Lambda} + \tau \boldsymbol{I}_m\right)\boldsymbol{U}^\top && (3.96)\\ &= \boldsymbol{U}\boldsymbol{\Lambda}\overline{\boldsymbol{V}}\boldsymbol{U}^\top, && (3.97) \end{aligned}$$

where $\overline{\boldsymbol{V}} = \boldsymbol{U}^\top \left(\boldsymbol{P} + \mu \boldsymbol{M}\right)\boldsymbol{U}\boldsymbol{\Lambda} + \tau \boldsymbol{I}_m \in \mathbb{R}^{m\times m}$. If $\tau > 0$, $\overline{\boldsymbol{V}}$ is invertible. In fact, let us first consider the case of $\tau > 0$. $\overline{\boldsymbol{V}}$ is the sum of a diagonal matrix $\tau \boldsymbol{I}_m$, whose diagonal elements and eigenvalues are strictly positive, and another matrix. Since the eigenvalues of $\overline{\boldsymbol{V}}$ have to be greater or equal than the smallest eigenvalue of $\tau \boldsymbol{I}_m$, as shown in Theorem 8.1.5 of [52], all the eigenvalues of $\overline{\boldsymbol{V}}$ are strictly greater than 0. The proof can be completed by following the same rationale employed in the proof of Theorem 3.1 (reported in Section 3.6 on 80), from Equation (3.76) on, with $\overline{\boldsymbol{V}}$ instead of $\boldsymbol{V}$. ■

Proof of Theorem 3.2. Using the Mercer decomposition (3.21), we can rewrite $\hat{g}_{\boldsymbol{c}_1}(\boldsymbol{x})$ and $\hat{g}_{\boldsymbol{c}_2}(\boldsymbol{x})$ as

$$\hat{g}_{\boldsymbol{c}_1}(\boldsymbol{x}) = \boldsymbol{c}^\top \boldsymbol{k}_*(\boldsymbol{x}) = \boldsymbol{c}_1^\top \boldsymbol{\Phi} \boldsymbol{f}(\boldsymbol{x}) \tag{3.98}$$

$$\hat{g}_{\boldsymbol{c}_2}(\boldsymbol{x}) = \boldsymbol{c}^\top \boldsymbol{k}_*(\boldsymbol{x}) = \boldsymbol{c}_2^\top \boldsymbol{\Phi} \boldsymbol{f}(\boldsymbol{x}) \tag{3.99}$$

here $\boldsymbol{\Phi} \in \mathbb{R}^{n\times m}$ is a matrix whose (i,j) entry is $\varphi_j(\boldsymbol{x}_i)$ and $\boldsymbol{f} : \mathcal{X} \to \mathbb{R}^{m\times 1}$ is a function such that $\boldsymbol{f}(\boldsymbol{x}) = \left[\sigma_1^2\varphi_1(\boldsymbol{x}), \ldots, \sigma_m^2\varphi_m(\boldsymbol{x})\right]^\top \in \mathbb{R}^{m\times 1}$.

Therefore, their difference is

$$\begin{aligned}
\hat{g}_{\boldsymbol{c}_1}(\boldsymbol{x}) - \hat{g}_{\boldsymbol{c}_2}(\boldsymbol{x}) &= \boldsymbol{c}_1^\top \boldsymbol{\Phi} \boldsymbol{f}(\boldsymbol{x}) - \boldsymbol{c}_2^\top \boldsymbol{\Phi} \boldsymbol{f}(\boldsymbol{x}) && (3.100)\\
&= \left(\boldsymbol{c}_1^\top - \boldsymbol{c}_2^\top\right) \boldsymbol{\Phi} \boldsymbol{f}(\boldsymbol{x}) && (3.101)\\
&= \boldsymbol{w}^\top \boldsymbol{\Phi} \boldsymbol{f}(\boldsymbol{x}) && (3.102)
\end{aligned}$$

where $\boldsymbol{w} = \boldsymbol{c}_1 - \boldsymbol{c}_2 \in \mathcal{N}\left(\boldsymbol{U}^\top\right)$.

Since $\boldsymbol{K} = \boldsymbol{U}\Lambda\boldsymbol{U}^\top$ and $\boldsymbol{U}$ and $\boldsymbol{\Lambda}$ are full-rank matrices, $\mathcal{N}\left(\boldsymbol{U}^\top\right) = \mathcal{N}(\boldsymbol{K})$. Using the Mercer decomposition (3.21), the kernel matrix $\boldsymbol{K}$ can be written as $\boldsymbol{K} = \boldsymbol{\Phi}\boldsymbol{\Sigma}\boldsymbol{\Phi}^\top$, where $\boldsymbol{\Sigma}$ is a diagonal matrix whose elements are the eigenvalues σ_i^2 of the the kernel. Since the eigenfunction φ_i can be selected in order to be orthogonal with each other (with respect of the L_2 inner-product with the same measure used for the Mercer theorem, see Section 1.1.2), the columns of the matrix $\boldsymbol{\Phi}$ are orthogonal with each other and therefore $\boldsymbol{\Phi}$ is a full rank matrix. From this fact, it is possible to see that $\mathcal{N}(\boldsymbol{K}) = \mathcal{N}\left(\boldsymbol{\Phi}^\top\right) = \mathcal{N}\left(\boldsymbol{U}^\top\right)$.

Therefore

$$\hat{g}_{\boldsymbol{c}_1}(\boldsymbol{x}) - \hat{g}_{\boldsymbol{c}_2}(\boldsymbol{x}) = \boldsymbol{w}^\top \boldsymbol{\Phi} \boldsymbol{f}(\boldsymbol{x}) = 0 \tag{3.103}$$

and the two functions have the same value. ■

Proof of Theorem 3.4. The proof follows the same line of that of Theorem 3.2 reported in Section 3.6 on 82. ■

CHAPTER 4

Kernel-based continuous-time linear system identification

This chapter introduces a method that employs kernel-based learning for the identification of continuous-time linear systems. The proposed algorithm is a non-parametric method and it identifies directly the transfer function of the system under exam. Since the method is designed for the identification of continuous-time linear systems, it can also work with irregularly sampled data.

This new method is an expansion of the algorithm, described in details in [95] and Section 2.2, that employs the RKHS theory to identifies the impulse response of the system from data. In particular, these kernel-methods limit themselves to the identification of a non-parametric impulse response function that has limited practical application. For this reason, in the literature, these methods are often used in conjunction with a method that can approximate the identified impulse response with a transfer function with a certain order [32]. Instead, the proposed method computes a non-parametric identified transfer function with an automatically selected order.

This chapter provides, also, some analysis about the stability of the identified model and it finishes with a numerical simulation that shows the performance of the method with respect to the state of the art methods for continuous-time system identification [46].

This chapter is organized as follow:

- Section 4.1 briefly refreshes the notions introduced in Section 2.2 about the impulse response identification;
- Section 4.2 explains how to select the various hyper-parameters of the proposed method;
- Section 4.3 contains an in-depth view on the stable-spline kernel that is used throughout the chapter;
- Section 4.4 discusses some important remarks on the computation of the derived kernel;
- Section 4.5 explains how to convert the impulse response identification to the non-parametric transfer function;

- Section 4.6 illustrates how to make the identified transfer function rational;
- Section 4.7 delves into the problems that arises when complex excitation signal are used in the experiment;
- Section 4.8 contains a brief summary of the proposed identification algorithm;
- Section 4.9 presents some numerical results of the proposed method compared with other methods;
- Section 4.10 contains the proofs of all the various theorems.

4.1 Non-parametric impulse response identification

Consider the continuous causal LTI system $\breve{\mathscr{G}}$ with impulse response $\breve{g} : \mathbb{R} \to \mathbb{R}$, then the input/output relation of $\breve{\mathscr{G}}$ is

$$y(t) = [\breve{g} \star u](t) = \int_0^{+\infty} \breve{g}(\xi)\, u(t-\xi)\, d\xi \tag{4.1}$$

where $u : \mathbb{R}_+ \to \mathbb{R}$ and $y : \mathbb{R}_+ \to \mathbb{R}$ are, respectively, the input and the output signal. In the Laplace domain, this relation becomes

$$Y(s) = \breve{G}(s)\, U(s) \tag{4.2}$$

where $U(s) = \mathcal{L}[u](s)$, $Y(s) = \mathcal{L}[y](s)$ and $\breve{G}(s) = \mathcal{L}[\breve{g}](s)$ is the transfer function of the system $\breve{\mathscr{G}}$.

Now, suppose to have at your disposal a dataset containing $n \in \mathbb{N}\backslash\{0\}$ noisy measurements obtained with an experiment on the plant

$$\mathcal{D} = \{(t_i, y_i), 1 \le i \le n\} \tag{4.3}$$

distributed according to the probabilistic model

$$y_i = [\breve{g} \star u](t_i) + e_i \qquad\qquad i = 1, \dots, n \tag{4.4}$$

where $e_i \sim \mathcal{N}(0, \eta^2)$ are IID output-error Gaussian distributed noises and $u : \mathbb{R}_+ \to \mathbb{R}$ is the known input excitations used during the experiment. For simplicity, we assume that

Assumption 4.1. *The time-instants t_i are in chronological order, i.e. $t_i \ge t_{i-1}$, $i = 1, \dots, n$.*

Assumption 4.2. *The excitation signal $u(t)$ is applied to the plant at the time instant $d \in \mathbb{R}$, i.e. $u(t) = 0$, $\forall t < d$.*

Both these assumptions are not restrictive. The first one imposes only a certain order of the dataset that is usually naturally respected and the second one assumes that the experiment on the system started on a certain time-instant d, as it is always done in a real case.

Following the rationale reported in [32, 95, 96] and described in Section 2.2, we can estimate $\breve{g}$ with the estimator

$$\begin{aligned}\hat{g} &= \underset{g\in\mathcal{H}_k}{\arg\min}\left\{J\left(g\right)\right\}\\ J\left(g\right) &= \sum_{i=1}^{n}\left(y_i - \left[g\star u\right]\left(t_i\right)\right)^2 + \tau\left\|g\right\|_{\mathcal{H}}^2\end{aligned} \tag{4.5}$$

where $\mathcal{H}$ is an RKHS with kernel $k : \mathbb{R}_+ \times \mathbb{R}_+ \to \mathbb{R}$, τ is a positive scalar and $\|\cdot\|_{\mathcal{H}}$ is the induced norm of the space $\mathcal{H}$. The first term of the cost function J is a loss term that becomes smaller when the model has a good fit on the dataset, while the second one is a regularization term that penalizes more complex models.

As shown in [40], this estimator can be written as

$$\hat{g}^u\left(t\right) = \sum_{i=1}^{n} c_i \hat{g}_i^u\left(t\right) \tag{4.6}$$

where the dependency on the input u is highlighted and

$$\hat{g}_i^u\left(t\right) = \int_0^{\infty} u\left(t_i - \xi\right) k\left(t, \xi\right)\, d\xi \tag{4.7}$$

and where the coefficients vector $\boldsymbol{c} = \left[c_1, c_2, \ldots, c_n\right]^\top \in \mathbb{R}^{n\times 1}$ can be found by solving the linear system

$$\boldsymbol{O}\left(\boldsymbol{O} + \tau\boldsymbol{I}_n\right)\boldsymbol{c} = \boldsymbol{O}\boldsymbol{y}^\top \tag{4.8}$$

where $\boldsymbol{y} = \left[y_1, y_2, \ldots, y_n\right] \in \mathbb{R}^{1\times n}$ and $\boldsymbol{O} \in \mathbb{R}^{n\times n}$ is a symmetric positive-definite matrix whose (i, j) element is

$$O_{i,j} = o^u\left(t_i, t_j\right) \tag{4.9}$$

where

$$o^u\left(t_i, t_j\right) = \int_0^{+\infty} u\left(t_i - \psi\right)\left(\int_0^{+\infty} u\left(t_j - \xi\right) k\left(\psi, \xi\right)\, d\xi\right) d\psi \tag{4.10}$$

For additional details, see Section 2.2.

4.2 Hyper-parameters selection

The before-mentioned algorithm requires the tuning of three hyper-parameters: the regularization strength τ and the kernel hyper-parameters $\boldsymbol{\psi} \in \mathbb{R}^{n_\psi\times 1}$.

To select them, it is useful to introduce the Bayesian interpretation of the method. The model described in (4.4) gives us the likelihood distribution $p\left(\boldsymbol{y}|g, \boldsymbol{\zeta}\right)$ where $\boldsymbol{\zeta} = \left[\boldsymbol{\psi}^\top, \tau\right]^\top \in \mathbb{R}^{n_\zeta\times 1}$. Imposing a Gaussian stochastic process prior on the impulse response $p\left(g|\boldsymbol{\zeta}\right)$ allows obtaining a posterior $p\left(g|\boldsymbol{y}, \boldsymbol{\zeta}\right)$ whose mean is equal to the estimator (4.6), as shown in [95].

From this different point of view, it is possible to compute the marginal likelihood pdf

$$p\left(\boldsymbol{y}|\boldsymbol{\zeta}\right) = \int p\left(\boldsymbol{y}|g, \boldsymbol{\zeta}\right) p\left(g|\boldsymbol{\zeta}\right)\, dg \tag{4.11}$$

$$= \mathcal{N}\left(\boldsymbol{y}^\top | \mathbf{0}_{n\times 1}, \boldsymbol{O} + \tau \boldsymbol{I}_n\right). \tag{4.12}$$

This distribution represents the likelihood to have a certain set of measurements $\boldsymbol{y}$ given a certain value of the hyper-parameters $\boldsymbol{\zeta}$. For this reason, it is possible to select $\boldsymbol{\zeta}$ by searching the one that maximizes the likelihood to have the set of measurements at our disposal. Therefore:

$$\hat{\boldsymbol{\zeta}} = \underset{\boldsymbol{\zeta}\in\mathbb{R}^{n_\zeta}}{\arg\min} \left\{\boldsymbol{y}\left(\boldsymbol{O} + \tau \boldsymbol{I}_n\right)^{-1}\boldsymbol{y}^\top + \log\det\left(\boldsymbol{O} + \tau \boldsymbol{I}_n\right)\right\} \tag{4.13}$$

where, instead of the maximization of $p\left(\boldsymbol{y}|\boldsymbol{\zeta}\right)$, the negative log-pdf of $p\left(\boldsymbol{y}|\boldsymbol{\zeta}\right)$ is minimized for computational reason, as explained in Section 1.5.

4.3 Kernel selection

The performance of the estimator $\hat{g}^u$ heavily depends on the kernel used. In particular, most kernels are not suitable for this application because they define spaces that contain functions that correspond to unstable systems as shown in Section 2.1 and in [95].

To solve this problem, it is necessary to use a so-called *stable kernel* [95]. An example of this kind of kernel is the stable-spline [95] $k_q : \mathbb{R}_+ \times \mathbb{R}_+ \to \mathbb{R}$ that is defined as:

$$k_q\left(a,b\right) = \lambda s_q\left(e^{-\beta a}, e^{-\beta b}\right) \tag{4.14}$$

where $q \in \mathbb{N}\setminus\{0\}$ is the stable-spline order, β and λ are two strictly positive scalar hyper-parameters to tune and $s_q : [0,1]\times[0,1] \to \mathbb{R}$ is the regular spline kernel of order q [131], i.e.

$$s_q\left(a,b\right) = \int_0^1 G_q\left(a,x\right) G_q\left(b,x\right)\, dx \tag{4.15}$$

where

$$G_q\left(a,x\right) = \frac{1}{(q-1)!}\begin{cases}(a-x)^{q-1} & \text{if } a \geq x\\ 0 & \text{if } a < x\end{cases} \tag{4.16}$$

Remark 4.1. The λ hyper-parameter is related to the static gain of the system at hand, while β define its bandwidth.

In the literature it is possible to find other stable kernels like the continuous DC kernel [32] (see [30] for a detailed analysis on how to select the right stable-kernel). However, in this thesis, the focus will be on the stable-spline kernel because they are general enough and they are the most used kernels for this of problem due to their flexibility and properties.

In order to have a more clean formulation of the stable spline kernel, we can consider the following theorem and its corollary.

Theorem 4.1. *The spline kernel $s_q : [0,1]\times[0,1] \to \mathbb{R}$ of order q can be written as:*

$$s_q\left(a,b\right) = \sum_{h=0}^{q-1} \gamma_{q,h}\begin{cases}a^{2q-h-1}b^h & \text{if } a \leq b\\ b^{2q-h-1}a^h & \text{if } a > b\end{cases} \tag{4.17}$$

where

$$\gamma_{q,h} = \frac{(-1)^{q+h-1}}{h!\,(2q-h-1)!} \tag{4.18}$$

Proof. See Section 4.10 on page 112. ∎

Corollary 4.1. *The stable-spline kernel $k_q : \mathbb{R}_+ \times \mathbb{R}_+ \to \mathbb{R}$ of order q can be written as:*

$$k_q(a,b) = \lambda \sum_{h=0}^{q-1} \gamma_{q,h} \begin{cases} e^{-\beta[(2q-h-1)a+hb]} & \text{if } a \geq b \\ e^{-\beta[(2q-h-1)b+ha]} & \text{if } a < b \end{cases} \tag{4.19}$$

Proof. See Section 4.10 on page 117. ∎

From Corollary 4.1, we can see that the stable-spline kernel of order q is a weighted sum of q negative exponential terms. For this reason, the stable-spline kernel is easily computable for every order and the order q can be treated as an additional hyper-parameter, i.e. $\boldsymbol{\zeta} = [\lambda, \beta, \tau, q]$. If this is the case, the optimization of (4.13) becomes a mixed real-integer optimization problem that requires suitable techniques. An easy solution for this problem is to select the order q with an exhaustive search from a certain pool of values.

4.4 Computation of the new derived kernel

The method described in Section 4.1 requires a way to compute the derived kernel o^u as defined in (4.10). Looking at the definition and remembering the Assumption 4.2, we can note that

- if $t_i \leq d$ then $u(t_i - \psi) = 0, \forall \psi \in \mathbb{R}_+$ and therefore $o(t_i, t_j) = 0$;
- if $t_j \leq d$ then $u(t_j - \xi) = 0, \forall \xi \in \mathbb{R}_+$ and therefore $o(t_i, t_j) = 0$.

Now, let us assume that there are $n_z \geq 0$ time instants t_i such that $t_i \leq d$, then the matrix $\boldsymbol{O}$ becomes

$$\boldsymbol{O} = \begin{bmatrix} \mathbf{0}_{n_z \times n_z} & \mathbf{0}_{n_z \times n-n_z} \\ \mathbf{0}_{n-n_z \times n_z} & \widetilde{\boldsymbol{O}} \end{bmatrix} \in \mathbb{R}^{n \times n} \tag{4.20}$$

where $\widetilde{\boldsymbol{O}} \in \mathbb{R}^{n-n_z \times n-n_z}$ is the kernel matrix obtained using only the time instants $t_i > t_{n_z}$. From this expression, it is clear that the matrix $\boldsymbol{O}$ has n_z rows equals to $\mathbf{0}_{1\times n}$ and therefore rank $[\boldsymbol{O}] \leq n - n_z \leq n$. For this reason, the linear system (4.8) is a singular system and does not have a unique solution. This kind of problem is tackled in detail in Chapter 3, but in this special case that we can treat differently.

In particular, the linear system (4.8) becomes

$$\begin{bmatrix} \mathbf{0}_{n_z \times n_z} & \mathbf{0}_{n_z \times n-n_z} \\ \mathbf{0}_{n-n_z \times n_z} & \widetilde{\boldsymbol{O}} \end{bmatrix} \begin{bmatrix} \tau \boldsymbol{I}_{n_z \times n_z} & \mathbf{0}_{n_z \times n_z - z} \\ \mathbf{0}_{n-n_z \times n_z} & \widetilde{\boldsymbol{O}} + \tau \boldsymbol{I}_{n_z - z} \end{bmatrix} \begin{bmatrix} \boldsymbol{c}_1 \\ \boldsymbol{c}_2 \end{bmatrix} = \begin{bmatrix} \mathbf{0}_{n_z \times n_z} & \mathbf{0}_{n_z \times n-n_z} \\ \mathbf{0}_{n-n_z \times n_z} & \widetilde{\boldsymbol{O}} \end{bmatrix} \begin{bmatrix} \boldsymbol{y}_1^\top \\ \boldsymbol{y}_2^\top \end{bmatrix} \tag{4.21}$$

where $\boldsymbol{c}_1 \in \mathbb{R}^{n_z \times 1}$ and $\boldsymbol{y}_1 \in \mathbb{R}^{1 \times n_z}$ are, respectively, the vector with the first n_z elements of $\boldsymbol{c}$ and $\boldsymbol{y}$ and $\boldsymbol{c}_2 \in \mathbb{R}^{n-n_z \times 1}$ and $\boldsymbol{y}_2 \in \mathbb{R}^{1 \times n-n_z}$ are the other parts of the vectors $\boldsymbol{c}$ and $\boldsymbol{y}$.

With some mathematical steps, we can write

$$\begin{bmatrix} \mathbf{0}_{n_z \times 1} \\ \widetilde{\boldsymbol{O}}\left(\widetilde{\boldsymbol{O}} + \tau \boldsymbol{I}_{n-n_z}\right) \boldsymbol{c}_2 \end{bmatrix} = \begin{bmatrix} \mathbf{0}_{n_z \times 1} \\ \widetilde{\boldsymbol{O}} \boldsymbol{y}_2^\top \end{bmatrix} \tag{4.22}$$

From this formulation, it is possible to note that the equality is verified for every value of $\boldsymbol{c}_1 \in \mathbb{R}^{n_z \times 1}$. For this reason, we can set $\boldsymbol{c}_1 = \mathbf{0}_{n_z \times 1}$ in order to reduce the computational complexity of the estimated model, as shown in Section 3.3. The other part of $\boldsymbol{c}$ can be computed by solving the linear system

$$\widetilde{\boldsymbol{O}}\left(\widetilde{\boldsymbol{O}} + \tau \boldsymbol{I}_{n-n_z}\right) \boldsymbol{c}_2 = \widetilde{\boldsymbol{O}} \boldsymbol{y}_2^\top \tag{4.23}$$

If the first n_z samples of the dataset $\mathcal{D}$ are discarded, then this new linear system will be equivalent to the one in (4.8). For this reason, for the rest of the chapter, the following Assumption is considered respected.

Assumption 4.3. *All the time instants t_i in the dataset $\mathcal{D}$ are strictly greater than d, i.e. $t_i > d$ for $i = 1, \dots, n$.*

Remark 4.2. If the original dataset does not respect this assumption then it is always possible to create a new dataset that respects it by discarding the data with time instants $t_i \leq d$ without losing any information.

Remark 4.3. The Assumption 4.3 is due to the causality of the LTI system under analysis. In fact, the samples taken before the injection of the input signal carries no information about the response of the input.

The formula (4.10) for the computation of the derived kernel needs to be computed analytically and change for every combination of input signal $u(t)$ and kernel used k. For some combination, this can be a long and not trivial task. If the kernel used is a stable-spline of order q, we can simplify this task by employing the following theorem.

Theorem 4.2. *Let:*

- *the kernel k be a stable-spline of order q;*
- *$u : \mathbb{R} \to \mathbb{R}$ be an input signal such that it respects Assumption 4.2;*

Then the derived kernel is equal to

$$o_q^u(t_i, t_j) = \lambda \sum_{h=0}^{q-1} \gamma_{q,h} \begin{cases} r_{q,h}^u(t_i, t_j) + w_{q,h}^u(t_i, t_j) & t_i \leq t_j \\ r_{q,h}^u(t_j, t_i) + w_{q,h}^u(t_j, t_i) & t_i > t_j \end{cases} \tag{4.24}$$

where

$$r_{q,h}^u(t_i, t_j) = \int_d^{t_i} \int_d^{t_j - t_i + \xi} u(\xi)\, u(\psi)\, e^{-\beta[(2q-h-1)(t_j-\psi)+h(t_i-\xi)]}\, d\psi\, d\xi \tag{4.25}$$

$$w_{q,h}^u(t_i, t_j) = \int_d^{t_i} \int_{t_j - t_i + \xi}^{t_j} u(\xi)\, u(\psi)\, e^{-\beta[(2q-h-1)(t_i-\xi)+h(t_j-\psi)]}\, d\psi\, d\xi \tag{4.26}$$

Proof. See Section 4.10 on page 118. ■

4.5 Transfer function estimation

For practical applications, like control design and behavior analysis, a non-parametric impulse response is not as useful as the transfer function representation. For this reason, the estimator $\hat{g}^u$, as defined in (4.6), is not practical. Therefore, it is useful to compute the corresponding transfer function $\hat{G}^u$.

Theorem 4.3. *Given the non-parametric estimator* $\hat{g}^u$, *as explained in* (4.6), *of an LTI system, the corresponding transfer function is*

$$\hat{G}^u(s) = \sum_{i=1}^{n} c_i \hat{G}_i^u(s) \tag{4.27}$$

where

$$\hat{G}_i^u(s) = \int_d^{t_i} u(x)\, K(s; t_i - x)\, dx \tag{4.28}$$

and

$$K(s; x) = \int_0^{\infty} k(t, x)\, e^{-s\tau}\, dt \tag{4.29}$$

Proof. See Section 4.10 on page 120. ∎

From this Theorem, it is possible to note that the estimated transfer function is composed by the convolution of two terms: the first one $u(x)$ depends only on the shape of the excitation signal while the second one $K(s; t_i - x)$ depends only on the kernel used.

For the stable-spline kernel of generic order q, it is possible to compute a more informative formulation thanks to the following theorem.

Theorem 4.4. *Let the kernel be a stable-spline* k_q *of order* q. *The identified transfer function can be written as*

$$\hat{G}^u(s) = \lambda \left[\sum_{h=0}^{q-1} Q_{q,h}^u(s) + H_q^u(s) \right] \tag{4.30}$$

where

$$Q_{q,h}^u(s) = \frac{\gamma_{q,h}}{s + \beta h} \left(\sum_{i=1}^{n} c_i A_i^u(\beta(2q - h - 1)) \right) \tag{4.31}$$

$$H_q^u(s) = \frac{(-1)^q \beta^{2q-1}}{\prod_{i=0}^{2q-1}(\beta i + s)} \left(\sum_{i=1}^{n} c_i A_i^u(s + \beta(2q - 1)) \right) \tag{4.32}$$

and

$$A_i^u(x) = \int_d^{t_i} u(t)\, e^{x(t - t_i)}\, dt \tag{4.33}$$

Proof. See Section 4.10 on page 121. ∎

Here, it is clear that the estimated transfer function is a sum of $q+1$ transfer functions. The first q of them have one real pole located in a multiple of the $-\beta$ and a gain that depends on the coefficients $\boldsymbol{c}$, the hyper-parameters λ and β, the spline order and the shape of the input signal $u(t)$. The last one is more complicated. It has $2q-1$ real poles that are multiple of $-\beta$ and, eventually, other poles that depend on the shape of the input $u(t)$. In particular, the transfer function $A_i^u(s+\beta(2q-1))$ can have some poles or zeros that will be added to $\hat{G}^u$. For this reason, to evaluate the stability of the identified system it is necessary to impose a condition on the excitation signal. This is achieved by the following theorem.

Theorem 4.5. *If the input signal $u(t)$ is such that $A_i^u(s+\beta(2q-1))$ is a transfer function whose poles are all strictly negative for $i=1,\dots,n$, then $\hat{G}^u(s)$ is an asymptotically stable transfer function.*

Proof. See Section 4.10 on page 124. ■

From this Theorem, it is clear that the terms

$$A_i^u(s+\beta(2q-1)) \qquad\qquad i=1,\dots,n \tag{4.34}$$

have an important role in the identification procedure and on the stability of the identified model. It is also important to note that the identified model is always at least BIBO stable because the stable-spline kernel is a stable kernel, as shown in [95] and in Section 2.2. In the author experience, the condition imposed by the theorem is not very restrictive. For a better understanding, in the following subsection, some common input signals are analyzed.

4.5.1 Impulse input

Let $u(t)$ be a Dirac delta

$$u(t)=\delta(t-d)\,. \tag{4.35}$$

In this case, using Assumption 4.3, we have:

$$A_i^u(x)=\int_d^{t_i}\delta(t-d)\,e^{x(t-t_i)}\,dt=e^{x(d-t_i)}=e^{-x(t_i-d)} \tag{4.36}$$

Stability check From (4.36), it is straightforward to check the condition of Theorem 4.5. In particular, we have:

$$A_i^u(s+\beta(2q-1))=e^{-(s+\beta(2q-1))(t_i-d)}=e^{-s(t_i-d)}e^{-\beta(2q-1)(t_i-d)} \tag{4.37}$$

this is an input-output delay with a certain gain and therefore it is asymptotically stable and Theorem 4.4 condition is respected for every value of the hyper-parameters.

Identified transfer function Applying Theorem 4.4 and using (4.36), it is straightforward to compute the identified transfer function $\hat{G}^u$. In particular, we have:

$$Q_{q,h}^u(s)=\frac{\gamma_{q,j}}{s+\beta h}\left(\sum_{i=1}^{n}c_iA_i^u(\beta(2q-h-1))\right) \tag{4.38}$$

$$=\frac{\gamma_{q,j}}{s+\beta h}\left(\sum_{i=1}^{n}c_ie^{-\beta(2q-h-1)(t_i-d)}\right) \tag{4.39}$$

$$H_q^u(s) = \frac{(-1)^q \beta^{2q-1}}{\prod_{i=0}^{2q-1}(\beta i + s)} \left(\sum_{i=1}^{n} c_i A_i^u (s + \beta(2q-1)) \right) \tag{4.40}$$

$$= \frac{(-1)^q \beta^{2q-1}}{\prod_{i=0}^{2q-1}(\beta i + s)} \left(\sum_{i=1}^{n} c_i e^{-\beta(2q-1)(t_i-d)} e^{-s(t_i-d)} \right) \tag{4.41}$$

In this case, the transfer function $H_q^u(s)$ is not rational. In particular, the numerator is composed of a sum of weighted input-output delays. To highlight this fact, we can define

$$T_q^u(s) = \sum_{i=1}^{n} c_i e^{-\beta(2q-1)(t_i-d)} e^{-s(t_i-d)} \tag{4.42}$$

in order to isolate the non-rational part of $H_q^u(s)$.

$$H_q^u(s) = \frac{(-1)^q \beta^{2q-1}}{\prod_{i=0}^{2q-1}(\beta i + s)} T_q^u(s) \tag{4.43}$$

Remark 4.4. The input-output delays in $T_q^u(s)$ are all actual delays and not advances because $t_i - d > 0$ for $i = 1 \dots n$ thanks to Assumption 4.3.

4.5.2 Step input

Let $u(t)$ be a step

$$u(t) = \begin{cases} 1 & \text{if } t \geq d \\ 0 & \text{if } t < d \end{cases} \tag{4.44}$$

In this case, using Assumption 4.3, we have:

$$A_i^u(x) = \int_d^{t_i} u(t)\, e^{x(t-t_i)}\, dt \tag{4.45}$$

$$= e^{-xt_i} \int_d^{t_i} e^{xt}\, dt \tag{4.46}$$

$$= e^{-xt_i} \left[\frac{e^{xt}}{x} \right]_d^{t_i} \tag{4.47}$$

$$= e^{-xt_i} \frac{e^{xt_i} - e^{xd}}{x} \tag{4.48}$$

$$= \frac{1 - e^{-x(t_i-d)}}{x} \tag{4.49}$$

Stability check From (4.49), it is straightforward to check the condition of Theorem 4.5. In particular, we have:

$$A_i^u(s + \beta(2q-1)) = \frac{1 - e^{-(s+\beta(2q-1))(t_i-d)}}{s + \beta(2q-1)} \tag{4.50}$$

$$= \frac{1}{s + \beta(2q-1)} - e^{-s(t_i-d)} \frac{e^{-\beta(2q-1)(t_i-d)}}{s + \beta(2q-1)} \tag{4.51}$$

this is a sum of two transfer functions, the second one with a input-output delay, that share the same pole in

$$p = -\beta\left(2q-1\right) \tag{4.52}$$

since $q \in \mathbb{N}$, $q \geq 1$ and $\beta > 0$, this pole is strictly negative for every value of the hyper-parameters.

Identified transfer function Applying Theorem 4.4 and using (4.49), it is straightforward to compute the identified transfer function $\hat{G}^u$.

$$\begin{aligned}
Q^u_{q,h}\left(s\right) &= \frac{\gamma_{q,h}}{s+\beta h}\left(\sum_{i=1}^{n} c_i A_i^u\left(\beta\left(2q-h-1\right)\right)\right) &(4.53)\\
&= \frac{\gamma_{q,h}}{s+\beta j}\left(\sum_{i=1}^{n} c_i \frac{1-e^{-\beta(2q-h-1)(t_i-d)}}{\beta\left(2q-h-1\right)}\right) &(4.54)\\
&= \frac{\gamma_{q,h}}{\beta\left(2q-h-1\right)\left(s+\beta h\right)}\left(\sum_{i=1}^{n} c_i\left(1-e^{-\beta(2q-h-1)(t_i-d)}\right)\right) &(4.55)
\end{aligned}$$

$$\begin{aligned}
H_q\left(s\right) &= \frac{\left(-1\right)^q\beta^{2q-1}}{\prod_{i=0}^{2q-1}\left(\beta i+s\right)}\left(\sum_{i=1}^{n} c_i A_i^u\left(s+\beta\left(2q-1\right)\right)\right) &(4.56)\\
&= \frac{\left(-1\right)^q\beta^{2q-1}}{\prod_{i=0}^{2q-1}\left(\beta i+s\right)}\left(\sum_{i=1}^{n} c_i \frac{1-e^{-(s+\beta(2q-1))(t_i-d)}}{\left(s+\beta\left(2q-1\right)\right)}\right) &(4.57)\\
&= \frac{\left(-1\right)^q\beta^{2q-1}}{\left(s+\beta\left(2q-1\right)\right)\prod_{i=0}^{2q-1}\left(\beta i+s\right)}\left(\sum_{i=1}^{n} c_i - \sum_{i=1}^{n} c_i e^{-\beta(2q-1)(t_i-d)}e^{-s(t_i-d)}\right) &(4.58)\\
&= \frac{\left(-1\right)^q\beta^{2q-1}}{\left(s+\beta\left(2q-1\right)\right)\prod_{i=0}^{2q-1}\left(\beta i+s\right)}\left(\sum_{i=1}^{n} c_i - T_q^u\left(s\right)\right) &(4.59)
\end{aligned}$$

here, we can note that the transfer function $H_q^u\left(s\right)$ contains a non-rational term $T_q^u\left(s\right)$ similar to the impulse excitation analyzed before.

4.5.3 Monomial input

Let $u\left(t\right)$ be a generic monomial of order $v \in \mathbb{Z}$

$$u\left(t\right) = \begin{cases} t^v & \text{if } t \geq d \\ 0 & \text{if } t < d \end{cases} \tag{4.60}$$

Remark 4.5. This is a signal that generalizes some of the most common excitations. For example:

- with $v = 0$ this a step;
- with $v = 1$ this a ramp;
- with $v = 2$ this a parable;

and so on.

In this case, using Assumption 4.3, we have:

$$A_i^u(x) = \int_d^{t_i} u(t)\, e^{x(t-t_i)}\, dt \tag{4.61}$$

$$= e^{-xt_i} \int_d^{t_i} t^v e^{xt}\, dt \tag{4.62}$$

$$= e^{-xt_i} \left[\frac{e^{xt}}{x^{v+1}} P_v(xt) \right]_d^{t_i} \tag{4.63}$$

where $P_v(h)$ is a polynomial of grade v that is defined as

$$P_v(h) = v! \sum_{z=0}^{v} \frac{(-1)^{v+z}}{z!} h^z \tag{4.64}$$

the integral $A_i^u(x)$ is, then, equal to

$$A_i^u(x) = e^{-xt_i} \left[\frac{e^{xt_i}}{x^{v+1}} P_v(xt_i) - \frac{e^{xd}}{x^{v+1}} P_v(xd) \right] \tag{4.65}$$

$$= \frac{P_v(xt_i)}{x^{v+1}} - e^{-x(t_i-d)} \frac{P_v(xd)}{x^{v+1}} \tag{4.66}$$

Stability check From (4.66), it is straightforward to check the condition of Theorem 4.5. In particular, we have:

$$\begin{aligned} A_i^u(s+\beta(2q-1)) =& \frac{P_v((s+\beta(2q-1))t_i)}{(s+\beta(2q-1))^{v+1}} + \\ & - e^{-(s+\beta(2q-1))(t_i-d)} \frac{P_v((s+\beta(2q-1))d)}{(s+\beta(2q-1))^{v+1}} \end{aligned} \tag{4.67}$$

this result is similar to the one obtained in the previous case. It is the sum of two transfer functions, one of them with an input-output delay, that share the same poles. In this case, there are $v+1$ poles in

$$p_{1,2,\dots,v+1} = -\beta(2q-1) \tag{4.68}$$

since $q \in \mathbb{Z}$, $q \geq 1$ and $\beta > 0$, these poles are strictly negative for every value of the hyper-parameters.

Remark 4.6. Since $P_v(h)$ is a polynomial of grade v, the numerators of the two transfer functions is a polynomial in s of grade v. Therefore, there are v zeros and $v+1$ poles.

Identified transfer function Applying Theorem 4.4 and using (4.66), it is straightforward to compute the identified transfer function $\hat{G}^u$ (some straightforward mathematical steps are skipped for space sake).

$$Q_{q,h}^u(s) = \frac{\gamma_{q,h}}{s+\beta h} \sum_{i=1}^{n} c_i A_i^u(\beta(2q-h-1)) \tag{4.69}$$

$$= \gamma_{q,h} \frac{\sum_{i=1}^{n} c_i P_v \left(\beta \left(2q-h-1\right) t_i\right)}{\left(s+\beta h\right) \left(\beta \left(2q-h-1\right)\right)^{v+1}} + \tag{4.70}$$

$$- \frac{P_v \left(\beta \left(2q-h-1\right) d\right) \sum_{i=1}^{n} c_i e^{-\beta(2q-h-1)(t_i-d)}}{\left(s+\beta h\right) \left(\beta \left(2q-h-1\right)\right)^{v+1}} \tag{4.71}$$

$$H_q^u\left(s\right) = \frac{\left(-1\right)^q \beta^{2q-1}}{\prod_{i=0}^{2q-1} \left(\beta i + s\right)} \left(\sum_{i=1}^{n} c_i A_i^u \left(s + \beta \left(2q-1\right)\right) \right) \tag{4.72}$$

$$= \left(-1\right)^q \beta^{2q-1} \left(\frac{\sum_{i=1}^{n} c_i P_v \left(\left(s+\beta\left(2q-1\right)\right) t_i\right)}{\left(s+\beta\left(2q-1\right)\right)^{v+1} \prod_{i=0}^{2q-1} \left(\beta i + s\right)} \right. \tag{4.73}$$

$$\left. - \frac{P_v \left(\left(s+\beta\left(2q-1\right)\right) d\right) T_q^u\left(s\right)}{\left(s+\beta\left(2q-1\right)\right)^{v+1} \prod_{i=0}^{2q-1} \left(\beta i + s\right)} \right)$$

here, it is clear that, once again, the transfer function $H_q^u\left(s\right)$ depends on a non-rational term $T_q^u\left(s\right)$ composed by a weighted sum of input-output delays.

4.5.4 Sinewave input

Let $u\left(t\right)$ be a sinewave

$$u\left(t\right) = \begin{cases} \sin\left(\omega t + \varphi\right) & \text{if } t \geq d \\ 0 & \text{if } t < d \end{cases} \tag{4.74}$$

where $\omega \in \mathbb{R}$ with $\omega > 0$ is the rotational velocity and $\varphi \in \mathbb{R}$ is the phase.

In this case, using Assumption 4.3, we have:

$$A_i^u\left(x\right) = \int_d^{t_i} u\left(t\right) e^{x(t-t_i)}\, dt \tag{4.75}$$

$$= e^{-xt_i} \int_d^{t_i} \sin\left(\omega t + \varphi\right) e^{xt}\, dt \tag{4.76}$$

$$= e^{-xt_i} \left[\frac{e^{tx}\left(x \sin\left(\omega t + \varphi\right) - \omega \cos\left(\omega t + \varphi\right)\right)}{\omega^2 + x^2} \right]_d^{t_i} \tag{4.77}$$

$$= e^{-xt_i} \left[\frac{e^{t_i x}\left(x \sin\left(\omega t_i + \varphi\right) - \omega \cos\left(\omega t_i + \varphi\right)\right)}{\omega^2 + x^2} + \right. \tag{4.78}$$

$$\left. - \frac{e^{dx}\left(x \sin\left(\omega d + \varphi\right) - \omega \cos\left(\omega d + \varphi\right)\right)}{\omega^2 + x^2} \right]$$

$$= \frac{x \sin\left(\omega t_i + \varphi\right) - \omega \cos\left(\omega t_i + \varphi\right)}{\omega^2 + x^2} + \tag{4.79}$$

$$- \frac{e^{-x(t_i-d)}\left(x \sin\left(\omega d + \varphi\right) - \omega \cos\left(\omega d + \varphi\right)\right)}{\omega^2 + x^2}$$

Stability check From (4.79), it is straightforward to check the condition of Theorem 4.5. In particular, we have:

$$A_i^u\left(s+\beta\left(2q-1\right)\right) = \frac{\left(s+\beta\left(2q-1\right)\right)\sin\left(\omega t_i+\varphi\right)-\omega\cos\left(\omega t_i+\varphi\right)}{\omega^2+\left(s+\beta\left(2q-1\right)\right)^2} - e^{-s(t_i-d)}\frac{e^{-\beta(2q-1)(t_i-d)}\left(\left(s+\beta\left(2q-1\right)\right)\sin\left(\omega d+\varphi\right)-\omega\cos\left(\omega d+\varphi\right)\right)}{\omega^2+\left(s+\beta\left(2q-1\right)\right)^2} \tag{4.80}$$

again this is the sum of two transfer functions that share the same poles in

$$p_{1,2} = \left(2q-1\right)\beta \pm j\omega; \tag{4.81}$$

following the same reasoning used for the other input, we can see that these poles have always a strictly negative real part equal to $\left(2q-1\right)\beta$.

