Conservative decision-making and inference in uncertain dynamical systems

A thesis submitted for the degree *Doctor of Philosophy*

Jan-Peter Calliess

St Edmund Hall

Supervisors: Prof. Stephen J. Roberts

Prof. Michael A. Osborne

Machine Learning Research Group

Department of Engineering Science

University of Oxford

Michaelmas 2014

Conservative decision-making and inference in uncertain dynamical systems

Abstract

The demand for automated decision making, learning and inference in uncertain, risk sensitive and dynamically changing situations presents a challenge: to design computational approaches that promise to be widely deployable and flexible to adapt on the one hand, while offering reliable guarantees on safety on the other. The tension between these desiderata has created a gap that, in spite of intensive research and contributions made from a wide range of communities, remains to be filled. This represents an intriguing challenge that provided motivation for much of the work presented in this thesis.

With these desiderata in mind, this thesis makes a number of contributions towards the development of algorithms for automated decision-making and inference under uncertainty. To facilitate inference over unobserved effects of actions, we develop machine learning approaches that are suitable for the construction of models over dynamical laws that provide uncertainty bounds around their predictions. As an example application for conservative decision-making, we apply our learning and inference methods to control in uncertain dynamical systems. Owing to the uncertainty bounds, we can derive performance guarantees of the resulting learning-based controllers. Furthermore, our simulations demonstrate that the resulting decision-making algorithms are effective in learning and controlling under uncertain dynamics and can outperform alternative methods.

Another set of contributions is made in multi-agent decision-making which we cast in the general framework of optimisation with interaction constraints. The constraints necessitate coordination, for which we develop several methods. As a particularly challenging application domain, our exposition focusses on collision avoidance. Here we consider coordination both in discrete-time and continuous-time dynamical systems. In the continuous-time case, inference is required to ensure that decisions are made that avoid collisions with adjustably high certainty even when computation is inevitably finite. In both discrete-time and finite-time settings, we introduce conservative decision-making. That is, even with finite computation, a coordination outcome is guaranteed to satisfy collision-avoidance constraints with adjustably high confidence relative to the current uncertain model. Our methods are illustrated in simulations in the context of collision avoidance in graphs, multi-commodity flow problems, distributed stochastic model-predictive control, as well as in collision-prediction and avoidance in stochastic differential systems.

Finally, we provide an example of how to combine some of our different methods into a multi-agent predictive controller that coordinates learning agents with uncertain beliefs over their dynamics. Utilising the guarantees established for our learning algorithms, the resulting mechanism can provide collision avoidance guarantees relative to the a posteriori epistemic beliefs over the agents' dynamics.

Acknowledgements

A great many people have supported me, may it be directly or indirectly, in my efforts of pursuing the work that is described in this document.

Firstly, I would like to thank my supervisors Professors Stephen Roberts and Michael Osborne for taking me on and granting me the academic freedom and the necessary means to explore. I shall always most gratefully and warmly remember them as a constant source of positivity, enthusiasm, support and encouragement throughout my time at Oxford.

I am very grateful for funds via the the EPSRC EP/I011587 "Orchid" project without which my doctoral studies would not have been possible.

Of course, I would like to thank all the members of the Machine Learning group (MLRG) for all the conversions (and banter) as well as for making MLRG such a pleasant environment to be in, which I will certainly miss. Special thanks go to Steve Reece, Mark Ebden, Chris Lloyd, and Nathan Korda for allowing me to consult them on my work and I hope our conversations will lead to collaboration in the future. Especially Nathan's kindness, actively engaged feedback and help with proofreading at the final stages of this work is most appreciated.

I would also like to thank all the collaborators and researchers who took the time to give useful hints that certainly have influenced my direction. I cannot name them all, but I am especially grateful for receiving suggestions and useful feedback from Antonis Papachristodoulou, Carl Rasmussen, Marc Deisenroth, Uwe Hanebeck and Andreas Krause. Many thanks also go to my collaborators, especially Daniel Lyons, Antonis Papachristodoulou, Uwe Hanebeck, Sarvapali Ramchurn and Feng Wu for their valuable time and effort to help creating new research. In addition, I am grateful to my examiners Stephen Duncan and Amos Storkey for their scrutiny, good suggestions and for helping to make the viva an experience that will always stay with me as an enjoyable memory.

Finally, I am most grateful for all the love, joy, consolation and strength I was allowed to draw from friends and family. Especially to my mother who gave me what it took to get here and to Gianna, whose company, moral support and affection has been a wonderful gift.

To everyone who has supported me, may it be directly or indirectly, let me simply say:
Thank you!

A note on joint publications and contributions of collaborators

Parts of this work contains improved versions of material that has been published and presented at conferences or workshops [48, 50–54, 122, 158].

I should like to direct attention to contributions from colleagues and collaborators that went into this thesis and are not solely my own work:

- Parts of Chapter 6 were published as [48] in Springers Lecture Notes series. For copyright reasons we point out that "The original publication of the article is available at www.springerlink.com." From the same article, parts went into the chapter that are not my own work. The inital impulse of trying to develop multi-agent methods for particle-based collision avoidance came from Uwe Hanebeck (although the solution is my own). The experiments of the particle-based methods in Fig. 6.7, the idea for the baseline method of centralized coordination by introducing constraints for all pairs of particles Daniel Lyons. While the idea of utilising tail inequalities was mine he came up with the idea of employing Whittle's inequality. The derivations around Eq. 6.19 are a result of repeated interactions and are hard to attribute to one person alone. All other aspects are my own contribution.

- The idea of the centralised version of the "particle-free" approach, discussed in the same section, was initiated by Daniel Lyons and resulted in a joint paper that was published in the proceedings of IEEE's ACC [158]. Part of the code that generated the plots in Sec. 6.3.2 was an extension of the code emerging from this work and was generously provided by Daniel.

- For reasons of copyright, we mention that large parts of Ch. 5 have appeared in the proceedings of AAMAS [52] and that an early version of this work was presented at an ICML [50] workshop. We should also like to point out that the idea of employing Whittle's inequality as one of the options in the construction of a criterion function certainly emerged under the impression of the experience gained during the previously discussed collaboration.

- I would like to thank Feng Wu for having provided an implementation of the Max-Sum algorithm for collision avoidance. This was used in the thesis to help producing part of the baseline comparisons in the graph planning tests of Ch. 6. Finally, I am grateful for discussions and code from Girish Chowdhari and Hassan Kinkgravi that allowed me to deliver a fair comparison of my control method in the wingrock control example of Sec. 4.4.2.

Contents

1. Introduction

> *"... if every instrument could accomplish its own work, obeying or anticipating the will of others...if the shuttle weaved and the pick touched the lyre without a hand to guide them, chief workmen would not need servants, nor masters slaves."*
>
> Aristotle (384-322 BCE)

We live in an increasingly fast-paced, interconnected world that requires rapid, sound decision making. The ubiquity of information and communication on the one hand, the demands of ageing populations and the requirement for efficiency under competitive pressures and resource scarcity on the other, provide both challenges and opportunity. In response, automated decision making has started to assume a pivotal role in a large number of industries and to affect our daily lives. Just to name a few examples, computers already direct supply chains, make decisions where to invest our pension funds, control robots in hazardous environments, regulate power supply in electricity networks, fly aircraft, drive trucks, assist surgeries, decide on credit card applications, prevent frauds, deliver web-search results and have begun to deliver packages via unmanned aerial vehicles. Expected to become ever more pervasive, systems of artificial decision makers will have great societal impact. Therefore, developing a better understanding of their properties and purposeful design will have to be one of the key questions of our time.

The demand for automated decision making, learning and inference in uncertain, risk sensitive and dynamically changing situations presents a *challenge: to design computational approaches that promise to be widely deployable and flexible to adapt on the one hand, while offering reliable guarantees on safety on the other*. The tension between these desiderata has created a gap that, in spite of intensive research and contributions made from a wide range of communities, remains to be filled. This represents an intriguing challenge that provided motivation for much of the work presented in this thesis.

Conservative inference for safe decision making under uncertainty

Owing to limited knowledge and processing capabilities, uncertainty necessarily permeates any feasible model attempting to capture processes and interactions in the real-world.

Therefore, when designing artificial decision-makers that act and inter-act in a real-world scenario, *inference* is required to fill the gaps of incomplete knowledge or insufficient computation.

Inference is an elusive term with many competing and inconsistent definitions. For a comparative discussion of various definitions and categorizations, the reader is referred to [98]. In our context, we refer to inference as a computational process for reaching conclusions (or making *predictions*) on the basis of knowledge. Knowledge can be given a priori and also fold in empirical evidence (data). Inference based on the latter, often referred to as *inductive* or *non-deductive inference*, is closely related to machine learning. We understand the process of automated *learning* to be the construction of an algorithmic *inference* or *prediction rule* from data. Frequently, this rule involves a model of an underlying data generating process. For instance, in the dynamical systems learning context we are most interested in, machine learning algorithms are often employed to construct a computational model of a dynamical law from observed system behaviour [77, 180]. In such a situation, the learned model can then be utilised to predict the effects of previously unobserved conditions.

In this work, we have a particular interest in inference rules that have a capability of delivering a quantification of the level of uncertainty around their predictions. The type and interpretation of this quantification will have to be based on a priori assumptions about the ground-truth [171]. As one of the most popular methods, probabilistic inference [123] places a probability measure over the space of inference targets which quantifies the uncertainty over events. Here, the uncertainty over predictions is quantified by a (often subjective) probability. Apart from the Bayesian approach, we will also consider set-based inference. Here, a priori knowledge is combined with observations to deduce the membership of a prediction to a (bounded) set. The uncertainty quantification could be the entire set or, in the case of symmetric intervals around a scalar prediction, the radius of the interval.

In relation to the quality of the predictions and the attached uncertainty quantifications, bounded computation can be an issue. For instance, while argued to be one of the most principled methods for performing inference under uncertainty [123], probabilistic inference can suffer from intractability problems. That is, while it might be clear how one should perform inference in accordance to the laws of probability, exact inference cannot be performed within the bounds of a finite computational budget. This means that in practice, the theoretically available inference rule has to be replaced by an approximation. It seems to be a largely unexplored question how the approximation impacts the prediction and the uncertainty. Unfortunately, this issue has plagued much of the state-of-the-art in model-learning and control with Gaussian processes (e.g. [79, 80, 173, 179, 180]). Owing to the great *flexibility* to learn rich model classes, this Bayesian nonparametric learning method has become very popular in recent years. However, in many of these methods the control actions rely on approximations for multi-step lookahead forecasts [80, 106] that provide no bounds for the approximated state predictions. Consequently, there are no finite-time stability bounds or tubes on the error dynamics for these approaches that would allow a quantification of the risk attached to the

control actions [180].

In the interest of safe decision making, this is of course an unfortunate state of affairs since decision making frequently involves risk. For instance, in aforementioned investment situation, one might like to devise artificial decision makers that allow for the control of the risk of bankruptcy. Or, in autonomous vehicle control, one would like to control the risk of collisions.

In such situations, we would be willing to favour *conservative decision makers* that choose actions that err on the side of caution in the face of uncertainty. And, since decision making under uncertainty is inevitably based on inference, the key aspect of facilitating safe, conservative decision making is to design *computationally tractable inference rules* that are *conservative* and never underestimate the true uncertainty associated with their predictions.

About this work

As suggested above, automated decision-makers that act and interact in a dynamically evolving, uncertain environment need to base their decisions on computationally tractable, adaptive models that facilitate inference over the consequences of actions and quantify the uncertainties around these predictions. Especially when rapid decision making in potentially risky situations is required, it is important to have models that are guaranteed to report the uncertainty or risk conservatively, even when operating under the constraints of a finite computational budget.

Motivated by these insights, we set out to devise methods for artificial decision making and inference under uncertainty. Owing to this widely applicable scope, we will develop methods that are suitable to address problems that cross the boundaries between a number of different fields, including multi-agent systems, control, machine learning and artificial intelligence in general.

In order to gain computational inference rules over unobserved effects of actions, we develop machine learning approaches for the construction of models over dynamical laws that facilitate conservative inference. As an example application for conservative decision making, we apply our learning and inference methods to control in uncertain dynamical systems. Due to the uncertainty bounds, we provide performance guarantees for some of the resulting learning-based controllers. Furthermore, our simulations demonstrate that the resulting decision making algorithms are effective in learning and controlling under uncertain dynamics and can outperform alternative methods.

In systems where multiple decision makers are involved, coordination becomes necessary. Connecting to our control applications, we develop a method for continuous-time collision prediction that can base its inferences conservatively on differential models of the uncertain plants of the agents involved. We then show how to utilise this collision prediction module for collision avoidance by combining it with coordination

methods in concert with an approach to incremental search over infinite-dimensional policy space. Finally, we consider discrete-time multi-agent decision tasks where the agent problems are coupled by non-convex interaction constraints. Stating the problem in the general framework of distributed optimisation, we propose three approaches. Being a problem that is particularly hard, we once again focus our exposition on collision avoidance as a test bed scenario and give simulations in the context of multi-commodity flow problems and show derive new methods for the application to distributed stochastic model-predictive control with collision avoidance. In the latter context, we contribute proposals for how to impose probabilistic interaction constraints that are conservative and promote safe and rapid distributed decision making of the multi-agent collective.

Finally, we provide an example of how to combine the different methods developed in the thesis into a multi-agent predictive controller that coordinates learning agents with uncertain beliefs over their dynamics. Utilising the guarantees established for our learning algorithms, the resulting mechanism can provide collision avoidance guarantees relative to the a posteriori epistemic beliefs over the agents' dynamics. It also features many of the properties we desire our inference and decision making processes to exhibit: rapid and safe decision making under uncertainty based on flexible, adaptive inference that takes uncertainty into account in a conservative manner.

Structure and contributions

Chapter 2 serves multiple purposes. Owing to its interdisciplinary nature, this thesis touches upon a great many different topics that are normally viewed, often in isolation, under the lenses of specialised fields. It is therefore necessary to provide an introduction to some of the relevant topics, discuss related work, provide background information and lay some preliminary mathematical groundwork that can be referred to throughout subsequent parts of the thesis. These are the things Chapter 2 was written to contain.

In **Chapter 3**, we first introduce a new framework, *SIIC*, for simultaneous system-identification and inversion control. Taking advantage of a priori assumed model structure of the underlying system dynamics it allows for a more fine-grained identification result than black-box models. Employing random fields for the identification of the constituent vector fields ensures a great deal of flexibility to learn rich classes of dynamics online. For control, we consider an application of (partial) feedback-linearisation [233, 248] which simplifies the control problem. On this basis and owing to the Bayesian interpretation of the learning algorithms it becomes possible to construct control laws that offer epistemic guarantees on global asymptotic stability of the expected closed-loop dynamics. We also use the derivation of this result as an opportunity to discuss the difficulties with attempting to extend such guarantees to probabilistic convergence guarantees. To illustrate some of the more practical merits and short-comings of the Bayesian nonparametric approach, we provide

simulations of simple mechanical systems.

In **Chapter 4** we develop a nonparametric inference approach and apply it to the learning and control of dynamical systems. We can incorporate a priori knowledge of Hölder continuity (with respect to arbitrary (pseudo-) metrics, boundedness as well as input and observational uncertainties into our inference rule. This information is also converted into bounded-set uncertainty estimates around the inferred predictions. We furthermore provide guarantees of *conservatism* of the uncertainty around the predictions and provide additional guarantees of optimality and convergence in the limit of data sets that converge uniformly to dense sets of the input space. To improve the applicability, we propose incremental update methods for estimating Hölder constants and bounds and present a modification of the standard inference rule that has a smooth prediction signal. Harnessed with this new toolbox, we tend to applications in model learning and control. In combination with model-reference adaptive control [63], we show how the uncertainty bounds can be utilised for state space exploration and to establish guarantees of (bounded) stability. In a wingrock control problem, we show that our new controller outperforms other recent methods [63, 65, 66]. We also combine our method with the SIIC framework proposed in the previous chapter. Moreover, to give an example of a controller based on black-box learning, we illustrate the viability of our inference approach in combination with inverse dynamics model learning and control [180].

The remaining two chapters are chiefly concerned with multi-agent decision making under uncertainty.

In **Chapter 5**, we consider the problem of multi-agent decision making in continuous-time dynamic systems. Here the agents' problems are coupled by an infinitum of (non-convex) interaction constraints that impose the need for coordination. While we believe most of our methods have the virtue of being applicable in general problem settings, we keep the exposition concrete and focus on multi-agent collision avoidance under stochastically uncertain plant dynamics. Here collisions are violations of probabilistic interaction constraints for each point in time. As a first contribution, we propose a novel, conservative approach for continuous-time collision prediction. The prediction method presented therein performs conservative inference to predict the presence or absence of a collision within a continuous but bounded time interval. Here conservatism means that the method is guaranteed not to miss any collisions that might occur with a given probability at any point in time. Inference is required since it is impossible to rule out collisions of a continuum of constraints when computation is finite. The new key idea is to convert the constraints into a function whose sign provides information about constraint violations. We show how the functions can inherit regularity properties from the involved agents' plant dynamics. This regularity allows the prediction algorithm to infer the presence or absence of a collision based on a finite number of function evaluations and can guide the search in manner closely related to ideas found in well-established optimisation methods [124, 230]. Furthermore, if we construct these functions on the basis of tail inequalities, this can increase conservatism but results in

a fast approach that is applicable to any stochastic trajectories whose first two moments can be computed (or at least be bounded). Building on the collision prediction model we move on to propose an iterative, heuristic collision resolution method that can be interpreted as an incremental policy search method over an infinite-dimensional parameter space.

Both the collision-avoidance and resolution approaches are combined with coordination methods. Next to considering a basic fixed-priority scheme commonly found in robotics (e.g. [26, 27]), we also propose a coordination method that has an auction interpretation. Designed for cooperative systems of artificial agents rather than self-interested settings [75], the bid computation is prescribed and serves as a distributed heuristic intended to facilitate collision avoidance and greedily reduce the social cost of the expected integral curves.

In comparison to the collision avoidance methods developed for finite-time systems, the continuous-time approach presented in its current form can neither provide guarantees on optimality nor on termination. However, if it does terminate, the coordinated trajectories are guaranteed to satisfy all collision constraints.

We provide simulations of agents with feedback controlled plant dynamics that can be represented as Ito stochastic differential equations [101] illustrating the viability of our approach.

Apart from our considerations in stochastic settings, we provide derivations that enable our methods to be employed in settings with robust tube uncertainties. This renders the approach applicable to the coordination of agents whose plants are controlled by learning-based controllers with bounded error dynamics. In particular, this allows an application to agents whose uncertainty stems from learning methods such as the ones we present in Ch. 4.

In **Chapter 6**, we consider multi-agent conflict avoidance in discrete-time systems. We pose coordination as a multi-agent optimisation problem and present several contributions to its solution. Two of the methods are provably optimal but not fully distributed. In lieu to the approach in the previous chapter we furthermore devise distributed methods that have an auction interpretation and where bids are computed to facilitate local gradient descent on the social cost function. While generally not optimal, this approach improves run-time significantly against the optimal methods we tested against and was able to produce the optimal solutions at a large number of problem instances.

As test-beds, we again focus our exposition on collision-avoidance problems. These have the advantage of being particularly challenging due to their inherent non-convexity and proven hardness. At first, we test our methods in the context of collision avoidance in graphs which can be stated as mixed-integer programming problems. Being easy to visualise and explain, the proven *NP*-completeness of collision avoidance in graphs also shows that they are essentially (i.e. up to a poly-time reduction cost) equivalent to any other finite multi-agent problem that belongs to the *NP* class. Therefore, the methods might also be of interest to a reader who is not necessarily interested in these graph problems. Linking our methods to the control parts of the

thesis, we proceed by formulating mixed-integer representations of multi-agent stochastic model-predictive control problems connected by probabilistic collision-avoidance constraints and explain how our coordination methods can be applied in this context. This introduces new tools to an application domain where existing work on multi-agent control has been very limited. To render our approaches as widely applicable as possible, we connect our multi-agent collision-avoidance methods to previous single-agent particle-based methods [39]. However, apart from the issue of being computationally burdensome, the approach is exposed to the risk of yielding non-conservative decision making. That is, owing to the fact that collision probabilities are estimated with sampling, the collision risk may well be underestimated. This may yield decisions on control actions that violate the imposed probabilistic collision avoidance constraints and hence are not safe. In order to address this issue, we make an additional contribution and derive avoidance constraints on the basis of tail inequalities. This significantly reduces computational effort of the joint optimisation problem and ensures conservatism. Since one can always employ tail inequalities that are distribution-independent, this approach retains much of the wide-ranging applicability. The viability of our methods are demonstrated in simulations of multi-robot collision avoidance, both in an open-loop and in a receding horizon control setup.

