Machine Learning Methods for Robust Uncertainty Quantification and Controller Approximation

THIS IS A TEMPORARY TITLE PAGE
It will be replaced for the final print by a version
provided by the registrar's office.

Thèse présentée le 14 février 2023
à la Faculté des Sciences et Techniques de l'Ingénieur
programme doctoral en Génie électrique
École polytechnique fédérale de Lausanne

pour l'obtention du grade de Docteur ès Sciences
par

Emilio Maddalena

acceptée sur proposition du jury :

Prof. Giancarlo Ferrari-Trecate, président du jury
Prof. Colin Neil Jones, directeur de thèse
Prof. Sandra Hirche, rapporteuse
Prof. Lorenzo Fagiano, rapporteur
Prof. Roy Smith, rapporteur

Lausanne, EPFL, 2023

Besides this'll be easy with the two of us.
We've got science on our side.
— Bonnibel Bubblegum

Acknowledgment

First and foremost, I acknowledge professor Colin Jones' fundamental contribution to the present thesis, which was made in two, both equally important, but distinct ways. Directly, through countless motivating discussions on different topics that would go on for hours, and an always-positive attitude towards work and research that helped me along my journey. Indirectly, through creating a wholesome group atmosphere where cooperation was encouraged over competition. Taking a step back, I am also deeply grateful to you for having invited me to join LA and allowing me to develop my work at EPFL.

I would like to thank professors Sandra Hirche from Technische Universität München, professor Lorenzo Fagiano from Politecnico di Milano, professor Roy Smith from Eidgenössische Technische Hochschule Zürich, and professor Giancarlo Ferrari-Trecate from École polytechnique fédérale de Lausanne, for having accepted composing my doctoral exam jury and for dedicating their time to reading my work and providing me with feedback.

The Laboratoire d'Automatique is a great place to do your PhD at. The group is diverse and amicable, people are intellectually bright, and the general atmosphere is on point. Here is a list of people that helped me during the course of my doctorate: Alessio, Baiwei, Clara, Harsh, Johannes, Johannes, Mustafa, Paul, Peter, Philippe, Pulkit, Roland, Sohail, Yingzhao, Yuning. You have contributed to making both my academic and my personal life richer and more fun.

The final group I need to acknowledge is my wife and my family. I have an immense respect for my parents who have risen me and my brothers ensuring the absorption of long-lasting core values, including fairness, empathy and hard work, through demonstrating them with their daily actions. *Muito obrigado*, Bruno, Erica, Fabio and Renato – I look up to you. Vivi has given me support, advice, company and love, aiding me in every single moment of the last years. Thank you, truly!

Bienne, February 14, 2023 Emilio Tanowe Maddalena

Abstract

This thesis is situated at the crossroads between machine learning and control engineering. Our contributions are both theoretical, through proposing a new uncertainty quantification methodology in a kernelized context; and experimental, through investigating the suitability of two machine learning techniques to integrate feedback loops in two challenging real-world control problems.

The first part of this document is dedicated to deterministic kernel methods. First, the formalism is presented along with some widespread techniques to craft surrogates for an unknown ground-truth based on samples. Next, standing assumptions are made on the ground-truth complexity and on the data noise, allowing for a novel robust uncertainty quantification (UQ) theory to be developed. By means of this UQ framework, hard out-of-sample bounds on the ground-truth values are computed through solving convex optimization problems. Closed-form outer approximations are also presented as a lightweight alternative to solving the mathematical programs. Several examples are given to illustrate how the control community could benefit from using this tool.

In the second part of the thesis, statistical models in the form of Gaussian processes (GPs) are considered. These are used to carry out a building temperature control task of a hospital surgery center during regular use. The engineering aspects of the problem are detailed, followed by data acquisition, the model training procedure, and the developed predictive control formulation. Experimental results over a four-day uninterrupted period are presented and discussed, showing a gain in economical performance while ensuring proper temperature regulation.

Lastly, a specialized neural network architecture is proposed to learn linear model predictive controllers (MPC) from state-input pairs. The network features a parametric quadratic program (pQP) as an implicit non-linearity and is used to reduce the storage footprint and online computational load of MPC. Two examples in the domain of power electronics are given to showcase the effectiveness of the proposed scheme. The second of them consists in enhancing the start-up response of a real step-down converter, deploying the learned control law on an 80 MHz microcontroller and performing the computations in under 30 microseconds.

Keywords: kernel learning, Gaussian processes, model predictive control, neural networks, power electronics, building control.

Résumé

Cette thèse se situe à la croisée de l'apprentissage automatique et de l'ingénierie des systèmes de contrôle. Nos contributions sont théoriques, en proposant une nouvelle méthodologie de quantification des incertitudes, ainsi que expérimentales, en étudiant l'adéquation de certaines techniques d'apprentissage automatique à intégrer des boucles de rétroaction dans le contexte de deux problèmes de contrôle réels et difficiles.

La première partie du document est consacrée aux méthodes noyau déterministes. Tout d'abord, le formalisme est présenté ainsi que certaines techniques pour créer des modèles pour une fonction inconnue à partir d'échantillons. Ensuite, des hypothèses sont faites sur la complexité de la fonction inconnue et sur le bruit affectant les données, ce qui permet de développer une nouvelle théorie robuste de quantification d'incertitude. Grâce à cette procédure de quantification de l'incertitude, des limites supérieures et inférieures hors échantillon sur les valeurs de la fonction peuvent être calculées en résolvant des problèmes d'optimisation convexes. Des approximations extérieures sous forme fermée sont présentées comme une alternative à la résolution des problèmes d'optimisation. Plusieurs exemples sont donnés pour illustrer comment la communauté du contrôle pourrait se bénéficier de l'utilisation de cet outil.

Dans la deuxième partie de la thèse, nous utilisons des modèles statistiques sous forme de processus Gaussiens sont considérés. Ceux-ci sont employés pour réaliser une tâche de contrôle de la température d'un centre de chirurgie hospitalier étant régulièrement utiliser. Les aspects techniques du problème sont détaillés, suivis de l'acquisition des données, de la procédure d'apprentissage du modèle et de la formulation de la commande prédictive développée. Les résultats expérimentaux sur une période ininterrompue de quatre jours sont présentés et discutés, montrant un gain de performance économique tout en assurant une régulation adéquate de la température.

Le dernier chapitre présente une architecture de réseau neuronal spécialisée pour apprendre des contrôleurs prédictifs basés sur des modèles linéaires à partir de paires état-commande. Le réseau dispose d'un programme quadratique paramétrique comme couche non-linéaire implicite et est utilisé pour réduire le besoin de stockage et la charge computationnelle en-ligne des contrôleurs prédictifs.

Résumé

Deux exemples dans le domaine de l'électronique de puissance sont donnés pour démontrer l'efficacité du schéma proposé. Le deuxième exemple consiste dans l'amélioration de la réponse du démarrage d'un convertisseur abaisseur réel, où la loi de commande a été déployer dans un microcontrôleur de 80 MHz et les calculs ont été effectuer en moins de 30 microsecondes.

Mots-clés : méthodes à noyaux, processus gaussien, commande prédictive, réseaux neuronaux, électronique de puissance, contrôle des bâtiments.

Contents

List of Figures

List of Tables

Acronyms

AHU air-handling unit

AI artificial intelligence

GP Gaussian processes

GTS ground-truth space

HS hypothesis space

HVAC heating, ventilation and air-conditioning

ML machine learning

MPC Model Predictive Control

NN neural network

PD positive-definite

pQP parametric quadratic programming

PWA piece-wise affine

RKHS reproducing kernel Hilbert space

UQ uncertainty quantification

1 Introduction

1.1 The broad context

Artificial intelligence (AI) and machine learning (ML) are on the rise. Both terms, often used interchangeably, are nowadays an integral part of the popular culture, being referenced in various shows, movies, and books written by laymen. Recent AI milestones that endorse the idea that important progress is being made include the 2016 AlphaGo's victory agains a 9-dan Go player (Silver et al., 2016), and the OpenAI generative models DALL·E and ChatGPT, which showcased impressive performance and drew the attention of the general population. On the business-side of the spectrum, companies are also trying to leverage the power of AI and ML to automate tasks, optimize processes and enhance productivity (Chui et al., 2022). Indeed, both AI and ML are often referred to by consulting firms as disruptive technologies whose potential is yet to be fully explored (Bechtel, 2022). These advances and excitement are generally associated with two main driving factors: the availability of tailored algorithms to solve specialized tasks and a wealth of informative data at one's disposal. Unfortunately, these are not equally present in all application domains.

Data is indeed abundant and even freely accessible when one needs to build an image classification model or a natural language processing algorithm. According to 2019 YouTube statistics, over 500 hours of videos are uploaded to the platform every minute (Hale, 2019). In the domain of automatic control, data is not as pervasive for the reasons highlighted next. First, control systems platforms, especially low-level ones, are typically not designed to store historical information and repeatedly replace old samples with more recent ones. Picture for instance microcontroller and FPGA boards, which only feature small memory blocks and are not always designed to export operational data to remote computers in real-

time. Additionally, whereas in fields such as games (e.g. Go) the *context* that dictates how the environment evolves in time is well-understood and sometimes completely recorded, in engineering systems it is most often not completely known and certainly not fully measured. In effect, robust control tools usually do not require access to disturbance values, or are fed estimates (Zhou and Doyle, 1998; Pannocchia, 2015). To point out one last difficulty, the tasks that need to be learned in controls are rather specific and dependent on the particular instance of the problem at hand. For example, an ML developer can teach a neural network how to tell cars from other objects while using pictures of a Volkswagen Kombi and a Lamborghini Aventador, but an automotive engineer would have a hard time teaching his model how to drive if data coming from both vehicles were mixed[1]. For these reasons, it is often hard for control engineers to gather large-enough batches of high-quality, informative data.

Aside from the issue of data availability, some classical algorithms are not suited for automatic control. Many ML application domains only involve off-line decision making with no stringent time constraints associated to them. The decisions made also do not normally affect the next inputs that the algorithm will receive, i.e., no closed-loop interactions are present (the prominent exception is reinforcement learning). ML algorithms are consequently devised without particular consideration for those aspects. Researchers have been trying to adapt certain models, making them more suited to describing and controlling engineering systems. One such line of investigation deals with incorporating physical knowledge into classical models (Galimberti et al., 2021; Di Natale et al., 2022b) with the hopes of attaining a more predictable behavior and, above all, system-level guarantees. Safety is another major concern (Hewing et al., 2020; Brunke et al., 2022). Since bad actuation frequently leads to causing materialistic or financial damage, if not human in the worst case, algorithms have to be predictable and comply with certain rules. In statistical learning, uncertainties are quantified and can be later used to establish safety guarantees as shown for example in Hewing et al. (2019) and Lederer et al. (2022). Nevertheless, these typically come in probabilistic forms, in contrast with hard properties that are independent of the uncertainty realizations and to which some control practitioners are more used to. Algorithms for which robust guarantees can be derived are way less common than probabilistic ones. Examples include Milanese and Novara (2004); Sabug Jr et al. (2021), where only mild continuity assumptions are posed on the phenomenon to be learned.

Success stories involving ML in control science of course exist and they can be

[1] In transfer learning (Pan and Yang, 2010), one identifies patterns in data coming from a certain task (driving the Kombi) to help him carry out a similar, but not identical task (driving the Aventador).

found in the fields of industrial mechatronics (Khosravi et al., 2022), building climate control (Lian et al., 2021) and autonomous racing (Hewing et al., 2019) to list a few. This thesis aims at strengthening this body of literature, corroborating the idea that ML can lend itself to controls and bring important value to it. We contribute by proposing a new algorithm that can be utilized for data-driven robust analysis and control, an experimental investigation of a promising ML technique, and a novel neural network architecture that can be used to scale-down the computational requirements of a well-known optimization-based control law.

1.2 Outline and contributions

The core of this dissertation is divided into three chapters, each with its own conclusions and envisioned future investigations. A brief description and the contributions made in each of them are outlined as follows.

Chapter 2: Safely learning with kernels.

In this part, we investigate the problem of robust uncertainty quantification, where *hard bounds* are be established for the values of an unknown function at unobserved locations. As opposed to other works found in the literature, our novel approach explores kernels, conferring on it a high degree of representation power. Another distinguishing feature is the presence of a bounded measurement noise model with no distributional assumptions imposed on it. Different versions of the bounds are presented, involving either the solution of convex optimization problems or closed-form expressions. Finally, examples are presented to illustrate their applicability in a number of different scenarios, including robust analysis and control problems. The contents of this part are based on the following works:

- P. Scharnhorst, E. T. Maddalena, Y. Jiang, and C. N. Jones. "Robust uncertainty bounds in reproducing kernel Hilbert spaces: A convex optimization approach." IEEE Transactions on Automatic Control – Early Access (2022).

- E. T. Maddalena, P. Scharnhorst, and C. N. Jones. "Deterministic error bounds for kernel-based learning techniques under bounded noise." Automatica 134 (2021): 109896.

- E. T. Maddalena, P. Scharnhorst, Y. Jiang, and C. N. Jones. "KPC: Learning-based model predictive control with deterministic guarantees." Learning for Dynamics and Control. PMLR, 2021.

Chapter 3: Building temperature control through Gaussian process and model predictive control.

In this chapter, an experimental investigation involving Gaussian processes (GP) dynamical models and Model Predictive Control (MPC) is reported. The techniques were combined to tackle the control of an industrial hospital cooling system for three adjacent rooms. To the best of our knowledge, no other similar investigation exists in the literature at present, validating the use of GP models in a multi-zone building control problem. Aside from detailing the developed project, we also contrast the approach to alternative methodologies with the aid of simulations, helping us understand how close each of them are to an ideal solution. The papers listed next are the ones more closely related to this part:

- Di Natale, L., Lian, Y., Maddalena, E. T., Shi, J., and Jones, C. N. (2022). "Lessons learned from data-driven building control experiments: Contrasting Gaussian process-based MPC, bilevel DeePC, and deep reinforcement learning". Conference on Decision and Control (pp. 1111-1117).

- Maddalena, E. T., Müller, S. A., dos Santos, R. M., Salzmann, C., Jones, C. N. (2022). "Experimental data-driven model predictive control of a hospital HVAC system during regular use" Energy and Buildings: 112316.

- Maddalena, E. T., Lian, Y., Jones, C. N. (2020). "Data-driven methods for building control—A review and promising future directions." Control Engineering Practice 95: 104211.

Chapter 4: Learing MPC controllers with pQP neural networks.

Instead of employing neural networks (NNs) to learn unknown relationships from data, in this chapter we utilize them to approximate a function that in principle can be computed in closed-form. The intention is that of attaining a simplified representation that can be more easily evaluated in real-time. More concretely, we propose a network architecture to learn MPC controllers from state-control samples that has two main advantages over competing strategies: it is shown to be capable of representing any linear MPC formulation, and the NN can be converted into a piece-wise affine format, similar to explicit MPC. Two examples, one in simulations and one experimental, are given to showcase the effectiveness of the technique in reducing the MPC computational burden with little impact on performance. The papers below are the texts from which most of the material was extracted:

- Maddalena, E. T., Moraes, C. G. da S., Waltrich, G., Jones, C. N. (2020). "A neural network architecture to learn explicit MPC controllers from data". IFAC-PapersOnLine 53 (2), 11362-11367.
- Maddalena, E. T., Specq, M. W. F., Wisniewski, V. L., Jones, C. N. (2021). "Embedded PWM predictive control of DC-DC power converters via piecewise-affine neural networks." IEEE Open Journal of the Industrial Electronics Society 2, 199-206.

A number of additional papers were written during the course of this PhD, but are not discussed in this thesis due to their being off-topic:

- Chalet, F.-X., Bujaroska, T., Germeni, E., Ghandri, N., Maddalena, E. T., Modi, K., Olopoenia, A., Thompson, J., Togninalli, M., Briggs, A. H. (2023). "Mapping the insomnia severity index instrument to EQ-5D health state utilities: A United Kingdom Perspective." PharmacoEconomics - Open.
- Rosolia, U., Lian, Y., Maddalena, E. T., Ferrari-Trecate, G., Jones, C. N. (2022). "On the optimality and convergence properties of the iterative learning model predictive controller." IEEE Transactions on Automatic Control 68.1: 556-563.
- Xu, W., Jiang, Y., Maddalena, E. T., Jones, C. N. (2022). "Lower bounds on the worst-case complexity of efficient global optimization." arXiv preprint arXiv:2209.09655.
- Chakrabarty, A., Maddalena, E. T., Qiao, H., Laughman, C. (2021). "Scalable bayesian optimization for model calibration: Case study on coupled building and HVAC dynamics." Energy and Buildings 253, 111460
- Maddalena, E. T., Jones, C. N. (2020). "NSM converges to a k-NN regressor under loose Lipschitz estimates." IEEE Control Systems Letters 134: 880-885.

Whether their are covered in this dissertation or not, all works developed throughout the past years were the result of fruitful collaborations with different scientists. Any merit or credit for the contributions made is therefore to be shared among authors and co-authors.

2 Safely learning with kernels

2.1 Introduction

At its core, learning refers to the process of gathering information and using it to improve one's knowledge about the phenomenon under study, which we will call the ground-truth. The standing assumption is that a link is in place, tying information and the ground-truth together even if such link is partially corrupted. Information typically comes in the form of data, samples, sometimes referred to as examples, and the mathematical formalism often used in modern machine learning to study the link between examples and the ground-truth is statistics. This choice is convenient because it can describe the possible non-determinism of outcomes through the concepts of distributions and samples; and because it provides us with plenty of tools to carry out learning, i.e., to gradually improve our knowledge about the underlying phenomenon. In this chapter, we will however adopt a different standpoint to study the problem of learning, relying on an *epistemic* description of uncertainty rather than on an aleatoric one (Hüllermeier and Waegeman, 2021).

When reporting physical parameters that are not exactly known, e.g. the friction coefficient between two surfaces or the leakage inductance of a transformer, it is common for engineers to specify ranges that are known to contain the *true* coefficient values. Assuming that a true, invariable value exists although it is not exactly known is the epistemic way of modeling uncertainty. This approach is customary in many applied sciences and can be generalized from simple scalar parameters to maps in function spaces. The entire field of approximation theory, from which we will borrow many tools, makes constant use of this framework to derive best approximations and error bounds for their nominal models (Wendland, 2004; Iske, 2018). Let us refer to this space as the ground-truth space (GTS), whereas the space from where our models are drawn will be called the hypothesis

space (HS). An advantage of working with functions spaces is the ease of packing information into them that we might have as domain experts: take for instance the Lipschitz continuity explored in Milanese and Novara (2004); Sabug Jr et al. (2021) or the weak derivatives assumptions in Sobolev GTSs by the authors of Novara et al. (2022). In this chapter, rather than posing assumptions directly on the ground-truth, we will make use of an indirect formalism that ties the GTS to maps named *kernels*. This approach guarantees the existence of a rich geometrical structure that will allow us to develop our theory.

Kernels are a fundamental building block of modern machine learning (Schölkopf and Smola, 2002). They enable computing similarities in extended feature spaces while dispensing with the need of lifting the data onto them. Alternatively, they can be viewed as a non-linearity library used to empower certain linear algorithms, the canonical example being support vector machines. If kernels are used to describe simultaneously the HS and the GTS, then optimal models can be found and error-bounds can be derived for them at possibly out-of sample locations, thus guaranteeing the quality of the model predictions (Schaback and Wendland, 2006; Kanagawa et al., 2018). One interesting aspect of those bounds are their determinism, in that the ground-truth values cannot lie outside the established "prediction envelope". Therefore, such methods could in principle be used in the engineering sciences to deterministically certify the performance or safety of algorithms that employ learned maps, which is also remarkable due to their not being asymptotic, but finite-sample guarantees. There is however one limiting factor that hinders its applicability, namely the absence of a noise model (Wendland, 2004, §11). Even though noise-free samples are to be expected in domains such as computer graphics and computer simulations, where these techniques are usually applied (see e.g. (Sarra, 2005; Nikan et al., 2022)), corrupted data is prevalent in electronics, robotics, building automation, among other fields.

In this chapter, we will explore kernel learning with hard guarantees. Our contributions, which are mainly reported in Section 2.4, consist in posing and solving an uncertainty quantification (UQ) problem to compute envelopes that bound the unknown ground-truth values. We highlight that the UQ problem does not require defining a nominal model, being thus agnostic to its choice. As opposed to other deterministic bounds found in the literature (see the classical book (Wendland, 2004, §11)), ours encompass an additive noise with bounded levels, a description that is not popular in general machine learning, but certainly relevant in domains such as robust analysis and control. A number of examples are presented at the end, illustrating different use-cases for the proposed theory, followed by concluding remarks and ideas for future investigations.

2.2 The formalism of kernels

Our goal is to learn maps of the form $f : \mathcal{X} \to \mathbb{R}$ and, to achieve that end, we will make extensive use of auxiliary functions called kernels.

Definition 1. (Kernel) Given an arbitrary non-empty set $\mathcal{X}$, a kernel k is any symmetric function of the form

$$k : \mathcal{X} \times \mathcal{X} \to \mathbb{R} \tag{2.1}$$

Definition 2. (Kernel matrix) Let $X = \{x_1, \ldots, x_n\} \subset \mathcal{X}$ be a finite set of points. The $n \times n$ matrix K_{XX} with entries $[K_{XX}]_{ij} = k(x_i, x_j)$ is called the kernel matrix of k associated with X.

Definition 3. (Positive-definite kernel) A kernel function k is said to be positive-definite if for any finite subset of points $X \subset \mathcal{X}$, the kernel matrix satisfies $K_{XX} \succeq 0$. If, in particular, $K_{XX} \succ 0$, then the kernel is strictly positive-definite.

Remark 1. Aside from the last definition, there exist broader classes of kernel functions such as the *conditionally positive-definite* one (Schölkopf and Smola, 2002, §2.4)(Wendland, 2004, §8). In addition, one can also generalize the co-domain k to be the field of complex numbers $\mathbb{C}$.

Instances of positive-definite (PD) kernel functions with index set $\mathcal{X} = \mathbb{R}^n$ are the linear, the squared-exponential (also known as Gaussian), the exponential (equivalent to the Matern12), the polynominal and the cosine kernels respectively given by

$$k_{\text{lin}}(x, x') = x \cdot x' \tag{2.2}$$

$$k_{\text{se}}(x, x') = \exp\left(-\frac{\|x - x'\|^2}{2\ell}\right) \tag{2.3}$$

$$k_{\text{exp}}(x, x') = \exp\left(-\frac{\|x - x'\|}{2\ell}\right) \tag{2.4}$$

$$k_{\text{pol}}(x, x') = \left(\sigma^2(x \cdot x') + \gamma\right)^d \tag{2.5}$$

$$k_{\cos}(x, x') = \cos\left(2\pi \sum_i ([x]_i - [x']_i)/\ell\right) \tag{2.6}$$

where $\cdot$ and $\|\cdot\|$ denote the usual inner-product and 2-norm in $\mathbb{R}^n$, respectively; and the constants $\ell \in \mathbb{R}_{>0}$, $\sigma, \gamma \in \mathbb{R}$ are the so-called *hyperparameters*. Plots of these functions are presented in Figure 2.1, illustrating how diverse they can be. Additionally, the product of any two PD kernels, such as the one depicted in the figure, is also a PD kernel (Schölkopf and Smola, 2002).

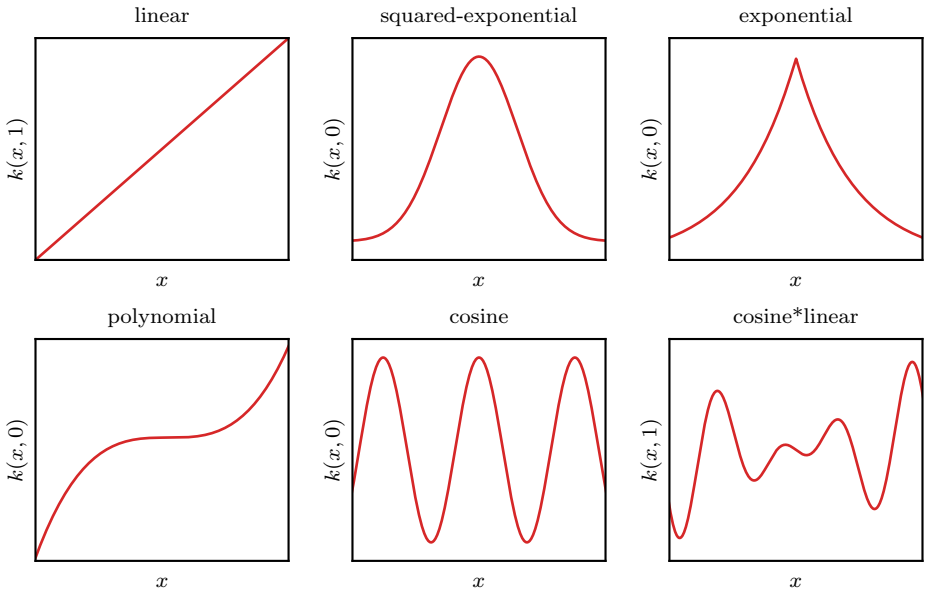

Figure 2.1: Examples of positive-definite kernels.

In order to construct surrogate models for the unknown f, one could partially evaluate a given k to match their domains and co-domains. In other words, fix one of its arguments so that $k(z,\cdot) : \mathcal{X} \to \mathbb{R}, x \mapsto k(z,x)$ for some $z \in \mathcal{X}$. Indeed, this was the approach taken to draw the plots in Figure 2.1. Approximating the unknown f with a single partially evaluated kernel however appears to be overly restrictive, limiting. A sensible next step would be to consider linear combinations of such kernel functions. It turns out that every PD kernel has a function space associated with it that contains these linear combinations and is endowed with plenty of useful geometric structure. The following concepts are presented to set up the stage for defining this special hypothesis space, the *reproducing kernel Hilbert space*(RKHS).

Proposition 1. (PD kernels have feature maps) Let $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ be a positive-definite kernel. Then there exists a Hilbert space $\mathbb{H}$ endowed with an inner product $\langle \cdot, \cdot \rangle_{\mathbb{H}}$ and a mapping $\Phi : \mathcal{X} \to \mathbb{H}$ such that

$$k(x, x') = \langle \Phi(x), \Phi(x') \rangle_{\mathbb{H}} \tag{2.7}$$

holds for any $x, x' \in \mathcal{X}$. Φ is known as a feature map from $\mathcal{X}$ to $\mathbb{H}$.

Proof. (Steinwart and Christmann, 2008, Theorem 4.16), modulo the nomenclature difference. □

Feature maps Φ are central in machine learning, allowing one to represent the data x he has in a more suitable format. At the same time, (2.7) unveils another aspect of a kernel evaluation $k(x, x')$: it returns the inner-product value of the transformed

inputs $\Phi(x)$, $\Phi(x')$. Educational textbooks often interpret these inner-products as a similarity measure between x and x' (Schölkopf and Smola, 2002).

The mappings Φ above as well as the $\mathbb{H}$ spaces are in general not unique (Steinwart and Christmann, 2008, §4). There is however one such $\mathbb{H}$ space that enjoys an extra property, ruling out some unexpected behavior from its members. This particular $\mathbb{H}$ is not an arbitrary Hilbert space, but a Hilbert space of functions.

Definition 4. (Reproducing kernel Hilbert space) Let $\mathcal{X} \neq \emptyset$ and $\mathbb{R}^{\mathcal{X}}$ the set of functions mapping $\mathcal{X}$ to $\mathbb{R}$. The subset $\mathcal{H} \subset \mathbb{R}^{\mathcal{X}}$ is called a reproducing kernel Hilbert space (RKHS) if it is a Hilbert space and if $\forall x \in \mathcal{X}$ the evaluation functionals

$$L_x : \mathcal{H} \to \mathbb{R},\ L_x(f) \mapsto f(x),\ \forall f \in \mathcal{H} \tag{2.8}$$

are bounded.

In order to see how useful such a property is, consider a sequence $\{f_n\}_{n=1}^{\infty}$ within a certain Hilbert function space $\mathbb{H} \subset \mathbb{R}^{\mathcal{X}}$. Intuitively, one would expect that if $f_n \to f^\star$ in $\mathbb{H}$, then the values $f_n(x)$ attained by the sequence would converge to the values $f^\star(x)$. Yet, this is not always the case (see Example 1 in Appendix A). If, on the other hand, the evaluation functionals are bounded as in Definition 4, then the connection between convergence in the function space and the pointwise convergence of functions is guaranteed. Indeed, if $\{f_n\}_{n=1}^{\infty}$ and $f^\star$ are members of an RKHS $\mathcal{H}$, then $|f_n(x) - f^\star(x)| = |L_x(f_n) - L_x(f^\star)| \leq \|L_x\|\, \|f_n - f^\star\|_{\mathcal{H}}$, where $\|L_x\|$ is the operator norm of L_x that is guaranteed by Definiton 4 to be a finite number. As a result, if $\|f_n - f^\star\|_{\mathcal{H}} \to 0$, the right-hand side of the inequality goes to zero, and so does the pointwise difference $|f_n(x) - f^\star(x)|$. Therefore, function convergence in an RKHS implies pointwise convergence, matching our intuition.

Proposition 2. (Every RKHS has a unique PD reproducing kernel) Let $\mathcal{H} \subset \mathbb{R}^{\mathcal{X}}$ be an RKHS. Then, the map $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, $k(x, x') := \langle L_x, L_{x'} \rangle_{\mathcal{H}}$ is a positive-definite kernel. Furthermore, k is the unique map to satisfy the reproducing property, i.e., for any $x \in \mathcal{X}$, $k(x, \cdot) \in \mathcal{H}$ and

$$\langle f, k(x, \cdot) \rangle_{\mathcal{H}} = L_x(f) = f(x),\ \forall f \in \mathcal{H} \tag{2.9}$$

Proof. (Berlinet and Thomas-Agnan, 2011, Lemma 2) along with (Steinwart and Christmann, 2008, Theorem 4.20). □

Proposition 3. (Every PD kernel has a unique RKHS) Let $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ be a PD kernel. If k is the reproducing kernel of an RKHS $\mathcal{H}_A$ and of another RKHS $\mathcal{H}_B$, then $\mathcal{H}_A = \mathcal{H}_B$.

Proof. (Steinwart and Christmann, 2008, Theorem 4.21). □

We understand from Propositions 2 and 3 that a special relationship exists between a kernel and its RKHS. Nevertheless, it is still unclear from Definition 4 alone how a given k influences or defines the members of $\mathcal{H}$. To shed light on the matter, it helps to explicitly construct $\mathcal{H}$ starting from a given k. Consider the so-called *pre-Hilbert space*

$$\mathcal{H}_0 := \operatorname{span}\{k(x,\cdot) \,|\, x \in \mathcal{X}\} \tag{2.10}$$

$$= \left\{ \sum_{i=1}^{n} c_i \, k(x_i,\cdot) \,|\, n \in \mathbb{N}, c_i \in \mathbb{R}, x_i \in \mathcal{X} \right\} \tag{2.11}$$

equipped with the real-valued map $\langle f, g \rangle_{\mathcal{H}_0} := \sum_{i=1}^{n} \sum_{j=1}^{m} a_i b_i k(x_i, x_j)$ for members $f, g \in \mathcal{H}_0, f = \sum_{i=1}^{n} a_i \, k(x_i,\cdot), g = \sum_{j=1}^{m} b_j \, k(x_j,\cdot)$, which can be shown to be a valid inner-product. This family $\mathcal{H}_0$ of functions is however not guaranteed to be complete, i.e., sequences $\{f_i\}_{i\in\mathbb{N}}$ of members might converge to functions outside $\mathcal{H}_0$. To transform it into a proper Hilbert space, one closes the space

$$\mathcal{H} := \operatorname{clos} \mathcal{H}_0 \tag{2.12}$$

thus encompassing all limit points[1]. Finally, the function space $\mathcal{H}$ defined in (2.12) can then be shown to be a valid RKHS[2] according to Definition 4. In fact, it is the *only* one associated with k. We therefore understand that the members of $\mathcal{H}$ are weighted sums of partially evaluated kernels as per (2.11) along with their limit points.

The questions of how expressive RKHSs can be still lingers on. To better examine the matter, consider the following measure of expressiveness.

Definition 5. (Universal kernel) Let k be a continuous PD kernel and the set $\mathcal{X}$ be a compact metric space. Then k is called universal if its RKHS $\mathcal{H} \subset \mathbb{R}^{\mathcal{X}}$ is dense in the space of real-valued continuous functions $C(\mathcal{X}) \subset \mathbb{R}^{\mathcal{X}}$ with respect to the maximum norm $\|\cdot\|_\infty$.

The above definition guarantees that for any target function $g \in C(\mathcal{X})$ and any tolerable error $\epsilon > 0$, there exists an f in the RKHS of a universal kernel such that

[1] Closing $\mathcal{H}_0$ requires defining an inner-product on the superset $\mathcal{H}$ that is consistent with the one present in the subset $\mathcal{H}_0$. Also, the closure of a set is always closed, which guarantees that sequences within $\mathcal{H}$ cannot converge to functions outside of it.

[2] For a proof of this statement, the reader is referred to (Berlinet and Thomas-Agnan, 2011, §3), or to (Sejdinovic and Gretton, 2012, §4) for a more step-by-step pedagogical exposition.

their mismatch is bounded $|f(x) - g(x)| \leq \epsilon, \forall x \in \mathcal{X}$. All in all, the universality property is an indication of how rich a hypothesis space is, thus reassuring the user that little bias error will be introduced by his choice.

Proposition 4. Let $\mathcal{X}$ be a compact subset of $\mathbb{R}^n$. The squared-exponential kernel with (2.3) with $\ell > 0$ is universal.

Proof. (Steinwart and Christmann, 2008, Corollary 4.58). □

Remark 2. The anisotropic squared exponential kernel, where the hyperparameter ℓ is not a constant, but a vector weighing each x dimension differently, is also universal. To see this, simply note that this generalized kernel function can recover any isotropic squared-exponential used to fulfill the universality condition.

In contrast with Proposition 4, some results on the restrictiveness of RKHSs can also be found in the literature. In Steinwart (2020) for example, the author shows that no RKHS can contain $C(X)$. On a looser note, some authors argue that the squared-exponential kernel (2.3) has an $\mathcal{H}$ that is too smooth when compared to alternative hypothesis spaces that are also associated with the same kernel (see the discussion in (Kanagawa et al., 2018, §4)). For a thorough exposition of universal kernels, the reader is referred to Micchelli et al. (2006). Lastly, we underline that other model classes in machine learning also enjoy the same universality properties that kernels do, notable certain architectures of deep neural networks (Kidger and Lyons, 2020).

To better visualize what an RKHS is and the types of functions we might find in it, the illustration in Figure 2.2 was created. First, we see on the top left corner the single kernel member of the exponential RKHS with norm 1. On the right part of the figure, we see more erratic maps displaying an aggressive behavior that is reflected in the high value of their norms. Finally, on the bottom right corner, a sinusoidal-like map is shown. The latter suggests that, even though its characteristics are fairly different from the standard exponential kernel, the map was not penalized with a high norm, but showed a reasonably low value.

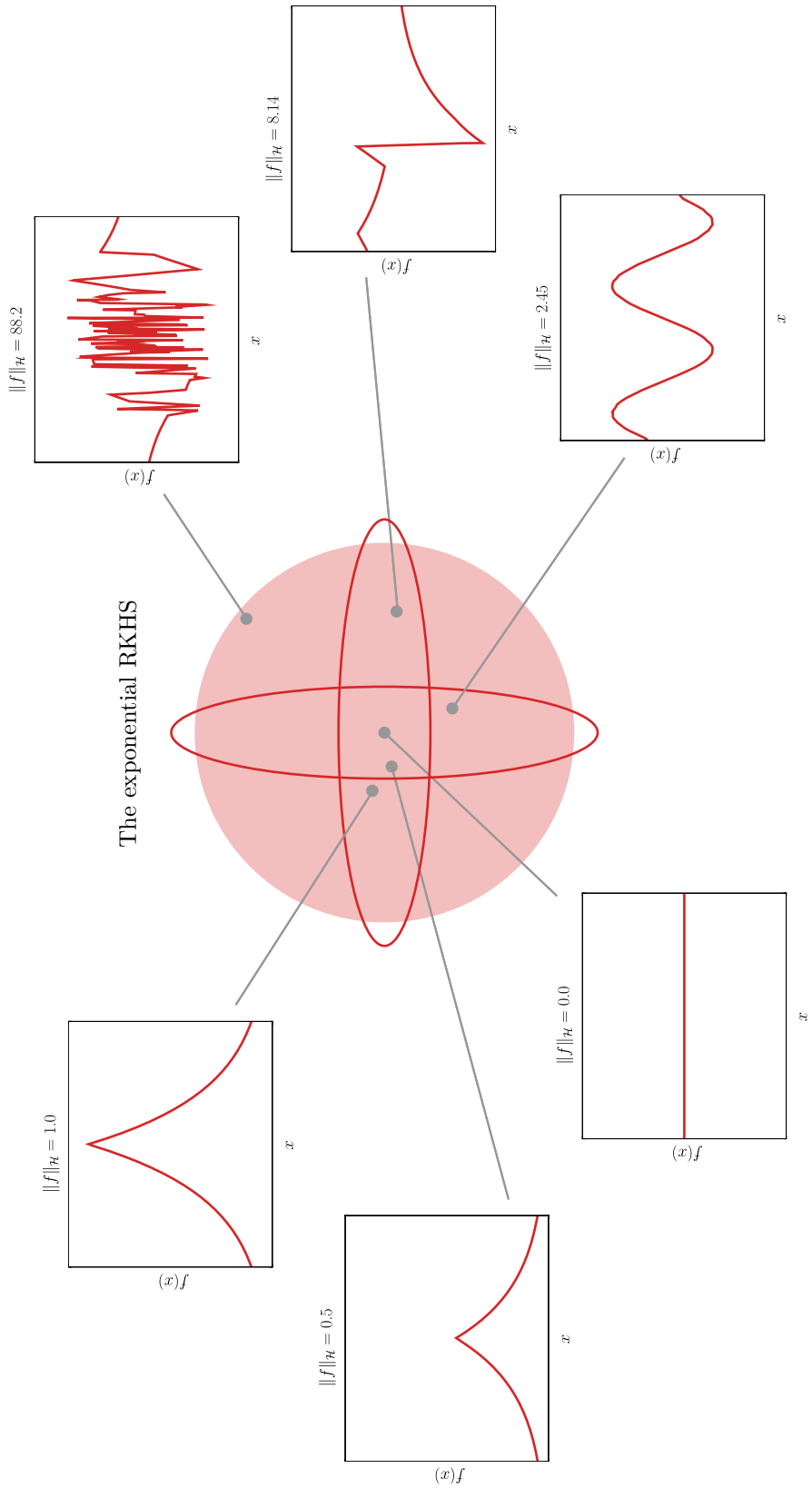

Figure 2.2: Members of the exponential RKHS and their respective norms.