Identified transfer function Applying Theorem 4.4and using (4.79), it is straightforward to compute the identified transfer function $\hat{G}^u$ (some straightforward mathematical steps are skipped for space sake).

$$Q_{q,h}^u\left(s\right) = \frac{\gamma_{q,j}}{s+\beta h}\sum_{i=1}^{n} c_i A_i^u\left(\beta\left(2q-h-1\right)\right) \tag{4.82}$$

$$= \frac{\gamma_{q,h}\beta\left(2q-h-1\right)\sum_{i=1}^n c_i\sin\left(\omega t_i+\varphi\right)-\omega\sum_{i=1}^n c_i\cos\left(\omega t_i+\varphi\right)}{\left(\omega^2+\beta^2\left(2q-h-1\right)^2\right)\left(s+\beta h\right)} + - \frac{\gamma_{q,j}\beta\left(2q-h-1\right)\left(\beta\left(2q-h-1\right)\sin\left(\omega d+\varphi\right)-\omega\cos\left(\omega d+\varphi\right)\right)e^{-\beta(2q-h-1)(t_i-d)}}{\left(\omega^2+\beta^2\left(2q-h-1\right)^2\right)\left(s+\beta h\right)} \tag{4.83}$$

$$H_q\left(s\right) = \frac{\left(-1\right)^q\beta^{2q-1}}{\prod_{i=0}^{2q-1}\left(\beta i+s\right)}\left(\sum_{i=1}^{n} c_i A_i^u\left(s+\beta\left(2q-1\right)\right)\right) \tag{4.84}$$

$$= \left(-1\right)^q\beta^{2q-1}\frac{\left(s+\beta\left(2q-1\right)\right)\sum_{i=1}^n c_i\sin\left(\omega t_i+\varphi\right)-\omega\sum_{i=1}^n c_i\cos\left(\omega t_i+\varphi\right)}{\left(\omega^2+\left(s+\beta\left(2q-1\right)\right)^2\right)\prod_{i=0}^{2q-1}\left(\beta i+s\right)} - \left(-1\right)^q\beta^{2q-1}\frac{\left(\left(s+\beta\left(2q-1\right)\right)\sin\left(\omega d+\varphi\right)-\omega\cos\left(\omega d+\varphi\right)\right)T_i^u\left(s\right)}{\left(\omega^2+\left(s+\beta\left(2q-1\right)\right)^2\right)\prod_{i=0}^{2q-1}\left(\beta i+s\right)} \tag{4.85}$$

Even in this case, the transfer function $H_q\left(s\right)$ contains the non-rational term $T_q^u\left(s\right)$ composed by a sum of weighted input-output delays.

4.5.5 Negative exponential input

Let $u(t)$ be a negative exponential

$$u(t) = \begin{cases} e^{-bt} & \text{if } t \geq d \\ 0 & \text{if } t < d \end{cases} \tag{4.86}$$

where $b \in \mathbb{R}$ with $b > 0$ is the decay velocity over time.

In this case, using Assumption 4.3, we have:

$$A_i^u(x) = \int_d^{t_i} u(t)\, e^{x(t-t_i)}\, dt \tag{4.87}$$

$$= e^{-xt_i} \int_d^{t_i} e^{-bt} e^{xt}\, dt \tag{4.88}$$

$$= e^{-xt_i} \left[\frac{e^{(x-b)t}}{x-b} \right]_d^{t_i} \tag{4.89}$$

$$= e^{-xt_i} \left[\frac{e^{(x-b)t_i} - e^{(x-b)d}}{x-b} \right] \tag{4.90}$$

$$= \frac{e^{-bt_i}}{x-b} - \frac{e^{-x(t_i-d)}}{x-b} \tag{4.91}$$

Stability check From (4.91), it is straightforward to check the condition of Theorem 4.5. In particular, we have:

$$A_i^u(s + \beta(2q-1)) = \frac{e^{-bt_i}}{s + \beta(2q-1) - b} - \frac{e^{-x(t_i-d)}}{s + \beta(2q-1) - b} \tag{4.92}$$

$$= \frac{e^{-bt_i}}{s + \beta(2q-1) - b} - e^{-s(t_i-d)} \frac{e^{-\beta(2q-1)(t_i-d)}}{s + \beta(2q-1) - b} \tag{4.93}$$

again this is the sum of two transfer functions that share the same pole in

$$p = b - \beta(2q-1); \tag{4.94}$$

this pole is strictly negative if and only if:

$$\beta > \frac{b}{2q-1} \tag{4.95}$$

therefore, the identified transfer function is not guaranteed to be asymptotically stable for every value of the hyper-parameters. In particular, we have to respect the condition (4.95).

Identified transfer function Applying Theorem 4.4 and using (4.91), it is straightforward to compute the identified transfer function $\hat{G}^u$.

$$Q^u_{q,h}(s) = \frac{\gamma_{q,j}}{s+\beta h}\sum_{i=1}^{n} c_i A^u_i\left(\beta\left(2q-h-1\right)\right) \tag{4.96}$$

$$= \frac{\gamma_{q,h}}{s+\beta h}\sum_{i=1}^{n} c_i\left(\frac{e^{-bt_i}}{\beta\left(2q-h-1\right)-b}-\frac{e^{-\beta(2q-h-1)(t_i-d)}}{\beta\left(2q-h-1\right)-b}\right) \tag{4.97}$$

$$= \frac{\gamma_{q,h}}{\left(\beta\left(2q-h-1\right)-b\right)\left(s+\beta h\right)}\sum_{i=1}^{n} c_i\left(e^{-bt_i}-e^{-\beta(2q-h-1)(t_i-d)}\right) \tag{4.98}$$

$$H^u_q(s) = \frac{(-1)^q\beta^{2q-1}}{\prod_{i=0}^{2q-1}\left(\beta i+s\right)}\left(\sum_{i=1}^{n} c_i A^u_i\left(s+\beta\left(2q-1\right)\right)\right) \tag{4.99}$$

$$= \frac{(-1)^q\beta^{2q-1}}{\prod_{i=0}^{2q-1}\left(\beta i+s\right)}\sum_{i=1}^{n} c_i\left(\frac{e^{-bt_i}}{s+\beta\left(2q-1\right)-b}-e^{-s(t_i-d)}\frac{e^{-\beta(2q-1)(t_i-d)}}{s+\beta\left(2q-1\right)-b}\right) \tag{4.100}$$

$$= \frac{(-1)^q\beta^{2q-1}\sum_{i=1}^{n} c_i e^{-bt_i}}{\left(s+\beta\left(2q-1\right)-b\right)\prod_{i=0}^{2q-1}\left(\beta i+s\right)}+ \tag{4.101}$$

$$-\frac{(-1)^q\beta^{2q-1}\sum_{i=1}^{n} c_i e^{-s(t_i-d)}e^{-\beta(2q-1)(t_i-d)}}{\left(s+\beta\left(2q-1\right)-b\right)\prod_{i=0}^{2q-1}\left(\beta i+s\right)}$$

$$= \frac{(-1)^q\beta^{2q-1}\sum_{i=1}^{n} c_i e^{-bt_i}}{\left(s+\beta\left(2q-1\right)-b\right)\prod_{i=0}^{2q-1}\left(\beta i+s\right)} \tag{4.102}$$

$$-\frac{(-1)^q\beta^{2q-1}}{\left(s+\beta\left(2q-1\right)-b\right)\prod_{i=0}^{2q-1}\left(\beta i+s\right)}T^u_q(s)$$

here, we can see that $H^u_q(s)$ depends on the non-rational term $T^u_q(s)$.

4.6 Padé approximant for a weighted sum of delays

In the various examples analyzed before (in Subsections 4.5.1, 4.5.2, 4.5.3, 4.5.4 and 4.5.5), the identified transfer function is not rational. In particular, all these models contain a term that is a weighted sum of n input-output delays. For this reason, we can consider the following non-parametric transfer function

$$T(s) = \sum_{i=1}^{n}\alpha_i e^{-s(t_i-d)} \tag{4.103}$$

where $\alpha_i \in \mathbb{R}$, with $i = 1,\ldots,n$, are coefficients that depend on the input used and the stable-spline order.

These type of transfer functions are not very easy to manage and, in general, classical dimensional reduction algorithms, such as the balance reduction [126], does not work on these type of models. For this reason, it is useful to develop a way to find a rational approximation $\widetilde{T}(s)$ of $T(s)$. This is achieved using a Padé approximant [90].

The padé approximant of a time delay transfer function, i.e. $e^{-s\tau}$ is well known and studied [3, 67], but, in this case, we need to approximate a weighted sum of delays. In general, the sum of the Padé approximation is not the Padé approximant of the sum. For this reason, it is convenient to derive the Padé approximant for the non-rational transfer function (4.103). This is achieved by the following Theorem.

Theorem 4.6. *Given the function $T(s)$, as introduced in* (4.103), *then its Padé approximant centered around* 0 *with $z \in \mathbb{N} \setminus \{0\}$ poles and z zeros is given by:*

$$\widetilde{T}(s) = \frac{N(s)}{D(s)} = \frac{\sum_{j=0}^{z} n_j \cdot s^j}{1 + \sum_{j=1}^{z} d_j \cdot s^j} \tag{4.104}$$

where the coefficients $\boldsymbol{n} = [n_0, \dots, n_z] \in \mathbb{R}^{z+1\times 1}$ and $\boldsymbol{d} = [d_1, \dots, n_z] \in \mathbb{R}^{z\times 1}$ can be computed as

$$\boldsymbol{d} = \boldsymbol{A}^{-1}\boldsymbol{b}_2 \tag{4.105}$$

$$\boldsymbol{n} = \boldsymbol{b}_1 + \boldsymbol{L}\boldsymbol{d} \tag{4.106}$$

where

$$\boldsymbol{A} = \begin{bmatrix} a_z & a_{z-1} & \cdots & a_1 \\ a_{z+1} & a_z & \cdots & a_2 \\ \vdots & \vdots & \vdots & \vdots \\ a_{2z-1} & a_{2z-2} & \cdots & a_z \end{bmatrix} \in \mathbb{R}^{z\times z} \tag{4.107}$$

$$\boldsymbol{L} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ a_0 & 0 & 0 & 0 \\ a_1 & a_0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots \\ a_{z-1} & a_{z-2} & \cdots & a_0 \end{bmatrix} \in \mathbb{R}^{z+1\times z} \tag{4.108}$$

$$\boldsymbol{b}_1 = \begin{bmatrix} a_0 & a_1 & \cdots & a_z \end{bmatrix}^\top \in \mathbb{R}^{z+1\times 1} \tag{4.109}$$

$$\boldsymbol{b}_2 = -\begin{bmatrix} a_{z+1} & a_{z+2} & \cdots & a_{2z} \end{bmatrix}^\top \in \mathbb{R}^{z\times 1} \tag{4.110}$$

and

$$a_j = \frac{1}{j!}\sum_{i=1}^{n} \alpha_i (d - t_i)^j \in \mathbb{R} \tag{4.111}$$

Proof. See Section 4.10 on page 125. ■

Remark 4.7. More generally, it is possible to compute the Padé approximant with a different number of zeros and poles, but for this particular application is not strictly necessary to generalize to this case.

Remark 4.8. The denominator coefficients are computed by solving the square linear system (4.105) of order z. This is a Toeplitz system and it can be solved efficiently with the Levinson algorithm [52] or some of its variants [38, 134] that have a quadratic computational complexity, i.e. $O(z^2)$.

In order to compute this approximation, we need to select the number of poles z of the approximant. In Theorem 1.1.1 of [7], it is shown that:

$$T(s) - \widetilde{T}(s) = o\left(s^{2z+1}\right) \tag{4.112}$$

therefore, larger z defines better approximation around 0. However, larger orders create larger systems (4.105) and the system (4.105) tends to becomes ill-conditioned and therefore hard to solve reliably. For this reason, there is a trade-off between the approximant performance and its computation. A second problem is the stability of approximant because, in this procedure, there is no guarantee that $\widetilde{T}(s)$ is stable for every number of poles z. The solution to these problems is left for future research. Here, the author proposes the trivial Algorithm 4.1, based on trial and error, to select this parameter.

Algorithm 4.1: Compute Padé approximant

```
Input: $b_i$ with $i = 1, \dots, n$
Input: $c_i$ with $i = 1, \dots, n$
Input: $z_{opt} \in \mathbb{N} \setminus \{0\}$
1  cont ← True;
2  $z \leftarrow z_{opt}$;
3  while cont do
4      Compute $\boldsymbol{d}$ and $\boldsymbol{n}$ with $z$ poles using the coefficients $b_i$ and $c_i$;
5      if $\widetilde{T}(s)$ is asymptotically stable then
6          cont ← False;
7      else
8          $z \leftarrow z - 1$;
9      end if
10 end while
Output: The vector $\boldsymbol{d}$
Output: The vector $\boldsymbol{n}$
```

To show the performance of this approximation, consider the following example.

Example 4.1: Padé approximant of a toy example

Consider the case where the input is a step signal, then the non-rational part is equal to

$$T_q^u(s) = \sum_{i=1}^{n} c_i e^{-\beta(2q-1)(t_i-d)} e^{-s(t_i-d)} \tag{4.113}$$

Suppose that, after the identification procedure, we obtain:

$$q = 2 \tag{4.114}$$

$$n = 5 \tag{4.115}$$

$$d = 0 \tag{4.116}$$

$$\boldsymbol{t} = \begin{bmatrix} 0.25 & 0.38 & 0.62 & 0.85 & 1 \end{bmatrix} \in \mathbb{R}^{1\times 5} \tag{4.117}$$

$$\boldsymbol{c} = \begin{bmatrix} 6 & -5 & 3 & 1 & -2 \end{bmatrix} \in \mathbb{R}^{1\times 5} \tag{4.118}$$

If we feed this system with an input signal $\bar{u}(t)$ the output is

$$\begin{aligned}\hat{y}(t) &= \sum_{i=1}^{n} c_i e^{-3t_i} \bar{u}(t-t_i) && (4.119)\\ &= 6e^{-0.75}\bar{u}(t-0.25) - 5e^{-1.14}\bar{u}(t-0.38) + 3e^{-1.86}\bar{u}(t-0.62) && (4.120)\\ &\quad + e^{-2.5}\bar{u}(t-0.85) - 2e^{-3}\bar{u}(t-1) && (4.121)\end{aligned}$$

In this case the coefficients a_j are

$$\begin{aligned}a_j &= \frac{1}{j!}\sum_{i=1}^{n} c_i e^{-3\beta t_i} (-t_i)^j && (4.122)\\ &= \frac{(-1)^j}{j!}\left(0.25^j e^{-0.75} - 0.38^j e^{-1.14} + 0.62^j e^{-1.86} + 0.85^j e^{-2.5} - e^{-3}\right) && (4.123)\end{aligned}$$

In Figure 4.1, it is possible to see a comparison between the true response, described before, and the response of the approximated model with different orders using the input signal

$$\bar{u}(t) = \begin{cases} \sin\left(\sqrt{t}\right) & t \geq 0 \\ 0 & t < 0 \end{cases} \qquad (4.124)$$

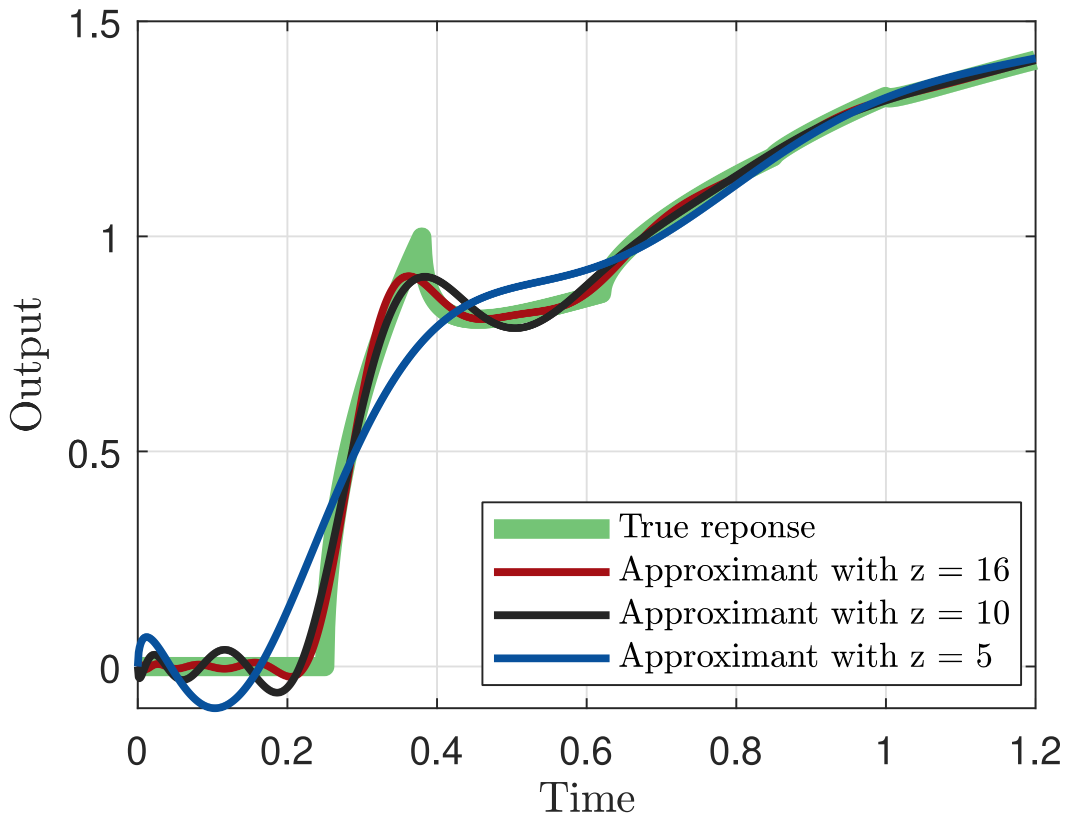

FIGURE 4.1: Comparison of the true output with the output of the Padé approximant with different orders.

4.7 Identification with more complex input signals

The proposed method is not trivial to implement because it requires the analytical solution of the integrals (4.10) and (4.33). In Section 4.5, we have reported the solutions of these integrals for some common simple inputs, but in practice, a lot of different and more complicated input can be used. Furthermore, in some application, the shape of the excitation signal $u(t)$ is not known and its sampled alongside the output. In these cases, the approach explained before is not directly usable.

In order to tackle these problems, consider the following input signal

$$u(t) = \sum_{p=1}^{m} a_p u_p(t) \tag{4.125}$$

where $m \in \mathbb{N} \setminus \{0\}$, $a_p \in \mathbb{R}$ with $p = 1, \dots, m$ and $u_p : \mathbb{R}_+ \to \mathbb{R}$ is a simpler signal. This is a very general input that can be used in a lot of different situations. To better understand this concept consider the following examples.

Example 4.2: Multisine input

The multi-sine, a very common input signal used in frequency-domain system identification (definition 5.4 of [98]), can be seen as a weighted sum of weighted sinewaves. In particular, we can write

$$u(t) = \sum_{p=0}^{m} a_p \sin(2\pi p f_0 t + \varphi_p) \tag{4.126}$$

where $f_0 \in \mathbb{R}_+$ is the fundamental frequency, pf_0 with $p = 1, \dots, m$ are the excited frequencies, $\varphi_p \in \mathbb{R}$ with $p = 1, \dots, m$ are the phased of the excited frequencies and $a_p \in \mathbb{R}$ with $p = 1, \dots, m$ their amplitude.

Example 4.3: Polynomial input

olynomial signals can be seen as a weighted sum of monomials

$$u(t) = \sum_{p=0}^{m} a_p z^p. \tag{4.127}$$

Example 4.4: Zero-order Holder input

If the input signal is not known, but it is sampled alongside the output, a possible solution is to use a Zero Order Holder (ZOH) to convert the samples in an approximation of the continuous signal. In particular, the approximated signal can be written as

$$u(t) = \sum_{i=1}^{n} \begin{cases} u_i - u_{i-1} & t \geq t_i \\ 0 & t < t_i \end{cases} \tag{4.128}$$

where u_i with $i = 1, \dots, n$ are the inputs samples and $u_0 = 0$. Now, it is trivial to see that this signal is a sum of n steps with different amplitudes and different starting times.

In these cases, $u(t)$ can inherit the properties of the various $u_p(t)$ signals thanks to the following theorems.

Theorem 4.7. *Let the input $u(t)$ be of the form* (4.125). *The operator $A_i^u(x)$, as described in* (4.33)*, is equal to*

$$A_i^u(x) = \sum_{p=1}^{m} a_p A_i^{u_p}(x) \tag{4.129}$$

Proof. See Section 4.10 on page 126. ■

Theorem 4.8. *Let the input $u(t)$ be of the form* (4.125). *If $\hat{G}^{u_p}(s)$ is asymptotically stable for $p = 1, \dots, m$ then $\hat{G}^u(s)$ is asymptotically stable.*

Proof. See Section 4.10 on page 126. ■

Theorem 4.9. *Let the input $u(t)$ be of the form* (4.125). *The derived kernel $o^u(t_i, t_j)$, as described in* (4.10)*, is equal to*

$$o^u(t_i, t_j) = \sum_{p_1=1}^{m} \sum_{p_2=1}^{m} a_{p_1} a_{p_2} o^{u_{p_1}, u_{p_2}}(t_i, t_j) \tag{4.130}$$

where

$$o^{u_{p_1}, u_{p_2}}(t_i, t_j) = \int_0^{+\infty} u_{p_1}(t_i - \psi) \left(\int_0^{+\infty} u_{p_2}(t_j - \xi) k(\psi, \xi)\, d\xi \right) d\psi \tag{4.131}$$

Proof. See Section 4.10 on page 127. ■

These theorems provide a way to handle complex signals by working on their simpler components. For instance, the examples before-mentioned are all composed by signals that were analyzed in Section 4.5 and therefore their implementation is straightforward.

According to Theorem 4.9, it is necessary to compute the term $o^{u_{p_1}, u_{p_2}}$ for any combination of the two inputs in the sum. To do so, it is possible to generalize Theorem 4.2 to the case where we need to compute $o^{u_{p_1}, u_{p_2}}$ thanks to the following Theorem.

Theorem 4.10. *Let:*

- *the kernel k be a stable-spline of order q;*
- *$u_1 : \mathbb{R} \to \mathbb{R}$ be an input signal such that $u_1(t) = 0, \forall t \le d_1$;*
- *$u_2 : \mathbb{R} \to \mathbb{R}$ be an input signal such that $u_2(t) = 0, \forall t \le d_2$.*

Then

$$o_q^{u_1, u_2}(t_i, t_j) = \lambda \sum_{h=0}^{q-1} \gamma_{q,h} \begin{cases} r_{q,h}^{u_1, u_2}(t_i, t_j) + w_{q,h}^{u_1, u_2}(t_i, t_j) & t_i - d_1 \le t_j - d_2 \\ r_{q,h}^{u_2, u_1}(t_j, t_i) + w_{q,h}^{u_2, u_1}(t_j, t_i) & t_i - d_1 > t_j - d_2 \end{cases} \tag{4.132}$$

where

$$r^{u_1,u_2}_{q,h}(t_i,t_j) = \int_{d_1}^{t_i} \int_{d_2}^{t_j - t_i + \xi} u_1(\xi)\, u_2(\psi)\, e^{-\beta[(2q-h-1)(t_j-\psi)+h(t_i-\xi)]}\, d\psi\, d\xi \tag{4.133}$$

$$w^{u_1,u_2}_{q,h}(t_i,t_j) = \int_{d_1}^{t_i} \int_{t_j - t_i + \xi}^{t_j} u_1(\xi)\, u_2(\psi)\, e^{-\beta[(2q-h-1)(t_i-\xi)+h(t_j-\psi)]}\, d\psi\, d\xi \tag{4.134}$$

Proof. See Section 4.10 on page 119. ■

4.8 Summary of the proposed algorithm

In order to implement the proposed algorithm, it is necessary to do some mathematical computation based on the excitation signal used for the experiment. In particular, it is necessary to compute two formulas

- The derived kernel o^u as described in (4.10). If the kernel used is a stable-spline then the Theorem 4.2 can be helpful.
- The identified transfer function $\hat{G}_u$ as described in Theorem 4.3. If the kernel used is a stable-spline then the Theorem 4.4 can be helpful.

For some common inputs the $\hat{G}_u$ is reported in Section 4.5 and in case of more complex inputs the theorems 4.7 and 4.9 can be useful. In the case of sampled inputs and $u(t)$ unknown, it is possible to interpolate the samples in some way. A possible method is reported in Example 4.4, but there are other possibilities, such an higher-order holder or the Whittaker–Shannon interpolation formula.

Given these two formulas the non-parametric system identification is carried out following Algorithm 4.2. The transfer function returned by this algorithm can be non-rational, as explained in Section 4.6, and it is possible to use Algorithm 4.1 in order to find a rational approximation. In the end, it is possible to reduce the dimension of the estimated rational model by using some dimensional reduction algorithm, such as the balance reduction [126].

Algorithm 4.2: Non-parametric transfer function identification

```
Input: The dataset $\mathcal{D}$
Input: A way to compute the function $o^u$ given $\boldsymbol{\zeta} = [\lambda, \beta, \tau, q]$ and two time instants
Input: A way to compute $\hat{G}^u$ given $\boldsymbol{\zeta} = [\lambda, \beta, \tau, q]$ and $\boldsymbol{c}$
1 Discard the part of the dataset $\mathcal{D}$ corresponding to time instants $t_i \leq d$ (see Section 4.4
    for more details)
2 Find the optimal hyper-parameters $\widetilde{\boldsymbol{\zeta}}$ by minimizing equation (4.13) (see Section 4.2 for
    more details)
3 Compute the matrix $\boldsymbol{O}$ using the hyper-parameters $\widetilde{\boldsymbol{\zeta}}$
4 Compute a valid solution $\boldsymbol{c}$ of the linear system (4.8)
5 Compute $\hat{G}^u$ given $\widetilde{\boldsymbol{\zeta}}$ and $\boldsymbol{c}$
Output: The transfer function $\hat{G}^u$
```

The proposed algorithm requires to solve the linear system (4.8). Following the reasoning of Chapter 3, this system can have infinite equivalent solutions even when the dataset respects

Assumption 4.3. It is advisable to use the LN1 solution explained in 3.3 in order to minimize the length of the vector $\boldsymbol{c}$. This simplifies the computational complexity of the next steps in a significant way. The Padé approximation is also more computationally reliable with a smaller coefficient vector.

Another important thing to keep in mind during the implementation of this algorithm is to try to minimize the number of transfer functions summed during the computation of $\hat{G}_u$ because the symbolic algorithms used to execute this sum are very computationally expensive and can make significant errors with a large number of addends.

4.9 Numerical results

To better understand the proposed method, consider the following transfer function models.

$$\mathscr{G}_1 \ : \ G_1(s) = -\frac{27}{20}\frac{2000s^3 + 3600s^2 + 2095s + 396}{1350s^4 + 7695s^3 + 12852s^2 + 7796s + 1520} \tag{4.135}$$

$$\mathscr{G}_2 \ : \ G_2(s) = 1600\frac{1 - 4s}{s^4 + 5s^3 + 408s^2 + 416s + 1600} \tag{4.136}$$

$$\mathscr{G}_3 \ : \ G_3(s) = -\frac{1}{10}\frac{1869s^4 + 17400s^3 + 68220s^2 + 72350s + 5075}{1000s^5 + 4419s^4 + 14160s^3 + 27180s^2 + 22220s + 5168} \tag{4.137}$$

The fundamental properties of these three systems are reported in Table 4.1, their Bode diagrams are presented in Figure 4.3 and their impulse response can be seen in Figure 4.2.

The model $\mathscr{G}_1$ is a simple model with 4 real poles and 3 real zeros that behaves like a low-pass filter with a negative gain, as shown in its Bode diagram. This results in a very smooth impulse response.

The second one $\mathscr{G}_2$ is a famous benchmark system used for continuous-time system identification called *Rao-Garnier system* [70]. This system is characterized by two couples of complex-conjugate poles that generate two different resonances. These system produces an oscillating impulse response of the system.

The third system has both a couple of conjugate complex poles and a couple of conjugate complex zeros. This results in a strange frequency response behavior, as shown in its Bode diagram, and an oscillating impulse response.

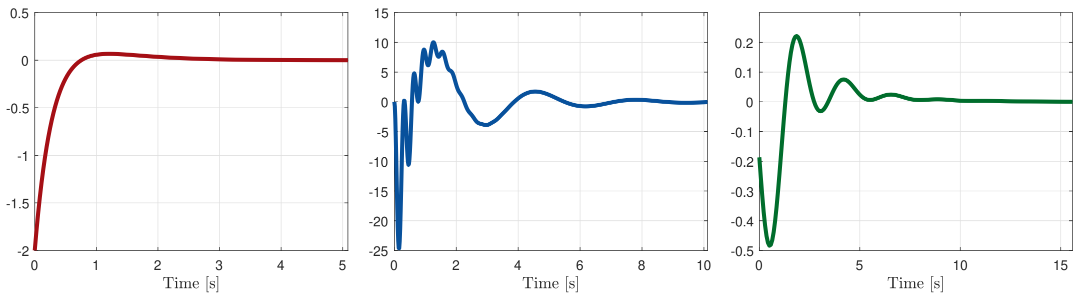

FIGURE 4.2: Impulse response of the three models used in the simulations. From left to right: $\mathscr{G}_1$, $\mathscr{G}_2$ and $\mathscr{G}_3$.

In the simulation described in this section, the starting Padé approximant order is 25 and the kernel used is a stable-spline whose order q is searched between 1 and 5.

System	Order	Static gain	Poles	Zeros
$\mathscr{G}_1$	4	$-\frac{2673}{7600} \simeq -1.98$	$p_1 \simeq -0.63$	$z_1 \simeq -0.55$
			$p_2 \simeq -0.4$	$z_2 \simeq -0.45$
			$p_3 \simeq -3.33$	$z_3 \simeq -8$
			$p_4 \simeq -1.33$	
$\mathscr{G}_2$	4	1	$p_1 \simeq -2 - j19.90$	$z_1 = 0.25$
			$p_2 \simeq -2 + j19.90$	
			$p_3 \simeq -0.5 - j1.94$	
			$p_4 \simeq -0.5 + j1.94$	
$\mathscr{G}_3$	5	$-\frac{1100}{421} \simeq 2.61$	$p_1 \simeq -0.63 - j2.61$	$z_1 \simeq -3.88 - j3.05$
			$p_2 \simeq -0.63 + j2.61$	$z_2 \simeq -3.88 + j3.05$
			$p_3 \simeq -0.37$	$z_3 \simeq -1.48$
			$p_4 \simeq -1.29$	$z_4 \simeq -0.08$
			$p_5 \simeq -1.49$	

Table 4.1: Fundamental parameters of the three models used in the simulations.

System	T	η^2_{imp}	η^2_{step}
$\mathscr{G}_1$	$4\,s$	$2.43 \cdot 10^{-2}$	$2.78 \cdot 10^{-2}$
$\mathscr{G}_2$	$12\,s$	$2.72 \cdot 10^{0}$	$6.80 \cdot 10^{-1}$
$\mathscr{G}_3$	$15\,s$	$2.69 \cdot 10^{-3}$	$6.74 \cdot 10^{-3}$

Table 4.2: Parameters of the two tests that change based on the system used.

4.9.1 Identification using impulse-response data

To evaluate the performance of the method when dealing with impulse response data consider the following dataset

$$\mathcal{D}_1 = \{(t_i, y_i) \,|i = 1, \ldots, 100\} \tag{4.138}$$

sampled from the probabilistic model

$$t_i \sim \mathcal{U}\left(0, T\right) \qquad i = 1, \ldots, 100 \tag{4.139}$$

$$e_i \sim \mathcal{N}\left(0, \eta^2_{imp}\right) \qquad i = 1, \ldots, 100 \tag{4.140}$$

$$y_i = r_\delta\left(t_i\right) + e_i \qquad i = 1, \ldots, 100 \tag{4.141}$$

where t_i and e_i, with $i = 1, \ldots, 100$, are all independent random variables and the function r_δ is the impulse response of the system. The value η^2_{imp} and T change for every system and their value is reported in Table 4.2. In particular, η^2_{imp} is chosen in order to obtain a

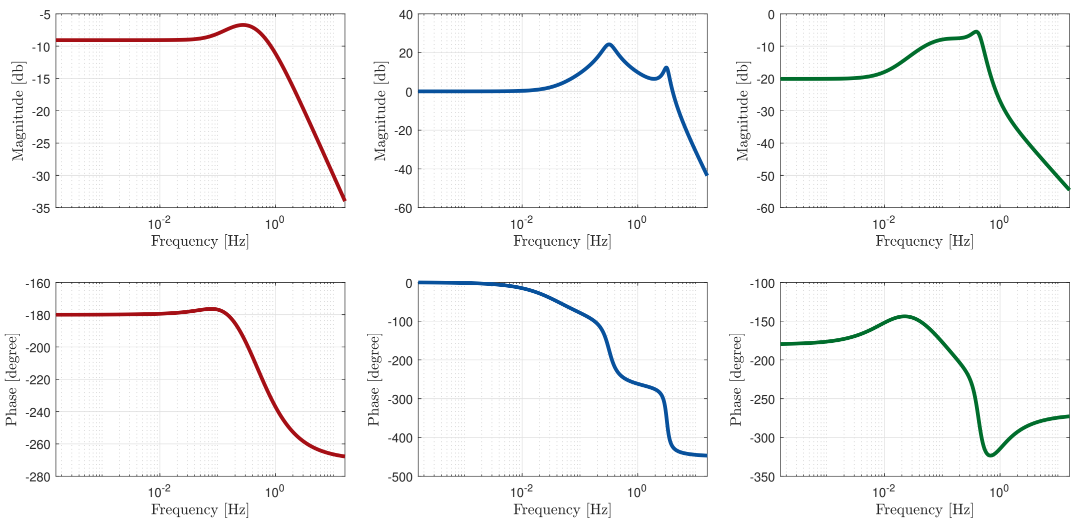

FIGURE 4.3: Bode diagrams of the three models used in the simulations. From left to right: $\mathscr{G}_1$, $\mathscr{G}_2$ and $\mathscr{G}_3$.

SNR of 5.

In Figure 4.4, it is possible to see the true impulse response compared with 100 different estimated impulse responses obtained with 100 different datasets $\mathcal{D}_1$ sampled as explained before. Analogously, in Figure 4.5 the same thing can be seen in the frequency domain.

From these graphs, it is possible to notice that the method works very well for the smooth systems $\mathscr{G}_1$ and $\mathscr{G}_3$, but it show worse performance on the fast varying system $\mathscr{G}_2$. This is due to the fact that the stable-splines, naturally, are not the most suitable kernel choice when dealing with more oscillating outputs.

A second reason, for this behavior, is the noise. In fact, the noise variance is higher than the fast oscillation of $\mathscr{G}_2$ and therefore, the optimization procedure tends to remove them in the estimation. This can be seen in the Bode diagram, where the second resonance is completely ignored by the estimated systems.

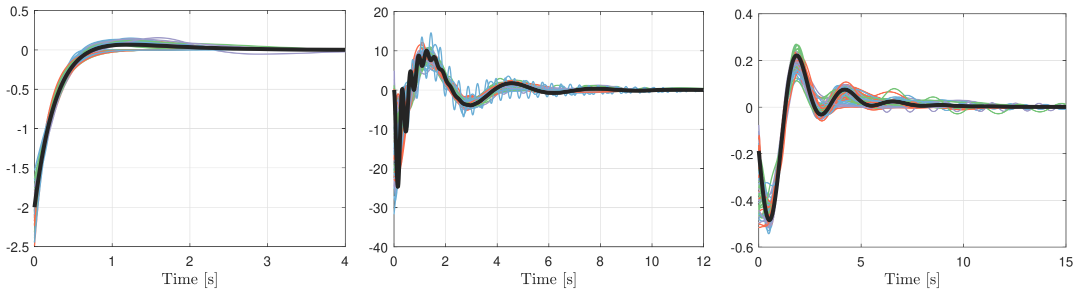

FIGURE 4.4: Impulse response of the true system (black line) compared with 100 different estimations (colored lines) obtained using impulse response data. From left to right: $\mathscr{G}_1$, $\mathscr{G}_2$ and $\mathscr{G}_3$.

In order to have a more quantitative measure of the performance, it is possible to compare the output of the true model with the one of the estimated model on a test dataset. This new dataset is obtained using a random White Gaussian Noise with 10 Hz of bandwidth as excitation signal. Both input and output are sampled regularly with the sampling frequency

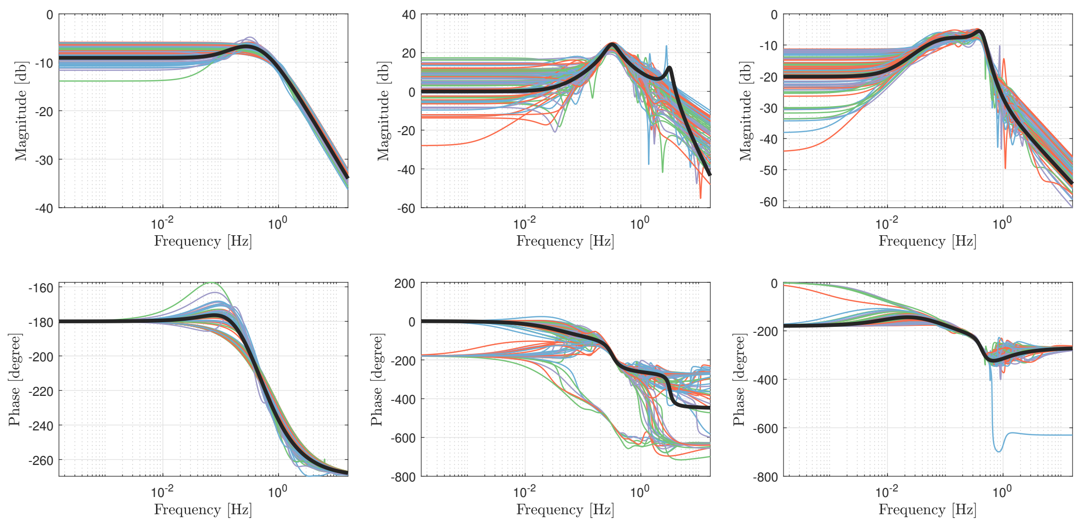

FIGURE 4.5: Bode diagrams of the true system (black line) compared with 100 different estimations (colored lines) obtained using impulse response data. From left to right: $\mathscr{G}_1$, $\mathscr{G}_2$ and $\mathscr{G}_3$.

of 1 kHz for 1000 s. Then the performances are computed using the following index

$$\text{Fit} = 1 - \frac{\sum_{t=1}^{n_v} (y_t - \hat{y}_t)^2}{\sum_{t=1}^{n_v} (y_t - \sum_{t=1}^{n_v} y_t)^2} \tag{4.142}$$

where n_v is the length of the obtained dataset, y_t and $\hat{y}_t$, with $t = 1, \dots, n_v$, are, respectively, the samples of the true response and the estimated one. The results can be seen in Figure 4.6, where the fit computed for 100 different datasets for each system is shown with three boxplots. Here, we can confirm the previous observations: the method works very well for the smooth systems $\mathscr{G}_1$ and $\mathscr{G}_3$, but it has worse performance on the fast varying model $\mathscr{G}_2$.

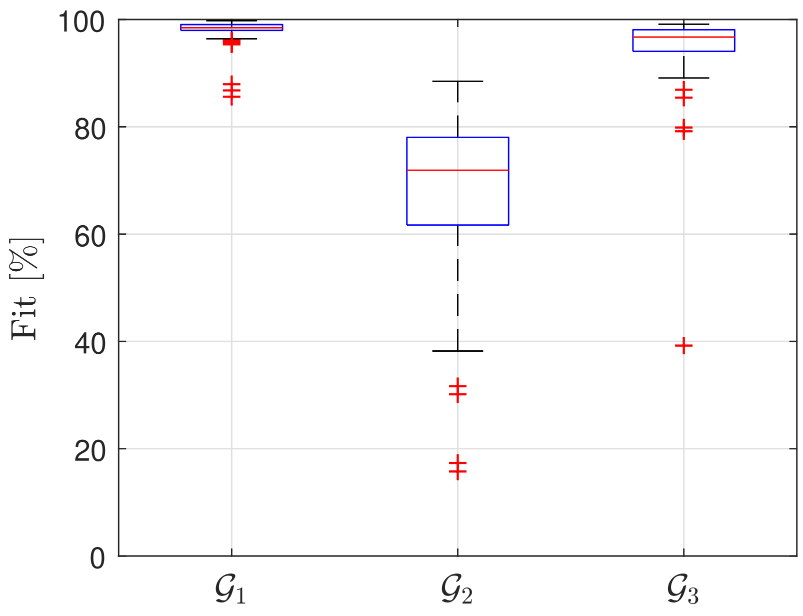

FIGURE 4.6: Boxplot of the performance on the test dataset obtained using impulse response data.

To better assess the performance with different models, a Monte Carlo simulation with a random generated asymptotically stable dynamic linear system with order 6. The results are shown in Figure 4.7, where the performances are evaluated on a test dataset generated as explained before.

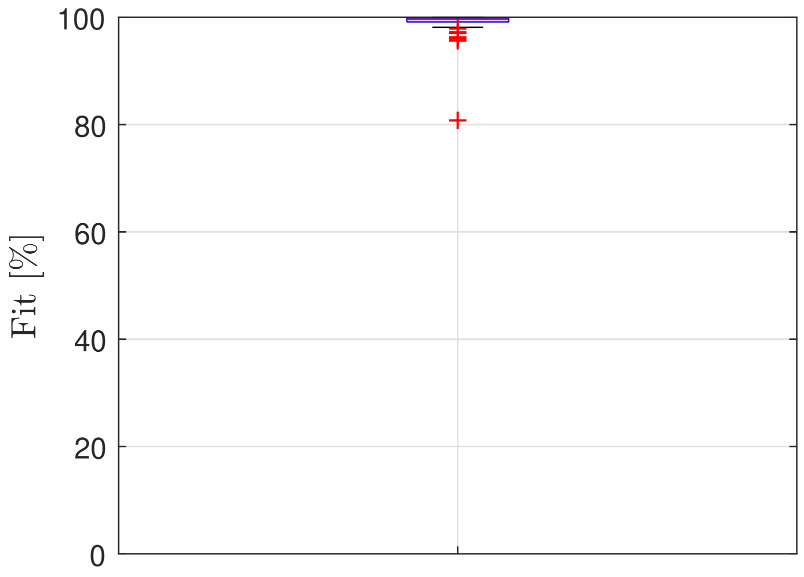

FIGURE 4.7: Boxplot of the performance on the test dataset obtained using impulse response data on randomly generated LTI models with order 6.