The chapter concludes by bringing together multiple techniques proposed in the thesis. In particular, we discuss how to combine the conservative optimisation-based predictive control approach with the conservative uncertainty bounds of our kinky inference rule of Ch. 4. This allows for safe and conservative coordination of learning agents. As predicted by our theoretical results, the simulations exhibit how the degree of conservatism is reduced with increasing learning experience. We believe that the merger of bounded learning on the one hand and guaranteed multi-agent collision avoidance on the other, renders the resulting approach the first of its kind.

Chapter 7 contains concluding remarks and **Chapter 8** outlines a selection of research questions to be considered in the context of future work.

2. Preliminaries, Background and Related Work

"The good life is one inspired by love and guided by knowledge."

Bertrand Russell

Owing to the interdisciplinary nature of the contributions of this thesis, we touch upon a great many different research areas including multi-agent systems, control and machine learning. Furthermore, in preparation of our results to come, we need to draw on (and develop) a number of theoretical results and concepts which we use throughout subsequent parts of the thesis.

To this end, this chapter serves as a conduit into the main body of the thesis, provides background, discusses related work and collects preliminary facts referred to in later chapters. However, a reader not interested in the details of certain sections is encouraged to skip ahead to the main body of the thesis and refer back to the relevant parts of this chapter if the need arises.

2.1. Uncertain dynamical systems and nonparametric machine learning for model learning and control

This work is about decision-making and inductive inference in dynamical systems that are uncertain.

Therefore, it touches upon a plethora of disciplines and the relevant literature is vast and spread across diverse communities such as system identification [35, 156], adaptive control (although adaptive control typically refers to the tuning of parametric control models [14]) as well as model learning in the machine learning and robotics communities (e.g. [77, 180]). For recent survey papers and books providing overviews and introductions from the perspectives of different communities, the interested reader is referred to [35, 77, 157, 180].

In the context of our work, inductive inference is to mean that, on the basis of current observations, we predict the state evolution of the system at future times. To this end, typically a mathematical model is constructed that can be used for computing the predictions (inferences). The art and science of model construction is at the heart of machine learning [171]. These days, *supervised* machine learning essentially is function estimation. That is, given a (noisy) sample of a target function, artificial learning is the process of constructing a computable model of a function that generalises beyond the observed sample. While parametric machine learning subsumes all the information contained in the sample in a parameter of fixed size,

nonparametric machine learning typically takes the entire sample into account when making inference. That is, it creates models for non-deductive inference whose complexity grows with the sample size. This conceptual difference typically leads to complementary advantages and disadvantages [171]: Parametric methods typically can take longer to aggregate all information into the finite parameter. However, once a model is learned the computational complexity for performing inference is typically determined by the size of the parameter, not the size of the data. This means that the inference can be very fast. For instance, in standard artificial neural networks, the parameter is given by the weights of the network links [36, 171]. Once the weights are determined from the training data, the required computation only depends on the number of weights, not on the size of the training data. As a rule of thumb, parametric methods tend to take longer for training but can be fast prediction-makers when the available data is large. However, the greatest downside of parametric methods is their strong inductive bias due to being restricted to a finite-dimensional hypothesis space [171].

In the control applications of Ch. 4, in keeping with results reported in [63, 66], we will also be able to observe the effect of this downside when the state of the system leaves a presumed area and the employed neural network-based controller is incapable of adjusting.

While the prediction time typically grows with the size of the data set, nonparametric approaches have the advantage of being flexible due to making few model assumptions and hence, tend to have higher capabilities of adapting to the unforeseen [171, 261]. This might explain, why they seemingly have become increasingly popular for dynamical system learning and control in recent years [40, 77, 180].

Having conservative, safe decision-making under uncertainty in mind, we are interested in flexible inference and learning methods that can quantify the uncertainty around their predictions. Few machine learning methods seem to offer this capability. An exception are Bayesian learning methods (e.g. [36, 123, 171]). However, most of these methods are parametric and suffer from the restrictions on the inductive bias imposed by the choice of a finite-dimensional function space [36, 171].

By contrast, learning methods based on *Gaussian processes* [201] are examples of nonparametric machine learning algorithms (cf. Sec. 2.6) that have the potential to learn functions in an infinite-dimensional Hilbert space. The resulting flexibility to learn rich function classes in a black-box type fashion has propelled them into the centre of much research attention over the past decade.

As is the case with most nonparametric learning methods, the computational complexity for training and prediction of Gaussian processes (GPs) grows with number of training examples. In fact, the computational complexity for training is cubic and the computational complexity for inference (prediction) is quadratic in the number of data points [201] (although the former can be reduced to quadratic either approximately [104] or exactly for specific cases [243] and the latter effort can be reduced to linear growth by sparse approximations

of GPs [225] or recent SDE conversion techniques [203, 215]). As in most non-parametric methods, the increased computational complexity is rewarded with a reduced inductive bias. That is, when choosing the right covariance function, Gaussian processes can learn any smooth function. Furthermore, to the best of our knowledge, *Gaussian process regression* (cf. Sec. 2.6, [201]) is the only approach in Bayesian nonparametric inference that combines the capability to learn rich function classes with analytic tractability.

If a method is employed for learning the transition function of a dynamical system, the question arises how to utilise the trained model to compute beliefs over the resulting state trajectory. Unfortunately, to date it does not seem to be well-understood how to fold in the uncertainty quantifications of GPs into the predictions of state-dependent system dynamics in a conservative (that is uncertainty-preserving) manner. This is important if we desire to provide bounds on the performance of a control law that is based on an uncertain dynamics model identified by a GP.

In Ch. 4, we take an alternative route. Instead of placing a prior measure over some probability space of function values, we leverage a priori knowledge about Hölder continuity and boundedness of the ground-truth function. This yields an inference rule, we refer to as *kinky inference*, that can be seen as a nonparametric machine learning algorithm.

Before going into further detail on learning methods, we will briefly go over different types of uncertain dynamic systems.

The most common types of mathematical descriptions of dynamical systems that are general enough to capture non-linear behaviour are state-space models in differential or difference form. A differential or difference equation (both will be abbreviated by the acronym *DE*), is a mathematical model to describe a dynamical system. Assuming it is at least approximately computable, the model can be utilised to predict a state by solving it, i.e. by computing a solution trajectory (integral curve) $X = (X_t)_{t \in \mathcal{I}_t}$. We assume the DE in question can be expressed as follows:

$$\mathrm{d}X_t = \psi(X_t, t, u; \phi). \tag{2.1}$$

Here, ψ is some deterministic function defining the structure of the DE, $t \in \mathcal{I}_t \subset \mathbb{R}$ is a *time* index assuming values in a *time index set* $\mathcal{I}_t$, $\mathrm{d}X_t$ is the *increment* at time t, u is a (control) *input* and ϕ a *parameter*. Being an index set, $\mathcal{I}_t$ is totally ordered and we write $t + dt$ for the time index following t. The DE gives a prescription on the successor state $X_{t+\,dt} = X_t + \mathrm{d}X_t$ where the state increment $\mathrm{d}X_t$ is given by Eq. 2.1. Of course, if the time index set is continuous then time and state increments dt and $\mathrm{d}X_t$ are infinitesimal.

In discrete time, the increment becomes $\mathrm{d}X_t = X_{t+\,dt} - X_t$. Since the index set is bijective to a subset of integers, it is more common to use integers as indices in discrete-time and, to write the equation in difference form, $X_{k+1} - X_k = \psi(X_k, k, u; \phi)$. Here $k \in \mathbb{N}$ denotes the time step index which corresponds to discretised

time $t_k = k\Delta$ given by time increment $\Delta = dt > 0$. Alternatively, a discrete-time dynamical system can also be written in *state-transition form*:

$$X_{k+1} = \Psi(X_k, k, u_k; \phi). \tag{2.2}$$

where $\Psi(X_k, k, u_k; \phi) = X_k + \psi(X_k, k, u_k; \phi)$.

In the continuous-time case, if ϕ is known and deterministic, one is dealing with an ordinary differential equation (ODE) which we shall discuss in Ch. 2.4.1. In discrete-time the terms ordinary difference equation, recurrence relation or auto-regressive model are most commonly found.

We assume the parameter ϕ is the source of uncertainty or randomness. Depending on the dimensionality of ϕ and the properties of the probability measure describing the uncertainty, different DE categories arise with sometimes contradicting definitions throughout the literature. We will adhere to the following nomenclature:

If ϕ is an uncertain (i.e. random) finite-dimensional vector, one commonly speaks of a *random differential equation (RDE)* [232]. Most widely studied are *stochastic differential equations (SDE)*[1] with orthogonal increments. Here, the parameter ϕ is infinite-dimensional and random (i.e. a random function) in such a fashion that the increments $\mathrm{d}X$ are statistically independent. In that case, one also speaks of an *orthogonal increments* (o.i.) process. In the most common case of SDEs, the equation has the form $dX = f(X_t) + g(X_t, u_t)\,\mathrm{d}W_t$ where $\mathrm{d}W_t$ is a Wiener increment at time t. In engineering it is not uncommon to employ the formal notation $dX = (f(X_t) + g(X_t, u_t))\phi_t\,dt$ where ϕ_t is the parameter coinciding with the "white noise" process. Here the uncertainty is injected via the white noise, which is typically representing a *disturbance*. That is, an external, time-dependent stochastic force that is not modelled as being subject to the internal state of the object described by the dynamical law.

The orthogonality assumption of SDEs is key in their analysis and yields their solutions via stochastic integrals [101, 110]. If the randomness is due to an o.i. process the stochastic integrals can be defined as mean-square limits of Cauchy sequences of random step functions (e.g. cf. [110], Ch. 9). Furthermore, the orthogonality assumption is the basis of Ito calculus, results in Markov properties of the stochastic integral curves [101], can make the analysis amenable to martingale arguments and allows for computationally efficient sampling. Of course, the latter is important for simulation and when closed-form solutions are not available, and crucial for the tractability of sampling-based estimation and planning. Furthermore, the white noise increments can be a suitable model for modelling uncertain influences such as sensor noise or turbulences whose exact signal can only be learned with difficulty (if at all).

These facts, in conjunction with the convenient analytic properties, might be the chief reason why SDEs are the dominant model for uncertain dynamical systems, may they occur in applications pertaining to physics,

[1] It should be noted that not all authors distinguish between random and stochastic differential equations.

control, quantitative finance, AI or robotics. Consequently, the identification of parametric SDE models is a classic topic that has continued to attract much attention to date (cf. e.g. [33], [118], [160], [133], Sec. 6.4).

As another thread of related work, in recent years, the machine learning community has looked at so-called latent-force models (LFMs) that replace the Wiener increments by a smooth time-indexed Gaussian process [5, 92, 204]. For certain covariances, closed-form solutions of the first two moments of the stochastic integral curves can be calculated, assuming that the DE is linear. That is, in our notation, for linear ψ, and $\phi \sim \mathcal{GP}(m, k)$ being a GP indexed by time, for certain covariances k, closed form solutions of the first two moments of Eq. 2.1 (as functions of the indices of the latent forces) have been derived via frequency transform approaches [153]. The model can therefore be utilised to predict in DE where the GP has been used to learn some smooth temporal process (the "latent force") driving an ODE and these *latent force models* have found a range of applications such as in the prediction of protein concentrations [153] , concentrations of pollutants in the Swiss Jura [5] to applications in prediction of queues and heating [204].

Unfortunately, the assumption that all uncertainty stemmed from certain Gaussian processes whose indices do not depend on the uncertain state, imposes a severe restriction on the expressiveness of LFMs. In particular, in this thesis, we are interested in modelling posterior beliefs over dynamical systems that are based on Bayesian nonparametric model learning methods aiming at identifying physical *laws*. When referring to a law, we mean a functional relationship between states of an uncertain dynamical system. Therefore, if learning of a physical law is conducted in a Bayesian nonparametric fashion, then it results in a belief over the law which is a random field f whose indices (inputs) will depend on the state of the dynamical system. When a controller is introduced to the dynamical system that bases its action on the uncertain belief over the uncertain physical law, the result is a predictive model with a driving "latent force" f whose index (i.e. input) at any given point in time in the future, depends on the uncertain history of states up to that point in time. In our notation, this means we have an infinite-dimensional random field parameter $\phi(X, t) = f(X, t)$ resulting in the *uncertain DE (UDE)*:

$$\mathrm{d}X_t = \psi(X_t, t, u; f(X_t, t)). \tag{2.3}$$

Since Bayesian nonparametric methods have been demonstrated to be highly successful in black-box system-identification [79, 173, 180] predicting with UDEs of the type given as in Eq. 2.3 is highly desirable. Unfortunately, it is evident that solving this UDE is difficult. Firstly, its solution process $X = \left(X_t\right)_{t \in \mathcal{I}_t}$ generally does not have to be Markovian because each increment can, depending on the nature of random field (r.f.) f, depend on the entire history of increments in the past. Also, two increments $dX_t = X_{t+\,dt} - X_t$, $dX_\tau = X_{\tau+\,dt} - X_\tau$ generally are *not* independent and so, the uncertain integral curve X generally is not an o.i. process.

Owing to these properties, the UDE model of Eq. 2.3 cannot be solved with classical approaches. To make

matters worse, the dependence of each increment on, potentially, the entire past means that simulating the UDE can involve rapidly growing computation since after each time step computing a new posterior is necessary from which the next sample has to be obtained. In the case of Gaussian processes, the computational effort for accurately computing a posterior grows cubically with the number of data points [201]. Furthermore, numerical problems often arise when one attempts to condition highly correlated data points which could render the accuracy of the simulation result questionable in many situations.

In spite of these issues, Ko et. al. [135] have succeeded in using sample trajectories from a GP-trained black-box forward model to train a policy in a reinforcement learning setup. However, the majority of GP-based control work that does inference with such forward models (e.g. [77–79, 106, 136, 173, 173, 196, 220, 239]) bases decision-making on approximation approaches based on a method proposed by Girard, Rasmussen, Candela and Murray-Smith [106]. The method, which is designed for multi-step look-ahead prediction in discrete-time, iteratively matches moments from time step to time step in a manner that resembles an extended Kalman filter. In order to take into account the uncertainty of the mean predictions that get fed back into the inputs when predicting, the authors use the approach of uncertain test inputs [106] to propagate the uncertainty forward through the multiple prediction steps.

Unfortunately, although exact for one-step lookahead predictions, the approximation ceases to provide any guarantees on the error of the approximation beyond the first prediction step. In particular, it cannot be ruled out that the variances might be underestimated in the process. Therefore, the resulting inference methods based on these approximations cannot be considered to be conservative. Since all GP-based control works that utilise an uncertain DE to predict the effect of control inputs are based on this approximation method, none of these works can provide any probabilistic guarantees for their controllers.

So far, we have considered forward models that give a prescription of how to arrive at a new state given the current state, time and control. An additional type of approach to learning-based control in UDEs is *inverse (dynamics) model learning* (IML). In continuous time, the inverse dynamics model is the mapping $\psi^- : \mathcal{X} \times \mathcal{X} \to \mathcal{U}, (x, \dot{x}_{\text{ref}}) \mapsto u$ where $\dot{x}_{\text{ref}}$ is a desired reference state derivative or increment. Using observations of states and state increments (or derivatives) a variety of nonparametric machine learning approaches has been applied to learning this mapping, including Gaussian processes [60, 182] and localised linear regression techniques [221, 257]. Analogous approaches have considered the application of nonparametric learning methods to learning inverse kinematics models [86] or inverse operational / task space models [197]. For instance, in robotics, task space learning might use a machine learning mapping to directly learn which controls to apply in order to move a tool centre point along a desired reference trajectory [183, 197].

While having the disadvantage of having to learn over a space of markedly higher dimensionality, the advantage is that, being a black-box model, the IML approach has a great degree of flexibility. Also, because

it does not linearise the dynamics, it might be able to benefit from the intrinsic properties of the dynamics rather than cancelling them via the control as is done in inversion control.

Surveying both inverse and forward model learning for control, the authors of [180] observed that dealing with the uncertainty of the learned dynamics models had not been investigated extensively (with the exception of a contribution by [176]), despite its central role in robotics and control and that approximation based control often still suffered from a lack of proper analysis of stability and convergence ([180], Sec. 4).

Part of this thesis will develop contributions towards closing this gap in the framework of robust control and collision avoidance. To be able to do so, we develop a nonparametric learning and inference method (cf. Ch. 4) for this purpose that provides interval-bounded guarantees on the errors of its predictions. When using it in closed-loop control, this yields an alternative type of UDE, where the values of f (or ψ) are contained within interval bounds.

In this case, the solution trajectories $t \mapsto X_t$ are no longer random fields, but are sets of trajectories often referred to as tubes in the (model-predictive) control literature [59, 152]. We use the tube approach as a stability guarantee and show how it connects to the collision avoidance approaches we develop in the thesis.

While we do provide some first results that ensure collision avoidance and stability of learning agents, we believe this work could benefit further from exploring connections to existing works in the robust MPC literature that assumes set-bounded disturbance models [56, 149, 166, 167, 251]. We would hope that this would yield improved robustness and stability guarantees.

2.2. Model-Predictive Control

Model predictive control (MPC) is an increasingly popular approach for controlling dynamical systems under constraints. In model predictive control the goal is to find an optimal control sequence that is feasible with respect to a given dynamic model. MPC is often synonymously used with *receding horizon control (RHC)*. In (discrete-time) RHC a sequence of control outputs is generated iteratively by optimizing the control actions $u_0, ..., u_{T-1}$ through the predictive model up to horizon T. Then, the first control action is executed, the resulting new state observed and the process starts over. This yields a succession of MPC optimizations of T control actions (where always only the first control action is executed at a time) and consequently, a temporal horizon receding with the number of iterations. For a general introduction to RHC and MPC in general the reader is referred to [108, 161].

2.2.1. Stochastic MPC

In Ch. 6, we develop coordination methods in the framework of constrained optimisation. The generality of this framework renders them applicable to a large number of planning contexts. Applications that can be

stated very naturally as an optimisation problem are *robust model-predictive control* (RMPC) and *stochastic model-predictive control* (SMPC).

There are different types of robust control. Most methods, such as H_∞ control, assume a bounded norm of the time series of disturbances. In Sec. 6.4, we will adapt our own collision avoidance methods to multi-agent control of learning agents whose predictions over the uncertain dynamics come with interval-bounded uncertainties (rather than energy bounded). In relation to this effort, more relevant work to consider in the control literature would be RMPC approaches where a nominal model is endowed with interval-bounded uncertainty specifying in what (bounded) range (*robust tube*) the actual trajectory can deviate from the model [150, 152, 161, 199, 251].

Considering stochastic uncertainties, in SMPC, the deviations from the nominal trajectory are assumed to be drawn (normally i.i.d.) from some distribution. If the distribution has bounded support, the SMPC controllers tend to produce less conservative results (than RMPC controllers utilising the supports of the distributions as the uncertainty intervals) because the additional distributional knowledge shifts attention from worst-case bounds to probabilistic bounds [59, 62].

Typically, SMPC focusses on discrete-time problems. That is, the time index set $\mathcal{I}_t \subset \mathbb{N}$ is a subset of the natural numbers. The plans are control inputs $(u_t)_{t\in\mathcal{I}_t}$ allowed to assume values in some control space $\mathcal{U}$ and the resulting state-trajectories are discrete-time stochastic processes $(x_t)_{t\in\mathcal{I}_t}$. In this section, we consider finite-time horizon problems where $\mathcal{I}_t = \{0, \dots, T-1\}$ for some time horizon $T \in \mathbb{N}$. In such cases, the notation $u_{0:T-1} = (u_t)_{t\in\mathcal{I}_t}$ is commonly employed.

As the word "model-predictive" indicates, there exists a model describing the relationship between control inputs and uncertain state trajectory that allows prediction (at least of some aspect) of the trajectory resulting from plan execution. Most typically, the model is a stochastic difference equation of the form $x_{t+1} = f(x_t, t, u_t, \nu_t)$ where often f is linear and $(\nu_t)_{t\in\mathcal{I}_t}$ an o.i. process.