2.3 Crafting models

Suppose a dataset of the form $\{(x_i, y_i)\}_{i=1}^n$ is given. The $x_i \in \mathcal{X}, \forall i$ elements are referred to as *inputs* and the $y_i \in \mathbb{R}, \forall i$ as *outputs*. In this section, we will discuss exclusively the case where $\mathcal{X} \subset \mathbb{R}^m$ is a compact set, and the outputs are real-valued. The values y_i are assumed to deliver information about an underlying ground-truth function $f^\star$ through the measurement model

$$y_i = f^\star(x_i) + \epsilon_i, \quad i = 1, \dots, n \tag{2.13}$$

As detailed in Section 2.2, weighted sums of partially evaluated kernel functions arise naturally in the context of kernel learning. In this section we shall see that, when given $\{(x_i, y_i)\}_{i=1}^n$, maps of the form

$$f(x) = \sum_{i=1}^n \alpha_i k(x_i, x) \tag{2.14}$$

are good candidates for acting as surrogate functions for the ground-truth $f^\star$. Indeed, they are known to solve a number of optimal fitting problems given appropriate weights α as we explain next.

In the absence of measurement noise, i.e., when $\epsilon_i = 0$, the outputs y_i perfectly represent $f^\star$. As a result, data interpolation can be a sensible task to carry out, which could done over $f \in \mathcal{H}$ while minimizing the resulting model norm.

Proposition 5. (Minimum-norm interpolation (MNI)) Let $\{(x_i, y_i)\}_{i=1}^n$ be a collection of points such that $x_i \in \mathbb{R}^m$ and $y_i \in \mathbb{R}$. Let k be a PD kernel and $\mathcal{H} \subset \mathbb{R}^{\mathcal{X}}$ its RKHS. Then, the variational problem

$$\bar{f} \in \arg\inf_{f \in \mathcal{H}} \left\{ \|f\|_{\mathcal{H}}^2 : f(x_i) = y_i,\ i = 1, \dots, n \right\} \tag{2.15}$$

admits the unique solution $\bar{f} \in \mathcal{H}$, $\bar{f}(x) = \sum_{i=1}^n \alpha_i k(x_i, x)$, $\alpha = y^\top K_{XX}^{-1}$.

Proof. (Kanagawa et al., 2018, Theorem 3.5). □

Remark 3. Some might think that models of the form (2.15) would perform poorly in real-world scenarios as overfitting goes against established machine learning guidelines. Yet, some authors have recently advocated for such models, and interpolants in general, stating that they possess strong generalization capabilities (see e.g. Belkin et al. (2018, 2019); Beaglehole et al. (2022)).

To tackle the approximation problem in the presence of measurement noise, a

compromise between fitting the data and rejecting uninformative fluctuations is sometimes desirable. One of the most standard tools used to achieve this balance is kernel ridge regression (KRR) as enunciated next.

Proposition 6. (Kernel ridge regression (KRR)) Let $\{(x_i, y_i)\}_{i=1}^n$ be a collection of points such that $x_i \in \mathcal{X}$ for a compact $\mathcal{X} \subset \mathbb{R}^m$ and $y_i \in \mathbb{R}$. Let k be a SPD kernel and $\mathcal{H} \subset \mathbb{R}^{\mathcal{X}}$ its RKHS. Then, the variational problem

$$\inf_{f \in \mathcal{H}} \frac{1}{n} \sum_{i=1}^{n} (y_i - f(x_i))^2 + \lambda \|f\|_{\mathcal{H}}^2 \tag{2.16}$$

with $\lambda > 0$ admits a single minimizer, the KRR model $f(x) = \sum_{i=1}^n \alpha_i x_i$ with $\alpha = (K_{XX} + n\lambda I)^{-1} y$.

Proof. Kimeldorf and Wahba (1971). □

To conclude this section, we will state a more general result, a theorem on which a large portion of kernel learning is grounded.

Theorem 1. (The representer theorem) Let $\{(x_i, y_i)\}_{i=1}^n$ be a collection of points such that $x_i \in \mathcal{X}$ for an arbitrary $\mathcal{X}$ and $y_i \in \mathbb{R}$. Let k be a PD kernel and $\mathcal{H} \subset \mathbb{R}^{\mathcal{X}}$ its RKHS. Consider an arbitrary function $c : (\mathcal{X} \times R^2)^n \to \mathbb{R} \cup \{\infty\}$ and a strictly monotonic increasing function $\Omega : [0, \infty) \to \mathbb{R}$. Then, if $f \in \mathcal{H}$ is a minimizer of the variational problem

$$\inf_{f \in \mathcal{H}} c((x_1, y_1, f(x_1)), \ldots, (x_n, y_n, f(x_n))) + \Omega(\|f\|_{\mathcal{H}}) \tag{2.17}$$

admits a representation of the form $f(x) = \sum_{i=1}^n \alpha_i x_i$, with $\alpha_i \in \mathbb{R}$.

Proof. Schölkopf et al. (2001). □

2.4 Quantifying uncertainty

Besides being able to craft surrogate functions for our unknown ground-truth, it is also a common desideratum to understand how far away our predictions can be from the real phenomenon. We will start this section by formalizing the problem of bounding the ground-truth values that can be attained even at unseen locations given the information at hand. It is important to highlight that this process will not require a model. Next, alternative bounds are developed, this time around nominal models such as the ones presented in Section 2.3. Rather than limiting ourselves to the theoretical sphere, the discussion will also touch on the computational aspects involved in evaluating the derived expressions.

2.4.1 The setting and problem definition

The theory developed in this section will revolve around a specific class of kernels and input spaces, and will be built on the following standing assumptions.

Assumption 1. k is strictly positive-definite and its index set $\mathcal{X} \subset \mathbb{R}^m$ is compact.

Assumption 2. $f^\star$ is contained in the RKHS $\mathcal{H}$ associated with k.

The available data $\{(x_i, \mathsf{y}_i)\}_{i=1}^d$ is such that $x_i \in \mathcal{X}$ and $\mathsf{y}_i \in \mathbb{R}^{n_i}$, where the vector y_i stacks n_i scalar outputs $y_{i,1}, \dots, y_{i,n_i}$ observed at the same input location x_i. The inputs x_i are assumed to be pairwise-distinct without loss of generality. The outputs are assumed to carry information about an underlying unknown ground-truth map $f^\star$ according to

$$y_{i,j} = f^\star(x_i) + \delta_{i,j} \tag{2.18}$$

where $\delta_{i,j}$ denotes an additive measurement noise. If only a single output is present at each input location, the observational model (2.18) simplifies to $y_i = f^\star(x_i) + \delta_i$. For brevity, let X denote the set of all inputs $x_1, \dots, x_d$ in the dataset. As for the nature of $\delta_{i,j}$, no specific distributional assumptions are made, but only that its magnitude is uniformly bounded by a known scalar.

Assumption 3. $\delta_{i,j}$ is bounded by a known scalar $\bar{\delta}$, i.e., $|\delta_{i,j}| \leq \bar{\delta}, \forall i, j$.

At this point one could ask himself if any out-of-sample guarantees could be already established on the values attained by $f^\star$. The answer is no. Indeed, for any tentative upper bound $\omega \in \mathbb{R}$ at $x \notin \{x_1, \dots, x_d\}$, regardless of the number of samples d, there is a member $f \in \mathcal{H}$ capable of reproducing any values at the x_i locations and additionally violating ω by an arbitrary level. Lower bounds could be equally violated as well. What we lack is a complexity bound, which will be posed by restricting $f^\star$ to lie within the Γ-ball of $\mathcal{H}$.

Assumption 4. An upper-bound $\Gamma \geq \|f^\star\|_{\mathcal{H}}$ is known.

Remark 4. The matter of exactly computing RKHS norms from weights and otherwise estimating them from data is discussed in Appendix 2.8.2.

With all assumptions in place, we can formulate the variational problem $\mathbb{P}0$ below, with the query point $x \in \mathcal{X}$ as a parameter

$$\boxed{\mathrm{F}(x) = \sup_{f \in \mathcal{H}} \{f(x) : \|f\|_{\mathcal{H}} \leq \Gamma, \|f_X - \mathsf{y}\|_\infty \leq \bar{\delta}\}} \tag{2.19}$$

where f_X is the vector of evaluations at the input locations, which are repeated whenever multiple outputs are available at a given input (see Appendix 2.8.3). We highlight that the supremum is guaranteed to be a finite number. To see this, notice that $|f(x)| = |\langle f, k(x,\cdot)\rangle_{\mathcal{H}}| \leq \|f\|_{\mathcal{H}} \|k(x,\cdot)\|_{\mathcal{H}} \leq \Gamma \sqrt{k(x,x)}$. This last bound is rather loose as it does not exploit any information present in X nor the outputs y, and is moreover uniform for translation-invariant kernels such as the squared-exponential. (2.19) on the other hand makes use the dataset $\{(x_i, \mathsf{y}_i)\}_{i=1}^d$ in its entirety as well as the complexity bound Γ.

2.4.2 The optimal solution

Consider now the convex parametric quadratically-constrained linear program $\mathbb{P}1$

$$\boxed{\begin{aligned} \mathrm{C}(x) \;=\; \max_{c\in\mathbb{R}^d, c_x\in\mathbb{R}} \quad & c_x && (2.20)\\ \text{subj. to} \quad & \begin{bmatrix} c \\ c_x \end{bmatrix}^\top \begin{bmatrix} K_{XX} & K_{Xx} \\ K_{xX} & k(x,x) \end{bmatrix}^{-1} \begin{bmatrix} c \\ c_x \end{bmatrix} \leq \Gamma^2 && (2.21)\\ & \|\Lambda c - \mathsf{y}\|_\infty \leq \bar{\delta} && (2.22) \end{aligned}}$$

for any $x \in \mathcal{X}\backslash X$, and extend its value function $C(x)$ to points $x = x_i \in X$ with the solution of $\mathbb{P}1' : \mathrm{C}(x) = \max_{c\in\mathbb{R}^d}\{c_i \,|\, c^\top K_{XX}^{-1} c \leq \Gamma^2,\, \|\Lambda\, c - \mathsf{y}\|_\infty \leq \bar{\delta}\}$. The two cases $\mathbb{P}1$ and $\mathbb{P}1'$ are distinguished due to the matrix in (2.21) becoming singular for any $x \in X$, and since it allows for one decision variable to be eliminated. The connection between this optimization problem and (2.19) is unveiled next.

Theorem 2. (Finite-dimensional equivalence): The objective in $\mathbb{P}0$ attains its supremum in $\mathcal{H}$ and $\mathrm{F}(x) = \mathrm{C}(x)$ for any $x \in \mathcal{X}$.

The proof involves showing that a solution to $\mathbb{P}0$ necessarily lies in a finite-dimensional subspace of $\mathcal{H}$, in a "representer theorem" spirit (Schölkopf et al., 2001). The attainment of the supremum is shown from topological aspects of the constraints in this subspace; and, finally, the match $\mathrm{F}(x) = \mathrm{C}(x)$ by re-evaluating the constraints in light of the solutions to $\mathbb{P}0$ being finitely representable.

Proof. Let $\mathbb{X} := X \cup \{x\}$ and define the finite-dimensional subspace $\mathcal{H}^{\|} = \{f \in \mathcal{H} : f \in \mathrm{span}(k(x_i,\cdot), x_i \in \mathbb{X})\}$. Furthermore, let $\mathcal{H}^{\perp} = \{g \in \mathcal{H} : \langle g, f^{\|}\rangle_{\mathcal{H}} = 0, \forall f^{\|} \in \mathcal{H}^{\|}\}$ be the orthogonal complement of $\mathcal{H}^{\|}$. Then, we have $\mathcal{H} = \mathcal{H}^{\|} \oplus \mathcal{H}^{\perp}$ and for all $f \in \mathcal{H}$, $\exists f^{\|} \in \mathcal{H}^{\|}, f^{\perp} \in \mathcal{H}^{\perp} : f = f^{\|} + f^{\perp}$. By employing the latter decomposition and using the reproducing property, we can reformulate $\mathbb{P}0$ in terms of $\mathcal{H}^{\|}$ and $\mathcal{H}^{\perp}$ as

$$\sup_{\substack{f^{\parallel}\in\mathcal{H}^{\parallel}\\ f^{\perp}\in\mathcal{H}^{\perp}}} \left\{ \langle f^{\parallel}+f^{\perp}, k(x,\cdot)\rangle_{\mathcal{H}} : \left\| f^{\parallel}+f^{\perp}\right\|_{\mathcal{H}}^2 \leq \Gamma^2, \left\|(f^{\parallel}+f^{\perp})_X - \mathsf{y}\right\|_{\infty} \leq \bar{\delta} \right\} \tag{2.23}$$

$$\overset{(i)}{=} \sup_{\substack{f^{\parallel}\in\mathcal{H}^{\parallel}\\ f^{\perp}\in\mathcal{H}^{\perp}}} \left\{ f^{\parallel}(x) : \left\| f^{\parallel}\right\|_{\mathcal{H}}^2 + \left\| f^{\perp}\right\|_{\mathcal{H}}^2 \leq \Gamma^2, \left\|f^{\parallel}_X - \mathsf{y}\right\|_{\infty} \leq \bar{\delta} \right\} \tag{2.24}$$

$$\overset{(ii)}{=} \sup_{f^{\parallel}\in\mathcal{H}^{\parallel}} \left\{ f^{\parallel}(x) : \left\| f^{\parallel}\right\|_{\mathcal{H}}^2 \leq \Gamma^2, \left\|f^{\parallel}_X - \mathsf{y}\right\|_{\infty} \leq \bar{\delta} \right\} \tag{2.25}$$

In (i), the $f^{\perp}$ component vanished from the cost and from the last constraint due to orthogonality w.r.t. $k(x_i,\cdot)\in\mathcal{H}^{\parallel}$ for any $x_i\in\mathbb{X}$; moreover, the Pythagorean relation $\|f\|_{\mathcal{H}}^2 = \|f^{\parallel}\|_{\mathcal{H}}^2 + \|f^{\perp}\|_{\mathcal{H}}^2$ was also used. To arrive at the second equality (ii), one only has to note that the objective is insensitive to $f^{\perp}$ and that any $f^{\perp}\neq 0_{\mathcal{H}}$ would tighten the first constraint.

The attainment of the supremum is addressed next. Consider (2.25) and denote the members of $\mathcal{H}^{\parallel}$ simply as f. $\|f\|_{\mathcal{H}}^2\leq\Gamma^2$ is a closed and bounded constraint as it is the sublevel set of a norm. We transform $\|f_X-\mathsf{y}\|_{\infty}\leq\bar{\delta}$ into $|f(x_i)-y_{i,j}|\leq\bar{\delta}$, $i=1,\ldots,d$, $j=1,\ldots,n_i$. Sets of the form $\{a\in\mathbb{R} : |a|\leq b\}$ are clearly closed in $\mathbb{R}$, hence $\{f(x_i)\in\mathbb{R} : |f(x_i)-y_{i,j}|\leq\bar{\delta},\forall i,j\}$ is also closed. For any x_i, the evaluation functional $L_{x_i}(f)=f(x_i)$ is a linear operator and thus pre-images of closed sets are also closed. Consequently, $\{f\in\mathcal{H}^{\parallel} : |f(x_i)-y_{i,j}|\leq\bar{\delta},\forall i,j\}$ is closed in $\mathcal{H}^{\parallel}$. The intersection of a finite number of closed sets is necessarily closed, thus all constraint present in (2.25) define a closed feasible set. Since $\mathcal{H}^{\parallel}$ is finite-dimensional, any closed and bounded subset of it is compact (Heine–Borel); therefore, the continuous objective $L_x(f)=f(x)$ in (2.25) attains a maximum by the Weierstrass extreme value theorem.

Finally, we establish the connection between $\mathbb{P}0$ and $\mathbb{P}1$. From the above arguments, an optimizer for $\mathbb{P}0$ must lie in $\mathcal{H}^{\parallel}$. The members $f\in\mathcal{H}^{\parallel}$ have the form $f(z)=\alpha^{\top}K_{\mathbb{X}z}$, being defined by the α weights. Due to the positive-definiteness of k, there exists a bijective map between outputs at the $\mathbb{X}$ locations $f_{\mathbb{X}}=\begin{bmatrix} f(x_1) & \ldots & f(x_d) & f(x)\end{bmatrix}^{\top}$ and the weights α, namely $\alpha=K_{\mathbb{X}\mathbb{X}}^{-1}f_{\mathbb{X}}$. $K_{\mathbb{X}\mathbb{X}}$ denotes the kernel matrix associated with $\mathbb{X}$. Consequently, optimizing over $f\in\mathcal{H}^{\parallel}$ is equivalent to optimizing over $\begin{bmatrix} f(x_1) & \ldots & f(x_d) & f(x)\end{bmatrix}^{\top} =: \begin{bmatrix} c^{\top} & c_x\end{bmatrix}^{\top}$. The bounded norm condition can be recast as $\|f\|_{\mathcal{H}}^2=\langle f,f\rangle_{\mathcal{H}}=\alpha^{\top}K_{\mathbb{X}\mathbb{X}}\alpha = \begin{bmatrix} c^{\top} & c_x\end{bmatrix} K_{\mathbb{X}\mathbb{X}}^{-1}\begin{bmatrix} c^{\top} & c_x\end{bmatrix}^{\top}$. The last constraint and the objective are straightforward, and this concludes the proof. □

Remark 5. (The optimal lower bound): In a way analogous to (2.19), the problem $\inf_{f\in\mathcal{H}}\{f(x) : \|f\|_{\mathcal{H}} \leq \Gamma, \|f_X - \mathsf{y}\|_\infty \leq \bar{\delta}\}$ could be posed to compute the minimum out-of-sample value that could be attained by the unknown ground-truth. Its finite-dimensional counter-part would be $\mathrm{B}(x) = \min_{c\in\mathbb{R}^d, c_x\in\mathbb{R}}\{c_x : (2.21), (2.22)\}$ for any $x \in \mathcal{X}\backslash X$, and extend it to points $x = x_i \in X$ with $\mathrm{B}(x) = \min_{c\in\mathbb{R}^d}\{c_i \,|\, c^\top K_{XX}^{-1} c \leq \Gamma^2, \|\Lambda\, c - \mathsf{y}\|_\infty \leq \bar{\delta}\}$. As a result, computing the whole "uncertainty envelope" requires solving two problems per query point.

An illustrative example is shown in Figure 2.3, where noisy samples were gathered from an unknown map (dashed line). The upper and lower bounds $\mathrm{C}(x)$ and $\mathrm{B}(x)$ were then computed based on an augmented norm estimate, and are shown in red. Finally, the ground-truth is shown to lie within the uncertainty envelope. We underline that this procedure does not require defining any nominal model.

Theorem 2 states that quantifying uncertainty in our particular kernelized setting can be done through convex programming involving $d + 1$ decision variables. According to (2.22), $c \in \mathbb{R}^d$ is constrained to be consistent with the already seen outputs y up to the tolerance $\bar{\delta}$; whereas according to (2.21), the ensemble c and c_x must lead to a total complexity not greater than Γ. It turns out that, for any query point x outside the data-set, the complexity bound is always activated since the objective is only sensitive to c_x, which is not constrained by the infinity norm. This is formalized next. Replacing the inequality in (2.21) by an equality would lead to the loss of convexity as is therefore not desirable.

Proposition 7. The inequality constraint (2.21) is always active, i.e., for any $x \in \mathcal{X}\backslash X$ let $(c^\star, c_x^\star)$ be an optimizer of $\mathbb{P}1$, then $\begin{bmatrix} c^\star \\ c_x^\star \end{bmatrix}^\top \begin{bmatrix} K_{XX} & K_{Xx} \\ K_{xX} & k(x,x) \end{bmatrix}^{-1} \begin{bmatrix} c^\star \\ c_x^\star \end{bmatrix} = \Gamma^2$.

Given our knowledge on the noise influence $\bar{\delta}$, it is natural to ask what the limits

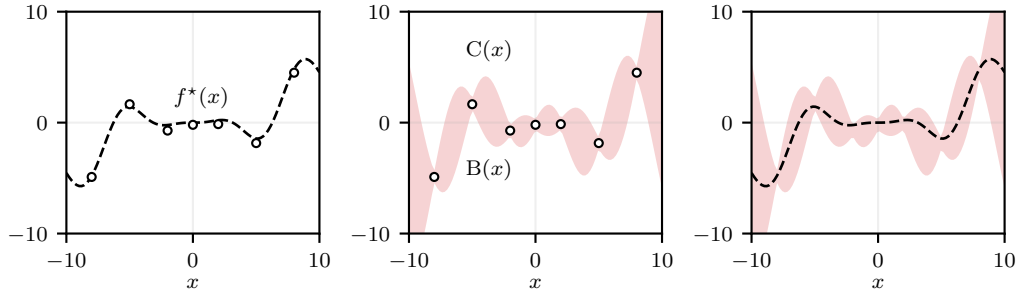

Figure 2.3: Optimal bounds example for the SE kernel (2.3) with $\ell = 2.5$. Left: a member $f \in \mathcal{H}$ with $\|f\|_{\mathcal{H}} = 16.42$ and 7 data-points with $\bar{\delta} = 0.5$. Center: data-points and the optimal bounds $\mathrm{C}(x)$, $\mathrm{B}(x)$ computed with $\Gamma = 1.1\,\|f\|_{\mathcal{H}}$. Right: the ground-truth $f(x)$ and the optimal bounds $\mathrm{C}(x)$, $\mathrm{B}(x)$.

of the uncertainty quantification technique considered herein are. More concretely, is the width of the envelope $\mathrm{C}(x) - \mathrm{B}(x)$ restricted to a certain minimum value that cannot be reduced even with the addition of new data? From (2.22), it is clear that at any input location $x_i \in X$, $\mathrm{C}(x_i)$ and $\mathrm{B}(x_i)$ cannot be more than $2\bar{\delta}$ apart. In addition to that, the presence of the complexity constraint (2.21) can bring the two values closer to each other. Depending on how restrictive this latter constraint is for a given $x = x_i$, the corresponding output y_i might lie outside the interval between $\mathrm{C}(x_i)$ and $\mathrm{B}(x_i)$. In this case, the resulting width is considerably reduced as stated next and as illustrated in Figure 2.4.

Proposition 8. (Width smaller than the noise bound): If $\exists \mathsf{y}_i$ such that $y_{i,j} > C(x_i)$ or $y_{i,j} < \mathrm{B}(x_i)$ for some j, then $\mathrm{C}(x_i) - \mathrm{B}(x_i) \leq \bar{\delta}$.

Suppose now one has sampled (x_i, y_i) with $\mathsf{y}_i = \begin{bmatrix} y_{i,1} & y_{i,2} \end{bmatrix}^\top$, $y_{i,1} = f^\star(x_i) + \bar{\delta}$ and $y_{i,2} = f^\star(x_i) - \bar{\delta}$. In this case, there is no uncertainty whatsoever about $f^\star$ at x_i since $f^\star(x_i) = (y_{i,1} + y_{i,2})/2$ is the only possible value attainable by the ground-truth. The possibility of having multiple outputs at the same location therefore allows for the uncertainty interval to shrink past the $\bar{\delta}$ width, and eventually be reduced to a singleton as shown in Figure 2.4. Notwithstanding, the addition of a new datum to an existing dataset, be it in the form of a new output at an already sampled location or a completely new input-output pair, can only reduce the uncertainty as guaranteed by the following proposition.

Proposition 9. (Decreasing uncertainty) Let $C_1(x)$ be the solution of $\mathbb{P}1$ with a dataset $D_1 = \{(x_i, \mathsf{y}_i)\}_{i=1}^d$, and $C_2(x)$ the solution with $D_2 = D_1 \cup \{(x_{d+1}, \mathsf{y}_{d+1})\}$, $\forall x_{d+1} \in \mathcal{X}, \mathsf{y}_{d+1} \in \mathbb{R}^{n_{d+1}}$. Then $C_2(x) \leq C_1(x)$ for any $x \in \Omega$.

Remark 6. An analogous result holds for the lower part of the envelope $\mathrm{B}(x)$.

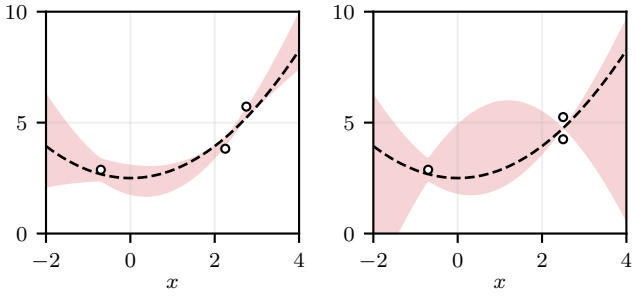

Figure 2.4: Left: samples lying outside of the uncertainty envelope, implying that its width is smaller than $\bar{\delta}$ at those locations. Right: redundant information is used to shrink the uncertainty envelope and recover the exact ground-truth value at $x = 2.5$ as $\mathrm{C}(2.5) = \mathrm{B}(2.5) = f^\star(2.5) = 4.75$.

The practical implications of Proposition 9 are shown in Figure 2.5, where samples are gradually added to the dataset, and the optimal bounds are re-computed. To create this example, the squared-exponential kernel was employed. Notice how new information decreases the uncertainty everywhere in the domain thanks to the global influence of the chosen kernel. This effect can be more clearly observed in the region $-8 \leq x \leq -5$ where the width is progressively reduced despite no new data-points being collected within it.

Remark 7. (On the accuracy of the noise bound): Recovering the ground-truth as shown in Figure 2.4 requires the noise realizations to match $\bar{\delta}$ and $-\bar{\delta}$; it is thus necessary to have *tight* noise bounds at hand for it to happen, which is not always a reality in practical applications. On the other hand, Proposition 9 guarantees the decreasing uncertainty property regardless of how accurate $\bar{\delta}$ is.

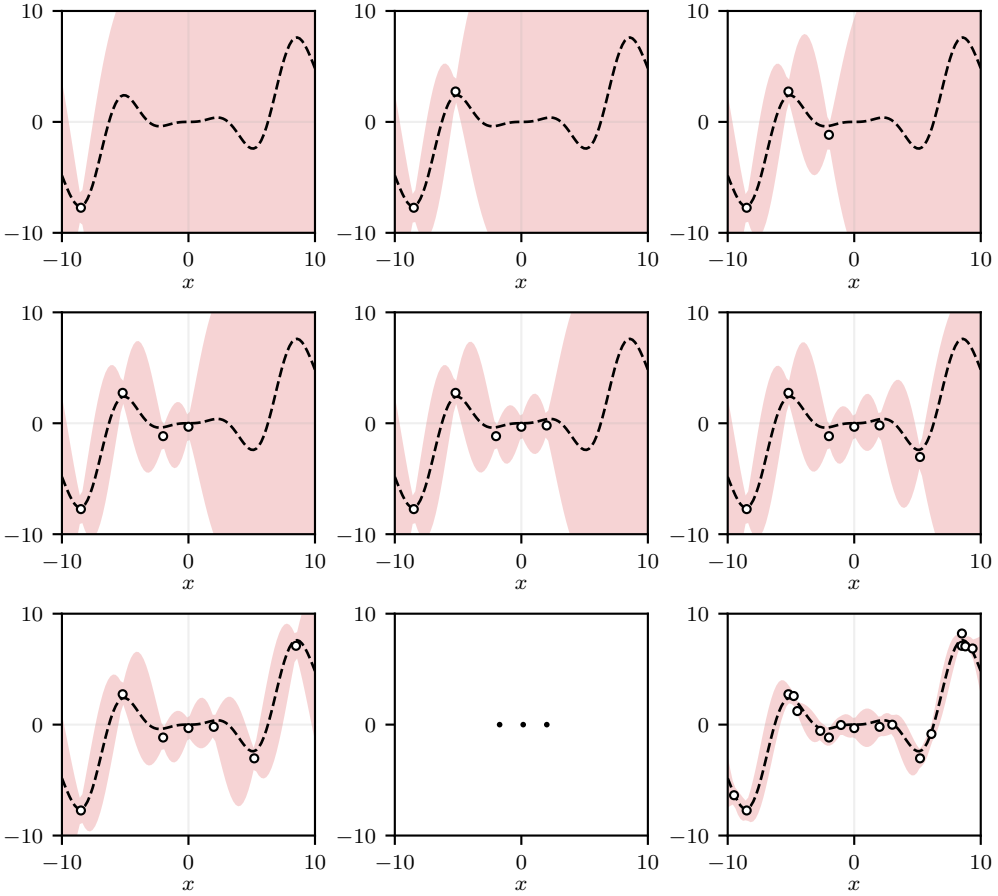

Figure 2.5: Adding new samples to a dataset can only cause uncertainty to be reduced everywhere in the domain. The noise was drawn from a uniform distribution bounded in absolute value by $\bar{\delta} = 0.8$. The last plot depicts the bounds after the collection of 10 new random samples.

One of the fundamental sources of computational complexity in problem (2.20) is the presence of the kernel matrix inverse. Indeed, it is well-known that commonly used algorithms for matrix inversion have cubic time-complexity. Note that the same problem is also faced when trying to scale other kernel-based algorithms (Zhang et al., 2013; Bauer et al., 2016; Lederer et al., 2021). In order to circumvent this obstacle, one could make use of the optimal-bounds dual formulation, which can be shown not to involve the aforementioned matrix.

Proposition 10. The Lagrangian dual of $\mathbb{P}1$ is the convex program $\mathbb{D}1$ given by

$$\min_{\nu\in\mathbb{R}^{\tilde{d}},\,\lambda>0} \frac{1}{4\lambda}\nu^\top \Lambda K_{XX}\Lambda^\top \nu + \left(\mathsf{y} - \frac{1}{2\lambda}\Lambda K_{Xx}\right)^\top \nu + \bar{\delta}\|\nu\|_1 + \frac{1}{4\lambda}k(x,x) + \lambda\Gamma^2 \quad (2.26)$$

where $\tilde{d} = \sum_{i=1}^{d} n_i$ is the total number of outputs, that is, the size of y.

The optimization problem above is convex since it is a quadratic-over-linear function with $\Lambda K_{XX}\Lambda^\top \succeq 0$ and λ restricted to the positive reals. The objective can moreover be decomposed into a differentiable part and a single non-differentiable term $\|\nu\|_1$, with ν unconstrained. This class of problems has long been studied and mature numerical algorithms exist to solve them, notably different flavors of splitting methods such as the alternating direction method of multipliers (Boyd et al., 2011, §6). Alternatively, a standard linear reformulation could be employed to replace $\|\nu\|_1$ by $\sum_i \eta_i$, with additional constraints $-\nu \leq \eta$, $\nu \leq \eta$, yielding a differentiable objective, but with extra decision variables and linear constraints.

By definition, weak duality (Bertsekas, 2009, §5) ensures that any feasible solution (ν^*, λ^*) for $\mathbb{D}1$ leads to an objective value greater or equal to the primal problem $\mathbb{P}1$ optimal value. As a result, any feasible solution for $\mathbb{D}1$ returns a valid upper-bound for the ground-truth $f^\star(x)$. When employed in real-time applications, users may thus choose not to solve $\mathbb{D}1$ to optimality since early-stopping solvers can always be done with a theoretical guarantee on the returned value. Next, a mild sufficient condition is given to ensure a zero duality gap between the primal and dual problems.

Proposition 11. (Strong duality): If $\bar{\delta} > |\delta_{i,j}|, \forall i, j$ and $\Gamma > \|f^\star\|_{\mathcal{H}}$, then no duality gap exists, i.e., $\max \mathbb{P}1 = \min \mathbb{D}1$.

2.4.3 Closed-form alternatives

The discussion in this subsection assumes that only one sample is present at each input location, i.e., $\mathsf{y}_i = y_i$ for $i = 1, \ldots, d$, so that $\mathsf{y} = y$ and $\Lambda = I$.

Before addressing the general case, consider first the simplified scenario of having

noiseless observations of the ground-truth. Closed-form prediction error bounds can then be established for MNI models, being based on the power function and on the difference between our estimate Γ and the MNI RKHS norm.

Proposition 12. Let Assumptions 1-4 hold with $\bar{\delta} = 0$. Let s be the minimum-norm interpolant $s(x) = \alpha^\top K_{Xx}, \alpha = y^\top K_{XX}^{-1}$. Then, for any $x \in \mathcal{X}$

$$\boxed{|s(x) - f^\star(x)| \leq P_X(x)\sqrt{\Gamma^2 - \|s\|_{\mathcal{H}}^2}} \tag{2.27}$$

With (2.27) in place, a more generalized inequality can be put forth, encompassing arbitrary noise levels and kernel models.

Proposition 13. Let Assumptions 1-4 hold with $\bar{\delta} \geq 0$. Let s be a given nominal model $s(x) = \alpha^\top K_{Xx}, \alpha \in \mathbb{R}^d$. Then, for any $x \in \mathcal{X}$

$$\boxed{|s(x) - f^\star(x)| \leq P_X(x)\,\sqrt{\Gamma^2 + \Delta} + \bar{\delta}\,\left\|K_{XX}^{-1}K_{Xx}\right\|_1 + |\tilde{s}(x) - s(x)|} \tag{2.28}$$

where $\tilde{s}(x) = y^\top K_{XX}^{-1} K_{Xx}$ and $\Delta = \min_{\nu \in \mathbb{R}^d} \left\{ \frac{1}{4}\nu^\top K_{XX}\nu + \nu^\top y + \bar{\delta}\,\|\nu\|_1 \right\}$.

Remark 8. (The sub-optimal bounds): Denote by $S(x)$ the right-hand side of the inequality (2.28). One can then bound the ground-truth evaluations simply through $s(x) - S(x) \leq f^\star(x) \leq s(x) + S(x)$. We shall refer to them as "the sub-optimal bounds" since the inequalities will always be looser than $B(x) \leq f^\star(x) \leq C(x)$ obtained by solving the optimization problems of Section 2.4.2.

The map $\tilde{s}(x)$ is an interpolant for the available noisy outputs y. Note also that none of the terms in (2.28) depend on the model weights α with the exception of the last term $|\tilde{s}(x) - s(x)|$. Therefore, the width $S(x)$ will be minimized when $s(x) = \tilde{s}(x) \Leftrightarrow \alpha = y^\top K_{XX}^{-1}$. Such a model choice is however not always desirable and at times a balance between smoothing the data and not diverging too much from $\tilde{s}(x)$ has to be found. This trade-off is illustrated in Figure 2.6 where the optimal bounds are compared against KRR sub-optimal ones built with the same RKHS norm estimate $\Gamma = 1.1\,\|f^\star\|_{\mathcal{H}}$ and noise bound $\bar{\delta} = 1$. Three regularization constants were employed when designing the nominal models (2.16). As can be seen from the plots, the sub-optimal bounds are always more conservative than the optimal ones and its conservativeness increases with λ.

Let us inspect more closely the source of sub-optimality in Proposition 13. To do so, the optimal bounds problem (2.20) will be reformulated and relaxed, allowing for a closed-form solution to be found.