4.9.2 IDENTIFICATION USING STEP-RESPONSE DATA

This section is organized as the previous one, but using step response data. Therefore, consider the following dataset

$$\mathcal{D}_2 = \{(t_i, y_i)\,|i = 1, \dots, 100\} \tag{4.143}$$

sampled from the probabilistic model

$$t_i \sim \mathcal{U}(0, T) \qquad i = 1, \dots, 100 \tag{4.144}$$
$$e_i \sim \mathcal{N}\left(0, \eta_{step}^2\right) \qquad i = 1, \dots, 100 \tag{4.145}$$
$$y_i = r_{\text{step}}(t_i) + e_i \qquad i = 1, \dots, 100 \tag{4.146}$$

where t_i and e_i, with $i = 1, \dots, 100$, are all independent random variables and the function r_{step} is the impulse response of the system. The value η_{step}^2 and T change for every system and their value is reported in Table 4.2. In particular, η_{step}^2 is chosen in order to obtain a SNR equal to 5.

As before, in Figure 4.8 is possible to see the true step response compared with 100 different estimated step responses obtained with 100 different datasets $\mathcal{D}_2$ sampled as explained before. Analogously, in Figure 4.9 the same thing can be seen in the frequency domain.

From these graphs, we can see that the results are similar to the one obtained from impulse response data. The method works well for the system with a smooth impulse response $\mathscr{G}_1$ and $\mathscr{G}_3$ and it struggles with the non-smooth $\mathscr{G}_2$. The motivations for this behavior are also the same.

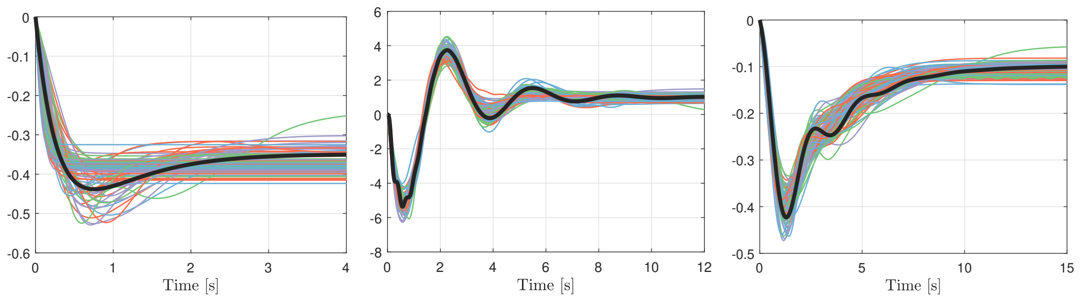

Figure 4.8: Step response of the true system (black line) compared with 100 different estimations (colored lines) obtained using step response data. From left to right: $\mathscr{G}_1$, $\mathscr{G}_2$ and $\mathscr{G}_3$.

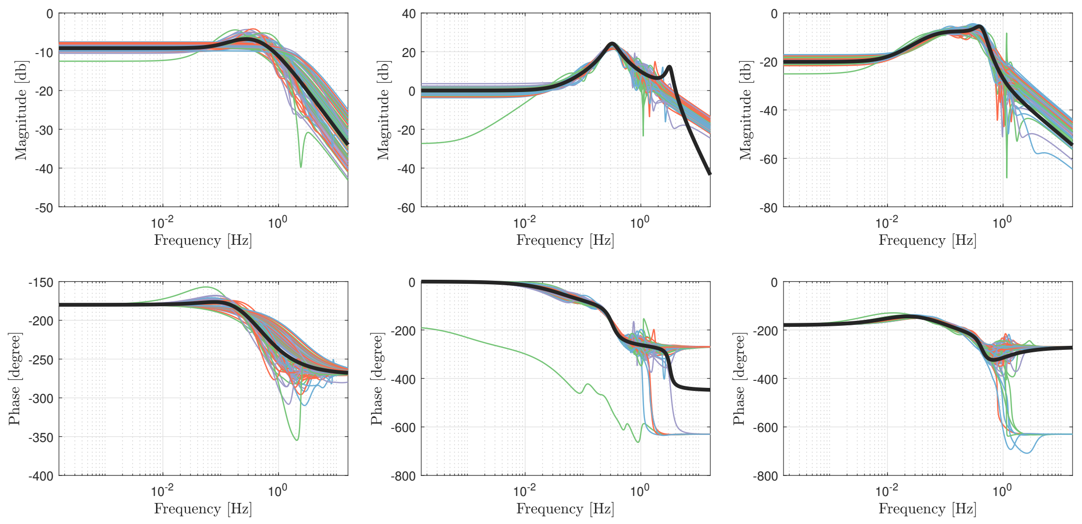

Figure 4.9: Bode diagrams of the true system (black line) compared with 100 different estimations (colored lines) obtained using step response data. From left to right: $\mathscr{G}_1$, $\mathscr{G}_2$ and $\mathscr{G}_3$.

These observations are also confirmed by the performance on the test dataset, as shown in Figure 4.10. Where the test dataset and the fit index are the same used in the previous section.

Following the same reasoning used for the identification with impulse response data, a Monte Carlo simulation with a random generated asymptotically stable dynamic linear system with order 6. The results are shown in Figure 4.11, where the performances are evaluated on test dataset generated as explained before.

4.9.3 Dimensional reduction of the estimated model

At the start of Section 4.5, it is said that the estimated the high-dimensional transfer function can be fed to dimensional reduction algorithm. In this section, we will delve into this aspect to see the effect of the dimensional reduction on the estimated model.

The algorithm used to execute the dimensional reduction is the Balance Reduction [126]. To select the right order automatically the singular value of the Hankel matrix [48] of the estimated model is used.

To evaluate the performance of the dimensional reduction, consider the case of identification using impulse response data, as in Section 4.9.1. In Figure 4.12, it is possible to see the

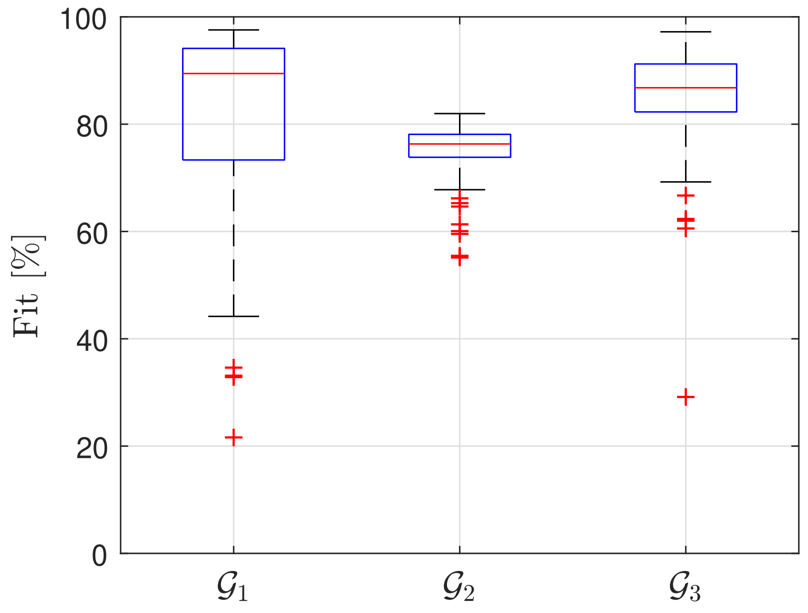

FIGURE 4.10: Boxplot of the performance on the test dataset obtained using impulse response data.

performance with different levels of reduction and in Figure 4.13, there is the histogram of the order at various levels of reductions. The level of reduction is given by the fraction of singular values of the Hankel matrix is kept.

From these graphs, it is possible to note that the performance does not decrease significantly because the estimated model has a very large number of redundant modes. For this reason, even when we take 99% of the singular values the order decrease significantly.

4.9.4 Comparison with the state of the art

In the last decades, continuous-time system identification was studied in detail [46, 47] and many methods were developed. In particular, the most recent methods are implemented in the `CONTSID toolbox` [45, 91] using MatLab. In this section, a comparison between the proposed kernel method and the `SRIVC` method [46, 136, 137] (using the implementation of the before-mentioned toolbox). This method requires the knowledge of the order of the system under analysis. In this comparison, the Young Information Criterion (YIC) [137] method is used to select the best order among a pool of candidates. This is implemented in the toolbox in the functions `srivcstruc` and `selcstruc`.

The comparison was done on the three models introduced at the start of this section in the following settings:

- the input signal is a step;
- the dataset is composed by 250 output measurements taken between 0 and T where T changes for every model as reported in Table 4.2;
- the dataset is sampled regularly;
- the measurements noise has variance η^2_{step} where η^2_{step} changes for every model as reported in Table 4.2 (the SNR is always equal to 5);
- the pool of possible number of poles for the `SRIVC` method is $\{1, 2, 3, 4, 5, 6\}$;

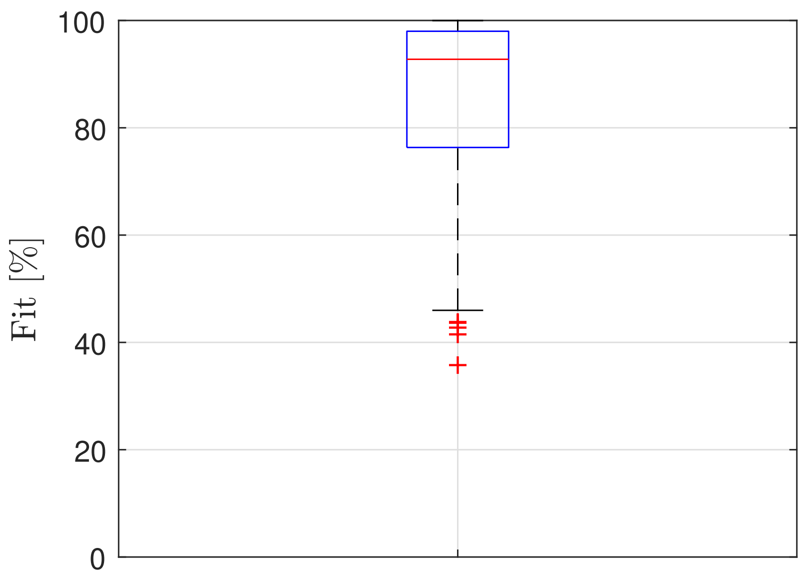

FIGURE 4.11: Boxplot of the performance on the test dataset obtained using step response data on randomly generated LTI models with order 6.

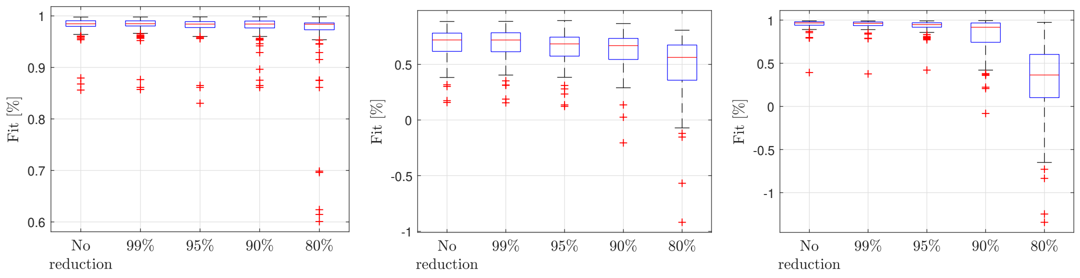

FIGURE 4.12: Boxplot of the performance on the test dataset using different level of dimensional reduction on the three benchmark models. The systems used are, from left to right, $\mathscr{G}_1$, $\mathscr{G}_2$ and $\mathscr{G}_3$.

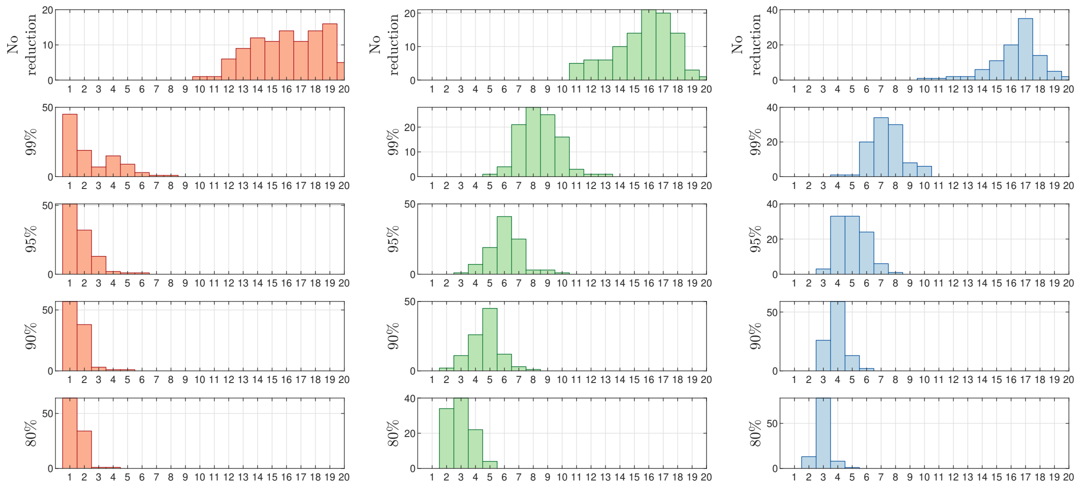

FIGURE 4.13: Histogram of the order of the estimated system at different level of dimensional reduction on the three benchmark models. The systems used are, from left to right, $\mathscr{G}_1$, $\mathscr{G}_2$ and $\mathscr{G}_3$.

- the pool of possible number of zeros for the SRIVC method is $\{1, 2, 3, 4, 5, 6\}$;

The results of a Monte Carlo simulation with 100 different noise values are reported in Figure 4.14, where it is clear that the proposed approach works significantly better. The second example, the Rao-Garnier system [70], is the one where the proposed kernel approach has more difficulties, but the median fit is still slightly better than the one obtained with the CONTSID toolbox.

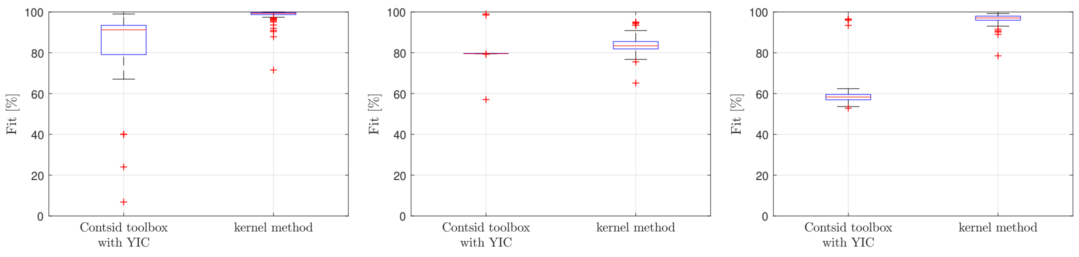

FIGURE 4.14: Comparison between the proposed method and SRIVC from the CONTSID toolbox. From left to right: $\mathscr{G}_1$, $\mathscr{G}_2$ and $\mathscr{G}_3$.

To further verify the performance of the proposed method with respect to the toolbox, it is possible to compare the performance on 100 randomly generated systems of order 6. Using the same settings as the ones reported before (with T equals to the settling time of the system and η^2_{step} selected in such a way that the SNR is 5), the obtained results are shown in Figure 4.15. Here, it is evident that the proposed approach is more robust than the one proposed in the CONTSID toolbox.

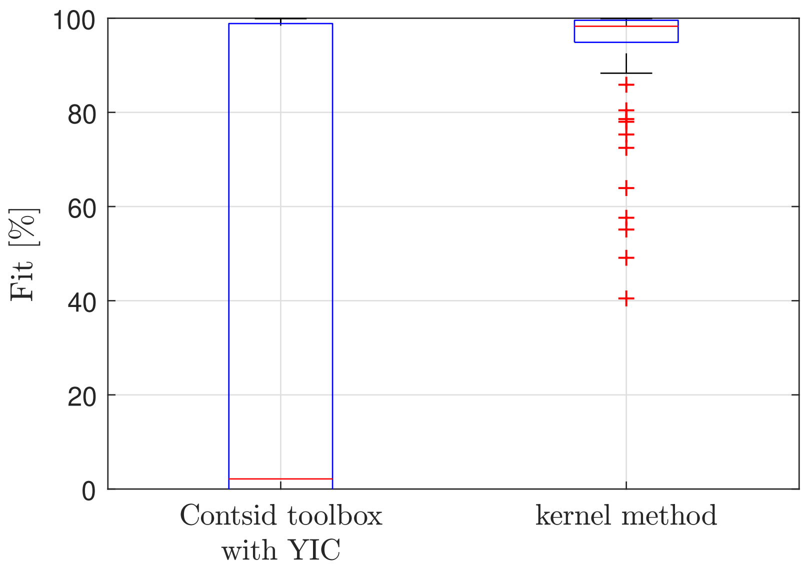

FIGURE 4.15: Comparison between the proposed method and SRIVC from the CONTSID toolbox on 100 randomly generated LTI models with order 6.

4.10 Proofs

The proofs of all the theorems presented in this chapter are reported in this section.

Proof of Theorem 4.1. First of all we can note that the term $G_q(a,x)\,G_q(b,x)$ is equal to zero when $a < x \vee b < x$, therefore

$$G_q(a,x)\,G_q(b,x) = \frac{(a-x)^{q-1}(b-x)^{q-1}}{((q-1)!)^2}\begin{cases}1 & \text{if } a \geq x \wedge b \geq x\\ 0 & \text{if } a < x \vee b < x\end{cases} \tag{4.147}$$

$$= \frac{(a-x)^{q-1}(b-x)^{q-1}}{((q-1)!)^2}\begin{cases}1 & \text{if } x \leq \min(a,b)\\ 0 & \text{if } x > \min(a,b)\end{cases} \tag{4.148}$$

following this fact, the integral (4.15) can be truncated to $\min(a,b)$:

$$s_q(a,b) = \frac{1}{((q-1)!)^2}\int_0^{\min(a,b)} (a-x)^{q-1}(b-x)^{q-1}\,dx \tag{4.149}$$

with the change of variable $y = a - x$ and some mathematical steps, we obtain

$$s_q(a,b) = \frac{1}{((q-1)!)^2}\int_a^{a-\min(a,b)} -y^{q-1}(b-(a-y))^{q-1}\,dy \tag{4.150}$$

$$= \frac{-1}{((q-1)!)^2}\int_a^{a-\min(a,b)} y^{q-1}(-(-b+a-y))^{q-1}\,dy \tag{4.151}$$

$$= \frac{-(-1)^{q-1}}{((q-1)!)^2}\int_a^{a-\min(a,b)} y^{q-1}(a-b-y)^{q-1}\,dy \tag{4.152}$$

$$= \frac{(-1)^q}{((q-1)!)^2}\int_a^{a-\min(a,b)} y^{q-1}(a-b)^{q-1}\left(1-\frac{y}{a-b}\right)^{q-1}dy \tag{4.153}$$

$$= \frac{(-1)^q(a-b)^{2q-2}}{((q-1)!)^2}\int_a^{a-\min(a,b)}\left(\frac{y}{a-b}\right)^{q-1}\left(1-\frac{y}{a-b}\right)^{q-1}dy \tag{4.154}$$

Let us first consider the case when $a = \min(a,b)$. Here it is useful to consider the change of variable

$$z = \frac{y}{a-b} \tag{4.155}$$

that transforms the integral in

$$s_q(a,b) = \frac{(-1)^q(a-b)^{2q-2}}{((q-1)!)^2}\int_{\frac{a}{a-b}}^{0} z^{q-1}(1-z)^{q-1}(a-b)\,dz \tag{4.156}$$

$$= -\frac{(-1)^q(a-b)^{2q-1}}{((q-1)!)^2}\int_0^{\frac{a}{a-b}}\left(z-z^2\right)^{q-1}dz \tag{4.157}$$

A very similar formulation can also be obtained in the case where $b = \min(a,b)$ using the change of variable

$$z = 1 - \frac{y}{a-b} \tag{4.158}$$

that allows writing

$$s_q(a,b) = \frac{(-1)^q (a-b)^{2q-2}}{((q-1)!)^2} \int_{\frac{b}{b-a}}^{0} -(1-z)^{q-1}(z)^{q-1}(a-b)\,dz \tag{4.159}$$

$$= \frac{(-1)^q (a-b)^{2q-1}}{((q-1)!)^2} \int_{0}^{\frac{b}{b-a}} \left(z-z^2\right)^{q-1} dz \tag{4.160}$$

Therefore, the kernel can be written as

$$s_q(a,b) = \frac{(-1)^q (a-b)^{2q-1}}{((q-1)!)^2} \begin{cases} -L_q\left(\frac{a}{a-b}\right) & a \leq b \\ L_q\left(\frac{b}{b-a}\right) & a > b \end{cases} \tag{4.161}$$

where

$$L_q(x) = \int_0^x \left(z-z^2\right)^{q-1} dz \tag{4.162}$$

The integral $L_q(x)$ can be easily solved using the binomial theorem

$$L_q(x) = \int_0^x \left(z-z^2\right)^{q-1} dz \tag{4.163}$$

$$= \int_0^x \left(\sum_{i=0}^{q-1} \frac{(q-1)!}{i!\,(q-1-i)!} z^{q-1-i}\left(-z^2\right)^i\right) dz \tag{4.164}$$

$$= \sum_{i=0}^{q-1} \frac{(q-1)!\,(-1)^i}{i!\,(q-1-i)!} \int_0^x z^{q-1-i+2i} dz \tag{4.165}$$

$$= \sum_{i=0}^{q-1} \frac{(q-1)!\,(-1)^i}{i!\,(q-1-i)!} \int_0^x z^{q+i-1} dz \tag{4.166}$$

$$= \sum_{i=0}^{q-1} \frac{(q-1)!\,(-1)^i}{i!\,(q-1-i)!\,(q+i)} x^{q+i} \tag{4.167}$$

obtaining

$$s_q(a,b) = \sum_{i=0}^{q-1} \frac{(-1)^{q+i}(a-b)^{2q-1}}{i!\,(q-1-i)!\,(q+i)\,(q-1)!} \begin{cases} -\left(\dfrac{a}{a-b}\right)^{q+i} & a \le b \\ \left(\dfrac{b}{b-a}\right)^{q+i} & a > b \end{cases} \tag{4.168}$$

$$= \sum_{i=0}^{q-1} \frac{(-1)^{q+i}}{i!\,(q-1-i)!\,(q+i)\,(q-1)!} \begin{cases} -(a-b)^{q-i-1}a^{q+i} & a \le b \\ (-1)^{2q-1}(b-a)^{q-i-1}b^{q+i} & a > b \end{cases} \tag{4.169}$$

considering the fact that $(-1)^{2q-1} = -1$ because $2q-1$ is always an odd number, we have

$$s_q(a,b) = \sum_{i=0}^{q-1} \frac{(-1)^{q+i-1}}{i!\,(q-1-i)!\,(q+i)\,(q-1)!} \begin{cases} (a-b)^{q-i-1}a^{q+i} & a \le b \\ (b-a)^{q-i-1}b^{q+i} & a > b \end{cases} \tag{4.170}$$

This expression can be further simplified by applying the binomial theorem where possible. In particular, we can note that

$$(a-b)^{q-i-1} = \sum_{h=0}^{q-i-1} \frac{(q-i-1)!\,(-1)^h}{h!\,(q-i-h-1)!} a^{q-i-h-1} b^h \tag{4.171}$$

$$(b-a)^{q-i-1} = \sum_{h=0}^{q-i-1} \frac{(q-i-1)!\,(-1)^h}{h!\,(q-i-h-1)!} b^{q-i-h-1} a^h \tag{4.172}$$

this results in the formulation

$$s_q(a,b) = \sum_{i=0}^{q-1} \sum_{h=0}^{q-i-1} \alpha_q(i,h) \begin{cases} a^{q-i-h-1}b^h a^{q+i} & a \le b \\ b^{q-i-h-1}a^h b^{q+i} & a > b \end{cases} \tag{4.173}$$

$$= \sum_{i=0}^{q-1} \sum_{h=0}^{q-i-1} \alpha_q(i,h) \begin{cases} a^{2q-h-1}b^h & a \le b \\ b^{2q-h-1}a^h & a > b \end{cases} \tag{4.174}$$

where

$$\alpha_q(i,h) = \frac{\cancel{(q-i-1)!}\,(-1)^h}{h!\,(q-i-h-1)!} \frac{(-1)^{q+i-1}}{i!\cancel{(q-1-i)!}\,(q-1)!\,(q+i)} \tag{4.175}$$

$$= \frac{(-1)^{q+i+h-1}}{h!i!\,(q-i-h-1)!\,(q-1)!\,(q+i)} \tag{4.176}$$

In order to remove one of the two summations, we note that

$$s_q(a,b) = \sum_{i=0}^{q-1} \sum_{h=0}^{q-1} \tilde{\alpha}_q(i,h) \begin{cases} a^{2q-h-1}b^h & a \le b \\ b^{2q-h-1}a^h & a > b \end{cases} \tag{4.177}$$

where

$$\tilde{\alpha}_q(i,h) = \begin{cases} \tilde{\alpha}_q(i,h) & \text{if } h \le q-i-1 \\ 0 & \text{if } h > q-i-1 \end{cases} \tag{4.178}$$

this is useful because, now, it is possible to switch the summations order and to bring all the terms that does not depend on i in front

$$k_q(a,b) = \sum_{h=0}^{q-1} \left(\begin{cases} a^{2q-h-1} b^h & a \le b \\ b^{2q-h-1} a^h & a > b \end{cases} \right) \underbrace{\left(\sum_{i=0}^{q-1} \tilde{\alpha}_q(i,h) \right)}_{\gamma_{q,h}} \tag{4.179}$$

$$= \sum_{h=0}^{q-1} \gamma_{q,h} \begin{cases} a^{2q-h-1} b^h & a \le b \\ b^{2q-h-1} a^h & a > b \end{cases} \tag{4.180}$$

To end this proof, we can note that the coefficients $\gamma_{q,h}$ are equal to

$$\gamma_{q,h} = \sum_{i=0}^{q-1} \tilde{\alpha}_q(i,h) \tag{4.181}$$

$$= \sum_{i=0}^{q-h-1} \alpha_q(i,h) + \sum_{i=q-h-1}^{q-1} 0 \tag{4.182}$$

$$= \sum_{i=0}^{q-h-1} \left(\frac{(-1)^{q+i+h-1}}{h!i!(q-i-h-1)!(q-1)!(q+i)} \right) \tag{4.183}$$

$$= \frac{(-1)^{q+h-1}}{h!(q-1)!} \sum_{i=0}^{q-h-1} \left(\frac{(-1)^i}{i!(q-i-h-1)!(q+i)} \right) \tag{4.184}$$

$$= \frac{(-1)^{q+h-1}}{h!(q-1)!} \sum_{i=0}^{q-h-1} \left(\frac{(q-h-1)!}{i!(q-i-h-1)!} \frac{(-1)^i}{(q-h-1)!(q+i)} \right) \tag{4.185}$$

$$= \frac{(-1)^{q+h-1}}{h!(q-1)!} \sum_{i=0}^{q-h-1} \left(\binom{q-h-1}{i} \frac{(-1)^i}{(q-h-1)!(q+i)} \right) \tag{4.186}$$

now, we can note that

$$\int_0^1 h^{q-1}(-h)^i \, dh = (-1)^i \int_0^1 h^{q+i-1} \, dh = (-1)^i \left[\frac{h^{q+i}}{q+i} \right]_0^1 = \frac{(-1)^i}{q+i} \tag{4.187}$$

this allows writing the slightly more complicate formulation

$$\gamma_{q,h} = \frac{(-1)^{q+h-1}}{h!(q-1)!(q-h-1)!} \sum_{i=0}^{q-h-1} \left(\binom{q-h-1}{i} \int_0^1 h^{q-1}(-h)^i \, dh \right) \tag{4.188}$$

$$= \frac{(-1)^{q+h-1}}{h!(q-1)!(q-h-1)!} \int_0^1 h^{q-1} \underbrace{\sum_{i=0}^{q-h-1} \binom{q-h-1}{i} (-h)^i}_{(1-h)^{q-h-1}} \, dh \tag{4.189}$$

$$= \frac{(-1)^{q+h-1}}{h!(q-1)!(q-h-1)!} \underbrace{\int_0^1 h^{q-1}(1-h)^{q-h-1} \, dh}_{B(q,q-h)} \tag{4.190}$$

$$= \frac{(-1)^{q+h-1}}{h!\,(q-1)!\,(q-h-1)!} B\,(q, q-h) \tag{4.191}$$

where $B\,(q, q-h)$ is the Beta function [88] evaluated in q and $q-h$. A famous property of this function allows us to simplify the formula as follow

$$\gamma_{q,h} = \frac{(-1)^{q+h-1}}{h!\cancel{(q-1)!}\cancel{(q-h-1)!}} \frac{\cancel{(q-1)!}\cancel{(q-h-1)!}}{(q+q-h-1)!} \tag{4.192}$$

$$= \frac{(-1)^{q+h-1}}{h!\,(2q-h-1)!} \tag{4.193}$$

as we wanted to show.

■

Proof of Corollary 4.1. From the definition of stable-spline and the formulation provided by Theorem 4.1, we have

$$k_q\,(a, b) = \lambda s_q \left(e^{-\beta a}, e^{-\beta b}\right) \tag{4.194}$$

$$= \lambda \sum_{h=0}^{q-1} \gamma_{q,h} \begin{cases} \left(e^{-\beta a}\right)^{2q-h-1} \left(e^{-\beta b}\right)^{h} & \text{if } e^{-\beta a} \leq e^{-\beta b} \\ \left(e^{-\beta b}\right)^{2q-h-1} \left(e^{-\beta a}\right)^{h} & \text{if } e^{-\beta a} > e^{-\beta b} \end{cases} \tag{4.195}$$

$$= \lambda \sum_{h=0}^{q-1} \gamma_{q,h} \begin{cases} e^{-\beta(2q-h-1)a} e^{-\beta hb} & \text{if } a \geq b \\ e^{-\beta(2q-h-1)b} e^{-\beta ha} & \text{if } a < b \end{cases} \tag{4.196}$$

$$= \lambda \sum_{h=0}^{q-1} \gamma_{q,h} \begin{cases} e^{-\beta[(2q-h-1)a+hb]} & \text{if } a \geq b \\ e^{-\beta[(2q-h-1)b+ha]} & \text{if } a < b \end{cases} \tag{4.197}$$

as we wanted to show. ■

Proof of Theorem 4.2. Thanks to Assumption 4.2 the two integrals can be truncated to $t_i - d$ and $t_j - d$.

$$o_q^u(t_i,t_j) = \int_0^{t_i-d} u(t_i-\psi)\left(\int_0^{t_j-d} u(t_j-\xi)\, k_q(\psi,\xi)\, d\xi\right) d\psi \tag{4.198}$$

Now, with the changes of variable $x = t_i - \psi$ and $z = tj - \xi$, we obtain

$$o_q^u(t_i,t_j) = \int_d^{t_i} u(x)\left(\int_d^{t_j} u(z)\, k_q(t_i-x, t_j-z)\, dz\right) dx \tag{4.199}$$

Using the result of Corollary 4.1, we can write

$$\begin{aligned}
o_q^u(t_i,t_j) &= \int_d^{t_i} u(x)\left(\int_d^{t_j} u(z)\, k_q(t_i-x, t_j-z)\, dz\right) dx && (4.200)\\
&= \int_d^{t_i} u(x)\left(\int_d^{t_j} u(z)\left[\lambda\sum_{h=0}^{q-1}\gamma_{q,h}\begin{cases} e^{-\beta[(2q-h-1)(t_i-x)+h(t_j-z)]} & \text{if } t_i-x \geq t_j-z \\ e^{-\beta[(2q-h-1)(t_j-z)+h(t_i-x)]} & \text{if } t_i-x < t_j-z\end{cases}\right] dz\right) dx && (4.201)\\
&= \lambda\sum_{h=0}^{q-1}\gamma_{q,h}\int_d^{t_i} u(x)\left(\int_d^{t_j} u(z)\begin{cases} e^{-\beta[(2q-h-1)(t_i-x)+h(t_j-z)]} & \text{if } z \geq t_j-t_i+x \\ e^{-\beta[(2q-h-1)(t_j-z)+h(t_i-x)]} & \text{if } z < t_j-t_i+x\end{cases} dz\right) dx && (4.202)
\end{aligned}$$

To further simplify, we can first consider the case where $t_i \leq t_j$ and therefore $t_j - t_i \geq 0$. Here, since $d \leq x \leq t_i$, the internal integral I can be rewritten as a sum of two parts

$$I_1(x) = \int_d^{t_j-t_i+x} u(z)\, e^{-\beta[(2q-h-1)(t_j-z)+h(t_i-x)]}\, dz \tag{4.203}$$

$$I_2(x) = \int_{t_j-t_i+x}^{t_j} u(z)\, e^{-\beta[(2q-h-1)(t_i-x)+h(t_j-z)]}\, dz \tag{4.204}$$

therefore, the derived kernel becomes

$$\begin{aligned}
o_q^u(t_i,t_j) &= \sum_{h=0}^{q-1}\gamma_{q,h}\int_d^{t_i} u(x)\left(I_1(x)+I_2(x)\right) dx && (4.205)\\
&= \sum_{h=0}^{q-1}\gamma_{q,h}\left(\int_d^{t_i} u(x)\, I_1(x)\, dx + \int_d^{t_i} u(x)\, I_2(x)\, dx\right) && (4.206)\\
&= \sum_{h=0}^{q-1}\gamma_{q,h}\left(r_{q,h}^u(t_j,t_i) + w_{q,h}^u(t_j,t_i)\right) && (4.207)
\end{aligned}$$

Consider, now, the case when $t_i > t_j$. Here, we can employ the fact that the kernel is symmetric obtaining

$$o_q^u(t_i, t_j) = o_q^u(t_j, t_i) = \sum_{h=0}^{q-1} \gamma_{q,h} \left(r_{q,h}^u(t_j, t_i) + w_{q,h}^u(t_j, t_i) \right) \tag{4.208}$$

therefore

$$o_q^u(t_i, t_j) = \begin{cases} \sum_{h=0}^{q-1} \gamma_{q,h} \left(r_{q,h}^u(t_i, t_j) + w_{q,h}^u(t_i, t_j) \right) & t_i \le t_j \\ \sum_{h=0}^{q-1} \gamma_{q,h} \left(r_{q,h}^u(t_j, t_i) + w_{q,h}^u(t_j, t_i) \right) & t_i > t_j \end{cases} \tag{4.209}$$

$$= \sum_{h=0}^{q-1} \gamma_{q,h} \begin{cases} r_{q,h}^u(t_i, t_j) + w_{q,h}^u(t_i, t_j) & t_i \le t_j \\ r_{q,h}^u(t_j, t_i) + w_{q,h}^u(t_j, t_i) & t_i > t_j \end{cases} \tag{4.210}$$

as we wanted to show.

■

Proof of Theorem 4.10. This proof follows the same reasoning of the one of Theorem 4.2.

■

Proof of Theorem 4.3. The transfer function of an LTI system correspond to the Laplace transform of its impulse response. For this reason, we have

$$\hat{G}^u(s) = \mathcal{L}[\hat{g}^u](s) \tag{4.211}$$

$$= \int_0^\infty \hat{g}^u(t)\, e^{-st}\, dt \tag{4.212}$$

$$= \int_0^\infty \left(\sum_{i=1}^n c_i \hat{g}_i^u(t)\right) e^{-st}\, dt \tag{4.213}$$

$$= \sum_{i=1}^n c_i \int_0^\infty \hat{g}_i^u(t)\, e^{-st}\, dt \tag{4.214}$$

$$= \sum_{i=1}^n c_i \hat{G}_i^u(s) \tag{4.215}$$

where $\hat{G}_i^u(s) = \mathcal{L}[\hat{g}_i^u](s)$.

$$\hat{G}_i^u(s) = \int_0^\infty \hat{g}_i^u(t)\, e^{-st}\, dt \tag{4.216}$$

$$= \int_0^\infty \left[\int_0^\infty u(t_i - \xi)\, k(t,\xi)\; d\xi\right] e^{-st}\, dt \tag{4.217}$$

$$= \int_0^\infty u(t_i - \xi) \left[\int_0^\infty k(t,\xi)\, e^{-st}\, dt\right] d\xi \tag{4.218}$$

$$= \int_0^\infty u(t_i - \xi)\, K(s;\xi)\; d\xi \tag{4.219}$$

To further simplicity this expression, we can consider Assumption 4.2 that allows limiting the integral to $t_i - d$

$$\hat{G}_i^u(s) = \int_0^{t_i - d} u(t_i - \xi)\, K(s;\xi)\; d\xi \tag{4.220}$$

At last, with the change of variable $x = t_i - \xi$, we obtain

$$\hat{G}_i^u(s) = \int_d^{t_i} u(x)\, K(s; t_i - x)\; dx \tag{4.221}$$

as we wanted to show. ■

Proof of Theorem 4.4. Let's start by analyzing the term $K\left(s;x\right)$ when the kernel is a stable-spline of order q. To do so, it is useful to note that the parameter $x \in \mathbb{R}$ is always greater than 0 because it is the integration variable of (4.28). For this reason, we can write:

$$\begin{aligned}
K_q\left(s;x\right) &= \int_0^\infty k_q\left(x,t\right) e^{-st}\, dt && (4.222)\\
&= \int_0^\infty \left[\lambda \sum_{h=0}^{q-1} \gamma_{q,h} \begin{cases} e^{-\beta[(2q-h-1)x+ht]} & \text{if } x \geq t \\ e^{-\beta[(2q-h-1)t+hx]} & \text{if } x < t \end{cases}\right] e^{-st}\, dt && (4.223)\\
&= \lambda \sum_{h=0}^{q-1} \gamma_{q,h} \int_0^\infty e^{-st} \begin{cases} e^{-\beta[(2q-h-1)x+ht]} & \text{if } x \geq t \\ e^{-\beta[(2q-h-1)t+hx]} & \text{if } x < t \end{cases} dt && (4.224)\\
&= \lambda \sum_{h=0}^{q-1} \gamma_{q,h} \left[\int_0^x e^{-\beta[(2q-h-1)x+ht]-st}\, dt + \int_x^\infty e^{-\beta[(2q-h-1)t+hx]-st}\, dt\right] && (4.225)
\end{aligned}$$

The two integrals can be easily solved analytically. In particular, we have:

$$\begin{aligned}
\int_0^x e^{-\beta[(2q-h-1)x+ht]-st}\, dt &= e^{-\beta(2q-h-1)x} \int_0^x e^{-(\beta h-s)t}\, dt && (4.226)\\
&= e^{-\beta(2q-h-1)x} \left[\frac{e^{-(s+\beta h)t}}{-\left(s+\beta h\right)}\right]_0^x && (4.227)\\
&= -\frac{e^{-\beta(2q-h-1)x}}{s+\beta h}\left(e^{-(s+\beta h)x}-1\right) && (4.228)\\
&= \frac{e^{-\beta(2q-h-1)x}}{s+\beta h} - \frac{e^{-[s+\beta(2q-1)]x}}{s+\beta h} && (4.229)
\end{aligned}$$

and

$$\begin{aligned}
\int_x^\infty e^{-\beta[(2q-h-1)t+hx]-st}\, dt &= e^{-\beta hx} \int_x^\infty e^{-[s+\beta(2q-h-1)]t}\, dt && (4.230)\\
&= e^{-\beta hx} \left[\frac{e^{-[s+\beta(2q-h-1)]t}}{-\left[s+\beta\left(2q-h-1\right)\right]}\right]_x^\infty && (4.231)\\
&= -\frac{e^{-\beta hx}}{s+\beta\left(2q-h-1\right)}\left(0-e^{-[s+\beta(2q-h-1)]x}\right) && (4.232)\\
&= \frac{e^{-[s+\beta(2q-1)]x}}{s+\beta\left(2q-h-1\right)} && (4.233)
\end{aligned}$$

therefore $K_q\left(s;x\right)$ can be reformulated as

$$K_q\left(s;x\right) = \lambda \sum_{h=0}^{q-1} \gamma_{q,h} \left[\frac{e^{-\beta(2q-h-1)x}}{s+\beta h} - \frac{e^{-[s+\beta(2q-1)]x}}{s+\beta h} + \frac{e^{-[s+\beta(2q-1)]x}}{s+\beta\left(2q-h-1\right)}\right] \qquad (4.234)$$

$$= \lambda \sum_{h=0}^{q-1} \gamma_{q,h} \left[\frac{e^{-\beta(2q-h-1)x}}{s+\beta h} + e^{-[s+\beta(2q-1)]x} \left(\frac{1}{s+\beta(2q-h-1)} - \frac{1}{s+\beta h} \right) \right] \tag{4.235}$$

$$= \lambda \left[\sum_{h=0}^{q-1} \frac{\gamma_{q,h} e^{-\beta(2q-h-1)x}}{s+\beta h} + e^{-[s+\beta(2q-1)]x} \sum_{h=0}^{q-1} \gamma_{q,h} \left(\frac{1}{s+\beta(2q-h-1)} - \frac{1}{s+\beta h} \right) \right] \tag{4.236}$$

this can be further simplified by noting that

$$\sum_{h=0}^{q-1} \gamma_{q,h} \left(\frac{1}{s+\beta(2q-h-1)} - \frac{1}{s+\beta h} \right) = \frac{(-1)^q \beta^{2q-1}}{\prod_{i=0}^{2q-1}(s+\beta i)} \tag{4.237}$$

obtaining

$$K_q(s;x) = \lambda \left(\sum_{h=0}^{q-1} \frac{\gamma_{q,h} e^{-\beta(2q-h-1)x}}{s+\beta h} + e^{-[s+\beta(2q-1)]x} \frac{(-1)^q \beta^{2q-1}}{\prod_{i=0}^{2q-1}(s+\beta i)} \right) \tag{4.238}$$

Now, it is possible to plug $K_q(s;a)$ in (4.28) to obtain $\hat{G}_i^u$ for the stable-spline kernel.

$$\hat{G}_i^u(s) = \int_d^{t_i} u(x) K(s; t_i - x)\, dx \tag{4.239}$$

$$= \int_d^{t_i} u(x) \left[\lambda \left(\sum_{h=0}^{q-1} \frac{\gamma_{q,h} e^{-\beta(2q-h-1)(t_i-x)}}{s+\beta h} + e^{-[s+\beta(2q-1)](t_i-x)} \frac{(-1)^q \beta^{2q-1}}{\prod_{i=0}^{2q-1}(s+\beta i)} \right) \right] dx \tag{4.240}$$

$$= \lambda \sum_{h=0}^{q-1} \frac{\gamma_{q,h}}{s+\beta h} \underbrace{\int_d^{t_i} u(x) e^{-\beta(2q-h-1)(t_i-x)}\, dx}_{A_i^u(\beta(2q-h-1))}$$

$$+ \frac{\lambda(-1)^q \beta^{2q-1}}{\prod_{i=0}^{2q-1}(s+\beta i)} \underbrace{\int_d^{t_i} u(x) e^{-[s+\beta(2q-1)](t_i-x)}\, dx}_{A_i^u(s+\beta(2q-1))} \tag{4.241}$$

$$= \lambda \sum_{h=0}^{q-1} \frac{\gamma_{q,h}}{s+\beta h} A_i^u(\beta(2q-h-1)) + \frac{\lambda(-1)^q \beta^{2q-1}}{\prod_{i=0}^{2q-1}(s+\beta i)} A_i^u(s+\beta(2q-1)) \tag{4.242}$$

In the end, the identified transfer function using the stable-spline kernel is

$$\hat{G}^u(s) = \sum_{i=1}^{n} c_i \hat{G}_i^u(s) \tag{4.243}$$

$$= \lambda \sum_{h=0}^{q-1} \underbrace{\frac{\gamma_{q,h}}{s+\beta h} \sum_{i=1}^{n} c_i A_i^u \left(\beta \left(2q-h-1\right)\right)}_{Q_{q,h}^u(s)}$$

$$+ \lambda \underbrace{\frac{\left(-1\right)^q \beta^{2q-1}}{\prod_{i=0}^{2q-1} \left(s+\beta i\right)} \sum_{i=1}^{n} c_i A_i^u \left(s+\beta\left(2q-1\right)\right)}_{H_q^u(s)} \tag{4.244}$$

$$= \lambda \left(\sum_{h=0}^{q-1} Q_{q,h}^u \left(s\right) + H_q^u \left(s\right) \right) \tag{4.245}$$

as we wanted to show.