Planning is done over the control inputs by optimising a cost function $c(\cdot)$ that typically depends on both the control inputs and the predicted state-trajectory or a statistic thereof. Typically, cost functions penalize some combination of the expected deviation from a target and control energy. In addition, constraints may be given on control inputs and desired properties of the trajectory. For instance, boundedness of the control inputs can be imposed by a constraint of the form $u_t \in [\underline{u}, \overline{u}]$ and one could constrain the trajectory to avoid collisions with static obstacles and other agents.

Since the trajectory is an uncertain prediction, constraints pertaining to the trajectory have to involve a statistic of future states or have to bound probabilities of each point in time or of the entire trajectory. That is, for some $\delta \in (0, 1)$ we impose instantaneous probabilistic constraints of the form $\forall t \in \mathcal{I}_t : \Pr[x_t \notin \mathfrak{F}] \leq \delta$ or, if possible, constraints on the entire trajectory $\Pr[\exists t \in \mathcal{I}_t : x_t \notin \mathfrak{F}] \leq \delta$. Here $\mathfrak{F}$ is some set whose

complement specifies a set of states to be avoided. Those can be states that lead to mission failure. For instance, in collision-avoidance scenarios it often is desired to bound the probabilities of collisions [37, 39] and $\mathfrak{F}$ would be free-space. For now, we will work with instantaneous constraints. As will be discussed in Ch. 6, by utilising the union bound, a trajectory constraint can be enforced by multiple instantaneous constraints.

In SMPC, at a given point in time t_0, planning is done by solving the optimisation problem (OP):

$$\min_{u_{0:T-1}} c(u_{0:T-1}) \tag{2.4}$$

$$\text{s.t.:} \forall t \in \mathcal{I}_t : u_t \in \mathcal{U} \tag{2.5}$$

$$\forall t \in \mathcal{I}_t : x_{t+1} = f(x_t, t, u_t, \nu_t) \tag{2.6}$$

$$\Pr[(x_t) \notin \mathfrak{F}] \leq \delta \tag{2.7}$$

where u_k is the control input planned to be effected at time $t_0 + k$.

The found plan trajectory $u = u_{0:T}$ could be executed in an open-loop fashion. In that case the entire control sequence is applied, without further updates, until time $t_0 + T$ is reached. Alternatively, if the underlying process is slow relative to the duration it takes to solve the OP, one can also employ aforementioned receding horizon control (RHC) approach where only the first control input u_0 is applied and in the next time step, an analogous optimisation problem is solved with the horizon shifted one time step into the future. The RHC approach is the most dominant setup in MPC [59, 161]. Since after every application of the first control input of the open-loop sequence, the control sequence is updated based on a new state measurement, the resulting controller is of the closed-loop variety. Injection of additional constraints or suitable choices of cost functions have allowed a number of authors to give closed-loop stability guarantees under the assumption of determinism or uncertainty with bounded-support (e.g. [57, 95, 161]). For more detailed discussions surveying these results the reader is referred to [59, 62, 144].

In the case of stochastic uncertainty, in order to be able to solve the optimisation problems, the probabilistic constraints are usually converted into deterministic ones in a conservative manner. By *conservative*, we mean that the probabilistic constraints are replaced by deterministic constraints such that the set $\mathbb{T}$ of trajectories that satisfy the deterministic constraints also satisfy the probabilistic ones. In the case of uncertainty with bounded support, it is possible to find conservative constraints that can be met almost surely. In such cases, the resulting sets of states $\mathbb{T}$ that satisfy such constraints are frequently referred to as (stochastic) *tubes*. Based on these tubes a variety of closed-loop stability results have been given [57, 59, 167]. Generally, strong results such as asymptotic stability in the mean-square can only be given for SDEs with multiplicative, but no additive noise. Weaker forms of stability have been established for additive noise under certain assumptions. For

instance, Cannon, Kouvaritakis and Wu [57] designed a cost function yielding convergence of the stage cost (involving a scaled variance term) in systems with both multiplicative and additive noise. Unfortunately, to the best of our knowledge, all existing stability guarantees in SMPC for systems with additive uncertainty hinge on the bounded-support assumption and typically also assume independent increments as well as a convex state-space. A variety of works considers noise with distributions of non-compact support, in particular with i.i.d. Gaussian noise. The probabilistic constraints frequently are approximated on the basis of distributional knowledge either by numerical integration or using the error function to bound the probability mass [6,7,224]. None of these methods provide guarantees of stability and, again, constraints are assumed to be convex. Indeed, as pointed out by [59], SMPC methods assuming unbounded noise (such as Gaussian noise) inevitably suffer from the difficulty of establishing recursive feasibility (and hence stability) due to the possibility of "infinite size" disturbances.

Therefore, no existing work that assumes unbounded uncertainties seems to exist that provides any stability guarantees. Furthermore, most works assume convex constraints and make assumptions on the distributions. Finally, most MPC work is based on linear dynamics with i.i.d. uncertainty. Unfortunately, if we desire to control agents interacting in a common environment with mutually exclusive resources and epistemic uncertainty about their non-linear dynamics, these assumptions normally do not hold. Therefore, we have developed our own approaches [48, 158, 159] that are applicable without distributional assumptions and can cope with non-convex probabilistic constraints as would be required in collision-avoidance. Our work, part of which we will describe in Ch. 6, took inspiration from pre-existing work in MPC-based obstacle avoidance, which will be briefly discussed next.

Mixed-integer programming for MPC with obstacle avoidance constraints

MPC comes with strong guarantees (such as stability) when all constraints are convex. However, obstacles inherently result in non-convex planning spaces which is typically addressed with disjunctive constraints yielding an overall planning problem that is equivalent to and often stated as a mixed-integer program. Assume polygonal obstacles O_j in state space $\mathcal{X}$, which can be represented by the intersection of say, $k \in \mathbb{N}$ half-spaces: $O_i = \{x \in \mathcal{X} | \langle x,n_{i1} \rangle \leq b_{i1}, \ldots, \langle x,n_{ik} \rangle \leq b_{ik}\}$ $(i = 1, ..., k)$. Hence, a polytope can be specified by a conjunction of linear constraints. Therefore, by de Morgan's rule, if all obstacles are polygonal, the free-space $\mathfrak{F} = \mathcal{X} - \{\bigcup_i O_i\}$ is the disjunction of linear constraints. That is,

$$\mathfrak{F} = \{x \in \mathcal{X} | \bigwedge_i (\bigvee_{j=1}^{k_i} \langle x,n_{ij} \rangle > b_{ij})\}. \tag{2.8}$$

We can process these disjunctive constraints with disjunctive programming methods [18]. One way to solve this is with a branch-and-bound approach yielding the necessity to solve up to a number of subproblems

that is exponential in the number of disjunctions [39]. The alternative approach we have followed in our implementations in Ch. 6 converts the disjunctions into mixed-integer (linear) constraints by introducing integer auxiliary variables that encode logic propositions between constraints (e.g. [30, 38, 89, 222]).

A mixed-integer linear programme (MILP) is a linear programme where any subset of the variables may be constrained to assume integer values only. That is, a MILP is an optimisation problem with a (piece-wise) linear objective function and a conjunction of linear and integer constraints.

To define the restriction to $\mathfrak{F}$ as a MILP, its defining constraints need to be stated as mixed-integer linear constraints. Examining Eq. 2.8, we see that therefore, we need to express the disjunction in mixed-integer linear form.

One way of doing so is by the so-called *Big-M method* [30,222]. Here, a disjunction of the form $\bigvee_{j=1}^{k} \langle x,n_{ij}\rangle > b_{ij}$ is converted into mixed-integer linear constraints by introducing k binary *slack* variables $e_1, \ldots, e_k \in \{0, 1\}$, a very large (in proportion to the constituent variables of the constraints) number $M > 0$ and imposing the constraints: $\langle x,n_{i1}\rangle + e_1 M > b_{i1}, \ldots, \langle x,n_{ik}\rangle + e_k M > b_{ik}$. If some e_j is set to one, the slacked constraint $\langle x,n_{ij}\rangle + e_j M > b_{ij}$ holds true even if the corresponding original constraint $\langle x,n_{ij}\rangle > b_{ij}$ is violated. To ensure that at least one of the original constraints $\langle x,n_{ij}\rangle > b_{ij}$ holds, we add the binary constraint $\sum_{j=1}^{k} e_j < k$ which ensures that at least one e_j is not active.

The restriction to piece-wise linear objective functions allows one to express cost functions that penalise control-relevant quantities such as $1-$norm distances to a goal or the sum of the absolute values of the control outputs (which is directly related to control energy). By the introduction of additional slack variables such cost functions can be converted into linear ones bounded by additional constraints [32, 222]. For instance, consider the cost function

$$c(u_{0:T-1}) = f(x_T) + \sum_{t=0}^{T-1} q^\top |x_t| + r^\top |u_t| \tag{2.9}$$

$$\text{s.t.: } \forall t \in \{0, \ldots T-1\} : x_{t+1} = Ax_t + Bu_t \tag{2.10}$$

where $f(x_T) = p^\top |x_T|$ is a terminal cost. For example, defining q to be a vector of ones would mean that $q^\top |x_t| = \|x_t\|_1$. That is, it would encourage control sequences that lead to state sequences whose cumulative 1-norm distance is not far away from goal state 0. On the other hand, setting the components of r to positive values would balance this speedy goal convergence with expended control magnitude.

The optimisation problem can be restated in MILP form as follows [222]:

$$c(u_{0:T-1}) = \min_{w_t, v_t} p^\top w_T + \sum_{t=0}^{T-1} q^\top w_t + r^\top v_t \tag{2.11}$$

$$\text{s.t.: } \forall t \forall j, k : x_{t+1} = Ax_t + Bu_t \tag{2.12}$$

$$x_{tj} \leq w_{tj}, \ -x_{tj} \leq w_{tj} \tag{2.13}$$

$$u_{tk} \leq v_{tk}, \ -u_{tk} \leq v_{tk} \tag{2.14}$$

where $t \in \{0, \ldots, T-1\}$ denotes the time index and the j and k index the components of slack variables w_t and v_t, respectively. The slack variables have converted the convex penalties to linear ones (in addition to linear constraints that enforce that the absolute values are realised). For instance, Eq. 2.13 ensures that $|x_{tj}| \leq w_{tj}$. Since the state is constrained by the dynamics, this ensures that the slack variables are bounded from below while the minimisation will ensure that the constraints are active at the optimum (thus, $|x_{tj}| = w_{tj}$). Hence, the convex problem was converted into an equivalent linear optimisation problem.

Of course, the introduction of the *slack variables* increases the dimensionality of the optimisation problem. Furthermore, if control energy is to be taken into account in the planning process, it would be more natural to include a penalty term $\sum_t u_t^\top R_t u_t$ for some pos. def. matrix R_t instead of using the $1-$norm penalty. This can be done utilising mixed-integer *quadratic* programming (MIQP) instead of MILP. As with MILP there exist nowadays potent MIQP solvers in optimisation suites such as CPLEX [119].

While mixed-integer programming (MIP) is known to be NP-hard in general [102], in practice there exist advanced solvers such as CPLEX [119] which typically find solutions to linear problems of lower complexity quite rapidly.

Nonetheless, optimising a MIP takes time. Therefore, when applying MPC in receding horizon mode, computational effort may need to be decreased in order to meet application-dependent real-time requirements. Aside from the number of constraints the key factor impacting optimization duration (even in average-case analysis) is known to be the dimensionality of the feasible set. Thus, in order to reduce computational effort as much as possible we could sparsify the temporal resolution of the optimized trajectory. Since such a sparsification may negatively affect performance and obstacle avoidance, we propose interpolating between the time steps in order to detect collisions in between and refine the temporal resolution dynamically, as required. An approach similar in spirit was proposed by Earl and D'Andrea [89] who interpolated between the optimized time steps based on perfect knowledge of deterministic and completely known, linear dynamics.

As a potential way to extend their approach to stochastic settings, we could utilise the collision-criterion functions we derive in Ch. 5 to incrementally refine the temporal resolution of the time-discretisation as needed. Being a conservative collision detection method for continuous-time, the criterion function approach

could be employed in a stochastic setting, as long as the first two moments can be evaluated. While we derive and employ the criterion function approach in the context of continuous control (Ch. 5), the combination with discrete-time MPC outined in this paragraph will have to be covered in future work.

DMPC

MPC in systems with multiple controllers (agents) is often referred to as distributed model-predictive control. As always, the literature is vast and we will have to limit our discussions to the most relevant works. For an up-to-date survey of DMPC methods the reader is referred to [178].

A straight-forward extension to multi-agent MPC is to combine the individual agents' optimization problems into one high-dimensional joint optimization problem that needs to be solved by a centralized planner [90, 91, 199, 222]. For the reasons mentioned above such a centralized approach can be expected to scale poorly in the number of agents. Moreover, a central planner constitutes a communication choke point and a single point of failure. To the best of our knowledge, extensions of MPC methods to the multi-agent case have been largely restricted to centralized methods, assume convex constraints or are restricted to the deterministic setting or bounded-support uncertainties [178].

The DMPC works that do consider non-convex constraints are robust control approaches. As far as we are aware, they tend to achieve coordination either by solving a centralized integer-programme or achieving collision avoidance by a fixed-priority type scheme [150, 199, 251]. Recent work by Kuwata et. al. [149] modified the fixed priority scheme by having agents that are close to each other perturb each other's decisions slightly and reiterating the process repeatedly until a local optimum is found. While they have the advantage of being able to give stability guarantees, the assumption of interval-bounded uncertainty limits their applicability to more general uncertainties of a stochastic nature. However, since they give guarantees on robust feasibility, it would be interesting to consider these methods' use for the control of agents who employ our kinky inference rule (see Ch. 4) for learning the dynamics. Furthermore, in Ch. 6, we propose a number of coordination mechanisms and describe how they can be brought to bear in SMPC with probabilistic collision-avoidance constraints both with and without distributional assumptions. In complete analogy, our methods could also be employed in the robust control context. This merger would allow us to achieve optimal coordination of learning agents with robust control guarantees.

2.2.2. Remarks on the relationship to planning as optimisation with soft-constraints and collision avoidance with MDPs

For the most part, we consider planning under hard constraints. That is, we assume a planning problem to be an optimisation problem of the form $\min_p c(p)$, subject to: $p \in F$ where F is the *feasible set*. By contrast,

many planning techniques assume that constraint violations, i.e plans $p \notin F$ are discouraged by a cost penalty $\beta(p; C)$ with parameter C. Here C is chosen so that $\beta(p; C)$ assumes high values for $p \notin F$.

A prominent example occurs in linear programming: a linear programme (LP) of the form

$$\min_{p} \langle c,p \rangle \tag{2.15}$$

$$\text{s.t.: } \langle n_j,p \rangle \leq \gamma_j \quad (j = 1, ..., n) \tag{2.16}$$

is equivalent to

$$\max_{\lambda_1,...,\lambda_n \in [0,\infty]} \min_{p} \langle c,p \rangle + L(p, \lambda_1, ..., \lambda_n) \tag{2.17}$$

where $L(p, \lambda_1, ..., \lambda_n) = \sum_{j=1}^{k} \lambda_j (\langle n_j,p \rangle - \gamma)$ is called the *Lagrangian dual function* [43]. The equivalence is easy to see when realising that by von Neumann's theorem, min and max can be exchanged (for a proof for convex-concave objectives see [46]). Provided that parameter $C = \infty$, one then notices that the ensuing penalty function $\beta(\cdot, C) = \max_{\lambda_1,...,\lambda_n \in [0,C]} L(\cdot, \lambda_1, ..., \lambda_n)$, which penalizes the constraint violations, assumes infinite values iff a constraint is violated and is zero otherwise [43]. Therefore, a minimiser of the cost will avoid the violation of the constraint, provided such a solution exists.

In practice, a relaxed version with $C < \infty$ is solved. Consequently, the ensuing solution only is approximately feasible with respect to the original constraints which are often referred to as *hard constraints*. Since in the relaxed version, these constraints can be violated, albeit for some additional cost, the penalty terms β consists oft are often referred to as *soft* constraints.

Problems with soft constraints can often be solved by dynamic programming [71] and are the basis of a large number of approaches in artificial intelligence (AI) including DCOPs (see Sec. 2.3) and the majority of Markov Decision Process (MDPs) planning approaches. Most MDP planning in AI assume discrete time and space. Since fine-grained discretisation renders the problems rapidly intractable, in applications to robotic systems, frequently policy search methods are more popular where search is conducted in the parameter space of a control policy and is the basis of the resulting expected cost (for a recent survey see [77]).

Recently, Ono, Williams and Blackmore [188] have discussed the employment of MDP-based planning for probabilistic collision avoidance in comparison to mixed-integer approaches. As explained by the authors, a probabilistic collision avoidance constraint of the form $\Pr[x \in O] < \delta$ (where O is an obstacle) could be approximated as an expected soft-constraint $\beta(\cdot; \lambda) = \lambda \langle \mathbf{1}_O(\cdot) \rangle$ where, as always, $\langle \cdot \rangle$ denotes an expectation operator and $\mathbf{1}_O(\cdot)$ the indicator function of set O. Thereby, probabilistic collision avoidance could be attempted with reinforcement learning and MDP-based approaches that work with expected costs. However, the authors make a case that it is unclear how to set parameter λ optimally such that the chance constraint is

adhered to. Owing to this difficulty, they demonstrate how MDP-based planning tends to generate controls that result in trajectories that are either too conservative or that collide too frequently with the obstacle. They conclude that MDP-based planning is more suitable in scenarios where there is a natural notion of cost rather than the desire to bound a collision probability.

In the light of our discussion above, one might wonder whether the right approach was to try considering a soft constraint $\beta(\cdot; C) = \max_{\lambda \in [0,C]} \lambda \langle \mathbf{1}_O(\cdot) - \delta \rangle$. For large $C > 0$, this approximates the hard probabilistic collision avoidance constraint well, since $\Pr[x \in O] = \langle \mathbf{1}_O(\cdot) \rangle$. Unfortunately, to completely embed this cost into MDP planning, the cost has to be expressed as the expectation of a cost rather than a function of an expectation. That, is we could seek to restate the soft constraint as $\langle \max_{\lambda \in [0,C]} \lambda(\mathbf{1}_O(\cdot) - \delta) \rangle = \langle \max\big(0, C(\mathbf{1}_O(\cdot) - \delta)\big) \rangle$.

The function $x \mapsto \max\big(0, Cx\big)$ is a hinge-loss and as such strictly convex. Therefore, pulling in the expectation into the hinge-loss can reduce the cost (by Jensen's inequality for strictly convex functions). Hence, solving an MDP with expected stage cost $\langle c^{\mathfrak{a}}(p^{\mathfrak{a}}) + \max\big(0, C(\mathbf{1}_O(x_t^{\mathfrak{a}}) - \delta)\big) \rangle$ penalises the violation to a greater degree than a stage cost $\langle c^{\mathfrak{a}}(p^{\mathfrak{a}}) \rangle + \max\big(0, C\langle \mathbf{1}_O(x_t^{\mathfrak{a}}) - \delta \rangle\big) = \langle c^{\mathfrak{a}}(p^{\mathfrak{a}}) \rangle + \max\big(0, C(\Pr[x \in O] - \delta)\big)$ which has the expectation inside the constraint-violation penalty term. Therefore, the conversion of the last soft constraint to an MDP-amenable form (that has the expectation on the outside) is conservative. This is desirable when trying to softly enforce a hard constraint.

Nonetheless, how to set C in a principled manner and the consequences of its choice to the learnability of the pertaining value function and hardness of the decision making process would still have to be investigated. For instance, one would imagine that if the problem is solved with a discrete MDP then choosing a larger C makes the value function change more rapidly. In this case, a finer state-space discretisation is required to avoid greater approximation error. This in turn, would increase computational requirements for planning. Investigating this has not been attempted in the thesis but might be an interesting direction to consider.

We should also like to stress that, non-withstanding these issues and the concerns expressed in [188], reinforcement learning and planning in (Po-) MDPs remain, especially in discrete domains, highly popular approaches for planning and learning in uncertain systems (e.g. [29, 77, 187, 202, 245, 249] to name a few). While our own work focusses on finite-horizon applications, future work might look into the matter in how far our techniques can be combined with results developed in infinite-horizon problems such as POMDP planning and reinforcement learning.

2.3. Multi-agent coordination

An agent is an entity that reasons and acts in an environment. As the name suggests, the study of *multi*-agent systems pertains to systems where multiple agents inter-act in a common environment. Naturally, such

systems arise in a wide range of contexts and therefore, multi-agent systems are studied through the lenses of a diverse spectrum of fields, including the social sciences and economics, biology, control and computer science.