Begin by employing a change of variables in $\mathbb{P}1$ and optimizing over (δ, c_x), $\delta := c - y$ rather than over (c, c_x). Next, apply the matrix inversion lemma (A.23) to decompose the quadratic constraint (2.21), carry out the vector-matrix-vector multiplication and solve for c_x. This process leads to

$$c_x \leq P_X(x)\sqrt{\Gamma^2 - \|\tilde{s}\|_{\mathcal{H}}^2 - \delta^\top K_{XX}^{-1}\delta + 2y^\top K_{XX}^{-1}\delta} + \tilde{s}(x) + \delta^\top K_{XX}^{-1}K_{Xx} \tag{2.29}$$

where $P_X^2(x) = k(x,x) - K_{xX}K_{XX}^{-1}K_{Xx}$, $\tilde{s}(x) = y^\top K_{XX}^{-1}K_{Xx}$ and $\|\tilde{s}\|_{\mathcal{H}}^2 = y^\top K_{XX}^{-1}y$. Since the norm constraint (2.22) is independent of c_x, (2.29) will always be active and we can optimize over the right-hand side of (2.29) instead, thus eliminating one decision variable

$$\max_{\|\delta\|_\infty \leq \bar{\delta}} P_X(x)\sqrt{\Gamma^2 - \|\tilde{s}\|_{\mathcal{H}} - \delta^\top K_{XX}^{-1}\delta + 2y^\top K_{XX}^{-1}\delta} + \tilde{s}(x) + \delta^\top K_{XX}^{-1}K_{Xx} \tag{2.30}$$

Now, relax the problem by allowing δ to attain different values inside and outside the square-root

$$\max_{\|\delta_1\|_\infty, \|\delta_1\|_\infty \leq \bar{\delta}} P_X(x)\sqrt{\Gamma^2 - \|\tilde{s}\|_{\mathcal{H}}^2 - \delta_1^\top K_{XX}^{-1}\delta_1 + 2y^\top K_{XX}^{-1}\delta_1} + \tilde{s}(x) + \delta_2^\top K_{XX}^{-1}K_{Xx} \tag{2.31}$$

The above objective is separable and the terms associated with δ_2 evaluate to $\max_{\delta_2 \in \mathbb{R}^d}\{\delta_2^\top K_{XX}^{-1}K_{Xx} : \|\delta_2\|_\infty \leq \bar{\delta}\} = \bar{\delta}\,\|K_{XX}^{-1}K_{Xx}\|_1$ since these norms are duals of each other (Boyd and Vandenberghe, 2004, §A.1.6). Notice how what is inside the square-root is independent of the parameter x and, therefore, only needs to be evaluated once for a fixed set of inputs X. Thanks to the strong duality of quadratic programming, we have that $\max_{\delta_1 \in \mathbb{R}^d}\left\{-\delta_1^\top K_{XX}^{-1}\delta_1 + 2y^\top K_{XX}^{-1}\delta_1 - \|\tilde{s}\|_{\mathcal{H}}^2 : \|\bar{\delta}_1\|_\infty \leq \bar{\delta}\right\}$ is equal to $\min_{\nu \in \mathbb{R}^d}\left\{\frac{1}{4}\nu^\top K_{XX}\nu + \nu^\top y + \bar{\delta}\,\|\nu\|_1\right\}$, which by definition is $\tilde{\Delta}$. Finally, recall that (2.31) was a (conservative) upper bound for $f^\star(x)$. Given an arbitrary

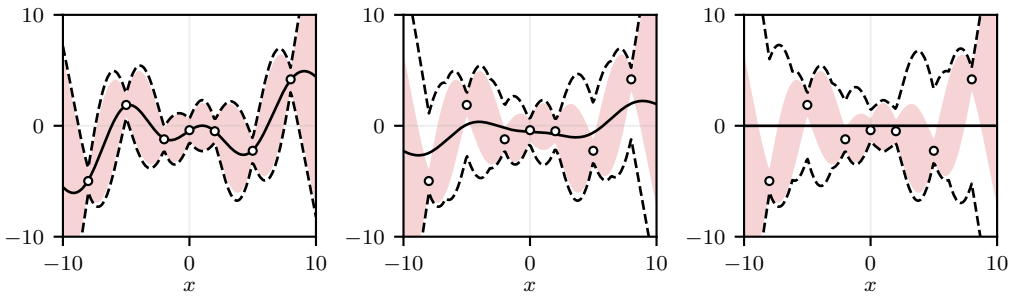

Figure 2.6: A comparison between the optimal bounds (red envelopes) and the closed-form sub-optimal bounds (dashed black lines) around KRR models (solid black lines). Three KRR distinct regularization constants were tested $\lambda = 10^{-3}$ (left), $\lambda = 10^{-1}$ (center) and $\lambda = 10^{2}$ (right).

model $s(x)$, the triangle inequality $|f(x) - s(x)| \leq |f(x) - \tilde{s}(x)| + |\tilde{s}(x) - s(x)|$ can be used to bound the distance between its predictions and the ground-truth values, where $|f(x) - \tilde{s}(x)|$ comes from the derivations made in this paragraph. We arrive thus at the same expressions presented in Proposition 13.

The proof of Proposition 13 presented in Section 2.8.4 follows a different, simpler argumentation. The derivation presented here however better highlights how the relaxed maximizaition (2.31) takes into account the worst-possible inner-product $\delta_2^\top K_{XX}^{-1} K_{Xx}$ and norm term associated with δ_1 jointly. Besides, Proposition 13 is also seen to strongly rely on the interpolant $\tilde{s}(x)$ as shown by the use of the triangle-inequality. Despite this fact, we have experimentally achieved reasonable results when in moderate noise level scenarios.

Remark 9. The sub-optimal bounds given in Proposition 13 feature a nominal model at their center, which is desirable in certain situations. In the optimal case, the minimum norm regressor $s^\star(x) = \alpha^{\star\top} K_{Xx}$, $\alpha^\star = \arg\min_{\alpha \in \mathbb{R}^d} \{\alpha^\top K_{XX} \alpha : \|K_{XX}\alpha - y\|_\infty \leq \bar{\delta}\}$ can be used as a nominal model. This choice is guaranteed to lie completely within $\mathrm{C}(x)$ and $\mathrm{B}(x)$, although not necessarily in the middle, since the map $s^\star$ belongs to $\mathcal{H}$ and is a feasible solution for $\mathbb{P}0$ in (2.19).

2.5 Kernel predictive control

The theory developed so far revolved purely around function approximation. In this subsection we make use of it to go from data to safely controlling a dynamical system by means of kernelized models and uncertianty estimates. For convenience and due to the shifting from one domain to another, notation will be overloaded.

Consider a discrete-time dynamical system of the form

$$x_{t+1} = f(x_t, u_t) \tag{2.32}$$

with time $t \in \mathbb{N}$, states $x_t \in \mathbb{R}^{n_x}$, inputs $u_t \in \mathbb{R}^{n_u}$, and an *unknown*[3] transition map $f : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \to \mathbb{R}^{n_x}$. Hard constraints $x_t \in \mathbb{X} = \{x \,|\, g_i(x) \leq 0, i = 1, \ldots, n_{\mathbb{X}}\}$ and $u_t \in \mathbb{U} = \{u \,|\, s_i(u) \leq 0, i = 1, \ldots, n_{\mathbb{U}}\}$ are imposed for all $t \in \mathbb{N}$, where $\mathbb{X}$ and $\mathbb{U}$ are polytopes. More general compact sets could also be considered, but the geometric assumptions made on $\mathbb{X}$ and $\mathbb{U}$ will later allow us to more easily verify set inclusions and constraint satisfaction.

The control goal is to steer (2.32) from a given initial condition x_0 to a polytopic

[3] If a partial model for the ground-truth function is available, the learning task can simply be performed on the residual dynamics.

safe subset of the state space

$$\mathbb{X}_{\text{safe}} \subseteq \mathbb{X} \tag{2.33}$$

while satisfying all constraints. Similarly to Koller et al. (2018), we also assume that a local policy $\pi_{\text{safe}} : \mathbb{X}_{\text{safe}} \to \mathbb{U}$ is available, making $\mathbb{X}_{\text{safe}}$ forward invariant, i.e., $\forall x_t \in \mathbb{X}_{\text{safe}} : f(x_t, \pi_{\text{safe}}(x_t)) \in \mathbb{X}_{\text{safe}}, \pi_{\text{safe}}(x_t) \in \mathbb{U}$. A frequent instance of this problem is the regulation of (2.32) to a specific fixed point, in which $\mathbb{X}_{\text{safe}} = \{x_{\text{eq}}\}$ and $\pi(x_{eq}) = u_{eq}$ is the equilibrium input. In order to accomplish our task, we assume to have access to noise-corrupted measurements of the unknown ground-truth dynamics in a format that will be later specified.

The approach chosen to tackle the problem is that of recursively solving a finite-horizon optimal control problem, i.e., the MPC approach. To perform predictions, data could be used to build a single-step surrogate model and iterate over it. The number one obstacle to be overcome in this case is that of uncertainty propagation. Indeed, propagating sets through non-linear maps usually cannot be done in closed-form and involves several overbouding steps (Koller et al., 2018). We therefore opt for learning various *multi-step ahead models*, one for each of the N prediction steps. Let $F_1 : \mathbb{X} \times \mathbb{U} \to \mathbb{X}$ be the one-step ahead predictor in which each dimension in learned separately by models $\hat{f}_1, \dots, \hat{f}_{n_x}$ [4]

$$(x_0, u_0) \mapsto F_1(x_0, u_0) = \left(\hat{f}_1(x_0, u_0), \dots, \hat{f}_{n_x}(x_0, u_0)\right) \tag{2.34}$$

Define F_2 as $(x_0, u_0, u_1) \mapsto F_2(x_0, u_0, u_1) = \left(\hat{f}_1(x_0, u_0, u_1), \dots, \hat{f}_{n_x}(x_0, u_0, u_1)\right)$, the two-step ahead model, and $F_3, \dots, F_N$ analogously.

Besides having nominal predictions, robust *confidence sets* are used to ensure constraint satisfaction. These are set-valued functions of the type $\mathbb{X}_1 : \mathbb{X} \times \mathbb{U} :\to 2^{\mathbb{X}}$, $\mathbb{X}_2 : \mathbb{X} \times \mathbb{U} \times \mathbb{U} :\to 2^{\mathbb{X}}$, ... that are guaranteed to contain the unknown system evolution

$$\begin{aligned} \mathbb{X}_1(x_0, u_0) &\ni f(x_0, u_0) \\ \mathbb{X}_2(x_0, u_0, u_1) &\ni f(f(x_0, u_0), u_1) \\ &\dots \\ \mathbb{X}_N(x_0, u_0, u_1, \dots, u_{N-1}) &\ni f(\dots f(f(x_0, u_0), u_1), \dots, u_{N-1}). \end{aligned} \tag{2.35}$$

The diagram shown in Figure 2.7 illustrates the nominal predictions F_i and confidence sets $\mathbb{X}_i$ all spawning from the initial state x_0. Notice how as defined above, F_i and $\mathbb{X}_i$ are independent elements and need not to satisfy $F_i \in \mathbb{X}_i$,

[4] Learning each state dimension separately is necessary because the theory developed in this chapter only encompasses real-valued functions.

let alone being exactly in the geometric center of it as depicted in Figure 2.7. Nevertheless, both these properties can be obtained if the nominal model and bounds given in Proposition 13 are employed. Also, the confidence sets are illustrated as hyper-rectangles to reflect the fact that uncertainty quantification is carried out individually in each dimension of the state-space.

For the sake of training the multi-step ahead models and confidence sets, a particular set of data has to be gathered. For F_1 and $\mathbb{X}_1$, one needs $\{(x_t, u_t), y_t\}_{t=1}^d$, $y_t = f(x_t, u_t) + \delta_t$, where $|\delta_t| \leq \bar{\delta}$ and where disconnected one-step sequences can be put together. For F_2 and $\mathbb{X}_2$, one needs $\{(x_t, u_t, u_{t+1}), y_t\}_{t=1}^d$, $y_t = f(f(x_t, u_t), u_{t+1}) + \delta_t$. Following the same logic, for a general pair F_N and $\mathbb{X}_N$, one needs tuples of states x_t and $N-1$ control moves $u_t, \ldots, u_{N-1}$ and the Nth step ahead state x_{t+N} possibly corrupted by noise.

Remark 10. (On collecting the multi-step ahead data): In practical scenarios, we typically perform various long experiments with the dynamical system to collect data and craft models. Clearly, an experiment that starts at a given initial state x_0, is followed by a sequence of inputs $u_t, \ldots, u_{N-1}$, and where all states are logged $x_1, \ldots, x_N$ (possibly with noise), yields one training tuple for every model F_t and confidence set $\mathbb{X}_t$, $t = 1, \ldots, N$. We highlight that it is however not possible to extract multiple tuples from a long experiment to assemble the 1-step ahead dataset if noise is present. This is due to our measurement model (2.18), whereby the initial state of a tuple cannot be corrupted.

With nominal models and confidence sets at hand, the finite-horizon optimal control problem can be formulated. Let x_0 be a given initial condition for the true dynamical system (2.32), our Kernel Predictive Control formulation then reads

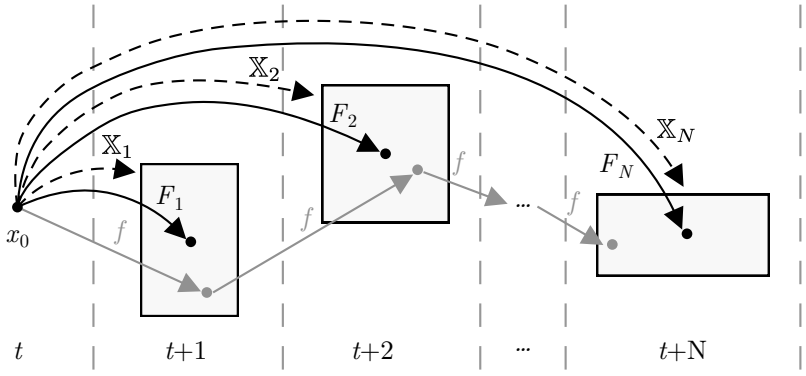

Figure 2.7: The outcome of each map: Nominal predictions F_t (—), confidence sets $\mathbb{X}_t$ (- -), and the unknown ground-truth f (—). All functions also depend on the chosen control sequence, which is omitted for clarity.

$$\text{KPC}: \min_{X,U} \quad \sum_{t=0}^{N-1} \ell(x_t, u_t) + \ell_f(x_N) \tag{2.36a}$$

$$\text{subj. to} \quad x_t = F_t(x_0, u_0, \ldots, u_{t-1}),\ \forall t \tag{2.36b}$$

$$\mathbb{X}_t(x_0, u_0, \ldots, u_{t-1}) \subseteq \mathbb{X},\ \forall t \tag{2.36c}$$

$$\mathbb{X}_N(x_0, u_0, \ldots, u_{N-1}) \subseteq \mathbb{X}_{\text{safe}} \tag{2.36d}$$

$$u_t \in \mathbb{U},\ \forall t \tag{2.36e}$$

where $X = (x_1, \ldots, x_N)$, $U = (u_0, \ldots, u_{N-1})$ are the decision variables, and $\ell(x, u)$ and $\ell_f(x)$ are appropriately designed stage and final costs. In a receding-horizon implementation, KPC is solved recursively, always supplying the current state of (2.32) to the optimization problem as the initial condition x_0, and only applying the first optimal control input to (2.32).

After having introduced (2.36), its properties should be discussed. First, recursive feasibility cannot be shown by means of the classical sequence shifting arguments due to the use of multiple confidence sets $\mathbb{X}_t$. Nevertheless, previous successful iterations can contribute to the feasibility of future KPC problems. More precisely, the closed-loop data of up to $N-1$ past steps can be used to reduce the uncertainty regarding the location of true next states *without updating any models*. This is formalized in the safe relaxation strategy (SRS) stated next.

Proposition 14. Assume that the $N-1$ previous consecutive *KPC* iterations were feasible. Denote by $(u_{-N+1}, \ldots, u_{-1})$ and $(x_{-N+1}, \ldots, x_{-1})$ the closed-loop sequences of past controls and states, and by x_0 the current state. Let KPC^+ be the *KPC* optimization problem (3.10) with set constraints $\mathbb{X}_1(x_0, u_0) \subseteq \mathbb{X}$, $\mathbb{X}_2(x_0, u_0, u_1) \subseteq \mathbb{X}$, $\ldots$, $\mathbb{X}_{N-1}(x_0, u_0, \ldots, u_{N-2}) \subseteq \mathbb{X}$ relaxed to

$$\begin{aligned} \mathbb{X}_1(x_0, u_0) \cap \left(\bigcap_{i=2}^{N} \mathbb{X}_i\left(x_{-i+1}, u_{-i+1}, \ldots, u_{-1}, u_0\right) \right) \subseteq \mathbb{X}, \\ \mathbb{X}_2(x_0, u_0, u_1) \cap \left(\bigcap_{i=3}^{N} \mathbb{X}_i\left(x_{-i+2}, u_{-i+2}, \ldots, u_{-1}, u_0, u_1\right) \right) \subseteq \mathbb{X},\ \ldots \end{aligned} \tag{2.37}$$

Let KPC^+ be feasible and $(X^\star, U^\star)$ be any of its feasible solutions. Then, $U^\star = (u_0^\star, \ldots, u_{N-1}^\star)$ drives the true system (2.32) from x_0 to the safe set $\mathbb{X}_{\text{safe}}$ while satisfying the constraints at all times, i.e., $f(x_0, u_0^\star)$, $f(f(x_0, u_0^\star), u_1^\star), \ldots \in \mathbb{X}$, and $f(\ldots f(f(x_0, u_0^\star), u_1^\star), \ldots, u_{N-1}^\star) \in \mathbb{X}_{\text{safe}}$.

Remark 11. If only $M < N-1$ previous KPC iterations were feasible, (2.37) can be adapted to have less set intersections and still achieve some relaxation.

Let the *KPC* problem (2.36) be feasible and $(X^\star, U^\star)$ be any of its feasible solutions. The sequence of inputs $U^\star = (u_0^\star, \ldots, u_{N-1}^\star)$ drives the true system (2.32) from x_0 to the safe set $\mathbb{X}_{\text{safe}}$ while satisfying the constraints at all times, i.e., $f(x_0, u_0^\star)$, $f(f(x_0, u_0^\star), u_1^\star), \ldots \in \mathbb{X}$, and $f(\ldots f(f(x_0, u_0^\star), u_1^\star), \ldots, u_{N-1}^\star) \in \mathbb{X}_{\text{safe}}$. This is evident from the robust confidence sets definition in (2.35). What the SRS property given in Proposition 14 ensures is that this latter guarantee remains unchanged even after imposing less strict set constraints.

In order to numerically solve the KPC problem (2.36), the set inclusions have to assume a more concrete form. Suppose that the confidence sets are described by

$$\mathbb{X}_t(x_0, u_o, \ldots, u_{t-1}) = \{x \,||x - F_t(x_0, u_o, \ldots, u_{t-1})| \leq \mathcal{B}_t(x_0, u_o, \ldots, u_{t-1})\} \quad (2.38)$$

where the absolute value $|\cdot|$ and the inequality are to be understood element-wise. Put differently, (2.38) defines $\mathbb{X}_t$ as a hyperrectangle whose center is the nominal prediction F_t, just as depicted in Figure 2.7. Such robust confidence sets can be obtained through the use of Proposition 13. In this case, a coordinate transformation can be employed along with the closed-form solution of the obtained hyper-cube support function to reformulate the constraint. More concretely, consider the condition $\mathbb{X}_t \subseteq \mathbb{X}$, where the arguments of $\mathcal{X}_t$ are omitted to ease notation. This constraint is satisfied if for each one of the half-spaces $g_i(x) = H_i x - h_i \leq 0$ that describe the polyhedron $\mathbb{X}$ it holds that

$$\begin{aligned}
&\forall x \in \mathbb{X}_t : \; g_i(x) \leq 0 && (2.39a)\\
\Leftrightarrow\; &\max\{g_i(x) \,|\, x \in \mathbb{X}_t\} \leq 0 && (2.39b)\\
\Leftrightarrow\; &\max\{H_i^\top x - h_i \,|\, |x - F_t| \leq \mathcal{B}_t\} \leq 0 && (2.39c)\\
\Leftrightarrow\; &\max\{H_i^\top (B_t\, x + F_t) - h_i \,|\, \|x\|_\infty \leq 1\} \leq 0 && (2.39d)\\
\Leftrightarrow\; &\|H_i^\top B_t\|_1 + H_i^\top F_t - h_i \leq 0 && (2.39e)
\end{aligned}$$

where the arguments of $\mathbb{X}_t$, F_t and $\mathcal{B}_t$ were omitted to ease notation, and $B_t := \text{diag}\,(\mathcal{B}_t) \in \mathbb{R}^{n_x \times n_x}$. In contrast with the abstract set inclusions found in (2.37), (2.39e) is implementable[5]. It should be underlined that (2.39e) is still a non-convex inequality constraint as B_t and F_t are kernelized functions of the decision variables.

Remark 12. The reformulation (2.39) is needed to tackle general polytopic feasible sets $\mathbb{X}$ that are not necessarily aligned with our uncertainty estimates. If, however, $\mathbb{X}$ is simply a hyper-box aligned with each axis and, thus, also with $\mathbb{X}_t$, verifying the inclusion becomes trivial.

[5]The one-norm can be eliminated from (2.39e) by introducing new auxiliary variables and inequality constraints, a standard linear programming procedure.

2.6 Numerical examples

A number of numerical examples are presented next to put to test the previously proposed strategies in different settings. This will also allow us to better understand what they entail and what practical aspects one should mind to avoid pitfalls.

Example 1: We start with the uncertainty quantification problem of the following map

$$f^\star(z_1, z_2) = 1 - 0.8z_1^2 + z_2 + 8\sin(0.8z_2) \tag{2.40}$$

restricted to the domain $\mathcal{X} = \{x \in \mathbb{R}^2 \,|\, \|x\|_\infty \leq 10\}$, $x = [z_1 \;\; z_2]$. A squared-exponential kernel $k(x, x') = \exp\left(-\frac{\|x-x'\|^2}{2\ell^2}\right)$ was chosen for our experiments with lengthscale $l = 5$, which was empirically estimated by gridding the search-space and performing a posterior validation.

Strictly speaking, the UQ techniques developed in this chapter require $f^\star \in \mathcal{H}$ as per Assumption 2. As opposed to the examples depicted in Figures 2.3–2.6 whose ground-truths were finite sums of partially evaluated kernels, (2.40) is a polynomial-trigonometric map and it is not clear if it belongs or not to the SE RKHS. One should nevertheless bear in mind that such an RKHS is dense in the space of continuous functions as stated in Proposition 4 and even if $f^\star \notin \mathcal{H}$, there exists a $f^* \in \mathcal{H}$ that is point-wise arbitrarily close to $f^\star$. As a result, it is customary in practice to overlook Assumption 2 if we are sure that the function being considered is continuous and the kernel, universal.

The norm estimate Γ was obtained through the sampling procedure described in Appendix 2.8.2 with a final value of $\Gamma = 1200$. Next, $d = 100$ samples were collected in two different ways: points lying on an equidistant grid, and being drawn randomly from a uniform distribution. Noise was added to both data-sets, sampled uniformly from the interval $-\bar{\delta} \leq \delta \leq \bar{\delta}$ with $\bar{\delta} = 1$ and later with $\bar{\delta} = 5$.

Figure 2.8 displays the ground-truth $f^\star$ along with the optimal upper bound $\mathrm{C}(x)$ computed by solving (2.20) for all four scenarios. Consider first the cases where $\bar{\delta} = 1$. Whereas the $\mathrm{C}(x)$ surface was overall tight for the grid-based dataset, with an average distance of 3.01 to the ground-truth, randomized data yielded a less regular bound with an average distance of 8.02. These numbers were increased respectively to 9.57 and 18.97 when the noise level was risen to $\bar{\delta} = 5$. The plots illustrate the disadvantages of relying on completely randomized input locations, which degrade especially the borders of $\mathrm{C}(x)$. An equidistant grid of points is highly favorable since it not only fills the domain well, but also ensures a minimum separation distance so that no two inputs are too close to each other, thus also avoiding numerical problems when handling the kernel matrix K_{XX}.

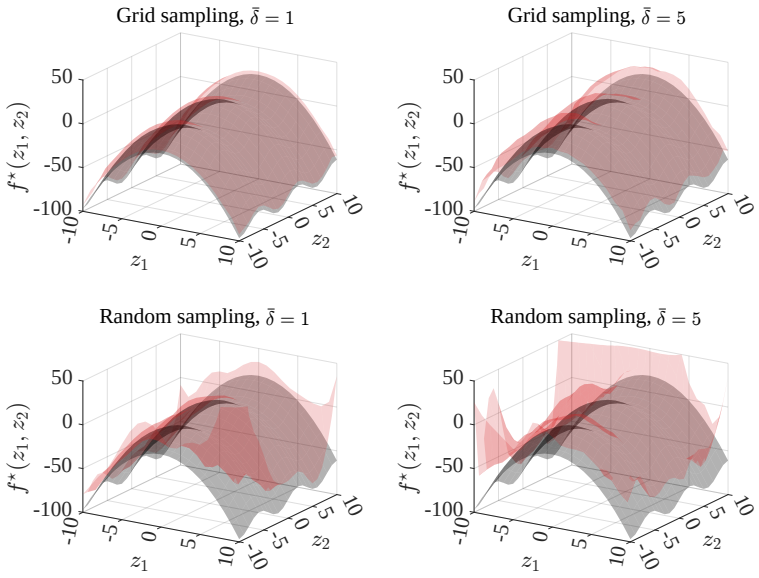

Figure 2.8: The ground-truth (black) and the optimal upper bound $\mathrm{C}(x)$ (red) with 100 data-points and different sampling methods and noise levels.

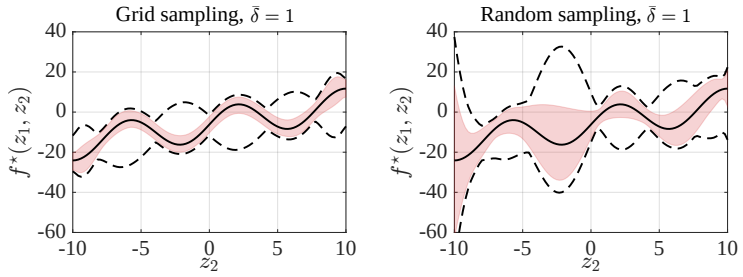

Figure 2.9: The ground-truth (solid black) sliced at $z_1 = -3$, the optimal bounds (red) and the sub-optimal bounds (dashed black).

The $f^\star(z_1, z_2)$ map was then sliced at $z_1 = -3$. The entire envelope $\mathrm{B}(x) \leq x \leq \mathrm{C}(x)$ was computed and compared to the sub-optimal bounds given in Proposition 13 for a kernel ridge regression model with $\lambda = 10^{-2}$. The obtained results for $\bar{\delta} = 1$ are shown in Figure 2.9. As can be seen from the plots, the optimal approach yielded tighter uncertainty intervals that were always within the sub-optimal ones. Moreover, whereas the average width of the red envelope was 8.93 and 18.96 respectively in the grid and random cases, the dashed black envelope displayed average widths of 21.13 and 34.62.

It is desirable to have tight bounds $\bar{\delta}$ and Γ in order to reduce conservativeness. This is however not possible in most practical applications where one has to estimate those quantities from data alone. To investigate the sensitivity of the resulting bounds with respect to those parameters, $\bar{\delta}$ and Γ were varied to build

Table 2.1: Noise uniformly sampled from $-1 \leq \delta \leq 1$. 100 data-points were collected from the ground-truth (2.40) in an equidistant grid (grid) or randomly following a uniform distribution (rnd). The table shows the average distance between the upper and lower bounds in the optimal (opt) and sub-optimal (sub) cases with various noise bounds $\bar{\delta}$ and RKHS norm estimates Γ.

	Γ	1200			1800			2400		
	$\bar{\delta}$	1	1.5	2	1	1.5	2	1	1.5	2
grid	opt	6.21	8.35	10.34	7.45	9.75	11.90	8.50	10.94	13.20
	sub	11.07	15.60	20.13	11.70	16.23	20.76	12.36	16.89	21.42
rnd	opt	14.62	19.02	22.89	18.05	23.08	27.51	20.85	26.39	31.26
	sub	64.78	93.99	123.20	65.91	95.12	124.33	67.07	96.28	125.49

Table 2.2: Noise uniformly sampled from $-5 \leq \delta \leq 5$. 100 data-points were collected from the ground-truth (2.40) in an equidistant grid (grid) or randomly following a uniform distribution (rnd). The table shows the average distance between the upper and lower bounds in the optimal (opt) and sub-optimal (sub) cases with various noise bounds $\bar{\delta}$ and RKHS norm estimates Γ.

	Γ	1200			1800			2400		
	$\bar{\delta}$	5	7.5	10	5	7.5	10	5	7.5	10
grid	opt	20.29	28.57	36.39	22.54	31.31	39.58	24.41	33.56	42.17
	sub	49.15	71.79	94.44	49.81	72.46	95.11	50.48	73.13	95.78
rnd	opt	39.95	53.43	65.41	47.00	62.15	75.57	52.76	69.32	83.89
	sub	312.44	458.51	604.57	313.61	459.68	605.74	314.79	460.85	606.91

a total of 18 test scenarios, divided into two noise cases as shown in Tables 2.1 and 2.2. The figures are consistent in showing that the average width was not significantly affected by an increase in Γ. Analyzing the numbers from Table 2.1, the width of the optimal envelope increased on average by 34% when augmenting Γ from 1200 to 2400, and the width of the sub-optimal envelope increased on average by 5.2%. Repeating the analysis on Table 2.2, the optimal envelope showed an average width increase of 24%, whereas the sub-optimal one, of 1.38%. Despite the larger relative increase of the optimal bounds width, their absolute values were on average 3.45 times smaller than the sub-optimal ones in Table 2.1 and 5.79 times smaller in Table 2.2, which exposes the degree of conservativeness of the latter approach.

Example 2: The next numerical experiment involves having the map (2.40) as a constraint in a static optimization problem[6]. Consider the following formulation

$$\min_{z \in \mathbb{R}^2} \quad (z_1 - 1)^2 + (z_2 - 5)^2 \tag{2.41a}$$

$$\text{subj. to} \quad f^\star(z) \leq -10, \tag{2.41b}$$

where the function $f^\star(z)$ given by (2.40) is not explicitly known, but can be measured, evaluated. Samples were used to establish an upper bound $\mathrm{C}(z)$ for $f^\star(z)$, hence providing an inner-approximation for the true feasible set of (2.41).

We considered the cases of having 64, 81 and 100 evaluations of $f^\star(z)$ affected by noise with $\bar{\delta} = 1$ and, once more, the data were collected by means of a uniform random distribution and an equidistant grid. The same kernel function and RKHS norm estimate of $\Gamma = 1200$ from Example 1 were used. In the surrogate optimization problems, (2.41b) was replaced by $C(z) \leq -10$. Optimizers $z^\star$ were computed by gridding the domain, and the results along with the estimated feasible sets (shaded areas) are shown in Figure 2.10. Notice how in some instances the set of feasible decisions is not connected. Thanks to Proposition 9, the addition of new data-points can only relax the approximate formulation, hence reducing the found minimum. Indeed, the obtained solutions for the approximate problems were 13.21, 11.36 and 10.96, respectively with 64, 81 and 100 samples taken randomly. When employing a grid, the figures were 10.67, 8.48 and 7.67. The solution of the real problem, i.e., the one with the ground-truth constraint, is 5.69.

Example 3: In this example, UQ will be used to verify the safety of a sequence of control actions driving a dynamical system. As opposed to the grid-based and randomized sampling methods used in the previous examples, the dynamics will

[6]Maximizing/minimizing a given objective under unknown constraints is typical in the field of real-time optimization, see e.g. (Chachuat et al., 2009).

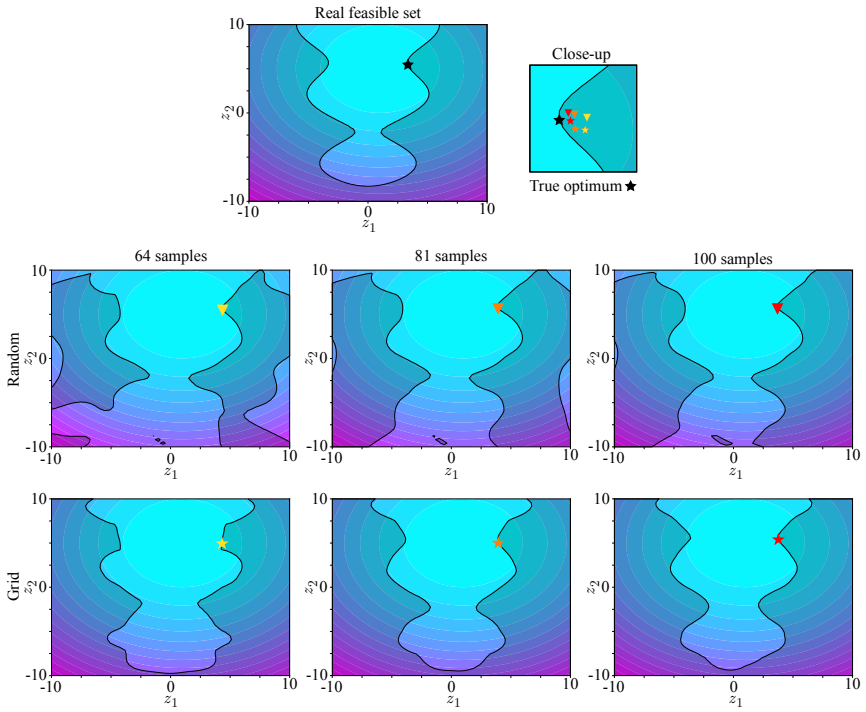

Figure 2.10: Solutions and feasible sets (shaded areas) of the surrogate optimization problems, i.e., (2.41) with the constraint replaced by its optimal upper bound $\mathrm{C}(x) \leq -10$. The individual solutions are denoted by the yellow, orange and red triangles and stars. The true optimum is indicated by a black star.

be taken into account and data will be gathered directly from trajectories.

Consider a continuous stirred-tank reactor (CSTR) described by the continuous-time differential equations

$$\dot{c}_A(t) = u(t)(c_{A0} - c_A(t)) - \rho_1 c_A(t) - \rho_3 c_A^2(t) \tag{2.42a}$$

$$\dot{c}_B(t) = -u(t)c_B(t) + \rho_1 c_A(t) - \rho_2 c_B^2(t) \tag{2.42b}$$

where c_A and c_B denote respectively the concentrations of cyclopentadiene and cyclopentenol, whereas u represents the feed inflow of cyclopentadiene. The parameters are $\rho_1 = \rho_2 = 4.1 \times 10^{-3}$ h^{-1}, $\rho_3 = 6.3 \times 10^{-4}$ h^{-1}, $c_{A0} = 5.1$ mol/l. The system is subject to the constraints $1 \leq c_A \leq 3$, $0.5 \leq c_B \leq 2$, $3 \leq u \leq 25$, and is sampled at a rate of 1/30 Hz. The goal is to drive the states from a number of initial conditions to the reference values $c_A^{\mathrm{ref}} = 2.14$, $c_B^{\mathrm{ref}} = 1.09$. To achieve this objective, a finite-horizon optimal control problem (OCP) with horizon $N = 8$ was formulated employing a standard quadratic cost with a larger terminal weight but no terminal set constraint. The OCP relied on a KRR nominal model learned from data and no uncertainty quantification, i.e., relied on certainty equivalence.

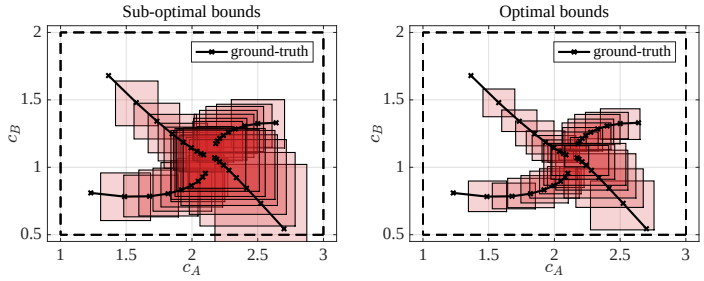

Figure 2.11: Phase portraits of the CSTR system under the same control inputs, but with different uncertainty quantification techniques. Constraints are represented by the dashed lines.

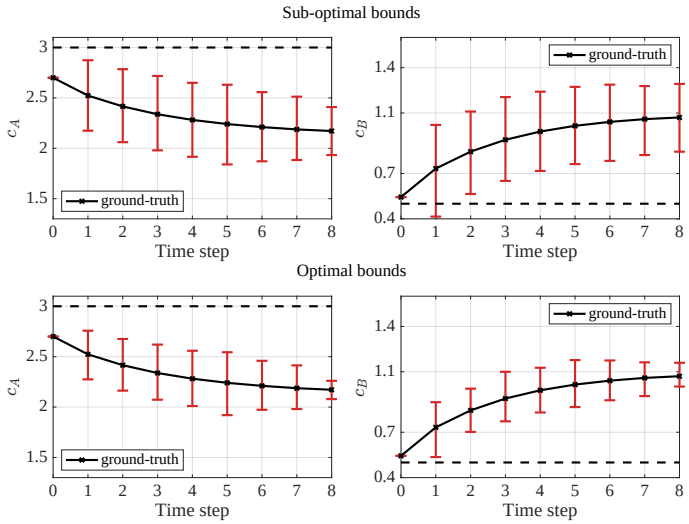

Figure 2.12: State trajectories of the CSTR system under the same control inputs, but with different uncertainty quantification techniques. Constraints are represented by the dashed lines.