■

Proof of Theorem 4.5. Since the transfer function $\hat{G}^u$ is defined as the sum of $q+1$ transfer function, we need to show that all these addends are asymptotically stable.

First, Let us consider the $q-1$ addends of the type

$$Q^u_{q,h}(s) = \lambda \frac{\gamma_{q,h}}{s+\beta h} \left(\sum_{i=1}^{n} c_i A^u_i \left(\beta \left(2q-h-1\right)\right) \right) \tag{4.246}$$

with $h > 0$. All these transfer functions have only one pole in $-h\beta$ and it is strictly less than zero because $h > 0$ and $\beta > 0$. Therefore, these first $q-1$ transfer functions are asymptotically stable. The remainder of $\hat{G}^u$ is

$$R(s) = \lambda Q^u_{q,0}(s) + \lambda H^u_q(s) \tag{4.247}$$

$$= \lambda \frac{\gamma_{q,0}}{s+\beta 0} \left(\sum_{i=1}^{n} c_i A^u_i \left(\beta \left(2q-1\right)\right) \right) + \lambda \frac{(-1)^q \beta^{2q-1}}{\prod_{i=0}^{2q-1} (\beta i + s)} \left(\sum_{i=1}^{n} c_i A^u_i \left(s + \beta \left(2q-1\right)\right) \right) \tag{4.248}$$

$$= \frac{\lambda}{s} \left[\gamma_{q,0} \left(\sum_{i=1}^{n} c_i A^u_i \left(\beta \left(2q-1\right)\right) \right) + \frac{(-1)^q \beta^{2q-1}}{\prod_{i=1}^{2q-1} (\beta i + s)} \left(\sum_{i=1}^{n} c_i A^u_i \left(s + \beta \left(2q-1\right)\right) \right) \right] \tag{4.249}$$

$$= \frac{\lambda}{s} \tilde{R}(s) \tag{4.250}$$

here, there are some poles that are strictly negative:

- $-h\beta$ for $h = 1, \dots, 2q-1$ that are negative because $\beta > 0$ and $h > 0$;
- the one provided by $A^u_i\left(s + \beta\left(2q-1\right)\right)$ that are strictly negative for the Theorem hypothesis;

additionally there is a pole in 0, luckily there is also a zero in 0 because

$$\tilde{R}(0) = \gamma_{q,0} \left(\sum_{i=1}^{n} c_i A^u_i \left(\beta \left(2q-1\right)\right) \right) + \frac{(-1)^q \beta^{2q-1}}{\prod_{i=1}^{2q-1} (\beta i + 0)} \left(\sum_{i=1}^{n} c_i A^u_i \left(0 + \beta \left(2q-1\right)\right) \right) \tag{4.251}$$

$$= \left(\frac{(-1)^{q+0-1}}{0!\,(2q-0-1)!} + \frac{(-1)^q \cancel{\beta^{2q-1}}}{\cancel{b^{2q-1}}(2q-1)!} \right) \left(\sum_{i=1}^{n} c_i A^u_i \left(\beta \left(2q-1\right)\right) \right) \tag{4.252}$$

$$= \left((-1)^{q-1} + (-1)^q \right) \frac{1}{(2q-1)!} \left(\sum_{i=1}^{n} c_i A^u_i \left(\beta \left(2q-1\right)\right) \right) \tag{4.253}$$

$$= 0 \tag{4.254}$$

because $(-1)^{q-1}$ and $(-1)^q$ have opposite sign for every positive integer value of q. Therefore, if $A^u_i\left(s + \beta\left(2q-1\right)\right)$ have only strictly negative poles for $i = 1, \dots, n$, the transfer $\hat{G}^u$ is asymptotically stable.

■

Proof of Theorem 4.6. Let a_j be the j-th coefficient of the Taylor expansion of $T(s)$ centered in 0, i.e.

$$a_j = \frac{1}{j!}\frac{d^j}{ds^j}T(s)\bigg|_{s=0} \tag{4.255}$$

$$= \frac{1}{j!}\frac{d^j}{ds^j}\sum_{i=1}^{n}\alpha_i e^{-s(t_i-d)}\bigg|_{s=0} \tag{4.256}$$

$$= \frac{1}{j!}\sum_{i=1}^{n}\alpha_i \frac{d^j}{ds^j}e^{-s(t_i-d)}\bigg|_{s=0} \tag{4.257}$$

$$= \frac{1}{j!}\sum_{i=1}^{n}\alpha_i (-1)^i (t_i-d)^i e^{-s(t_i-d)}\bigg|_{s=0} \tag{4.258}$$

$$= \frac{1}{j!}\sum_{i=1}^{n}\alpha_i (d-t_i)^i e^{-s(t_i-d)}\bigg|_{s=0} \tag{4.259}$$

$$= \frac{1}{j!}\sum_{i=1}^{n}\alpha_i (d-t_i)^j \tag{4.260}$$

.

Following the procedure explained in details in [7] we obtain the following system of linear equation

$$\begin{bmatrix} -\boldsymbol{L} & \boldsymbol{I}_{z+1} \\ \boldsymbol{A} & \boldsymbol{0}_{z\times z+1} \end{bmatrix}\begin{bmatrix} \boldsymbol{d} \\ \boldsymbol{n} \end{bmatrix} = \begin{bmatrix} \boldsymbol{b}_1 \\ \boldsymbol{b}_2 \end{bmatrix} \tag{4.261}$$

where the matrices $\boldsymbol{A}$, $\boldsymbol{L}$, $\boldsymbol{b}_1$ and $\boldsymbol{b}_2$ are the one described in the Theorem statement. This linear system can be divided in two parts because the lower-right block of the system matrix is $\boldsymbol{0}_{z\times z+1}$. From this observation, we obtain

$$\begin{cases} -\boldsymbol{L}\boldsymbol{d} + \boldsymbol{I}_{z+1}\boldsymbol{n} = \boldsymbol{b}_1 \\ \boldsymbol{A}\boldsymbol{d} + \boldsymbol{0}_{z\times z+1}\boldsymbol{n} = \boldsymbol{b}_2 \end{cases} \tag{4.262}$$

therefore

$$\boldsymbol{d} = \boldsymbol{A}^{-1}\boldsymbol{b}_2 \tag{4.263}$$

$$\boldsymbol{n} = \boldsymbol{b}_1 + \boldsymbol{L}\boldsymbol{d} \tag{4.264}$$

as we wanted to show.

■

Proof of Theorem 4.7. Starting from the definition of $A_i^u(x)$ described in (4.33), we have

$$A_i^u(x) = \int_d^{t_i} u(t)\, e^{x(t-t_i)}\, dt \tag{4.265}$$

$$= \int_d^{t_i} \sum_{p=1}^m a_p u_p(t)\, e^{x(t-t_i)}\, dt \tag{4.266}$$

$$= \sum_{p=1}^m a_p \int_d^{t_i} u_p(t)\, e^{x(t-t_i)}\, dt \tag{4.267}$$

$$= \sum_{p=1}^m a_p A_i^u(x) \tag{4.268}$$

as we wanted to shown. ■

Proof of Theorem 4.8. Starting from the result of Theorem 4.7, we can compute $\hat{G}^u$ using Theorem 4.3. In particular

$$Q_{q,h}^u(s) = \frac{\gamma_{q,h}}{s+\beta h}\left(\sum_{i=1}^n c_i A_i^u(\beta(2q-h-1))\right) \tag{4.269}$$

$$= \frac{\gamma_{q,h}}{s+\beta h}\left(\sum_{i=1}^n c_i \sum_{p=1}^m a_p A_i^u(\beta(2q-h-1))\right) \tag{4.270}$$

$$= \sum_{p=1}^m a_p \left(\frac{\gamma_{q,h}}{s+\beta h}\left(\sum_{i=1}^n c_i A_i^u(\beta(2q-h-1))\right)\right) \tag{4.271}$$

$$= \sum_{p=1}^m a_p Q_{q,h}^{u_p}(s) \tag{4.272}$$

$$H_q^u(s) = \frac{(-1)^q \beta^{2q-1}}{\prod_{i=0}^{2q-1}(\beta i + s)}\left(\sum_{i=1}^n c_i A_i^u(s+\beta(2q-1))\right) \tag{4.273}$$

$$= \frac{(-1)^q \beta^{2q-1}}{\prod_{i=0}^{2q-1}(\beta i + s)}\left(\sum_{i=1}^n c_i \sum_{p=1}^m a_p A_i^{u_p}(s+\beta(2q-1))\right) \tag{4.274}$$

$$= \sum_{p=1}^m a_p \left(\frac{(-1)^q \beta^{2q-1}}{\prod_{i=0}^{2q-1}(\beta i + s)}\left(\sum_{i=1}^n c_i A_i^{u_p}(s+\beta(2q-1))\right)\right) \tag{4.275}$$

$$= \sum_{p=1}^m a_p H_q^{u_p}(s) \tag{4.276}$$

therefore

$$\hat{G}^u(s) = \lambda\left[\sum_{j=0}^{q-1} Q_{q,j}^u(s) + H_q^u(s)\right] \tag{4.277}$$

$$= \lambda \left[\sum_{p=1}^{m} a_p Q_{q,j}^{u_p}(s) + \sum_{p=1}^{m} a_p H_q^{u_p}(s) \right] \tag{4.278}$$

$$= \sum_{p=1}^{m} a_p \left(\lambda \left[Q_{q,j}^{u_p}(s) + H_q^{u_p}(s) \right] \right) \tag{4.279}$$

$$= \sum_{p=1}^{m} a_p \hat{G}^{u_p}(s) \tag{4.280}$$

Here, it is clear that the identified transfer function with the input $u(t)$ is equal to the sum of the transfer function $\hat{G}^{u_p}(s)$. Since $\hat{G}^{u_p}(s)$ is asymptotically stable for $p = 1, \ldots, m$ for hypothesis, the identified transfer function $\hat{G}^{u}(s)$ is also asymptotically stable. ■

Proof of Theorem 4.9. Starting from the definition of $o^u(t_i, t_j)$ described in (4.10), we have

$$o^u(t_i, t_j) = \int_0^{+\infty} u(t_i - \psi) \left(\int_0^{+\infty} u(t_j - \xi) k(\psi, \xi)\, d\xi \right) d\psi \tag{4.281}$$

$$= \int_0^{+\infty} \sum_{p_1=1}^{m} a_{p_1} u_{p_1}(t_i - \psi) \left(\int_0^{+\infty} \sum_{p_2=1}^{m} a_{p_2} u_{p_2}(t_j - \xi) k(\psi, \xi)\, d\xi \right) d\psi \tag{4.282}$$

$$= \sum_{p_1=1}^{m} \sum_{p_2=1}^{m} a_{p_1} a_{p_2} \int_0^{+\infty} u_{p_1}(t_i - \psi) \left(\int_0^{+\infty} a_{p_2}(t_j - \xi) k(\psi, \xi)\, d\xi \right) d\psi \tag{4.283}$$

$$= \sum_{p_1=1}^{m} \sum_{p_2=1}^{m} a_{p_1} a_{p_2} o^{u_p, u_h}(t_i, t_j) \tag{4.284}$$

as we wanted to shown. ■

CHAPTER 5

Manifold regularization for non-linear dynamic systems identification

This chapter presents a novel nonparametric approach to the identification of nonlinear dynamical systems based on the manifold regularization explained in 1.4 in semi-supervised settings.

The proposed methodology exploits an artificially augmented dataset, obtained without running new experiments on the plant, in order to learn the intrinsic manifold where the regressors lie. The knowledge of the manifold acts as a prior information on the system, that induces a proper regularization term on the identification cost. The new regularization term, as opposite to the standard Tikhonov one, enforces local smoothness of the function along the manifold. A graph-based algorithm tailored to dynamical systems is proposed to generate the augmented dataset. The hyperparameters of the method, along with the order of the system, are estimated from the available data. Numerical results on a benchmark Nonlinear Finite Impulse Response (NFIR) system show that the proposed approach may outperform the state of the art nonparametric methods.

The content of this Chapter is partially taken from the scientific publications [43, 78, 79] written by this Thesis author and his Ph.D. tutors. The remainder of the Chapter is organized as follow:

- Section 5.1 explains why the manifold regularizer can be a suitable approach in system identification;
- Section 5.2 introduces the problem that is tackled in the next sections;
- Section 5.3 briefly recalls the concept of manifold regularization explained in more details in Section 1.4;
- Section 5.4 presents a way to select unsupervised regressors in the case of NFIR system identification;
- Section 5.5 illustrates a methodology for the hyperparameters tuning;

- Section 5.6 delves into the regressors graph definition needed to employ the manifold regularization;
- Section 5.7 reports some numerical experiments that show the proposed method performance;
- Section 5.8 finishes the chapter with some concluding remarks;

5.1 Background and motivation

Semi-supervised learning is not a new concept in data-driven function mapping and has been widely used both in classification [27] and regression [87] problems. In both cases, the aim is to learn the function that generates the output. When, in addition to the supervised data, other inputs are available (without the corresponding output), their position in the regressors space gives additional information about the values of the unknown outputs [27].

It becomes clear that, whenever the input points belong to a manifold in the regressors space, their distribution provides additional information about the function to learn. Consider a classification problem where only some (labeled) points are known to belong to a certain class, whereas the others (unlabeled) correspond to an unknown class. Intuitively, if regressors lie on a manifold, the class of unlabeled points is likely to be the same of the nearest (along the manifold) labeled ones, as explained in details in Section 1.4. This rationale can be extended to dynamical systems. As an example, consider the linear Finite Impulse Response (FIR) model:

$$y_i = u_i + u_{i-1} + e_i, \tag{5.1}$$

where

$$u_i = 0.8u_{i-1} + \eta_i \tag{5.2}$$

and the terms e_i and η_i are IID noises with 0 mean and variance 1. Figure 5.1 depicts a random sampling of the regressors $\boldsymbol{x}_i = [u_i, u_{i-1}]^\top \in \mathbb{R}^{2\times 1}$ over a given time window for model (5.1).

It can be noticed that, due to the intrinsic correlation among the regressors components in dynamical models, the position of the points within the regressors space is not random. Instead, one may argue that the points are likely to lie on a certain manifold. This observation is confirmed if Principal Component Analysis (PCA) [44] is applied to the data of Figure 5.1: in fact, the first principal component can explain 93% of the data variance. This means that one dimension can be neglected without significant loss of information and, therefore, bias and variance can be effectively traded off to improve the model estimate. In this work, the case of dynamical systems will be treated for the first time.

5.2 Problem formulation

$$y\left(t_i\right) = \breve{g}\left(u\left(t_i - 1\right), \cdots, u\left(t_{i-n_u}\right)\right) + e\left(t_i\right) \tag{5.3}$$

where

- $u : \mathbb{R} \to \mathbb{R}$ is the input signal;
- $y : \mathbb{R} \to \mathbb{R}$ is the output signal;

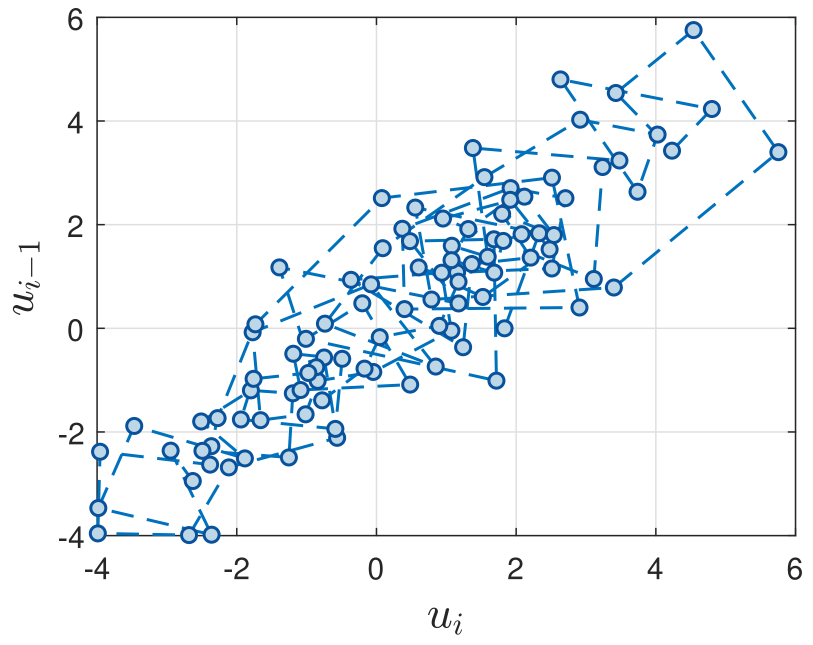

FIGURE 5.1: Regressor sampling for the system in (5.1).

- $t_i = i \cdot T_s$, with $i \in \mathbb{Z}$, are the time instants selected by the sampling process and $T_s \in \mathbb{R}_+$ is the sampling period;
- n_u is the order of the system;
- $\breve{g} : \mathbb{R}^{n_\varphi \times 1} \to \mathbb{R}$, with $n_\varphi = n_u$, is a function that describes the model behavior;
- $e : \mathbb{R} \to \mathbb{R}$ is the noise term.

The samples of the noise are considered IID and the sampling period is considered to be known. For compactness sake, as in Section 2.1, the i-th sample of the input, output and noise signal are indicated, respectively, with $u_i = u(t_i)$, $y_i = y(t_i)$ and $e_i = e(t_i)$. Now, the recursive equation 2.68 can be written as

$$y_i = \breve{g}(\boldsymbol{x}_i) + e_i \tag{5.4}$$

where

$$\boldsymbol{x}_i = \begin{bmatrix} u_{i-1} & \cdots & u_{i-n_u} \end{bmatrix}^\top \in \mathbb{R}^{n_x \times 1} \tag{5.5}$$

The function $\breve{g}$ characterize the behavior of the system and it is considered unknown in the identification problem. An estimation of the model order n_x will be provided in Section 5.5.

Furthermore, we suppose that two different datasets are available: a supervised dataset $\mathcal{D}_S$ and an unsupervised dataset $\mathcal{D}_U$.

$$\mathcal{D}_s = \{(u_i, y_i) \,|\, i = 1, \ldots, n_{Ts}\} \qquad \text{Supervised dataset} \tag{5.6}$$

$$\mathcal{D}_u = \{u_i | i = n_{Ts} + 1, \ldots, n_T\} \qquad \text{Unsupervised dataset} \tag{5.7}$$

where n_{Ts} is the number of supervised samples, n_{Tu} is the number of unsupervised samples and $n_T = n_{Ts} + n_{Tu}$ is the total number of samples.

For compactness sake, we will represent the observations and the regressors in a matrix form. Concerning the supervised dataset $\mathcal{D}_S$, we define the output vector $\boldsymbol{y}_S$ as:

$$\boldsymbol{y} = \begin{bmatrix} y_{n_x+1} & \cdots & y_{n_{Ts}} \end{bmatrix}^\top \in \mathbb{R}^{1\times n_S} \tag{5.8}$$

where $n_S = n_{Ts} - n_x$ is the number of output samples that can be employed for the identification part, given the model order n_x. In the same way, it is possible to construct the n_S supervised model regressors as

$$\boldsymbol{x}_i^S = \begin{bmatrix} u_{i-1} & \cdots & u_{i-n_u} \end{bmatrix}^\top \in \mathbb{R}^{n_x\times 1} \qquad \text{for } i = (n_x+1), \cdots, n_{Ts} \tag{5.9}$$

Analogously, $n_U = n_{Tu} - n_x$ unsupervised model regressors

$$\boldsymbol{x}_i^U = \begin{bmatrix} u_{n_{Ts}+i} & \cdots & u_{n_{Ts}+i-n_u+1} \end{bmatrix}^\top \in \mathbb{R}^{n_x\times 1} \qquad i = (n_x+1), \cdots, n_{Tu} \tag{5.10}$$

For simplicity, we define the generic regressor $\boldsymbol{\varphi}(t)$ as:

$$\boldsymbol{x}_i = \begin{cases} \boldsymbol{x}_{i+n_x}^S & 1 \le i \le n_S \\ \boldsymbol{x}_{i+n_x-n_S}^U & n_S+1 \le i \le n \end{cases} \tag{5.11}$$

where $n = n_S + n_U$ is the total number of regressors.

The aims of this chapter are:

- to develop a semi-supervised learning method to identify the NFIR system by employing the information contained both in $\mathcal{D}_S$ and in $\mathcal{D}_U$;
- to devise a method to artificially generate the unsupervised data;
- to propose a rigorous guideline for tuning the parameters of the algorithm;
- to leverage the dynamic properties of the system that we want to identify;

In order to do so, the manifold regularization, explained in 1.4, is employed. For this reason, the next section is dedicated to briefly recall this argument, for more details refer to Section 1.4.

5.3 Manifold regularization

This section briefly recalls how unsupervised data can be effectively employed in a learning framework (for more details see Section 1.4). In particular, the use of additional unsupervised data helps approximate the manifold where the regressors evolve. The discussion of the manifold regularization concepts will use the notation introduced in Section 5.2.

Section 5.1 gave intuitive motivations of how the geometry of the inputs space acts as additional information that can be employed for learning. In order to embed this notion into a learning framework, we can resort to the following rationale. In the classical literature on learning from examples [17, 44, 125], the aim is to estimate the conditional distribution $p_{y|x=\boldsymbol{x}^*}$ describing possible outputs values, given the corresponding input regressor $\boldsymbol{x}^*$. To do this, some regressors $\boldsymbol{x}_i^S$ are sampled from the marginal distribution p_x and then some outputs y_i are drawn from $p_{y|x=\boldsymbol{x}_i^S}$ to build the dataset $\mathcal{D}_S$.

Unsupervised examples $\boldsymbol{x}_i^U$ can also be extracted according to the marginal distribution p_x and used to build the dataset $\mathcal{D}_U$ As explained in Section 1.4, the knowledge of p_x can be useful if a specific assumption is made about the connection between the marginal and the conditional distributions [11]. For example, one may assume that, if two points $\boldsymbol{x}_1, \boldsymbol{x}_2$ are close according to some metrics in p_x, then the conditional distributions $p_{y|x=\boldsymbol{x}_1}$ and $p_{y|x=\boldsymbol{x}_2}$ are similar. In other words, the conditional probability distribution $p_{y|x=\boldsymbol{x}^*}$ varies smoothly along the intrinsic geometry of p_x.

The aforementioned assumption can be stated as follows [11]:

Assumption 5.1 (Semi-supervised smoothness)**.** *The conditional distribution $p_{y|x=\boldsymbol{x}^*}$ varies smoothly alongside $\boldsymbol{x}^*$ and its intrinsic geometry p_x.*

Note that, if Assumption 5.1 holds, the solution is constrained to be locally smooth, i.e., smooth over the manifold where the regressors lie. Therefore, it can be formulated as a constraint (or an equivalent regularization term) for the learning algorithm. An effective way to write a regularization term enforcing Assumption 5.1 has been first proposed in [24]. In detail, if the support of p_x is a compact manifold $\mathcal{G} \subset \mathbb{R}^m$, a common indicator of the degree of smoothness over the manifold is

$$S_g = \int_{\mathcal{G}} \left\|\nabla g\left(\boldsymbol{x}\right)\right\|^2 p_x\left(\boldsymbol{x}\right)\, d\boldsymbol{x} = \int_{\mathcal{G}} g\left(\boldsymbol{x}\right) \varDelta g\left(\boldsymbol{x}\right) p_x\left(\boldsymbol{x}\right)\, d\boldsymbol{x} \tag{5.12}$$

where ∇ and $\varDelta$ are, respectively, the gradient and the Laplace-Beltrami operators along the manifold $\mathcal{G}$.

The main idea behind such a manifold regularization is that, if Assumption 5.1 holds, the gradient of g (along $\mathcal{G}$), and so S_g, must be small. Then, minimizing S_g is a way to leverage Assumption 5.1. From (5.12), we see that the Laplacian is related to the squared norm of the gradient.

Unfortunately, p_x and $\mathcal{G}$ are usually unknown and the smoothness index S_g in (5.12) cannot be computed. One way to model the manifold is by employing a regressor graph [10, 13, 14, 36]. The graph is a weighted and completely connected graph, with the (supervised and unsupervised) regressors as its vertices, for more details see Section 1.4.2. The intrinsic structure of the regressors space is thus revealed by both supervised and unsupervised points. The weight of each edge, where $\sigma_e \in \mathbb{R}$ is a tuning parameter, is defined as

$$w_{i,j} = e^{-\frac{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2}{2\sigma_e^2}} \tag{5.13}$$

A high value of $w_{i,j}$ indicates that two regressors are similar. Notice that the concept of "smoothness over a manifold" expressed through (5.12) has been casted into a discrete graph domain.

Consider the Laplacian graph matrix

$$\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{W} \tag{5.14}$$

where $\boldsymbol{D} \in \mathbb{R}^{n_r \times n_r}$ is the diagonal matrix whose i-th diagonal element is

$$d_{i,i} = \sum_{j=1}^{n_r} w_{i,j} \tag{5.15}$$

and $\boldsymbol{W} \in \mathbb{R}^{n_r \times n_r}$ is the matrix composed by the weights $w_{i,j}$.

It can be shown that using exponential weights the operator $\boldsymbol{L}$ defined on the graph converges, with large amount of data, to the Laplace-Beltrami operator Δ [9, 10]. By considering graph derivatives [114], the term (5.12) can be represented by the Laplacian quadratic form [10, 11, 114]:

$$S_g \simeq \boldsymbol{g}\boldsymbol{L}\boldsymbol{g}^\top, \tag{5.16}$$

where the vector

$$\boldsymbol{g} = \begin{bmatrix} g\left(\boldsymbol{x}_1\right) & \cdots & g\left(\boldsymbol{x}_n\right) \end{bmatrix} \in \mathbb{R}^{1\times n} \tag{5.17}$$

depends only upon the unknown g and the input regressors[I]. It follows that both supervised and unsupervised datasets can be employed for weighting S_g within a learning task for regularizing the manifold. We will refer to (5.16) as the manifold regularization term.

Remark 5.1. From the above discussion, it comes out that, if Assumption 5.1 is not satisfied, the use of an additional unsupervised dataset is not beneficial. However, in all cases where Assumption 5.1 holds, the proposed approach may take advantage of such prior information to more accurately identify the unknown system.

Suppose now that $\breve{g}$ belongs to a RKHS $\mathcal{H}$ defined using a kernel $k : \mathbb{R}^{1\times n} \times \mathbb{R}^{1\times n} \to \mathbb{R}$. The kernel can depend by some hyperparameters $\boldsymbol{\psi}$. The typical formulation consists into finding the best function $\widehat{g}$ according to the criterion [97, 101, 111]:

$$\widehat{g} = \underset{g\in\mathcal{H}}{\arg\min} \left\{ \|\boldsymbol{y} - \boldsymbol{g}_S\|_2^2 + \tau \|g\|_{\mathcal{H}}^2 \right\}, \tag{5.18}$$

where $\boldsymbol{g}_S \in \mathbb{R}^{1\times n_S}$ is the part of $\boldsymbol{g}$ that correspond to the supervised regressors, $\|g\|_{\mathcal{H}}^2$ is the Tikhonov regularizer term and $\tau \in \mathbb{R}_+$ controls the regularization strength.

The solution to (5.18) can be found by employing the representer theorem [40, 61, 111]:

$$\widehat{g}\left(\boldsymbol{x}^*\right) = \sum_{i=1}^{n_S} c_i k\left(\boldsymbol{x}_i, \boldsymbol{x}^*\right) \tag{5.19}$$

for a n_S-tuple

$$\boldsymbol{c} = \begin{bmatrix} c_1 & \cdots & c_{n_S} \end{bmatrix}^\top \in \mathbb{R}^{n_S\times 1}. \tag{5.20}$$

Making use of (5.19), the Tikhonov regularization term of (5.18) can be restated as

$$\|g\|_{\mathcal{H}}^2 = \boldsymbol{c}^\top \boldsymbol{K}_S \boldsymbol{c} \tag{5.21}$$

where $\boldsymbol{K}_S \in \mathbb{R}^{n\times n}$ is a semidefinite positive and symmetric matrix (also called Gram matrix or kernel matrix) whose (i,j) entry is $k\left(\boldsymbol{x}_i, \boldsymbol{x}_j\right)$. The matrix $\boldsymbol{K}_S$ is formed by using only the supervised regressors.

Using (5.19), we can write the minimization problem (5.18) in such a way that it depends only on the unknown vector $\boldsymbol{c} \in \mathbb{R}^{n\times 1}$:

$$\widehat{\boldsymbol{c}} = \underset{\boldsymbol{c}\in\mathbb{R}^{n\times 1}}{\arg\min} \left\{ \left\| \boldsymbol{y}_S - \boldsymbol{c}^\top \boldsymbol{K}_S \right\|_2^2 + \tau \boldsymbol{c}^\top \boldsymbol{K}_S \boldsymbol{c} \right\}. \tag{5.22}$$

[I]The structure of the regularization term in (5.16) is shared by many manifold learning methods, where $\boldsymbol{L}$ is substituted by other symmetric matrices [25]. The reason is that such algorithms are still based on Assumption 5.1, but they formalize it from different perspectives.

It is then possible to find the estimate of the vector c by solving the system:

$$(\boldsymbol{K}_S + \tau \boldsymbol{I}_{n_S})\,\widehat{\boldsymbol{c}} = \boldsymbol{y}^\top \tag{5.23}$$

In order to include information about the local smoothness of the function (using the unsupervised data points), it is meaningful to add the manifold regularization term (5.16) to (5.18), leading to [11]:

$$\widehat{g} = \arg\min_{g \in \mathcal{H}} \left\{ \|\boldsymbol{y} - \boldsymbol{g}_S\|_2^2 + \tau \|g\|_{\mathcal{H}}^2 + \mu \boldsymbol{g}\boldsymbol{L}\boldsymbol{g}^\top \right\} \tag{5.24}$$

where $\mu \in \mathbb{R}_+$ plays the same weighting role as τ.

It is possible to show that the representer theorem still holds for the cost function (5.24) and the solution can be written by considering all $n = n_S + n_U$ regressors [11]:

$$\widehat{g}(\boldsymbol{x}^*) = \sum_{i=1}^{n} c_i k(\boldsymbol{x}_i, \boldsymbol{x}^*) \tag{5.25}$$

for a n-tuple

$$\boldsymbol{c} = \begin{bmatrix} c_1 & \cdots & c_n \end{bmatrix}^\top \in \mathbb{R}^{n \times 1}. \tag{5.26}$$

The vector $\boldsymbol{g}$, introduced in (5.16), can now the be rewritten as $\boldsymbol{g} = \boldsymbol{c}^\top \boldsymbol{K}$, where $\boldsymbol{K} \in \mathbb{R}^{n \times n}$ is the kernel matrix constructed considering both supervised and unsupervised regressors. Notice that $\boldsymbol{K}$ depends on the kernel hyperparameters $\boldsymbol{\psi}$ and may depend also on some hyperparameters $\boldsymbol{\rho}$ used to generate the augmented dataset and the regressors graph.

Now, by means of (5.25), it is possible to write the minimization problem (5.24) in such a way that it depends only on the unknown vector $\boldsymbol{c} \in \mathbb{R}^{n_r \times 1}$:

$$\widehat{\boldsymbol{c}} = \arg\min_{\boldsymbol{c} \in \mathbb{R}^{n_r \times 1}} \left\{ \left\| \boldsymbol{y} - \boldsymbol{c}^\top \boldsymbol{P}\boldsymbol{K} \right\|_2^2 + \tau \boldsymbol{c}^\top \boldsymbol{K}\boldsymbol{c} + \mu \boldsymbol{c}^\top \boldsymbol{K}\boldsymbol{L}\boldsymbol{K}\boldsymbol{c} \right\} \tag{5.27}$$

where

$$\boldsymbol{y} = \begin{bmatrix} \boldsymbol{y} & \boldsymbol{0}_{1 \times n_U} \end{bmatrix} \in \mathbb{R}^{1 \times n} \tag{5.28}$$

$$\boldsymbol{P} = \begin{bmatrix} \boldsymbol{I}_{n_S} & \boldsymbol{0}_{n_S \times n_U} \\ \boldsymbol{0}_{n_U \times n_S} & \boldsymbol{0}_{n_U \times n_U} \end{bmatrix} \in \mathbb{R}^{n \times n} \tag{5.29}$$

that is such that $\boldsymbol{P} \in \mathbb{R}^{n \times n}$, permits to select only the elements of $\boldsymbol{K}$ explaining the n_S supervised data points.

Since (5.27) is now quadratic in $\boldsymbol{c}$, its minimization can be carried out analytically and the minimizer is found by solving the linear system:

$$(\boldsymbol{P}\boldsymbol{K} + \tau \boldsymbol{I}_n + \mu \boldsymbol{L}\boldsymbol{K})\,\widehat{\boldsymbol{c}} = \boldsymbol{y}^\top \tag{5.30}$$

Remark 5.2. The role of additional data can be clearly seen in (5.30). In fact, the unsupervised points contribute here to the overall estimated function via the matrices $\boldsymbol{K}$ and $\boldsymbol{L}$.

5.4 A criterion for data augmentation

In dynamical system identification, unlike many static semi-supervised learning applications, the unsupervised data set $\mathcal{D}_U$ should better be seen as a design parameter, rather than an input of the problem. In some cases, $\mathcal{D}_U$ may contain some input time series which are likely to excite the system dynamics in future operating conditions (when the model will be used). Alternatively, $\mathcal{D}_U$ could be chosen to enforce Assumption 5.1 to be true.

Since Assumption 5.1 requires only that, inside the same high-density region, the regressors have a similar corresponding output (namely their difference must be "small"), a reasonable method is to generate the unsupervised regressors in the neighborhood of the supervised ones, where, if the system is smooth enough, they should have a similar corresponding output. This approach will generate a regressors set looking as the one exemplified in Figure 5.2, where it is possible to list n_{Ts} regions, containing a supervised regressor and some unsupervised ones.

A possible algorithm to select $\mathcal{D}_U$ as discussed above is as follows. Let $\mathcal{D}_U$ be the union of p unsupervised datasets

$$\mathcal{D}_U = \bigcup_{j=1}^{p} \mathcal{D}_U^j \tag{5.31}$$

$$\mathcal{D}_U^j = \left\{ u_i^j \middle| i = 1, \dots, n_{Ts} \right\} \tag{5.32}$$

where $u_i^j = u_i + v_i^j$, v_i^j is a random variable and $p \in \mathbb{N}$ is a free parameter of the method.

From such p datasets, it is possible to determine the quantities defined in Section 5.2. Since the unsupervised points are generated in correspondence of the supervised ones, we have n_S employable unsupervised regressors for each of the p datasets. This leads to $n_U = p \cdot n_S$ unsupervised regressors $\boldsymbol{x}_i^j \in \mathbb{R}^{n_x \times 1}$, for $j = 1, \dots, p$. Each one of them is such that, according to (5.10), for $m \leq t \leq n_S - 1$:

$$\boldsymbol{x}_i^j = \begin{bmatrix} u_{i-1}^j & \cdots & u_{i-n_u}^j \end{bmatrix}^\top \in \mathbb{R}^{n_x \times 1} \qquad i = (n_x + 1), \cdots, n_{Tu} \tag{5.33}$$

The value of v_i^j determines the distance of the p unsupervised points from the supervised one. Therefore, v_i^j has to be small enough to guarantee that the system output does not vary significantly inside these regions. A reasonable criterion for its selection is to consider that the regions should not mix with each other, since this might lead to non-smooth functions.

A possible way is to use a uniform distribution:

$$v_i^j \sim \mathcal{U}(-h, h) \qquad \begin{matrix} i = 1, \dots, n_{Ts} \\ j = 1, \dots, p \end{matrix} \tag{5.34}$$

where $h \in \mathbb{R}_+$ determines the area of the unsupervised points regions. To impose distinct regions, the following inequalities must hold:

$$\left\| \boldsymbol{x}_i^j - \boldsymbol{x}_i^S \right\|_2 \leq \frac{d}{2} \qquad \begin{matrix} i = n_x + 1, \dots, n_{Ts} \\ j = 1, \dots, p \end{matrix} \tag{5.35}$$

where d denotes the Euclidean distance between the two closest supervised regressors. After some computations, it can be shown that (5.35) can be written as:

$$\sum_{q=1}^{n_x} \left(v_i^j\right)^2 \leq \left(\frac{d}{2}\right)^2 \qquad \begin{array}{r} i = n_x + 1, \ldots, n_{Ts} \\ j = 1, \ldots, p \end{array} \tag{5.36}$$

Since $\left|v_i^j\right| \leq h$ (it is generated from the random variable (5.34)), the inequalities (5.36) hold if

$$\sum_{q=1}^{n_x} h^2 \leq \left(\frac{d}{2}\right)^2 \tag{5.37}$$

Recalling that $h \geq 0$, this corresponds to

$$h \leq \frac{d}{2\sqrt{n_x}} \tag{5.38}$$

This condition imposes a constraint for h to maintain n_{Ts} distinct regions. To make such a constraint more or less conservative, a tuning parameter $\alpha \in \mathbb{R}$ can be introduced, allowing to regulate the region maximum area, as, e.g., as follows:

$$h = \frac{d}{2\alpha\sqrt{m}}. \tag{5.39}$$

In the above criterion, $\alpha = 1$ corresponds to the threshold between mixed regions (achieved using $\alpha < 1$) and completely distinct regions ($\alpha > 1$). In Figure 5.2, an example of supervised regressors and unsupervised regressors selected with the proposed methodology (with $n_x = 2$, $p = 5$ and $\alpha > 1$) is reported.

Remark 5.3. The regressors $\boldsymbol{x}_i^j$ may improve the quality of the supervised estimate only if they lie on the same manifold spanned by the $\boldsymbol{x}_i^S$. This is indeed not difficult to obtain. Suppose that the input signal u_i is a zero-mean white noise with variance of γ^2, i.e. $u_i \sim WN\left(0, \gamma^2\right)$. We have that the regressors $\boldsymbol{x}_i^S$ are composed by lagged version of the white noise u_i. Now, assume that $u_i^j = u_i + v_i^j$, with $u_i \perp v_q^j$, $\forall i, j, q$, and $v_i^j \perp v_i^q$, $\forall j \neq q$. Then, it follows that $u_i^j \sim WN\left(0, \widetilde{\gamma}^2\right)$, with

$$\widetilde{\gamma}^2 = \gamma^2 + \frac{4h^2}{12} = \gamma^2 + \frac{h^2}{3}. \tag{5.40}$$

Therefore, $\boldsymbol{x}_i^j$ will span the same manifold of $\boldsymbol{x}_i^S$, but, since the underlying process has greater variance, the additional regressors will cover a greater area of the regressors manifold. Thus, the use of additional regressors is useful to better approximate the manifold. The same reasoning applies when u_i is a stationary zero-mean stochastic process and the independence assumptions hold.