In this section, we will briefly touch upon several strands of work in multi-agent systems that seem most related to the work presented in this thesis. In the apparent absence of a unified treatment of multi-agent systems that reaches across inter-disciplinary boundaries the reader is referred to the following more specialised surveys and books [75, 82, 178, 228, 229, 265] for a coverage that is more extensive than can be offered within the scope of this thesis.

In contrast to economic settings where agents are assumed to act selfishly [75], we restrict ourselves mainly to cooperative, artificial agents which we will assume to be compliant with an imposed coordination protocol. Fitting into this approach is multi-agent and robot control [265] which focusses on problems that reflect the specific nature of robotic agents and their physical environment. Here, our interest lies in collision avoidance applications which present a particularly challenging multi-agent problem due to strong coupling between the agents and inherent non-convexity of the planning problem. In connection to the unifying theme of this thesis, our proposed approaches are tailored to coordinating agents in a physical, dynamic world under uncertainty and enable them to jointly conceive plans or controls that takes their uncertainty into account when aiming to avoid collisions.

As we will show in Chapters 5 and 6, such planning and control problems can be stated in terms of a solution to a constrained optimisation problem. The degree of challenge these problems present both from a tractability and agents perspective depend on the mathematical nature of the optimisation problem and the inter-action constraints. For instance, in cases where the joint planning problem can be modelled as a linear programme (LP) or a convex-concave minimax problem, a plethora of tractable multi-agent planning methods have been proposed and applied to domains such as planning in MDPs [112] and multi-agent flow routing [47]. Some of them rely on decomposition approaches such as Danzig-Wolfe decomposition [74, 112] where the large LP is broken up into repeated interactions between sub-problems with guaranteed convergence to the optimal solutions. Even the scalability in the number of such inter-actions no longer presents a problem. For instance, Calliess and Gordon [47] have presented a learning-based market-mechanism that solves any multi-agent convex problem with linear constraints approximately with error scaling sub-linearly in the number of agents.

Unfortunately, many other important (multi-agent) optimisation problems are provably hard. One very expressive subclass of such optimisation problems are mixed-integer programs (MIPs) in general and *mixed-integer* linear *programs* (MILP) or a mixed integer quadratic programs (MIQP) in particular. Among many other applications, mixed-integer programming has been utilised for MAP inference in graphical models

[211], target assignment [3], logic-based planning [109, 259], multi-UAV path planning [114] and model-predictive control with probabilistic collision-avoidance [38, 158]. An example we will consider later on is particle-based stochastic model-predictive control that has been considered for single-vehicle control and trajectory planning under uncertainty in recent years [38]. The drawn particles can serve to approximately bound the probability of a collision with an obstacle via chance constraints that are added as binary constraints to the MILP formulation of the vehicle's cost-optimizing control problem [38]. The resulting plans (sequences of control inputs) are shown to result in low-cost trajectories that avoid all obstacles with adjustably high certainty.

A simple method to extend single-agent problems to multi-agent problems is to combine their individual optimization problems into one large, centralized MILP/MIQP (e.g. [222]). While delivering cost-optimal results, such approaches have the architectural disadvantages of centralized approaches, and scale poorly in the number of agents and interaction-constraints. Therefore, their application is typically restricted to coordination tasks of low complexity. As finding a socially optimal solution is known to be NP-complete, most practically applicable coordination methods constitute a compromise between tractability and optimality.

Multi-agent coordination is a broad topic with numerous strands of works ranging across different communities. In Ch. 6, we present an approach to multi-agent collision avoidance and control that is germane to a number of these strands. It is beyond the scope of this work to present an exhaustive survey of the extensive body of previous work that ranges across various disciplines. For surveys focussing on market-based approaches refer to [82]. For computational mechanism design suitable for selfish agents refer to [75]. For a general introduction, the reader is referred to a fairly recent book on multi-agent systems from a computer science perspective [228].

As a rather coarse taxonomy, present methods can be divided into centralized, distributed and decentralized approaches. *Centralized approaches* (e.g. [222] [194]) typically rely on combining the individual agents' plans into one large, joint plan and optimizing it in a central planner. Typically, they are guaranteed to find an optimal solution to the coordination problem (with respect to an optimality criterion, such as the sum of all costs). However, since optimal coordination is NP-hard it is not surprising that these methods scale poorly in the number of participating agents and the complexity of the planning environment. With worst-case computational effort growing exponentially with the number of agents, these methods do provide the best overall solutions, but are generally intractable except for small teams.

In contrast, distributed and decentralized methods distribute the computational load on multiple agents and, combined with approximation methods, can factor the optimal problem into more tractable chunks.

The distinction between distributed and decentralised multi-agent systems lies in the degree to which communication is deemed possible.

In *decentralised* coordination, often *local interaction rules* or observation-based behavioural patterns are designed to induce a global behaviour that emerges with little or no communication overhead [20, 178, 216, 253]. For instance, based on a specific robot motion model, Pallottino et. al. [193] propose interaction policies that result in guaranteed collision avoidance and can accommodate new robots entering the system on-line. Furthermore, under the assumption that robots reaching their goals vanish from the system, the authors prove that eventually all robots will reach their respective destination locations.

The methods based on local interaction rules have the advantage of being fully decentralised and saving communication and requiring little computation. This allows them to be deployed online and has enabled authors of these methods (by employing massively parallel computing) to simulate collision avoidance test cases with large numbers of agents. However, to the best of our knowledge, the solutions do not consider any type of cost notion (and hence, do not optimise for a notion of social cost) and do not take uncertainty or dynamic models into account. However, it may be worthwhile endowing their methods with an explicit error model and performing a similar analysis to that we provide in Sec. 6.3.1.

The second class of distributed methods focusses on the development of mechanisms where coordination is achieved through information exchange succeeding the distributed computations.

Distributed optimization techniques have been successfully employed to substitute the solution of a centralized optimization problem with the solution of a sequence of smaller, decoupled problems (e.g. [184], [117] [47], [185], [29] and [112]).

For example, Bererton et. al. [29] employ Dantzig-Wolfe Decomposition [74] to decentralize a relaxed version of a Bellman MILP to compute an optimal policy. However, due to the relaxation of the collision constraints, collisions are only avoided in expectation.

Later work has focussed on decentralised POMDPs (Dec-POMDP) [22, 107, 174, 175] for finding locally optimal policies by *alternating optimisation approaches* [174, 175]. In the latter, policies are sequentially updated over the course of a number of iterations. In each iteration, the policies of all agents except one are held fixed. The latter computes his best response policy to the fixed policies of the other agents. The result is communicated to the other agents among which a new agent can alter his policy in response. The procedure continues until an equilibrium is reached.

In parallel to aforementioned strands of work, a community has formed under the banner of *distributed constraint optimisation (DCOP)* [97, 266, 267]. An introduction into DCOPs is given in [267]. Typically, DCOP methods model constraints of an optimisation problem as soft constraints, i.e. via penalty cost terms in the objective function. Often dependencies are modelled by factor graphs or dependency graphs. These serve as the communication topologies of repeated optimisation efforts of multiple agents solving the interdependent sub-problems. Yeoh [267] classifies distributed DCOPs into the categories of search-based methods

and inference methods and points out that the former need to send a number of messages increasing exponentially with the number of agents, whereas the latter have exponential (in the size of the factor graph) memory requirements.

For this reason, *Max-Sum*, a message passing algorithm adapted from probabilistic inference, has become a popular choice for solving DCOPs in recent years and has been applied to solve a large number of multi-agent problems [96, 209]. As with most other DCOP methods, agent interaction is modelled to take place exclusively via the agents' cost functions and coordination is achieved by message passing in a factor graph that represents the mutual dependencies of the coordination problem.

Since in our work, we mostly consider collision avoidance as a test scenario, we will have to contemplate the applicability of these methods. While dualisation of our inter-agent (collision-avoidance) constraints into the objective function could be leveraged to translate our setting into theirs, several problems remain. First, the resulting factor graph would be exceptionally loopy and hence, no performance or convergence guarantees of max-sum can be given. Second, the interconnecting edges would have high weights (cf. [97]) whose removal would correspond to a relaxation of the collision-avoidance constraints and hence, render pruning-based max-sum-based methods [97] inapplicable. While message passing has been reported to work sometimes in practice even in scenarios with loopy factor graphs, our tests with maxsum in the context of graph routing confirmed that max-sum often failed to generate feasible plans. Recently, an improved version of max-sum, *max-sum-AD*, has been proposed that guarantees convergence even in the presence of cycles [276]. While it might be generally interesting to test it for collision avoidance, max-sum-AD requires more computational effort than max-sum and does not guarantee to be optimal.

Another related community is concerned with distributed constraint satisfaction problems (DiCSP) [268]. In contrast to DCOPs, constraint satisfaction problems are concerned with hard constraints. However, typically no notion of a cost function is considered. Furthermore, the number of decisions are restricted to finite sets. Therefore, a reduction to the general situation we consider in collision avoidance where actions (and possibly time) might be continuous would seem to be less obvious than an application to the graph planning problems we consider in Sec. 6.2.

While for the purposes of this work, the graph planning application merely serves as a simple test bed to explain our general methods in, research on distributed and decentralised algorithms for a slightly more constrained variety of graph planning than we consider, has attracted much research attention in its own right under the banner of *multi-agent path-finding*. Being a sub-community within the larger communities of artificial intelligence and robotics, much work has recently focussed on the development of specialised algorithms for solving these graph planning problems either with optimal or sub-optimal algorithms (e.g. [48, 76, 128, 237, 273, 274]). We would like to stress that, in a recent paper, Roeger and Helmert try to raise

awareness in the artificial intelligence community that the problem of sub-optimal (i.e. in the absence of a cost) path-finding had been exhaustively addressed from a group theoretic perspective in the nineteen-eighties under the name of *pebble motions on graphs* by Kornhauser [141]. In very recent work, Wilde et. al. [76] argue that their own method would probably be a preferred choice for path finding since Kornhauser's work was hard to understand and had been reported to exhibit worse average-time performance than a competing method [244].

Despite this and although our coordination methods of Ch. 6 are cast in a more general optimisation framework, we would be interested in how far arguments contained Kornhauser's work might be generalised to analyse our methods.

Furthermore, while multi-agent path-finding in graphs is a more restricted domain, we have discovered work that has taken place in this context and shares similar ideas with our own work (as well as with other strands including DCOPs and DMPC methods). For instance, utilising a centralized MILP for solving multi-agent trajectory planning in graphs by reduction to multi-commodity flows was also proposed by Yu and Lavalle [273, 274]. A year earlier, we have first described the same trick to achieve this reduction and utilised essentially their approach for our baseline comparison for our lazy auction approach, not realising the centralised method might be novel [48, 121]. Interestingly, Yu and Lavalle reported rapid optimal planning success with up to 150 agents in densely packed graph environments. This suggests that our methods are much more scalable when using the Gurobi MILP solver [113] than we could anticipate based on our own simulations presented in Sec. 6.2 that utilised Matlab's inbuilt *bintprog*. As an additional similarity, we recently found out that the idea of combining conflict detection with an incremental merging of sub-problems that is at the heart of our SDCG method (cf. Sec. 6.1.2) can also be found in recent developments in the multi-agent path-finding community [237].

While the basic ideas seem similar, we have phrased our approach in the more general context of MILP [54] which widens their applicability substantially. In particular, in Ch. 6, we show how to connect our methods to collision avoidance in SMPC which is possible due to their formulation in a MILP context. However, it may be fruitful to consider exploring in how far recent and future improvements of methods developed within the multi-agent path-finding community could be combined with our methods and hence, be generalised into the MILP framework. This would render them applicable to distributed SMPC or other domains that can be framed in an optimisation context. We believe that the work presented in Ch. 6, provides the links necessary for such investigations.

Some distributed optimisation methods (e.g. [47, 112]) have a market interpretation due to passing of costs pertaining to Lagrangian multipliers among the sub-problems.

Generally, *market-based approaches* have been heavily investigated for multi-robot coordination over the

past years [241] [103] [82]. Among these, *auction mechanisms* allow us to employ techniques drawn from Economics. They are attractive since the communication overhead they require is low-bandwidth due to the fact that the messages often only consist of bids. However, as optimal bidding and winner determination for a large number of resources (as typically encountered in multi-robot problems) is typically NP-hard, all tractable auction coordination methods constitute approximations and few existing works provide any proof of the social performance of the resulting overall planning solution beyond experimental validation [**?**, 151]. An exception are *SSI-auctions* [**?**, 151]. For instance, Lagoudakis et. al. [151] propose an auction-based coordination method for multi-robot routing. They discuss a variety of bidding rules for which they establish performance bounds with respect to an array of team objectives, including social cost. While multi-robot routing is quite different from the motion control problem, we consider some of their bid design to be related in spirit to the one we propose. It may be worthwhile considering under which circumstances one could transfer their bidding rules and theoretical guarantees to our setting. One of the main obstacles here may be the fact that in SSI auctions, a single multi-round auction for all existing resources (or bundles) is held. This may be difficult to achieve, especially if we, as in Sec. 6.3.1, desire to avoid prior space discretisation and take uncertainty into account.

While most market-based planning systems described before offer distributed, applicable and computationally feasible approaches, almost none of them are concerned with self-interested behaviour in the participating agents. Similarly, our lazy auction method, introduced in Ch. 5 and Ch. 6, is used primarily as a tool to achieve socially desirable performance in collectives of truthful and cooperative agents. However, if we consider human-agent collectives, the necessity may arise to address the issue of self-interested agents which may not be truthful during bid computation if it is to their benefit. Thereby, self-interest may hurt the common good. To avoid such a behaviour one would desire mechanisms that reduce an agent's incentive to deviate from prescribed behaviour.

Designing mechanisms capable of handling selfish behaviour is exactly field of study of *mechanism design and auction theory* [132]. In the economic theory of mechanism design, the goal is to elicit private information from each of multiple agents in order to select a desirable system-wide outcome, despite agents' self-interest in promoting individually beneficial outcomes. Auctions provide a canonical example, with information elicited in the form of bids, and an allocation of resources and payments defining an outcome. The classic branch of the field is highly developed and the quantity of related publications is vast. A prominent class of auctions is the *Groves* class [111]. They are known to be *efficient* (i.e. socially optimal) and *incentive-compatible* (i.e. no single agent has an incentive to unilaterally submit a bid that does not reflect its true valuation of the good). An important subclass of the Groves mechanism are the *VCG*-auctions that were extracted from the findings of a triplet of papers [68, 111, 256] by Vickrey, Clarke and Groves. They

are known to be efficient and incentive compatible. However, like members of the Vickrey family, they are generally not budget-balanced.

Unfortunately, the basic setting of VCG-mechanisms only considers a single or a finite number of independent, simultaneously available goods and hence, is not immediately applicable. For such cases, there are two primary kinds of auctions – *combinatorial* [217] and *sequential* [42]. In the former, all agents are required to compute valuations and bids for all subsets of resources. This entails a combinatorial blow-up and the optimal combinatorial allocation problem is known to be NP-hard [217]. Sequential auctions are predominantly used when goods are auctioned at different points in time. However, they have also been suggested as a computationally more feasible (but suboptimal) alternative to combinatorial auctions [42, 139]. Unfortunately, to the best of our knowledge, existing approaches for solving sequential auctions still assume that the number of resources are finite and the schedule determining in which sequence each resource is to be auctioned is known to all bidders ahead of time [42, 139].

In contrast, in the collision avoidance situations we consider, the goods correspond to points of trajectories in potentially continuous spaces. Hence, we cannot expect to be able to compute bids for an uncountably-infinite number of resources. Therefore, we relied on collision detection and institution of virtual obstacles to define resources dynamically during run-time. However, in settings with discrete resources we may consider replacing our lazy auction mechanism by sequential auction approaches. In addition, we may consider how to modify our lazy auctions to become incentive-compatible. For instance, we could try introducing play money into our lazy auction mechanism (cf. Ch. 5, Ch. 6) and make payments according to a second-price auction. Investigating the impact of such modifications in terms of mechanism design properties will have to be deferred to future work.

Among all multi-agent path planning approaches, *(fixed) priority methods* are perhaps the most established ones. Their simplicity and computational tractability (even for large agent collectives) may be the reason why they have attracted sustained popularity in practice, especially in control and robotics [94], [27], [255], [58] [150]. In its most basic form introduced by Erdmann and Lozano-Perezx [94], agents are prioritized according to a fixed ranking. Planning is done sequentially according to the fixed priority scheme where higher ranking agents plan before lower ranking agents. Once a higher ranking agent has concluded planning, his trajectories become *dynamic obstacles*[2] for all lower ranking agents, which the latter are required to avoid. If independent planning under these conditions is always successful, coordination is achieved in A planning iterations that spawn the necessity to broadcast $A - 1$ messages in *t*otal (plans of higher priority agents to lower priority ones) where A is the number of agents.

[2]The notion *dynamic obstacle* loosely corresponds to our *virtual obstacles* (cf. Sec. 6.3.1). The difference is that our virtual obstacles are only present at a particular time step whereas the dynamic obstacles span the whole range of all time steps. Furthermore, we described how to adjust the box-sizes to control the collision probability in the presence of uncertainty.

By contrast, in our auction mechanism presented in Ch. 5, Ch. 6 A such messages need to be sent *per coordination iteration*. Although our results indicate that the number of these iterations scale mildly in the number of agents and obstacles in typical obstacle avoidance settings, such an additional computation and communication overhead needs to be justified with better coordination performance. Our experiments in Ch. 6 indeed illustrate the superior performance of our flexible bidding approach over fixed priorities. Furthermore, it is possible to manage communication via resource agents. So, whenever, an agent plans to use a resource (e.g. nodes in a graph, locations in state-space) it communicates with the resource agent who is collecting all resource requests. This can reduce the total amount of communication required [47].

In priority methods, the overall coordination performance depends on the choice of the ranking and a number of works have proposed methods for a priori ranking selection (e.g. [27]). Conceivably, it is possible to improve our method further by optimizing its in-auction prioritization (agent indexing) with such methods. Exploring how to connect our mechanism to extensions of priority methods, such as [254], could have the potential to improve the communication overhead. Investigating the feasibility of such extension will have to be done in the course of future research efforts.

2.4. Mathematical background and preliminary derivations

This section collects a number of basic facts to be referenced throughout this manuscript. Most of the material presented is standard. However, at several occasions the research problems under investigation in this thesis required our own derivations which the reader can identify as being those statements that contain a proof.

Needless to say we cannot provide a comprehensive background on all the required mathematical pre-requisites that were needed to derive the results in the thesis and our exposition had to be highly selective. However, citations to related standard textbooks are provided for readers who wish to review the related concepts in greater depth.

2.4.1. Ordinary differential equations

Let $I \subset \mathbb{R}$ be an index set, $\mathbb{F}$ be a field (eg $\mathbb{R}$), $x : I \to \mathbb{F}^n$ be a vector valued function. A system of first-order differential equations can be written as

$$\dot{x} = F(t, x) \tag{2.18}$$

where $F : U \subset \mathbb{R} \times \mathbb{F}^n \to \mathbb{F}^n$ is a continuous (possibly) time-dependent vector field or *dynamical system*.

Definition 2.4.1 (Integral curve)**.** *An integral curve of a vector field $v : \Omega \subset \mathbb{R}^n \to \mathbb{R}^n$ is a differentiable curve $x : I \to \Omega$ such that $\dot{x}(t) = v(x(t)), \forall t \in I$.*

Note, that an integral curve of the vector field F in Eq. 2.18 could be a solution of the pertaining ODE.

Definition 2.4.2 (Initial value problem (IVP))**.** *An* initial value problem *(IVP) is a differential equation along with a constraint assigning a specified initial value to the desired solution.*

For instance, we may have the following IVP:

$$\dot{x} = F(t, x(t)),\ x(t_0) = x_0 \tag{2.19}$$

where $(t_0, x_0) \in U$ is the initial value the solution x is required to attain. The constraint $x(t_0) = x_0$ often is referred to as the *initial conditon*.

Under mild conditions on the vector field (such as Lipschitz continuity), the prominent Picard-Lindelöf theorem asserts that each IVP has a unique integral curve that satisfies the initial condition. In such cases one often refers to such an integral curve as *the solution trajectory*.