To verify if computed OCP control actions would not lead to the system violating constraints, we employed our kernelized UQ technique. First, the problem was broken down into several steps: the 1-step ahead analysis, the 2-step ahead analysis, and so on. The datasets associated with each step are composed of initial states and sequences of control actions to form x_i, and the final state to form y_i (this multi-step approach is the same as the one explained in Section 2.5). The squared-exponential kernel was used throughout the whole process and the various lengthscales were selected through a 5-fold cross-validation process based on 400 samples (see Section 2.8.1). The same batch of data were exploited to estimate the different RKHS norms Γ following the procedure of Section 2.8.2. The obtained lower estimates were then augmented to account for possible unseen complexity. Distinct datasets were gathered to compute the bounds by starting the system at

800 initial conditions and solving OCP from those locations. We highlight that, as there are two states and one control variable, the domain of the function mapping the initial condition to the 8-th step ahead state has dimension 10, hence justifying the need for a large dataset. The noise affecting the measurements was drawn uniformly from the interval -0.05 to 0.05, and a bound of $\bar{\delta} = 0.06$ was used.

The optimal bounds (2.20) and the sub-optimal ones (2.28) were employed to define robust confidence sets for each time-step. These are guaranteed to contain the true system evolution, the ground-truth, as illustrated by the phase portrait and the bounding boxes in Figure 2.11. Based solely on the sub-optimal approach, one would certify three out of four control sequences as the one confidence set in the lower right portion of the space extends beyond the state constraints. Nevertheless, by using the optimal bounds we can conclude that no violation would occur as all confidence sets are completely within the feasible region. A time-domain view of that particular trajectory is shown in Figure 2.12. It is important to underline that both approaches are based exactly on the same information (datasets, noise bounds and RKHS norm bounds), and only differ in the way they are computed.

Example 4: The continuous-time angular dynamics of a pendulum with a rigid rod can be described by $\dot{x}_1 = x_2$, $\dot{x}_2 = (g/l)\sin(x_1) - (\nu/(ml^2))x_2 + (1/(ml^2))u$, where x_1 is its angular position, x_2 its angular velocity, and u the torque applied to it. The parameters are: $m = 0.15$ the mass of the pendulum, $l = 0.5$ the length of the rod, $g = 9.81$ the gravitational constant, and $\nu = 0.1$ a constant for the friction model. Let the constraints be $|x_1| \leq 3$, $|x_2| \leq 1$, and the input be limited to $|u| \leq 1$. We selected a prediction horizon of $N = 4$ and collected $D = 100$ uniformly random data-points for each of the eight regression tasks. The noise was drawn randomly from a uniform distribution with bound $\bar{\delta} = 0.01\,\mathbf{1}$, where $\mathbf{1} \in \mathbb{R}^D$ is a vector of ones. Similarly to the previous example, we used a squared-exponential kernel with increasing lengthscales and employed an augmentation factor of 300% on the nominal predictor norms to estimate the Γ constants. The sampling and control period was 0.2 seconds. We compared KPC against *nominal* kernel-MPC, i.e., a certainty equivalence approach where the state-constraints were imposed directly on the nominal predictions (2.36b). In the latter case, uncertainty was not quantified and the confidence sets were not present. The cost used in both formulations was a positive definite function of the states and control inputs, and included a terminal penalty term.

Predictions and the system angular velocity evolution from two different initial conditions are shown in Figure 2.13. As can be seen from the plots, imposing the state-constraints on the nominal model predictions was not sufficient to guarantee safety as the closed-loop system behavior violated the $x_2 \geq -1$ restriction. On

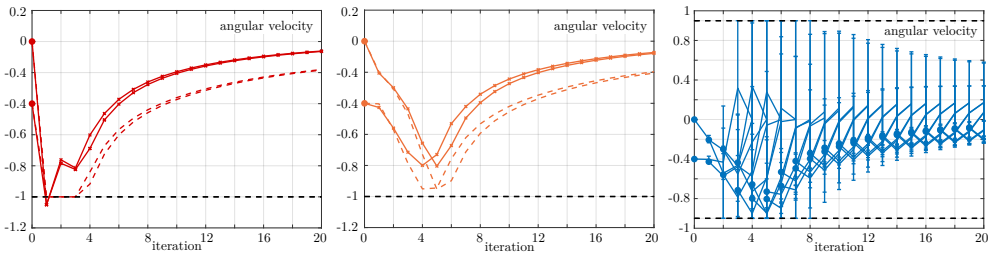

Figure 2.13: Left: Nominal kernel-MPC predictions (dashed) and closed-loop trajectory (solid). Center: KPC predictions (dashed) and closed-loop trajectory (solid). Right: KPC full open-loop predictions, confidence intervals, and initial points depicted as circular markers.

the other hand, since KPC quantified and incorporated the associated uncertainty into the optimization problems, the constraints were satisfied. The error bars on the right plot show the all predictions and uncertainty values at each step in the form of error bars. Note that the safety constraints were active at multiple points in time.

2.7 Conclusions and outlook

In this first part of the thesis, we introduced and developed a novel way of quantifying the out-of-sample uncertainty of the values of an unknown function. The framework was built around kernels, their RKHS spaces, and a bounded measurement noise assumption from which no statistical information is known. Optimal hard upper and lower bounds were then derived, being computable through convex optimization. Closed-form conservative approximation schemes were given to be used in time-constrained scenarios. Some properties of the optimal and sub-optimal approaches were derived and their sensitivity to the internal parameters (the RKHS norm estimate and the noise bound) were studied. The techniques were finally employed to solve a number of problems ranging from a direct function bounding one, to solving an optimization problem with unknown constraints, to certifying optimal control actions.

We envision multiple future investigations and extensions of the theory presented in this chapter. First, additional properties of the optimal bounds could be established, in particular, although we have not had the time to prove it, we suspect the upper and lower envelopes $\mathrm{C}(x)$ and $\mathrm{B}(x)$ to be continuous and piecewise smooth whenever the base kernel k is smooth. Deriving this result would likely involve studying the solution set and the value function of the parametric program (2.20). One necessary extension to allow for the learning of dynamical systems from trajectories, and not resorting to multi-step ahead models, is to consider

uncertainty in x itself. This would call of course for robust parametric programming to be employed. Finally, on the application side, the KPC framework presented in Section 2.5 guarantees constraint satisfaction, but can be rather challenging to implement in practice due to the use of multiple models. Long prediction horizons are generally desirable to favor stability, however, this requires more models to be trained in higher dimensional spaces since longer and longer control sequences are incorporated into the feature vector. Estimating hyperparameters and filling the space with data then become difficult tasks in such scenarios, limiting KPC to short prediction horizons. To eliminate some of these difficulties, a different approach to kernel-based predictive control could be adopted, that of optimism. Instead of being robust against all possible ground-truth dynamics, one could pick a dynamic that is consistent with the historical data and complexity bound, the most convenient one. As the system is operated and more information is gathered, the set of possibilities is reduced and the adopted dynamic would be more consistent with the true unknown dynamics. We believe that, based on recent regret bounds for kernelized global optimization, convergence rates could be derived for the iterative performance improvement of this scheme and bounds could be established for the constraint violation frequency.

2.8 Appendices

2.8.1 Estimating kernel hyperparameters

Kernel functions typically feature a number of internal constants that need to be specified by the user, the so-called *hyperparameters*. Although some theoretical properties remain insensitive to the final choice of hyperparameters[7], the numerical stability and real-world performance of kernel algorithms highly depend on the tuning of these numbers (Fasshauer, 2011).

One popular approach to optimizing kernel hyperparameters is to consider the log marginal likelihood objective of a Gaussian process with Gaussian measurement noise, and apply a gradient-based numerical algorithm to it (Williams and Rasmussen, 2006, §5.4.1). This amounts to solving a smooth, unconstrained non-convex optimization problem to local optimality. Two appealing features of this approach are the continuous nature of the search and the inherent regularization properties of the objective, known to combat overfitting.

Bayesian optimization (BayesOpt) is another widely adopted methodology to

[7] As an example, the squared-exponential kernel is universal (see Definition 5) regardless of the its lengthscale value.

fine-tune hyperparameters against a pre-specified objective function (Snoek et al., 2012; Shahriari et al., 2015). BayesOpt operates on pairs of hyperparameters and their associated objective function values, and constructs a model for their unknown relationship. Next, a so-called *acquisition function* based on the model is then employed to tell bad hyperparameters from promising ones (Wilson et al., 2018). A new set of parameters is finally chosen to be experimented with and its performance is measured. This new piece of information is incorporated into the dataset and the process is repeated until some termination criterion is reached. Interestingly, the most popular model class used when reconstructing the aforementioned unknown relationship is Gaussian process, which is itself based on kernels whose own hyperparameters have to be set by the users.

Lastly, we could also mention the different flavors of cross-validation (CV). The procedure builds an estimate of how suitable a set of hyperparameters are by evaluating the resulting model performance on unseen data. More specifically, it consists in shuffling and splitting the available data into batches, say N; making use of $N-1$ batches to define the model and the last one to assess its performance; the process is repeated N times rotating the batches and yielding N performance measures, which could be combined through averaging for instance. By following this algorithm, the most suitable hyperparameter value could be found from a predefined list of possible values. An extreme version of CV is leave-one-out cross-validation (LOOCV), whereby a d-sample dataset is divided into d batches. Finally, note that the log marginal likelihood itself could be a valid performance objective within CV (Williams and Rasmussen, 2006, §5.4.2).

2.8.2 Estimating RKHS norms

Members $f \in \mathcal{H}$ are either finite weighted sums of partially evaluated kernels $k(x, \cdot)$ or limits of sequences of such sums (see Section 2.2). Assume that f enjoys a finite expansion of N terms, then

$$\|f\|_{\mathcal{H}}^2 = \langle f, f \rangle_{\mathcal{H}} \tag{2.43}$$

$$= \left\langle \sum_{i=1}^{N} \alpha_i k(x_i, \cdot), \sum_{i=1}^{N} \alpha_i k(x_i, \cdot) \right\rangle_{\mathcal{H}} \tag{2.44}$$

$$= \alpha^\top K_{XX} \alpha \tag{2.45}$$

where (2.43) is the RKHS norm definition, (2.44) is the inner-product definition, and (2.45) follows from the inner-product linearity and from the reproducing property of kernels (see (2.9)). As a result, the norm $\|f\|_{\mathcal{H}}$ can be exactly and easily computed as long as we have at hand the weights α that define f.

In practice, it is hard to imagine a situation where the coefficients α would be known for a given physical system. Suppose however that we have a dataset at hand $\{(x_i, f_{x_i})\}_{i=1}^n$, where $f_{x_i} = f(x_i)$. The inputs x_i need to be pairwise-distinct. Assume that k is SPD and consider the function $s_n(x) := \sum_{i=1}^n \alpha_i k(x_i, x)$, $\alpha = K_{XX}^{-1} f_X$, which reproduces our dataset at every input, i.e., $s_n(x_i) = f_{x_i}$. According to the well-known optimal recovery property (Iske, 2018, §8.3), s_n not only interpolates the dataset, but also attains a minimum RKHS norm while doing so (Proposition 5) and, moreover, $\|s_n\|_{\mathcal{H}} \leq \|f\|_{\mathcal{H}}$ for any number of samples n. In view of the quadratic form (2.45), it is evident that for any new pair $(x_{n+1}, f_{x_{n+1}})$ added to the dataset, $\|s_{n+1}\|_{\mathcal{H}} \geq \|s_n\|_{\mathcal{H}}$ holds. One can finally show that if samples are acquired in such a way to fill the domain, then $\|f\|_{\mathcal{H}}$ is the least upper bound of the $\|s_n\|_{\mathcal{H}}$ sequence and, from the monotone convergence theorem, $\|s_n\|_{\mathcal{H}} \to \|f\|_{\mathcal{H}}$ is guaranteed.

The discussion above showed that the RKHS norm $\|f\|_{\mathcal{H}}$ can indeed be estimated from below from samples $\{(x_i, f_{x_i})\}_{i=1}^n$. This is done by simply evaluating the norm of the MNI interpolant for which we know the weights α

$$\|s_n\|_{\mathcal{H}}^2 = \alpha^\top K_{XX} \alpha = f_X^\top K_{XX}^{-1} f_X \tag{2.46}$$

The more samples we have, the closer the quadratic form will be from the target value $\|f\|_{\mathcal{H}}$. Consider the example below for observations on how the estimation process unfolds in a practical scenario.

Let $k : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ be the squared-exponential with lengthscale $\ell = 0.5$. A member $f \in \mathcal{H}$ of its RKHS is shown in Figure 2.14 (left), being composed of 100 partially evaluated kernels whose centers and weights were randomly generated. The exact $\|f\|_{\mathcal{H}}$ value was computed through (2.45), yielding $\|f\|_{\mathcal{H}} = 11.24$. Next,

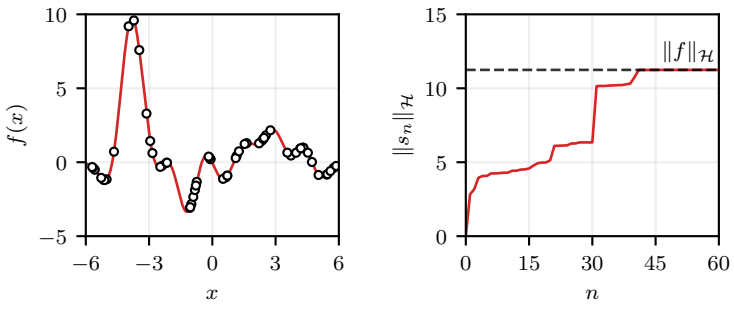

Figure 2.14: Estimating the RKHS norm of a function (left) from samples. The norm value of the MNI interpolant for an increasing number of samples (right) is shown to gradually approach the RKHS norm of the ground-truth.

the same quantity was estimated through data simply drawn randomly from the domain, following a uniform distribution. The associated $\|s_n\|_{\mathcal{H}}$ from $n = 1$ to 60 is shown in Figure 2.14 (right). At around $n = 42$, the lower estimate is seen to have essentially reached the $\|f\|_{\mathcal{H}}$ level, capturing all the complexity there was to capture. Lastly, we selected a subset of only six of the previously samples points, those closer to some of the function peaks, and computed the associated interpolant norm. The value of 10.02 was obtained, i.e., 89% of $\|f\|_{\mathcal{H}}$, illustrating that a few well-located samples can already offer us a reasonable measure of the unknown ground-truth RKHS norm.

Up to this point, only noiseless samples were consider. Suppose now the data come in the form $\{(x_i, y_i)\}_{i=1}^n$ with x_i pairwise distinct and $y_i = f(x_i) + \delta_i$. Let the additive noise be bounded by some known value $\bar{\delta} \geq |\delta_i|$ for all i. Since we do not have access to the evaluations of f anymore, $\|s_n\|_{\mathcal{H}}$ cannot be computed. If we naively interpolate the data with the model $\tilde{s}_n(x) := \sum_{i=1}^n \alpha_i k(x_i, x)$, $\alpha = K_{XX}^{-1} y$, the resulting norm $\|\tilde{s}_n\|_{\mathcal{H}}^2 = y^\top K_{XX}^{-1} y$ can be either larger or smaller than its noise-free counterpart. The mismatch between the two will depend on how each δ_i will disturb the samples, and its maximum effects can be exactly computed.

Proposition 15. Let $\{(x_i, y_i)\}_{i=1}^n$ be such that x_i are pairwise distinct and $y_i = f(x_i) + \delta_i$ with $\bar{\delta} \geq |\delta_i|$ for all i. Let s_n and $\tilde{s}_n$ be respectively the models interpolating the noise-free f_X and the noisy y values of f, i.e., $s_n(x) = \sum_{i=1}^n \alpha_i k(x_i, x)$, $\alpha = K_{XX}^{-1} f_X$ and $\tilde{s}_n(x) = \sum_{i=1}^n \alpha_i k(x_i, x)$, $\alpha = K_{XX}^{-1} y$. It then holds that

$$\nabla \leq \|\tilde{s}_n\|_{\mathcal{H}}^2 - \|s_n\|_{\mathcal{H}}^2 \leq \Delta \tag{2.47}$$

where Δ and ∇ denote respectively the maximum and minimum of $-\delta^\top K_{XX}^{-1} \delta + 2y^\top K_{XX}^{-1} \delta$ over δ, subject to $|\delta| \leq \bar{\delta}$.

Calculating Δ amounts to solving a convex optimization problem since it is the maximum of a strictly concave function, whereas ∇ is not as simple.

2.8.3 Auxiliary definitions

Recall that $n_1, n_2, \ldots, n_d$ are the number of outputs available at the input locations $x_1, x_2, \ldots, x_d$. Let Λ be the matrix of size $(\sum_i n_i) \times d$ defined as

$$\Lambda := \begin{bmatrix} \mathbf{1}_{n_1} & \mathbf{0}_{n_1} & \mathbf{0}_{n_1} & \cdots & \mathbf{0}_{n_1} \\ \mathbf{0}_{n_2} & \mathbf{1}_{n_2} & \mathbf{0}_{n_2} & \cdots & \mathbf{0}_{n_2} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathbf{0}_{n_d} & \mathbf{0}_{n_d} & \mathbf{0}_{n_d} & \cdots & \mathbf{1}_{n_d} \end{bmatrix} \tag{2.48}$$

where $\mathbf{1}_{n_i}$ and $\mathbf{0}_{n_i}$ are respectively column vectors of ones and zeros of size n_i. If only a single output is available at every input, Λ simplifies to an identity matrix.

The column vector f_X is made of function evaluations $f(x_i)$, which are repeated whenever multiple outputs are available at the same input location x_i. More concretely, $f_X := \Lambda \begin{bmatrix} f(x_1) & \dots & f(x_d) \end{bmatrix}^\top$.

2.8.4 Proofs

In this section we provide proofs for all the propositions found throughout the chapter that were left unproven.

Proof of Proposition 7: It follows from the objective being linear and only sensitive to c_x and from reformulating (2.21). More concretely, we use the matrix inversion lemma and the definition of $P_X(x)$ to re-write the complexity constraint as

$$\begin{bmatrix} c \\ c_x \end{bmatrix}^\top \begin{bmatrix} K_{XX} & K_{Xx} \\ K_{xX} & k(x,x) \end{bmatrix}^{-1} \begin{bmatrix} c \\ c_x \end{bmatrix} \leq \Gamma^2 \tag{2.49a}$$

$$\Leftrightarrow \; c^\top K_{XX}^{-1} c + P_X^{-2}(x) \left(c^\top K_{XX}^{-1} K_{Xx} - c_x \right)^2 \leq \Gamma^2 \tag{2.49b}$$

Note that $P_X^{-2}(x) > 0$ since $P_X(x) > 0$ for any x not in X (Karvonen, 2022). As a result, we see that (2.49b) depends quadratically on c_x. Therefore, for any feasible (c, c_x) such that (2.49a) is inactive, there exists $\tilde{c}_x := c_x + \varepsilon, \varepsilon > 0$ such that $(c, \tilde{c}_x)$ attains a higher objective while satisfying the constraints. □

Proof of Proposition 8: Follows from $\mathrm{C}(x_i) \geq \mathrm{B}(x_i)$, $\mathrm{C}(x_i) \leq y_{i,j} + \bar{\delta}$ and $\mathrm{B}(x_i) \geq y_{i,j} - \bar{\delta}$ for any $i = 1, \dots, d$ and any $j = 1, \dots, n_i$. □

Proof of Proposition 9: Denote by $\mathbb{P}1_1$ the problem solved with D_1 and decision variables $\begin{bmatrix} c & c_x \end{bmatrix}$. Similarly, $\mathbb{P}1_2$ is associated with the dataset D_2 and the decision variables $\begin{bmatrix} c & c_x & c_z \end{bmatrix}$, where c_z are due to the additional input in D_2. Since D_2 contains all members of D_1, the ∞-norm constraint of $\mathbb{P}1_2$ can be recast as that of $\mathbb{P}1_1$ and an additional constraint for c_z and the new outputs. Let $\mathbb{X} := X \cup \{x\}$, $\bar{c} := \begin{bmatrix} c^\top c_x \end{bmatrix}^\top$ and $z := x_{d+1}$ be shorthand variables to ease notation. The complexity constraint of $\mathbb{P}1_2$ is then

$$\begin{bmatrix}\bar{c}\\ c_z\end{bmatrix}^\top \begin{bmatrix}K_{\mathbb{X}\mathbb{X}} & K_{\mathbb{X}z}\\ K_{z\mathbb{X}} & k(z,z)\end{bmatrix}^{-1} \begin{bmatrix}\bar{c}\\ c_z\end{bmatrix} \leq \Gamma^2 \quad (2.50a)$$

$$\overset{(i)}{\Leftrightarrow} \bar{c}^\top K_{\mathbb{X}\mathbb{X}}^{-1}\bar{c} + P_{\mathbb{X}}^{-2}(z) \left\| \begin{bmatrix}K_{\mathbb{X}\mathbb{X}}^{-1}K_{\mathbb{X}z}\\ -1\end{bmatrix} \begin{bmatrix}\bar{c}\\ c_z\end{bmatrix} \right\|_2^2 \leq \Gamma^2 \quad (2.50b)$$

$$\overset{(ii)}{\Leftrightarrow} \begin{bmatrix}c\\ c_x\end{bmatrix}^\top \begin{bmatrix}K_{XX} & K_{Xx}\\ K_{xX} & k(x,x)\end{bmatrix}^{-1} \begin{bmatrix}c\\ c_x\end{bmatrix} + P_{\mathbb{X}}^{-2}(z) \left(\bar{c}^\top K_{\mathbb{X}\mathbb{X}}^{-1}K_{\mathbb{X}z} - c_z\right)^2 \leq \Gamma^2 \quad (2.50c)$$

where the matrix identity (A.23) was used in (i) along with $P_{\mathbb{X}}^2(z) = k(z,z) - K_{z\mathbb{X}}K_{\mathbb{X}\mathbb{X}}^{-1}K_{\mathbb{X}z}$. In (ii), the definitions of $\bar{c}$ and $\mathbb{X}$ were used. Thanks to $P_{\mathbb{X}}(z) \geq 0, \forall z$ (Karvonen, 2022) and the quadratic term multiplying it, we conclude that for any choice of the decision variable c_z, (2.50c) is a tightened version of the complexity constraint of $\mathbb{P}1_1$. As a result, the maximum of $\mathbb{P}1_2$ is lower or equal than that of $\mathbb{P}1_1$. □

Proof of Proposition 10: Consider the case $x \notin X$. Let $z := \begin{bmatrix}c^\top & c_x\end{bmatrix}^\top$, $a := \begin{bmatrix}\mathbf{0}^\top & 1\end{bmatrix}^\top$, $A := \begin{bmatrix}\mathbf{I} & \mathbf{0}\end{bmatrix}$. The Lagrangian of $\mathbb{P}1$ is

$$\mathcal{L}(z,\lambda,\beta,\gamma) = a^\top z - \lambda(z^\top K_{\mathbb{X}\mathbb{X}}^{-1}z - \Gamma^2) - \beta^\top(\Lambda A z - \mathsf{y} - \bar{\delta}\mathbf{1}) - \gamma^\top(\mathsf{y} - \Lambda A z - \bar{\delta}\mathbf{1}) \quad (2.51)$$

where $K_{\mathbb{X}\mathbb{X}}$ denotes the kernel matrix evaluated at $X \cup \{x\}$. Suppose $\lambda > 0$. Computing $\nabla_z\mathcal{L}(z^\star) = 0$ leads to

$$z^\star = -\frac{1}{2\lambda}K_{\mathbb{X}\mathbb{X}}\left(A^\top\Lambda^\top(\beta-\gamma) - a\right).$$

Note that β and γ cannot be simultaneously zero element-wise. Defining $\nu := \beta - \gamma$ and substituting $z^\star$ into (2.51) gives the dual objective

$$\begin{aligned} g(\lambda,\nu) &= \frac{1}{4\lambda}\nu^\top\Lambda A K_{\mathbb{X}\mathbb{X}}A^\top\Lambda^\top\nu + \left(\mathsf{y} - \frac{1}{2\lambda}\Lambda A K_{\mathbb{X}\mathbb{X}}a\right)^\top\nu \\ &\quad + \bar{\delta}\|\nu\|_1 + \frac{1}{4\lambda}a^\top K_{\mathbb{X}\mathbb{X}}a + \lambda\Gamma^2 \qquad (2.52)\\ &= \frac{1}{4\lambda}\nu^\top\Lambda K_{XX}\Lambda^\top\nu + \left(\mathsf{y} - \frac{1}{2\lambda}\Lambda K_{Xx}\right)^\top\nu \\ &\quad + \bar{\delta}\|\nu\|_1 + \frac{1}{4\lambda}k(x,x) + \lambda\Gamma^2 \qquad (2.53)\end{aligned}$$

where in the second equality the matrix $K_{\mathbb{X}\mathbb{X}}$ was expanded and the resulting terms were reorganized. Since $\beta, \gamma \in \mathbb{R}^{\tilde{d}}_{\geq 0}$ and $\nu = \beta - \gamma$ then ν is unconstrained.

Now if $\lambda = 0$, the Lagrangian (2.51) simplifies to $\mathcal{L}(z,\nu) = (a - A^\top\Lambda^\top\nu)^\top z + \nu^\top\mathsf{y} +$

$\bar{\delta}\|\nu\|_1$, which is linear in z. Its supremum w.r.t. z is only finite if $a = A^\top \Lambda^\top \nu$. Recalling the definitions of a, A and Λ, one can see that $\nexists \nu$ that could satisfy the latter condition. Therefore, $\lambda = 0 \implies \sup_z \mathcal{L}(z, \lambda, \nu) = +\infty$, meaning that the dual problem is infeasible. As a conclusion, the Lagrangian dual of $\mathbb{P}1$ in (**??**) is precisely $\mathbb{D}1$ in (2.26).

Next, consider the case $x \in X$, $x = x_i$. The objective of $\mathbb{P}1'$ can be written as $a^\top c$ with $a_i = 1$ and $a_n = 0, n \neq i$. When deriving its Lagrangian, one obtains again (2.51) with the simplifications: $z \leftarrow c$, $K_{\mathbb{X}\mathbb{X}} \leftarrow K_{XX}$ and $A \leftarrow \mathbf{I}$. We proceed by analyzing the two scenarios for λ as before. If $\lambda > 0$, the previous derivations apply, leading to the same the quadratic-over-linear objective (2.53). However, if $\lambda = 0$, the Lagrangian becomes $\mathcal{L}(z, \nu) = (a - \Lambda^\top \nu)^\top z + \nu^\top \mathsf{y} + \bar{\delta}\|\nu\|_1$, whose supremum w.r.t. z is only finite if $a = \Lambda^\top \nu$. In contrast with the previous paragraph, this condition now can be satisfied. It is equivalent to $\nu_{i,1} + \cdots + \nu_{i,n_i} = 1$, where the variables are all the multipliers associated with the i-th input location x_i. The resulting expression can be minimized analytically, yielding the minimum $\min_j y_{i,j} + \bar{\delta}$, i.e., the smallest output available at x_i augmented by the noise bound. Finally, we conclude that the dual objective for $\mathbb{P}1'$ is

$$g(\lambda, \nu) = \begin{cases} (2.53), & \text{if } \lambda > 0 \\ \min_j y_{i,j} + \bar{\delta}, & \text{if } \lambda = 0 \end{cases} \tag{2.54}$$

As a last observation, a dual problem can also be derived for (**??**), calculating the lower part of the envelope. The formulation is analogous to (2.26), assuming the form

$$\max_{\nu \in \mathbb{R}^{\bar{d}}, \lambda > 0} \; -\frac{1}{4\lambda} \nu^\top \Lambda K_{XX} \Lambda^\top \nu - \left(\mathsf{y} + \frac{1}{2\lambda} \Lambda K_{Xx} \right)^\top \nu - \bar{\delta}\|\nu\|_1 - \frac{1}{4\lambda} k(x, x) - \lambda \Gamma^2 \tag{2.55}$$

Note that these are distinct objectives, not merely opposites. Therefore, two problems have to be solved to fully quantify the ground-truth uncertainty. □

Proof of Proposition 11: Consider the primal problem $\mathbb{P}1$ and select $c = f_X^\star$ and $c_x = f^\star(x)$. Let $\mathbb{X} := X \cup \{x\}$ and $K_{\mathbb{X}\mathbb{X}}$ denote the kernel matrix associated with $\mathbb{X}$. Thanks to the optimal recovery property (Wendland, 2004, Theorem 13.2), $\begin{bmatrix} c^\top & c_x \end{bmatrix} K_{\mathbb{X}\mathbb{X}} \begin{bmatrix} c^\top & c_x \end{bmatrix}^\top \leq \|f^\star\|_{\mathcal{H}}^2$, which in turn is strictly smaller than Γ^2 by assumption. Also, $\|\Lambda c - \mathsf{y}\|_\infty = \|\Lambda f_X^\star - \mathsf{y}\|_\infty = \left\| \begin{bmatrix} \delta_{1,1} & \dots & \delta_{2,1} & \dots \end{bmatrix}^\top \right\|_\infty < \bar{\delta}$. Therefore, the ground-truth values constitute a feasible solution that lies in the interior of the primal problem feasible set. As a result, Slater's condition is met and, since the primal is convex, there is no duality gap. □

Proof of Proposition 12: First note that

$$\begin{aligned}
\|f-s\|_{\mathcal{H}}^2 &= \langle f-s, f-s\rangle_{\mathcal{H}} && (2.56)\\
&= \langle f,f\rangle_{\mathcal{H}} - 2\langle f,s\rangle_{\mathcal{H}} + \langle s,s\rangle_{\mathcal{H}} && (2.57)\\
&= \|f\|_{\mathcal{H}}^2 - 2\langle f,s\rangle_{\mathcal{H}} + \|s\|_{\mathcal{H}}^2 && (2.58)\\
&= \|f\|_{\mathcal{H}}^2 - 2\|s\|_{\mathcal{H}}^2 + \|s\|_{\mathcal{H}}^2 && (2.59)\\
&\leq \Gamma^2 - \|s\|_{\mathcal{H}}^2 && (2.60)
\end{aligned}$$

where (2.59) was established thanks to

$$\begin{aligned}
\langle f,s\rangle_{\mathcal{H}} &= \left\langle f, \sum_{i=1}^{d} \alpha_i k(\cdot,x_i)\right\rangle_{\mathcal{H}} && (2.61)\\
&= \sum_{i=1}^{d} \alpha_i f(x_i) && (2.62)\\
&= \sum_{i=1}^{d} \alpha_i s(x_i) && (2.63)\\
&= \sum_{i=1}^{d} \alpha_i \left(\sum_{j=1}^{d} \alpha_j k(x_j,x_i)\right) && (2.64)\\
&= \|s\|_{\mathcal{H}}^2 && (2.65)
\end{aligned}$$

and recalling that $s(x_i) = f(x_i)$ $i = 1, \dots d$ given the noise-free setting. Combining (2.60) with the inequality $|s(x) - f(x)| \leq P_X(x) \|f - s\|_{\mathcal{H}}$ (Wendland, 2004, §11.5) completes the proof. □

Proof of Proposition 13: For any given $s(x) = \alpha^\top K_{Xx}$, we have

$$\begin{aligned}
|f^\star(x) - s(x)| &= |f^\star(x) - \tilde{s}(x) + \tilde{s}(x) - s(x)|\\
&\leq |f^\star(x) - (f_X^\star + \delta_X)K_{XX}^{-1}K_{Xx}| + |\tilde{s}(x) - s(x)| && (2.66)\\
&\leq |f^\star(x) - \bar{s}(x)| + \bar{\delta}\left\|K_{XX}^{-1}K_{Xx}\right\|_1 + |\tilde{s}(x) - s(x)| && (2.67)\\
&\leq P(x)\sqrt{\Gamma^2 - \|\bar{s}\|_{\mathcal{H}}^2} + \bar{\delta}\left\|K_{XX}^{-1}K_{Xx}\right\|_1 + |\tilde{s}(x) - s(x)| && (2.68)\\
&\leq P(x)\sqrt{\Gamma^2 + \Delta - \|\tilde{s}\|_{\mathcal{H}}^2} + \bar{\delta}\left\|K_{XX}^{-1}K_{Xx}\right\|_1 + |\tilde{s}(x) - s(x)| && (2.69)
\end{aligned}$$

with $f_X^\star$ being the vector of true function values at the sample locations in X and δ_X the vector of additive measurement noise for the samples y. (2.66) follows from the triangle inequality and the additive noise property of y. Using the triangle inequality again, we arrive at (2.67), where $\bar{s}$ denotes the noise-free interpolant of $f_X^\star$. The noise-free interpolation error bound gives the estimation in the first term of (2.68), while (2.69) follows from (Maddalena et al., 2021a, Lemma 1), with

$\Delta = \max_{\|\delta\|_\infty \leq \bar{\delta}} (-\delta^\top K_{XX}^{-1} \delta + 2y^\top K_{XX}^{-1} \delta)$. A standard Lagrangian dualization procedure leads to the dual formulation

$$\min_{\nu \in \mathbb{R}^d} \frac{1}{4} \nu^\top K_{XX} \nu + \nu^\top y + \bar{\delta} \|\nu\|_1 + y^\top K_{XX}^{-1} y \tag{2.70}$$

for Δ. Notice that the last term in (2.70) is constant and the same as the squared interpolant norm $\|\tilde{s}\|_{\mathcal{H}}^2$. Therefore, these terms cancel in (2.69) and we are left with

$$|f^\star(x) - s(x)| \leq P(x) \sqrt{\Gamma^2 + \tilde{\Delta}} + \bar{\delta} \left\| K_{XX}^{-1} K_{Xx} \right\|_1 + |\tilde{s}(x) - s(x)| \tag{2.71}$$

where $\tilde{\Delta}$ represents (2.70) without the constant term. □

Proof of Proposition 14: From construction, we know that $\mathcal{X}_t(x_0, u_0^\star, \ldots, u_{t-1}^\star) \ni f(\ldots f(f(x_0, u_0^\star), \ldots), u_{t-1}^\star)$ for any t. If the previous KPC iteration was feasible, then $\mathcal{X}_{t+1}(x_{-1}, u_{-1}, u_0^\star, \ldots, u_{t-1}^\star)$ also contains the same point, where x_{-1} and u_{-1} are past closed-loop data. Considering a total of $N-1$ previous consecutive feasible KPC iterations yields a total of N set conditions for $f(x_0, u_0^\star)$, $N-1$ set conditions for $f(x_0, u_0^\star, u_1^\star)$, …, and 1 set condition for $f(\ldots f(f(x_0, u_0^\star), u_1^\star), \ldots, u_{N-1}^\star)$. At any time $t = 1, \ldots, N$, the true system state is therefore contained in the intersection of the associated sets and, hence, enforcing the relaxed constraints suffices to enforce constraint satisfaction. □

Proof of Proposition 15: It follows from expanding $\|s_n\|_{\mathcal{H}}^2$ and $\|\tilde{s}_n\|_{\mathcal{H}}^2$. □

3 Building temperature control through Gaussian process and model predictive control

3.1 Introduction

Buildings are responsible for a large share of the global CO_2 emissions, more concretely, for around 37% of them when taking into account the estimated emissions associated with producing their materials (United Nations Environment Programme, 2022, §3.2). Having in place energy-efficient heating and cooling systems is often highlighted as a key requirement for significantly lowering their environmental impact and marching towards a greener future (IEA, 2022). This in turn could either be achieved by structurally changing the built environment, for instance, improving thermal isolation between indoor spaces and the outdoor space, or by making a more intelligent use of the energy at hand, guaranteeing the occupants' comfort while minimizing metrics such as thermal wasting. Control engineers are specialists in carrying out the latter task.

Besides its clear economical and political pertinence, the topic of raising the efficiency of buildings has also had a significant academic relevance, particularly for the automatic control community. One idea that is often explored is that well tuned, specialized control algorithms can deliver better performance compared to simplistic strategies commonly deployed in the field (Stluka et al., 2018). These specialized algorithms often rely on solving optimization problems online, and can exploit forecasts of important quantities such as the weather and the building occupancy schedule (Oldewurtel et al., 2012). Arguably the most common framework used by researchers to tackle building climate control is model predictive control thanks to its flexibility and interpretability. The reader is referred to Drgoňa et al. (2020) for quite a comprehensive survey on the matter. In Sturzenegger et al. (2015) for instance, the authors report in detail the results of using MPC to operate an occupied 6-story Swiss office building over multiple

months. Moreover, the study concluded that the energy savings were not sufficient to justify the engineering effort required to deploy MPC in everyday building projects. In fact, the modeling stage is frequently pointed out by many studies as being the most burdensome one, preventing the wider adoption of this technique in the field (Drgoňa et al., 2020; Bünning et al., 2020).