5.5 Estimating hyperparameters and model order

The proposed method requires the tuning of the hyper-parameters $\boldsymbol{\zeta} = [\boldsymbol{\psi}, \boldsymbol{\rho}, \mu, \tau]$. In [11], no explicit guidelines for hyperparameters tuning is given. In this work, the hyperparameters $\boldsymbol{\zeta}$ is estimated via Generalized Cross-Validation (GCV) [44], by relying on the available data.

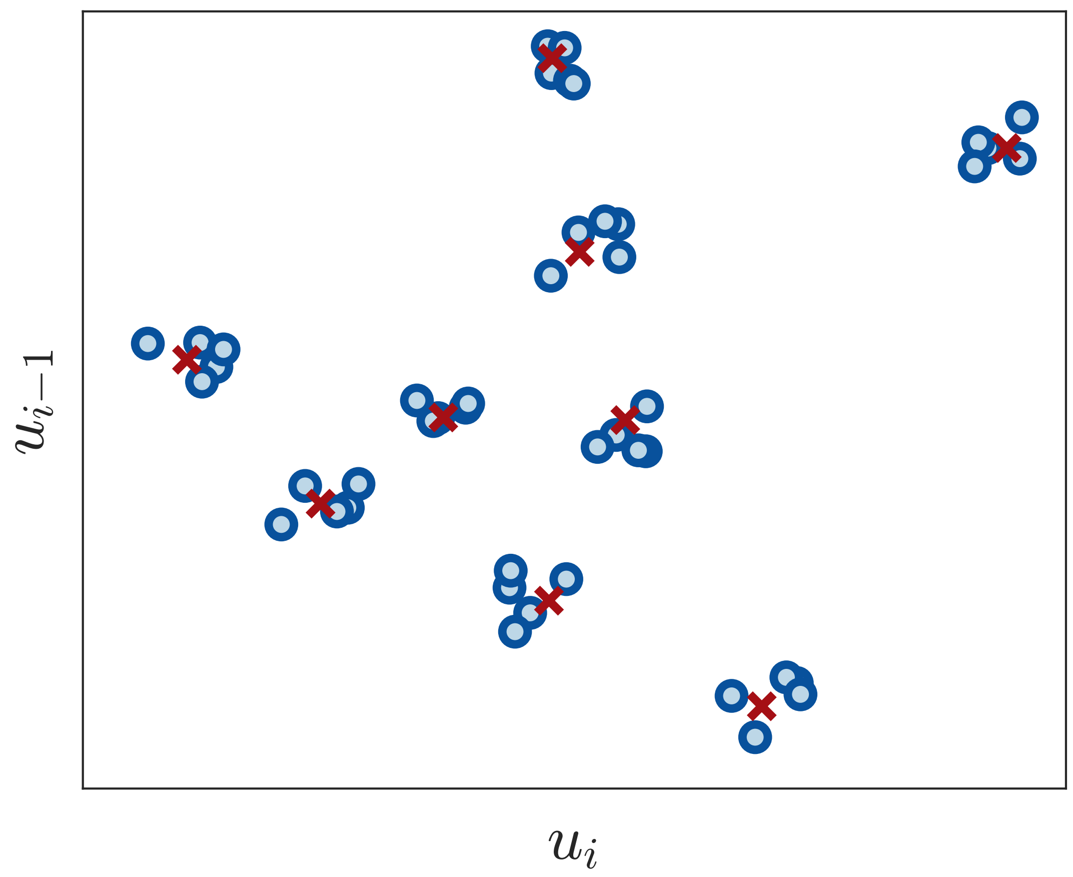

FIGURE 5.2: An example of unsupervised regressors selection, for a system with $n_x = 2$ using $p = 5$. The plot represents the supervised regressors (red crosses) and the unsupervised regressors (blue circles).

This formulation, introduced in Section 1.5.2, computes an approximation of the Leave One Out Cross-Validation (LOOCV) score in the following way. Recall that, in Tikhonov-regularized estimation, the model prediction $\widehat{\boldsymbol{y}} \in \mathbb{R}^{1\times n}$ can be computed by referring to (5.19) and (5.23) as

$$\widehat{\boldsymbol{y}}^\top = \boldsymbol{K}_S\widehat{\boldsymbol{c}} = \boldsymbol{K}_S\left(\boldsymbol{K}_S + \tau\boldsymbol{I}_n\right)^{-1}\boldsymbol{y}^\top = \boldsymbol{S}_S\left(\zeta\right)\boldsymbol{y}^\top \tag{5.41}$$

where $\boldsymbol{S}_S\left(\zeta\right) = \boldsymbol{K}_S\left(\boldsymbol{K}_S + \tau\boldsymbol{I}_n\right)^{-1}$.

In the case of the semi-supervised approach, the prediction $\widehat{\boldsymbol{y}} \in \mathbb{R}^{1\times n}$ can be cast by referring to (5.25) and (5.30) as

$$\widehat{\boldsymbol{y}}^\top = \boldsymbol{BK}\hat{\boldsymbol{b}} = \boldsymbol{BK}\left(\boldsymbol{PK} + \tau\boldsymbol{I}_{n_r} + \mu\boldsymbol{LK}\right)^{-1}\boldsymbol{y}^\top = \boldsymbol{S}\left(\zeta\right)\boldsymbol{y}^\top \tag{5.42}$$

where $\boldsymbol{S}\left(\zeta\right) = \boldsymbol{BK}\left(\boldsymbol{PK} + \tau\boldsymbol{I}_{n_r} + \mu\boldsymbol{LK}\right)^{-1}$ and $\boldsymbol{B} = \begin{bmatrix} \boldsymbol{I}_n & \boldsymbol{0}_{n\times n_{rU}} \end{bmatrix} \in \mathbb{R}^{n\times n_r}$ is used to select only the supervised components. Following [44], the number of effective degrees of freedom of a linear smoother, as in our case, can be found as:

$$\text{dof}\left(\zeta\right) = \text{Tr}\left(\bar{\boldsymbol{S}}\left(\zeta\right)\right) \qquad \bar{\boldsymbol{S}}\left(\zeta\right) = \left\{\boldsymbol{S}\left(\zeta\right), \boldsymbol{S}_S\left(\zeta\right)\right\} \tag{5.43}$$

The quantity in (5.43) can be used to efficiently compute the GCV score. The hyperparameters estimate is then computed as:

$$\widehat{\boldsymbol{\zeta}}_m = \arg\min_{\boldsymbol{\zeta}} \left\{ J_m\left(\boldsymbol{\zeta}\right) \right\} \tag{5.44}$$

$$= \arg\min_{\boldsymbol{\zeta}} \left\{ \frac{1}{n} \sum_{t=1}^{N} \left(\frac{y\left(t\right) - \widehat{y}\left(t\right)}{1 - \dfrac{\mathrm{dof}\left(\boldsymbol{\zeta}\right)}{n}} \right)^2 \right\} \tag{5.45}$$

where y and $\widehat{y}$ are the observed output and prediction at a specific time instant t. The subscript m on $J_m\left(\boldsymbol{\zeta}\right)$ and $\widehat{\boldsymbol{\zeta}}_m$ is used to highlight the dependency on the model order m. Since the model order is a discrete variable, the optimization becomes hybrid. For this reason, it is estimated as described in [97]. Specifically, the estimated order $\widehat{m}$ is obtained by computing $J_m\left(\boldsymbol{\zeta}\right)$ for a grid of chosen order values, such that:

$$\widehat{m} = \arg\min_{m} \left\{ J_m\left(\widehat{\boldsymbol{\zeta}}_m\right) \right\} \tag{5.46}$$

In light of the same rationale, we fixed the value of p (the number of additional datasets) in our simulations.

5.6 Graph topology selection

The method presented in previous sections is strongly related to the well-known approach for manifold regularization in [11]. In such a paper, it was implicitly assumed that all the regressors are connected. In this work, instead, the role of the dynamic dependency among the regressors can be explicitly taken into consideration to determine the most suitable structure of the graph describing the manifold[II].

To this end, firstly we need to distinguish between[III]:

Spatial connections among different regressors in the regressor space, they are used to constrain the outputs corresponding to close regressors to be similar;

Temporal connections among different time samples of $g\left(\boldsymbol{x}_i^S\right)$, they are used to constrain the time trajectories to be smooth.

Following the above distinction, we connect each additional regressor $\boldsymbol{x}_i^j$ to its "parent" $\boldsymbol{x}_S^j$, and each $\boldsymbol{x}_i^j$ to its "brothers" $\boldsymbol{x}_i^q$, with $j \neq q$, for every time instant i. The output that corresponds to the unsupervised regressors $\boldsymbol{x}_i^j$ is forced to be "close" to the output of the supervised regressor $\boldsymbol{x}_i^S$ from which they are generated. Consider now the time dimension and assume that the input u_i of the considered NFIR system is a zero-mean white noise signal. Then, each regressor $\boldsymbol{x}_i^S$ is correlated to the $n_x - 1$ regressors

$$\left\{ \boldsymbol{x}_{i-1}^S, \ldots, \boldsymbol{x}_{i-n_u+1}^S \right\}, \tag{5.47}$$

as well as to the $n_x - 1$ regressors

$$\left\{ \boldsymbol{x}_{i+1}^S, \ldots, \boldsymbol{x}_{i+n_u-1}^S \right\}. \tag{5.48}$$

[II]Recall that (5.16) penalizes the variations of the unknown function among the connected nodes (i.e., the regressors), thus the choice of the graph topology plays a key role to enforce smoothness.

[III]In the case of static systems, only spatial connections are meaningful, in that there is no time shift (nor correlation) among the regressors and the outputs.

Thus, we also need to connect the supervised regressors at different time instants according to the system memory (i.e. model order). Figure 5.3 shows an example of how regressors can be connected according to the proposed approach (considering both spatial and temporal connections).

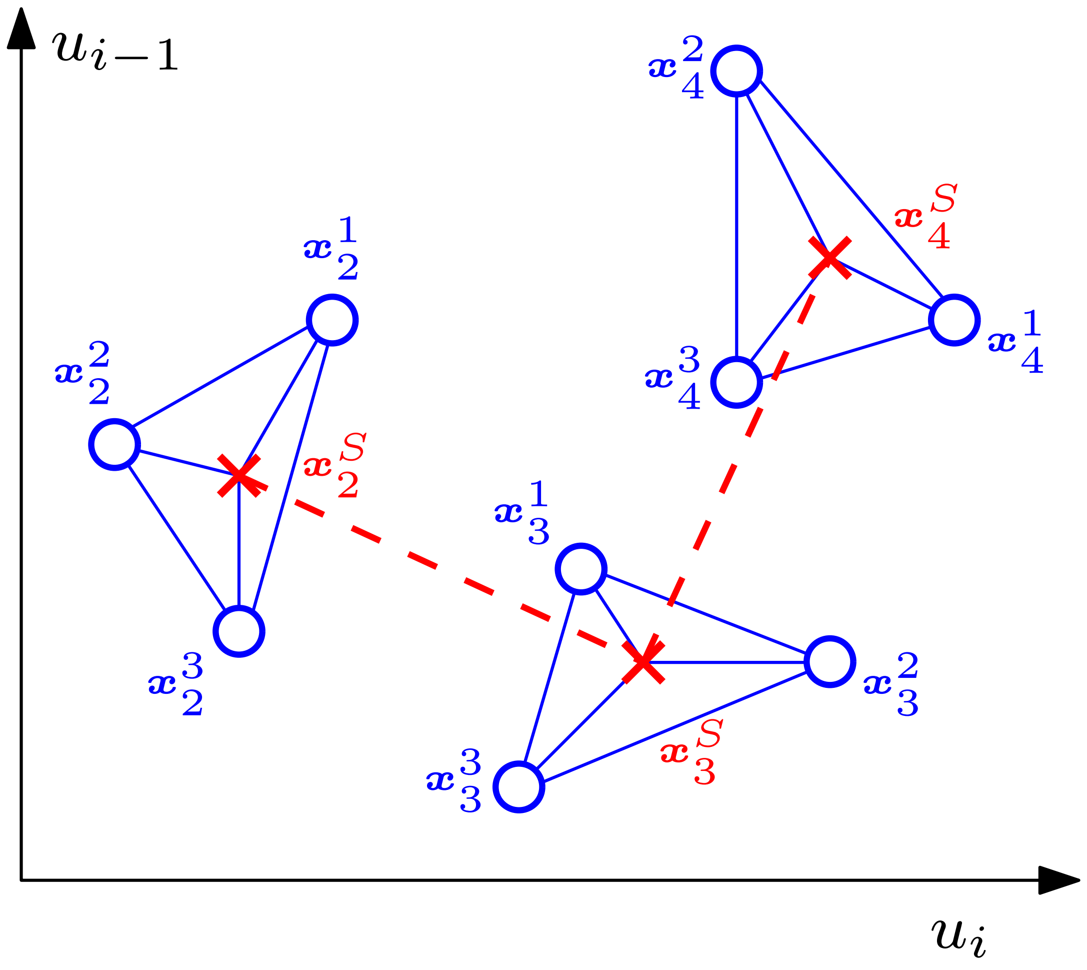

FIGURE 5.3: Example of connections in the regressor space setting the structure of the graph, with $n_x = 2$, $p = 3$ and $n_{Ts} = 3$. Temporal connections in dashed red and spatial connections in solid blue.

Remark 5.4. It is worth to point out that the proposed rationale is only one possible scheme for connecting the regressors. One may also connect the unsupervised regressors at different time instants, e.g. $\boldsymbol{x}_i^j$ with $\boldsymbol{x}_{i-1}^j$ and $\boldsymbol{x}_{i+1}^j$ in Figure 5.3. However, these additional links in the regressors graph may impose a too strong condition on the set of possible functions to be learned. In fact, consider Figure 5.4, where the solid line represents the true output, while the measurements are denoted by $y_{i]}$. Since each unsupervised regressor $\boldsymbol{x}_i^j$ is connected to its supervised "parent" $\boldsymbol{x}_S^j$, their outputs are constrained to be similar, i.e. $g\left(\boldsymbol{x}_i^j\right) \approx g\left(\boldsymbol{x}_i^S\right)$.

Temporal connections between $\boldsymbol{x}_i^S$, $\boldsymbol{x}_{i-1}^S$ and $\boldsymbol{x}_{i+1}^S$ can also be imposed to constrain the output of the function g to be smooth in time. However, since the unsupervised regressors $\boldsymbol{x}_i^j$ are generated by randomly perturbing the input sequence u_i (see again Section 5.4), temporal dependence may be partially lost, e.g., an admissible output behavior could turn out to be the dotted blue curve of Figure 5.4 (which varies more rapidly than the observed one). Therefore, the output at $g\left(\boldsymbol{x}_1^j\right)$ and $g\left(\boldsymbol{x}_2^j\right)$ should not be required to be smooth in time, but only to be similar to $g\left(\boldsymbol{x}_1^S\right)$ and $g\left(\boldsymbol{x}_2^S\right)$, respectively. Connecting $g\left(\boldsymbol{x}_i^j\right)$ at different time instants may instead lead to the dash-dotted green curve of Figure 5.4, which could be not acceptable, unless additional prior knowledge on the output dynamics is available.

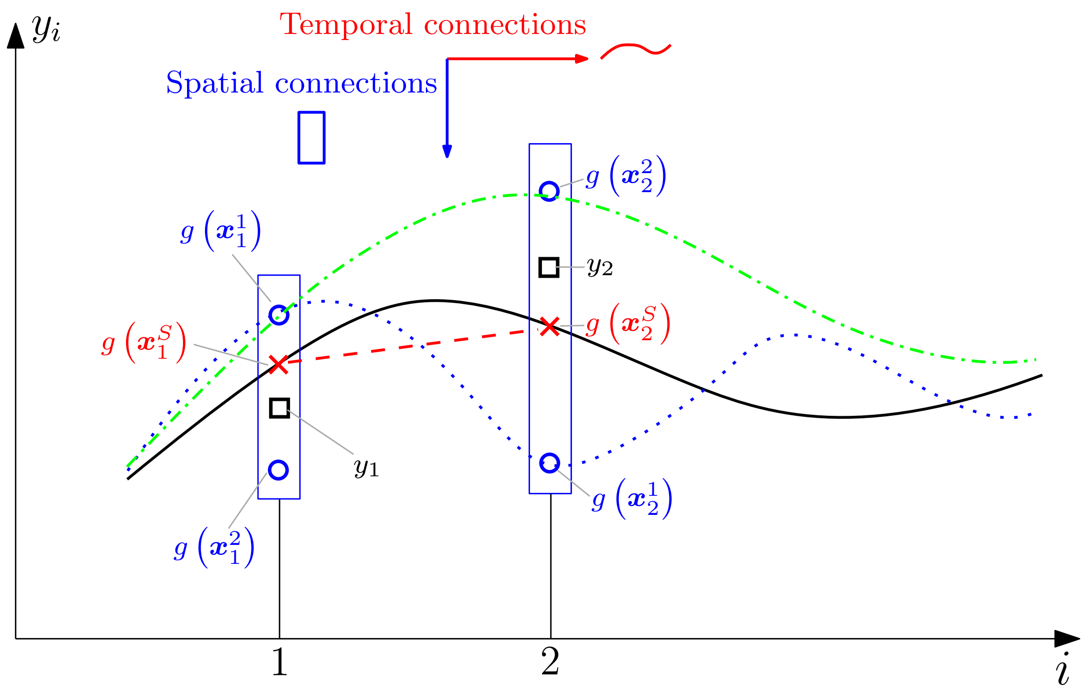
y_i
Temporal connections
Spatial connections
$g\left(\boldsymbol{x}_2^2\right)$
$g\left(\boldsymbol{x}_1^1\right)$
y_2
$g\left(\boldsymbol{x}_1^S\right)$
$g\left(\boldsymbol{x}_2^S\right)$
y_1
$g\left(\boldsymbol{x}_2^1\right)$
$g\left(\boldsymbol{x}_1^2\right)$
1
2
i

FIGURE 5.4: Representation of spatial and temporal connections in the time domain: true output (black bold line), measured output (black squares), output at supervised regressors (red crosses), output at unsupervised regressors (blue circles), possible output trajectory in case of temporal connections among supervised regressors (blue dotted line) and possible trajectory in case of temporal connections among both supervised and unsupervised regressors (green dash-dotted line).

5.7 A NUMERICAL CASE STUDY

We test the presented methodologies on the following NFIR system taken from [97]

$$\begin{aligned} y_i = u_{i-1} + 0.6u_{i-2} + 0.35u_{i-3} + 0.9u_{i-4} + 0.35u_{i-5} + 0.2u_{i-6} + \\ + 0.2u_{i-7} + 0.5u_{i-1}^2 - 0.25u_{i-4}^2 + 0.75u_{i-3}^3 + 0.25u_{i-1}u_{i-2} \\ + 0.5u_{i-1}u_{i-3} - u_{i-2}u_{i-3} + 0.5u_{i-2}u_{i-4} + e_i \end{aligned} \tag{5.49}$$

where $e_i \sim \text{WGN}(0, 0.2)$ is the measurement noise and $u(t) \sim \text{WGN}(0, 1)$ is the input signal. The identification is tackled using the Gaussian kernel

$$k(\boldsymbol{a}, \boldsymbol{b}) = \lambda e^{-\frac{\|\boldsymbol{a}-\boldsymbol{b}\|^2}{\sigma^2}} \tag{5.50}$$

where $\boldsymbol{\psi} = [\sigma, \lambda]$ are positive kernel hyperparameters.

In particular, the following approaches are compared:

(Appr. 1) **Tikhonov regression**, as in (5.18) or in Section 2.3, with $\boldsymbol{\zeta} = [\tau, \boldsymbol{\psi}]$;

(Appr. 2) **The approach of [11]** where the hyperparameters are estimated via a grid search strategy using a part of the data set for validation;

(Appr. 3) **The Kernel-based approach of [97]**;

(Appr. 4) **The proposed approach**, as in (5.24), with $\boldsymbol{\zeta} = [\tau, \mu, \boldsymbol{\psi}, \boldsymbol{\rho}]$.

The hyperparameter p, that governs how many unsupervised datasets to generate, is fixed to $p = 3$. The SNR was set to $5\,dB$. In order to assess the overall performance of the estimation methods, a supervised testing dataset $\mathcal{D}_T$ of $n_T = 10^4$ points is employed, generated analogously to $\mathcal{D}_S$. Using $\mathcal{D}_T$, it is possible to evaluate the Normalized Mean Absolute Error (NMAE) metric:

$$\text{NMAE} = \frac{\sum_{t=1}^{n_T} \left| \widehat{y}\,(t) - y_T\,(t) \right|}{\sum_{t=1}^{n_T} \left| y_T\,(t) - \overline{y}_T \right|}, \tag{5.51}$$

where $\widehat{y}\,(t)$ is the predicted test output in correspondence of a test regressor, $y_T\,(t)$ is the true test output, and $\overline{y}_T$ is the mean value of the test outputs. A Monte Carlo simulation is carried out to show the statistical significance of the proposed methodology, using 1000 runs. At each run, a different generation of the random noise was considered. The hyperparameters of the proposed method were estimated on the training set via GCV.

The experimental setup problem is highly challenging: in fact, only $n_S = 30$ supervised data are available for training. The hyperparameters of the first and third approach are estimated via marginal likelihood optimization [95, 97], according to the original formulations of the methods. For the second approach, we used $n_V = 10$ data for validation (drawn from the original dataset). Once the hyperparameters are estimated, the model is identified on all the available data.

Figure 5.5 shows the simulation results over all the Monte Carlo runs. In this critical example, the proposed approach statistically outperforms all the state of the art methods, thus showing the effectiveness of the approach in the considered setting.

5.8 Chapter concluding remarks

In this chapter, it is presented a method for learning nonlinear dynamical systems by employing augmented datasets. The additional data are generated by perturbing the measured regressors. In order to leverage such information, manifold regularization is employed, which uses additional information on the distribution of the input regressors. The dynamical structure of the NFIR systems has been taken into consideration to best select the graph connections. Numerical results showed that the proposed approach may outperform the state of the art methods. Future research will be devoted to:

- an extensive numerical assessment of the method;
- the extension of the approach to models with auto-regressive terms;
- the development of a data-driven graph topology selection policy.

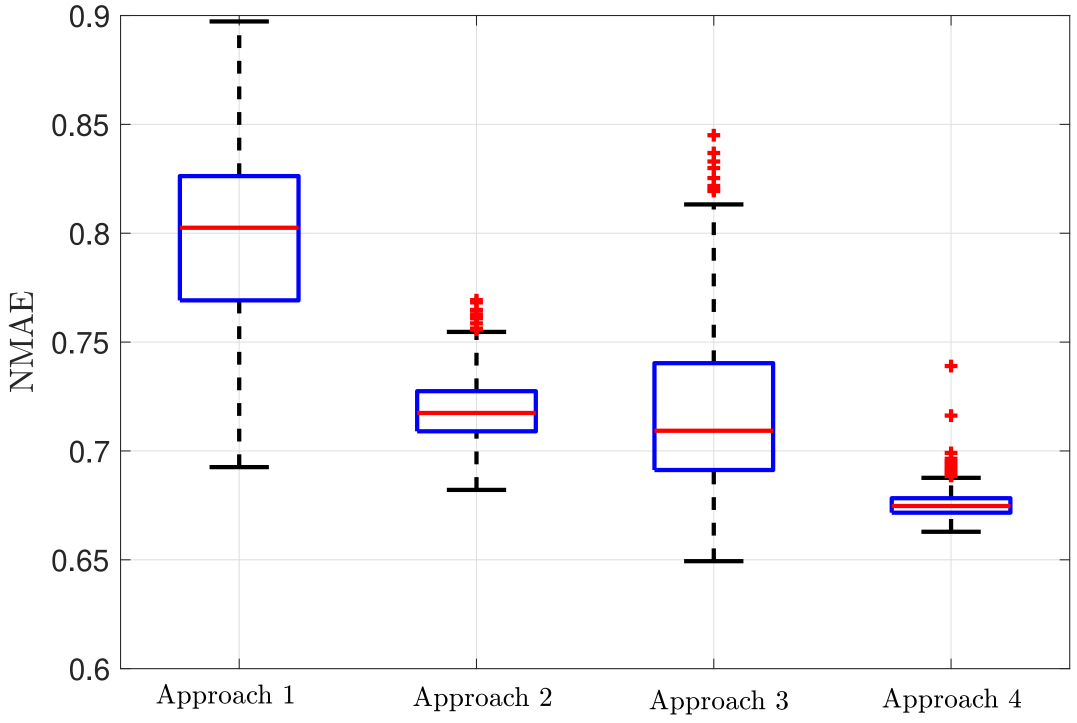

FIGURE 5.5: A numerical comparison of the proposed approach with the state of the art methods.

CHAPTER 6

Bayesian manifold regularization

This chapter presents a novel Bayesian interpretation of the manifold regularization rationale. In particular, it is shown that the manifold regularization term corresponds to an additional likelihood term that imposes smoothness along the manifold of the estimated function. The proposed approach allows defining the variance of the prediction and the possibility to employ the marginal likelihood for the hyper-parameters estimation, as shown in Section 1.5.3.

Results on a benchmark nonlinear system show improved estimation performance with respect to employing only Tikhonov regularization or manifold regularization equipped with the Generalized Cross-Validation (GCV) estimator.

The work presented in this Chapter was developed with the collaboration of Prof. Alessandro Chiuso. The remainder of the Chapter is organizer as follow:

- Section 6.1 briefly synthesizes the Bayesian interpretation of the normal Tikhonov regularization;
- Section 6.2 introduces the new likelihood term and explains how it influences the posterior distribution;
- Section 6.3 explains how to compute the marginal likelihood needed to tune the hyper-parameters;
- Section 6.4 contains some numerical examples of the proposed method;
- Section 6.5 finishes the chapter with some concluding remarks;

6.1 Bayesian interpretation of the Tikhonov regularization

This works aims to learn a generic, possibly nonlinear, mapping $g : \mathcal{X} \to \mathbb{R}$, where $\mathcal{X} \subseteq \mathbb{R}^{d \times 1}$, such that

$$y_i = g(\boldsymbol{x}_i) + e_i \tag{6.1}$$

where $\boldsymbol{x}_i \in \mathcal{X}$ and $y_i \in \mathbb{R}$ are the i-th samples of, respectively, the system input regressor and output, and $e_i \sim \mathcal{N}(0, \beta^2)$ are IID measurements additive noises.

Remark 6.1. This model corresponds to the one described in Section 1.2.2 for the identification of a static model. However, this formulation is general enough to comprehend the dynamical system case. In particular, when $\boldsymbol{x}$ is composed by the past input-output samples this formulation is equivalent at the one used in Section 2.1.4 for the non-linear system identification. Furthermore, it is trivial to change $g(\boldsymbol{x}_i)$ with the functional used for the identification of the impulse-response of a linear system, as shown in 2.1.

Suppose that we have n observations of regressor-output data

$$\mathcal{D} = \{(\boldsymbol{x}_i, y_i) \,|1 \leq t \leq n\} \tag{6.2}$$

and an additional regressor $\boldsymbol{x}_*$. In the Bayesian settings, the aim is to find the distribution of the output y^* given its corresponding regressor $\boldsymbol{x}_*$ and the dataset $\mathcal{D}$.

In order to so, it is necessary to define a prior distribution on the unknown function g. As shown in Section 1.3, we can use a Gaussian process distribution, i.e.

$$g \sim \mathcal{GP}(0_{\mathcal{X}}, k) \tag{6.3}$$

where $0_{\mathcal{X}}$ is the function that returns 0 for every regressor inside $\mathcal{X}$ and $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is a valid kernel, i.e. its symmetric and positive semi-definite. Therefore, for the definition of Gaussian process, we can write:

$$p(\boldsymbol{g}, g^* \,|\boldsymbol{X}, \boldsymbol{x}^*) = \mathcal{N}\left(\begin{bmatrix} \boldsymbol{g}^\top \\ g^* \end{bmatrix} \middle\| \begin{bmatrix} \boldsymbol{0}_{n\times 1} \\ 0 \end{bmatrix}, \begin{bmatrix} \boldsymbol{K} & \boldsymbol{k}^*(\boldsymbol{x}^*) \\ \boldsymbol{k}^*(\boldsymbol{x}^*)^\top & k(\boldsymbol{x}^*, \boldsymbol{x}^*) \end{bmatrix}\right) \tag{6.4}$$

where $g^* = g(\boldsymbol{x}^*)$ and

$$\boldsymbol{g} = \begin{bmatrix} g(\boldsymbol{x}_1) & \cdots & g(\boldsymbol{x}_n) \end{bmatrix} \in \mathbb{R}^{1\times n} \tag{6.5}$$

$$\boldsymbol{k}^*(\boldsymbol{x}^*) = \begin{bmatrix} k(\boldsymbol{x}_1, \boldsymbol{x}^*) & \cdots & k(\boldsymbol{x}_n, \boldsymbol{x}^*) \end{bmatrix}^\top \in \mathbb{R}^{n\times 1} \tag{6.6}$$

$$\boldsymbol{K} = \begin{bmatrix} k(\boldsymbol{x}_1, \boldsymbol{x}_1) & \cdots & k(\boldsymbol{x}_1, \boldsymbol{x}_n) \\ \vdots & \ddots & \vdots \\ k(\boldsymbol{x}_n, \boldsymbol{x}_1) & \cdots & k(\boldsymbol{x}_n, \boldsymbol{x}_n) \end{bmatrix} \in \mathbb{R}^{n\times n} \tag{6.7}$$

$$\boldsymbol{X} = \begin{bmatrix} \boldsymbol{x}_1 & \cdots & \boldsymbol{x}_n \end{bmatrix} \in \mathbb{R}^{d\times n} \tag{6.8}$$

Then, from the model (6.1), we can write the likelihood distribution of the measurements as

$$p(\boldsymbol{y}, y^* \,|\boldsymbol{g}, g^*, \boldsymbol{X}, \boldsymbol{x}^*) = \mathcal{N}\left(\begin{bmatrix} \boldsymbol{y}^\top \\ y^* \end{bmatrix} \middle\| \begin{bmatrix} \boldsymbol{g}^\top \\ g^* \end{bmatrix}, \begin{bmatrix} \beta^2 \boldsymbol{I}_n & \boldsymbol{0}_{n\times 1} \\ \boldsymbol{0}_{1\times n} & \beta^2 \end{bmatrix}\right) \tag{6.9}$$

where

$$\boldsymbol{y} = \begin{bmatrix} y_1 & \cdots & y_n \end{bmatrix} \in \mathbb{R}^{1\times n} \tag{6.10}$$

Since both the prior (6.4) and likelihood (6.9) are Gaussian distributed, it is possible to employ the conjugacy relations of the normal distribution [17] to obtain the marginal distribution

$$p\left(\boldsymbol{y}, y^* \left|\boldsymbol{X}, \boldsymbol{x}^*\right.\right) = \mathcal{N}\left(\left[\begin{array}{c} \boldsymbol{y}^\top \\ y^* \end{array}\right] \middle| \left[\begin{array}{c} \boldsymbol{0}_{n\times 1} \\ 0 \end{array}\right], \left[\begin{array}{cc} \boldsymbol{K} + \beta^2 \boldsymbol{I}_n & \boldsymbol{k}^*\left(\boldsymbol{x}^*\right) \\ \boldsymbol{k}^*\left(\boldsymbol{x}^*\right)^\top & \beta^2 + k\left(\boldsymbol{x}^*, \boldsymbol{x}^*\right) \end{array}\right]\right) \tag{6.11}$$

this is the joint distribution of the output that corresponds to the regressor $\boldsymbol{x}^*$, the outputs of the given dataset $\boldsymbol{y}$ and their corresponding regressors. Therefore, it is possible to obtain the desired distribution as

$$p\left(y^* \left|\boldsymbol{y}, \boldsymbol{X}, \boldsymbol{x}^*\right.\right) = \frac{p\left(\boldsymbol{y}, y^* \left|\boldsymbol{X}, \boldsymbol{x}^*\right.\right)}{p\left(\boldsymbol{y} \left|\boldsymbol{X}, \boldsymbol{x}^*\right.\right)} \tag{6.12}$$

that can be easily computed since we are dealing with normal distribution [17]

$$p\left(y^* \left|\boldsymbol{y}, \boldsymbol{X}, \boldsymbol{x}^*\right.\right) = \mathcal{N}\left(y^* \left|\rho_T^*, \sigma_T^*\right.\right) \tag{6.13}$$

where

$$\rho_T^* = \boldsymbol{k}^*\left(\boldsymbol{x}^*\right)^\top \left(\boldsymbol{K} + \beta^2 \boldsymbol{I}_n\right)^{-1} \boldsymbol{y}^\top \tag{6.14}$$

$$\sigma_T^* = \beta^2 + k\left(\boldsymbol{x}^*, \boldsymbol{x}^*\right) - \boldsymbol{k}^*\left(\boldsymbol{x}^*\right)^\top \left(\boldsymbol{K} + \beta^2 \boldsymbol{I}_n\right)^{-1} \boldsymbol{k}^*\left(\boldsymbol{x}^*\right) \tag{6.15}$$

here, we can see that the mean of the prediction distribution ρ_T^* is equal to the estimate obtained using the Tikhonov regularization with $\tau = \beta^2$. For this reason, this Bayesian approach, known as *Gaussian regression*, can be considered as the Bayesian interpretation of the Tikhonov regularization. This provides additional information of what the Tikhonov regularization do, for more details refer to Section 1.3 or [104].

In the next section, as a new contribution of this thesis, we will add a new likelihood term that results in an additional regularization term that corresponds to the manifold regularizer.

6.2 Bayesian interpretation of the manifold regularization

Before delving into the Bayesian interpretation of the manifold regularization, it is necessary to introduce a new important mathematical object: the incidence matrix of the graph.

As explained in Section 1.4, the manifold regularization term can be written as

$$\boldsymbol{g}\boldsymbol{L}\boldsymbol{g}^\top = \sum_{i=1}^{n}\sum_{j=1}^{n} w_{i,j}\left(\boldsymbol{x}_i - \boldsymbol{x}_j\right)^2 \tag{6.16}$$

where $\boldsymbol{L} \in \mathbb{R}^{n\times n}$ is the Laplacian matrix of the used regressors graph [9, 11, 114] and $w_{i,j}$ is the weight of the edge between the regressors $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$. If the graph is not complete, the weight of the missing edges can be considered equal to 0. To understand how to define the regressors graph refers to Section 1.4.2 or to Section 5.6 when the system under exam is dynamic.

In Section 1.4, the Laplacian matrix was defined as:

$$\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{W} \tag{6.17}$$

where $\boldsymbol{D}$ is a diagonal matrix whose i-th diagonal element is

$$d_{i,i} = \sum_{j=1}^{n} w_{i,j} \tag{6.18}$$

and $\boldsymbol{W}$ is the weighted adjacency matrix of the regressors graph. However, it is also possible to show that

$$\boldsymbol{L} = \boldsymbol{R}^\top \boldsymbol{R} \tag{6.19}$$

where $\boldsymbol{R} \in \mathbb{R}^{m \times n}$ is one of the possible weighted oriented incidence matrices of the regressors graph and m is the number of edges on the graph. Let us denote with a_i the weight of the i-th edge of the graph, then the entry (i, j) of $\boldsymbol{R}$ is

$$\boldsymbol{R}_{i,j} = \begin{cases} \sqrt{a_i} & \text{if the } i\text{-th edge enters the } j\text{-th node} \\ -\sqrt{a_i} & \text{if the } i\text{-th edge leaves the } j\text{-th node} \\ 0 & \text{otherwise} \end{cases} \tag{6.20}$$

Therefore, the matrix $\boldsymbol{R}$ is a sparse matrix whose rows have only two elements non-zero elements.

Remark 6.2. The incidence matrix can be seen as a discretization of the gradient operator on the regressors manifold that is approximated using the regressors graph. To better understand this concept, consider the graph in Figure 6.1 and the generic 4 samples signal

$$\boldsymbol{s} = \begin{bmatrix} s_1 & s_2 & s_3 & s_4 \end{bmatrix} \in \mathbb{R}^{1 \times 4}. \tag{6.21}$$

Then, it is possible to note that

$$\boldsymbol{R}\boldsymbol{s}^\top = \begin{bmatrix} w_{12}\left(s_1 - s_2\right) & w_{23}\left(s_2 - s_3\right) & w_{34}\left(s_3 - s_4\right) \end{bmatrix}^\top. \tag{6.22}$$

that can be seen as the weighted discrete gradient of the signal $\boldsymbol{s}$.

$$\underset{3\,\times\,4}{\boldsymbol{R}} = \begin{array}{cccc|c} 1 & 2 & 3 & 4 & \\ \sqrt{w_{12}} & -\sqrt{w_{12}} & 0 & 0 & e_1 \\ 0 & \sqrt{w_{23}} & -\sqrt{w_{23}} & 0 & e_2 \\ 0 & 0 & \sqrt{w_{34}} & -\sqrt{w_{34}} & e_3 \end{array}$$

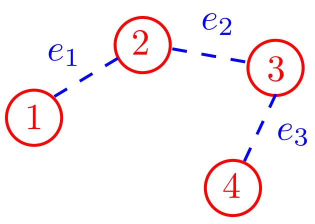

FIGURE 6.1: Example of weighted oriented incidence matrix for an undirected graph with four nodes (red circles) and three edges (dashed blue lines).

The main idea behind the Bayesian Manifold regularization is to introduce m new artificial measurements $\boldsymbol{z} \in \mathbb{R}^{1 \times m}$ that are sampled as follows

$$\boldsymbol{z}^\top = \boldsymbol{R}\boldsymbol{g}^\top + \boldsymbol{r}^\top \tag{6.23}$$

where $\boldsymbol{r}^\top \sim \mathcal{N}\left(\mathbf{0}_{m\times 1}, \eta^2 \boldsymbol{I}_m\right)$ and $\eta^2 \in \mathbb{R}_+$. We will assume that $\boldsymbol{r}$ and $\boldsymbol{e} = [e_1, \ldots, e_n] \in \mathbb{R}^{1\times n}$ are independent random variables.

Remark 6.3. Since $\boldsymbol{R}\boldsymbol{g}^\top$ corresponds to the gradient of g along the regressors graph, it is possible to impose smoothness by constraining the artificial measurements to $\mathbf{0}_{1\times m}$. For this reason, the posterior distribution will be computed considering that the artificial measurements $\boldsymbol{z}$ to be known and equal to $\mathbf{0}_{1\times m}$.

With this new measurements, the likelihood distribution becomes

$$p\left(\boldsymbol{y}, \boldsymbol{z}, y^* \left|\boldsymbol{g}, g^*, \boldsymbol{X}, \boldsymbol{x}^*\right.\right) = \mathcal{N}\left(\left.\begin{bmatrix} \boldsymbol{y}^\top \\ \boldsymbol{z}^\top \\ y^* \end{bmatrix}\right| \boldsymbol{\rho}_{lh}^*, \boldsymbol{\Sigma}_{lh}^*\right) \tag{6.24}$$

where

$$\boldsymbol{\rho}_{lh}^* = \begin{bmatrix} \boldsymbol{g}^\top \\ \boldsymbol{R}\boldsymbol{g}^\top \\ g^* \end{bmatrix} = \begin{bmatrix} \boldsymbol{I}_n & \mathbf{0}_{n\times 1} \\ \boldsymbol{R} & \mathbf{0}_{m\times 1} \\ \mathbf{0}_{1\times n} & 1 \end{bmatrix} \begin{bmatrix} \boldsymbol{g}^\top \\ g_* \end{bmatrix} \tag{6.25}$$

$$\boldsymbol{\Sigma}_{lh}^* =, \begin{bmatrix} \beta^2 \boldsymbol{I}_n & \mathbf{0}_{n\times m} & \mathbf{0}_{n\times 1} \\ \mathbf{0}_{m\times n} & \eta^2 \boldsymbol{I}_m & \mathbf{0}_{m\times 1} \\ \mathbf{0}_{1\times n} & \mathbf{0}_{1\times m} & \beta^2 \end{bmatrix} \tag{6.26}$$

using this likelihood with the Gaussian process prior (6.4), it is possible to compute the marginal likelihood using the conjugacy formula of the normal distribution [17]. Obtaining:

$$p\left(\boldsymbol{y}, \boldsymbol{z}, y^* \left|\boldsymbol{X}, x^*\right.\right) = \int p\left(\boldsymbol{y}, \boldsymbol{z}, y^* | \boldsymbol{g}, g^*\right) p\left(\boldsymbol{g}, g^* \left|\boldsymbol{X}, \boldsymbol{x}^*\right.\right) d\boldsymbol{g} dg_* \tag{6.27}$$

$$= \mathcal{N}\left(\left.\begin{bmatrix} \boldsymbol{y}^\top \\ \boldsymbol{z}^\top \\ y^* \end{bmatrix}\right| \begin{bmatrix} \mathbf{0}_{n\times 1} \\ \mathbf{0}_{m\times 1} \\ 0 \end{bmatrix}, \boldsymbol{\Sigma}_{mlh}^*\right) \tag{6.28}$$

where

$$\boldsymbol{\Sigma}_{mlh}^* = \begin{bmatrix} \boldsymbol{K} + \beta^2 \boldsymbol{I}_n & \boldsymbol{K}\boldsymbol{R}^\top & \boldsymbol{k}^*\left(\boldsymbol{x}^*\right) \\ \boldsymbol{R}\boldsymbol{K} & \boldsymbol{R}\boldsymbol{K}\boldsymbol{R}^\top + \eta^2 \boldsymbol{I}_m & \boldsymbol{k}^*\left(\boldsymbol{x}^*\right) \\ \boldsymbol{k}^*\left(\boldsymbol{x}^*\right)^\top & \boldsymbol{k}^*\left(\boldsymbol{x}^*\right)^\top \boldsymbol{R}^\top & \beta^2 + k\left(x_*, x_*\right) \end{bmatrix} \tag{6.29}$$

This is the joint distribution of the output that corresponds to the regressor $\boldsymbol{x}^*$, the output of the given dataset $\boldsymbol{y}$ with their corresponding regressors and the artificial variables $\boldsymbol{z}$. Therefore, in a similar way as the case without the manifold regularization, it is possible to obtain the desired distribution as

$$p\left(y^* | \boldsymbol{y}, \boldsymbol{z}\right) = \mathcal{N}\left(y^* \left|\rho_M^*, \sigma_M^*\right.\right) \tag{6.30}$$

where

$$\rho_M^* = \boldsymbol{k}^*(\boldsymbol{x}^*)^\top \begin{bmatrix} \boldsymbol{I}_n & \boldsymbol{R}^\top \end{bmatrix} \begin{bmatrix} \boldsymbol{K} + \beta^2 \boldsymbol{I}_n & \boldsymbol{K}\boldsymbol{R}^\top \\ \boldsymbol{R}\boldsymbol{K} & \boldsymbol{R}\boldsymbol{K}\boldsymbol{R}^\top + \eta^2 \boldsymbol{I}_m \end{bmatrix}^{-1} \begin{bmatrix} \boldsymbol{y}^\top \\ \boldsymbol{z}^\top \end{bmatrix} \tag{6.31}$$

$$\sigma_M^* = \beta^2 + k(x_*, x_*) \tag{6.32}$$

$$- \boldsymbol{k}^*(\boldsymbol{x}^*)^\top \begin{bmatrix} \boldsymbol{I}_n & \boldsymbol{R}^\top \end{bmatrix} \begin{bmatrix} \boldsymbol{K} + \beta^2 \boldsymbol{I}_n & \boldsymbol{K}\boldsymbol{R}^\top \\ \boldsymbol{R}\boldsymbol{K} & \boldsymbol{R}\boldsymbol{K}\boldsymbol{R}^\top + \eta^2 \boldsymbol{I}_m \end{bmatrix}^{-1} \begin{bmatrix} \boldsymbol{I}_n \\ \boldsymbol{R} \end{bmatrix} \boldsymbol{k}^*(\boldsymbol{x}^*) \tag{6.33}$$

In order to show that this method corresponds to the Manifold regularization, the mean of the prediction ρ_M^* needs to be equal to the manifold regularization estimate, as show in Section 1.4. In other words, we need to show that

$$\rho_M^* = \boldsymbol{k}^*(\boldsymbol{x}^*)^\top (\boldsymbol{K} + \tau \boldsymbol{I}_n + \mu \boldsymbol{L}\boldsymbol{K})^{-1} \boldsymbol{y}^\top \tag{6.34}$$

for some $\mu \in \mathbb{R}_+$ and $\tau \in \mathbb{R}_+$. For this reason, let us focus on the term $\rho_M^* \in \mathbb{R}$.