In the construction of solutions of differential equations it may often be helpful to consider an equivalent integral equation which can be found via the fundamental theorem of analysis (cf. [125]).

Lemma 2.4.3 (Integral version of a IVP (from [125]))**.** *Let $F : U \to \mathbb{F}^n$ continuous on the open set $U \subset \mathbb{R} \times \mathbb{F}^n$. Let $x : I \to \mathbb{F}^n$ continuous and $(t, x(t)) \in U, \forall t \in I$. x solves IVP 2.19 if and only if we have*

$$\forall t \in I : x(t) = x(t_0) + \int_{t_0}^{t} F(\tau, x(\tau))\, d\tau.$$

As a matter of fact, the integral version often is taken as the definition of the differential IVP. This has the advantage that the problem then is defined for the wider class of integrable vector fields. That is, the differential equation is defined even if its vector field F exhibits up to a countable number of non-differentiabilities.

2.4.2. Metrics and norms

Whether explicitly or implicitly, most machine learning methods are based on a mathematical notion of similarity or dissimilarity between the objects of enquiry in order to generalise beyond unobserved examples. We will now review some essentials pertaining to *metrics* which are the most commonly encountered mathematical models of distances. For additional reading, the interested reader is invited to refer to any standard textbook on metrics, Banach space, Hibert space or even analysis. As a good place to start the reader is referred to [125, 270].

Definition 2.4.4. *A pseudo-metric space is a pair $(\mathcal{X}, \mathfrak{d})$ where $\mathcal{X}$ is a set and $\mathfrak{d} : \mathcal{X}^2 \to \mathbb{R}_{\geq 0}$ is a mapping that satisfies the following three properties:*

1. $\mathfrak{d}(x, x) = 0, \forall x \in \mathcal{X}$,
2. $\mathfrak{d}(x, x') = \mathfrak{d}(x', x), \forall x, x' \in \mathcal{X}$,

3. $\mathfrak{d}(x, x') \leq \mathfrak{d}(x, x'') + \mathfrak{d}(x'', x'), \forall x, x', x'' \in \mathcal{X}$.

Mapping $\mathfrak{d}$ is called a pseudo-metric. If in addition, we also have $\forall x, x' : \mathfrak{d}(x, x') = 0 \Leftrightarrow x = x'$ then $\mathfrak{d}$ is called a metric and $(\mathcal{X}, \mathfrak{d})$ is called a metric space.

Related to the notion of a metric is the notion of a norm. A norm is a mapping on a space that can be interpreted as a length. Its axiomatised definition is as follows:

Definition 2.4.5. *A normed space is a pair $(\mathcal{X}, \|\cdot\|)$ where $\mathcal{X}$ is a vector space over the field $\mathbb{F}$ of real or complex numbers and $\|\cdot\| : \mathcal{X} \to \mathbb{R}_{\geq 0}$ is a mapping that satisfies the following three properties:*

1. $\forall x \in \mathcal{X} : \|x\| = 0 \Rightarrow x = 0$,
2. $\forall x \in \mathcal{X} \forall s \in \mathbb{F} : \|s\, x\| = |s|\, \|x\|$,
3. $\forall x, x' \in \mathcal{X} : \|x + x'\| \leq \|x\| + \|x'\|$.

Mapping $\|\cdot\|$ is called a norm.

Note, if we are given a normed space $(\mathcal{X}, \|\cdot\|)$, the mapping $\mathfrak{d} : x \mapsto \|x - x'\|$ is a metric on $\mathcal{X}$. Here, $\mathfrak{d}$ is sometimes referred to as the *canonical*, *standard* or *induced* metric. Note, while every norm induces a metric, not every metric can be induced by a norm.

A standard example of norms on d-dimensional vector space is the class of the so-called p-norms. Here, p is a number in $(0, \infty)$. Given a coordinate vector representation $(x_1, ..., x_d) \in \mathbb{R}^d$ of vector x with respect to a fixed basis, the p-norm is defined as $\|x\|_p = \left(\sum_{i=1}^{d} |x_i|^p\right)^{\frac{1}{p}}$. A special case is $p = \infty$. Here $\|x\|_p = \max_{i=1,..,d} |x_i|$ which arise as the limit of $p \to \infty$.

The norms are all equivalent. In d-dimensional vector space ($d \in \mathbb{N}$) that is to say that we have $\forall p, q \in (0, \infty], q > p : \|\cdot\|_q \leq \|\cdot\|_p \leq d^{1/p - 1/q} \|\cdot\|_q$. A consequential relation which we will make frequent use of are the inequalities:

$$\|\cdot\|_\infty \leq \|\cdot\|_2 \leq \sqrt{d}\, \|\cdot\|_\infty . \tag{2.20}$$

Note, that these inequalities hold for finite-dimensional vector spaces only. In infinite dimensional vector space, alternative relations exist [270] which may be in reverse to finite-dimensional case.

2.4.3. Matrix properties

Let $\mathcal{M}_d$ denote the set of all $d \times d$ square matrices over a given field ($\mathbb{C}$ or $\mathbb{R}$). Let $\sigma(M)$ denote the spectrum of eigenvalues of matrix M.

Definition 2.4.6 (Stable and positively stable matrices)**.** *A matrix M is stable if the real parts of its eigenvalues are negative, i.e. if Re $\sigma(M) \subset \mathbb{R}_-$.0 A matrix M is positively stable if the real parts of its eigenvalues are positive, i.e. if Re $\sigma(M) \subset \mathbb{R}_+$.*

The context of the definition is that a dynamic system $\dot{x} = Ax$ is globally asymptotically stable iff A is a stable matrix.

(Co-) variance matrices

We define the covariance between two d-dimensional random vectors X, y as

$$\mathrm{Cov}(X, Y) = \langle (X - \langle X \rangle)\,(Y - \langle Y \rangle)^\top \rangle \in \mathbb{R}^{d \times d}.$$

We adopt the nomenclature of Feller. That is for random vector X we define its *variance (matrix)* as the $d \times d$- matrix:

$$\mathrm{Var}[X] = \mathrm{Cov}(X, X).$$

Lemma 2.4.7. *Let $(X_i)_{i\in\mathcal{I}}, (Y_j)_{j\in\mathcal{I}}$ be two sequences of random vectors with index set $\mathcal{I}$, $|\mathcal{I}| > \min\{m, n\}$. We have*

$$\mathit{Cov}(\sum_{i=1}^{n} X_i, \sum_{j=1}^{m} Y_i) = \sum_{i=1}^{n}\sum_{j=1}^{m} \mathit{Cov}(X_i, Y_j).$$

Lemma 2.4.8. *Let X, Y be random vectors and A, B be matrices, d, c be two deterministic vectors. Let $(X_i)_{i\in\mathcal{I}}$ be a sequence of random vectors. We have*

1. $\mathit{Cov}(AX + c, BY + d) = A\,\mathit{Cov}(X, Y)\,B^\top$
2. $\mathit{Var}[c^\top X] = c^\top\,\mathit{Var}[X]\,c$
3. $\mathit{Var}[\sum_{i\in\mathcal{I}} X_i] = \sum_{i,j\in\mathcal{I}} \mathit{Cov}(X_i, X_j)$.

Corollary 2.4.9. *Let $(X_i)_{i\in\mathcal{I}}$ be a sequence of random vectors and $(A_i)_{i\in\mathcal{I}}$ be a sequence of deterministic matrices. We have*

$$\mathit{Var}[\sum_{i\in\mathcal{I}} A_i\,X_i] = \sum_{i,j\in\mathcal{I}} A_i\,\mathit{Cov}(X_i, X_j)\,A_j^\top.$$

Covariance matrices belong to the class of positive (semi-)definite and symmetric matrices.

Properties germane to spectra and positive (semi-) definite matrices

Definition 2.4.10 (Positive Operator)**.** *A linear operator $A : \mathcal{X} \to \mathcal{X}$ is called non-negative definite or positive semi-definite, PSD for short, if $\forall x : \langle x, Ax\rangle_{\mathcal{X}} \geq 0$. We write $A \geq 0$ for PSD A. It is (strictly) positive definite, PD for short, if $\forall x : \langle x, Ax\rangle_{\mathcal{X}} > 0$. We write $A > 0$ for PD A.*

Remark 2.4.11 ($\succ$). For PSD (PD) A we write $A \geq 0$ ($A > 0$). To make our life easier in cases where a property applies to both PD and PSD operators, we use $\succ$ as a placeholder for either $>$ or $\geq$. For operators A, B we define $B \succ A$ to mean $B - A \succ 0$.

Theorem 2.4.12. *Real-valued matrix $A \succ 0$ iff A is symmetric and $\sigma(A) \subset \mathbb{R}_+$.*

Matrix norms and geometric series of matrices

We define $\mathcal{M}_d := \mathbb{R}^{d\times d}$ to be the vector space of $d \times d$ matrices with real entries.

Definition 2.4.13 (Matrix norm)**.** *$|||\cdot||| : \mathcal{M}_d \to \mathbb{R}_{\geq 0}$ is a matrix norm, if it is a vector-space norm on $\mathcal{M}_d$ and, in addtion, is sub-multiplicative. That is, $|||A\,B||| \leq |||A|||\,|||B|||$.*

Example 2.4.14 (Spectral and Frobenius norms). An example is the Frobenius norm: $|||A|||_F := \sqrt{\mathrm{tr}(A^*A)}$. Another is the matrix norm induced by the vector space norm $\|\cdot\|$ as follows: $|||A|||_{\text{ind}} = \max_{\|x\|=1} \|Ax\|$. A special matrix norm is the one induced by the Euclidean vector norm:

$$|||A|||_2 = \sqrt{\rho(A^*A)}$$

where $\rho(A^*A)$ is maximum modulus of eigenvalues of pos. semi-def. matrix A^*A (i.e. its largest singular value). This norm is called the *spectral norm* or Schatten$-\infty$ norm. All these matrix norms are sub-multiplicative. The spectral norm assumes a special position in that it is minimal, in the sense that $|||A|||_2 \leq |||A|||$ for any sub-multiplicative matrix norm $|||\cdot|||$. Moreover, if A is pos. semi-definite then $\rho(A) = |||A|||_2$.

Definition 2.4.15 (Spectral radius)**.** *Let $M \in \mathcal{M}_d$ be a matrix (real or complex) with eigenspectrum $\lambda_1, \ldots, \lambda_d$. Its spectral radius is defined as $\rho(M) = \max_i |\lambda_i|$ being the maximum of the moduli values of the eigenvalues.*

Lemma 2.4.16. *Let $\rho(M)$ be the spectral radius of matrix M, endowed with sub-multiplicative matrix norm $|||\cdot|||$. We have $\rho(M) \leq |||M|||$.*

Lemma 2.4.17. *For every $\epsilon > 0, M \in \mathcal{M}_d$ there exists a matrix norm $|||\cdot|||$ such that $\|M\| \leq \rho(M) + \epsilon$.*

Lemma 2.4.18. *Let $S \in \mathcal{M}_d$ be a symmetric matrix with eigenvalues $\lambda_1 \leq \ldots \leq \lambda_d$. We have: $\lambda_1 I_d \leq S \leq \lambda_d I_d$. In particular, we have*

$$S \leq \lambda_d I_d \leq |||S|||_2 I_d.$$

Proof. The proof follows the lecture notes "Notes on Symmetric Matrices", by Nick Harvey, 2011-2012.

(i) We show $S \leq \lambda_d I_d$: Clearly, $M := \lambda_d I_d - S$ is Hermitian. So, it remains to show that the smallest Eigenvalue of M is non-negative. We have $\det(M - \lambda I_d) = \det((\lambda_d - \lambda)I_d - S)$. Hence, the eigenvalues of matrix M are just $E := \{\nu_1, \ldots, \nu_d | \forall i : \nu_i = \lambda_d - \lambda_i\}$. Since $\lambda_d \geq \lambda_i, \forall i$, we have $\min E \geq 0$. (ii) $S \geq \lambda_1 I_d$ is shown completely analogously. □

Lemma 2.4.19 (Gelfand's formula)**.** *$\rho(M) = \lim_{k\to\infty} \left|||M^k|||\right|^{\frac{1}{k}}$ for any matrix norm $|||\cdot|||$.*

The formula allows us to use the spectral radius to make inferences over the long-term growth of powers of matrices. For instance, if $\rho(M) < 1$ we know that M^k will converge to the zero-matrix as $k \to \infty$. This fact is utilised in the proof of the following lemma:

Lemma 2.4.20 (Geometric Series of Matrices). *Let $P \in \mathbb{R}^{d\times d}$ be a square matrix with eigenvalues $\lambda_1, \ldots, \lambda_d$. If $\max_{i=1}^{d} |\lambda_i| < 1$ (equiv. to $\lim_{i\to\infty} P^i = 0$) then we have*

1. *$I - P$ is invertible*
2. *The geometric series satisfies the equality $\sum_{i=0}^{k} P^i = (I-P)^{-1}(I-P^{k+1})$*
3. *and it converges: $\sum_{i=0}^{k} P^i \overset{k\to\infty}{\longrightarrow} (I-P)^{-1}$*

The following theorem, which can be found in any introductory analysis textbook (e.g. [125]), is a criterion for determining whether a series convergences :

Lemma 2.4.21 (Cauchy root test). $|\sum_{k=0}^{\infty} a_k| \begin{cases} < \infty & , \text{ if } \limsup_{k\to\infty} |a_k|^{\frac{1}{k}} < 1 \\ = \infty & , \text{ if } \limsup_{k\to\infty} |a_k|^{\frac{1}{k}} > 1. \end{cases}$

2.4.4. Derivation of a concentration inequality based on the spectral norm of the covariance

Lemma 2.4.22 (Ferentinos' inequality, 1982). *Let $X = (X_1, \ldots, X_d)^\top$ be a d-dimensional random vector. Let $v := (var[X_1], \ldots, var[X_d])^\top$ be the vector of component variances and assume $\|v\|_\infty < \infty$. Let $\|\cdot\|_2$ denote the canonical Euclidean vector-space norm. For $r > 0$, we have $\Pr[\|X - \langle X\rangle\|_2 \geq r \|v\|_2] \leq \frac{1}{r^2}$.*

We can use this inequality to derive another concentration inequality based on the spectral norm of a random vector's variance matrix:

Theorem 2.4.23. *Let $X = (X_1, \ldots, X_d)^\top$ be a d-dimensional random vector with variance matrix $V := Var[X]$ with finite spectral norm $|||V|||_2 \in \mathbb{R}_+$. Let $\|\cdot\|_\infty, \|\cdot\|_2$ be the standard maximum and Euclidean vector-space norms, respectively. $s := (\sqrt{var[X_1]}, \ldots, \sqrt{var[X_d]})^\top$ be the vector of component standard deviations and assume $\|s\|_\infty < \infty$. Let $\|\cdot\|_2$ denote the canonical Euclidean vector-space norm. For $r > 0$, we have $\Pr[\|X - \langle X\rangle\|_2 \geq r] \leq \frac{d|||V|||_2}{r^2}$.*

Proof. Let $s_i := \sqrt{\text{var}[X_i]}, s = (s_1, \ldots, s_d)^\top$. For the canonical basis vectors $e_i = (\delta(j-i))_{j=1,\ldots,d}$ we have $\|e_i\| = 1$ $(i = 1, \ldots, d)$. Furthermore, $(Ve_i)_i = V_{ii} = \text{var}[X_i]$. Consequently, $|||V|||_2 = \max_{\|x\|_2=1} \|Vx\|_2 \geq \|Ve_i\|_2 \geq \|Ve_i\|_\infty \geq \text{var}[X_i] = s_i^2, \forall i$. Thus, $\sqrt{|||V|||_2} \geq \|s\|_\infty$. The well-known norm equivalence inequalities, $\|s\|_\infty \leq \|s\|_2 \leq \sqrt{d}\|s\|_\infty$, entail

$$\sqrt{d}\sqrt{|||V|||_2} \geq \sqrt{d}\|s\|_\infty \geq \|s\|_2 . \tag{2.21}$$

Let $q > 0$ and define $E_1 := \{x| \ \|x - \langle X\rangle\|_2 \geq q\|s\|_2\}$ and $E_2 := \{x| \ \|x - \langle X\rangle\|_2 \geq q\sqrt{d}\sqrt{|||V|||_2}\}$. Let $x \in E_2$ arbitrary. Hence, $\|x - \langle X\rangle\|_2 \geq s\sqrt{d}\sqrt{|||V|||_2} \geq q\|s\|_2$. Hence, $x \in E_1$. Since $x \in E_2$ was an arbitrary choice, we have shown $E_2 \subseteq E_1$. Thus, $\Pr[E_2] \leq \Pr[E_1] \leq \frac{1}{q^2}$ where the last inequality is due

to Ferentino's inequality. Setting $r := s\sqrt{d}\sqrt{|||V|||_2}$ we can rewrite $\Pr[E_2] \leq \frac{1}{s^2}$ as $\Pr[\|x - \langle X\rangle\|_2 \geq r] \leq \frac{d|||V|||_2}{r^2}$.

□

2.4.5. Random fields and stochastic processes

Let $(\Omega, \Sigma, \mathbb{P})$ be a probability space and let $\mathcal{I} \subseteq \mathbb{R}^d$ be an *index set*. A *random field (r.f.)* (over this probability space) is a $\mathbb{P}$-measurable sequence $X = \Big(X_t\Big)_{t\in\mathcal{I}}$ of random variables (r.v.s) or random vectors (r.vec.s) X_t.

A special case is that of a *stochastic process (s.p.)*. A stochastic process is a random field with one-dimensional index set. That is, $\mathcal{I} \subseteq \mathbb{R}$. One way of thinking about a s.p. is that it represents an uncertain or random temporal evolution of states $X_t(\omega)$. For this reason, the index t of a s.p. is often called "time". Owing to this interpretation (as well as to its greater simplicty), the theory of s.p.s is more mature than that of the more general random fields. Of course, an even simpler sub-case arises when the index set is finite. For instance, if $|\mathcal{I}| = d$, the resulting s.p. simply is a $d-$dimensional random vector.

In general, random fields where all finite subsets have a common joint distribution inherit the name of that distribution. An important example is a *Gaussian* or *normal* random field (*GRF*). Here, any finite number of random vectors of the r.f. are jointly Gaussian distributed. Owing to its importance in machine learning and its prominence in stochastic differential equations, we will examine the favourable analytic properties of this specific r.f. in geater detail in Sec. 2.6. For any outcome $\omega \in \Omega$, the *realization* of (or *draw* from) the r.f. can be represented as a function $f_\omega : \mathcal{I} \to \mathbb{R}^d, t \mapsto X_t(\omega)$. So, one way of thinking about a r.f. is as a "distribution over functions". The necessity to ensure the space of functions under consideration is measurable by some probability measure, restricts the class of functions on which there is a non-zero measure (i.e. which can be learned). However, as we will see for the case of GRFs, the function classes can still be very rich. When the function space interpretation is emphasised, we will sometimes move the index into the parentheses. That is, we may sometimes write $X(t;\omega)$ or simply $X(t)$ for $X_t(\omega)$ or X_t, respectively.

Remark 2.4.24 (Terminology in machine learning). It is to be pointed out that in machine learning, often no distinction is made between a r.f. and an s.p.. Perhaps most prominently in Bayesian nonparametric machine learning based on "Gaussian processes (GPs)", the latter typically refer to Gaussian random fields and processes alike. In machine learning, GRFs that are sequences of r.vecs. are often referred to as *multi-output GPs* (e.g. see [191]).

While our work was predominantly conceived from an artificial intelligence point of view, upholding the distinction between random fields and stochastic processes will prove conducive to clarity of exposition.

Definition 2.4.25 (Continuous version, [210])**.** *Let $\mathcal{I} = \mathbb{R}^d$, $X = (X_t)_{t\in\mathcal{I}}$ be a r.f. with random vectors mapping into separable Hilbert space $(\mathcal{H}, \mathfrak{d})$. We say X has a continuous version if there exists a $\mathcal{H}-$valued s.p. X' such that the the following conditions hold:*

- *Trajectory $t \mapsto X'_t(\omega)$ is continuous for almost all draws ω.*
- $\mathfrak{d}(X'_t(\omega), X_t(\omega)) = 0$*, almost surely for all* $t \in \mathcal{I}$.