Eliciting knowledge from domain experts or using architectural data to develop linear thermal models for buildings was proven to be an effective approach in many distinct scenarios (see e.g. the experimental works Váňa et al. (2014); Bünning et al. (2022)). As an alternative, it is also possible to employ black-box linear structures, which requires less access to building-specific information (Fabietti et al., 2016, 2018). A more recent trend is that of experimenting with contemporary machine learning models and methods, whose promise is to further ease the modeling step while being flexible enough to capture intricate non-linear relationships present in the data and deliver superior performance. Some variants of reinforcement learning (RL) for instance are model-free and can optimize their actions over time by only interacting with the environment (Du et al., 2021). Nevertheless, since RL typically converges slowly and long-lasting experiments with low performance on occupied real buildings are not desirable, model-free RL designers make use of simulation environments to warm-start their RL agents, thus not escaping the need of high-fidelity models (Di Natale et al., 2022a). Going from historical data directly to on-line decision making is possible through data-enabled predictive control (DeePC) (Coulson et al., 2019), which bypasses the modeling step completely and directly incorporates sensor information into a special MPC formulation. This paradigm was used in Lian et al. (2021) to regulate the indoor temperature of a university lecture hall over several weeks. DeePC was however developed to handle linear relationships and more flexible tools are necessary to model more complex dynamics, e.g. the formalism of Gaussian processes (GPs).

Gaussian processes are a well-established, principled way of crafting possibly non-linear statistical models. In contrast with the techniques listed in the previous paragraph, GPs yield not only nominal predictions, but also a measure of confidence around them that can be taken into account during decision making. Practical applications of this tool can be easily found in the area of robotics, computer simulation, global optimization, among others. In the building community, they have been used to calibrate digital twins (Chakrabarty et al., 2021), forecast temperature evolution (Gray and Schmidt, 2016), detect faults in air-handling units (Van Every et al., 2017), and create abstract models for how consumers react to demand-adjustment signals (Nghiem and Jones, 2017). All of the previously cited works consisted of off-line analyses on real data or purely of simulations and, to the best of our knowledge, the use of Gaussian processes to carry out

closed-loop control of a building has not yet been reported in the literature. The experimental investigation presented in this chapter aims at filling that gap.

We start by reviewing the basic theory that is necessary to understand how GPs were applied in this particular project. Next, the building under study, a hospital surgery center, is presented along with its industrial cooling system and the sensor networks deployed on-site. Data collection and model training are covered, followed by our MPC formulation where the GP dynamical models were incorporated. Finally, the obtained experimental results are discussed, and the chosen approach is compared with a number of alternative techniques in two criteria, average thermal comfort and incurred electrical energy consumption.

3.2 A brief overview of Gaussian process

Gaussian processes lie in the class of non-parametric Bayesian models. Whereas one can easily and intuitively explain GPs in a regression setting by making use of a few normal distribution properties, they also constitute a large and still active research field on their own. One classical GP reference is the acclaimed book Williams and Rasmussen (2006), which covers most of the basic concepts. Complementary views on the subject are offered by the texts Gramacy (2020) and Kanagawa et al. (2018). Two prominent lines of research that have remained active during the last decades are scaling Gaussian processes to large datasets (Ferrari-Trecate et al., 1998; Quinonero-Candela and Rasmussen, 2005; Yang et al., 2012; Bauer et al., 2016; Bui et al., 2017b; Lederer et al., 2021), and incorporating online data into the models without the need to retrain them from scratch (Csató and Opper, 2002; Bui et al., 2017a; Maddox et al., 2021). The control community, among others, has also been making use of Gaussian processes for some years (see Liu et al. (2018) for an informative tutorial). Early investigations include Kocijan et al. (2003); Hansen et al. (2005) and among more recent ones one has Diwale et al. (2014); Hewing et al. (2019); Lederer et al. (2022); Khosravi et al. (2022). In the cited applied works, GPs are preferred over alternative modeling strategies mainly thanks to their uncertainty quantification feature.

Our goal here is not to thoroughly discuss the topic of Gaussian processes, but simply to touch on the basics and understand how they were used to tackle this particular building temperature control problem.

Assume one wants to model a given scalar phenomenon $f(x)$, which is known to depend on a set of variables x, and is given observations of the form $\{(\mathbf{x}_i, \mathsf{y}_i)\}_{i=1}^{d}$ with $\mathsf{y}_i = f(\mathbf{x}_i) + \varepsilon$, for some random ε. For convenience, denote by $\mathbf{X}$ the collection

of all inputs $\mathbf{x}_i$, by $\mathbf{y}$ the collection of all outputs y_i and by $\mathbf{f}$ the (unobserved) collection of all noise-free function evaluations at the inputs $f(\mathbf{x}_i)$.

The Gaussian process approach to crafting models consists in placing a multivariate normal *prior* distribution over $\mathbf{f}$, that is

$$p(\mathbf{f}|\mathbf{X}) = \mathcal{N}(m_{\mathbf{X}}, K_{\mathbf{XX}}) \tag{3.1}$$

where $m_{\mathbf{X}}$ is a vector and $K_{\mathbf{XX}}$ is a positive definite matrix, both of appropriate dimensions. The vector $m_{\mathbf{X}}$ has $m(\mathbf{x}_i)$ at its ith component, for some mean function m, and the matrix $K_{\mathbf{XX}}$ has $k(\mathbf{x}_i, \mathbf{x}_j)$ at its ith row and jth column, for some positive definite kernel function k.

Next, an assumption is made on the nature of ε, the most common one being $\varepsilon \sim \mathcal{N}(0, \sigma^2)$. This zero-mean, fixed variance assumption on the noise distribution guarantees that the subsequent steps all have closed-form solutions thanks to Gaussian distributions being closed under the operations that need to be performed[1]. Thanks to the measurement model $\mathsf{y}_i = f(\mathbf{x}_i) + \varepsilon$, we then have the distribution $p(\mathbf{y}|\mathbf{f}, \mathbf{X}) = \mathcal{N}(\mathbf{f}, \sigma^2 I)$, known as the *likelihood*[2].

By using Bayes theorem, it is possible to show that the *posterior* distribution of a new value f_* located at a new input x_* is (Williams and Rasmussen, 2006, §2.2)

$$\begin{aligned} p(f_*|x_*, \mathbf{y}, \mathbf{X}) = \mathcal{N}(&m(x_*) + K_{x_*\mathbf{X}}(K_{\mathbf{XX}} + \sigma^2 I)^{-1}(y - m_{\mathbf{X}}), \\ &k(x_*, x_*) + K_{x_*\mathbf{X}}(K_{\mathbf{XX}} + \sigma^2 I)^{-1} K_{\mathbf{X}x_*}) \end{aligned} \tag{3.2}$$

where $K_{\mathbf{X}x_*}$ is a column vector with $k(\mathbf{x}_i, x_*)$ at its ith position and $K_{x_*\mathbf{X}}$ denotes its transpose. Finally, (3.2) can be used to make predictions with the model.

Most models, including the Gaussian process class, feature internal parameters that need to be adjusted, fit, to the data at hand. This is crucial to achieving good performance. In (3.1) and (3.2), the constants to be calibrated are the ones internal to the mean function $m(x)$, the kernel function $k(x, x)$ and the noise strength σ^2. Denote the ensemble of all those constants by Θ. One widely adopted approach to solving this calibration problem is maximizing the model marginal likelihood, that is, the quantity $p(\mathbf{y}|\mathbf{X})$. Intuitively, this can be thought of as finding the model that would lead to the available dataset having come from it with the highest likelihood. As it turns out, it is more convenient not to directly optimize $p(\mathbf{y}|\mathbf{X})$, but its logarithm instead, so we define $\mathcal{L} := \log p(\mathbf{y}|\mathbf{X})$, which

[1] The noise is said to be heteroscedastic if its variance depends on another random variable under study, e.g., if it varies across the x space. GPs under heteroscedastic noise are, in general, not analytically tractable (see for example Goldberg et al. (1997); Binois et al. (2018)).

[2] Not to be confused with the marginal likelihood, which is often used as a training objective.

can be shown to be (Williams and Rasmussen, 2006, §2.3)

$$\mathcal{L} = -\frac{1}{2}(\mathbf{y} - m_{\mathbf{X}})^{\top}(K_{\mathbf{XX}} + \sigma^2 I)^{-1}(\mathbf{y} - m_{\mathbf{X}}) - \frac{1}{2}\log\det(K_{\mathbf{XX}} + \sigma^2 I) - \frac{d}{2}\log(2\pi) \tag{3.3}$$

and solve the problem $\Theta^{\star} := \arg\max_{\Theta} \mathcal{L}(\Theta)$ to find the most suitable set of parameters for our GP model. Notice how this amounts to solving a non-convex but typically smooth optimization problem to local optimality. This procedure is widely known as maximum likelihood estimation (MLE).

In the applied sciences, users may abstract most of the above concepts and think of GPs in a more pragmatic way. They can define the (posterior) mean and variance functions respectively as

$$\mu(x) := m(x) + K_{x\mathbf{X}}(K_{\mathbf{XX}} + \sigma^2 I)^{-1}(y - m_{\mathbf{X}}) \tag{3.4}$$

$$\mathrm{var}(x) := k(x, x) + K_{x\mathbf{X}}(K_{\mathbf{XX}} + \sigma^2 I)^{-1}K_{\mathbf{X}x} \tag{3.5}$$

and simply make use of the first expression to make predictions, indeed the most likely ones, and the second as uncertainty level around the nominal point predictions. It should be underlined that there is a significant cost associated with (3.4) and (3.5), which mainly stems from the inverse of a $d \times d$ matrix. A second point that cannot be overlooked is the numerical stability of this operation: if two inputs $\mathbf{x}_i$ and $\mathbf{x}_j$ are too close to each other problems might arise despite the presence of the diagonal term $\sigma^2 I$. Both these issues have to be taken into account in practical scenarios to guarantee the feasibility of this model class.

An example of univariate Gaussian process regression is shown in Figure 3.1. The ground truth is a shifted Gramacy and Lee function $f(x) = \frac{\sin(10\pi x)}{2x} + (x-1)^4 + 1$ from which 60 points were sampled with $\sigma \sim \mathcal{N}(0, 0.15^2)$. The rational quadratic kernel was chosen to parametrize the GP covariance and the hyperparameters were

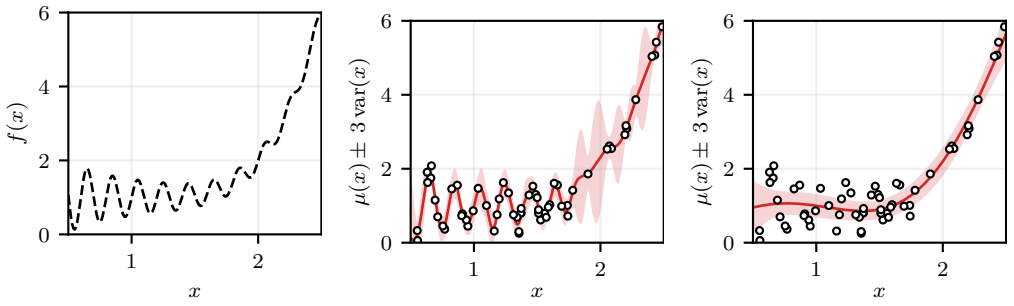

Figure 3.1: A Gaussian process regression example. The ground-truth (left), Gaussian process mean and the 3σ confidence interval for a model with log marginal likelihood $\mathcal{L} = -29.8$ (middle) and $\mathcal{L} = -48.5$ (right).

tuned through MLE. The `GPflow` package (Matthews et al., 2017) was used in the numerical experiment and the objective was optimized through a BFGS method. In the figure, two distinct GP models are depicted, which were attained by means of MLE and BFGS under the same numerical tolerance and maximum number of iterations, but with different hyperparameter initial guesses. This illustrates how arriving at a favorable GPmodel through MLE requires escaping local maxima.

Up to this point, only "static" functions have been addressed and the GP modelling dynamical systems was not touched on. If the system under study is described by the familiar equations $x_{t+1} = f(x_t, u_t) + \epsilon$, $y_t = g(x_t) + \nu$ and the unknown f is to be modelled from output data y_t only, then the problem is difficult. More precisely, even if g is assumed linear and known, finding a closed-form posterior expression for f is in general intractable and approximation schemes have to be used (e.g. variational inference or Markov-Chain Monte Carlo methods) (Turner et al., 2010). Tuning the hyperparameters in this so-called Gaussian process state-space model (GPSSM) setting is another non straightforward step (Eleftheriadis et al., 2017). The control community has been working on making GPSSM computationally more efficient (Berntorp, 2021) and even on understanding the properties of the learned dynamics (Beckers and Hirche, 2016), however an alternative approach exists that circumvents some of the aforementioned difficulties under an additional assumption. The method lies in modelling the unknown dynamics in an auto-regressive form and utilizing feature selection techniques and expert knowledge to decide on which signals and how many past samples to include, i.e., deciding what the regression vector should be. In such case, building a GP model for $x_{t+1} = f(x_t, u_t) + \epsilon$ reduces to learning a static map again as one would be assuming that full state measurement is possible. This was the direction taken in this investigation.

3.3 The building and its HVAC system

The building considered in this study was a surgery center situated in the São Julião hospital complex, in the city of Campo Grande, MS, Brazil (Figure 3.2, top). The 51 rooms that compose it are in permanent use and, for information purposes, 528 surgical procedures were carried out in it during August 2021. We were concerned with three thermal zones in its ophthalmology section: two operating rooms (ORs) and one waiting room (WR), all located on the West end of the building (Figure 3.2, bottom). Whereas the former rooms are only connected to the waiting room, the latter has a door to the rest of the surgery center. Opaque glass bricks are present in the waiting room as can be seen in the picture, allowing some natural light to enter the space; the operating rooms on the other hand do not feature them, nor do they have any windows. All spaces have exterior walls,

Figure 3.2: The facade of the Hospital São Julião surgery center, situated in Campo Grande, MS, Brazil (top). The three thermal zones considered in this study (bottom): The waiting room and the doors that lead to the two operating rooms, all in the ophthalmology section of the building, located on its West end.

but the right-hand side operating room is significantly more affected by direct solar radiation due to the disposition of the nearby trees.

A forced-air cooling plant is in place to provide the occupants with a suitable indoor climate in accordance with local regulations. A total of seven air-handling units (AHUs) collect outdoor air that is then treated and filtered before being pumped into the several indoor spaces. We had control only over three AHUs, one for each aforementioned thermal zone. A central chiller connected to an external cooling tower provides chilled water to all AHUs, which in turn feature three-way valves to control the flow of water through their cooling coils. The AHU fans are operated always at constant speed, resulting in a constant volumetric flow through the air-ducts and into the zones. As per the regulations, no air recycling is possible and all return air is directly discharged into the atmosphere. As the temperature in Campo Grande is typically high, the heating, ventilation and air-conditioning (HVAC) system was conceived to only cool the space, not having the means to provide positive thermal energy (for more details, see Section 3.3.1).

Two distinct sensor networks were deployed to monitor the HVAC plant and the

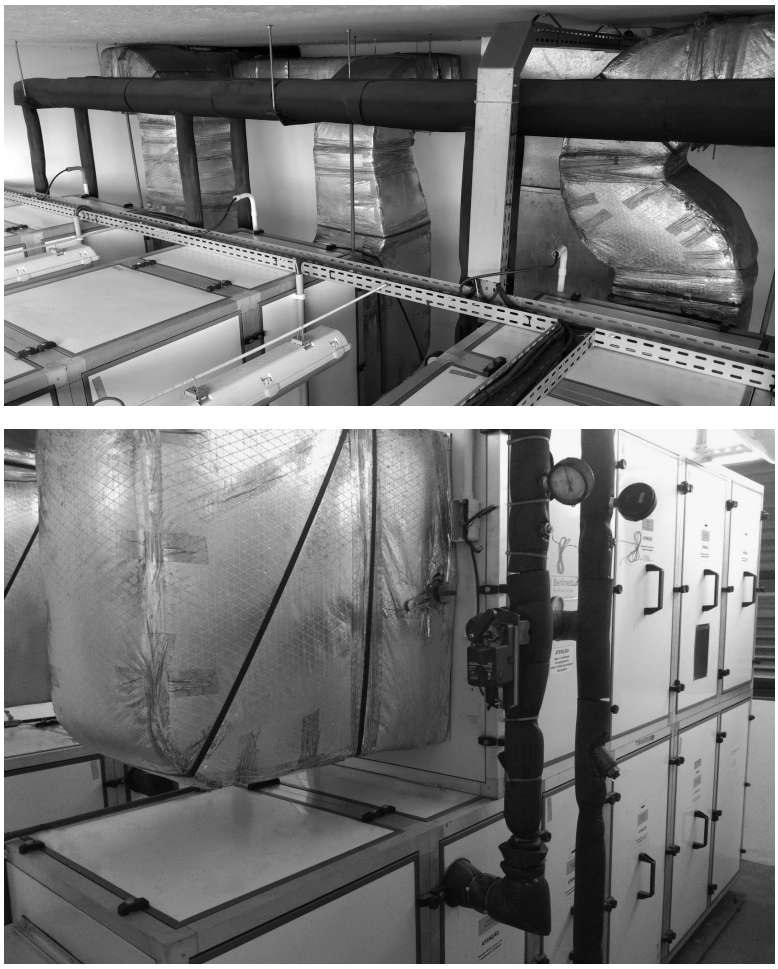

Figure 3.3: Photos of the AHU room where one can identify the air ducts (top), the supply and return water pipes (top and bottom), and one of the three-way valve servomotors that control the flow of chilled water (bottom).

indoor spaces. Firstly, we will describe the one located in the AHU room. One local controller (LCO)–a National Instruments myRIO–was attached to each air-handling unit to read all sensors and monitor the AHUs: supply and return water temperature probes, a water flow meter, an anemometer, as well as an angular position sensor. The LCOs were moreover responsible for running low-level signal processing routines and implementing control actions, i.e., acting on the three-way valve servomotor to change the chilled water flow, hence influencing the supply air temperature. Photos of the AHU room are shown in Figure 3.3. Next, in order to measure the indoor temperatures in a flexible way, a wireless network of Z-wave sensors was set up in the operating rooms and waiting room. These were equipped with external temperature probes (Dallas DS18B20) to guarantee fast and precise readings, reporting their measurements periodically to a local computer (LC) that featured a Z-wave transceiver attached to it.

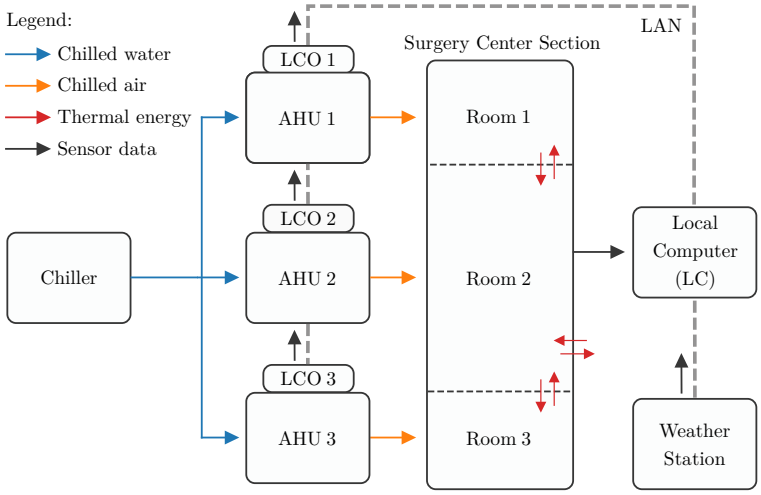

Figure 3.4: The overall system architecture, including the three AHUs and their local controllers (LCOs). Information is exchanged between the LCOs and the local computer (LC) through a local area network (LAN).

The LC was a 3 GHz, 16 GB RAM, core i7 machine installed in the waiting room and acting as the main computer platform for the project. This computer and the AHU LCOs were all connected to a local area network (LAN) to exchange information, which was done by using the UDP protocol at a rate of 1 Hz. We highlight that this same LAN was used to send control signals from the LC to the LCOs in order to modulate the AHU valves. Lastly, a weather station was deployed on site to measure the outdoor temperature and the solar radiation acting on the building with high accuracy. All signals were sampled with a period of two minutes and stored into a local time-series database, InfluxDB. A block-diagram of the complete system is depicted in Figure 3.4.

3.3.1 Analysis of the control problem

The control goal was to regulate the indoor temperature within the three zones (T_i, $i = 1, 2, 3$), keeping it below a pre-specified value $T_{\max}$ at all times. Although defining two-level temperature envelopes that have relaxed constraints at night is common for residences and offices, some employees still made use of the surgery center spaces during nighttime and, thus, the indoor temperature has to stay below $T_{\max}$ even then. Furthermore, this was to be done while minimizing the chiller energy consumption, which is a function of its coefficient of performance (COP) curve and the building thermal load.

The controlled variables are the angular positions of each AHU three-way valve (θ_i, $i = 1, 2, 3$) that regulate the flow of water across their cooling coils. Naturally, these

quantities are physically limited between a minimum $\theta_{\min}$ and a maximum value $\theta_{\max}$. Several disturbances both of internal and external nature act on the system. The measured ones include the outdoor temperature T_{out}, the solar radiation R_{sol}, and the temperature of the water supplied by the chiller to the AHUs, T_{sup}. The variables T_{out} and R_{sol} directly affect the indoor climate by heating the external walls. Also, both T_{out} and T_{sup} can be regarded as input disturbances; indeed, these quantities define the HVAC system actuation capabilities along with the valve positions θ_i. The unmeasured disturbances are the internal heat gains generated by occupants and equipment, as well as the eventual opening and closing of doors that lead to air mix among rooms. A summary of the relevant control information described here can be found in Table 3.1.

In order to operate optimally the industrial cooling system while minimizing the operational costs, we opted for designing and recursively solving a model-based finite-horizon optimal control problem. To craft dynamical models for the temperature evolution in the rooms and the AHU effects on them, we made use of the Gaussian process framework described in Section 3.2.

3.3.2 Data collection, model training and testing

Aiming at capturing different weather conditions and a reasonable number of chiller states, a representative dataset was gathered from August to November 2021. The batch consisted of 22'455 points sampled at $T_{\text{samp}} = 2$ mins and comprised closed-loop operation with PI and rule-based controllers (RBCs), as well as a variety of open-loop excitation signals such as ramps and uniformly random inputs. The latter signals were mainly tested during weekends and Holidays to avoid disturbing occupants. After examining the obtained curves, we concluded that a control period of 10 mins would be a good compromise between operating the cooling system effectively and not oversampling the temperatures – given that our model complexity grows with the size of the dataset, the latter aspect was rather important. The dataset was then downsampled by a factor of 5 times, resulting in 4'491 points, which corresponds to 748 hours.

A meticulous post-processing step was necessary next to ensure that unreliable periods were discarded. We highlight that this was not simply a matter of finding spurious outliers, but of identifying complete periods of time when the plant was not operating normally due to, for instance, a chiller fault, an unannounced maintenance in an AHU or even the manual closing of some air dampers by a technician. This process was carried out by jointly studying the obtained curves with the help of the local maintenance team. Finally, missing entries in the

Table 3.1: The main physical quantities influencing the HVAC plant and the temperature dynamics inside the rooms.

	Symbols	Description
Inputs	θ_1, θ_2, θ_3	Valve position of AHU 1, AHU 2 and AHU 3
Outputs	T_1, T_2, T_3	Temperatures within zone 1, zone 2 and zone 3
Measured disturbance	T_{sup}, T_{out}, R_{sol}	Chiller supply water temperature, outdoor temperature, solar radiation
Unmeasured disturbance	–	Internal heat gains (e.g. occupants), opening and closing of doors

Table 3.2: Delays for each feature of the Gaussian process auto-regressive models. The symbol "–" indicates what signals were neglected.

	T_1	T_2	T_3	θ_1	θ_2	θ_3	T_{sup}	T_{out}
GP1 (OR 1)	2	1	–	1	–	–	1	1
GP2 (WR)	1	2	1	–	1	–	1	1
GP3 (OR 2)	–	1	2	–	–	1	1	1

time-series were imputed using a simple linear interpolation scheme. Thanks to the reliability of the sensor networks shown in Section 3.3, only a small percentage of entries was not present. With the right set of data at hand, we proceeded to define and train one GP model for each room.

The process of designing GP models can be summarized in three steps: select suitable mean and covariance functions, define the feature vectors, and optimize the models hyperparameters. Among the many kernel maps available in the literature, we chose the anisotropic squared-exponential function, a very popular alternative due to its smoothness and expressive power. This kernel has the form

$$k_{\text{SE}}(x, x') = \sigma^2 \exp\left(-\frac{1}{2}\sum_{i=1}^{d}\left(\frac{x_i - x'_i}{\ell_i}\right)^2\right) \tag{3.6}$$

where x_i is the ith component of the feature vector x. In (3.6), σ is the so called vertical scale hyperparameter and ℓ_i are the horizontal scale (a.k.a. lengthscale) hyperparameters. However, relying solely on (3.6) can be dangerous as the resulting GP model would not extrapolate well. For this reason, a linear mean function $m(x) = Ax + b$ was also employed, causing the Gaussian process to behave linearly when far away from the training data. As for optimizing the

hyperparameters we made use of the log-marginal likelihood objective and the Broyden–Fletcher–Goldfarb–Shanno (BFGS) first-order algorithm. The development was carried out in `Python` with the aid of the `GPflow2` (Matthews et al., 2017) and the `SciPy` (Virtanen et al., 2020) packages.

To model the temperature dynamics in the rooms, an auto-regressive structure was preferred over a state-space one (see discussion in Section 3.2). In other words, future predictions of a signal depend on the current and past values of itself as well as on current and past values of other relevant quantities. Since there were three temperatures to predict, three distinct GPs were trained and, to avoid augmenting the Gaussian process with unnecessary features, we made use of domain knowledge. Rooms that are not neighbors do not directly influence each other's temperatures; similarly, changing the valve position of AHU 1 has no effect on any temperature besides T_1. Initially, all exogenous signals T_{sup}, T_{out} and R_{sol} had been included into all models to boost their prediction capabilities. Nevertheless, we later realized that R_{sol} was a significant covariate only for T_3. Field tests also unveiled a high correlation between the MPC computation times over the day and the solar radiation curve. Since having consistent rather than fluctuating solve times was a project requirement, we decided not to employ R_{sol} as a feature in any GP. The definitive set of features and signal delays is reported in Table 3.2. Finally, by using the mean functions (3.7) to evolve the temperature dynamics, we arrive at the final models

$$T_{1,t+1} = \mu_1(T_{1,t}, T_{1,t-1}, T_{2,t}, \theta_{1,t}, T_{\text{sup},t}, T_{\text{out},t}) \tag{3.7a}$$

$$T_{2,t+1} = \mu_2(T_{1,t}, T_{2,t}, T_{2,t-1}, T_{3,t}, \theta_{2,t}, T_{\text{sup},t}, T_{\text{out},t}) \tag{3.7b}$$

$$T_{3,t+1} = \mu_3(T_{2,t}, T_{3,t}, T_{3,t-1}, \theta_{3,t}, T_{\text{sup},t}, T_{\text{out},t}) \tag{3.7c}$$

Regarding the variances (3.5), those were used in a point-wise fashion to measure uncertainty. In other words, multi-step ahead predictions were performed by propagating only the mean values forward. The more complex alternative, which was discarded in this study, is to propagate also the higher-order moments and employ numerical methods to approximate the intractable integrals in the hope of arriving at more precise uncertainty estimates (Girard, 2004; McHutchon and Rasmussen, 2011).

In order to train the GP models in an efficient way, balancing the information brought by the data-set and non-parametric model size, nearby feature vectors were dropped. The Euclidean norm of the difference was chosen as the distance function. Such a dropping also improves the numerical stability of the model kernel matrix. Finally, the GPs modeling rooms 1, 2 and 3 had respectively 235, 245 and 314 points, which correspond to approximately 38, 51 and 47 hours worth

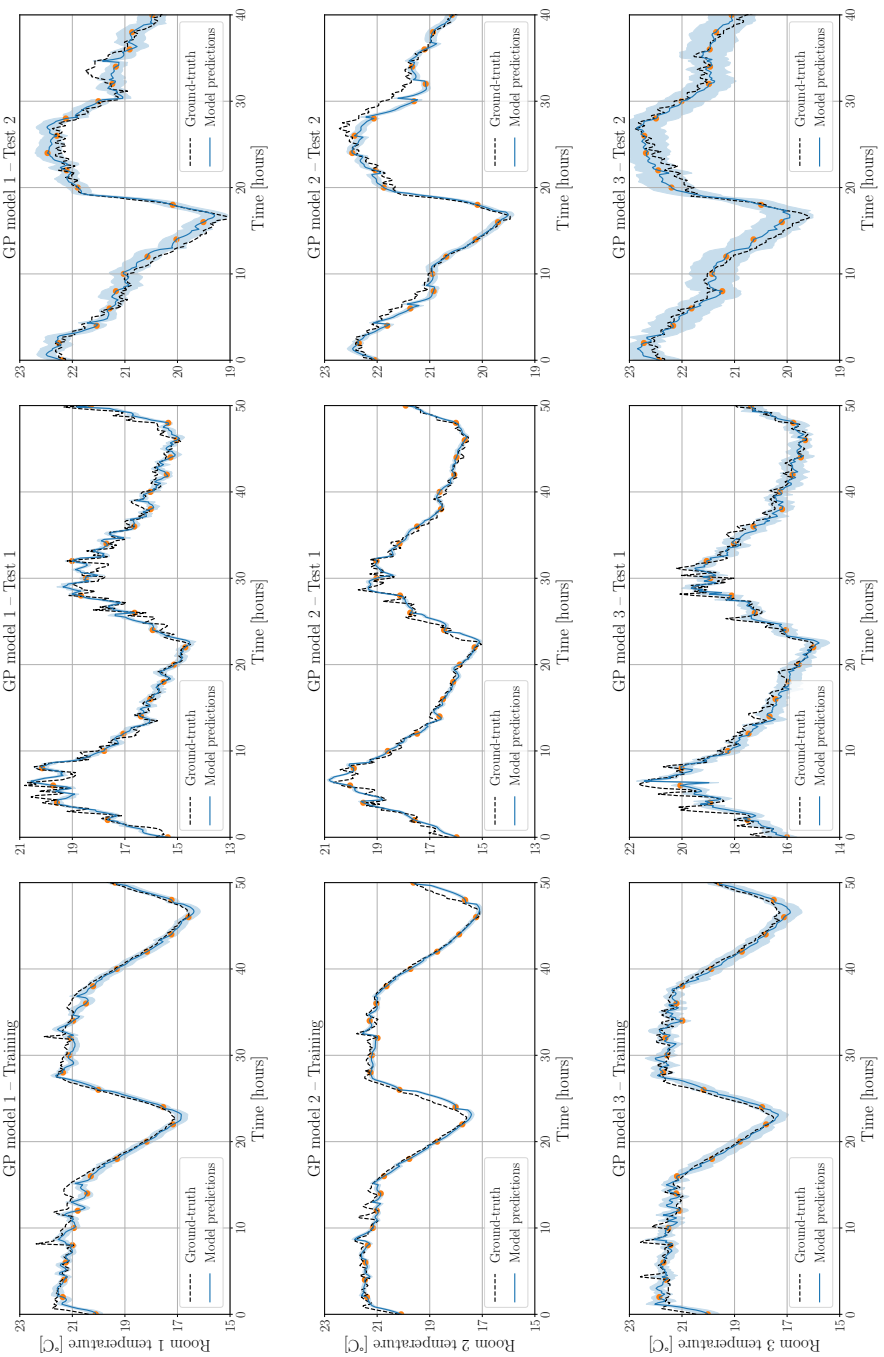

Figure 3.5: Train and test results for the Gaussian process auto-regressive models predicting the temperature evolution within rooms 1, 2 and 3. The training plots only display a portion of the training set. The GP means are depicted in solid blue and the two standard deviation interval in light blue. The real temperatures are shown in dashed black.

of data. Since these sets did not come from a single experiment, but are an informative subset of the 748 hours initially available to us, they provided enough prediction capabilities to our non-parametric models. Training the models with the aforementioned number of points took consistently less than 5 seconds each on a 2.4 GHz, core i9-9980HK machine. The obtained training and test results can be seen in Figure 3.5. We highlight that the plots show multi-step ahead predictions over a horizon of 2 hours, that is, 12 time steps, correcting for the temperature mismatch only at the orange points. Assessing the prediction quality of the models in this way was necessary as they were to be used within an MPC formulation. It is worth noting that, even though the left plots are labeled as "training results", the models incorporated only a small fraction of those features due to our dropping of nearby data-points. The central and right-side plots show the predictions over a period of 50 and 40 hours, but in completely new scenarios, never presented to the model during training.

Inspecting Figure 3.5, we see that the mean predictions mostly follow the underlying ground-truth signal during training. The disparity among the rooms is in their uncertainty bands: whereas model 1 and 2 presented moderate levels of spread, model 3 showed a fairly large one. We believe this uncertainty to stem from room 3 being the most exposed one in terms of direct solar radiation, and from R_{sol} not being a feature of its GP. We remind the reader that R_{sol} was disregarded to accelerate the real-time computations and ensure that the optimization problem was solved within the time allocated to it. By taking this larger uncertainty into account, we were able to avoid violating constraints when closing the loop with the MPC controller. The outcome of the test phase was qualitatively similar to the training results, aside from some additional performance degradation close to the high temperature peaks. Overall, we deemed the results reasonable given the challenging two-hour horizon of the prediction task.

3.3.3 Learning the chiller energy consumption

The problem of building a meaningful objective to minimize the system energy consumption was tackled with the two-step method described next. The first goal was to reconstruct the chiller refrigeration curve from historical data, more specifically, from the volumetric air-flow rates along with the outdoor temperature and the supplied air temperatures. Based on these quantities, the thermal power delivered by the chiller was inferred. Next, we used as features the outdoor temperature T_{out} and the sum of the valve positions $\Theta = \theta_1 + \theta_2 + \theta_3$, the latter correlating with the water flow through the AHU coils (see Section 3.3.1). A representative dataset was gathered over a period of 203 hours, which encompassed

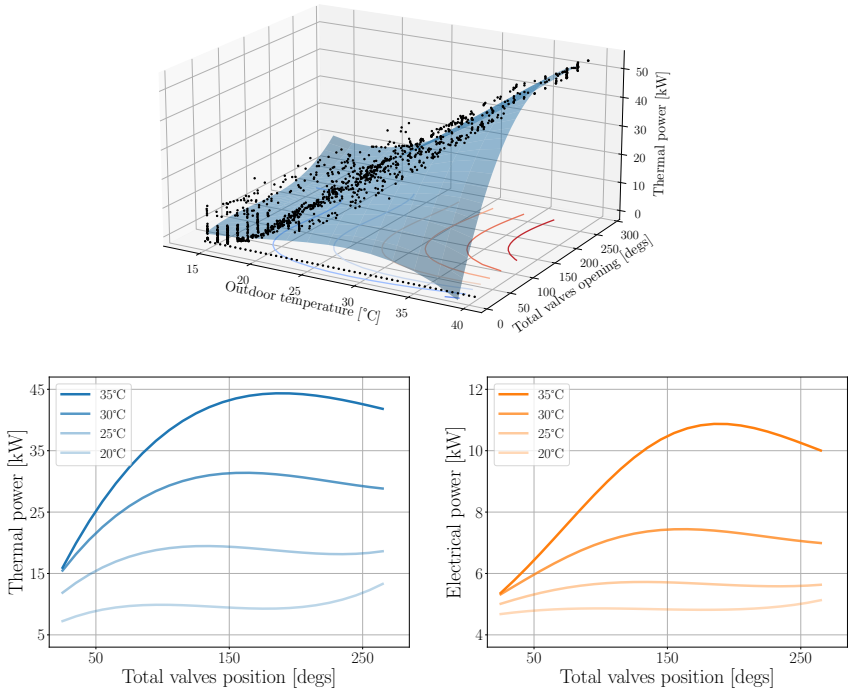

Figure 3.6: Reconstruction of the chiller refrigeration surface (top). Data were collected during a period of 203 hours, including both of open-loop excitation as well as closed-loop operation. Thermal power $Q(T_{\text{out}}, \Theta)$ (bottom left) and electrical power $E(T_{\text{out}}, \Theta)$ (bottom right) curves of the chiller as a function of the valves openings Θ. The plots consider typical outdoor temperature values.

both random open-loop excitation and closed-loop operation. During this period, the AHU valves ranged from being completely open to being fully closed, and the ambient temperature varied from 15 to 40 degrees Celsius. The data distribution can be seen in Figure 3.6. We remark that the portion of the domain where the outdoor temperature is high and the total valve openings are low is not populated with samples due to operational constraints of the system, a common issue in HVAC control. The last step was to augment the batch with a grid of T_{out} values paired with $\Theta = 0\,\text{degs}$ and $0\,\text{kW}$ labels to represent the zero water-flow regime.