6.2.1 Mean of the posterior

Let us start by recalling the block matrix inverse formula [12]

$$\begin{bmatrix} \boldsymbol{A} & \boldsymbol{B} \\ \boldsymbol{C} & \boldsymbol{D} \end{bmatrix}^{-1} = \begin{bmatrix} \boldsymbol{Q} & -\boldsymbol{Q}\boldsymbol{B}\boldsymbol{H} \\ -\boldsymbol{H}\boldsymbol{C}\boldsymbol{Q} & \boldsymbol{H} + \boldsymbol{H}\boldsymbol{C}\boldsymbol{Q}\boldsymbol{B}\boldsymbol{H} \end{bmatrix} \tag{6.35}$$

where $\boldsymbol{A}$ and $\boldsymbol{B}$ are two square and invertible matrices, $\boldsymbol{C}$, $\boldsymbol{D}$ are two matrices of a coherent dimension and

$$\boldsymbol{H} = \boldsymbol{D}^{-1} \tag{6.36}$$

$$\boldsymbol{Q} = (\boldsymbol{A} - \boldsymbol{B}\boldsymbol{H}\boldsymbol{C})^{-1}. \tag{6.37}$$

Therefore:

$$\begin{bmatrix} \boldsymbol{K} + \beta^2 \boldsymbol{I}_n & \boldsymbol{K}\boldsymbol{R}^\top \\ \boldsymbol{R}\boldsymbol{K} & \boldsymbol{R}\boldsymbol{K}\boldsymbol{R}^\top + \eta^2 \boldsymbol{I}_m \end{bmatrix}^{-1} = \begin{bmatrix} \boldsymbol{Q} & -\boldsymbol{Q}\boldsymbol{K}\boldsymbol{R}^\top\boldsymbol{H} \\ -\boldsymbol{H}\boldsymbol{R}\boldsymbol{K}\boldsymbol{Q} & \boldsymbol{H} + \boldsymbol{H}\boldsymbol{R}\boldsymbol{K}\boldsymbol{Q}\boldsymbol{K}\boldsymbol{R}^\top\boldsymbol{H} \end{bmatrix} \tag{6.38}$$

where

$$\boldsymbol{H} = \left(\boldsymbol{R}\boldsymbol{K}\boldsymbol{R}^\top + \eta^2 \boldsymbol{I}_m\right)^{-1} \in \mathbb{R}^{m\times m} \tag{6.39}$$

$$\boldsymbol{Q} = \left(\boldsymbol{K} - \boldsymbol{K}\boldsymbol{R}^\top\boldsymbol{H}\boldsymbol{R}\boldsymbol{K} + \beta^2 \boldsymbol{I}_n\right)^{-1} \in \mathbb{R}^{n\times n}. \tag{6.40}$$

This fact can be used to rewrite ρ_M^* as:

$$\rho_M^* = \boldsymbol{k}^*(\boldsymbol{x}^*)^\top \begin{bmatrix} \boldsymbol{I}_n & \boldsymbol{R}^\top \end{bmatrix} \begin{bmatrix} \boldsymbol{Q} & -\boldsymbol{Q}\boldsymbol{K}\boldsymbol{R}^\top\boldsymbol{H} \\ -\boldsymbol{H}\boldsymbol{R}\boldsymbol{K}\boldsymbol{Q} & \boldsymbol{H} + \boldsymbol{H}\boldsymbol{R}\boldsymbol{K}\boldsymbol{Q}\boldsymbol{K}\boldsymbol{R}^\top\boldsymbol{H} \end{bmatrix} \begin{bmatrix} \boldsymbol{y}^\top \\ \boldsymbol{z}^\top \end{bmatrix} \tag{6.41}$$

Here, we can employ the fact that the artificial measurements $\boldsymbol{z}$ are equal to $\mathbf{0}_{1\times m}$, as described in Remark 6.3. Obtaining

$$\begin{aligned}\rho^*_M &= \boldsymbol{k}^*\left(\boldsymbol{x}^*\right)^\top \boldsymbol{Q}\boldsymbol{y}^\top - \boldsymbol{k}^*\left(\boldsymbol{x}^*\right)^\top \boldsymbol{R}^\top\boldsymbol{H}\boldsymbol{R}\boldsymbol{K}\boldsymbol{Q}\boldsymbol{y}^\top && (6.42)\\ &= \boldsymbol{k}^*\left(\boldsymbol{x}^*\right)^\top \left(\boldsymbol{I}_n - \boldsymbol{R}^\top\boldsymbol{H}\boldsymbol{R}\boldsymbol{K}\right)\boldsymbol{Q}\boldsymbol{y}^\top. && (6.43)\end{aligned}$$

After some mathematical steps, we can write

$$\begin{aligned}\rho^*_M &= \boldsymbol{k}^*\left(\boldsymbol{x}^*\right)^\top \left(\boldsymbol{I}_n - \boldsymbol{R}^\top\boldsymbol{H}\boldsymbol{R}\boldsymbol{K}\right)\boldsymbol{Q}\boldsymbol{y}^\top && (6.44)\\ &= \boldsymbol{k}^*\left(\boldsymbol{x}^*\right)^\top \boldsymbol{K}^{-1}\boldsymbol{K}\left(\boldsymbol{I}_n - \boldsymbol{R}^\top\boldsymbol{H}\boldsymbol{R}\boldsymbol{K}\right)\left(\boldsymbol{K}+\beta^2\boldsymbol{I}_n - \boldsymbol{K}\boldsymbol{R}^\top\boldsymbol{H}\boldsymbol{R}\boldsymbol{K}\right)^{-1}\boldsymbol{y}^\top && (6.45)\\ &= \boldsymbol{k}^*\left(\boldsymbol{x}^*\right)^\top \boldsymbol{K}^{-1}\underbrace{\left(\boldsymbol{K} - \boldsymbol{K}\boldsymbol{R}^\top\boldsymbol{H}\boldsymbol{R}\boldsymbol{K}\right)}_{\boldsymbol{P}}\left(\boldsymbol{K}+\beta^2\boldsymbol{I}_n - \boldsymbol{K}\boldsymbol{R}^\top\boldsymbol{H}\boldsymbol{R}\boldsymbol{K}\right)^{-1}\boldsymbol{y}^\top && (6.46)\\ &= \boldsymbol{k}^*\left(\boldsymbol{x}^*\right)^\top \boldsymbol{K}^{-1}\boldsymbol{P}\left(\boldsymbol{P}+\beta^2\boldsymbol{I}_n\right)^{-1}\boldsymbol{y}^\top && (6.47)\end{aligned}$$

Now, recall the Woodbury formula [52] that states

$$\left(\boldsymbol{A}+\boldsymbol{U}\boldsymbol{C}\boldsymbol{V}\right)^{-1} = \boldsymbol{A}^{-1} - \boldsymbol{A}^{-1}\boldsymbol{U}\left(\boldsymbol{C}^{-1}+\boldsymbol{V}\boldsymbol{A}^{-\mathbf{1}}\boldsymbol{U}\right)^{-1}\boldsymbol{V}\boldsymbol{A}^{-1} \qquad (6.48)$$

where $\boldsymbol{A}$ and $\boldsymbol{C}$ are two square invertible matrices and $\boldsymbol{U}$ and $\boldsymbol{V}$ are two matrices of coherent dimension. This can be employed to simplify ρ^*_M, considering $\boldsymbol{A} = \boldsymbol{P}$, $\boldsymbol{C} = \beta^2\boldsymbol{I}_n$ and $\boldsymbol{V} = \boldsymbol{U} = \boldsymbol{I}_n$, we obtain

$$\begin{aligned}\rho^*_M &= \boldsymbol{k}^*\left(\boldsymbol{x}^*\right)^\top \boldsymbol{K}^{-1}\boldsymbol{P}\left(\boldsymbol{P}+\beta^2\boldsymbol{I}_n\right)^{-1}\boldsymbol{y}^\top && (6.49)\\ &= \boldsymbol{k}^*\left(\boldsymbol{x}^*\right)^\top \boldsymbol{K}^{-1}\boldsymbol{P}\left[\boldsymbol{P}^{-1} - \boldsymbol{P}^{-1}\left(\beta^{-2}\boldsymbol{I}_n+\boldsymbol{P}^{-1}\right)^{-1}\boldsymbol{P}^{-1}\right]\boldsymbol{y}^\top && (6.50)\\ &= \boldsymbol{k}^*\left(\boldsymbol{x}^*\right)^\top \boldsymbol{K}^{-1}\left[\boldsymbol{P}\boldsymbol{P}^{-1} - \boldsymbol{P}\boldsymbol{P}^{-1}\left(\beta^{-2}\boldsymbol{P}+\boldsymbol{P}\boldsymbol{P}^{-1}\right)^{-1}\right]\boldsymbol{y}^\top && (6.51)\\ &= \boldsymbol{k}^*\left(\boldsymbol{x}^*\right)^\top \boldsymbol{K}^{-1}\left[\boldsymbol{I}_n - \left(\beta^{-2}\boldsymbol{P}+\boldsymbol{I}_n\right)^{-1}\right]\boldsymbol{y}^\top && (6.52)\end{aligned}$$

Here, it is possible to use the Woodbury formula [52] in reverse with $\boldsymbol{A}^{-1} = \boldsymbol{V} = \boldsymbol{U} = \boldsymbol{I}_n$ and $\boldsymbol{C}^{-1} = \beta^{-2}\boldsymbol{P}$

$$\begin{aligned}\rho^*_M &= \boldsymbol{k}^*\left(\boldsymbol{x}^*\right)^\top \boldsymbol{K}^{-1}\left[\left(\boldsymbol{I}_n+\beta^2\boldsymbol{P}\right)^{-1}\right]\boldsymbol{y}^\top && (6.53)\\ &= \boldsymbol{k}^*\left(\boldsymbol{x}^*\right)^\top \left(\boldsymbol{K}+\beta^2\boldsymbol{P}^{-1}\boldsymbol{K}\right)^{-1}\boldsymbol{y}^\top && (6.54)\\ &= \boldsymbol{k}^*\left(\boldsymbol{x}^*\right)^\top \left(\boldsymbol{K}+\beta^2\left(\boldsymbol{K}-\boldsymbol{K}\boldsymbol{R}^\top\boldsymbol{H}\boldsymbol{R}\boldsymbol{K}\right)^{-1}\boldsymbol{K}\right)^{-1}\boldsymbol{y}^\top && (6.55)\\ &= \boldsymbol{k}^*\left(\boldsymbol{x}^*\right)^\top \left(\boldsymbol{K}+\beta^2\left(\boldsymbol{I}_n-\boldsymbol{R}^\top\boldsymbol{H}\boldsymbol{R}\boldsymbol{K}\right)^{-1}\cancel{\boldsymbol{K}^{-1}\boldsymbol{K}}\right)^{-1}\boldsymbol{y}^\top && (6.56)\\ &= \boldsymbol{k}^*\left(\boldsymbol{x}^*\right)^\top \left(\boldsymbol{K}+\beta^2\left(\boldsymbol{I}_n-\boldsymbol{R}^\top\left(\boldsymbol{R}\boldsymbol{K}\boldsymbol{R}^\top+\eta^2\boldsymbol{I}_m\right)^{-1}\boldsymbol{R}\boldsymbol{K}\right)^{-1}\right)^{-1}\boldsymbol{y}^\top && (6.57)\end{aligned}$$

Using for one last time the Woodbury formula [52] in reverse with $\boldsymbol{A}^{-1} = \boldsymbol{I}_n, \boldsymbol{U} = \boldsymbol{R}^\top$, $\boldsymbol{V} = \boldsymbol{RK}$ and $\boldsymbol{C}^{-1} = \eta^2 \boldsymbol{I}_m$, we have:

$$\rho_M^* = \boldsymbol{k}^* (\boldsymbol{x}^*)^\top \left(\boldsymbol{K} + \beta^2 \left(\left(\boldsymbol{I}_n + \boldsymbol{R}^\top \frac{1}{\eta^2} \boldsymbol{I}_m \boldsymbol{RK} \right)^{-1} \right)^{-1} \right)^{-1} \boldsymbol{y}^\top \tag{6.58}$$

$$= \boldsymbol{k}^* (\boldsymbol{x}^*)^\top \left(\boldsymbol{K} + \beta^2 \left(\boldsymbol{I}_n + \frac{1}{\eta^2} \boldsymbol{R}^\top \boldsymbol{RK} \right) \right)^{-1} \boldsymbol{y}^\top \tag{6.59}$$

$$= \boldsymbol{k}^* (\boldsymbol{x}^*)^\top \left(\boldsymbol{K} + \beta^2 \boldsymbol{I}_n + \frac{\beta^2}{\eta^2} \boldsymbol{R}^\top \boldsymbol{RK} \right)^{-1} \boldsymbol{y}^\top \tag{6.60}$$

$$= \boldsymbol{k}^* (\boldsymbol{x}^*)^\top \left(\boldsymbol{K} + \beta^2 \boldsymbol{I}_n + \frac{\beta^2}{\eta^2} \boldsymbol{LK} \right)^{-1} \boldsymbol{y}^\top \tag{6.61}$$

Therefore, the mean of the prediction distribution ρ_M^* is equal to the estimate obtained using the manifold regularization with $\beta^2 = \tau$ and $\mu = \frac{\beta^2}{\eta^2}$. As we wanted to show.

Remark 6.4. Since we have shown that:

$$\rho_M^* = \boldsymbol{k}^* (\boldsymbol{x}^*)^\top \left(\boldsymbol{I}_n - \boldsymbol{R}^\top \boldsymbol{HRK} \right) \boldsymbol{Qy}^\top \tag{6.62}$$

$$= \boldsymbol{k}^* (\boldsymbol{x}^*)^\top \left(\boldsymbol{K} + \beta^2 \boldsymbol{I}_n + \frac{\beta^2}{\eta^2} \boldsymbol{LK} \right)^{-1} \boldsymbol{y}^\top \tag{6.63}$$

we can note that:

$$\left(\boldsymbol{K} + \beta^2 \boldsymbol{I}_n + \frac{\beta^2}{\eta^2} \boldsymbol{LK} \right)^{-1} = \left(\boldsymbol{I}_n - \boldsymbol{R}^\top \boldsymbol{HRK} \right) \boldsymbol{Q} \tag{6.64}$$

this will be useful later.

6.2.2 Variance of the posterior

With some mathematical steps and using the inversion formula (6.38), we can write:

$$\sigma_M^* = \beta^2 + k(x_*, x_*) \tag{6.65}$$

$$- \boldsymbol{k}^* (\boldsymbol{x}^*)^\top \begin{bmatrix} \boldsymbol{I}_n & \boldsymbol{R}^\top \end{bmatrix} \begin{bmatrix} \boldsymbol{K} + \beta^2 \boldsymbol{I}_n & \boldsymbol{KR}^\top \\ \boldsymbol{RK} & \boldsymbol{RKR}^\top + \eta^2 \boldsymbol{I}_m \end{bmatrix}^{-1} \begin{bmatrix} \boldsymbol{I}_n \\ \boldsymbol{R} \end{bmatrix} \boldsymbol{k}^* (\boldsymbol{x}^*) \tag{6.66}$$

$$= \beta^2 + k(x_*, x_*) - \boldsymbol{k}_* \left(\boldsymbol{I}_n - \boldsymbol{R}^\top \boldsymbol{HRK} \right) \boldsymbol{Qk}_*^\top + \tag{6.67}$$

$$- \boldsymbol{k}_* \left[\boldsymbol{I}_n + \boldsymbol{R}^\top \boldsymbol{HRKQK} - \boldsymbol{QK} \right] \boldsymbol{R}^\top \boldsymbol{HRk}_*^\top \tag{6.68}$$

then using Remark 6.4, we can write:

$$\sigma_M^* = \beta^2 + k(x_*, x_*) - \boldsymbol{k}^* (\boldsymbol{x}^*)^\top \left(\boldsymbol{K} + \beta^2 \boldsymbol{I}_n + \frac{\beta^2}{\eta^2} \boldsymbol{LK} \right)^{-1} \boldsymbol{k}^* (\boldsymbol{x}^*) + \tag{6.69}$$

$$- \boldsymbol{k}^* (\boldsymbol{x}^*)^\top \Big[\boldsymbol{I}_n + \boldsymbol{R}^\top \boldsymbol{H}\boldsymbol{R}\boldsymbol{K}\boldsymbol{Q}\boldsymbol{K} - \boldsymbol{Q}\boldsymbol{K}\Big] \boldsymbol{R}^\top \boldsymbol{H}\boldsymbol{R}\boldsymbol{k}_*^\top \tag{6.70}$$

6.3 Marginal likelihood computation

As shown in Section 1.5.3, it is possible to estimate the hyper-parameters $\boldsymbol{\zeta}$ by maximizing the marginal likelihood (ML). However, that method was usable only when the manifold regularizer was not employed because the Bayesian framework described in Section 1.3 is well defined only when the Tikhonov regularizer is the only one employed. In the last section, this problem was resolved by introducing a a Bayesian framework that can be employed when $\mu > 0$. For this reason, this section is dedicated to how to evaluate the ML in order to estimate the hyperparameters.

In this case, the hyperparameters vector is:

$$\boldsymbol{\zeta} = \begin{bmatrix} \boldsymbol{\psi} & \boldsymbol{\rho} & \beta^2 & \eta^2 \end{bmatrix} \tag{6.71}$$

where $\boldsymbol{\psi}$ are the kernel hyperparameters and $\boldsymbol{\rho}$ are the hyperparameters needed for the regressors graph selection.

From Equation (6.27), it is possible to write the marginal likelihood of the measurements available by simply removing the y^* because the distribution is Gaussian [17]. Therefore:

$$p\left(\boldsymbol{y}, \boldsymbol{z} \,|\boldsymbol{g}, \boldsymbol{\zeta}\right) = \mathcal{N}\left(\begin{bmatrix} \boldsymbol{y}^\top \\ \boldsymbol{z}^\top \end{bmatrix} \middle| \middle| \begin{bmatrix} \boldsymbol{0}_{n\times 1} \\ \boldsymbol{0}_{m\times 1} \end{bmatrix}, \Sigma_{mlh}(\boldsymbol{\zeta}) \right) \tag{6.72}$$

where the dependency on the hyperparameters is highlighted and

$$\Sigma_{mlh}(\boldsymbol{\zeta}) = \begin{bmatrix} \boldsymbol{K} + \beta^2 \boldsymbol{I}_n & \boldsymbol{K}\boldsymbol{R}^\top \\ \boldsymbol{R}\boldsymbol{K} & \boldsymbol{R}\boldsymbol{K}\boldsymbol{R}^\top + \eta^2 \boldsymbol{I}_m \end{bmatrix} \tag{6.73}$$

Then, following the reasoning of Section 1.5.3, the hyperparameters are estimated by solving the optimization problem

$$\widehat{\boldsymbol{\zeta}} = \arg\min_{\boldsymbol{\zeta}} \left\{ \begin{bmatrix} \boldsymbol{y} & \boldsymbol{z} \end{bmatrix} \left(\Sigma_{mlh}(\boldsymbol{\zeta})\right)^{-1} \begin{bmatrix} \boldsymbol{y}^\top \\ \boldsymbol{z}^\top \end{bmatrix} + \log\det\left(\Sigma_{mlh}(\boldsymbol{\zeta})\right) \right\} \tag{6.74}$$

Using the inversion formula (6.38) and remembering that $\boldsymbol{z} = \boldsymbol{0}_{1\times m}$ (see Remark 6.3), it is possible to write:

$$\begin{aligned} m_1 &= \begin{bmatrix} \boldsymbol{y} & \boldsymbol{z} \end{bmatrix} \left(\Sigma_{mlh}(\boldsymbol{\zeta})\right)^{-1} \begin{bmatrix} \boldsymbol{y}^\top \\ \boldsymbol{z}^\top \end{bmatrix} && (6.75) \\ &= \begin{bmatrix} \boldsymbol{y} & \boldsymbol{0}_{1\times m} \end{bmatrix} \begin{bmatrix} \boldsymbol{Q} & -\boldsymbol{Q}\boldsymbol{K}\boldsymbol{R}^\top \boldsymbol{H} \\ -\boldsymbol{H}\boldsymbol{R}\boldsymbol{K}\boldsymbol{Q} & \boldsymbol{H} + \boldsymbol{H}\boldsymbol{R}\boldsymbol{K}\boldsymbol{Q}\boldsymbol{K}\boldsymbol{R}^\top \boldsymbol{H} \end{bmatrix} \begin{bmatrix} \boldsymbol{y}^\top \\ \boldsymbol{0}_{m\times 1} \end{bmatrix} && (6.76) \\ &= \boldsymbol{y}\boldsymbol{Q}\boldsymbol{y}^\top && (6.77) \end{aligned}$$

in order to compute this term, we can note that, using Remark 6.4, we can write:

$$\left(\boldsymbol{I}_n - \boldsymbol{R}^\top \boldsymbol{H}\boldsymbol{R}\boldsymbol{K}\right)\boldsymbol{Q}\boldsymbol{y}^\top = \left(\boldsymbol{K} + \beta^2\boldsymbol{I}_n + \frac{\beta^2}{\eta^2}\boldsymbol{L}\boldsymbol{K}\right)^{-1}\boldsymbol{y}^\top \tag{6.78}$$

$$\boldsymbol{Q}\boldsymbol{y}^\top = \left(\boldsymbol{I}_n - \boldsymbol{R}^\top \boldsymbol{H}\boldsymbol{R}\boldsymbol{K}\right)^{-1}\left(\boldsymbol{K} + \beta^2\boldsymbol{I}_n + \frac{\beta^2}{\eta^2}\boldsymbol{L}\boldsymbol{K}\right)^{-1}\boldsymbol{y}^\top \tag{6.79}$$

therefore, we can compute $\boldsymbol{Q}\boldsymbol{y}^\top$ by solving two linear systems in succession:

$$\boldsymbol{c} = \left(\boldsymbol{K} + \beta^2\boldsymbol{I}_n + \frac{\beta^2}{\eta^2}\boldsymbol{L}\boldsymbol{K}\right)^{-1}\boldsymbol{y}^\top \tag{6.80}$$

$$\boldsymbol{b} = \left(\boldsymbol{I}_n - \boldsymbol{R}^\top \boldsymbol{H}\boldsymbol{R}\boldsymbol{K}\right)^{-1}\boldsymbol{c} \tag{6.81}$$

then:

$$m_1 = \boldsymbol{y}\boldsymbol{b} \tag{6.82}$$

Let us now consider the second term. From Equation (3.13) of [57], we have that:

$$m_2 = \log\det\left(\begin{bmatrix} \boldsymbol{K} + \beta^2\boldsymbol{I}_n & \boldsymbol{K}\boldsymbol{R}^\top \\ \boldsymbol{R}\boldsymbol{K} & \boldsymbol{R}\boldsymbol{K}\boldsymbol{R}^\top + \eta^2\boldsymbol{I}_m \end{bmatrix}\right) \tag{6.83}$$

$$= \log\det\left(\boldsymbol{K} + \beta^2\boldsymbol{I}_n\right) + \log\det\left(\boldsymbol{R}\boldsymbol{K}\boldsymbol{R}^\top + \eta^2\boldsymbol{I}_m - \boldsymbol{R}\boldsymbol{K}\left(\boldsymbol{K} + \beta^2\boldsymbol{I}_n\right)^{-1}\boldsymbol{K}\boldsymbol{R}^\top\right) \tag{6.84}$$

In order to compute $\log\det\left(\boldsymbol{K} + \beta^2\boldsymbol{I}_n\right)$ it is possible to employ again the Cholesky decomposition [52]

$$\boldsymbol{K} + \beta^2\boldsymbol{I}_n = \boldsymbol{\Pi}\boldsymbol{\Pi}^\top \tag{6.85}$$

where $\boldsymbol{\Pi} \in \mathbb{R}^{n\times n}$ is lower triangular. It follows that (see Equation (A.18) of [104]):

$$\log\det\left(\boldsymbol{K} + \beta^2\boldsymbol{I}_n\right) = 2\sum_{i=1}^{n}\log\Pi_{ii} \tag{6.86}$$

where Π_{ii} is the i-th diagonal element of $\boldsymbol{\Pi}$.

Summarizing, the marginal likelihood cost to be minimized is:

$$\widehat{\zeta} = \arg\min_{\zeta}\left\{\boldsymbol{y}\boldsymbol{b} + 2\sum_{i=1}^{n}\log\Pi_{ii} + \log\det\left(\boldsymbol{O}\right)\right\} \tag{6.87}$$

where

$$\boldsymbol{O} = \boldsymbol{R}\boldsymbol{K}\boldsymbol{R}^\top + \eta^2\boldsymbol{I}_m - \boldsymbol{R}\boldsymbol{K}\left(\boldsymbol{K} + \beta^2\boldsymbol{I}_n\right)^{-1}\boldsymbol{K}\boldsymbol{R}^\top \tag{6.88}$$

Remark 6.5. This cost function is composed by three addends:

- the first term is a data fit penalty that depends on the measured;
- the second term is a complexity penalty depending only on the covariance function and the inputs (analogous to the only Tikhonov regularization case);
- the third term is a penalty induced by the manifold regularization setting.

The following example gives some intuition about the cost function (6.87). Consider the Gaussian kernel

$$k(x_i, x_j) = e^{-\frac{(x_i - x_j)^2}{\sigma_k}} \tag{6.89}$$

where $x_i, x_j \in [-6, 6]$. From this covariance function, we generate $n = 55$ observations corrupted by zero mean Gaussian noise with variance $\beta^2 = 0.1$. Default values for hyper-parameters are $\sigma_k = 1$, $\beta^2 = 0.1$, $\eta^2 = 0.1$, $\sigma_m = 1$. To compute the regressors graph, we use the rationale for connecting the regressors as described in Section 5.6, with order $m = 2$ and without the spatial connection (there are not any additional unsupervised regressors).

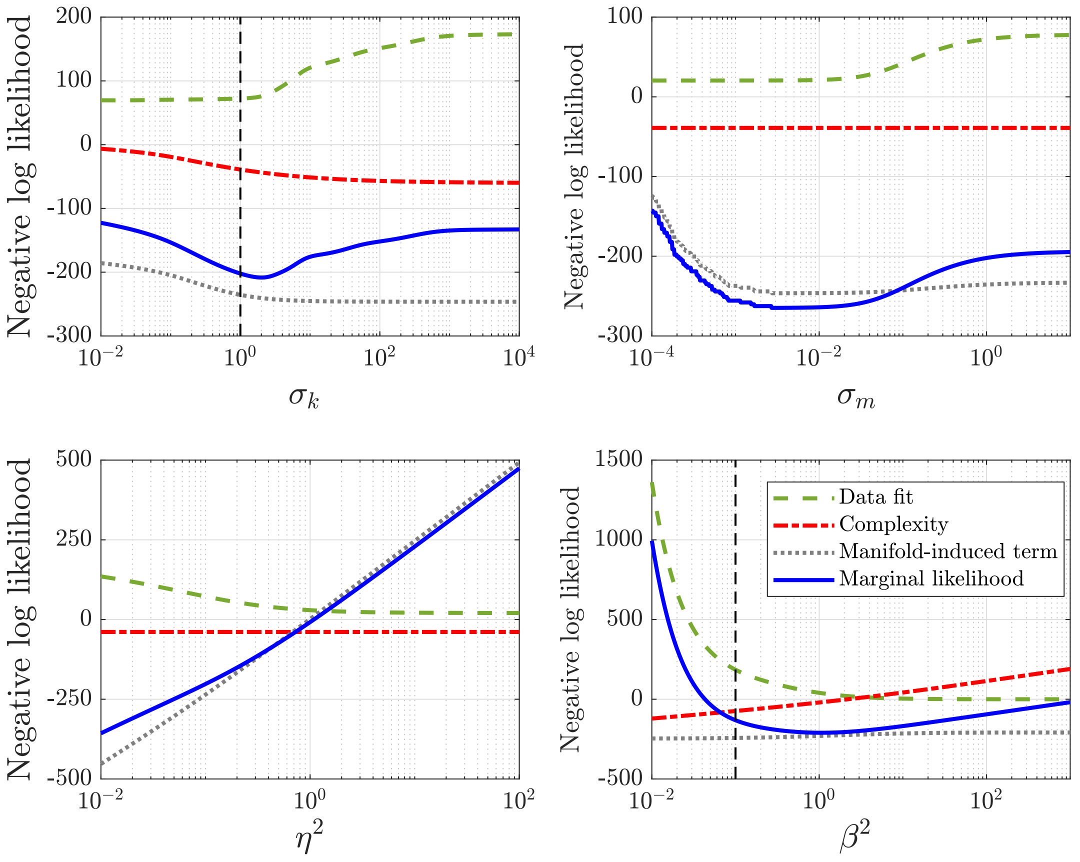

FIGURE 6.2: marginal likelihood components as function of σ_k, η^2 and σ_m. (Green dashed) data fit penalty; (Red dot-dash) complexity penalty; (Gray dotted) manifold-induced penalty; (Solid blue) negative marginal likelihood cost. True values are represented by vertical black dashed lines.

Figure 6.2 depicts the three components of (6.87) as one single hyperparameter varies. The first plot shows the ML terms as a function of the kernel variance σ_k. As σ_k increases, the model becomes less complex: the complexity term and the manifold-inducted terms decrease but the data fit term increases. Notice how the ML has its minimum near the true value $\sigma_k = 0.1$.

The second plot shows the ML terms as a function of the variance of the manifold weights σ_m. As σ_m increases, more importance is given to manifold regularization: the data fit penalty increases and the manifold-induced term decreases up to a plateau point. The complexity term is not influenced by σ_m since it depends only on the chosen kernel.

The third plot shows how the ML terms depend on the variance of noise on artificial gradient observations. As η^2 decreases, the gradient along the manifold graph is required to be lower: therefore, the function is smoother and the data fit penalty increases. As opposed to previous cases, here we do not have local minima, and the minimizer tries to get η^2 as low as possible.

The last plot represents an inverse relationship of the cost terms as a function of the noise variance β^2. As β^2 grows, the model gets more regularized, so its complexity decreases: due to the inverse relationship (caused by the $\boldsymbol{Q}$ and $\left(\boldsymbol{K} + \beta^2 \boldsymbol{I}_n\right)^{-1}$ terms in (6.82) and (6.83) respectively), we have the opposite, i.e. as β^2 grows, the model complexity increases, and other terms behaves accordingly. Notice how a trade-off is reached near the true value $\beta^2 = 1$.

Figure 6.3 depicts contour plots of (6.87) as a function of two hyperparameters, where reasoning analogous to the previous ones apply.

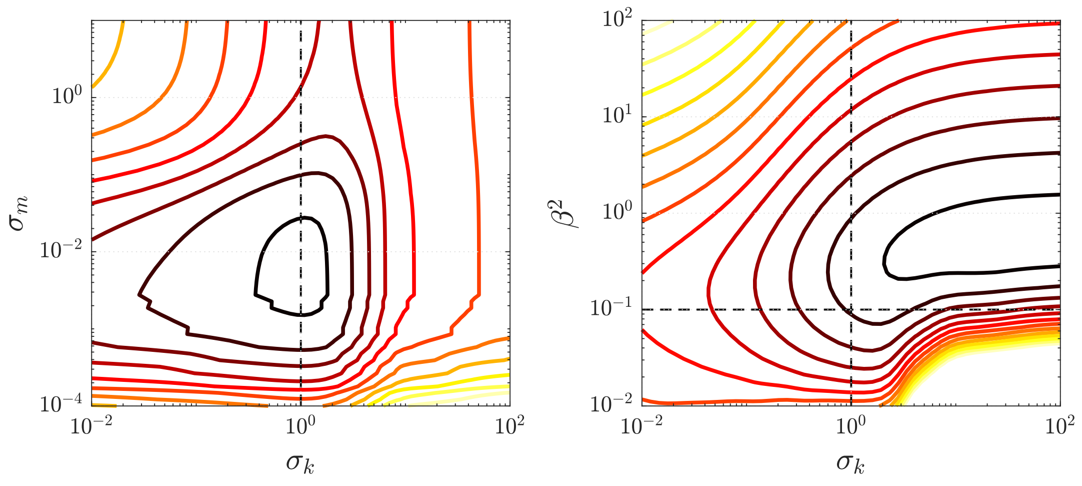

FIGURE 6.3: Contour plots of the negative marginal likelihood as function of σ_k vs. σ_m (left) and as function of σ_k vs. β^2 (right). True values are represented by vertical black dashed lines. Darker colors represent lower ML values.

6.4 Experimental results

This section evaluates the proposed approach, i.e. the use of manifold regularization with hyperparameters tuned by marginal likelihood optimization for non-linear dynamical system identification, by defining: **(i)** the kernels employed; **(ii)** the choices about graph topology and weights; **(iii)** the compared methods.

6.4.1 Kernel employed for the simulations

Since we are dealing with non-linear system identification, the regressors have the form described in Section 2.3. In particular,the regressor $\boldsymbol{x}_t$ is built as follows:

$$\boldsymbol{x}_t = \begin{bmatrix} y_t & \cdots & y_{t-n_y} & u_t & \cdots & u_{t-n_u} \end{bmatrix}^\top \tag{6.90}$$

where n_y and n_u are the orders of the autoregressive and exogenous parts, respectively. Then the kernel is composed of two components: one that defines the non-linear iteration between the past inputs and one for the iteration of the past inputs. In details, we have:

$$k\left(\boldsymbol{x}_a, \boldsymbol{x}_b\right) = \lambda_u \sum_{i=1}^{n_u - p_u + 1} e^{-\beta_u i} e^{-\frac{\sum_{j=0}^{p_u-1}\left(u_{a-i-j} - u_{b-i-j}\right)^2}{\sigma_u}} + \tag{6.91}$$

$$+ \lambda_y \sum_{i=1}^{n_y - p_y + 1} e^{-\beta_y t} e^{-\frac{\sum_{j=0}^{p_y-1}\left(y_{a-i-j} - y_{b-i-j}\right)^2}{\sigma_y}} \tag{6.92}$$

where the hyper-parameters are:

- $1 \leq p_u \leq n_u$ and $1 \leq p_y \leq n_y$ define the order of interaction between past inputs and past outputs, respectively;
- $\lambda_u \in \mathbb{R}_+$ and $\lambda_y \in \mathbb{R}_+$ define the strength of the two kernel components;
- $\beta_u \in \mathbb{R}_+$ and $\beta_y \in \mathbb{R}_+$ define the rate of decay in time of the two kernel components;

Therefore, we have:

$$\boldsymbol{\psi} = \begin{bmatrix} \lambda_u & \lambda_y & \beta_u & \beta_y & \sigma_u & \sigma_y \end{bmatrix} \in \mathbb{R}^{1\times 6} \tag{6.93}$$

and he interaction orders p^y, p^u can be estimated via a grid search as discussed in [97]. Hyperparameters n^y, n^u can be set to a suitable high number.

Remark 6.6. This kernel is a specialized version of the kernel, described in 2.3 with long regressors and fading memory and initially proposed in [97] and explained in Section 2.3, for systems where there is not any non-linear relation between the past input and output samples.

6.4.2 Choice of the regressors graph topology and weights

As shown in Section 1.4, the manifold regularization term encodes the smoothness properties of the function g along the regressors manifold, approximated by the regressors graph. There are two choices that must be made: **(i)** how to select the best graph structure (i.e. link connections); **(ii)** how to define the weights on the edges. When the graph characteristics are not guided by the application, the connection structure of the graph is usually set to "all connected" or "k-connected" (i.e. only k neighbors are connected to each node), usually with Gaussian weights [9, 11, 76]. For other possible choices of weights, see [13, 14, 35, 54]. For more details, see Section 1.4.2.

Here, we employ the rationale of connecting regressors to their neighbors based on their order of interactions, such that each node is connected to its $p = \max\left(p_y, p_u\right)$ nearest nodes in time (similar to what has been done in [43, 78, 79] and in Section 5.6), such that a regressor $\boldsymbol{x}_i$ is connected to regressors $\{\boldsymbol{x}_{i-1}, \ldots, \boldsymbol{x}_{i-p}\}$ and $\{\boldsymbol{x}_{i+1}, \ldots, \boldsymbol{x}_{i+p}\}$.

This rationale has two advantages: **(i)** it is computationally much cheaper with respect to connecting all the nodes; **(ii)** it reflects the fact that data came from a dynamical system. The first advantage is related to the row dimension of the $\boldsymbol{R}$ matrix: if there are more connections, the incidence matrix becomes taller, and the marginal likelihood computations get more expensive. Furthermore, by connecting only regressors closer in time, the smoothness is enforced only for those regressors that are more correlated or that interact in a stronger way. In the simulations, we considered Gaussian weights on the edges, such that

$$w_{i,j} = e^{-\frac{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|_2^2}{\sigma_m}}, \tag{6.94}$$

where $\sigma_m \in \mathbb{R}_+$ is an hyperparameter of the method.

6.4.3 Dynamical model example and methods comparison

We tested the proposed method on a NARX system taken from the literature [97]:

$$\begin{aligned} y_i = {} & 0.8y_{i-1} + u_{i-1} - 0.3u_{i-1}^3 + 0.25u_{i-1}u_{i-2} + \\ & - 0.3u_{i-2} + 0.24u_{i-2}^3 - 0.2u_{i-2}u_{i-3} - 0.4u_{i-3} + e_i \\ e_i & \sim \text{WGN}\left(0, 0.14^2\right) \end{aligned} \tag{6.95}$$

We performed $M = 100$ Monte Carlo runs, varying the noise realization on identification data, with zero initial condition. The train input is $u \sim \text{WGN}\left(0, 1^2\right)$, where WGN stands for White Gaussian Noise. The number of regressors used for the identification of the models is $n = 55$. As already mentioned, manifold regularization can be especially useful in a small data regime. In order to better assess the performance of the proposed additional manifold regularization term, we fixed the orders n_y, n_u, p_y, p_u to their true values.

We compare the following approaches:

(Appr. 1) **Tikhonov regression**with kernel in [97], hyperparameters estimated via ML optimization;

$$\psi = \begin{bmatrix} \lambda_1 & \lambda_2 & \sigma & \beta^2 \end{bmatrix} \qquad n = 3 \qquad p = 2 \tag{6.96}$$

(Appr. 2) **Tikhonov regularization** with the kernel (6.91), hyperparameters estimated via ML optimization;

$$\psi = \begin{bmatrix} \lambda_u & \lambda_y & \beta_u & \beta_y & \sigma_u & \sigma_y \end{bmatrix} \qquad \begin{matrix} n_y = 3 & p_y = 3 \\ n_u = 1 & p_u = 1 \end{matrix} \tag{6.97}$$

(Appr. 3) **Tikhonov + manifold regularization** with the kernel (6.91), hyperparameters estimated via GCV (see Section 1.5.2 or [44]);

$$\psi = \begin{bmatrix} \lambda_u & \lambda_y & \beta_u & \beta_y & \sigma_u & \sigma_y \end{bmatrix} \qquad \begin{matrix} n_y = 3 & p_y = 3 \\ n_u = 1 & p_u = 1 \end{matrix} \tag{6.98}$$

(Appr. 4) **Tikhonov + manifold regularization** with the kernel (6.91), hyperparameters estimated via ML, i.e. the proposed approach;

$$\psi = \begin{bmatrix} \lambda_u & \lambda_y & \beta_u & \beta_y & \sigma_u & \sigma_y \end{bmatrix} \qquad \begin{matrix} n_y = 3 & p_y = 3 \\ n_u = 1 & p_u = 1 \end{matrix} \tag{6.99}$$

We tested the performance of the methods on a separate test dataset

$$\mathcal{D}_T = \left\{ \left(u_i^*, y_i^*\right) | 1 \le i \le 1000 \right\}, \tag{6.100}$$

generated in the same way as the training one (i.e. with autoregressive noise). The performance of the estimated model is measured via the Fit index:

$$\text{Fit} = 1 - \sqrt{\frac{\sum_{i=1}^{1000} \left(\hat{y}_i^* - y_i^*\right)^2}{\sum_{i=1}^{1000} \left(\overline{y}_i^* - y_i^*\right)^2}} \tag{6.101}$$

$$\hat{y}_i^* = \hat{g}\left(\boldsymbol{x}_i^*\right) \tag{6.102}$$

$$\overline{y}_i^* = \frac{\sum_{t=1}^{1000} y_i^*}{1000} \tag{6.103}$$

where $\boldsymbol{x}_i^*$ is the i-th test regressor, $\overline{y}_i^*$ is the mean value of the test data output, and $\hat{g}$ is the estimate model.