Theorem 2.4.26 (Kolmogorov, [210])**.** *Let $\mathcal{I} = \mathbb{R}^d$. If $X = (X_t)_{t\in\mathcal{I}}$ is a r.f. with random vectors mapping into separable Hilbert space $(\mathcal{H}, \mathfrak{d})$, and if $\exists\alpha, C, \epsilon > 0 \forall t, s \in \mathcal{I} : \mathbb{E}\,\mathfrak{d}(X_s, X_t)^\alpha \leq C \|s - t\|^{d+\epsilon}$ then there exists a Hölder continuous version of X of order θ for each $\theta < \epsilon/\alpha$.*

Corollary 2.4.27 ([210])**.** *Let $\mathcal{I} = \mathbb{R}^d$ and $X = (X_t)_{t\in\mathcal{I}}$ be a zero-mean Gaussian process with covariance function $k(t, s) := Cov(X_t, X_s) = \langle X_t X_S\rangle$. A sufficient condition for the existence of a continuous version is that k should be locally Hölder continuous: for each $N \in \mathbb{N}$ there exists $\theta = \theta(N) > 0$, $C = C(N)$ such that for $\|t\|, \|s\| \leq N : \|k(s, t) - k(t, t)\| \leq C \|s - t\|^\theta$.*

Note that for a stationary GRF, the condition reduces to the Hölder continuity of k at 0 (which implies that k is Hölder continuous everywhere) [210].

2.4.6. The Ornstein-Uhlenbeck Process (OU Process)

A stochastic process which we will consider repeatedly is the Ornstein-Uhlenbeck process. It plays a prominent role in many fields such as physics, statistics, control and finance. In the thesis we use these processes as a conservative model of closed-loop dynamics in planning and multi-agent coordination.

In the one-dimensional case, an Ornstein-Uhlenbeck (OU) process is the stochastic solution trajectory of the IVP given by a linear Ito-differential equation:

$dZ(t) = K(\xi - Z(t))\,dt + \sigma\,\mathrm{d}W(t), Z(0) = X_0$. Harnessed with Ito's lemma and the standard product rule one is able to show [101] that the solution is $Z(T) = x_0 \exp\big(-KT\big) + \xi\big(1 - \exp(-KT)\big) + \int_0^T \sigma \exp\big(K(\tau - T)\big)$.

Assuming the initial condition X_0 is Gaussian distributed, it is easy to see that the OU process is a Gaussian Markov process.[3] Owing to Gaussianity, the s.p. is given by its mean and covariance function, which are given by:

$$\langle Z(T)\rangle = x_0 \exp\big(-KT\big) + \xi\big(1 - \exp(-KT)\big) \tag{2.22}$$

$$\mathrm{cov}(Z(t), Z(s)) = \frac{\sigma^2}{2K}\Big(\exp\big(-K\,|s - t|\big) - \exp\big(-K(s + t)\big)\Big) \tag{2.23}$$

and in particular, $\mathrm{var}[Z(t)] = \frac{\sigma^2}{2K}\Big(1 - \exp\big(-2Kt\big)\Big)$. As one can see from Eq. 2.23, $\lim_{T\to\infty} \mathrm{var}[Z(T)] = \frac{\sigma^2}{2K}$ where convergence is strictly isotone (i.e. mon. increasing). This means that the stationary variance $\frac{\sigma^2}{2K}$ exists and is an upper bound on the transient one.

[3] As a standard argument [101], one can rewrite the corresponding stochastic integral equation as a limit of partial sums (Euler approximations). Each sum then is a sum of independent Gaussians. Showing that the second-order limit converges completes the proof.

2.4.7. Tail bounds

Throughout this work, we make intensive use of concentration inequalities. Therefore, we review existing ones and derive a few variants that will become useful later.

Tail inequalities.

For a zero-mean random variable X and a non-negative function $h : \mathbb{R} \to [0, \infty)$ it is known [110] that $\Pr[h(X) \geq r] \leq \frac{\langle h(X) \rangle}{r} \forall r > 0$. This result entails famous concentration inequalities. For instance, Markov's inequality is the special case $h : x \mapsto |x|$ and Chebyshev's follows from setting $h : x \mapsto x^2$ [110]. The proof is very easy and we noted that it can be extended with ease to the multi-variate case. The result is encapsulated in the following theorem:

Theorem 2.4.28. *Let $X = (X_1, \ldots, X_d)$ be a zero-mean random vector on a probability space with probability measure $\mathbb{P}$. Let $\phi : \mathbb{R}^d \to [0, \infty)$ be a non-negative, $\mathbb{P}-$measurable function. Then*

$$\Pr[\phi(X) > r] \leq \frac{\langle \phi(X) \rangle}{r}, \ \forall r > 0.$$

Proof. Let $E := \{x | \phi(x) > r\}$ be the event of function ϕ being above threshold $r > 0$. We have the point-wise inequality $0 \leq r\mathbf{1}_E(X) \leq \phi(X)$. Hence, $\Pr[\phi(X) > r] = \Pr[E] = \int_{\mathbb{R}^d} \mathbf{1}_E(x) \, \mathrm{d}\mathbb{P}(x) \leq \frac{1}{r} \int_{\mathbb{R}^d} \phi(x) \, \mathrm{d}\mathbb{P}(x) = \frac{1}{r} \langle \phi(X) \rangle$. □

Analogously to the one-dimensional case, the theorem allows us to derive a number of (existing and new) concentration bounds for random vectors.

For instance, *Ferentinos' inequality* states that $\Pr[\|X - \langle X \rangle\|_2 \geq r] \leq \frac{\|s\|_2^2}{r^2}$, $(r > 0)$ where $s_i := \sqrt{\mathrm{var}[X_i]}$ is a vector of standard deviations. We see this statement is a consequence of Thm. 2.4.28, by setting $\phi(\cdot) := \|\cdot\|_2, \bar{X} := X - \langle X \rangle$. Then $\Pr[\|\bar{X}\|_2 > r] = \Pr[\|\bar{X}\|_2^2 > r^2] \leq \frac{1}{r^2} \langle \|\bar{X}\|_2^2 \rangle = \frac{1}{r^2} \|s\|_2^2$.

Due to the simplicity of the proof of Thm. 2.4.28, we believe it can be extended even to continuously indexed stochastic processes (for instance GPs). Since elaboration of this is beyond the scope of this work, we state this as a mere conjecture:

Conjecture 2.4.29. *Let $X = (X_t)_{t \in \mathcal{I}}$ be a zero-mean stochastic process on a probability space over a Hilbert space $\mathcal{H}$ endowed with probability measure $\mathbb{P} : \mathcal{H} \to [0, 1]$. Let $\phi : \mathcal{H} \to [0, \infty)$ be a non-negative, functional that is $\mathbb{P}-$measurable. Then*

$$\Pr[\phi(X) > r] \leq \frac{\langle \phi(X) \rangle}{r}, \ \forall r > 0.$$

Proof. (Proof sketch). Let $E := \{x | \phi(x) > r\}$ be the event of function ϕ being above threshold $r > 0$. We have the point-wise inequality $0 \leq r\mathbf{1}_E(X) \leq \phi(X)$. Hence, $\Pr[\phi(X) > r] = \Pr[E] = \int_{\mathbb{R}^d} \mathbf{1}_E(x) \, \mathrm{d}\mathbb{P}(x) \leq \frac{1}{r} \int_{\mathbb{R}^d} \phi(x) \, \mathrm{d}\mathbb{P}(x) = \frac{1}{r} \langle \phi(X) \rangle$. □

The (Sub-) Gaussian case. The distribution-independent inequalities have the virtue of applying equally to all classes of random vectors, regardless of their distribution. The benefit of generality of course comes at the expense of conservatism. That is, if more about the distribution of the given random deviates is known, often tighter bounds can be obtained. One such case arises in the case of sub-Gaussian processes.

Let $\tau \in \mathbb{R}_{\geq 0}$. A r.vec. X is called *τ-sub-Gaussian* if

$$\langle \exp(tX) \rangle \leq \exp\Big(\frac{b^2 t^2}{2}\Big), \forall b \geq \tau.$$

Number τ is called the *standard* of X. It can be shown that the standard is an upper bound of the standard deviation of a sub-Gaussian and that every sub-Gaussian is zero-mean [207].

Remark 2.4.30 (Gaussians and a.s. bounded r.v.). Trivial examples sub-Gaussians are Gaussians as well as r.vec.s with bounded support. A centred Gaussian r.v. with standard deviation σ has a density with Laplace transform $\langle \exp(tX) \rangle = \exp\Big(\frac{\sigma^2 t^2}{2}\Big)$. Hence, a Gaussian is a sub-Gaussian with standard σ. Following a simple derivation [207] one can show that a centred r.v. X with a.s. bounded absolute value $|X| \leq b$ is sub-Gaussian with standard b.

Sub-Gaussians have tail bounds that do not exceed those of Gaussians. For instance, as recently shown by [45], we have the following tail inequality:

Theorem 2.4.31. *Let $\mathcal{I} \subset N$, $X = \Big(X_k\Big)_{k \in \mathcal{I}}$ be a sequence of centred sub-Gaussian r.v.s. Let $\tau(X_k)$ be the standard of sub-Gaussian X_k. For $p \in [1, \infty]$, let $\|X\|_p$ denote the standard p-norm on the process X. Assume $B(X) := \sum_{k \in \mathcal{I}} \tau(X_k) < \infty$. Then we have:*

$$\Pr[\|X\|_p \geq r] \leq 2 \exp\Big(-\frac{r^2}{2B^2(X)}\Big). \tag{2.24}$$

As one can see from the statement, sub-Gaussian random vectors have tails that decrease exponentially with r^2. Needless to say, this can be a significant improvement over the polynomial decrease with r afforded by the distribution-independent bounds considered above.

Theorem 2.4.32 (taken from Boucheron et. al [41])**.** *Let X be a d-dimensional random vector of d independent standard normal random variables. Let $f : \mathbb{R}^d \to \mathbb{R}$ denote a $L-$Lipschitz function.*

Then, for all $r > 0$,

$$\Pr[f(X) - \langle f(X) \rangle \geq r] \leq \exp(-\frac{r^2}{2L^2}).$$

The theorem implies that the image of Gaussian random vector mapped through a Lipschitz function is sub-Gaussian.

We can leverage the theorem to derive a tail bound for a normally distributed random vector as follows:

Corollary 2.4.33. *Let X be a Gaussian r.vec. in d-dimensional space, with mean vector $\langle X \rangle$ and covariance matrix $C = Var[X]$. For $r > 0$, we have:*

$$\Pr[\|X - \langle X \rangle\|_2 \geq r] \leq 2 \exp\Big(-\frac{r^2}{2\rho(C)}\Big)$$

where $\rho(C)$ is the spectral radius of the covariance matrix. Note, since $C \geq 0$, we have $|||C|||_2 = \rho(C)$.

Proof. Let R be a cholesky factor of covariance matrix C. That is, $C = R^\top R$. Let $f : v \mapsto Rv$ be a function on $\mathbb{R}^d$. It is well-known that one can represent X as $X = f(Y)$ where Y is a r.vec. with components drawn i.i.d. from a standard normal distribution – $Y_i \sim \mathcal{N}(0, 1)$. Assuming that f is Lipschitz with Lipschitz constant $L_f = \sqrt{\rho(C)}$, we can apply Thm. 2.4.32 to deduce the claim of the corollary. The former assumption can be seen as follows: $\forall x, y \in \mathbb{R}^d : \|f(x) - f(y)\|_2 = \|R(x - y)\|_2 \leq |||R|||_2 \, \|x - y\|_2$ where the last inequality is a general norm inequality valid for all bounded operators. This tells us that $|||R|||_2$ is a Lipschitz constant. By definition, the spectral norm of $|||R|||_2$ is just the square root of the spectral radius of $R^\top R = C$.

□

Note, we could have also derived a bound via Thm. 2.4.31. However, this would have yielded a bound involving the Frobenius norm rather than the spectral norm and hence, resulted in a looser inequality than the one established in Cor. 2.4.33.

A tail inequality that is very commonly used in computer science is Hoeffding's inequality, which exists in several versions. To us, the one of interest is the following:

Theorem 2.4.34 (Hoeffding's inequality)**.** *Let $X_1, ..., X_n$ be independent r.v. with $X_i \in [a_i, b_i]$, $(i = 1, ..., n)$ and let $\bar{X} = \frac{1}{n} \sum_{i=1}^n X_i$ denote their sample mean. Then for $t > 0$, we have* $\Pr[\bar{X} - \langle X \rangle > t] \leq \exp\Big(-2n^2t^2 / \sum_{i=1}^n (b_i - a_i)^2\Big)$.

Head and tail radii

Let $\delta \in (0, 1)$. Let $\|\cdot\|$ be a norm and $\mathfrak{B}_\varrho\big(\langle x \rangle\big) = \{x | \, \|x - \langle x \rangle\| \leq \varrho\}$ be a ball with respect to that norm with radius ϱ centred at $\langle x \rangle$. Throughout the thesis, we will frequently come across the problem of determining a radius ϱ such that the ball carries probability mass of at least $1 - \delta$. We will call it either the $1 - \delta$- *head radius* or δ-*tail radius* around the mean.

We will briefly introduce tail radii both for the Gaussian and the distribution-independent case.

In the case of r.v X being *Gaussian*, Cor. 2.4.33 states that $\Pr[\|X - \langle X \rangle\|_2 \geq \varrho] \leq 2 \exp\Big(-\frac{\varrho^2}{2|||\text{Var}[X]|||_2}\Big)$ where $\text{Var}[X]$ denotes the variance-covariance matrix. Setting the right -hand side less or equal to δ and solving for ϱ shows that choosing

$$\varrho = \sqrt{2|||\text{Var}[X]|||_2 \log\big(\frac{2}{\delta}\big)} \tag{2.25}$$

implies that ϱ is a valid δ-tail radius with respect to the Euclidean norm.

2.4.8. Some properties of linear stochastic recurrences

In this section, we will establish basic properties of difference equations with drift vector fields drawn from a stochastic process.

We assume a discrete-time dynamic system defined by the recurrence relation:

$$X_{k+1} = M_k\, X_k + \Delta\, F_k + \Delta\, \mu_k \tag{2.26}$$

Here, $\Delta \in \mathbb{R}_+$ can be interpreted as a time-increment, $M_k \in \mathbb{R}^{d\times d}$ is a *transition matrix* at time step k, each $\mu_k \in \mathbb{R}^d$ is a deterministic vector and could represent an open-loop control input acting on the last m components. Increments process $F := \Big(F_k\Big)_{k\in\mathbb{N}_0}$ is assumed to be a centred (i.e. zero-mean) vector-valued stochastic process. We do not make any further assumptions on F. In particular, we do not assume that F is uncorrelated.

Let I_d denote the d-dimensional identity matrix.

For notational convenience, we define the matrix products

$$P_{j,i} = \begin{cases} M_j\, M_{j-1}\, \ldots M_i & , i \leq j \\ I_d, & , i > j \end{cases} \tag{2.27}$$

In the following lemma, we solve the recurrence relation. Of particular interest will be the first two central moments of the process.

Lemma 2.4.35. *Assume $X = \Big(X_k\Big)_{k\in\mathbb{N}_0}$ adheres to the recurrence given in Eq. 2.26. Let Var$[X_k] \in \mathbb{R}^{d\times d}$ denote the variance-covariance matrix (following Feller's notation) and let $\langle X_k\rangle \in \mathbb{R}^d$ denote the expected*

value of random vector X_k. For all $k \in \mathbb{N}$, we have:

$$X_k = M_{k-1}X_{k-1} + \Delta F_{k-1} + \Delta\mu_{k-1} = \ldots = P_{k-1,0}X_0 + \Delta\sum_{i=0}^{k-1} P_{k-1,i+1}(F_i + \mu_i) \tag{2.28}$$

$$\langle X_k\rangle = P_{k-1,0}\langle X_0\rangle + \Delta\sum_{i=0}^{k-1} P_{k-1,i+1}\,\mu_i, \tag{2.29}$$

$$\mathit{Var}[X_k] = \Delta^2\sum_{i,j=0}^{k-1} P_{k-1,i+1}\mathit{Cov}(F_i,F_j)\,P_{k-1,j+1}^\top \tag{2.30}$$

$$+\Delta\sum_{i=0}^{k-1} P_{k-1,0}\mathit{Cov}(X_0,F_i)P_{k-1,i+1}^\top + P_{k-1,i+1}\mathit{Cov}(F_i,X_0)P_{k-1,0}^\top \tag{2.31}$$

$$\mathit{Var}[X_{k+1}] = M_k\,\mathit{Var}[X_k]\,M_k^\top + \Delta^2\mathit{Var}[F_k] + \Delta M_k\mathit{Cov}(X_k,F_k) + \Delta\mathit{Cov}(F_k,X_k)\,M_k^\top \tag{2.32}$$

$$= M_k\,\mathit{Var}[X_k]\,M_k^\top + \Delta^2\mathit{Var}[F_k] + \Delta P_{k,0}\mathit{Cov}(X_0,F_k) + \Delta\mathit{Cov}(F_k,X_0)P_{k,0}^\top \tag{2.33}$$

$$+\Delta^2\sum_{i=0}^{k-1} P_{k,i+1}\mathit{Cov}(F_i,F_k) + \mathit{Cov}(F_k,F_i)P_{k,i+1}^\top. \tag{2.34}$$

Proof. (i) We show Eq. 2.28, by induction.

Base case: k=1: follows directly from Eq. 2.26:

$X_1 = P_{0,0}X_0 + \Delta\sum_{i=0}^{0} P_{0,1}(F_i+\mu_i) = M_0X_0 + \Delta(F_i+\mu_i)$.

I.H.: $\exists k \geq 1$ such that $X_k = P_{k-1,0}X_0 + \Delta\sum_{i=0}^{k-1} P_{k-1,i+1}(F_i+\mu_i)$.

I.S. ($k \to k+1$):

$X_{k+1} = M_kX_k + \Delta(F_k+\mu_k)$

$\overset{I.H.}{=} M_k\big(P_{k-1,0}X_0 + \Delta\sum_{i=0}^{k-1} P_{k-1,i+1}(F_i+\mu_i)\big) + I_d\Delta(F_k+\mu_k)$

$\overset{P_{k,k+1}=I_d}{=} \big(P_{k,0}X_0 + \Delta\sum_{i=0}^{k-1} P_{k,i+1}(F_i+\mu_i)\big) + P_{k,k+1}\Delta(F_k+\mu_k)$

$= P_{k,0}X_0 + \Delta\sum_{i=0}^{k} P_{k,i+1}(F_i+\mu_i)$.

(ii) Eq. 2.29 is obtained from Eq. 2.28 and leveraging the linearity of the expectation operator.

(iii) Eq. 2.32 is obtained, by taking the variance operator on both sides of Eq. 2.26 and applying Cor. 2.4.9.

(iv) Eq. 2.31 follows from leveraging in in Eq. 2.28 that a covariance is a bi-linear form (cf. Lem. 2.4.8).

(v) Finally, we show Eq. 2.34: Substituting Eq. 2.28 into $\mathrm{Cov}(X_k,F_k)$ and application of Lem. 2.4.8 and 2.4.7 allows the conclusion: $\mathrm{Cov}(X_k,F_k) = \mathrm{Cov}(P_{k-1,0}X_0 + \Delta\sum_{i=0}^{k-1} P_{k-1,i+1}(F_i+\mu_i), F_k) \overset{\mu_i \text{ determ.}}{=} \mathrm{Cov}(P_{k-1,0}X_0 + \Delta\sum_{i=0}^{k-1} P_{k-1,i+1}F_i, F_k) = P_{k-1,0}\mathrm{Cov}(X_0,F_k) + \Delta\sum_{i=0}^{k-1} P_{k-1,i+1}\mathrm{Cov}(F_i,F_k)$. Substitution into Eq. 2.32 and acknowledging that $M_kP_{k-1,i+1} = P_{k,i+1}$ and $\mathrm{Cov}(X_k,F_k)^\top = \mathrm{Cov}(F_k,X_k)$ yields the desired result. □

2.4.9. Martingale techniques

Martingales, sub- and super-Martingales are stochastic processes for which powerful convergence properties have been developed. Therefore, if a control can be found that converts the closed-loop dynamics into

a stochastic process exhibiting martingale-type properties, the tools afforded by martingale theory can be applied. Without further ado, we will briefly review the definitions and theorems utilised throughout this work.