Polynomial ridge regression was performed to fit the data described above. After experimenting with different model orders, we found that a cubic model provided a good balance between describing the observed points and not overfitting them. The results are presented in Figure 3.6 and the final model was

$$\begin{aligned} \mathrm{Q}(T_{\text{out}}, \Theta) = {} & 20.22 - 3.15\,T_{\text{out}} - 3.03\,\mathrm{e}^{-2}\,\Theta + 1.73\,\mathrm{e}^{-1}\,T_{\text{out}}^2 - 1.56\,\mathrm{e}^{-3} T_{\text{out}}\Theta \\ & + 3.09\,\mathrm{e}^{-4}\Theta^2 - 2.75\,\mathrm{e}^{-3} T_{\text{out}}^3 + 4.90\,\mathrm{e}^{-4} T_{\text{out}}^2\Theta - 6.86\,\mathrm{e}^{-5} T_{\text{out}}\Theta^2 + 2.56\,\mathrm{e}^{-6}\Theta^3 \end{aligned} \tag{3.8}$$

with $\mathtt{e}^n$ being a shorthand for $\times 10^n$. The model attained a mean absolute error of 2.88 and a mean squared error of 12.97 kW. As a second step, we fit a concave coefficient of performance (COP) curve, which is typical for variable-speed compressor chillers and dictates how efficient they are in converting electrical to thermal energy. The final utilized COP curve was

$$\mathrm{COP}(Q) = -4.45\,\mathtt{e}^{-4} + 2.34\,\mathtt{e}^{-1}Q - 2.67\,\mathtt{e}^{-3}Q^2 - 2.69\,\mathtt{e}^{-5}Q^3 + 3.30\,\mathtt{e}^{-7}Q^4 \quad (3.9)$$

which, given the thermal range displayed in Figure 3.6, implies in a performance coefficient varying approximately between 1.5 and 4.5.

With the thermal model (3.8) and the COP curve (3.9) at hand, the electrical power could be calculated according to $E = Q(T_{\text{out}}, \Theta)/\mathrm{COP}(Q(T_{\text{out}}, \Theta))$, measured in kW. Several slices of the thermal power surface, and their associated electrical power counterparts are presented in Figure 3.6. The plots illustrate how the curves change depending on the outdoor temperature, and how strongly the electrical power profile is affected by this external factor. In particular, one notices that when the outside temperature is high, it is more economic to open the valves and increase the chilled water flow rather than keeping them partially closed. Clearly though, the real-time optimal position for them will depend on the system dynamics, the desired temperature envelope and the external disturbances.

3.4 MPC formulation and numerical computations

Given the learned GP models μ_i, $i = 1, 2, 3$ described in (3.7), a given maximum temperature $T_{\max}$, and our reconstructed electrical power surface, we formulate the following optimization problem to control the valves θ_i while reducing the chiller energy consumption E_t

$$\begin{aligned}
\min \quad & \sum_{t=0}^{N-1} (E_t + \rho\Delta_t) + \rho_N \Delta_N && \text{(3.10a)}\\
\text{s.t.} \quad & T_{t+1} = \mu(T_t, \theta_t, T_{\text{sup}}, T_{\text{out}}) && \text{(3.10b)}\\
& T_t + \beta\,\mathrm{var}^{1/2}(T_t, \theta_t, T_{\text{sup}}, T_{\text{out}}) \le T_{\max} + \delta_t && \text{(3.10c)}\\
& E_t = Q(T_{\text{out}}, \Theta_t)/\,\mathrm{COP}(Q(T_{\text{out}}, \Theta_t)) && \text{(3.10d)}\\
& \theta_{\min} \le \theta_t \le \theta_{\max} && \text{(3.10e)}\\
& \delta_t \ge 0 && \text{(3.10f)}
\end{aligned}$$

where $\Theta_t = \sum_{i=1}^{3} \theta_{i,t}$ is the sum of all valve positions. The variables δ_t in (3.10c) are positive slacks introduced to avoid infeasibility. If needed, these can relax the temperature constraint so that the solver can return a viable control plan.

Of course, their use is heavily penalized in the objective, where $\Delta_t = \sum_{i=1}^{3} \delta_{i,t}^2$ and ρ, ρ_N are large constants, which in our case were respectively set to 100 and 200. The temperature constraint (3.10c) also accounts for prediction uncertainty as it includes the standard deviation $\text{var}^{1/2}$. Its use confers on the formulation a risk-aware quality and robustifies the closed-loop operation. The degree of conservativeness is controlled by the constant β, chosen to be 2 as in Figure 3.5. The prediction horizon was set to $N = 12$ steps, which translates to 2 hours. As suggested by our notation, T_{out} and T_{sup} were kept constant throughout all prediction steps–but updated from one sampling period to the next. Finally, our maximum temperature value was $T_{\max} = 21$ degrees Celsius.

The optimization problem (3.10) was written in `Python` with the aid of CasADi (Andersson et al., 2019), an automatic-differentiation package that provides gradient information for numerical solvers–in our case, the interior-point method IPOPT. As is customary in predictive control, (3.10) was recursively solved on-line with the most recently available system information, with only the first optimal control action being transmitted to the valves. We underline that the main source of complexity in (3.10) is the presence of the constraints (3.10b) and (3.10c), which are highly non-linear due the GP mean and variance. Since convexity is absent, multiple local optima might exist, a fact that was indeed verified in practice. By intelligently providing solvers with high-quality initial guesses, this problem can be mostly overcome. Our particular case study relied on initializing the numerical solver with control, temperature, slack and energy trajectories obtained with a virtual PI controller. The intuition was to allow the MPC loop to build on such an initial guess and further optimize operation. For a detailed study on solve times and how the number of GP data-points impacted them, see Appendix A.

In order to shed light on how the number of training points affects the solve times of the non-convex optimization problem (3.10), the following study was conducted. Five sets of GP models were trained on distinct datasets with cardinalities $N = 352$, 794 (precisely the one used in the experiments), 1280, 1910 and 2643. In all scenarios, the total number of points present in each of the GPs was approximately a third of the total number N, so that the models were balanced. We then generated random initial conditions, uniformly sampled from sensible intervals: $16 \leq T_{1,2,3} \leq 23$, $9 \leq T_{\text{sup}} \leq 13$, and $15 \leq T_{\text{out}} \leq 35$. Finally, we solved the warm-started non-convex MPC (3.10) on a 2.4 GHz, i9 machine 50 times per scenario and recorded their run times.

The results are presented in Figure 3.7, where the vertical scale is logarithmic. The median values of the box plots rose from 4.19 to 18.69, 85.42, 121.12 and 255.68 seconds respectively from the smallest to the largest dataset. Although using

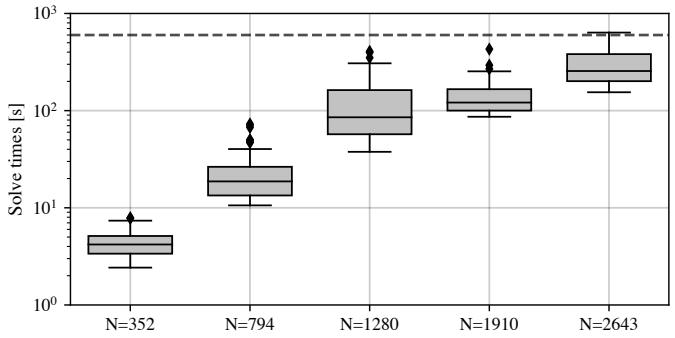

Figure 3.7: Box and whisker plots of the MPC solve times considering different dataset sizes. The dashed gray line marks our sampling period of 10 mins. Each boxplot is based on 50 time samples, obtained using randomized initial conditions.

$N = 1910$ points does not seem unreasonable at first, challenging initial conditions such as ones close to violating constraints can easily increase the problem solve time: the highest point obtained for the $N = 1910$ scenario was 429 seconds. Ideally, and specially when occupants experience thermal discomfort, the control action has to be computed in negligible time to be applied as soon as possible to the system. Keeping the solve times below a low percentile of the total sampling period is thus a common desideratum. In our application, we regarded $N = 794$ to be an adequate choice.

3.5 Experimental results

The previously described Gaussian process-based MPC formulation was deployed on the local computer and used to operate the surgery center cooling system during multiple days in the months of October and November 2021. We report in Figure 3.8 a four-day uninterrupted experiment carried out from November 10 to November 13 that is rather representative of the local internal and external conditions. We highlight that the curves displayed in the figure were not filtered in any way; the sole manipulation performed with the data was the imputation of the missing temperature entries using linear interpolation. These points, however, accounted for only 43 out of the 1728 indoor temperature values gathered during the four-day experiment. The first and third top plots show the room temperatures and the immediate uncertainty associated with the GP predictions $T_{\text{unc}} = \beta \text{var}^{1/2}$ as employed in the formulation (3.10c). The second plot displays the valve positions, that is, the control variables. Both outdoor signals, the external temperature and the solar radiation are also given. The reader is reminded that, although the latter contributes with additional heat gains, it is completely unknown to the controller as explained in Section 3.3.2. Lastly, the bottom plot shows the supply

water temperature T_{sup}, an external disturbance that was incorporated in the GP models. This signal evolves according to the chiller own dynamics, over which we had no direct control.

Consider first the day, November 10, and note the relatively high room temperatures when the experiment started, which were the consequence of a harsh previous day. The MPC controller made use of some control authority to bring the temperatures below the 21-degree line before partially closing the valves. After the morning shift started (7 am), even though θ_2 and θ_3 were fully open, T_2 and T_3 violated the constraints and were only brought below 21 degrees late that evening. High initial conditions along with a peak outdoor temperature of 35 degrees overloaded the cooling system, causing violations of the indoor temperature constraint in two rooms. With regard to T_1, it raised quickly from 19 to 21 degrees before noon, but then stayed essentially constant at the 21-degree mark for several hours before being lowered during the evening.

The two days that followed, November 12 and 13, were less warm and, as a result, the MPC controller successfully modulated the valves to guarantee constraint satisfaction. From the plots, one can notice how θ_1, θ_2 and θ_3 assumed approximately constant intermediate values (around 60 degrees) during the evenings, and were more aggressively modulated on working hours. We believe this pattern to be due to the economic objective shown in Figure 3.6.

Lastly, we focus on the data from November 13, where it is possible to see a sudden peak in the indoor temperatures and in T_{sup} around midnight. This was caused by a momentary halt in the water pumps responsible for the chilled water circuit, an event that could be regarded as a fault from a control system perspective. During this period, as there was no water circulation through the AHU cooling coils, there was also no refrigeration and the indoor spaces received warm air since the fans were kept on. As soon as the pumps were reactivated, the chiller immediately lowered the supply water temperature and normal operation was restored. During daytime, the indoor climate was kept within the desired limits despite the valves staying saturated at their low values, even at noon. The fact that almost no additional actuation was needed is due to that day being a Saturday, when no operations are scheduled for and the three doors present in the environment are minimally opened and closed. This showcases how strong the internal heat gains and unmeasured disturbances normally are.

To assess the efficiency gains as well as the thermal performance of the deployed strategy, MPC was compared to alternative algorithms, all subject to exactly the same environmental conditions by means of simulations. We underline that

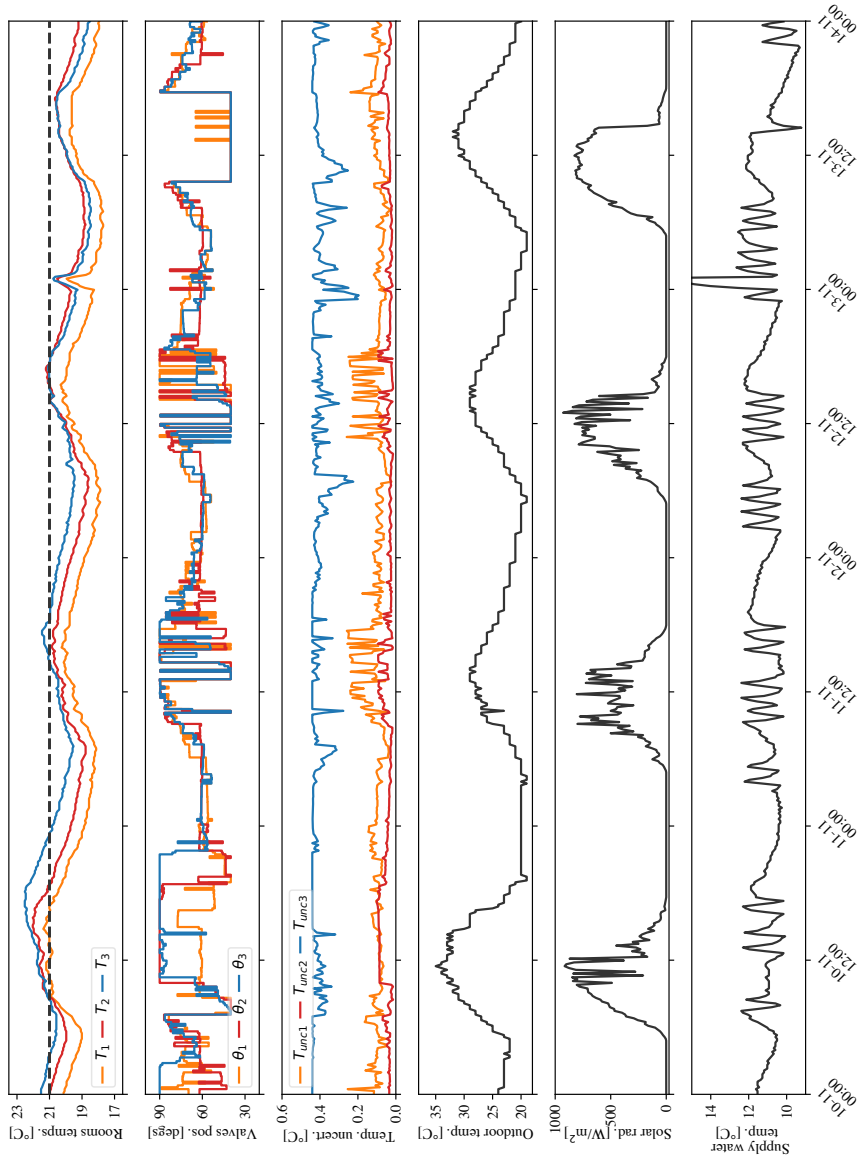

Figure 3.8: MPC experimental results over four days: indoor temperature, valve position and uncertainty estimate associated with each room (top three plots); outdoor temperature, solar radiation and AHUs supply water temperature (bottom three plots). The system was sampled and controlled with a periodicity of 10 minutes.

this simulation model was calibrated on data that was *not* included in the GPs training set, thus putting to test the MPC prediction capabilities. The disturbance signals T_{out}, T_{sup} and R_{sol} from November 10 were employed, and the indoor temperatures of the three rooms were uniformly initialized at values ranging from 17 to 21 degrees Celsius. The outdoor temperature profile was processed to yield three different weather scenarios: hot weather, which was exactly the same T_{out} curve seen in Figure 3.8; warm weather, a -2°C shifted version of it, peaking at 33°C around noon; and mild weather, a -5°C shifted version of it, peaking at 30°C. Besides the MPC algorithm (3.10), the following were also tested:

- An MPC controller, which will be referred to as REF, with perfect prediction capabilities, perfect disturbance information (T_{out} and T_{sup}) and a long prediction horizon of five hours.
- PI controllers featuring anti-windup schemes and feedforward components to enhance their performance.
- Rule-based ON/OFF controllers that set the valves respectively to $\theta_{\min}$ and $\theta_{\max}$ if the indoor temperatures were below or above the set-point.
- An average $(\theta_{\max} - \theta_{\min})/2$ controller (AVG) whose instantaneous values are selected by sampling the interval $\theta_{\min}$ to $\theta_{\max}$ using a uniform distribution.

The REF strategy described above was conceived and tested precisely gauge the energy saving potential of the hospital HVAC plant. This algorithm, thanks to the way it was designed with a perfect internal model and perfect forecast of the outdoor temperature and supply water temperatures, will indicate what can be achieved realistically.

The obtained normalized energy consumption results and average room thermal comfort violations are shown in Figure 3.9. Since the non-linear chiller curves described in Section 3.3.3 were employed to measure the energy consumption, the reader is reminded that there is a non-trivial relationship between the weather conditions and indoor temperatures, and the final consumed energy. This aspect is due to the nature of the chiller and not the use of any particular control technique. As a last note, one specific energy normalization factor was used for each weather scenario shown in Figure 3.9 to enhance clarity.

Glancing at the mild and warm weather plots, one notices how the REF and MPC data tended to be close together, and relatively far from the PI, ON/OFF and AVG clusters. Moreover, the REF and MPC points were also mostly to the left side and vertically below the other data given the same indoor temperature

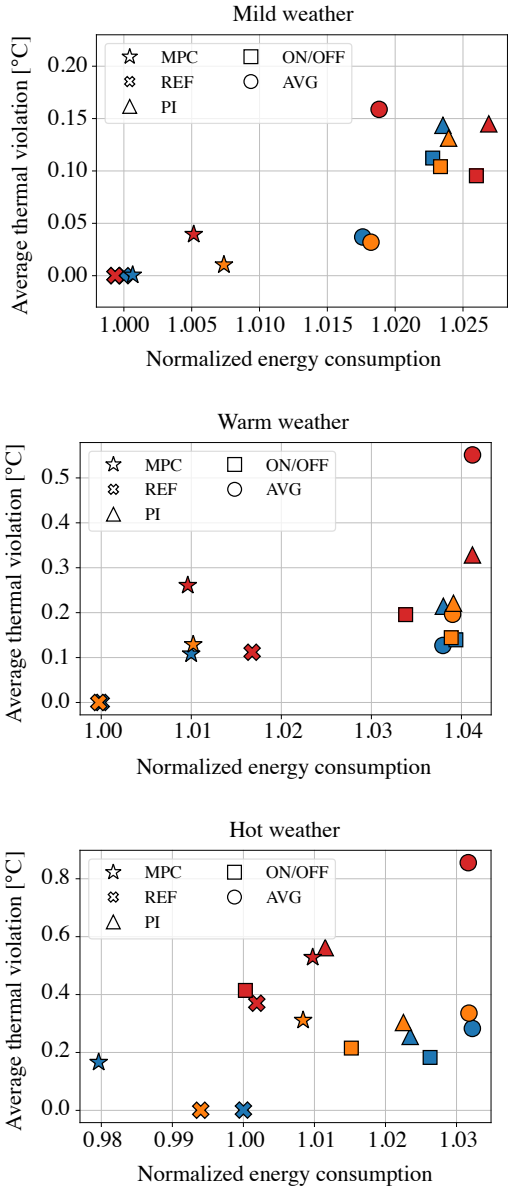

Figure 3.9: Simulation results of the normalized energy consumption and thermal performance (average temperature bound violation) of different control strategies. Three weather profiles were considered: mild, warm and hot. The indoor temperatures were initialized at three different values according to the color scheme: 17°C (blue), 19°C (orange) and 21°C (red).

conditions–thus confirming their superior performance in terms of energy efficiency and indoor climate regulation. The ON/OFF and PI controllers yielded overall similar numerical results and, surprisingly, were outperformed by the AVG scheme under mild weather and starting indoor temperatures of 17 °C and 19 °C. AVG nevertheless performed poorly under warm weather and 21 °C, and hot weather

in general. All in all, the predictive control strategies MPC and REF yielded the best results in the mild and warm weather cases, whereas the separation among them and the other techniques became less evident under hot weather, indicating a less important advantage over classical control.

By analyzing the horizontal scales and contrasting REF to PI, ON/OFF and AVG, one concludes that this particular HVAC plant could have its efficiency boosted by approximately 2.5%, 4% and 5% respectively in the mild, warm and hot weather scenarios. Notice how, as opposed to studies such as Bünning et al. (2020), these numbers refer to the electrical energy associated with a chiller, and not to a cumulative thermal energy. Moreover, as the hospital was not subject to time-varying electricity prices, the contrast among control strategies was not as stark as for instance the one reported in Joe (2022). The proposed MPC strategy (3.10) attained results close to the aforementioned maximum percentages. Quantitatively, MPC lead to a maximum energy efficiency improvement of 2.29%, 3.13% and 4.76% respectively in mild, warm and hot weather, with respect to the PI and ON/OFF counterparts.

3.6 Conclusions and outlook

The practical investigation presented in this chapter revolved around the use of Gaussian process models paired with MPC to autonomously control the industrial cooling system of a hospital building. The study was complex in comparison with many works found in the literature: it involved three thermal zones, multiple subsystems, two distinct sensor networks, four strong exogenous disturbances, and the direct control of low-level elements. From a computational viewpoint, expensive non-parametric models were used along with a non-convex MPC formulation, both in the cost and in the constraints, which brought about numerical challenges. All things considered, the chosen approach was able to guarantee the indoor comfort whenever enough actuation power was available, while also delivering superior energy performance when compared to alternative strategies.

During the course of the project, a large portion of time was dedicated to studying the building and the AHUs, deciding what signals to measure, and setting up the necessary hardware. The second most time-consuming task was certainly building a functional data ingestion pipeline for the GPs: applying the necessary filters, detecting non-reliable periods, and crafting the feature vectors by delaying the time-series and patching multiple experiments together. We therefore believe that the control community could greatly benefit from a reliable toolbox to design and test GP autoregressive models with ease, which would certainly further increase

the popularity of this technique.

With regard to possible future investigations, we consider exploring sparse Gaussian processes based on pseudo-inputs (Bauer et al., 2016) to lower the computational burden of the MPC controller and allow for longer prediction horizons. Taking a step back, it would also be interesting to experimentally investigate with different combinations of models (not necessarily Gaussian processes) and training objectives to find a pair that is able to automatically reject spurious data periods, not only isolated points, and dispense with the need of a meticulous pre-processing stage. Indeed, being robust to an abnormal period of operation at training time could alleviate the workload of field engineers and help fulfill the promising an effortless modeling phase. In our view, parametric techniques such as neural networks are less sensitive as their internal constants are adjusted gradually by the data, whereas non-parametric techniques absorb the whole dataset and make use of it during inference. That said, perhaps additional (unobserved) covariates could be used to explain an abnormal trajectory, essentially acting as a disturbance model. Finally, from an HVAC systems perspective, more encompassing investigations could include the control of additional variables, e.g. the air flow delivered to the rooms, as well as the regulation of physical quantities other than the temperatures such as the indoor spaces humidity levels.

4 Learning MPC controllers with pQP neural networks

4.1 Introduction

Consider the standard MPC formulation for linear dynamical systems

$$\mathbb{P}1 : \min_{X,U} \quad \sum_{k=0}^{N-1} \left(x_k^\top Q x_k + u_k^\top R u_k \right) + x_N^\top P x_N \tag{4.1a}$$

$$\text{subj. to} \quad x_{k+1} = Ax_k + Bu_k, \, \forall k = 0, \dots, N-1 \tag{4.1b}$$

$$x_k \in \mathbb{X}, \, \forall k = 0, \dots, N-1 \tag{4.1c}$$

$$u_k \in \mathbb{U}, \, \forall k = 0, \dots, N-1 \tag{4.1d}$$

$$x_H \in \mathbb{X}_N \tag{4.1e}$$

$$x_0 = x(0) \tag{4.1f}$$

where $x_k \in \mathbb{R}^{n_x}$, $u_k \in \mathbb{R}^{n_u}$, for all k; $X := \{x_1, \dots, x_N\}$ and $U := \{u_0, \dots, u_{N-1}\}$ are the collections of decision variables; $Q \succeq 0$, $R \succeq 0$ and $P \succ 0$ are cost matrices of adequate size; and the sets $\mathbb{X}$, $\mathbb{X}_N$ and $\mathbb{U}$ are polyhedra. Also, denote by

$$\pi : \mathcal{X} \to \mathbb{U}, \quad x(0) \mapsto \pi(x(0)) = u_0^\star \tag{4.2}$$

the mapping defined point-wise from the parameter $x(0)$ of $\mathbb{P}1$ to the first control component of its optimizer[1]. The domain $\mathcal{X} \subset \mathbb{R}^{n_x}$ is the collection of all $x(0)$ for which $\mathbb{P}1$ is finite, i.e., the problem is feasible. It is well-known that, under reasonable standard on the terminal ingredients P and $\mathbb{X}_N$, the map π is a piece-wise affine (PWA) continuous function over a polyhedral partition of the polyhedron $\mathcal{X}$ (Bemporad et al., 2002; Borrelli et al., 2017).

[1] Since the objective is strictly convex and the constraints define a convex feasible set, if an optimizer exists for a given parameter $x(0)$, then this optimizer is unique.

Automatic control involves real-time decision making and, in a number of different scenarios, it becomes infeasible or simply undesirable to solve an optimization problem such as (4.1) on-line. Picture, for instance, a small embedded system that does not have enough computational resources to arrive at a solution within the time allocated to it. One method to circumvent solving $\mathbb{P}1$ is to seek π, which can explicitly written in closed form with multi-parametric programming tools (Bemporad et al., 2002). This approach is known as explicit MPC and has become an establish control tool (Mariéthoz and Morari, 2008; Naus et al., 2010) for which free software exists (Herceg et al., 2013). Unfortunately, π can also be challenging both to compute and to store as the number of regions of its domain grows exponentially in the worst case with the number of inequality constraints in $\mathbb{P}1$ (Alessio and Bemporad, 2009), thus limiting its applicability. Simplifications strategies for explicit MPC exist, each coming with a different set (sometimes empty) of guarantees. In Geyer et al. (2008) and Wen et al. (2009), for example, the authors build equivalent but more convenient representations of the control policy; in Jones and Morari (2008) an incremental method is proposed to fit π without requiring its explicit description; and in Christophersen et al. (2007) and Maddalena et al. (2019) safe elimination procedures are presented to drop certain regions without disrupting certain closed-loop properties.

With the recent popularization of deep learning and the plethora of easy-to-use toolboxes available to users, NN approximations of the map linking the current state $x(0)$ to the MPC optimal control action $u_0^\star$ have also become trendy (see e.g. Karg and Lucia (2020); Kumar et al. (2021)). Although the idea itself is not new (Parisini and Zoppoli, 1995), the latest investigations tend to make use of more modern analysis and synthesis tools such as better activation functions and training algorithms. The works Fazlyab et al. (2020) and Schwan et al. (2022) address the problem of verifying a NN approximation a posteriori, employing respectively semi-definite and mixed-integer linear programming techniques. On a different spirit, Paulson and Mesbah (2020) tackles designing NNs that approximate predictive controllers and are safe by design. Arguably the main motivation behind this entire line of research is the expressive power of NNs, especially when deep architectures are employed Karg and Lucia (2020), and the fact that they lie in the same class of PWA functions as MPC controllers[2].

As opposed to the previously cited works where rather multi-purpose NN architectures were used, we will explore a structure that is tailored to the problem $\mathbb{P}1$ and its solution π. Our proposal, which was published in Maddalena et al. (2020) and Maddalena et al. (2021b), revolves around utilizing convex programs as layers within neural networks as proposed in Amos and Kolter (2017) and Agrawal et al.

[2] As long as a PWA activation function such as the rectified linear unit (ReLU) is used.

(2019). More specifically, the parameter to optimizer map of a convex program is employed as an internal layer, offering a particular inductive bias. A shallow architecture is proposed and shown to be able to represent the map π of any predictive controller (4.1) given a sufficiently large size. Finally, two examples are presented in the area of power electronics, allowing for a drastic speed-up in computation times with only a minor performance degradation.

4.2 The proposed architecture

An illustration of the parametric quadratic programming (pQP) neural network is shown in Figure 4.1. It is composed of two linear layers (L1 and L3), a single non-linear layer with adjustable parameters (L2), and a fixed output layer (L4). The central part of this structure is the pQP layer L2, which is an implicit function described by the optimization model

$$y_2 = \arg\min_{z \geq 0} \ ||Hz + y_1||^2 + \epsilon||z||^2 \tag{4.3}$$

that has y_1, the output of layer L1, as a parameter; the optimizer y_2 is then a function of y_1. Note how (4.3) is always feasible and bounded from below, thus well posed for any y_1. The regularizing constant $\epsilon > 0$ is a fixed parameter that contributes with the numerical stability of the scheme. Additionally, if $\epsilon > 0$, then (4.3) has always a unique minimizer and thus the map $y_1 \mapsto y_2$ is well defined.

Intuitively, one can think of the first layer L1 as a lifting, or projection, of the current state $x \in \mathbb{R}^{n_x}$ onto another space $\mathbb{R}^{n_z}$ whose dimension is a NN hyperparameter to be chosen by the user. L2 then is the key non-linear transformation that should capture most of the MPC policy complexity. Next, the affine map in L3 takes the y_2 value to the input space $\mathbb{R}^{n_u}$ and finally the control actions are guaranteed to satisfy the control constraints by projecting them onto $\mathbb{U}$, which is accomplished by L4 (see (A.24)). By adjusting n_z, the designer increases or reduces the representation power of the pQP NN although its size remains constant.

Let $\hat{\pi} : x \mapsto u$ be the pQP NN map defined by the composition of the layers in Figure 4.1, and whose parameters to be trained are $F \in \mathbb{R}^{n_z \times n_x}$, $f \in \mathbb{R}^{n_z}$, $H \in \mathbb{R}^{n_z \times n_z}$, $G \in \mathbb{R}^{n_u \times n_z}$ and $g \in \mathbb{R}^{n_u}$. The final goal is to tune these to fit a dataset of the form

$$\{(x(0)_i, u^\star_{0,i})\}_{i=1}^d \tag{4.4}$$

composed of initial conditions $x(0)$ and the first optimal control action $u_0^\star$ coming from (4.1). Typically, this is done by defining a suitable loss function and minimizing the mismatch between $\hat{\pi}(x(0))$ and $u_0^\star$, i.e., the control action produced

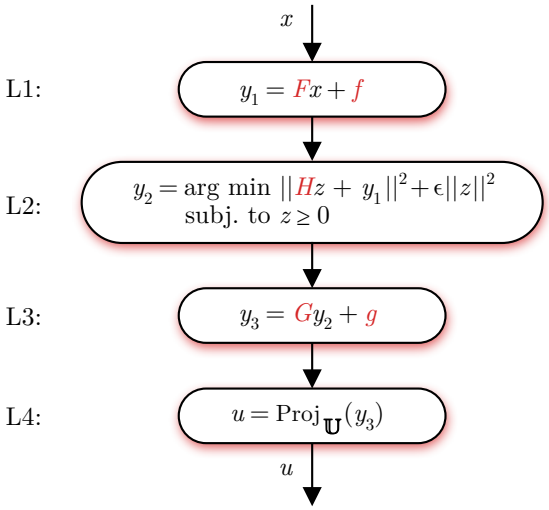

Figure 4.1: A pQP neural network architecture tailored to MPC. The variables in red are weights to be adjusted during training.

by the pQP NN and the target optimal value. The parameters are then tuned by means of a gradient-based backpropagation procedure.

Remark 13. As shown in (Amos and Kolter, 2017, Theorem 1), OptNet layers are differentiable everywhere, but not in a set of measure zero of parameters, where subgradients exist. Since (4.3) is a particular instance of the OptNet layer, it inherits its properties. Moreover, we assume the set of feasible control actions $\mathbb{U}$ to be a polyhedron. The projection operator in L4 is therefore a quadratic program with y_3 as its only parameter and, thus, also an OptNet layer.

4.3 Properties of the approximator

Regard the size n_z in L2 as a hyperparameter. It is then clear that as n_z increases, the expressive power of the pQP NN also increases. Moreover, as $\hat{\pi}$ is defined by a composition of piece-wise affine functions, it is itself piece-wise affine. Next we prove that $\hat{\pi}$ can not only approximate well any linear MPC controller, but also recover an exact representation if an appropriate size n_z is chosen.

Theorem 3. (The pQP NN can learn any linear MPC controller): Let $\hat{\pi} : \mathbb{R}^{n_x} \to \mathbb{R}^{n_u}$ be the map defined by the composition of all four layers, i.e., $\hat{\pi}(x) := L4 \circ L3 \circ L2 \circ L1(x)$. Set $\epsilon = 0$, then $\exists F$, f, H, G and g with appropriate dimensions such that $\forall x \in \mathcal{X}$, $\hat{\pi}(x) = \pi(x)$.

Proof: In essence, the proof consists in showing that L2 has exactly the same

structure as the dual of the MPC formulation (4.1), and in showing that L3 can recover the primal solution from the dual.

Start by condensing the MPC problem $\mathbb{P}1$, i.e., using the equality constraints to eliminate all state decision variables except for the initial state $x(0)$. This leads to the following parametric problem

$$\mathbb{P}2 : \min_{U} \quad U^\top \Lambda U + x(0)^\top \Gamma U \tag{4.5a}$$

$$\text{subj. to} \quad \Phi U \leq \Omega x(0) + \omega \tag{4.5b}$$

The step by step condensing procedure can be found in Wright (2019), which also shows that $\Lambda \succ 0$. The problems $\mathbb{P}2$ and $\mathbb{P}1$ are then equivalent in the sense that the solutions $U^\star$ of $\mathbb{P}2$ and $(X^\star, U^\star)$ of $\mathbb{P}1$ share the same $U^\star$ component. Next, derive the dual problem of $\mathbb{P}2$, which is

$$\begin{aligned}\mathbb{D}2 : \min_{\lambda \geq 0} \ \frac{1}{4}\Big[\lambda^\top \Phi \Lambda^{-1} \Phi^\top \lambda + \ (4x(0)^\top \Omega^\top + 2x(0)^\top \Gamma \Lambda^{-1} \Phi^\top + 4\omega^\top)\lambda \ldots \\ + x(0)^\top \Gamma \Lambda^{-1} \Gamma^\top x(0)\Big]\end{aligned} \tag{4.6}$$

And use the KKT stationarity condition of $\mathbb{P}2$ to arrive at an expression that relates linearly the optimal primal and dual solutions

$$U^\star = -0.5\, \Lambda^{-1} \Phi^\top \lambda^\star - 0.5\, \Lambda^{-1} \Gamma^\top x(0) \tag{4.7}$$

Next, rewrite L2 in the standard QP form

$$\min_{z \geq 0} \ z^\top (H^\top H + \epsilon I) z + (2H^\top y_1)^\top z + y_1^\top y_1 \tag{4.8}$$

and match[3] the two quadratic programs (4.6) and (4.8) through

$$\tilde{H}^\top \tilde{H} + \epsilon I = 0.25\, \Phi \Lambda^{-1} \Phi^\top \tag{4.9a}$$

$$2\tilde{H}^\top \tilde{y}_1 = \Omega\, x(0) + 0.5\, \Phi \Lambda^{-1} x(0) + \omega \tag{4.9b}$$

which leads to $\epsilon = 0$, $\tilde{H} = 0.5\, (\Phi \Lambda^{1/2})^\top$, and $\tilde{y}_1 = (\Phi \Lambda^{1/2})^{-1}(\Omega\, x(0) + 0.5\, \Phi \Lambda^{-1} x(0) + \omega)$. This value of $\tilde{y}_1$ also defines the weights F and f of the L1 layer simply as $\tilde{F} = (\Phi \Lambda^{1/2})^{-1}(\Omega + 0.5\, \Phi \Lambda^{-1})$ and $\tilde{f} = (\Phi \Lambda^{1/2})^{-1} \omega$.

Tilde superscripts were used for H, y_1, F and f in the last paragraph because those are not the final values of those parameters. Indeed, they need to be augmented to resolve the following issue. For the layer L3 to implement (4.7), it would need

[3] The last terms of the two QPs are disregarded since their are independent of the optimization variables and thus have no effect on their respective optimizers.

access to $x(0)$, whereas it only receives y_2 from the previous layer. The solution lies in augmenting L1 and L2 to match the MPC dual as previously done and, in addition, also let the NN input $x(0)$ pass through them and arrive at L3. More concretely, the steps below have to be followed.

Set the first layer weights to $F = [-I\ I\ \tilde{F}]^\top$ and $f = [0\ 0\ \tilde{f}]^\top$ so that $y_1 = [-x\ x\ \tilde{F}x + \tilde{f}]^\top$. Set the L2 weights to $\epsilon = 0$ and $H = [I\ 0\ 0;\ 0\ I\ 0;\ 0\ 0\ \tilde{H}]$. Partitioning the decision vector into three components $z = [x^p\ x^n\ \tilde{z}]^\top$ leads to

$$\min_{\tilde{z}, x^p, x^n \geq 0} ||x^p - x(0)||^2 + ||x^n + x(0)||^2 + ||\tilde{H}\tilde{z} + \tilde{y}_1||^2 \tag{4.10}$$

which is a separable objective in $\tilde{z}$, $\tilde{x}^p$ and $\tilde{x}^n$. Thanks to the choice of $\tilde{L}$ and $\tilde{y}_1$ we have that $\tilde{z}^\star$ in (4.10) matches $\lambda^\star$ in (4.6). Regarding $x^{p\star}$, the nth optimizer components will satisfy $\forall i = 1, \dots, n$, $x_i^{p\star} = x_i(0)$ if $x_i(0) \geq 0$, else $x_i^{p\star} = 0$. Similarly, $x_i^{n\star} = -x_i(0)$ if $x_i(0) \leq 0$, else $x_i^{n\star} = 0$. Therefore, $x^{p\star} - x^{n\star} = x(0)$, and the output of the pQP layer (4.10) will contain the dual optimizer $\lambda^\star$ and the NN input value $x(0)$ encoded in it.