Simulation results are reported in Figure 6.4. With only Tikhonov regularization, notice that the specialized kernel in Equation (6.91) performs better than the general kernel in [97]. We, therefore, compare the use of manifold regularization to the only Tikhonov one, both equipped with the specialized kernel. In this example, the use of manifold regularization with GCV for hyperparameters estimation leads to better test performance, although with higher variance. marginal likelihood optimization, as enabled by the Bayesian interpretation, gives hyperparameters estimates that are still able to provide better performance with respect to using only Tikhonov regularization, while keeping under control the variance of the model estimates and their test performance.

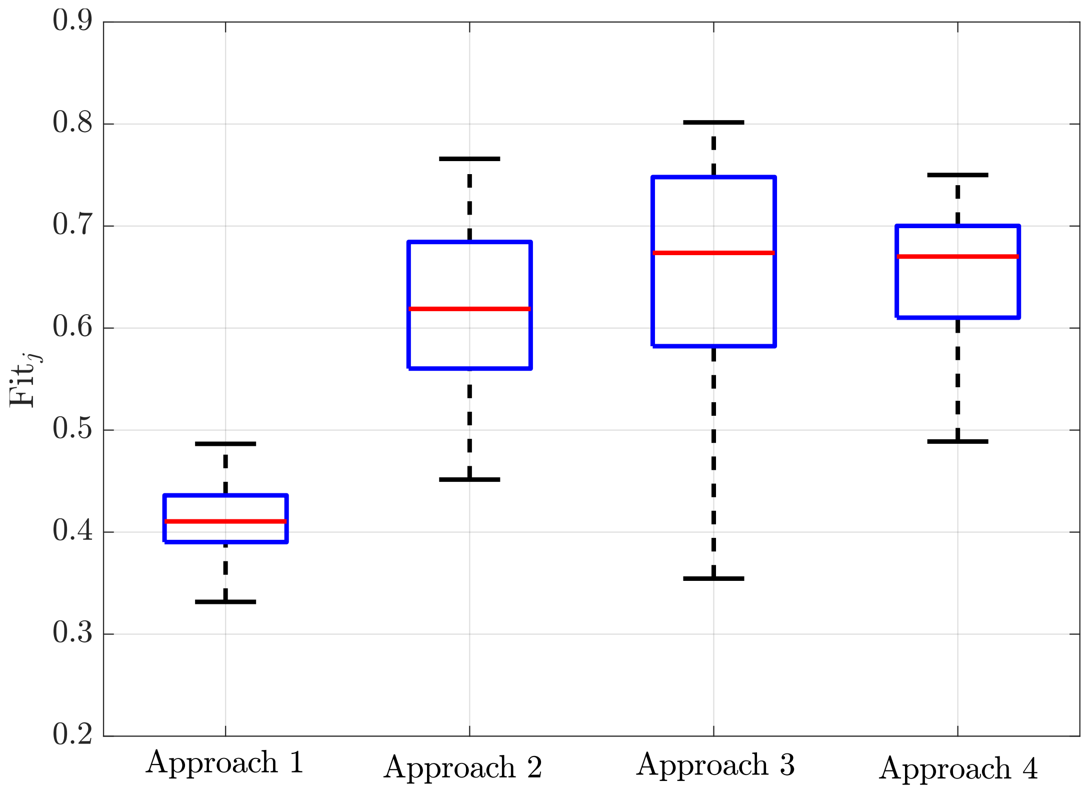

FIGURE 6.4: Simulation results. The number of regressors used for identification is $n = 55$.

6.5 Concluding remarks

In this chapter, it is presented a novel approach for nonlinear nonparametric system identification, based on a Bayesian view of manifold regularization. The advantages of the new rationale are twofold:

- it unveils a new interpretation of the manifold regularization based on the gradient of the function along the manifold;
- it allows using the marginal likelihood optimization for tuning the hyperparameters of the method.

Results have shown that the proposed approach can have better performance with respect to both Tikhonov regularization and manifold regularization with hyperparameters tuned by generalized cross-validation. Future research is devoted to the development of a Bayesian interpretation for the semi-supervised case and new design strategies for the graph topology.

CHAPTER 7

Classification of light charged particles

This chapter presents an application of kernel-based linear identification of discrete linear models.

In more details, this work describes a nonparametric learning approach for the automatic classification of particles produced by the collision of a heavy ion beam on a target, by focusing on the identification of isotopes Light Charged Particles (LCP). In particular, it is shown that the measurement of the particle collision can be traced back to the impulse response of a linear dynamical system and, by employing recent kernel-based approaches, a nonparametric model is found that effectively trades off bias and variance of the model estimate. Then, the smoothened signals can be employed to classify the different types of particles. Experimental results show that the proposed method outperforms the state of the art approaches. All the experiments are carried out with the large detector array CHIMERA (Charge Heavy Ions Mass and Energy Resolving Array) in Catania, Italy.

The content of this Chapter is partially taken from the scientific publications [77] written by this Thesis author and his Ph.D. tutors. The remainder of the Chapter is organized as follow:

- Section 7.1 contains a brief introduction of the application and the methodologies used to tackled it;
- Section 7.2 describes the experimental setup of the application;
- Section 7.3 presents the proposed approach employed for modeling the nuclear phenomena;
- Section 7.4 is dedicated to the machine learning classifiers used for the particles classification;
- Section 7.5 discusses the obtained results;
- Section 7.6 contains some concluding remarks and future developments.

7.1 Introduction

One of the most interesting goals of the intermediate energy heavy ion research is to investigate the characteristics of the nuclei under extreme conditions of density and temperature [1]. In these types of physics experiments, the standard approach is the measurement and analysis of the collision effects of a heavy ion beam over a target. The nuclear reactions induced by the nucleus-nucleus collision produce a large number of fragments with different energy, charge and mass values. This multifragmentation is predicted to be the major decay mode produced for a nuclear system at high density and temperature [55]. Thus, a complete experimental investigation, that should identify almost all the produced fragments, needs to ground on a suitable experimental device able to capture the particles that move away from the collision point in all directions [74].

These devices present specific detector cells that generate an electrical signal when hit by a particle. The availability of these detectors, however, does not automate the classification of the detected particles fragments. In fact, this task is often performed manually by visual inspection of the measured electrical quantities [42]. An efficient automatic algorithm is therefore strongly advised.

One of the first attempts to develop a fully automated algorithm for isotopic classification of the most energetic Light Charged Particles (LCP) has been presented in [102]. Here, the authors tackled the problem from a system identification point of view, identifying the dynamical system that generated the measurements.

In this chapter, we extend previous research by employing kernel methods for system identification, following the advice given in [59] (based on the separation/invariance principle) to always first model as well as possible. A model reduction step is then performed using a numerical algorithm for N4SID (subspace state-space system identification) method [62].

Kernel methods are nonparametric learning techniques that very recently undergone a large interest from the system identification community [78, 79, 95]. They are based on the definition of a kernel function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, with $\mathcal{X}$ a generic set where the input regressors belong, that embodies the properties of the functional space in which the desired function has to be searched. The main advantage is that they are shown to effectively trade off the bias/variance of the identification procedure, outperforming classical Prediction Error Method (PEM) equipped with model selection criteria such as Akaike Information Criterion (AIC) [2, 95]. The separation principle perfectly applies to these approaches. First, given data and prior information on the system behavior, fit a low-bias and minimum variance model. Then, perform a further approximation via model reduction. The prior information is used to design the kernel function employed.

In light of the previous sections, the innovative contributions of this work are three-fold:

- the author proposes the framework of Gaussian process (GP) [17, 104] to first fit a low-bias model, followed by an N4SID model reduction step, in order to model the nuclear measurements;
- the author employs for the first time (as far as the authors are aware) the stable-spline kernel [95] within a real-world experimental setting;
- the author proposes a black-box classification scheme that is tailored to the application and that highlights interpretability of its predictions.

7.2 Problem statement and experimental setup

The detector considered in this work is the large detector array CHIMERA (Charge Heavy Ions Mass and Energy Resolving Array) [1], installed at Laboratori Nazionali del Sud (Catania, Italy), see Figure 7.1.

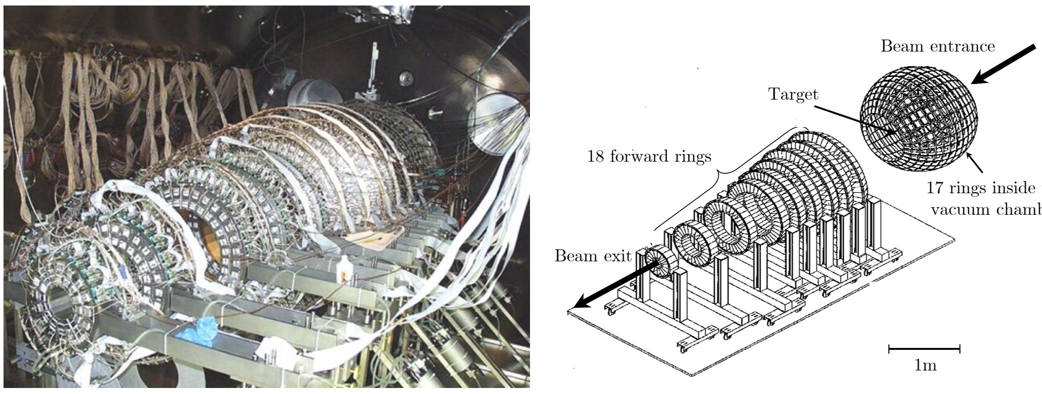

Figure 7.1: A photography of the CHIMERA detector array (left image) and a schematic representation (right image).

.

The CHIMERA detector is designed for the study of heavy-ion reactions at intermediate energy (up to $100\,{}^{MeV}/_{\mathrm{nucleon}}$). The multifragmentation phenomenon (i.e. the focus of this work) is produced by a beam of accelerated nuclei delivered by a superconducting cyclotron over a thin target, placed inside a vacuum chamber. When an accelerated nucleus collides over a target one, the hot and compressed system formed in the early stage of the collision can de-excite, leading to the generation of many fragments with a different charge, mass, and energy.

The detector is constituted by a set of 1192 detection cells arranged in 35 rings with cylindrical geometry around the beam axis. The rings are divided into two blocks. The first block is a set of 18 rings, composed of 688 detectors, arranged with cylindrical geometry. The second block is a set of 17 rings, composed of 504 detectors, placed around the target with a $0.4\,m$ radius spherical geometry inside the reaction vacuum chamber.

CHIMERA perceives the surrounding phenomena employing detection cells. Each detection cell is a telescope composed of a CsI(Tl) scintillation crystal with a thin Si detector in front of it. When hit by a particle, the CsI(Tl) element produces a light impulse. A photodiode collects the emitted light producing a current output which is converted into a measurable voltage signal $v\left(t\right)$ via a charge amplifier. Similarly, the output of the Si detector (produced by a charge displacement when hit by a particle) is fed into a preamplifier and a signal $u\left(t\right)$ is generated. The measurement chain is depicted in Figure 7.2.

The signal $v\left(t\right)$ is the most informative for the classification of LCP [102, 116]. This classification is performed by means of the pulse shape analysis, i.e. based on the hypothesis that particles with different mass and charge generate current pulses with different shape. In order to discriminate the pulse shapes, the produced impulse measurements can be modeled by an exponential law which decay rate depends on two time constants, a "fast" one τ_f, and a "slow" one τ_s [120]. The voltage signal $v\left(t\right)$ is sampled at $T_s = 10\,ns$ with a 14-bit resolution. For each pulse, 2048 samples are measured.

A set of pulses produced by known particles (manually labeled with visual methods [102]) are collected in an experiment where a beam of $20\,N^e$ at $21\,MeV$ per nucleon bombards

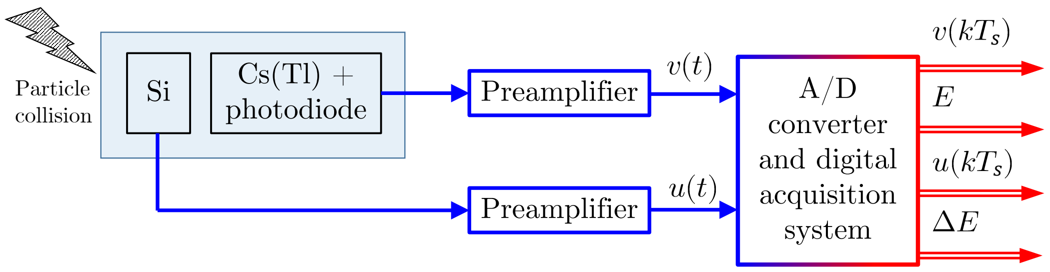

FIGURE 7.2: Measurement chain, representing analog signals (blue) and digital signals (red)

Isotope	**Atomic number (Z)**	**Atomic mass number (A)**	**Number of employed pulses**
^{1}H (protons)	1	1	904
^{2}H (deuterons)	1	2	980
^{3}H (tritons)	1	3	992
^{3}He	2	3	989
^{4}He	2	4	991
^{6}Li	3	6	989
^{7}Li	3	7	897
^{7}Be	4	7	510
^{9}Be	4	9	524
Heavy ions	≥ 5	≥ 10	979

TABLE 7.1: Dataset employed in this work

a ^{12}C target. The dataset employed for this work consists of 8751 pulses, about 20μs long, described in Table 7.1. A total of 10 different particle types are considered. Particles with the atomic number $Z \geq 5$ and with the atomic mass number $A \geq 10$ are regarded as Heavy Ions.

The next section describes the following aspect:

a) the observation motivating the modeling of the CsI(Tl) light impulse as the impulse response of an LTI;

b) the preprocessing steps performed on the raw measured data;

c) the nonparametric smoothing procedure performed utilizing Gaussian process;

d) the subspace system identification technique employed using the smoothened data.

7.3 Modeling the impulse response

7.3.1 Working assumptions

Following the results in [102], we chose to model the signal $v(t)$, measured from the CsI(Tl) detector, as the impulse response of a Single-Input Single-Output (SISO) LTI system, with transfer function $V(s)$. Based on [102] and references therein, the following dynamic system model is employed:

$$V(s) = \frac{1}{1+s\tau_m}\left(\frac{\mu_f}{1+s\tau_f}+\frac{\mu_s}{1+s\tau_s}\right), \tag{7.1}$$

where τ_f and τ_s denotes the fast and slow time constant of the light impulse response, respectively, the gains μ_f and μ_s are related to the energy of the particle, and τ_m models the dynamic response of a unitary-gain sensor. Notice how, in this case, the time constant of the sensor is higher than the phenomenon that it is measured. Furthermore, we suppose that the data are affected by a stationary zero-mean additive noise

$$\widetilde{v}(t_i) = v(t_i) + e(t_i) \qquad\qquad i = 1, \dots, 2048 \tag{7.2}$$

where $t_i = i \cdot T_s$ is the time instant of the i-th samples, $\widetilde{v}(t_i)$ is the noisy sample of the impulse response taken at the time instant t_i and $e(t_i)$ is the measurement noise. For compactness sake, from now on, the i-th sample of the impulse response, the noisy signal and the noise are indicated, respectively, with $v_i = v(t_i)$, $\widetilde{v}_i = \widetilde{v}(t_i)$ and $e_i = e(t_i)$. Then equation (7.2) can be rewritten as

$$\widetilde{v}_i = v_i + e_i \qquad\qquad i = 1, \dots, 2048 \tag{7.3}$$

7.3.2 Preprocessing steps

A set of preprocessing steps has been performed on raw data. These precautions are mandatory for the application of the subsequent modeling steps. An example of measured impulse response is shown in Figure 7.3. It is possible to observe a “deadzone” prior to the impulse’s starting. This is due to the post-triggering acquisition setup and acquisition chain’s offsets. Thus, two actions are mandatory: **(i)** the baseline removal and **(ii)** the detection of the impulse starting time.

The baseline removal process is made by fit a line on the first $4\,\mu s$ of the measurement, such that $g_i = m \cdot i + l$, with $m, l \in \mathbb{R}$ the line’s coefficients. The fitted line is then removed from the measurements, obtaining the signal $z_i = \widetilde{v}_i - g_i$. The obtained signal is depicted in Figure 7.4.

The detection of the starting time required special treatment since impulses have different amplitudes and shapes. The following procedure was devised by the authors:

1. The discrete time derivative of z_i is computed

$$dz_i = \frac{z_i - z_{i-1}}{T_s}. \tag{7.4}$$

   A first estimate, i.e. k_1, of the initial condition is made when dz_i exceeds a predefined threshold;

2. A third order polynomial $p(t)$ is fit on the 10 points after k_1;

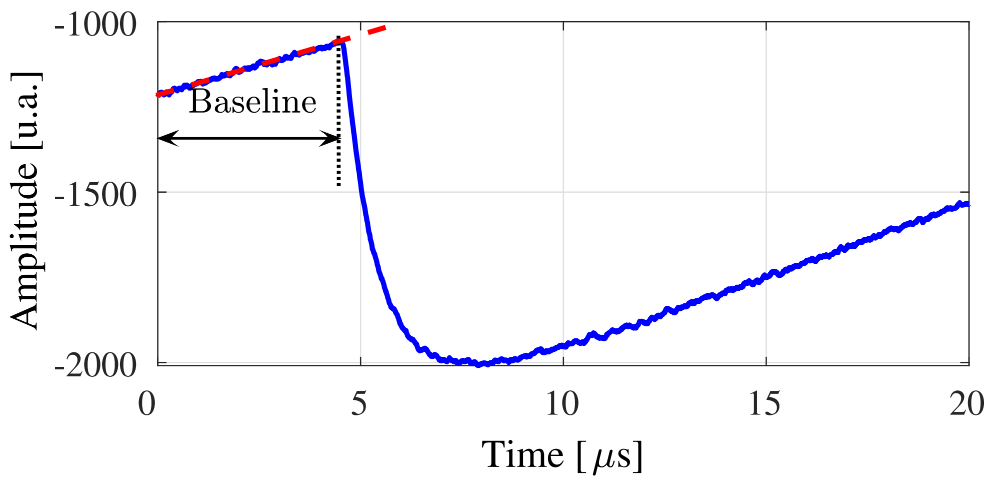

FIGURE 7.3: Example of a measured v_i response (blue). The baseline value is highlighted with its fitted line (dotted red).

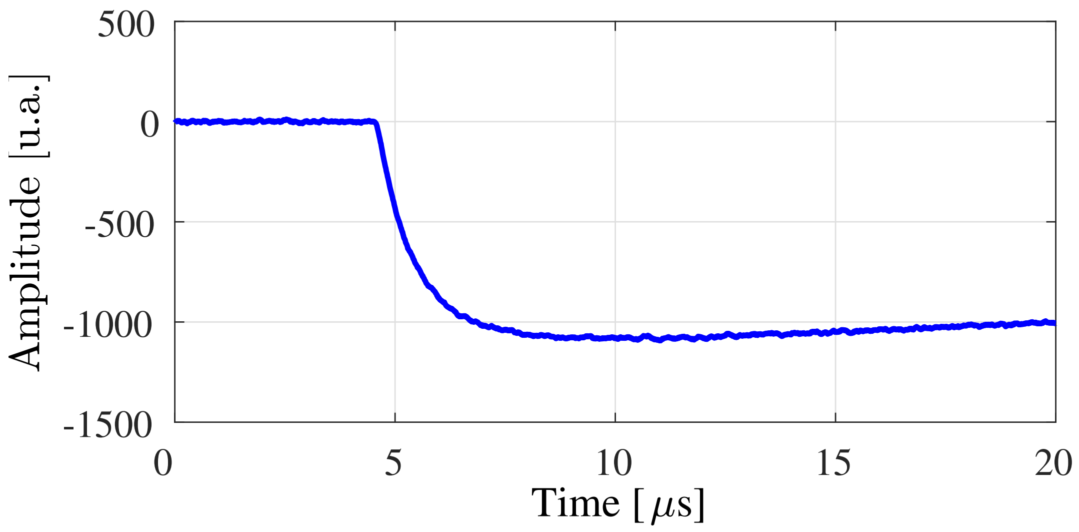

FIGURE 7.4: Example of a computed signal z_i after baseline removal.

3. The root r of $p(t)$ that is nearest to k_1 is computed. The nearest sampled point k_2 successive to r is taken as the first non-null impulse sample;
4. The starting point $\bar{i}$ is taken as the time instant before k_2, posing $z_{\bar{i}} = 0$. Samples before $\bar{i}$ are deleted. We denote the final preprocessed signal as y_i, with $i = 1, \ldots, n$, where n is the length of the particular measurement (since the baseline length is different for each acquisition, the cleaned data can have different lengths).

The procedure is depicted in Figure 7.5 after that the baseline was removed. Each impulse now lasts about $16\,\mu s$. The last caution was to multiply the data for minus one, in order to obtain an impulse response of a system with positive gain, as should be from physics relations.

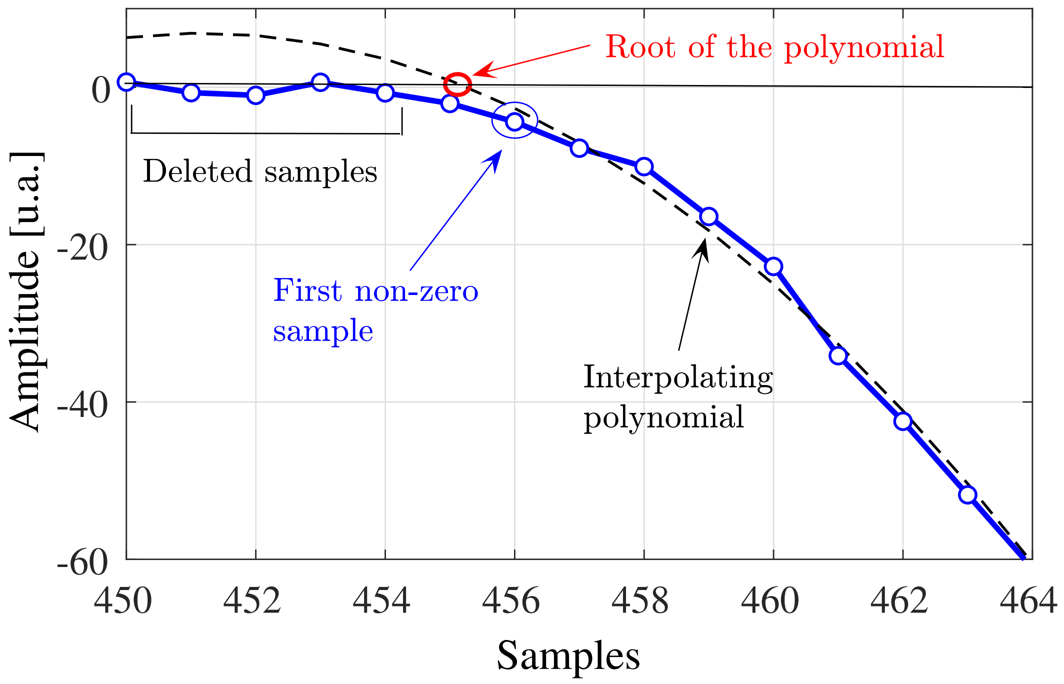

FIGURE 7.5: The rationale for choosing the starting time.

7.3.3 NONPARAMETRIC SYSTEM IDENTIFICATION

Adhering to the rationale presented in the chapter introduction, we chose to use the framework of Gaussian process (GP) to model the time-domain impulse responses, presented in Section 2.2. In this way, a low-bias and flexible model is obtained. The estimated response is the minimum variance estimate when error measurements and data are considered as Gaussian random variables. Given that the data are interpreted as impulse responses, the use of the stable-spline kernel is the most natural choice. This is a particular type of kernel function that has been designed in order to model LTI systems [95, 96].

We employed the so called continuous-time second-order stable-spline kernel [95]:

$$k\left(a,b\right) = \lambda \left(\frac{e^{-\beta(a+b+\max(a,b))}}{2} - \frac{e^{-3\beta \max(a,b)}}{6} \right), \tag{7.5}$$

where $a, b \in \Omega \subseteq \mathbb{R}_+$ are generic continuous-time instants, and $\lambda, \beta \in \mathbb{R}_+$ are hyperparameters that determine the shape of the kernel function (and therefore of the estimated one). Since, in this scenario, the function that we want to estimate is an impulse response, the domain of the kernel is the continuous-time. Thus, the regressors are the time instants of the measurements. A more in-details explanation of this kernel can be found in Section 2.1 and in 4.3.

Consider now the vector $\boldsymbol{y} \in \mathbb{R}^{1\times n}$ formed by stacking the impulse response samples y_i. As stated in (7.2), we can model the measurements as

$$\boldsymbol{y} = \boldsymbol{g} + \boldsymbol{e} \tag{7.6}$$

where $\boldsymbol{g} \in \mathbb{R}^{1\times n}$ contains the noiseless data $\boldsymbol{g}_i$, i.e. the noiseless version of y_i, and $\boldsymbol{e} \in \mathbb{R}^{1\times n}$ contains the noise terms e_i. We will suppose now that the errors e_i are IID with variance σ^2. The distribution of the observed values given the noiseless ones is (omitting the dependence on the input variables):

$$p(\boldsymbol{y}\,|\boldsymbol{g}) = \mathcal{N}\left(\boldsymbol{y}^\top \left|\boldsymbol{g}^\top, \sigma^2 \boldsymbol{I}_n\right.\right), \tag{7.7}$$

From the Gaussian process definition [17], the marginal distribution $p(\boldsymbol{g})$ is Gaussian with zero mean and variance defined by the kernel matrix $\boldsymbol{K} \in \mathbb{R}^{n\times n}$:

$$p(\boldsymbol{g}) = \mathcal{N}\left(\boldsymbol{g}^\top \left|\boldsymbol{0}_{n\times 1}, \boldsymbol{K}\right.\right) \tag{7.8}$$

The matrix $\boldsymbol{K}$ (also known as Gram matrix) is a symmetric semidefinite positive matrix whose (i, j) entry is $k(t_i, t_j)$ Therefore, instead of placing a prior on the parameters, we put a prior over the noiseless data $\boldsymbol{g}$.

The marginal distribution of $\boldsymbol{y}$ can be found by marginalizing over $\boldsymbol{g}$, using known properties of Gaussian distributions (see Equation (2.115) of [17], as:

$$\begin{aligned} p(\boldsymbol{y}) &= \int p(\boldsymbol{y}\,|\boldsymbol{g})\, p(\boldsymbol{g})\; d\boldsymbol{g} \qquad (7.9)\\ &= \mathcal{N}\left(\boldsymbol{y}|\boldsymbol{0}_{n\times 1}, \boldsymbol{K} + \sigma^2 \boldsymbol{I}_n\right) \qquad (7.10)\\ &= \mathcal{N}\left(\boldsymbol{y}|\boldsymbol{0}_{n\times 1}, \boldsymbol{Z}(\boldsymbol{\zeta})\right) \qquad (7.11)\end{aligned}$$

where

$$\boldsymbol{\zeta} = \begin{bmatrix} \lambda & \beta & \sigma \end{bmatrix}^\top \in \mathbb{R}^{3\times 1} \tag{7.12}$$

contains the hyperparameters of the method.

The prediction of the output sample taken at time $t^* \in \mathbb{R}$ can be obtained as the expected value of the predictive distribution $p(y^*\,|\boldsymbol{y}, t^*)$. This distribution can be computed by applying standard formulas for conditioned Gaussian distributions (see Equations (2.81) – (2.82) of [17]). Its expected value is:

$$y^* = \boldsymbol{k}(t^*)^\top \boldsymbol{Z}(\boldsymbol{\zeta})^{-1} \boldsymbol{y}^\top, \tag{7.13}$$

where

$$\boldsymbol{k}(t^*) = \begin{bmatrix} k(t_i, t^*) & \cdots & k(t_i, t^*) \end{bmatrix}^\top \in \mathbb{R}^{n\times 1} \tag{7.14}$$

In this work, we only perform smoothing: the test data are equal to the train data.

For each impulse response, we performed a hyperparameters optimization procedure, by maximizing the marginal likelihood [17] (see Section 1.5.3 for more details). This technique consists into maximizing the marginal likelihood of the data,that depends on $\boldsymbol{\zeta}$, given by (7.9). An estimate of the values of the hyperparameters can be obtained as [17]:

$$\widehat{\boldsymbol{\zeta}} = \underset{\boldsymbol{\zeta}\in\mathbb{R}^{3\times 1}}{\arg\min} \left\{\boldsymbol{y}^\top \boldsymbol{Z}(\boldsymbol{\zeta})^{-1} \boldsymbol{y} + \log\det\left(\boldsymbol{Z}(\boldsymbol{\zeta})\right)\right\} \tag{7.15}$$

To efficiently compute (7.15), the Cholesky decomposition [52] of the matrix $\boldsymbol{Z}(\boldsymbol{\zeta})$ is used as explained in Section 1.5. The results of the applied procedure are shown in Figure 7.6, where it can be observed how the method has efficiently reduced the noise present in the data.

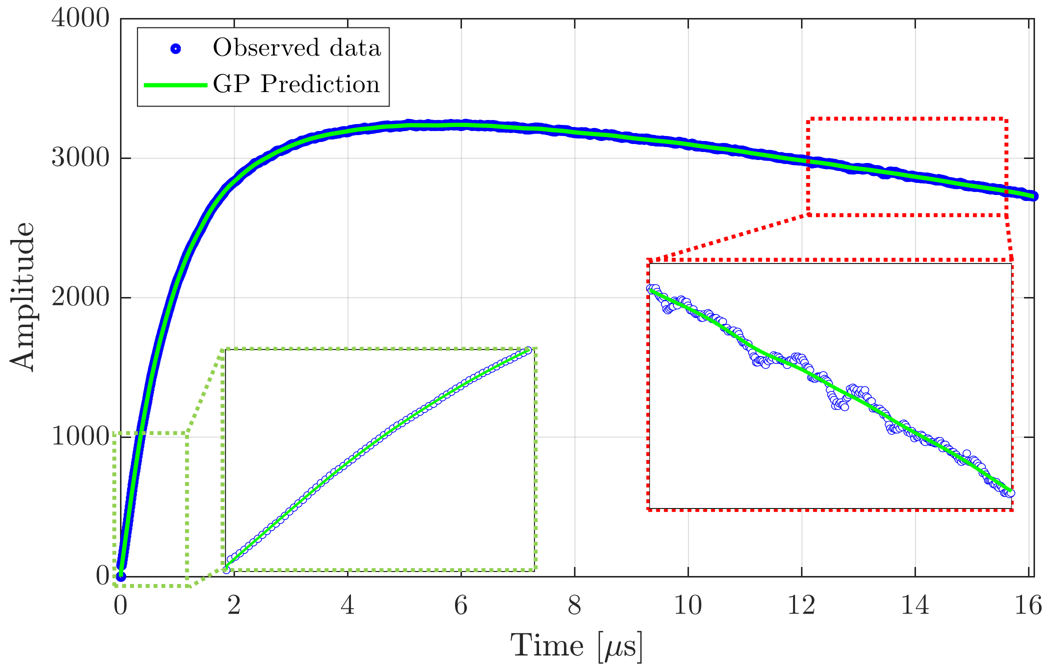

FIGURE 7.6: Example of a measured impulse response (blue) with superimposed Gaussian process prediction (green). The smoothing effect is clearly visible.

7.3.4 SUBSPACE SYSTEM IDENTIFICATION

We now turn our attention to the identification of the system (7.1). Consider the state-space representation of a discrete-time SISO LTI system:

$$x_{i+1} = \boldsymbol{A}\boldsymbol{x}_i + \boldsymbol{B}u_i \tag{7.16}$$

$$y_i = \boldsymbol{C}\boldsymbol{x}_i + \boldsymbol{D}u_i, \tag{7.17}$$

where $\boldsymbol{x}_i \in \mathbb{R}^{p\times 1}$, $u_i \in \mathbb{R}$ and $y_i \in \mathbb{R}$ are the system state (of dimension p), input and output, respectively. We set $D = 0$ since we preprocessed the impulse data to start from zero. With the data obtained by the flexible model devised in the previous section, a minimum-order realization of (7.16) can be found by employing the N4SID procedure described in [62, 102].

The method briefly consists into creating a Hankel matrix $\boldsymbol{H}$ composed by the noisy impulse measurements. The Singular Value Decomposition (SVD) is then employed to suitably reduce the rank of H to the chosen model order. With the reduced Hankel matrix, it is possible to obtain an estimate of the Observability and Reachability matrices of the system, from which an estimate $\left\{\widehat{\boldsymbol{A}}, \widehat{\boldsymbol{B}}, \widehat{\boldsymbol{C}}\right\}$ can be computed.

Instead of creating the matrix $\boldsymbol{H}$ with the noisy data, the idea is to use the smoothened ones, and apply the N4SID procedure. This approach permits to avoid optimization procedures that can get stuck in local minima, i.e. estimating the parameters of a predefined transfer function. As further check, the inspection of the SVD singular values showed that the order of the system is indeed three.

After that the matrices $\left\{\widehat{\boldsymbol{A}}, \widehat{\boldsymbol{B}}, \widehat{\boldsymbol{C}}\right\}$ are available, an estimate of the unknown parameters of (7.1), i.e. $\{\mu_f, \mu_s, \tau_f, \tau_s, \tau_m\}$ can be computed by converting the discrete system into a continuous one. It should be noticed that this conversion can produce a couple of complex poles, that do not adhere with the modeling of (7.1). Those tests were discarded, resulting in the dataset of Table 7.1. We leave to future research the case where N4SID results are used as the initial condition for an optimization procedure.

The results of the N4SID procedure are perfectly in line with those obtained in [102]. Boxplots of the estimates are shown in Figures 7.7, 7.8, 7.9 and 7.10.

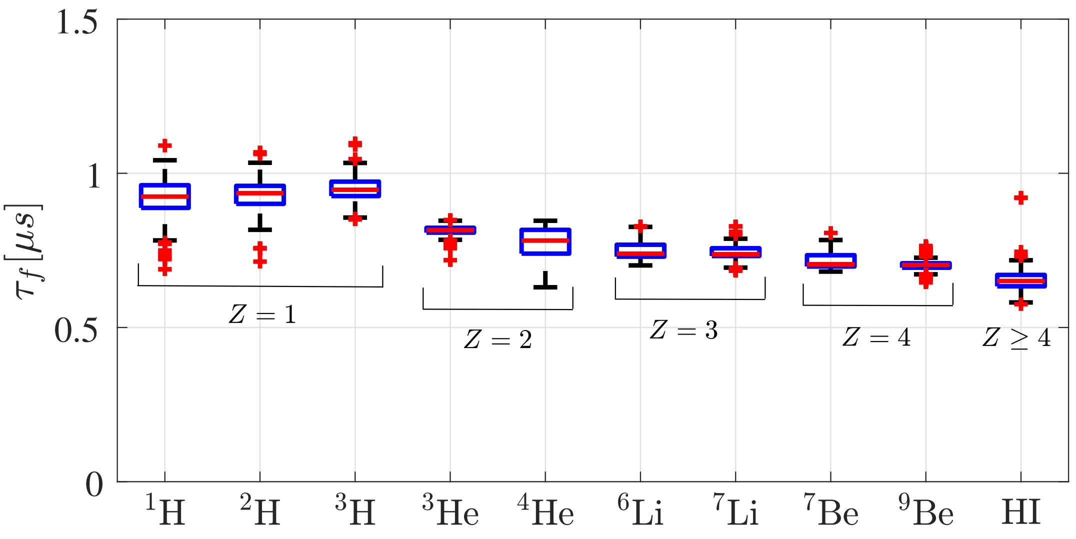

FIGURE 7.7: Fast time constant.

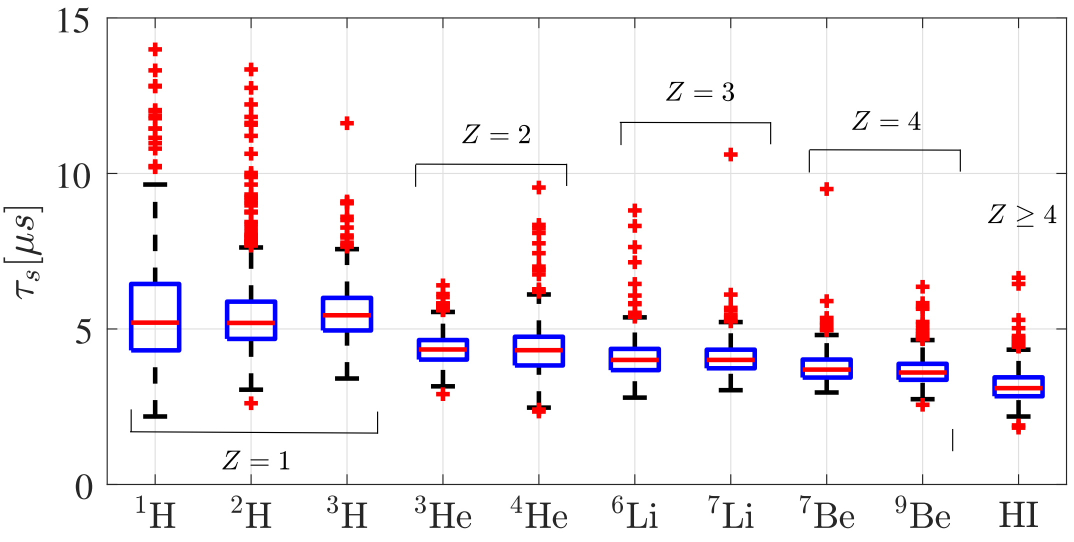

FIGURE 7.8: Slow time constant.

7.4 PARTICLES CLASSIFICATION

A particle type is completely defined by its charge, given by its atomic number Z, and its mass, given by its atomic mass number A. In the previous sections, we applied a system identification point of view to characterize each impulse response of LCP ($Z \leq 4, A \leq 9$). Following the separation principle, we first fit a low-bias model with Gaussian process regression. Then, a model reduction has been performed. Each measurement is now condensed in an estimate of the parameters $\{\mu_f, \mu_s, \tau_f, \tau_s, \beta\}$.

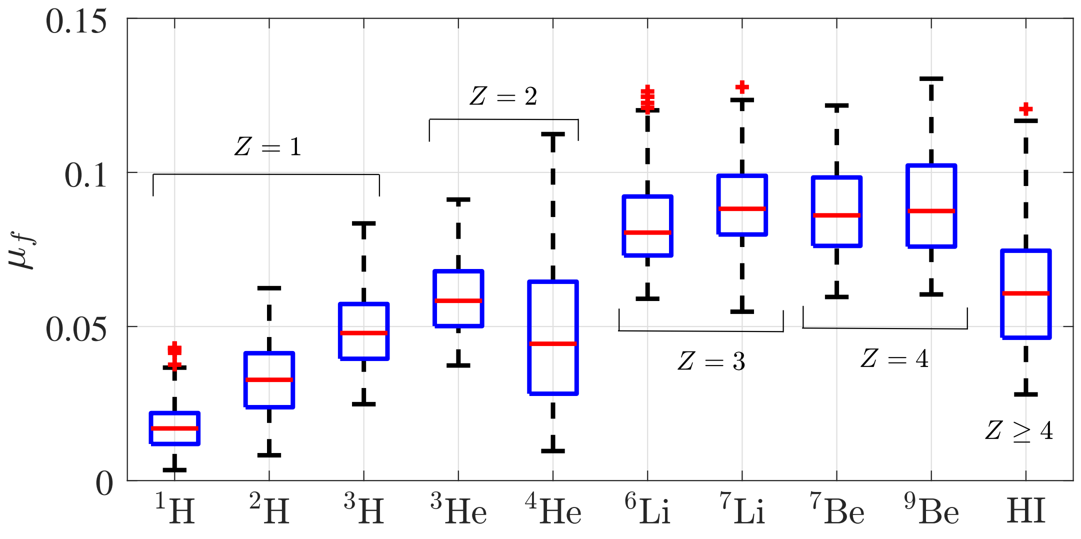

FIGURE 7.9: Gain of the fast component.

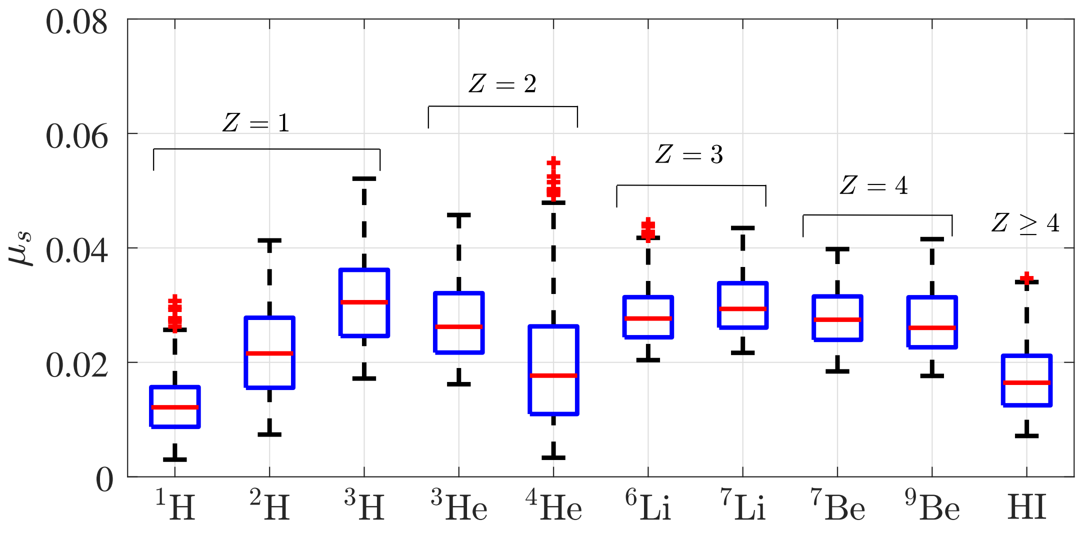

FIGURE 7.10: Gain of the slow component.