Definition 2.4.36. *Let $\mathcal{I}$ be a totally ordered index set and $(\Omega, \Sigma, \mathbb{P})$ be a probability space. Let $\mathcal{F} = (\mathcal{F}_t)_{t\in\mathcal{I}}$ be a filtration, $\mathcal{F}_s \subset \mathcal{F}_t \subset \Sigma, \forall s \leq t, s, t \in \mathcal{I}$. Let $(X_t)_{t\in\mathcal{I}}$ be an adapted stochastic process on a probability space such that for all $t \in \mathcal{I}$ we have: X_t is measurable with respect to $\mathcal{F}_t$ and $\langle|X_t|\rangle < \infty$. The pair $(X, \mathcal{F})$ is*

- *a martingale, iff $\langle X_t|\mathcal{F}_s\rangle = X_s, \forall s < t$,*
- *a sub-martingale, iff $\langle X_t|\mathcal{F}_s\rangle \geq X_s, \forall s < t$,*
- *a super-martingale, iff $\langle X_t|\mathcal{F}_s\rangle \leq X_s, \forall s < t$.*

A reader unfamiliar with the notions of filtrations may think of $\mathcal{F}_s$ as the information available up until time s. The requirement $\mathcal{F}_s \subset \mathcal{F}_t\,(t \geq s)$ essentially means that information does not vanish over time. Often, it suffices to read $\langle X_t|\mathcal{F}_t\rangle$ as the conditional expectation $\langle X_t|\{X_s : s \in \mathcal{I}, s < t\}\rangle$ conditioned on the realisation of the process before the current index t. If the filtration contains some other sequence $(Y_t)_{t\in\mathcal{I}_t}$ of r.v., X is called a martingale with respect to Y. The statement that X_t is *adapted* formally means that it is $\mathcal{F}_t$-measurable with respect to the underlying probability measure. It is well-known that this essentially means that the outcome of X_t is non-anticipating. That is, informally speaking, it is a function of the content of $\mathcal{F}_t$, but does not dependent upon future outcomes. In other words, $X_t(\omega)$ must be known at time t. Furthermore, the definition allows a s.p. to be a martingale with respect to one probability measure but not to be a martingale with respect to another.

Note, in case of discretely indexed processes, the martingale condition often is translated to $\langle X_{k+1} \mid X_k, ..., X_0\rangle = X_k$. This is a special case of our more general definition which encompasses both discrete-time and continuous-time cases as well as richer information sets.

A discrete-time process $\big((C_k)_{k\in\mathcal{I}}, \mathcal{F}\big)$ with respect to a filtration $\mathcal{F}$ is called predictable if C_k is $\mathcal{F}_{k-1}$-measurable. In particular, deterministic sequences are predictable processes. [110].

As first demonstrated by Doob for discrete index sets [85] and later by Meyer for the continuous case [168, 169], (sub-) super-martingales can be uniquely represented as the sum of a martingale and a so-called *compensator*, being a predictable (isotone) antitone process. The decomposition is called Doob-decomposition and is the subject of the following theorem.

Theorem 2.4.37 (Doob decomposition [85, 110])**.** *Let $\mathcal{I} = \mathbb{N}_0$. Let $\big((X_k)_{k\in\mathcal{I}}, \mathcal{F}\big)$ be a sub-martingale with respect to filtration $\mathcal{F} = (\mathcal{F}_k)_{k\in\mathcal{I}}$. There exists a unique martingale $\Big(M_k\Big)_{k\in\mathcal{I}}$ and a unique predictable isotone process $\Big(C_k\Big)_{k\in\mathcal{I}}$ such that*

$$X_k = M_k + C_k, \forall k \in \mathcal{I}$$

with $M_0 = X_0$ and $C_0 = 0$. The constituent processes are given by :

$$M_k = X_0 + \sum_{i=1}^{k} \big(X_i - \langle X_i | \mathcal{F}_{i-1} \rangle\big) \tag{2.35}$$

$$C_k = \sum_{i=1}^{k} \big(\langle X_i | \mathcal{F}_{i-1} \rangle - X_{i-1}\big). \tag{2.36}$$

If we have a super-martingale rather than a sub-martingale, we can apply the Doob decomposition theorem to $-X_k$ (which is a sub-martingale).

Since we will be concerned with bounding the probability of the state trajectory leaving a predefined corridor, the following two theorems inequalities will be of greatest interest.

The first one pertains to discrete-time processes:

Theorem 2.4.38. *Let $n \in \mathbb{N}$, $\mathcal{I} = \{1, \ldots, n\}$ and $(X_k)_{k \in \mathcal{I}}$ be a sub-martingale.*

$$\forall r \geq 0 : \Pr[\max\{X_1, \ldots, X_n\} \geq r] \leq \frac{\langle \max\{0, X_n\} \rangle}{r}.$$

Proof. See [21]. □

The next theorem provides Markov-type inequalities for martingales which are often referred to as *Doob's $\mathcal{L}_p$- inequalities*. The theorem is applicable to continuously-indexed processes. However, with the restriction to martingales that are a.s. right-continuous.

Theorem 2.4.39. *Let $T > 0, \mathcal{I} = [0, T]$, $X = (X_t)_{t \in \mathcal{I}}$ and $\big(X, \mathcal{F}\big)$ a martingale with a.s. right-continuous sample paths. For all $r > 0, p \in [1, \infty)$, we have:*

$$\Pr[\sup_{t \in \mathcal{I}} |X_t| \geq r] \leq \frac{\langle |X_T|^p \rangle}{r^p}, \tag{2.37}$$

$$\langle \big(\sup_{t \in \mathcal{I}} |X_t|\big)^p \rangle \leq \big(\frac{p}{p-1}\big)^p \langle |X_T|^p \rangle. \tag{2.38}$$

Proof. See [21]. □

Gilat [105] has shown that every nonnegative, right-continuous sub-martingale is the absolute value of a right-continuous martingale. Combining this insight with Thm. 2.4.39 yields the following result:

Corollary 2.4.40. *Let $T > 0, \mathcal{I}_t = [0, T]$ and $(S_t)_{t \in \mathcal{I}}$ a nonnegative sub-martingale with a.s. right-continuous sample paths. For all $r \geq 0, p \in [1, \infty)$ we have:*

$$\Pr[\sup_{t \in \mathcal{I}_t} S_t \geq r\,] \leq \frac{\langle S_T^p \rangle}{r^p}.$$

2.4.10. Maximal tail inequalities of stochastic recurrences based on martingale properties

Planning and control under uncertainty can often be done on the basis of the nominal model given by the expected trajectory. In the presence of stochastic uncertainty, stochastic guarantees will then bound the belief of deviation from the nominal model. Often, deviation bounds satisfied in the point-wise sense will be sufficient. That is, bounds that hold for any given r.v. in the uncertain trajectory. However, stronger guarantees would impose constraints on the deviation from the nominal at any given point in time within a bounded set of indices. For instance, in collision avoidance, one may wish to bound the collision probability within a time interval. This can be addressed with maximal inequalities. For an important sub-case of open-loop control where the uncertainty stems from an o.i. process, we will now derive such bounds on the basis of the martingale properties listed above. It would be surprising if the results we derive in what is to follow were new, but not having found them in the literature, we have to give our own proofs.

Discrete time

Let $X = (X_k)_{k\in\mathcal{I}}$ be a vector-valued process given by the uncertain recurrence relation:

$$X_{k+1} - X_k = \Delta\, F_k + \Delta\, \mu_k \tag{2.39}$$

$$X_0 = x_0. \tag{2.40}$$

Here, x_0 is an initial condition, μ_k some deterministic increment, $\Delta > 0$ and F_k is a centred (i.e. zero-mean) stochastic increment. We assume F_k is defined so that $k \mapsto F_k$ is an orthogonal increments process. This would be the case for instance if we chose $F_k = \frac{1}{\Delta} B(k)\, \mathrm{d}W(\Delta)$ where $\mathrm{d}W(\Delta)$ is a Wiener increment of length Δ and B is a non-anticipating function [101].

This type of dynamics might arise in model predictive control where μ_k is chosen via optimisation and F encodes the uncertainty around the nominal trajectory that is directed via the control inputs (cf. Sec. 2.2).

We will now derive a maximal inequity that only depends on the variance of the last time step of the bounded index set. The idea is to show that $S = \left(S_k\right)_{k\in\mathbb{N}}$ (with $S_k = \|X_k - \langle X_k\rangle\|\,, \forall k \in \mathbb{N}$) is a sub-martingale and then to apply the maximal inequalities for sub-martingales listed in Sec. 2.4.9. This approach results in the following statement:

Theorem 2.4.41. *Let $X = \left(X_k\right)_{k\in\mathbb{N}}$ be the solution process of our uncertain discrete-time initial value problem as per Eq. 2.39, $\mathcal{I}_n = \{0, \ldots, n\}$. Let $r > 0$, $p \geq 1$ and $\|\cdot\|$ a suitable norm. For any $p \geq 1, r \geq 0$ and norm $\|\cdot\|$ we have*

$$\Pr[\max_{k\in\mathcal{I}_n} \|X_k - \langle X_k\rangle\| \geq r] \leq \frac{\langle \|X_n - \langle X_n\rangle\|^p \rangle}{r^p}.$$

In particular, we have

$$\Pr[\max_{k\in\mathcal{I}_n} \|X_k - \langle X_k\rangle\|_2 \geq r] \leq \frac{tr\big(Var[X_n]\big)}{r^2}.$$

Proof. We define a filtration $(\mathcal{F}_k)$ and assume $\mathcal{F}_k$ to contain the initial condition X_0 as well as the history of all increments $F_0, ..., F_{k-1}$ representing all the information (we assume the (μ_k) is know a priori anyway) that is necessary to calculate all states up to and including time k. Hence, conditioning on $\mathcal{F}_k$ is at least as informative as conditioning on $\phi(X_k)$, for any known deterministic function ϕ.

(i) Let $\bar{X} = \left(\bar{X}_k\right)_{k\in\mathcal{I}_t} = X - \langle X\rangle$ be the centred version of our original process X. We show that (each component of) $\bar{X}$ is a martingale : Inspecting Eq. 2.39, it is easy to prove by induction that $\bar{X}_k = \Delta\sum_{i=0}^{k-1} F_i$. Thus, $\forall j \geq 1 : \langle \bar{X}_{k+j} \,|\, \mathcal{F}_k\rangle = \langle \Delta\sum_{i=0}^{j-1} F_{k+i} + \Delta\sum_{i=0}^{k-1} F_i \,|\, \mathcal{F}_k\rangle = \langle \Delta\sum_{i=0}^{j-1} F_{k+i} \,|\, \mathcal{F}_k\rangle + \langle \bar{X}_k \,|\, \mathcal{F}_k\rangle = \bar{X}_k$ since $\langle \Delta\sum_{i=0}^{j-1} F_{k+i} \,|\, \mathcal{F}_k\rangle = 0$ due to orthogonality and the zero-mean assumption.

(ii) Define $S_k := \left\|\bar{X}_k\right\|^p, \forall k \in \mathcal{I}_n$. Obviously $S = (S_k)$ is a non-negative process. We will now show it also is a sub-martingale: Since $p \geq 1$, $\|\cdot\|^p$ is a convex function. Hence, we can apply Jensen's inequality[4] to infer: $\langle S_{k+1} \,|\, \mathcal{F}_k\rangle = \langle \left\|\bar{X}_{k+1}\right\|^p \,|\, \mathcal{F}_k\rangle \geq \left\|\langle \bar{X}_{k+1} \,|\, \mathcal{F}_k\rangle\right\|^p \overset{(i)}{=} \left\|\bar{X}_k\right\|^p = S_k$.

(iii) We have shown that $(S, \mathcal{F}_k)$ is a non-negative sub-martingale. This allows us to apply Thm. 2.4.38 as follows: Let $n \in \mathbb{N}$, $\mathcal{I} = \{1, \ldots, n\}$ and $(X_k)_{k\in\mathcal{I}}$ be a sub-martingale. $\Pr[\max_{i\in\mathcal{I}_n} S_i \geq r^p] \leq \frac{\langle S_n\rangle}{r^p}$. Resubstitution of the definition of S_i yields: $\frac{\langle \|X_n - \langle X_n\rangle\|^p\rangle}{r^p} \geq \Pr[\max_{i\in\mathcal{I}_n} \|X_i - \langle X_i\rangle\|^p \geq r^p] = \Pr[\max_{i\in\mathcal{I}_n} \|X_i - \langle X_i\rangle\| \geq r]$.

(iv) The corollary statement follows when setting $p = 2$ and choosing the Euclidean norm. Then, $\frac{\langle \|X_n - \langle X_n\rangle\|_2^2\rangle}{r^2} = \frac{\sum_{i=1}^d \text{var}[X_{n,i}]}{r^2} = \frac{\text{tr}\left(\text{Var}[X_n]\right)}{r^2}$ where $X_{n,i}$ denotes the ith component of random vector X_n. □

The theorem provides us with a way of bounding the probability of deviating from the expected trajectory by a given magnitude based on the variances of the last random vector within a given time horizon. In comparison to bounds one could derive on the basis of the union bound, the bound we have derived only depends on the variance of the final random variable. Therefore, in situations where the theorem applies, the bounds given by Thm. 2.4.41 will be much tighter. However, it should be noted that this does not mean the variance is bounded. Depending on the nature of F, as the size of the index set increases, so may the variance bound.

Continuous time

Next, we will translate the result into processes with continuous index sets. The proofs are entirely analogous but are given for completeness. Let $\mathcal{I}_t \subset \mathbb{R}_+$ be a bounded set of ordered time indices and $T := \sup\{t \in \mathcal{I}_t\}$.

[4] We apply the multi-variate, probabilistic version for conditional expectations.

Define X to be the r.f. that solves the stochastic initial value problem:

$$\mathrm{d}X(t) = \mu(t)\,dt\, + g(t)\,\mathrm{d}W(t) \tag{2.41}$$

$$X(0) = X_0 \tag{2.42}$$

where μ is deterministic, g is a square-integrable, non-anticipating, continuous s.p. and $W(\cdot)$ denotes the Wiener process. Note, the stochastic IVP is solved by the process $X(\cdot)$ with

$$X(t) = X_0 + \int_0^t \mu(\tau)\,\mathrm{d}\tau + \int_0^t g(\tau)\,\mathrm{d}W(\tau) \tag{2.43}$$

where the stochastic integral $\int_0^t g(\tau)\,\mathrm{d}W(\tau)$ is interpreted in the sense of Ito (cf. [101, 110]).

We will now bound the probability of the maximal deviation from the norm. The idea is to show that $S = \Big(S_t\Big)_{t\in\mathcal{I}}$ (with $S_k = \|X_t - \langle X_t\rangle\|\,, \forall t \in \mathbb{N}$) is a sub-martingale and then to apply the maximal inequalities for sub-martingales listed above. This approach results in the following statement:

Theorem 2.4.42. *Let $X(\cdot)$ be defined as per Eq. 2.43. We have*

For any $p \geq 1, r > 0$ and norm $\|\cdot\|$ we have

$$\Pr[\max_{k\in\mathcal{I}_t} \|X(t) - \langle X(t)\rangle\| \geq r] \leq \frac{\langle\|X(T) - \langle X(T)\rangle\|^p\rangle}{r^p}.$$

In particular, we have

$$\Pr[\max_{k\in\mathcal{I}_t} \|X(t) - \langle X(t)\rangle\|_2 \geq r] \leq \frac{tr\big(Var[X(T)]\big)}{r^2}.$$

Proof. We define a filtration $(\mathcal{F}_s)$ to contain the history of all states and draws of $F(X(\tau)), (\tau \leq s)$ up to and including time s.

(i) It is well known [101] that for non-anticipating g, $\langle\int_0^t g(\tau)\,\mathrm{d}W(\tau)\rangle = 0$. Hence, $\langle X(t)\rangle = \langle X_0\rangle + \int_0^t \mu(\tau)\,\mathrm{d}\tau$. Hence, $\bar{X}(t) := X(t) - \langle X(t)\rangle = X_0 - \langle X_0\rangle + \int_0^t g(\tau)\,\mathrm{d}W(\tau)$. The last stochastic integral is well-known to be a martingale ([110], p. 543). Therefore, $\bar{X}$ is a martingale.

(ii) Define $S(t) := \big\|\bar{X}(t)\big\|^p, (t \in \mathcal{I}_t)$. Obviously $S = \big(S(t)\big)_{t\in\mathcal{I}_t}$ is a non-negative process. We will now show it also is a sub-martingale: Since $p \geq 1$, $\|\cdot\|^p$ is a convex function. Hence, we can apply Jensen's inequality[5] to infer: $\langle S(t)\,|\,\mathcal{F}_s\rangle = \langle\big\|\bar{X}(t)\big\|^p\,|\,\mathcal{F}_s\rangle \geq \big\|\langle\bar{X}(t)\,|\,\mathcal{F}_s\rangle\big\|^p \overset{(i)}{=} \big\|\bar{X}(s)\big\|^p = S(s)$.

(iii) We have shown that S is a non-negative sub-martingale. This allows us to apply Thm. 2.4.39 to S which entails: $\Pr[\max_{t\in\mathcal{I}_t} S(t) \geq r^p] \leq \frac{\langle S(t)\rangle}{r^p}$. Resubstitution of the definition of $S(t)$ yields: $\frac{\langle\|X(T)-\langle X(T)\rangle\|^p\rangle}{r^p} \geq \Pr[\max_{t\in\mathcal{I}_t} \|X(t) - \langle X(t)\rangle\|^p \geq r^p] = \Pr[\max_{t\in\mathcal{I}_t} \|X(t) - \langle X(t)\rangle\| \geq r]$. (iv) The corollary statement follows from setting $p = 2$ and choosing the Euclidean norm. Then, $\frac{\langle\|X(T)-\langle X(T)\rangle\|_2^2\rangle}{r^2} =$

[5] We apply the multi-variate, probabilistic version for conditional expectations.

$\frac{\sum_{i=1}^{d} \text{var}[X_i(T)]}{r^2} = \frac{\text{tr}\big(\text{Var}[X(T)]\big)}{r^2}$ where $X_i(T)$ denotes the ith component of random vector $X(T)$. □

The theorem provides a way of bounding the probability of deviating from the expected trajectory by a given magnitude, based on the variances of the last random vector within a given time horizon. In comparison to instantaneous boundedness guarantees that hold for any given index, the new bounds have the advantage of allowing us to bound the deviation within a continuous index set.

If the process models the dynamics of a physical plant, this property is highly advantageous. It allows us not only to bound the probability of reaching an infeasible part of state space at an arbitrary point in time, but also during a pre-defined time horizon of interest.

2.5. Hölder and Lipschitz continuity for inference

Hölder continuous functions are uniformly continuous functions that may exhibit infinitely many points of non-differentiability and yet are sufficiently regular to facilitate inference. That is, they have properties that make it possible to make assertions of global properties on the basis of a finite function sample.

Hölder continuity is a generalisation of Lipschitz continuity. Lipschitz properties are widely used in applied mathematics to establish error bounds and, among many other, find application in optimisation [124, 230] and quadrature [19, 73, 81] and are a key property to establish convergence properties of approximation rules in (stochastic) differential equations [101, 133]. Furthermore, most machine learning methods for function interpolation seem to impose smoothness (and thereby, Hölder continuity) on the function. For instance, with our Lem. 2.5.6 derived below, it would be possible to show that any finite *radial basis function neural network* [44] with a smooth basis function is Hölder continuous on a compact domain. Or, a *Gaussian process* with a smooth covariance function also has a smooth mean function and a.s. smooth sample paths (see Sec. 2.6, [110, 201]). Therefore, posterior inference over functions on compact support made with such a Gaussian process on the basis of a finite sample is Hölder continuous.

Throughout this thesis, we take advantage of a priori knowledge about Hölder continuity to perform inference. For instance, in Ch. 5, we leverage this regularity property to infer the absence of negative function values of a criterion function in the context of collision detection. In Ch. 4, Hölder continuity restricts the posterior hypothesis space of our *kinky inference (KI)* framework and facilitates learning.