Next set the weights of L3 to match (4.7), but only extracting the first optimal control action rather than the whole vector. This is accomplished by $G = \nabla\,[-0.5\Lambda^{-1}\Gamma^\top \quad 0.5\Lambda^{-1}\Gamma^\top \quad -0.5\Lambda^{-1}\Phi^\top]$ and $g = 0$. Therefore, $y_3 = G\,[x^{p\star}\ x^{n\star}\ \tilde{z}^\star]^\top = G\,[x^{p\star}\ x^{n\star}\ \lambda^\star]^\top = \nabla\,U^\star = u_0^\star$, where ∇ is a matrix containing zeros everywhere and a $n_u \times n_u$ identity matrix in its lower right corner.

Finally, note that L4 will evaluate to $u_0^\star$ since $y_3 = u_0^\star$ necessarily belongs to $\mathbb{U}$. The proof is concluded after observing that the value of $x(0)$ in the above calculations can be taken to be any point x in $\mathcal{X}$. □

Remark 14. Given a large enough size n_z, $\hat{\pi}(x)$ can match $\pi(x)$ under appropriate parameters. In practice however, one is rather interesting in experimenting with small sizes n_z to reduce the complexity of the final map.

Remark 15. As opposed to $\pi(x)$, $\hat{\pi}(x)$ is defined for any $x \in \mathbb{R}^{n_x}$.

The deployment of the pQP NN does not have to involve solving L2 on-line. Indeed, its explicit solution could be found off-line, in which case the real-time evaluation of $\hat{\pi}(x)$ would boil down to computing affine and piece-wise affine expressions. By adjusting n_z, the designer can therefore tune $\hat{\pi}(x)$ to its needs. The whole process can be than seen as the fitting of a PWA function, the pQP NN, to another PWA function, the MPC explicit solution.

Approximating the MPC controller (4.1) with our network architecture does not come with *a priori* guarantees on closed-loop stability or convergence to

an equilibrium point. Nevertheless, once the pQP NN is trained, tools from mixed-integer programming can be exploited to compute the worst-case mismatch between $\pi(x)$ and $\hat{\pi}(x)$ as well as basins of attraction for the closed-loop system (Schwan et al., 2022). Alternatively, a robust MPC controller could be designed taking into account a certain level of input disturbace, which will be incurred later by the approximation scheme. This is the idea proposed in Hertneck et al. (2018), where statistical guarantees are derived based on samples, and without needing access to the worst-case approximation error. Finally, safety filters could be employed not to certify the pQP NN, but to modify minimally its control actions to ensure system theoretical properties (Wabersich and Zeilinger, 2018).

4.4 Numerical validation

In this section we make use of the previously proposed architecture to approximate the MPC controller of a 4-state, 3-input model of a multi-cell DC-DC power converter. The size n_z is then varied and its effect on the quality and size of the approximate solution is studied.

4.4.1 The dynamics and the target predictive controller

Parallelism is a key concept to increase the efficiency and power levels of electronic converters. Still, this design choice has to be followed by proper current and voltage balancing techniques to ensure that no single stage is subjected to a exceedingly high electrical stress when compared to the others.

A schematic representation of a multi-cell step-down converter is shown in Figure 4.2, and its parameters can be found in Table 4.1. The topology features three arms that are connected to a coupled inductor, and an L-C output filter. All self inductances are assumed equal $L_1 = L_2 = L_3 = L_s$, and all mutual inductances have value L_m. The switches of each arm operate in a complementary fashion at a fixed frequency $f = 15$ kHz, and with variable but constrained duty cycle $0 \leq d_i \leq 0.9$, $i = 1, 2, 3$. Let the average voltage applied by the arms over one switching period be denoted by $v_i := d_i V_{in}, i = 1, 2, 3$. In order to ease the analysis, we apply the Lunze transform Ψ to all variables, decomposing the phase voltages and currents into differential and common mode components

$$\begin{bmatrix} i_{dm1} & i_{dm2} & i_{cm} \end{bmatrix}^\top := \Psi \begin{bmatrix} i_1 & i_2 & i_3 \end{bmatrix}^\top \tag{4.11}$$

$$\begin{bmatrix} v_{dm1} & v_{dm2} & v_{cm} \end{bmatrix}^\top := \Psi \begin{bmatrix} v_1 & v_2 & v_3 \end{bmatrix}^\top \tag{4.12}$$

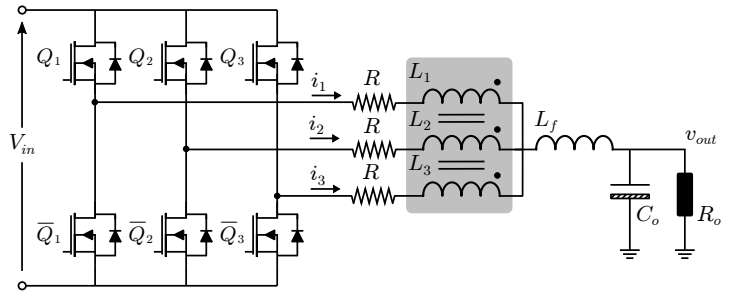

Figure 4.2: A diagram of the multi-cell step-down DC-DC converter.

where $\Psi = (1/3)\,[2\ -1\ -1;\ -1\ \ 2\ -1;\ 1\ \ 1\ \ 1]$.

The control input is defined as $u := [v_{dm1}\ v_{dm2}\ v_{cm}]^\top$ and the continuous-time state vector, by appending the output voltage to the transformed currents $x := [i_{dm1}\ i_{dm2}\ i_{cm}\ v_{out}]^\top$. By using Kircchoff's circuit laws, a linear model of the form $\dot{x} = A_{ct}x + B_{ct}u$ can be derived with

$$A_{ct} = \begin{bmatrix} \frac{-R}{L_s - L_m} & 0 & 0 & 0 \\ 0 & \frac{-R}{L_s - L_m} & 0 & 0 \\ 0 & 0 & \frac{-R}{L_s + 2L_m + 3L_f} & \frac{-1}{L_s + 2L_m + 3L_f} \\ 0 & 0 & \frac{3}{C_o} & \frac{-1}{R_o C_o} \end{bmatrix} \tag{4.13}$$

$$B_{ct} = \begin{bmatrix} \frac{1}{L_s - L_m} & 0 & 0 \\ 0 & \frac{1}{L_s - L_m} & 0 \\ 0 & 0 & \frac{1}{L_s + 2L_m + 3L_f} \\ 0 & 0 & 0 \end{bmatrix} \tag{4.14}$$

Finally, discretization at frequency f is carried out using the zero-order hold method, yielding $x_{k+1} = Ax_k + Bu_k$.

The control goal is to regulate the output voltage v_{out} to 300 V while maintaining the phase currents balanced at all times, which translates to driving the differential currents to zero. More specifically we have the following fixed reference $x_{eq} = [0\ 0\ 16\ 300]^\top$ with $u_{eq} = B^\dagger(I - A)x_{eq}$, where $B^\dagger$ denotes the pseudo-inverse of B. Moreover, the controller approximation procedure must not incur a steady-state error larger than 200 mA for i_{dm1} and i_{dm2}, and 5% for the common mode component i_{cm} and output voltage v_{out}. The chosen MPC cost function was[4]

$$J = \sum_{k=0}^{N-1} (||x_k - x_{eq}||_Q^2 + ||u_k - u_{eq}||_R^2) \ + \ ||x_N - x_{eq}||_P^2 \tag{4.15}$$

where $Q = \mathrm{diag}(10,\ 10,\ 0.1,\ 0.1)$, $R = 0.1\,I$, P is the solution to the associated the

[4] $||a - b||_C^2 := (a - b)^\top C(a - b)$.

Table 4.1: The parameters of the multi-cell step-down DC-DC converter.

V_{in}	L_s	L_m	R	L_f	C_o	R_o
350 V	4 mH	-2 mH	10 mΩ	270 μH	20 μF	6.25 Ω

discrete-time algebraic Riccati equation, and $N = 10$. For all time instants, box state constraints were imposed $\begin{bmatrix}-5 & -5 & -10 & -20\end{bmatrix}^\top \leq x_k \leq \begin{bmatrix}5 & 5 & 30 & 400\end{bmatrix}^\top$ and polyhedron constraints on the controls $H_u\, u_k \leq h_u$ that simply mapped the duty cycle saturation to the Lunze domain. Due to the polytopic input constraints, the system cannot be decomposed into three decoupled parts as the structure of matrices A_{ct} and B_{ct} suggest. Furthermore, the standard terminal set constraint was imposed on x_N, defined as the invariant set associated to the unconstrained infinite-time problem formulation.

With the aid of the Multi-Parametric Toolbox (MPT) for MATLAB (Herceg et al., 2013), the explicit MPC solution $\pi(x)$ was computed and consisted of 2'337 regions. By counting the number of parameters of each halfspace and control gain, the memory requirement of this PWA function was found to be 518 kB, considering a 4-byte representation for both integers and floating point numbers.

4.4.2 Learning the optimal controller

A total of 5'000 samples were acquired from the explicit MPC policy using a uniform distribution across the state-space. The dataset outputs $u^\star_{0i}$ were then normalized since their first two components had considerably low amplitudes compared to the third due to the structure of the Lunze transform Ψ. Instead of implementing a general projection operator in L4, the layer was simplified to $u = \Psi\,\mathrm{sat}(y_3)$ with saturation limits 0 and $0.9\,V_{in}$, which clearly guarantees control feasibility without the need of a second quadratic program. Several pQP NNs were then trained with $n_z = 1, \ldots, 7$. The MPC approximators were trained using `PyTorch` and the `OptNet` frameworks, and mean squared error loss function. Mini-batch stochastic gradient descent with Adam (Kingma and Ba, 2014) was the optimization algorithm of choice to minimize the loss with batch size of 50 and 150 epochs. Training a single pQP NN took on average 42 minutes on a 3.1 GHz Intel Core i7 machine without GPU acceleration, and 23 minutes with a single NVIDIA Tesla T4 graphics card. Seven models were trained per n_z size and only the best scoring one was kept per size, whose results are shown in Figure 4.3.

Analyzing Figure 4.3, one sees that an increase in the n_z size tends to lead to a

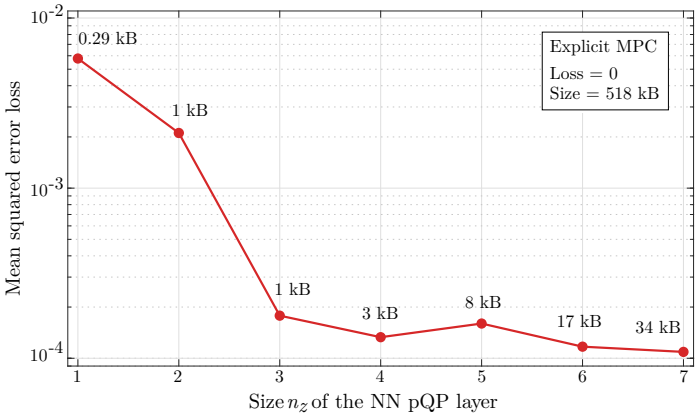

Figure 4.3: Neural network training loss as a function of the pQP layer size, and storage requirements associated with their PWA representations.

better fit. This however does not always translate to a decrease in the final loss since the training process is affected by the weights initialization, the randomized nature of the optimizer, among other factors. As for the size complexity of the pQP NN, the larger the n_z, the more space is needed to store it. Yet, only 6.5% of the original explicit MPC space when $n_z = 7$. Slices of the learned control policies are show in Figure 4.4, where it is possible to visually verify the increasing complexity of $\hat{\pi}(x)$ and how well it resembles the original $\pi(x)$ when $n_z = 7$.

In order to validate the pQP NN, the converter was simulated from four initial conditions and under all 7 different approximate controllers, and only the two largest ones ($n_z = 6$ and $n_z = 7$) met the target specifications given in the previous section. We refer to these two solutions as the *viable learned controllers*. A phase portrait of the closed-loop system evolution over multiple steps starting from these initial conditions is depicted in Figure 4.5. A summary of the two viable learned controller key figures is presented in Table 4.2, including their number of polytopic regions, storage requirements, worst-case computation time and the output steady-state (SS) error. Even though four initial states were given, the systems always converged to the same points and, hence, only one SS error is reported. Plus, the storage numbers also take into account all the remaining layers parameters. Analyzing the obtained results we see that the approximations drastically reduced the storage requirements by 93.4% and 96.7%, and sped up the average evaluation time by 83.7% and 88.4%, respectively for the $n_z = 7$ and $n_z = 6$ cases. The closed-loop trajectories with the proposed $\hat{\pi}(x)$ remained reasonably close to the scenario with the optimal $\pi(x)$, converging to nearby equilibrium points. In practice, steady-state errors are typically counteracted by using disturbance observers, which would require however learning an MPC

controller defined on an extended parameter space (Pannocchia, 2015).

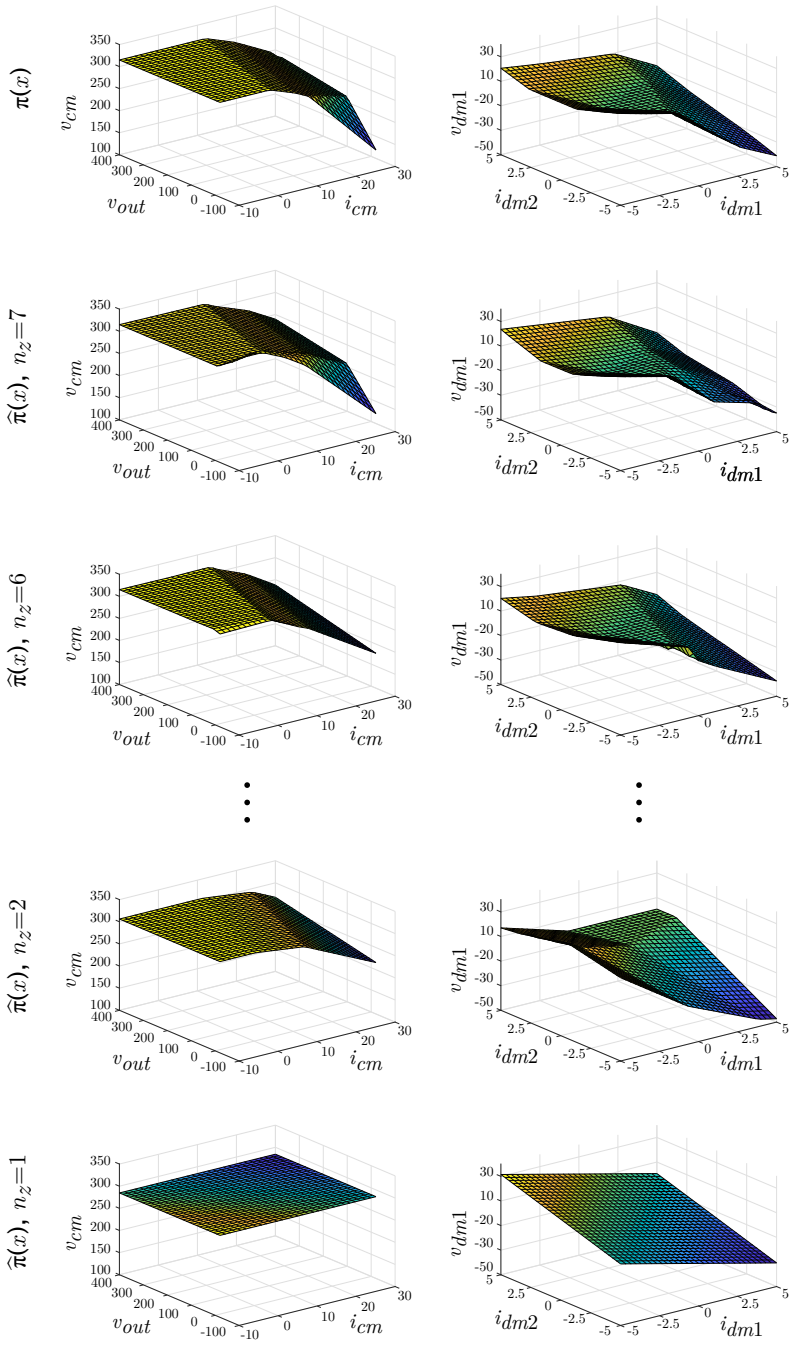

Figure 4.4: Slices of the explicit MPC controller $\pi(x)$ and the pQP NN approximations $\hat{\pi}(x)$ for two control variables: v_{cm} (left) and v_{dm1} (right).

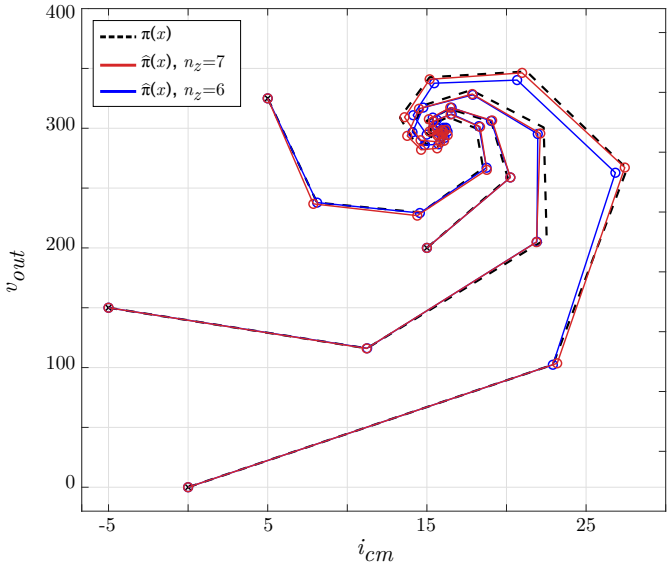

Figure 4.5: Output voltage and common mode current phase portraits under the explicit MPC controller and the two viable pQP NN approximations.

Table 4.2: Explicit MPC and viable pQP NN controllers key features.

Controller	Regions	Storage	Computational time	SS error
$\pi(x)$	2'337	518 kB	12.9 ms	0%
$\hat{\pi}(x)$, $n_z = 7$	107	34 kB	2.1 ms	0.59%
$\hat{\pi}(x)$, $n_z = 6$	56	17 kB	1.5 ms	1.25%

4.5 Experimental validation

We now describe an experimental validation of the proposed pQP NN architecture, again in the context of power electronics. The system under study is a Buck converter, a building block of many switched-mode power supplies. The approximate MPC scheme was trained and then deployed on an inexpensive microcontroller, and the approach lead to an important start-up transient response enhancement.

4.5.1 The system and the target predictive controller

A schematic representation of the Buck converter is shown in Figure 4.6 and its parameters are found in Table 4.3. V_{IN}, V_D, L, and C refer respectively to the input voltage, the diode forward drop, the inductance and the capacitance; whereas

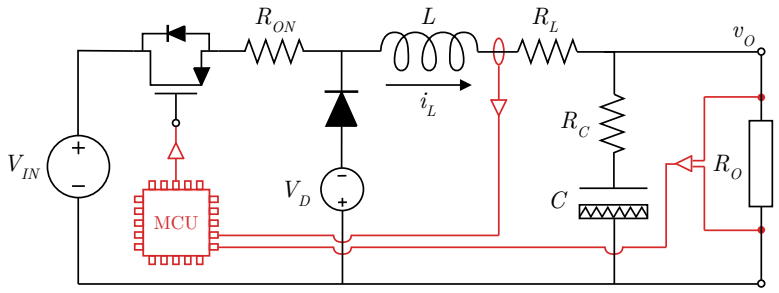

Figure 4.6: A circuit diagram of the Buck converter including its parasitic resistances and the diode forward voltage drop. The feedback loop is closed by the microcontroller that implements our proposed pQP NN scheme.

Table 4.3: Parameters of the DC-DC converter

V_{IN}	V_{OUT}	V_D	L	C	R_{ON}	R_L	R_C	R_O	f_{sw}
15 V	5 V	0.1 V	10 mH	56 μF	5 mΩ	2 Ω	330 mΩ	100 Ω	20 kHz

R_{ON}, R_L, R_C and R_O refer to the switch on-resistance, the inductor parasitic resistance, the capacitor parasitic resistance, and the output load. The purpose of this topology is to supply energy to the load at a voltage level equal or lower than the input source, which is accomplished by modulating the main switch.

We choose as state variables the inductor current and the output voltage

$$x = \begin{bmatrix} x_1 & x_2 \end{bmatrix}^\top = \begin{bmatrix} i_L & v_O \end{bmatrix}^\top \tag{4.16}$$

The power switch is operated at a constant frequency f_{sw} and variable duty cycle, which is taken to be the control variable $u = \delta$. Following the classical time-averaging technique (Middlebrook and Cuk, 1976), Kircchoff's circuit laws are first used to derive differential equations for both when the switch is closed, and when it is open. These are then averaged with δ and $(1 - \delta)$ as weights, yielding

$$\dot{x}_1 = -\frac{R_L}{L}x_1 - \frac{1}{L}x_2 + \frac{V_{IN} + V_D}{L}u - \frac{R_{ON}}{L}x_1 u - \frac{V_D}{L} \tag{4.17a}$$

$$\dot{x}_2 = -\frac{R_C R_O R_L C + R_O L}{(R_C + R_O)LC}x_1 - \frac{R_C R_O C + L}{(R_C + R_O)LC}x_2 + \frac{R_C R_O (V_{IN} + V_D)}{(R_C + R_O)L}u - \frac{R_C R_O R_{ON}}{(R_C + R_O)L}x_1 u - \frac{R_C R_O V_D}{(R_C + R_O)L} \tag{4.17b}$$

The bi-linear expressions above are finally linearized around an equilibrium point, which is chosen by fixing the output voltage to the desired value x_{2eq} and solving

for the current and duty cycle steady-state values

$$x_{1eq} = \frac{x_{2eq}}{R_O} \tag{4.18}$$

$$u_{eq} = \frac{R_O V_D + (R_L + R_O) x_{2eq}}{R_O(V_{IN} + V_D) - R_{ON} x_{2eq}} \tag{4.19}$$

Finally, (4.17a) and (4.17b) are expanded around (x_{1eq}, u_{eq}) and only the linear terms are kept, leaving us with $\dot{x} = A_{ct}x + B_{ct}u$ where

$$A_{ct} = \begin{bmatrix} -\frac{R_L + R_{ON} u_{eq}}{L} & -\frac{1}{L} \\ -\frac{R_C R_O (R_L C - R_{ON} C u_{eq}) + R_O L}{(R_C + R_O) LC} & -\frac{R_C R_O C + L}{(R_C + R_O) LC} \end{bmatrix} \tag{4.20}$$

$$B_{ct} = \begin{bmatrix} \frac{V_{IN} + V_D - R_{ON} x_{1eq}}{L} \\ \frac{R_C R_O (V_{IN} + V_D - R_{ON} x_{1eq})}{(R_C + R_O) L} \end{bmatrix} \tag{4.21}$$

As a last step, a discrete-time model $x_{t+1} = Ax_t + Bu_t$ is obtained by integrating the continuous-time dynamics using the zero-order hold method at $f_{\text{samp}} = 10\,\text{kHz}$.

The controller goal was to attain a fast start-up response[5] with as little overshoot as possible and to regulate the output voltage v_O to $v_{\text{eq}} = 5\,\text{V}$. Furthermore, an inductor maximum current constraint of 200 mA and maximum voltage constraint of 7 V were imposed. The prediction horizon was $N = 10$ and the standard quadratic cost (4.15) was employed with weights $Q = \text{diag}(90, 1)$, $R = 1$, and P as the solution of the associated discrete-time algebraic Ricatti equation. The reference equilibrium values $x_{\text{eq}} = \begin{bmatrix} 0.05 & 5 \end{bmatrix}^\top$, $u_{\text{eq}} = 0.3379$ were used, along with the constraints

$$x^{\min} = \begin{bmatrix} i_L^{\min} & v_O^{\min} \end{bmatrix}^\top = \begin{bmatrix} 0\,\text{mA} & 0\,\text{V} \end{bmatrix}^\top \tag{4.22}$$

$$x^{\max} = \begin{bmatrix} i_L^{\max} & v_O^{\max} \end{bmatrix}^\top = \begin{bmatrix} 200\,\text{mA} & 7\,\text{V} \end{bmatrix}^\top \tag{4.23}$$

$$u^{\min} = 0, \quad u^{\max} = 1 \tag{4.24}$$

and $x_N \in \mathbb{X}_N$, with $\mathbb{X}_N$ being the maximum invariant set for the dynamical system under the corresponding LQR control law.

After computing the explicit solution $\pi(x)$ with the MPT toolbox, the map was found to have 70 regions as shown in Figure 4.7. Although this number may not seem too large for a general purpose computer, it can still be challenging for certain microcontrollers (MCU) to handle, especially since the algorithm has to

[5] The start-up is defined as the system evolution from having zero energy in its storage elements to reaching the desired target output voltage.

be executed in under 100 μs. In fact, pQP NN approximations can be beneficial to the power supply industry since the computing platforms are often MCUs with limited memory and relatively slow clocks.

4.5.2 Learning the optimal controller and deploying the algorithm

The explicit control law $\pi(x)$ was sampled in order to collect a set of state-control pairs for a total of $5'000$ points. A uniform distribution over the set of feasible states was used throughout. In order to achieve a balanced learning over the domain, the currents and voltages values that formed the initial states were normalized to a range of $[0, 1]$ The same software framework, loss function and optimization algorithm from the last example were used with a mini-batch size of 50 and 150 epochs. All pQP weights were initialized randomly. The code was run on a 3.1 GHz Intel Core i7 laptop with 16 GB 2133 MHz of memory, and training the network once took on average 35 mins without any GPU acceleration. The size n_z was gradually increased to improve the fitting results and with $n_z = 3$, after only 5 re-initializations, a low mean squared error training loss was attained: 1.66×10^{-7}.

Both the map and the domain partition of the pQP NN can be seen in Figure 4.7, where the number of regions was reduced from 70 to 6, and the memory required to store the control law, 9.25 kB to 528 B. This last improvement can be particularly useful to fit all the parameters in specialized, small memory slots that are closed to the processor core than the main memory. Visually inspecting the surfaces shown in Figure 4.7, one can note how similar the two are. As pointed out in Section 4.3, the domain of the learned controller is larger than the original one, which can be verified in the partition plots. We underline that the 6 regions of the pQP NN are associated with the pQP layer L2 and that the map $\hat{\pi}(x)$ is composed of other elements such as the linear layers and the saturation element in L4. Finally, closed-loop simulations based on the nominal system model were performed from different initial conditions around the origin to test the learned controller, after which we proceeded to the embedded implementation phase.

A photo of the Buck converter prototype designed and assembled in the Automatic Control Laboratory at EPFL is presented in Figure 4.8, and its parameters are listed in Table 4.3. The current monitoring circuit consisted of a 150 mΩ shunt resistor and a INA282 differential amplifier, whereas the output voltage was simply scaled down by means of a voltage divider with no isolation. In order to control the step-down converter, an STM32L476 was chosen as the embedded platform: the MCU runs at 80 MHz, it has a 32-bit RISC architecture and two independent

12-bit ADC channels. The firmware was written in an interrupt-driven, bare metal fashion and was triggered at 10 kHz. To avoid using raw measurements and also to filter noise, the ADCs were operated at a faster pace and 5 consecutive readings were averaged before being sent to the pQP NN routine. Only integers were used to encode all control law parameters (the polytopes and feedback gains) and the direct memory access peripheral was activated to link the ADC blocks directly to memory. As the PWA description of the pQP layer only featured 6 regions, a simple sequential search was employed. All these steps were needed to ensure that all instructions could be executed within the 100 μs time slot.

The natural open-loop start-up transient is reported in Figure 4.9 (top), which shows peaks of 7.97 V and 330 mA, corresponding respectively to overshoots of 59% and 660%. In Figure 4.9 (middle), how the transient response was improved under the pQP NN control law, with voltage and current peaks of 5.16 V and 202 mA, respectively. By examining the yellow curves, one sees that its initial derivative was reduces from the top to the middle oscilloscope prints, indicating a slower flow of energy from the source to the output capacitor. This was expected as the inductor current had to satisfy the constraints and was capped[6]. Yet, the output voltage settling time was reduced from 6.73 ms to 2.33 ms. Lastly, we have also examined the execution time of the controller routine, responsible for averaging ADC samples, locating the pertinent PWA region, computing the control action and updating the PWM peripheral with the new duty cycle value. The execution period did not show much variance and lasted between 22.0 μs and 27.5 μs as shown by the blue signal in Figure 4.9 (bottom). In a practical scenario, this is rather positive since the MCU core would not be always busy, but would have time available to execute additional side tasks.

4.6 Conclusions and outlook

In this chapter, we put forward a novel neural network architecture featuring a pQP layer to approximate linear MPC controllers. Similarly to the process of computing an explicit MPC control law, a PWA description of the pQP problem can be obtained, and its complexity can be controlled by the user and regarded as a network hyperparameter. As a result, once the architecture is trained, a forward pass simplifies to evaluating a chain of affine and PWA expressions, without requiring solving any optimization problem. Additionally, we showed that the pQP layer is a suitable inductive bias for MPC, indeed, it was shown that any linear

[6]The average dynamical model developed for the converter in Section 4.5.1 predicts the average state values between the ON and the OFF semi-cycles, whereas the 202 mA peak current reported by the oscilloscope are absolute.

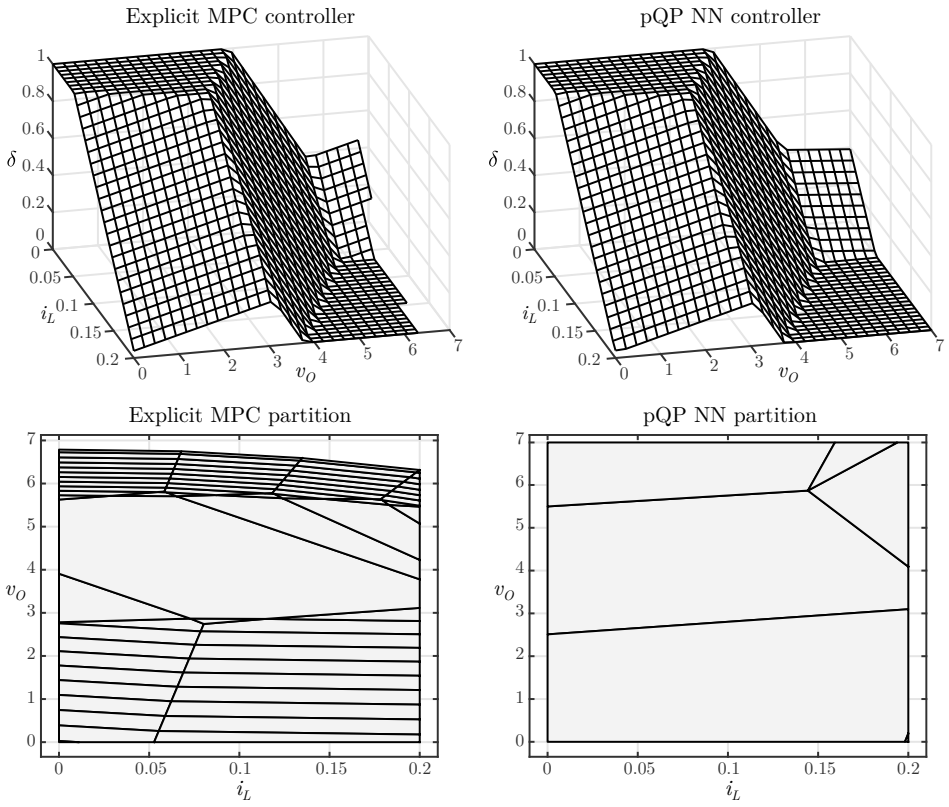

Figure 4.7: The original explicit MPC map and its domain partition (left), and the learned pQP NN map and its domain partition (right).

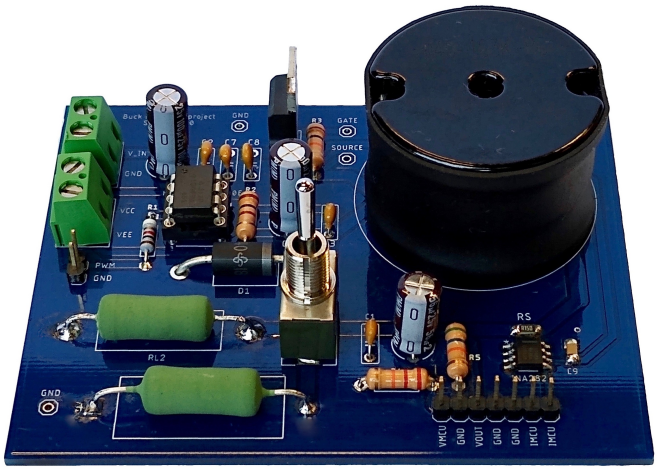

Figure 4.8: A picture of the Buck converter prototype designed and assembled in the Automatic Control Laboratory at EPFL.

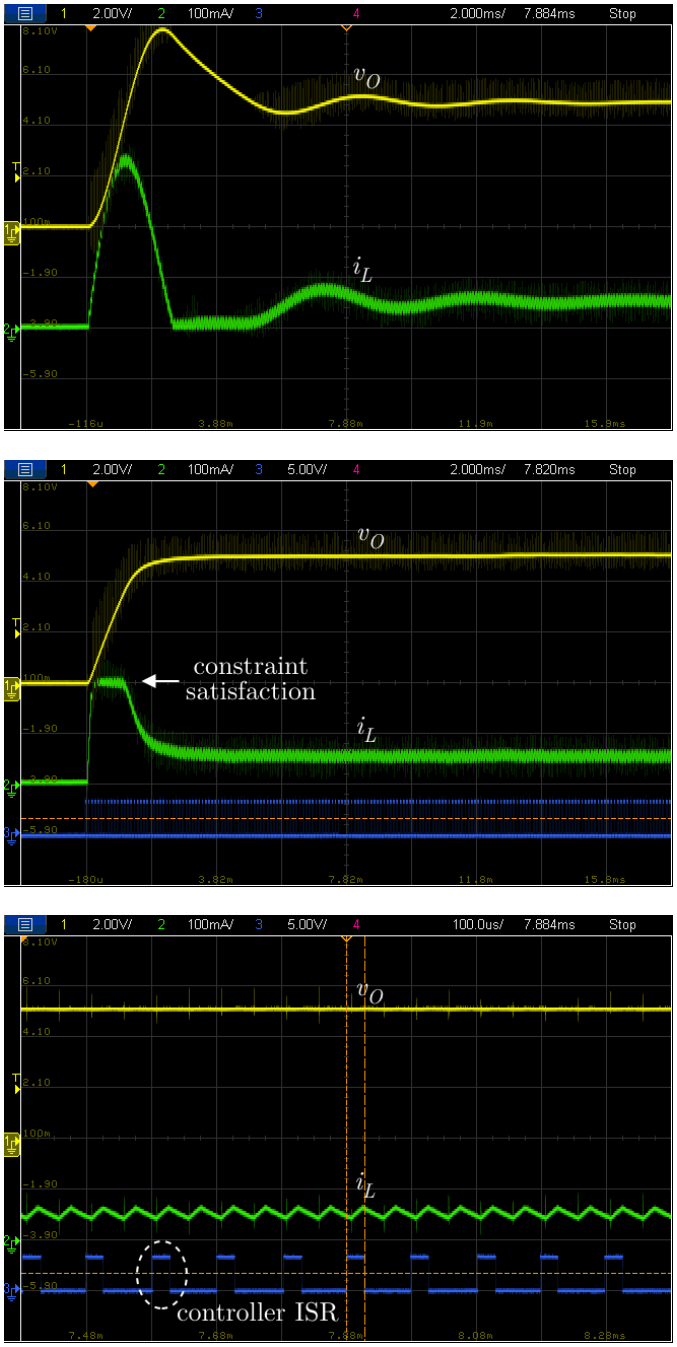

Figure 4.9: Open-loop start-up response (top), closed-loop start-up response (middle), and a close-up view of the closed-loop start-up response highlighting individual switching cycles and the controller interrupt service routine (ISR) execution time of approximately 27 μs.

MPC policy can be learned exactly by the proposed network given a suitable size.

Two application examples in the domain of power electronics were presented, showcasing the merit of the MPC learning scheme. First, through simulations, the effects of the pQP layer size on the fitting error, memory requirements and number of regions was investigated; and the simplified control laws were tested by comparing closed-loop trajectories. Promising results were observed, suggesting that complex PWA functions with thousands of regions can be fit with around one hundred regions without a significant deterioration of the dynamical system response. Second, a smaller-scale problem was tackled, but this time involving a real power supply. The pQP NN was used to reduce by 90% the complexity of the original explicit control law and was subsequently deployed on a low-cost microcontroller. The necessary computations were carried out in under $27\,\mu$s and the transient response was substantially enhanced. Finally, our experiments included a trajectory where a current constraint was activated and not violated by the pQP NN approximate policy.