We can now represent each impulse response as a feature vector

$$\phi = \begin{bmatrix} \mu_f & \mu_s & \tau_f & \tau_s & \beta \end{bmatrix}^\top \in \mathbb{R}^{5\times 1} \tag{7.18}$$

A feedforward neural network (NN) [17] is trained to predict, for each observation, its atomic number Z and atomic mass number A. The choice of using a NN model relies on the fact that it can efficiently handle multi-dimensional outputs as in this case. In fact, it is crucial to take into account label correlations during the classification process [50].

The NN is composed of 2 hidden layers with 10 neurons each, and a final layer with 2 outputs. The hidden layers have a hyperbolic tangent activation function. The NN structure has been chosen by cross-validation. The output layer has a linear activation function. The labeled outputs consist in the couple

$$Q = \begin{bmatrix} A & Z \end{bmatrix}^\top \in \mathbb{R}^{2\times 1}. \tag{7.19}$$

The training data were standardized to zero mean on unitary variance. The same transformation, with mean and variance computed on the training set, is applied to the test data. The training of the NN has been performed using the well documented Levenberg-Marquardt minimization algorithm [56]. The NN predicts a vector

$$\boldsymbol{q} = \begin{bmatrix} q_1 & q_2 \end{bmatrix}^\top \in \mathbb{R}^{2\times 1} \tag{7.20}$$

which is the real-valued prediction of A and Z. The predictions were then rounded to the nearest integer value. The test set consisted of 100 samples from each type of particle.

The predictions of the NN model are then fed to a second classifier. A decision tree [44] is employed to predict the type of each particle. The inputs are the estimated values of A and Z, while the output is an integer number that represents the class of each observation. The complete classification procedure is reported in Figure 7.11. We could have employed just one classifier, mapping the features vectors directly to the particle classes. However, the proposed chain of classifiers is not only tailored to the classification of different particles, but it is also highly interpretable because they can be clustered according to the predicted atomic number Z and atomic mass number A, as will be shown in the next section.

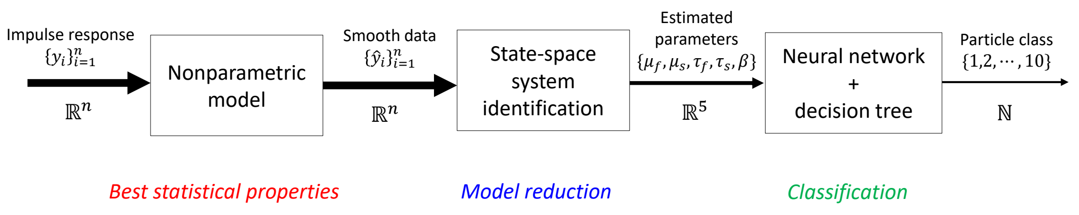

FIGURE 7.11: Schematic of the classification procedure.

7.5 Results and discussion

Several observations can be made from the results of Figures 7.7, 7.8, 7.9 and 7.10. The mean value of the fast time constant τ_f and of the slow one τ_s decreases (tendentially) with the atomic number Z. The standard deviation also decreases. The gains μ_f and μ_s tend to increase with Z and A, apart for the heavy ions (HI) and the ^{4}He particles. The hyperparameter β increases with Z. This is in line with the behavior of τ_f and τ_s. In fact, lower time constants indicate a higher decay rate. This is the exact information that β encodes. These estimates are in line with the literature [102].

The classification results of the proposed approach are compared with the method proposed in [102]. Here, the author directly performed the N4SID step on noisy impulse data $\widetilde{v}\,(k)$ (after data preprocessing). Notice how the task is quite challenging because the previous results obtained very high classification rates.

In this work, we reimplemented the method proposed in [102] to make the comparison. The purpose is to test the effectiveness of the proposed two-step identification procedure. The classification accuracies are reported in Figure 7.12 and Figure 7.13. The heatmaps represent the percentage of corrected classifications, comparing the predicted particle types with the known ones. Darker colors indicate a higher classification accuracy. The proposed approach obtained a classification accuracy of 96%. The method in [102] correctly classified the 93% of the test particles. It is important to emphasize how a 3% improvement in classification accuracy is a significant contribution to this problem since this is important to determine the properties of investigated physical phenomena. Figure 7.14 plots a subset of the test samples along with the classification bounds discovered by the decision tree. Notice how the learned bounds are very intuitive and could be set by human visual inspection.

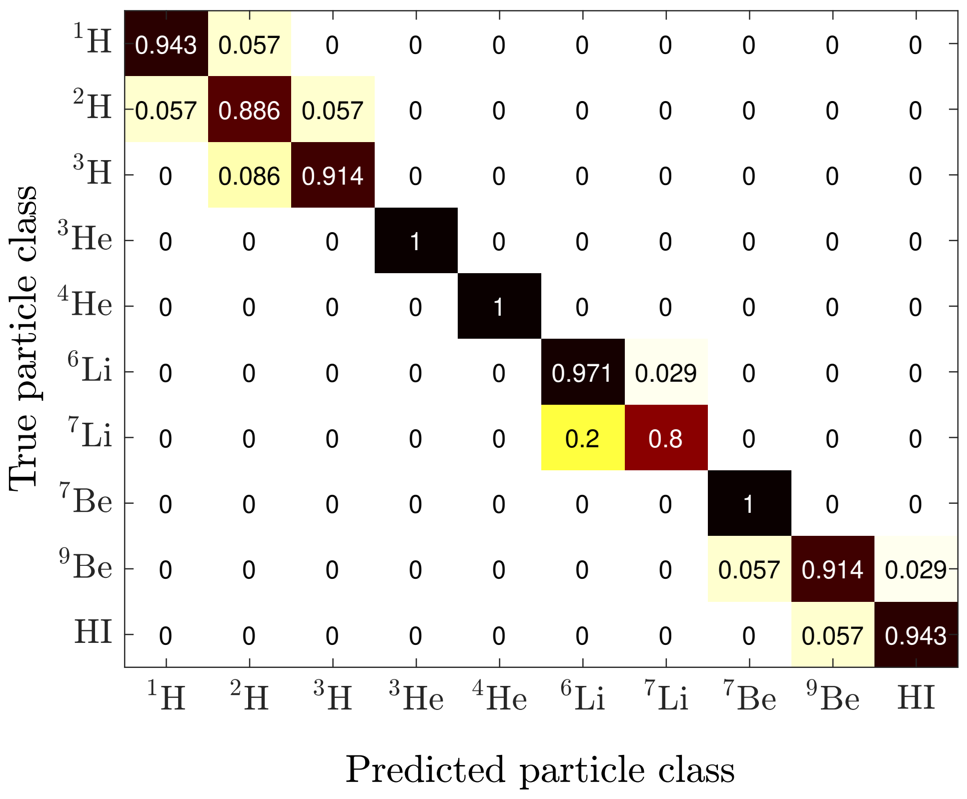

FIGURE 7.12: Classification results of the method in [102].

7.6 Conclusions

In this chapter, the use of the Gaussian process framework to identify a low-bias dynamic model is investigated. The flexibility of GP allows capturing the dynamics that are required

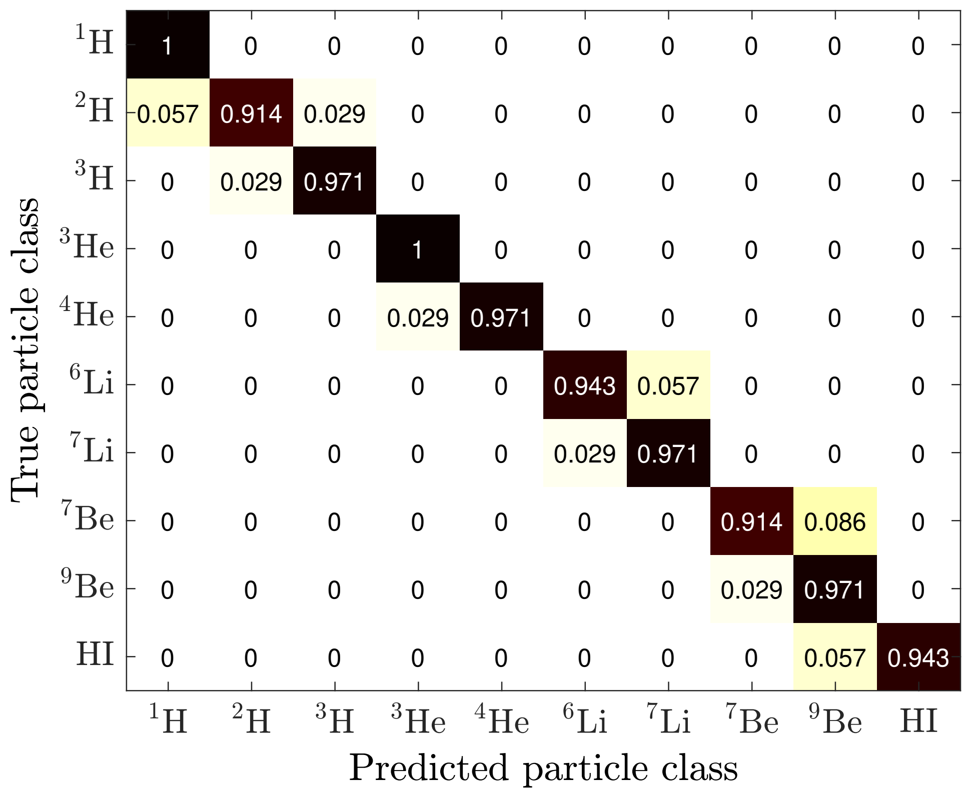

FIGURE 7.13: Classification results of the proposed method.

for a specific application. If a low-order model is needed, a model reduction technique can be employed as a subsequent step. This rationale has been applied to the classification of Light Charged Particles. First, a nonparametric model has been identified, employing a specific kernel function developed for linear system identification. Then, the model reduction step is performed via a subspace identification method.

The parameters of the identified system are fed to a combination of classifiers to predict the particle type. The classification procedure is a black-box model that is, however, highly interpretable. Results showed how the combination of nonparametric and parametric modeling improved the classification accuracy of the previous method, that did not leverage the nonparametric modeling step. Further research is devoted to a better investigation of the sensor's model, comparison with other model reduction techniques and the design of an ad-hoc kernel [30].

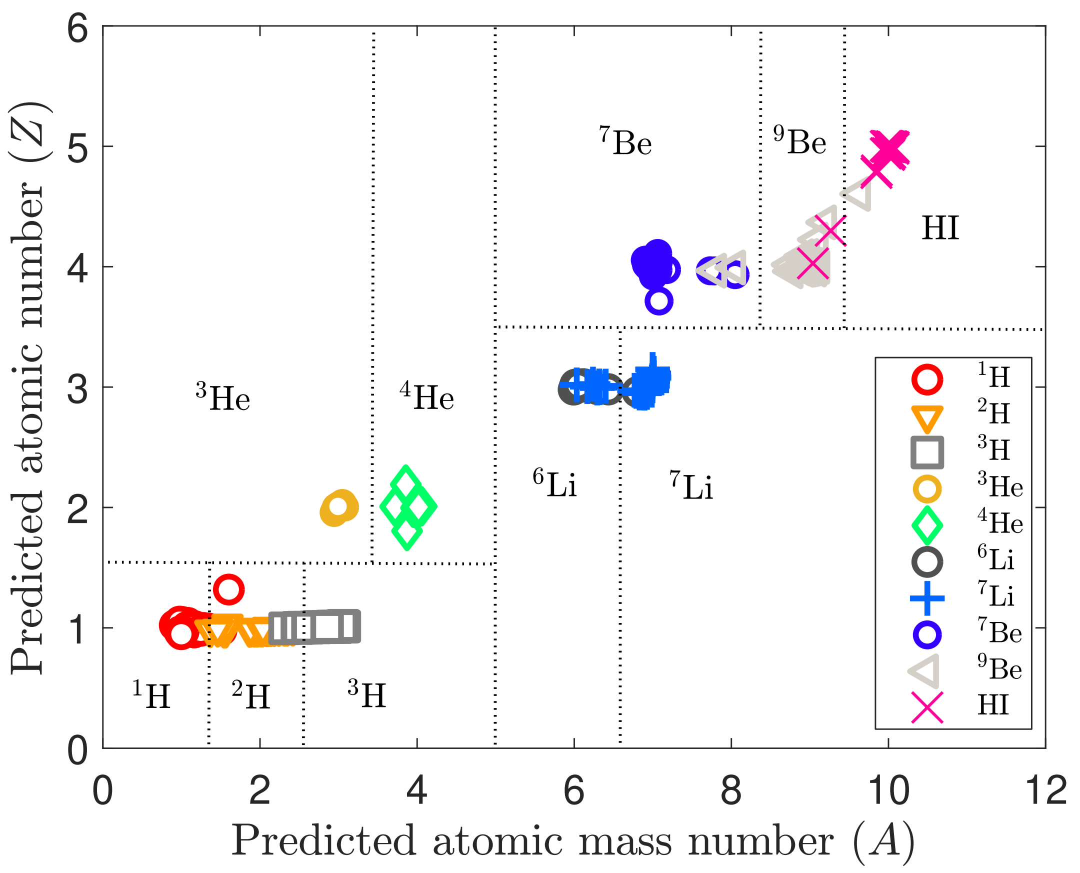

FIGURE 7.14: A subset of test samples with the classification bounds learned by the decision tree.

Conclusion

This Thesis proposed four different theoretical contributions to the kernel-based system identification research field.

Firstly, in Chapter 3, it is shown that, thanks to the limited computational precision, there exists an infinite amount of equivalent solutions of the kernel-based regression problem. This newfound freedom is then exploited to compute the solution that minimizes the computational complexity of the estimated model. Furthermore, it is proposed a new approach that can be used to attenuate the intrinsic ill-conditioning of the semi-supervised manifold regularization. Therefore, this new point of view to the solution of kernel-based regression spawns promising approaches that should encourage new researches on the topic.

As a second contribution, in Chapter 4, a novel black-box non-parametric continuous-time LTI identification technique that employs the RKHS properties is presented. This methodology, based on the work of [95], identifies directly the transfer function of the system and it can work with non-regularly sampled data-points. This method has shown very good performance even when employed with low-exciting input signals with respect to the method proposed in the literature [45, 46].

The third contribution introduces the concept of semi-supervised manifold regularization in case of nonlinear dynamical systems as shown in Chapter 5. Here, a combination of an algorithm that generates new unsupervised points and a criterion for the selection of the underlying regressors graph topology is proposed as new contributions. Results showed that this regularization technique may outperform the classical Tikhonov regularization for the identification of nonlinear dynamical systems.

Lastly, in Chapter 6, a Bayesian perspective of the manifold regularization is presented as the fourth contribution. In particular, it is shown that a new likelihood term can be coupled with the standard one and with a Gaussian process prior in order to obtain the desired effect. Then, thanks to this new perspective, the method hyper-parameters can be tuned using the marginal likelihood maximization approach. Monte Carlo simulations were performed on a benchmark dynamical system. From the results, it is clear that the proposed tuning procedure may increase the performance in some cases.

Finally, Chapter 7 presents an application of kernel-based learning techniques on a practical application with real data in the field of nuclear physics. In particular, the nuclear reaction induced by the nucleus-nucleus collision produces a certain quantity of energy that decays over time. The aim was the classification of the type of particles using the measured energy produced after a collision. These time-series were modeled as an impulse response of an LTI system. Then, a kernel-based approach was employed to identify the dynamical features of this system and a decision tree is used to classify the particles using the identified features of the LTI system. This approach has shown nearly perfect classification performance.

Bibliography

[1] S. Aiello, A. Anzalone, M. Baldo, G. Cardella, S. Cavallaro, E. De Filippo, A. Di Pietro, S. Femino, P. Figuera, P. Guazzoni, C. Iacono-Manno, G. Lanzanò, U. Lombardo, S. Lo Nigro, A. Musumarra, A. Pagano, M. Papa, S. Pirrone, G. Politi, F. Porto, A. Rapisarda, F. Rizzo, S. Sambataro, M.L. Sperduto, C. Sutera, and L. Zetta. Chimera: a project of a 4π detector for heavy ion reactions studies at intermediate energy. *Nuclear Physics A*, 583:461–464, feb 1995.

[2] Hirotugu Akaike. A new look at the statistical model identification. In *Springer Series in Statistics*, pages 215–222. Springer New York, 1974.

[3] S. H. Al-Amer and F. M. Al-Sunni. Approximation of time-delay systems. In *Proceedings of the 2000 American Control Conference. ACC (IEEE Cat. No.00CH36334)*, volume 4, pages 2491–2495 vol.4. IEEE, June 2000.

[4] N. Aronszajn. Theory of reproducing kernels. *Transactions of the American Mathematical Society*, 68(3):337–404, mar 1950.

[5] Er-Wei Bai. Identification of an additive NFIR system and its applications in generalized hammerstein models. In *Proceedings of the 44th IEEE Conference on Decision and Control*, pages 6406–6411. IEEE, Dec 2005.

[6] Er-Wei Bai, Roberto Tempo, and Yun Liu. Identification of IIR nonlinear systems without prior structural information. *IEEE Transactions on Automatic Control*, 52(3):442–453, March 2007.

[7] George A. Baker and Peter Graves-Morris. *Padé approximants*, volume 59. Cambridge University Press, 1996.

[8] Bassam Bamieh and Laura Giarré. Identification of linear parameter varying models. *International Journal of Robust and Nonlinear Control*, 12(9):841–853, jul 2002.

[9] M. Belkin. *Problems of Learning on Manifolds.* PhD thesis, 2003. AAI3097083.

[10] Mikhail Belkin and Partha Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computation*, 15(6):1373–1396, jun 2003.

[11] Mikhail Belkin, Partha Niyogi, and Vikas Sindhwani. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *Journal of machine learning research*, 7(Nov):2399–2434, 2006.

[12] Dennis S Bernstein. *Matrix mathematics: theory, facts, and formulas.* Princeton university press, July 2009.

[13] Tyrus Berry and John Harlim. Variable bandwidth diffusion kernels. *Applied and Computational Harmonic Analysis*, 40(1):68 – 96, jan 2016.

[14] Tyrus Berry and Timothy Sauer. Local kernels and the geometric structure of data. *Applied and Computational Harmonic Analysis*, 40(3):439–469, may 2016.

[15] Tyrus Berry and Timothy Sauer. Consistent manifold representation for topological data analysis. *Foundations of Data Science*, (0):0–0, 2019.

[16] Dimitri P Bertsekas. *Constrained optimization and Lagrange multiplier methods*. Academic press, 2014.

[17] Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

[18] Mauro Bisiacco and Gianluigi Pillonetto. Kernel absolute summability is only sufficient for rkhs stability, 2019.

[19] Sergio Bittanti. *Model Identification and Data Analysis*. John Wiley and Sons Ltd, mar 2019.

[20] M. Bonin, V. Seghezza, and L. Piroddi. NARX model selection based on simulation error minimisation and LASSO. *IET Control Theory & Applications*, 4(7):1157–1168(11), July 2010.

[21] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, March 2019.

[22] Emmanuel J. Candès, Michael B. Wakin, and Stephen P. Boyd. Enhancing sparsity by reweighted ℓ 1 minimization. *Journal of Fourier Analysis and Applications*, 14(5):877–905, Dec 2008.

[23] Gunnar Carlsson. Topology and data. *Bulletin of the American Mathematical Society*, 46(2):255–308, jan 2009.

[24] V. Castelli and T.M. Cover. The relative value of labeled and unlabeled samples in pattern recognition with an unknown mixing parameter. *IEEE Transactions on Information Theory*, 42(6):2102–2117, Nov 1996.

[25] Lawrence Cayton. Algorithms for manifold learning. *Univ. of California at San Diego Tech. Rep*, 12(1-17):1, 2005.

[26] Vito Cerone, Dario Piga, and Diego Regruto. Set-membership LPV model identification of vehicle lateral dynamics. *Automatica*, 47(8):1794 – 1799, aug 2011.

[27] Olivier Chapelle, Bernhard Scholkopf, and Alexander Zien. Semi-supervised learning. *IEEE Transactions on Neural Networks*, 20(3):542–542, 2009.

[28] S. Chen, S. A. Billings, and P. M. Grant. Non-linear system identification using neural networks. *International Journal of Control*, 51(6):1191–1214, jan 1990.

[29] S.L. Chen, H.C. Lai, and K.C. Ho. Identification of linear time varying systems by haar wavelet. *International Journal of Systems Science*, 37(9):619–628, jul 2006.

[30] Tianshi Chen. On kernel design for regularized lti system identification. *Automatica*, 90:109 – 122, April 2018.

[31] Tianshi Chen and Lennart Ljung. Implementation of algorithms for tuning parameters in regularized least squares problems in system identification. *Automatica*, 49(7):2213 – 2220, jul 2013.

[32] Tianshi Chen, Henrik Ohlsson, and Lennart Ljung. On the estimation of transfer functions, regularizations and gaussian processes—revisited. *Automatica*, 48(8):1525 – 1535, August 2012.

[33] Tianshi Chen and Gianluigi Pillonetto. On the stability of reproducing kernel hilbert spaces of discrete-time impulse responses. *Automatica*, 95:529 – 533, sep 2018.

[34] Fan Chung and Mary Radcliffe. On the spectra of general random graphs. *the electronic journal of combinatorics*, 18(1):215, 2011.

[35] R. R. Coifman, S. Lafon, A. B. Lee, M. Maggioni, B. Nadler, F. Warner, and S. W. Zucker. Geometric diffusions as a tool for harmonic analysis and structure definition of data: Diffusion maps. *Proceedings of the National Academy of Sciences*, 102(21):7426–7431, may 2005.

[36] Ronald R. Coifman and Stéphane Lafon. Diffusion maps. *Applied and Computational Harmonic Analysis*, 21(1):5–30, jul 2006. Special Issue: Diffusion Maps and Wavelets.

[37] Mohamed Abdelmonim Hassan Darwish, Pepijn Bastiaan Cox, Ioannis Proimadis, Gianluigi Pillonetto, and Roland Tóth. Prediction-error identification of LPV systems: A nonparametric gaussian regression approach. *Automatica*, 97:92–103, nov 2018.

[38] P. Delsarte and Y. Genin. The split levinson algorithm. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 34(3):470–478, jun 1986.

[39] Francesco Dinuzzo. Kernels for linear time invariant system identification. *SIAM Journal on Control and Optimization*, 53(5):3299–3317, jan 2015.

[40] Francesco Dinuzzo and Bernhard Schölkopf. The representer theorem for hilbert spaces: a necessary and sufficient condition. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 189–196. Curran Associates, Inc., 2012.

[41] T.W. Flint and R.J. Vaccaro. Performance analysis of n4sid state-space system identification. In *Proceedings of the 1998 American Control Conference. ACC (IEEE Cat. No.98CH36207)*, volume 5, pages 2766–2767 vol.5. IEEE, June 1998.

[42] A.S. Fomichev, I. David, S.M. Lukyanov, Yu.E. Penionzhkevich, N.K. Skobelev, O.B. Tarasov, A. Matthies, H.-G. Ortlepp, W. Wagner, M. Lewitowicz, M.G. Saint-Laurent, J.M. Corre, Z. Dlouhý, I. Pecina, and C. Borcea. The response of a large CsI(tl) detector to light particles and heavy ions in the intermediate energy range. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 344(2):378–383, may 1994.

[43] Simone Formentin, Mirko Mazzoleni, Matteo Scandella, and Fabio Previdi. Nonlinear system identification via data augmentation. *Systems & Control Letters*, 128:56 – 63, jun 2019.

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

[45] H. Garnier and M. Gilson. CONTSID: a matlab toolbox for standard and advanced identification of black-box continuous-time models. *IFAC-PapersOnLine*, 51(15):688 – 693, 2018. 18th IFAC Symposium on System Identification SYSID 2018.

[46] Hugues Garnier. Direct continuous-time approaches to system identification. overview and benefits for practical applications. *European Journal of Control*, 24:50 – 62, jul 2015. SI: ECC15.

[47] Hugues Garnier, Liuping Wang, and Peter C. Young. Direct identification of continuous-time models from sampled data: Issues, basic solutions and relevance. In *Identification of Continuous-time Models from Sampled Data*, pages 1–29. Springer London, 2008.

[48] Wodek Gawronski and Jer-Nan Juang. Model reduction in limited time and frequency intervals. *International Journal of Systems Science*, 21(2):349–376, feb 1990.

[49] Robert Ghrist. Barcodes: The persistent topology of data. *Bulletin of the American Mathematical Society*, 45(01):61–76, oct 2007.

[50] Shantanu Godbole and Sunita Sarawagi. Discriminative methods for multi-labeled classification. In *Pacific-Asia conference on knowledge discovery and data mining*, pages 22–30. Springer, Springer Berlin Heidelberg, 2004.

[51] Arash Golabi, Nader Meskin, Roland Toth, and Javad Mohammadpour. A bayesian approach for LPV model identification and its application to complex processes. *IEEE Transactions on Control Systems Technology*, 25(6):2160–2167, nov 2017.

[52] Gene H. Golub. *Matrix Computations*. J. Hopkins Uni. Press, January 2013.

[53] I. S. Gradshteyn and I. M. Ryzhik. *Table of Integrals, Series, and Products*. Academic Press, 2014.

[54] Leo J. Grady and Jonathan R. Polimeni. *Discrete Calculus*. Springer London, 2010.

[55] D H E Gross. Statistical decay of very hot nuclei-the production of large clusters. *Reports on Progress in Physics*, 53(5):605–658, may 1990.

[56] M. T. Hagan and M. B. Menhaj. Training feedforward networks with the marquardt algorithm. *IEEE Transactions on Neural Networks*, 5(6):989–993, Nov 1994.

[57] David A. Harville. Matrix algebra from a statistician's perspective. *Technometrics*, 40(2):164–164, may 1998.

[58] Matthias Hein, Jean-Yves Audibert, and Ulrike von Luxburg. From graphs to manifolds – weak and strong pointwise consistency of graph laplacians. In *Learning Theory*, pages 470–485. Springer Berlin Heidelberg, 2005.

[59] Håkan Hjalmarsson. From experiment design to closed-loop control. *Automatica*, 41(3):393 – 438, mar 2005. Data-Based Modelling and System Identification.

[60] Charles David Keeling and Timothy P Whorf. Atmospheric co2 concentrations derived from flask air samples at sites in the sio network. *Trends: a compendium of data on Global Change*, 2004.

[61] George Kimeldorf and Grace Wahba. Some results on tchebycheffian spline functions. *Journal of Mathematical Analysis and Applications*, 33(1):82 – 95, jan 1971.

[62] A.M. King, U.B. Desai, and R.E. Skelton. A generalized approach to q-markov covariance equivalent realizations for discrete systems. *Automatica*, 24(4):507 – 515, jul 1988.

[63] I. Kollár, R. Pintelon, and J. Schoukens. Frequency domain system identification toolbox for matlab: Improvements and new possibilities. *IFAC Proceedings Volumes*, 30(11):943 – 946, jul 1997. IFAC Symposium on System Identification (SYSID'97), Kitakyushu, Fukuoka, Japan, 8-11 July 1997.

[64] Boris P. Kovatchev, Eric Renard, Claudio Cobelli, Howard C. Zisser, Patrick Keith-Hynes, Stacey M. Anderson, Sue A. Brown, Daniel R. Chernavvsky, Marc D. Breton, Anne Farret, Marie-Josée Pelletier, Jérôme Place, Daniela Bruttomesso, Simone Del Favero, Roberto Visentin, Alessio Filippi, Rachele Scotton, Angelo Avogaro, and Francis J. Doyle. Feasibility of outpatient fully integrated closed-loop control: First studies of wearable artificial pancreas. *Diabetes Care*, 36(7):1851–1858, jun 2013.

[65] Ming-Jun Lai. On sparse solutions of underdetermined linear systems. *Journal of Concrete and Applicable Mathematics*, 8(2):296–327, 2010.

[66] Ming-Jun Lai and Paul Wenston. L1 spline methods for scattered data interpolation and approximation. *Advances in Computational Mathematics*, 21(3):293–315, Oct 2004.

[67] James Lam. Model reduction of delay systems using pade approximants. *International Journal of Control*, 57(2):377–391, February 1993.

[68] John Lataire, Rik Pintelon, Dario Piga, and Roland Tóth. Continuous-time linear time-varying system identification with a frequency-domain kernel-based estimator. *IET Control Theory & Applications*, 11(4):457–465, February 2017.

[69] K. Liu. Identification of linear time-varying systems. *Journal of Sound and Vibration*, 206(4):487 – 505, oct 1997.

[70] L. Ljung. Initialisation aspects for subspace and output-error identification methods. In *2003 European Control Conference (ECC)*, pages 773–778. IEEE, Sep. 2003.

[71] Lennart Ljung. *System identification toolbox: User's guide.* Citeseer, 1995.

[72] Lennart Ljung. *System Identification: Theory for the User (2nd Edition).* Prentice Hall, 1999.

[73] J. Löfberg. Yalmip : A toolbox for modeling and optimization in matlab. In *In Proceedings of the CACSD Conference*, Taipei, Taiwan, 2004.

[74] O. Lopez, M. Pârlog, B. Borderie, M.F. Rivet, G. Lehaut, G. Tabacaru, L. Tassangot, P. Pawłowski, E. Bonnet, R. Bougault, A. Chbihi, D. Dell'Aquila, J.D. Frankland, E. Galichet, D. Gruyer, M. La Commara, N. Le Neindre, I. Lombardo, L. Manduci, P. Marini, J.C. Steckmeyer, G. Verde, E. Vient, and J.P. Wieleczko. Improving isotopic identification with INDRA silicon–CsI(tl) telescopes. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 884:140 – 149, mar 2018.

[75] Mihaela T. Matache and Valentin Matache. Hilbert spaces induced by toeplitz covariance kernels. In Bozenna Pasik-Duncan, editor, *Stochastic Theory and Control*, pages 319–333, Berlin, Heidelberg, 2002. Springer Berlin Heidelberg.

[76] Gonzalo Mateos, Santiago Segarra, Antonio G. Marques, and Alejandro Ribeiro. Connecting the dots: Identifying network structure via graph signal processing. *IEEE Signal Processing Magazine*, 36(3):16–43, May 2019.

[77] M. Mazzoleni, M. Scandella, S. Formentin, and F. Previdi. Classification of light charged particles via learning-based system identification. In *2018 IEEE Conference on Decision and Control (CDC)*, pages 6053–6058. IEEE, Dec 2018.

[78] Mirko Mazzoleni, Simone Formentin, Matteo Scandella, and Fabio Previdi. Semi-supervised learning of dynamical systems: a preliminary study. jun 2018. 17th IEEE European Control Conference (ECC), Lymassol, Cyprus.

[79] Mirko Mazzoleni, Matteo Scandella, Simone Formentin, and Fabio Previdi. Identification of nonlinear dynamical system with synthetic data: a preliminary investigation. *IFAC-PapersOnLine*, 51(15):622–627, 2018. 18th IFAC Symposium on System Identification (SYSID), Stockholm, Sweden.

[80] P.M. Mäkilä. LTI modelling of NFIR systems: near-linearity and control, LS estimation and linearization. *Automatica*, 41(1):29–41, jan 2005.

[81] Biqiang Mu, Tianshi Chen, and Lennart Ljung. Asymptotic properties of generalized cross validation estimators for regularized system identification. *IFAC-PapersOnLine*, 51(15):203 – 208, 2018. 18th IFAC Symposium on System Identification SYSID 2018.

[82] Biqiang Mu, Tianshi Chen, and Lennart Ljung. Asymptotic properties of hyperparameter estimators by using cross-validations for regularized system identification. In *2018 IEEE Conference on Decision and Control (CDC)*, pages 644–649. IEEE, Dec 2018.

[83] Biqiang Mu, Tianshi Chen, and Lennart Ljung. On asymptotic properties of hyperparameter estimators for kernel-based regularization methods. *Automatica*, 94:381 – 395, aug 2018.

[84] B. K. Natarajan. Sparse approximate solutions to linear systems. *SIAM Journal on Computing*, 24(2):227–234, apr 1995.

[85] Arnold Neumaier. Solving ill-conditioned and singular linear systems: A tutorial on regularization. *SIAM Review*, 40(3):636–666, jan 1998.

[86] G. De Nicolao and G.F. Trecate. Consistent identification of narx models via regularization networks. *IEEE Transactions on Automatic Control*, 44(11):2045–2049, Nov 1999.

[87] Henrik Ohlsson, Jacob Roll, and Lennart Ljung. Manifold-constrained regressors in system identification. In *2008 47th IEEE Conference on Decision and Control*, pages 1364–1369. IEEE, Dec 2008.

[88] Frank WJ Olver, Daniel W Lozier, Ronald F Boisvert, and Charles W Clark. *NIST Handbook of Mathematical Functions Hardback and CD-ROM*. Cambridge University Press, 2010.

[89] Peter Van Overschee and Bart De Moor. *Subspace Identification for Linear Systems*. Springer US, 1996.

[90] Henri Padé. Sur la représentation approchée d'une fonction par des fractions rationnelles. In *Annales scientifiques de l'École Normale Supérieure*, volume 9, pages 3–93, 1892.

[91] Valentin Pascu, Hugues Garnier, Lennart Ljung, and Alexandre Janot. Benchmark problems for continuous-time model identification: Design aspects, results and perspectives. *Automatica*, 107:511 – 517, sep 2019.

[92] Eduard Petlenkov, Sven Nomm, and Ulle Kotta. Neural networks based ANARX structure for identification and model based control. In *2006 9th International Conference on Control, Automation, Robotics and Vision*, pages 1–5. IEEE, Dec 2006.

[93] Dario Piga, Pepijn Cox, Roland Tóth, and Vincent Laurain. LPV system identification under noise corrupted scheduling and output signal observations. *Automatica*, 53:329 – 338, mar 2015.

[94] Gianluigi Pillonetto and Alessandro Chiuso. Tuning complexity in regularized kernel-based regression and linear system identification: The robustness of the marginal likelihood estimator. *Automatica*, 58:106 – 117, aug 2015.

[95] Gianluigi Pillonetto, Francesco Dinuzzo, Tianshi Chen, Giuseppe De Nicolao, and Lennart Ljung. Kernel methods in system identification, machine learning and function estimation: A survey. *Automatica*, 50(3):657–682, March 2014.

[96] Gianluigi Pillonetto and Giuseppe De Nicolao. A new kernel-based approach for linear system identification. *Automatica*, 46(1):81 – 93, jan 2010.

[97] Gianluigi Pillonetto, Minh Ha Quang, and Alessandro Chiuso. A new kernel-based approach for nonlinearsystem identification. *IEEE Transactions on Automatic Control*, 56(12):2825–2840, dec 2011.

[98] Rik Pintelon and Johan Schoukens. *System identification: a frequency domain approach.* John Wiley & Sons, mar 2012.

[99] L. Piroddi and W. Spinelli. An identification algorithm for polynomial NARX models based on simulation error minimization. *International Journal of Control*, 76(17):1767–1781, nov 2003.

[100] Luigi Piroddi. Simulation error minimisation methods for NARX model identification. *International Journal of Modelling, Identification and Control*, 3(4):392, 2008.

[101] T. Poggio and F. Girosi. Networks for approximation and learning. *Proceedings of the IEEE*, 78(9):1481–1497, Sep. 1990.

[102] Fabio Previdi, Sergio M Savaresi, Paolo Guazzoni, and Luisa Zetta. Detection and clustering of light charged particles via system-identification techniques. *International Journal of Adaptive Control and Signal Processing*, 21(5):375–390, 2007.

[103] S. Joe Qin. An overview of subspace identification. *Computers & Chemical Engineering*, 30(10-12):1502 – 1513, sep 2006. Papers form Chemical Process Control VII.

[104] Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning).* The MIT Press, 2005.

[105] Frigyes Riesz. *Sur une espèce de géométrie analytique des systèmes de fonctions sommables.* Gauthier-Villars, 1907.

[106] J. Rissanen. Modeling by shortest data description. *Automatica*, 14(5):465 – 471, sep 1978.

[107] Syed Z. Rizvi, Javad Mohammadpour, Roland Tóth, and Nader Meskin. A kernel-based approach to MIMO LPV state-space identification and application to a nonlinear process system. *IFAC-PapersOnLine*, 48(26):85–90, 2015. 1st IFAC Workshop on Linear Parameter Varying Systems LPVS 2015.

[108] Walter Rudin. *Real and Complex Analysis.* MCGRAW HILL BOOK CO, May 1986.

[109] Saburou Saitoh and Yoshihiro Sawano. *Theory of reproducing kernels and applications.* Springer, 2016.

[110] Sandro Salsa. *Partial Differential Equations in Action: From Modelling to Theory.* Springer, 2016.

[111] Bernhard Schölkopf and Alexander J. Smola. *Learning with kernels: support vector machines, regularization, optimization, and beyond.* The MIT Press, 2018.

[112] Gideon Schwarz. Estimating the dimension of a model. *The Annals of Statistics*, 6(2):461–464, mar 1978.

[113] Igor R. Shafarevich and Alexey O. Remizov. *Linear Algebra and Geometry*. Springer Berlin Heidelberg, 2013.

[114] D. I. Shuman, S. K. Narang, P. Frossard, A. Ortega, and P. Vandergheynst. The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains. *IEEE Signal Processing Magazine*, 30(3):83–98, May 2013.

[115] David I. Shuman, Sunil K. Narang, Pascal Frossard, Antonio Ortega, and Pierre Vandergheynst. Signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular data domains. *IEEE Signal Processing Magazine*, 30(3):83–98, may 2013.

[116] W. Skulski and M. Momayezi. Particle identification in csi(tl) using digital pulse shape analysis. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 458(3):759 – 771, feb 2001.

[117] Torsten Soderstrom and Petre Stoica. *System Identification*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1989.

[118] Charles M. Stein. Estimation of the mean of a multivariate normal distribution. *The Annals of Statistics*, 9(6):1135–1151, 11 1981.

[119] I. Steinwart, D. Hush, and C. Scovel. An explicit description of the reproducing kernel hilbert spaces of gaussian rbf kernels. *IEEE Transactions on Information Theory*, 52(10):4635–4643, Oct 2006.

[120] R. S. Storey, W. Jack, and A. Ward. The fluorescent decay of CsI(tl) for particles of different ionization density. *Proceedings of the Physical Society,*, 72(1):1–8, jul 1958.

[121] S. Sundararajan and S. Sathiya Keerthi. Predictive approaches for choosing hyperparameters in gaussian processes. In S. A. Solla, T. K. Leen, and K. Müller, editors, *Advances in Neural Information Processing Systems 12*, volume 13, pages 631–637. MIT Press - Journals, may 1999.

[122] V.N. Temlyakov. Weak greedy algorithms[*]this research was supported by national science foundation grant dms 9970326 and by onr grant n00014-96-1-1003. *Advances in Computational Mathematics*, 12(2):213–227, Feb 2000.

[123] Roland Tóth, Peter S.C. Heuberger, and Paul M.J. Van den Hof. Asymptotically optimal orthonormal basis functions for LPV system identification. *Automatica*, 45(6):1359 – 1370, jun 2009.

[124] J.A. Tropp. Greed is good: Algorithmic results for sparse approximation. *IEEE Transactions on Information Theory*, 50(10):2231–2242, Oct 2004.

[125] Vladimir N. Vapnik. *Statistical Learning Theory*. JOHN WILEY & SONS INC, September 1998.

[126] A. Varga. Balancing free square-root algorithm for computing singular perturbation approximations. In *[1991] Proceedings of the 30th IEEE Conference on Decision and Control*, pages 1062–1065 vol.2. IEEE, December 1991.

[127] Vincent Verdult and Michel Verhaegen. Kernel methods for subspace identification of multivariable LPV and bilinear systems. *Automatica*, 41(9):1557 – 1565, sep 2005.

[128] Michel Verhaegen and Vincent Verdult. *Filtering and System Identification.* Cambridge University Press, November 2011.

[129] Régis Vert and Jean-Philippe Vert. Consistency and convergence rates of one-class svms and related algorithms. *Journal of Machine Learning Research*, 7(May):817–854, 2006.

[130] Ernesto De Vito, Lorenzo Rosasco, Andrea Caponnetto, Michele Piana, and Alessandro Verri. Some properties of regularized kernel methods. *Journal of Machine Learning Research*, 5(Oct):1363–1390, 2004.

[131] Grace Wahba. *Spline Models for Observational Data.* Society for Industrial and Applied Mathematics, jan 1990.

[132] X. Wang. NFIR nonlinear filter. *IEEE Transactions on Signal Processing*, 39(7):1705–1708, July 1991.

[133] H. L. Wei, S. A. Billings, and M. A. Balikhin. Wavelet based non-parametric NARX models for nonlinear input–output system identification. *International Journal of Systems Science*, 37(15):1089–1096, dec 2006.

[134] A. E. Yagle. A fast algorithm for toeplitz-block-toeplitz linear systems. In *2001 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings (Cat. No.01CH37221)*, volume 3, pages 1929–1932 vol.3. IEEE, May 2001.

[135] Jing Yang, Hua-Liang Wei, Visakan Kadirkamanathan, and Xiaofang Lin. System identification from small data sets using an output jittering method with application to model estimation of bioethanol production. In *2012 International Conference on Machine Learning and Cybernetics*, volume 3, pages 949–955. IEEE, July 2012.

[136] Peter Young and Anthony Jakeman. Refined instrumental variable methods of recursive time-series analysis part III. extensions. *International Journal of Control*, 31(4):741–764, apr 1980.

[137] Peter C. Young. *Recursive Estimation and Time-Series Analysis.* Springer Berlin Heidelberg, 2011.

[138] Huaiyu Zhu, Christopher KI Williams, Richard Rohwer, and Michal Morciniec. Gaussian regression and optimal finite dimensional linear models. 1997.