From a theoretical perspective, one aspect that could be researched in the future is the link between the training cost and the worst-case approximation error. Intuitively, the training loss carries information between the approximation discrepancy at the sample locations. In addition, it is known that both the ground-truth and the approximator are continuous maps over compact sets. Therefore, with an appropriate sampling strategy, out-of-samples error bounds could potentially be established for the pQP NN scheme based solely on the training error. Another direction that could be taken regard is to devise initialization strategies for the pQP layer. In effect, this would amount to finding suitable dual problems that approximate well the MPC dual while being smaller in size. The envisioned advantage is a faster convergence of the network to better solutions.

From a practical viewpoint, we see at least two possible paths to be explored. One concerns the advantages of adopting the pQP NN over a generic deep network (DNN) with ReLU activation functions. We suspect DNNs to require in general a large number of neurons and to result in a final map with a considerably larger number of regions and kinks, whereas we expect our scheme to be more economical. Second, we strongly believe that the problem of approximating mixed-integer MPC problems deserves more attention. In particular, users of the so-called finite-set MPC formulation (Karamanakos and Geyer, 2019) that has become popular in recent years, have often to restrict their prediction horizons to one for computational reasons (Kim et al., 2015). We suspect that modern classifiers could be employed to efficiently learn the optimal control actions from a discrete set in long horizon formulations, unlocking important performance gains.

4.7 Appendix

As mentioned in Section 4.1, a fair number of linear MPC simplification techniques exists, most of which were proposed in the late 2000s and early 2010s. Our goal in this appendix is to compare the pQP NN learning strategy with the methods implemented in the MPT toolbox (Herceg et al., 2013) and that can be easily accessed through the `simplify()` function. The reader is referred to the MPT documentation for details on each method and the associated publications. Herein we will adopt an user's perspective and simply apply each of them.

First, consider the MPC formulation used in our experimental investigation (Section 4.5.1), which has a PWA form $\pi(x)$ composed of 70 regions. Four different simplification techniques were applied to $\pi(x)$ and the resulting number of regions was counted. Additionally, a grid of 6'561 points was laid on the domain to collect new state-control pairs and compute an average mean squared validation error. The results are displayed in Table 4.4 and are compared to the pQP NN with $n_z = 3$ shown in Figure 4.9. The `clipping` method returned a controller with less than haft of the original complexity, but with comparatively high validation error. The `greedy` strategy lead to a map with only 20 regions and virtually no validation error. The last three approaches could be considered as equally competitive since they delivered maps with a low number of regions and low validation error.

Next, the MPC formulation was modified with weight matrices $Q = \text{diag}(1, 1)$, $R = 100$ and a horizon of $N =$100. In this case, the explicit MPC controller $\pi(x)$ had 189 regions. This second batch of results is displayed in Table 4.5. As can be seen, most of the MPT techniques failed to simplify $\pi(x)$ and essentially preserved the original number of regions, hence the low validation error. The `fitting` approach was the only one able to significantly reduce the map complexity while attaining a good fit. The pQP NN, on the other hand, lead to a PWA function with the same 5 partitions and less than half of the validation error when compared to the `fitting` scheme, although in the same order of magnitude.

From our numerical experiments, we have observed that some simplification schemes rely on features that are often observed in simplified case studies such as clear saturation regions, or neighboring parts sharing the same affine gains. On the other hand, the pQP NN is a more general approximator that does not explicitly make use of that information. As a result, the pQP NN tends to return good results in a large number of cases, whereas the specialized techniques sometimes work well, but fail in other situations. Nevertheless, it is fair to mention that training a single pQP network took around 30 minutes, whereas executing any of the `simplify()` methods, less than 30 seconds.

Table 4.4: Complexity reduction results for an explicit MPC control law $\pi(x)$ with 70 regions.

MPT method	num. regions	validation error
`clipping`	25	$1.49\,10^{-4}$
`greedy`	20	$< 10^{-20}$
`separation`	7	$< 10^{-20}$
`fitting`	5	$3.03\,10^{-8}$
pQP NN	6	$6.07\,10^{-5}$

Table 4.5: Complexity reduction results for an explicit MPC control law $\pi(x)$ with 189 regions.

MPT method	num. regions	validation error
`clipping`	187	$6.31\,10^{-8}$
`greedy`	183	$< 10^{-20}$
`separation`	187	$< 10^{-20}$
`fitting`	5	$3.71\,10^{-4}$
pQP NN	5	$1.75\,10^{-4}$

A Elements of analysis and geometry

For a comprehensive presentation of the concepts, the reader os referred to Searcóid and Searcoid (2002); Pugh (2002).

All vector spaces herein are defined over the field of real numbers $\mathbb{R}$.

Definition 6. (Metric space) A metric space is a vector space $(V, +, \times)$ equipped with a map $d(\cdot, \cdot) : V \times V \to \mathbb{R}$ called a *metric* satisfying

$$(i)\, d(v, v) \geq 0 \tag{A.1}$$
$$(ii)\, d(v, w) = 0 \Leftrightarrow v = w \tag{A.2}$$
$$(iii)\, d(v, w) = d(w, v) \tag{A.3}$$
$$(iv)\, d(v, w) \leq d(v, z) + d(z, w) \tag{A.4}$$

for any $v, w, z \in V$.

For simplicity, we write (V, d) instead of $(V, +, \times, d)$.

Definition 7. (Convergent sequence) Given a metric space (X, d), a sequence $\{x_n\}_{n\in\mathbb{N}}$ in X is said to *converge* to an element $x \in X$ if

$$\forall \epsilon > 0 \, : \, \exists N \in \mathbb{N} \, : \, \forall n \geq N \, : \, d(x_n, x) < \epsilon \tag{A.5}$$

Convergent sequences are usually written $\lim_{n\to\infty} x_n = x$ or more simply $x_n \to x$. Moreover, sequences cannot converge to two or more points.

Definition 8. (Cauchy sequence) Given a metric space (X, d), a sequence $\{x_n\}_{n\in\mathbb{N}}$ in X is said to be *Cauchy* if

$$\forall \epsilon > 0 \, : \, \exists N \in \mathbb{N} \, : \, \forall n, m \geq N \, : \, d(x_n, x_m) < \epsilon \tag{A.6}$$

Cauchy sequences are a superset of convergent sequences.

Definition 9. (Complete space) A metric space is $(V, +, \times, d)$ is said to be complete if every Cauchy sequence $\{x_n\}_{n\in\mathbb{N}}$ converges to an element $x \in X$.

Definition 10. (Normed space) A normed space is a vector space $(V, +, \times)$ equipped with a map $\|\cdot\| : V \to \mathbb{R}$ called a *norm* satisfying

$$(i)\ \|v\| \geq 0 \tag{A.7}$$
$$(ii)\ \|v\| = 0 \Leftrightarrow v = 0 \tag{A.8}$$
$$(iii)\ \|\alpha v\| = |\alpha|\,\|v\| \tag{A.9}$$
$$(iv)\ \|v + w\| \leq \|v\| + \|w\| \tag{A.10}$$

for any $v, w \in V$ and any $\alpha \in \mathbb{R}$.

Metrics can be defined via norms through $d(x, y) := \|x - y\|$. As a result, every normed space is a metric space.

Definition 11. (Banach space) A normed space $(X, \|\cdot\|)$ is called a Banach space if it is complete.

Definition 12. (Inner-product space) An inner-product space is a vector space $(X, +, \times)$ equipped with a map $\langle\cdot,\cdot\rangle : X \times X \to \mathbb{R}$ called an *inner-product* satisfying

$$(i)\ \langle x, y\rangle = \langle y, x\rangle \tag{A.11}$$
$$(ii)\ \langle \alpha x + \beta y, z\rangle = \alpha\langle x, z\rangle + \beta\langle y, z\rangle \tag{A.12}$$
$$(iii)\ \langle x, x\rangle \geq 0 \tag{A.13}$$
$$(iv)\ \langle x, x\rangle = 0 \Leftrightarrow x = 0 \tag{A.14}$$

for any $v, w \in V$ and any $\alpha \in \mathbb{R}$.

Norms can be defined via inner-products through $\|x\| := \sqrt{\langle x, x\rangle}$. As a result, every inner-product space is also a normed space.

Definition 13. (Hilbert space) An inner-product space $(X, \langle\cdot,\cdot\rangle)$ is called a Hilbert space if it is complete.

Definition 14. (Bounded linear operator) Let $(V, \|\cdot\|_V)$ and $(W, \|\cdot\|_W)$ be two Banach spaces. A map $A : V \mapsto W$ is said to be a bounded linear operator if

$$\sup_{v\in V\setminus\{0\}} \frac{\|Av\|_W}{\|v\|_V} < \infty \tag{A.15}$$

Definition 15. (Operator norm) Let $A : V \mapsto W$ be a bounded linear operator. The operator norm is defined as

$$\|A\| := \sup_{v \in V \setminus \{0\}} \frac{\|Av\|_W}{\|v\|_V} \tag{A.16}$$

Definition 16. (Pointwise convergence) Let X, Y be two metric spaces and $\{f_n\}_{n \in \mathbb{N}}$ be a sequence of functions where $f_n : X \to Y$ for all n. The sequence is said to converge to a function $f : X \to Y$ if for every $x \in X$

$$\lim_{n \to \infty} f_n(x) = f(x) \tag{A.17}$$

The example below, taken from (Berlinet and Thomas-Agnan, 2011, §1), highlights an issue one has to pay attention to when working with spaces of functions.

Example 1. (Convergence does not imply pointwise convergence) Let P be the vector space of all polynomials over $[0, 1]$ and endow it with the norm

$$\|f\|_P = \left(\int_0^1 |f(x)|^2 \mathrm{d}x \right)^{1/2} \tag{A.18}$$

The sequence $\{p_n\}_{n \in \mathbb{N}}$, $p_n(x) = x^n$ converges to the zero function since

$$\lim_{n \to \infty} \|p_n - 0\|_P = \lim_{n \to \infty} \left(\int_0^1 x^{2n} \mathrm{d}x \right)^{1/2} \tag{A.19}$$

$$= \lim_{n \to \infty} \frac{1}{\sqrt{2n+1}} \tag{A.20}$$

$$= 0 \tag{A.21}$$

and yet $p_n(1) = 1, \forall n$, i.e., $|p_n(x) - 0(x)| \nrightarrow 0$.

Definition 17. (Span) Let X be a vector space and $B \subseteq X$ be a subset of it. The *span* of B is defined as the set

$$\operatorname{span} B = \left\{ \sum_{i=1}^{n} \lambda_i b_i \mid \lambda_i \in \mathbb{R}, b_i \in B, n \in \mathbb{N} \right\} \tag{A.22}$$

Definition 18. (Linear independence) Let X be a vector space and $B \subseteq X$ be a subset of it. B is said to be linearly independent if for every finite subset $\{b\}_{i=1}^n \subseteq B$, $\sum_{i=1}^n \lambda_i b_i = 0 \iff \lambda_1 = \cdots = \lambda_n = 0$.

Definition 19. (Hamel basis) Let X be a vector space and $B \subseteq X$. B is called a Hamel basis for X if B is linearly independent and $\operatorname{span} B = X$.

Proposition 16. Every vector space has a Hamel basis.

Proposition 17. All Hamel bases of a vector space have the same cardinality.

The concept of a Hamel basis is aligned with the more specific concept of a "basis" in finite-dimensional vector spaces.

Definition 20. (Dimension of a vector space) The dimension of a vector space denoted $\dim X$ is the cardinality of any Hamel basis B of X. If any B is not finite, X is said to be infinite-dimensional.

Proposition 18. Let $A \in \mathbb{R}^{d\times d}$ be an invertible matrix, $B \in \mathbb{R}^d$ and $c \in \mathbb{R}$. The following identity holds

$$\begin{bmatrix} A & B \\ B^\top & c \end{bmatrix}^{-1} = \begin{bmatrix} A^{-1} + \frac{1}{d}A^{-1}BB^\top A^{-1} & -\frac{1}{d}A^{-1}B \\ -\frac{1}{d}B^\top A^{-1} & \frac{1}{d} \end{bmatrix} \tag{A.23}$$

where $d = c - B^\top A^{-1}B$.

Definition 21. (Polyhedron) A polyhedron $P \subseteq \mathbb{R}^n$ is a set $P = \{x \in \mathbb{R}^n : p_i(x) \leq 0, i = 1, \ldots, n_P\}$ with $n_P < \infty$ and $p_i(x)$ being are affine functions.

Definition 22. (Polytope) A polytope $P \subset \mathbb{R}^n$ is a bounded polyhedron.

Definition 23. (Polyhedral projection) Let $P \subset \mathbb{R}^n$ be a polyhedron described by $\{x \in \mathbb{R}^n \,|\, Hx \leq h\}$ for some $H \in \mathbb{R}^{n\times m}$ and $h \in \mathbb{R}^m$. The projection of a point $z \in \mathbb{R}^n$ onto P is defined as the vector $p^\star \in \mathbb{R}^n$

$$\begin{aligned} p^\star = \underset{p\in\mathbb{R}^n}{\arg\min}\ & \|p - z\|_2^2 \\ \text{subj. to } & Hp \leq h \end{aligned} \tag{A.24}$$

and it is usually denoted as $\mathrm{Proj}_P(x)$.

Bibliography

Agrawal, A., Amos, B., Barratt, S., Boyd, S., Diamond, S., and Kolter, J. Z. (2019). Differentiable convex optimization layers. *Advances in neural information processing systems*, 32.

Alessio, A. and Bemporad, A. (2009). A survey on explicit model predictive control. In *Nonlinear model predictive control*, pages 345–369. Springer.

Amos, B. and Kolter, J. Z. (2017). Optnet: Differentiable optimization as a layer in neural networks. In *International Conference on Machine Learning*, pages 136–145. PMLR.

Andersson, J. A., Gillis, J., Horn, G., Rawlings, J. B., and Diehl, M. (2019). Casadi: a software framework for nonlinear optimization and optimal control. *Mathematical Programming Computation*, 11(1):1–36.

Bauer, M., van der Wilk, M., and Rasmussen, C. E. (2016). Understanding probabilistic sparse gaussian process approximations. *Advances in neural information processing systems*, 29.

Beaglehole, D., Belkin, M., and Pandit, P. (2022). Kernel ridgeless regression is inconsistent for low dimensions. *arXiv preprint arXiv:2205.13525*.

Bechtel, M. (2022). Opening up to AI: Learning to trust our AI colleagues.

Beckers, T. and Hirche, S. (2016). Stability of gaussian process state space models. In *2016 European Control Conference (ECC)*, pages 2275–2281. IEEE.

Belkin, M., Hsu, D., Ma, S., and Mandal, S. (2019). Reconciling modern machine-learning practice and the classical bias–variance trade-off. *Proceedings of the National Academy of Sciences*, 116(32):15849–15854.

Belkin, M., Ma, S., and Mandal, S. (2018). To understand deep learning we need to understand kernel learning. In *International Conference on Machine Learning*, pages 541–549. PMLR.

Bemporad, A., Morari, M., Dua, V., and Pistikopoulos, E. N. (2002). The explicit linear quadratic regulator for constrained systems. *Automatica*, 38(1):3–20.

Berlinet, A. and Thomas-Agnan, C. (2011). *Reproducing kernel Hilbert spaces in probability and statistics*. Springer Science & Business Media.

Berntorp, K. (2021). Online bayesian inference and learning of gaussian-process state–space models. *Automatica*, 129:109613.

Bertsekas, D. (2009). *Convex optimization theory*, volume 1. Athena Scientific.

Binois, M., Gramacy, R. B., and Ludkovski, M. (2018). Practical heteroscedastic gaussian process modeling for large simulation experiments. *Journal of Computational and Graphical Statistics*, 27(4):808–821.

Borrelli, F., Bemporad, A., and Morari, M. (2017). *Predictive control for linear and hybrid systems*. Cambridge University Press.

Boyd, S., Parikh, N., and Chu, E. (2011). *Distributed optimization and statistical learning via the alternating direction method of multipliers*. Now Publishers Inc.

Boyd, S. P. and Vandenberghe, L. (2004). *Convex optimization*. Cambridge university press.

Brunke, L., Greeff, M., Hall, A. W., Yuan, Z., Zhou, S., Panerati, J., and Schoellig, A. P. (2022). Safe learning in robotics: From learning-based control to safe reinforcement learning. *Annual Review of Control, Robotics, and Autonomous Systems*, 5:411–444.

Bui, T. D., Nguyen, C., and Turner, R. E. (2017a). Streaming sparse gaussian process approximations. *Advances in Neural Information Processing Systems*, 30.

Bui, T. D., Yan, J., and Turner, R. E. (2017b). A unifying framework for gaussian process pseudo-point approximations using power expectation propagation. *The Journal of Machine Learning Research*, 18(1):3649–3720.

Bünning, F., Huber, B., Heer, P., Aboudonia, A., and Lygeros, J. (2020). Experimental demonstration of data predictive control for energy optimization and thermal comfort in buildings. *Energy and Buildings*, 211:109792.

Bünning, F., Huber, B., Schalbetter, A., Aboudonia, A., de Badyn, M. H., Heer, P., Smith, R. S., and Lygeros, J. (2022). Physics-informed linear regression is competitive with two machine learning methods in residential building mpc. *Applied Energy*, 310:118491.

Chachuat, B., Srinivasan, B., and Bonvin, D. (2009). Adaptation strategies for real-time optimization. *Computers & Chemical Engineering*, 33(10):1557–1567.

Chakrabarty, A., Maddalena, E., Qiao, H., and Laughman, C. (2021). Scalable bayesian optimization for model calibration: Case study on coupled building and hvac dynamics. *Energy and Buildings*, 253:111460.

Christophersen, F. J., Zeilinger, M. N., Jones, C. N., and Morari, M. (2007). Controller complexity reduction for piecewise affine systems through safe region elimination. In *2007 46th IEEE Conference on Decision and Control*, pages 4773–4778. IEEE.

Chui, M., Roberts, R., and Yee, L. (2022). Generative AI is here: How tools like ChatGPT could change your business.

Coulson, J., Lygeros, J., and Dörfler, F. (2019). Data-enabled predictive control: In the shallows of the deepc. In *2019 18th European Control Conference (ECC)*, pages 307–312. IEEE.

Csató, L. and Opper, M. (2002). Sparse on-line gaussian processes. *Neural computation*, 14(3):641–668.

Di Natale, L., Lian, Y., Maddalena, E. T., Shi, J., and Jones, C. N. (2022a). Lessons learned from data-driven building control experiments: Contrasting gaussian process-based mpc, bilevel deepc, and deep reinforcement learning. *arXiv preprint arXiv:2205.15703*.

Di Natale, L., Svetozarevic, B., Heer, P., and Jones, C. N. (2022b). Physically consistent neural networks for building thermal modeling: theory and analysis. *Applied Energy*, 325:119806.

Diwale, S. S., Lymperopoulos, I., and Jones, C. N. (2014). Optimization of an airborne wind energy system using constrained gaussian processes. In *2014 IEEE Conference on Control Applications (CCA)*, pages 1394–1399. IEEE.

Drgoňa, J., Arroyo, J., Figueroa, I. C., Blum, D., Arendt, K., Kim, D., Ollé, E. P., Oravec, J., Wetter, M., Vrabie, D. L., et al. (2020). All you need to know about model predictive control for buildings. *Annual Reviews in Control*, 50:190–232.

Du, Y., Zandi, H., Kotevska, O., Kurte, K., Munk, J., Amasyali, K., Mckee, E., and Li, F. (2021). Intelligent multi-zone residential hvac control strategy based on deep reinforcement learning. *Applied Energy*, 281:116117.

Eleftheriadis, S., Nicholson, T., Deisenroth, M., and Hensman, J. (2017). Identification of gaussian process state space models. *Advances in neural information processing systems*, 30.

Fabietti, L., Gorecki, T. T., Qureshi, F. A., Bitlislioğlu, A., Lymperopoulos, I., and Jones, C. N. (2016). Experimental implementation of frequency regulation services using commercial buildings. *IEEE Transactions on Smart Grid*, 9(3):1657–1666.

Fabietti, L., Qureshi, F. A., Gorecki, T. T., Salzmann, C., and Jones, C. N. (2018). Multi-time scale coordination of complementary resources for the provision of ancillary services. *Applied energy*, 229:1164–1180.

Fasshauer, G. E. (2011). Positive definite kernels: past, present and future. *Dolomites Research Notes on Approximation*, 4:21–63.

Fazlyab, M., Morari, M., and Pappas, G. J. (2020). Safety verification and robustness analysis of neural networks via quadratic constraints and semidefinite programming. *IEEE Transactions on Automatic Control*.

Ferrari-Trecate, G., Williams, C., and Opper, M. (1998). Finite-dimensional approximation of gaussian processes. *Advances in neural information processing systems*, 11.

Galimberti, C. L., Furieri, L., Xu, L., and Ferrari-Trecate, G. (2021). Hamiltonian deep neural networks guaranteeing non-vanishing gradients by design. *arXiv preprint arXiv:2105.13205*.

Geyer, T., Torrisi, F. D., and Morari, M. (2008). Optimal complexity reduction of polyhedral piecewise affine systems. *Automatica*, 44(7):1728–1740.

Girard, A. (2004). *Approximate methods for propagation of uncertainty with Gaussian process models*. University of Glasgow.

Goldberg, P., Williams, C., and Bishop, C. (1997). Regression with input-dependent noise: A gaussian process treatment. *Advances in neural information processing systems*, 10.

Gramacy, R. B. (2020). *Surrogates: Gaussian process modeling, design, and optimization for the applied sciences*. Chapman and Hall/CRC.

Gray, F. M. and Schmidt, M. (2016). Thermal building modelling using gaussian processes. *Energy and Buildings*, 119:119–128.

Hale, J. (2019). More than 500 hours of content are now being uploaded to YouTube every minute.

Hansen, J., Murray-Smith, R., and Johansen, T. A. (2005). Nonparametric identification of linearizations and uncertainty using gaussian process models-application to robust wheel slip control. In *Proceedings of the 44th IEEE Conference on Decision and Control*, pages 5083–5088. IEEE.

Herceg, M., Kvasnica, M., Jones, C. N., and Morari, M. (2013). Multi-parametric toolbox 3.0. In *2013 European control conference (ECC)*, pages 502–510. IEEE.

Hertneck, M., Köhler, J., Trimpe, S., and Allgöwer, F. (2018). Learning an approximate model predictive controller with guarantees. *IEEE Control Systems Letters*, 2(3):543–548.

Hewing, L., Kabzan, J., and Zeilinger, M. N. (2019). Cautious model predictive control using gaussian process regression. *IEEE Transactions on Control Systems Technology*, 28(6):2736–2743.

Hewing, L., Wabersich, K. P., Menner, M., and Zeilinger, M. N. (2020). Learning-based model predictive control: Toward safe learning in control. *Annual Review of Control, Robotics, and Autonomous Systems*, 3:269–296.

Hüllermeier, E. and Waegeman, W. (2021). Aleatoric and epistemic uncertainty in machine learning: An introduction to concepts and methods. *Machine Learning*, 110:457–506.

IEA, I. E. A. (2022). Roadmap for energy-efficient buildings and construction in asean.

Iske, A. (2018). *Approximation theory and algorithms for data analysis*. Springer.

Joe, J. (2022). Investigation on pre-cooling potential of ufad via model-based predictive control. *Energy and Buildings*, 259:111898.

Jones, C. N. and Morari, M. (2008). The double description method for the approximation of explicit mpc control laws. In *2008 47th IEEE Conference on Decision and Control*, pages 4724–4730. IEEE.

Kanagawa, M., Hennig, P., Sejdinovic, D., and Sriperumbudur, B. K. (2018). Gaussian processes and kernel methods: A review on connections and equivalences. *arXiv preprint arXiv:1807.02582*.

Karamanakos, P. and Geyer, T. (2019). Guidelines for the design of finite control set model predictive controllers. *IEEE Transactions on Power Electronics*, 35(7):7434–7450.

Karg, B. and Lucia, S. (2020). Efficient representation and approximation of model predictive control laws via deep learning. *IEEE Transactions on Cybernetics*, 50(9):3866–3878.

Karvonen, T. (2022). Error bounds and the asymptotic setting in kernel-based approximation. *Dolomites Research Notes on Approximation*, 15(3).

Khosravi, M., Koenig, C., Maier, M., Smith, R. S., Lygeros, J., and Rupenyan, A. (2022). Safety-aware cascade controller tuning using constrained bayesian optimization. *IEEE Transactions on Industrial Electronics*.

Kidger, P. and Lyons, T. (2020). Universal approximation with deep narrow networks. In *Conference on learning theory*, pages 2306–2327. PMLR.

Kim, S.-K., Choi, D.-K., Lee, K.-B., and Lee, Y. I. (2015). Offset-free model predictive control for the power control of three-phase ac/dc converters. *IEEE Transactions on Industrial Electronics*, 62(11):7114–7126.

Kimeldorf, G. and Wahba, G. (1971). Some results on tchebycheffian spline functions. *Journal of mathematical analysis and applications*, 33(1):82–95.

Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Kocijan, J., Murray-Smith, R., Rasmussen, C. E., and Likar, B. (2003). Predictive control with gaussian process models. In *The IEEE Region 8 EUROCON 2003. Computer as a Tool.*, volume 1, pages 352–356. IEEE.

Koller, T., Berkenkamp, F., Turchetta, M., and Krause, A. (2018). Learning-based model predictive control for safe exploration. In *IEEE Conference on Decision and Control (CDC)*, pages 6059–6066. IEEE.

Kumar, P., Rawlings, J. B., and Wright, S. J. (2021). Industrial, large-scale model predictive control with structured neural networks. *Computers & Chemical Engineering*, 150:107291.

Lederer, A., Conejo, A. J. O., Maier, K. A., Xiao, W., Umlauft, J., and Hirche, S. (2021). Gaussian process-based real-time learning for safety critical applications. In *International Conference on Machine Learning*, pages 6055–6064. PMLR.

Lederer, A., Yang, Z., Jiao, J., and Hirche, S. (2022). Cooperative control of uncertain multi-agent systems via distributed gaussian processes. *IEEE Transactions on Automatic Control*.

Lian, Y., Shi, J., Koch, M., and Jones, C. N. (2021). Adaptive robust data-driven building control via bi-level reformulation: an experimental result. *arXiv preprint arXiv:2106.05740*.

Liu, M., Chowdhary, G., Da Silva, B. C., Liu, S.-Y., and How, J. P. (2018). Gaussian processes for learning and control: A tutorial with examples. *IEEE Control Systems Magazine*, 38(5):53–86.

Maddalena, E. T., Galvão, R. K. H., and Afonso, R. J. M. (2019). Robust region elimination for piecewise affine control laws. *Automatica*, 99:333–337.

Maddalena, E. T., Moraes, C. d. S., Waltrich, G., and Jones, C. N. (2020). A neural network architecture to learn explicit mpc controllers from data. *IFAC-PapersOnLine*, 53(2):11362–11367.

Maddalena, E. T., Scharnhorst, P., and Jones, C. N. (2021a). Deterministic error bounds for kernel-based learning techniques under bounded noise. *Automatica*, 134:109896.

Maddalena, E. T., Specq, M. W., Wisniewski, V. L., and Jones, C. N. (2021b). Embedded pwm predictive control of dc-dc power converters via piecewise-affine neural networks. *IEEE Open Journal of the Industrial Electronics Society*, 2:199–206.

Maddox, W. J., Stanton, S., and Wilson, A. G. (2021). Conditioning sparse variational gaussian processes for online decision-making. *Advances in Neural Information Processing Systems*, 34:6365–6379.

Mariéthoz, S. and Morari, M. (2008). Explicit model-predictive control of a pwm inverter with an lcl filter. *IEEE Transactions on Industrial Electronics*, 56(2):389–399.

Matthews, A. G. d. G., Van Der Wilk, M., Nickson, T., Fujii, K., Boukouvalas, A., León-Villagrá, P., Ghahramani, Z., and Hensman, J. (2017). Gpflow: A gaussian process library using tensorflow. *J. Mach. Learn. Res.*, 18(40):1–6.

McHutchon, A. and Rasmussen, C. (2011). Gaussian process training with input noise. *Advances in Neural Information Processing Systems*, 24.

Micchelli, C. A., Xu, Y., and Zhang, H. (2006). Universal kernels. *Journal of Machine Learning Research*, 7(12).

Middlebrook, R. D. and Cuk, S. (1976). A general unified approach to modelling switching-converter power stages. In *1976 IEEE power electronics specialists conference*, pages 18–34. IEEE.

Milanese, M. and Novara, C. (2004). Set membership identification of nonlinear systems. *Automatica*, 40(6):957–975.

Naus, G., Ploeg, J., Van de Molengraft, M., Heemels, W., and Steinbuch, M. (2010). Design and implementation of parameterized adaptive cruise control: An explicit model predictive control approach. *Control Engineering Practice*, 18(8):882–892.

Nghiem, T. X. and Jones, C. N. (2017). Data-driven demand response modeling and control of buildings with gaussian processes. In *2017 American Control Conference (ACC)*, pages 2919–2924. IEEE.

Nikan, O., Golbabai, A., Machado, J. T., and Nikazad, T. (2022). Numerical approximation of the time fractional cable model arising in neuronal dynamics. *Engineering with Computers*, 38(1):155–173.

Novara, C., Nicolì, A., and Calafiore, G. C. (2022). Nonlinear system identification in sobolev spaces. *International Journal of Control*, pages 1–16.

Oldewurtel, F., Parisio, A., Jones, C. N., Gyalistras, D., Gwerder, M., Stauch, V., Lehmann, B., and Morari, M. (2012). Use of model predictive control and weather forecasts for energy efficient building climate control. *Energy and Buildings*, 45:15–27.

Pan, S. J. and Yang, Q. (2010). A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359.

Pannocchia, G. (2015). Offset-free tracking mpc: A tutorial review and comparison of different formulations. In *2015 European control conference (ECC)*, pages 527–532. IEEE.

Parisini, T. and Zoppoli, R. (1995). A receding-horizon regulator for nonlinear systems and a neural approximation. *Automatica*, 31(10):1443–1451.

Paulson, J. A. and Mesbah, A. (2020). Approximate closed-loop robust model predictive control with guaranteed stability and constraint satisfaction. *IEEE Control Systems Letters*, 4(3):719–724.

Pugh, C. C. (2002). *Real mathematical analysis*. Springer.

Quinonero-Candela, J. and Rasmussen, C. E. (2005). A unifying view of sparse approximate gaussian process regression. *The Journal of Machine Learning Research*, 6:1939–1959.

Sabug Jr, L., Ruiz, F., and Fagiano, L. (2021). Smgo: A set membership approach to data-driven global optimization. *Automatica*, 133:109890.

Sarra, S. A. (2005). Adaptive radial basis function methods for time dependent partial differential equations. *Applied Numerical Mathematics*, 54(1):79–94.

Schaback, R. and Wendland, H. (2006). Kernel techniques: from machine learning to meshless methods. *Acta numerica*, 15:543–639.

Schölkopf, B., Herbrich, R., and Smola, A. J. (2001). A generalized representer theorem. In *International conference on computational learning theory*, pages 416–426. Springer.

Schölkopf, B. and Smola, A. J. (2002). *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press.

Schwan, R., Jones, C. N., and Kuhn, D. (2022). Stability verification of neural network controllers using mixed-integer programming. *arXiv preprint arXiv:2206.13374*.

Searcóid, M. Ó. and Searcoid, M. (2002). *Elements of abstract analysis*. Springer.

Sejdinovic, D. and Gretton, A. (2012). What is an RKHS? *Lecture Notes*.

Shahriari, B., Swersky, K., Wang, Z., Adams, R. P., and De Freitas, N. (2015). Taking the human out of the loop: A review of bayesian optimization. *Proceedings of the IEEE*, 104(1):148–175.

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., et al. (2016). Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484–489.

Snoek, J., Larochelle, H., and Adams, R. P. (2012). Practical bayesian optimization of machine learning algorithms. *Advances in neural information processing systems*, 25.

Steinwart, I. (2020). Reproducing kernel hilbert spaces cannot contain all continuous functions on a compact metric space. *arXiv preprint arXiv:2002.03171*.

Steinwart, I. and Christmann, A. (2008). *Support vector machines*. Springer Science & Business Media.

Stluka, P., Parthasarathy, G., Gabel, S., and Samad, T. (2018). Architectures and algorithms for building automation—an industry view. In *Intelligent building control systems*, pages 11–43. Springer.

Sturzenegger, D., Gyalistras, D., Morari, M., and Smith, R. S. (2015). Model predictive climate control of a swiss office building: Implementation, results, and cost–benefit analysis. *IEEE Transactions on Control Systems Technology*, 24(1):1–12.

Turner, R., Deisenroth, M., and Rasmussen, C. (2010). State-space inference and learning with gaussian processes. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pages 868–875. JMLR Workshop and Conference Proceedings.

United Nations Environment Programme (2022). 2022 global status report for buildings and construction: Towards a zero-emission, efficient and resilient buildings and construction sector.

Van Every, P. M., Rodriguez, M., Jones, C. B., Mammoli, A. A., and Martínez-Ramón, M. (2017). Advanced detection of hvac faults using unsupervised svm novelty detection and gaussian process models. *Energy and Buildings*, 149:216–224.

Váňa, Z., Cigler, J., Širokỳ, J., Žáčeková, E., and Ferkl, L. (2014). Model-based energy efficient control applied to an office building. *Journal of Process Control*, 24(6):790–797.

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., et al. (2020). Scipy 1.0: fundamental algorithms for scientific computing in python. *Nature methods*, 17(3):261–272.

Wabersich, K. P. and Zeilinger, M. N. (2018). Linear model predictive safety certification for learning-based control. In *2018 IEEE Conference on Decision and Control (CDC)*, pages 7130–7135. IEEE.

Wen, C., Ma, X., and Ydstie, B. E. (2009). Analytical expression of explicit mpc solution via lattice piecewise-affine function. *Automatica*, 45(4):910–917.

Wendland, H. (2004). *Scattered data approximation*. Cambridge university press.

Williams, C. K. and Rasmussen, C. E. (2006). *Gaussian Processes For Machine Learning*. MIT press Cambridge, MA.

Wilson, J., Hutter, F., and Deisenroth, M. (2018). Maximizing acquisition functions for bayesian optimization. *Advances in neural information processing systems*, 31.

Wright, S. J. (2019). Efficient convex optimization for linear mpc. In *Handbook of model predictive control*, pages 287–303. Springer.

Yang, T., Li, Y.-F., Mahdavi, M., Jin, R., and Zhou, Z.-H. (2012). Nyström method vs random fourier features: A theoretical and empirical comparison. *Advances in neural information processing systems*, 25.

Zhang, Y., Duchi, J., and Wainwright, M. (2013). Divide and conquer kernel ridge regression. In *Conference on Learning Theory (COLT)*, pages 592–617. PMLR.

Zhou, K. and Doyle, J. C. (1998). *Essentials of robust control*, volume 104. Prentice hall Upper Saddle River, NJ.

Emilio Tanowe Maddalena

Rue du Débarcadère, 51. 2503 Bienne, Switzerland
+41 76 408-5990 emilio.tanowe.maddalena@gmail.com
Nationalities: Italian and Brazilian
Linkedin GitHub Scholar

Education

2018–2023 **École polytechnique fédérale de Lausanne (EPFL)** Lausanne, Switzerland
Ph.D. in Electrical Engineering

2016–2018 **Brazilian Institute of Aeronautics (ITA)** São José dos Campos, Brazil
M.Sc. in Systems and Control

2010–2015 **Universidade Federal de Mato Grosso do Sul (UFMS)** Campo Grande, Brazil
B.Sc. in Electrical Engineering

Academic Output

Publications Authored or co-authored a total of 14 journal papers and 8 conference papers in the fields of Control Systems and Electrical Engineering.

Supervision Supervised 11 master thesis and semester project students.

Professional Experience

2022–2022 **Visium AG** Lausanne, Switzerland
AI Technology Consultant (Internship)
Ideated and scoped AI-related projects (e.g. inventory optimization, bank fraud detection and state of health prediction), and supported Visium's COO and CTO in several initiatives. Assessed the clients' needs and translated them into data and algorithmic requirements for Visium's engineers.

2020–2020 **Mitsubishi Electric** Cambridge, USA
Machine Learning Consultant (Internship)
Developed scalable and efficient Bayesian optimization algorithms to calibrate digital-twin models of air-conditioning systems. The approach yielded an 80% speed-up in the calibration process and a patent was filed as a result of the internship.

2015–2017 **Silis Tecnologia Ltda** Campo Grande, Brazil
R&D Engineer
Designed an electronic PV-based water-pumping system (hardware and firmware). Conducted field tests of various versions of the system. The product was finally deployed in Guinea-Bissau and Jamaica to aid isolated communities of farmers by providing them with clean water.

Core Skills

Languages English – Full professional proficiency French – Intermediate proficiency
Italian – Full professional proficiency Portuguese – Full professional proficiency

Technical **Computer programming:** Python and MATLAB (intermediate), C and JavaScript (basic). **Data science:** Solid skills in data analysis and modeling: auto-regressive models, Bayesian regression, decision trees, random forests, etc. **Database technologies:** SQL and noSQL databases (for time-series). **Cloud technologies:** AWS and Heroku for application hosting.