Upon submission of this work, we have become aware of related work published in mathematical and operations research journals [24, 25, 69, 70, 275]. For instance, Zabinsky et. al. [275] consider the problem of estimating a one-dimensional Lipschitz function (with respect to the canonical norm-induced metric). Similar to the analysis we employ to establish our guarantees, they use a pair of bounding functions and make predictions by taking the average of these functions. While we have developed our kinky inference

rule independently, it can be seen as a generalisation of their approach. Our method provides extensions to Hölder continuous multi-output functions over general, multi-dimensional (pseudo-) metric spaces, can cope with with erroneous observations and inputs, can fold in additional knowledge about boundedness, learn parameters from data and provides different guarantees such as (uniform) convergence of the prediction uncertainty. As part of the analysis of our method (see Ch. 4 and Ch. C), we construct delimiting functions we refer to as *ceiling* and *floor* functions. The construction of similar functions is a recurring theme that, in the standard Lipschitz context, can be found in global optimisation [230], quadrature [19], interpolation [24, 25], as well as in the analysis of linear splines for function estimation [69]. Cooper [69, 70] utilises such upper and lower bound functions in a multi-dimensional setting to derive probabilistic PAC-type error bounds [252] for a linear interpolation rule. He assumes the data is sampled uniformly at random on a hypercube domain. This precludes the application of his results to much of our control applications where the data normally is collected along continuous trajectories visited by a controlled plant. Our inference rule is different from his and our guarantees do not rely on distributional assumptions. However, in future work, we would be interested in how far some of Cooper's statistical analysis techniques could be employed to translate our own guarantees to stochastic settings.

As a special case of our kinky inference framework, we consider *kNN-KI*. Here, the inference over function values is based on up to (approximate) k nearest neighbours aiming to reduce computational prediction effort to log-time. Approaches to this end include maintaining a data structure for fast nearest neighbour search [28] or utilising locally sensitive hash functions [213]. Strongly related to kNN-KI and the concept of maintaining a data structure to speed up predictions is the work of Beliakov [24] who derives an interpolation rule that is a special case of a kinky inference rule: For a real-valued single-output function that is Lipschitz with respect to a special input space norm and where the data is error-free the authour provides an algorithm that operates in concert with a tayloured tree data structure and promises logarithmic prediction time. An investigation in how far his ideas can be translated to our more general KI framework, especially in an online learning context, as well as a comparison with our kNN-KI approach will have to be deferred to future work.

None of the aforementioned works seems to consider interval-bounded input uncertainty and observational errors, fold in additional knowledge such as boundedness or considers inference over multi-output functions as we do. Finally, we are not aware of any work that employs these methods in the context of learning in dynamical systems, control or collision avoidance.

At the bottom line, while it seems that the general idea that is at the heart of our kinky inference framework is not new, our approach offers several novel extensions and generalisations that are important for our control and coordination applications. Most importantly for the aims of this thesis, the merger with system identification and control will yield a variety of new adaptive controllers with guarantees of boundedness we

also believe to be novel.

In preparation of subsequent parts of the thesis that take advantage of Hölder properties this section will proceed to establish essential prerequisites. The remainder of this section is structured as follows: Firstly, we will go over basic definitions and engage in some preliminary derivations that will be of importance throughout the thesis. Afterwards we will briefly talk about utilising Hölder properties for conservative inference over extrema – that is, for bounded optimisation. While we do not claim novelty on any of the results we provide proofs for in this section, we had not found them in the literature and hence, had to derive them on our own.

Basic facts and derivations

Before proceeding, we will establish basic properties of Hölder continuity.

Definition 2.5.1. *Let $(\mathcal{X}, \mathfrak{d}_\mathcal{X}), (\mathcal{Y}, \mathfrak{d}_\mathcal{Y})$ be two (pseudo-) metric spaces and $I \subset \mathcal{X}$ be an open set. A function $f : \mathcal{X} \to \mathcal{Y}$ is called (L-p-)* Hölder *(continuous) on $I \subset \mathcal{X}$ if there exists a* (Hölder) constant $L \geq 0$ *and* (Hölder) exponent $p \geq 0$ *such that*

$$\forall x, x' \in I : \mathfrak{d}_\mathcal{Y}\big(f(x), f(x')\big) \leq L \left(\mathfrak{d}_\mathcal{X}(x, x')\right)^p.$$

We denote the space of all L-p- Hölder functions by $\mathfrak{H}_\mathfrak{d}(L, p)$.

Hölder functions are known to be uniformly continuous. A special case of importance is the class of L-*Lipschitz* functions. These are Hölder continuous functions with exponent $p = 1$. In this context, coefficient L is referred to as *Lipschitz constant* or *Lipschitz number*.

Example 2.5.2 (Square root function). As an example of a Hölder function that is not Lipschitz we can consider $x \mapsto \sqrt{x}$ on domain $I = [0 + \epsilon, c]$ where $c > \epsilon \geq 0$. For $\epsilon > 0$ the function is Lipschitz with $L = \sup_{x \in I} \frac{1}{2\sqrt{x}}$. We can see that the coefficient grows infinitely large as $\epsilon \to 0$. By contrast, the function is Hölder continuous with Hölder coefficient $L = 1$ and exponent $p = \frac{1}{2}$ for any bounded $I \subset \mathbb{R}$. We can see this as follows: Let $\epsilon = 0$, $x, y \in I$ and, without loss of generality, $y \geq x$. Let $\xi := \sqrt{x}, \gamma := \sqrt{y}$ and thus, $\gamma \geq \xi$. We have: $\xi \leq \gamma \Leftrightarrow 2\xi^2 \leq 2\xi\gamma \Leftrightarrow \gamma^2 - 2\xi\gamma + \xi^2 \leq \gamma^2 - \xi^2 \Leftrightarrow (\gamma - \xi)^2 \leq \gamma^2 - \xi^2$ $\Leftrightarrow |\gamma - \xi|^2 \leq \left|\gamma^2 - \xi^2\right| \Leftrightarrow |\gamma - \xi| \leq \sqrt{|\gamma^2 - \xi^2|} \Leftrightarrow \left|\sqrt{x} - \sqrt{y}\right| \leq |y - x|^{\frac{1}{2}}$. Since, x, y were chosen arbitrarily, we have shown Hölder continuity as desired.

Most commonly, one considers Hölder continuity for the special case of the standard metric induced by a norm, i.e. $\mathfrak{d}(x, x') = \|x - x'\|$. For a function $f : \mathcal{X} \to \mathcal{Y}$, the Hölder condition becomes:

$$\forall x, x' \in I : \left\|f(x) - f(x')\right\|_\mathcal{Y} \leq L \left\|x - x'\right\|_\mathcal{X}^p.$$

Similarly, we can consider Hölder continuity for each output component:

Definition 2.5.3. *Let $\mathcal{Y} \subseteq \mathbb{R}^m$ and $\mathcal{X}$ be a space endowed with a metric (or indeed a semi-metric) $\mathfrak{d}_{\mathcal{X}}$. Then, the function $f : \mathcal{X} \to \mathcal{Y}$ is output-component-wise Hölder continuous with exponent p and constant $L \in \mathbb{R}^m_{\geq 0}$ if $f \in \mathfrak{H}_{\mathfrak{d}_{\mathcal{X}}}(L, p)$ where*

$$\mathfrak{H}_{\mathfrak{d}_{\mathcal{X}}}(L, p) := \left\{\phi : \mathcal{X} \to \mathcal{Y} \mid \forall j \in \{1, ..., m\}, \forall x, x' \in \mathcal{X} : \left|\phi_j(x) - \phi_j(x')\right| \leq L_j \, \mathfrak{d}^p_{\mathcal{X}}(x, x')\right\}$$

is the set of all functions whose component functions are Hölder continuous with respect to input space metric $\mathfrak{d}_{\mathcal{X}}$ and an output space metric that is induced by the canonical norm $\mathfrak{d}_{\mathcal{Y}}(x, x') = |x - x'|$.

Remark 2.5.4 (Best Hölder constant). Note for $p \in (0, 1], 0 \leq L_1 \leq L_2$ we have $\mathfrak{H}_{\mathfrak{d}_{\mathcal{X}}}(L_1, p) \subseteq \mathfrak{H}_{\mathfrak{d}_{\mathcal{X}}}(L_2, p)$. The smallest $L^* \geq 0$ such that function if is $L^* - p-$ Hölder, $f \in \mathfrak{H}_{\mathfrak{d}_{\mathcal{X}}}(L^*, p)$, is called the *best* Hölder constant of f.

Theorem 2.5.5. *Let $(\mathcal{X},\ d)$ be a metric space and $f, g : \mathcal{X} \to \mathcal{X}$ be two Hölder continuous mappings with Hölder constants $L(f), L(g)$ and Hölder exponents p_f, p_g, respectively. Then, the concatenation $h = f \circ g : \mathcal{X} \to \mathcal{X}$ is also Hölder continuous with Hölder constant $L(h) := L(f)L(g)^{p_f}$ and exponent $p_h := p_g \, p_f$. That is,*

$$\forall x, x' \in \mathcal{X} :\ d\big(h(x), h(x')\big) \leq L(h) \left(\ d(x, x')\right)^{p_h}.$$

Proof. $\mathrm{d}\big(f \circ g(x), f \circ g(x')\big) \leq L(f) \left(\ \mathrm{d}(g(x), g(x'))\right)^{p_f} \leq L(f) \left(L(g)\ \mathrm{d}(x, x')^{p_g}\right)^{p_f}$
$= L(f)\, L(g)^{\,p_f} \left(\ \mathrm{d}(x, x')\right)^{p_g \, p_f}$ where in the first step we were using Hoelder-continuity of f and in the second, we were using Hölder continuity of g combined with the fact that $(\cdot)^{p_f}$ is a monotonically increasing function. □

Several numerical methods, such as Lipschitz optimisation [230], rely on the knowledge of a Lipschitz constant. In the more general case of Hölder continuous functions this will correspond to the need of knowing a Hölder constant and exponent. To avoid having to derive these for each new function from first principles, we establish the following collection of facts that allows us determine bounds on Hölder constants of combinations of functions with known Hölder parameters. While we provide proofs for a restatement in the Hölder continuous setting, a number of the following statements have also been proven in [262] in the context of Lipschitz algebras.

Lemma 2.5.6 (Hölder arithmetic)**.** *Let, $I, J \subset \mathcal{X}$ where $\mathcal{X}$ is a metric space endowed with metric $\mathfrak{d}$. Let $f : \mathcal{X} \to \mathbb{R}$ be Hölder on I with constant $L_I(f) \in \mathbb{R}_+$ and $g : \mathcal{X} \to \mathbb{R}$ be Hölder on J with constant $L_J(g) \in \mathbb{R}_+$. Assume both functions have the same Hölder exponent $p \in (0, 1]$. That is, $\forall x, x' \in \mathcal{X} : |f(x) - f(x')| \leq L(f)\, \mathfrak{d}(x, x')^p$ and $\forall x, x' \in \mathcal{X} : |g(x) - g(x')| \leq L(g)\, \mathfrak{d}(x, x')^p$. We have:*

1. *Mapping $x \mapsto |f(x)|$ is Hölder on I with constant $L_I(f)$ and exponent p_f.*
2. *If g is Hölder on all of $J = f(I)$ the concatenation $g \circ f : t \mapsto g(f(t))$ is Hölder on I with constant $L_I(g \circ f) \leq L_J(g)\, L^p_I(f)$ and exponent p^2.*

3. *Let $r \in \mathbb{R}$. $r\,f : x \mapsto r\,f(x)$ is Hölder on I having a constant $L_I(r\,f) = |r|\,L_I(f)$.*

4. *$f + g : x \mapsto f(x) + g(x)$ has Hölder constant at most $L_I(f) + L_J(g)$.*

5. *Let $m_f = \sup_{x\in\mathcal{X}} f(x)$ and $m_g = \sup_{x\in\mathcal{X}} g(x)$. Product function $f \cdot g : x \mapsto f(x)\,g(x)$ has Hölder exponent p and a Hölder constant on I which is at most $(m_f\,L_J(g) + m_g\,L_I(f))$.*

6. *For some countable index set $\mathcal{I}_t$, let the sequence of functions f_i be Hölder with exponent p and constant $L(f_i)$ each. Furthermore, let $H(x) = \sup_{i\in\mathcal{I}_t} f_i(x)$ and $h(x) := \inf_{i\in\mathcal{I}_t} f_i(x)$ be finite for all x. Then H, h are also Hölder with exponent p and have a Hölder constant which is at most $\sup_{i\in\mathcal{I}_t} L(f_i)$.*

7. *Let $b := \inf_{x\in\mathcal{X}} |f(x)| > 0$ and let $\phi(x) = \frac{1}{f(x)}, \forall x \in \mathcal{X}$ be well-defined. Then $L_I(\phi) \leq b^{-2}\,L_I(f)$.*

8. *Let $p = 1$ (that is we consider the Lipschitz case), let I be convex and $\mathfrak{d}(x, x') = \|x - x'\|$. f cont. differentiable on $I \Rightarrow L_I(f) \leq \sup_{x\in I} \|\nabla f(x)\|$. For one-dimensional input space, $\mathcal{X} = \mathbb{R}$, $L_I(f) = \sup_{x\in I} |\nabla f(x)|$ is the smallest Lipschitz number.*

9. *Let $c \in \mathbb{R}$, $f(t) = c, \forall x \in I$. Then f is Hölder continuous with constant $L_I(f) = 0$ and for any coefficient $p_f \in \mathbb{R}$.*

10. *$L_I(f^2) \leq 2\,L_I(f)\,\sup_{t\in I} f$.*

11. *With conditions as in 8), and input space dimension one, we have $\forall q \in \mathbb{Q} : L_I(f^q) = |q|\,\sup_{\tau\in\mathcal{I}} |f^{q-1}(\tau)\,\dot{f}(\tau)|$.*

Proof. **1)** We show $|f|$ has the same constant and exponent as f. Let $X, X' \in \mathcal{X}$ arbitrary. We enter a case differentiation:

1st case: $f(x), f(x') \geq 0$.
Hence, $\big||f(x)| - |f(x')|\big| = \big|f(x) - f(x')\big| \overset{f\,Hoelder}{\leq} L_I(f)\,\mathfrak{d}(x, x')^p$.

2nd case: $f(x) \geq 0, f(x') \leq 0$.
Note, $|y| = -y$, iff $y \leq 0$. Hence, $\big||f(x)| - |f(x')|\big| \leq \big||f(x)| + |f(x')|\big| = \big||f(x)| - f(x')\big| = \big|f(x) - f(x')\big| \leq L_I(f)\,\mathfrak{d}(x, x')^p$.

3rd case: $f(x) \leq 0, f(x') \geq 0$. Completely analogous to the second case.

4th case: $f(x), f(x') \leq 0$.
$\big||f(x)| - |f(x')|\big| = \big|f(x') - f(x)\big| = \big|f(x) - f(x')\big| \overset{f\,Hoelder}{\leq} L_I(f)\,\mathfrak{d}(x, x')^p$.

The remaining points are also proven in [262] in the context of Lipschitz functions.

2) Special case of Thm. 2.5.5.

3) For arbitrary $x, x' \in \mathcal{X}, r \in \mathbb{R}$ we have:

$\big|r\,f(x) - r\,f(x')\big| = |r|\,|f(x) - f(x')| \leq |r|\,L_I(f)\;\mathfrak{d}(x,x')^p$.

4) For arbitrary $x, x' \in \mathcal{X}, r \in \mathbb{R}$ we have:

$\big|g(x) + f(x) - (g(x') + f(x'))\big| = \big|g(x) - g(x') + f(x) - f(x')\big| \leq \big|g(x) - g(x')\big| + \big|f(x) - f(x')\big|$
$\leq (L_J(g) + L_I(f))\;\mathfrak{d}(x,x')^p$.

5) Let $x, x' \in \mathcal{X}$, $d := f(t) - f(t')$.

$\big|g(x)f(x) - g(x')f(x')\big| = \big|g(x)(f(x') + d) - g(x')f(x')\big| = \big|\big(g(x) - g(x')\big)f(x') + g(x)d\big|$
$\leq \big|g(x) - g(x')\big||f(x')| + \big|g(x)\big|\,|d| \leq L_I(g)\,\mathfrak{d}(x,x')^p|f(x')| + \big|g(x)\big|\,L_I(f)\,\mathfrak{d}(x,x')^p$
$\leq L_I(g)\,\mathfrak{d}(x,x')^p \sup_{x'\in\mathcal{X}}\{|f(x')|\} + \sup_{x\in\mathcal{X}}\{\big|g(x)\big|\}\,L_I(f)\,\mathfrak{d}(x,x')^p$
$= \Big(L_I(g)\sup_{x'\in\mathcal{X}}\{|f(x')|\} + \sup_{x\in\mathcal{X}}\{\big|g(x)\big|\}\,L_I(f)\Big)\,\mathfrak{d}(x,x')^p$.

6) The proof of Proposition 1.5.5 in [262] proves our statement if one replaces their metric ρ by $\mathfrak{d}^p$.

7) Let $x, x' \in \mathcal{X}$. $\big|\frac{1}{f(x)} - \frac{1}{f(x')}\big| = \big|\frac{f(x')}{f(x')f(x)} - \frac{f(x)}{f(x')f(x)}\big| = \frac{\big|f(x')-f(x)\big|}{\big|f(x')\big|\big|f(x)\big|} \leq \frac{L_I(f)\,\mathfrak{d}(x,x')^p}{\inf_{x\in\mathcal{X}}|f(x)|^2}$.

8) Let $p = 1$ and $\mathfrak{d}(x,x') = \|x - x'\|$ be the standard norm. Define $\ell := \sup_{x\in I}\|\nabla f(x)\| = L_I(f)$. Firstly, we show that it is a Lipschitz constant: Let $x, x' \in I$ and $\overline{xx'} := \{tx + (1-t)x' \,|\, t \in [0,1]\}$. Owing to convexity of I, $\overline{xx'} \subset I$. Due to the mean value theorem $\exists \xi \in \overline{xx'} \subset I : |f(x) - f(x')| = T_\xi(x - x')$. where $T_\xi(x) = \langle \nabla f(\xi), x\rangle$ is a linear OP. Assuming the derivative of f is bounded, T_ξ is a bounded OP and we have $|T_\xi(x - x')| \leq |||T_\xi||| \, \|x - x'\|$ where $|||T_\xi||| = \sup_{\|x\|=1}\langle \nabla f(\xi), x\rangle \leq \|\nabla f(\xi)\|$. In conjunction, $|f(x) - f(x')| \leq \|\nabla f(\xi)\|\,\|x - x'\|$.

Secondly, we show that ℓ is the smallest Lipschitz constant in the one-dimensional case: Let $\bar{\ell}$ be another Lipschitz constant on I such that $\bar{\ell} \leq \ell$. Of course, here $\|\cdot\| = |\cdot|$. Since I is compact and $\|\nabla f(\cdot)\|$ is continuous, there is some $\xi \in I$ such that $\|\nabla f(\xi)\| = \sup_{x\in I}\|\nabla f(\xi)\| = \ell$. Pick any sequence $(y_k)_{k=1}^\infty$ contained in I such that $y_k \overset{k\to\infty}{\longrightarrow} \xi$ and $y_k \neq \xi$. $\forall k : y_k \in I$ and $\bar{\ell}$ is a Lipschitz constant on I. Hence, $\bar{\ell} \geq \frac{|f(y_k)-f(\xi)|}{\|y_k-\xi\|} \overset{k\to\infty}{\longrightarrow} \|\nabla f(\xi)\| = \ell$. Thus, $\bar{\ell} = \ell$.

9) Trivial.

10) Special case of property 5).

11) $L(f^q) \overset{8)}{=} \sup_{\tau\in\mathcal{I}}|\frac{\mathrm{d}}{\mathrm{d}t}f^q(\tau)| = |q|\,\sup_{\tau\in\mathcal{I}}|f^{q-1}(\tau)\,\dot{f}(\tau)|$.

□

As a simple illustration, assume we desire to establish that $f(t) = \max\{1 - 3\sin(t), \exp\big(-\sin(t)\big)\}$ is Lipschitz and to find a Lipschitz number on $\mathbb{R}$. Application of 8. shows that $t \mapsto \sin(t)$ and $t \mapsto \exp(-t)$ have a Lipschitz number of 1. Application, of 2., 9. 1. and 6. then show that $L(f) = 3$ is a Lipschitz number of f.

An example that will be of relevance to our collision avoidance applications in Ch. 5 is the following:

Example 2.5.7. Consider the function $h(t) = f \circ g(t)$ where $f(\cdot) = \sqrt{\cdot}$ is the square root function and $g(t) = \frac{B^2}{2K}(1 - \exp(-2Kt))$, $K > 0$. g is the non-stationary factor of the variance trajectory of an Ornstein-Uhlenbeck process (cf. Sec. 2.4.6). We show that h is Hölder and has constant $L(h) = \sqrt{2K}$ and Hölder exponent $p_h = \frac{1}{2}$. This can be easily done by applying Lem. 2.5.6.8 to show that $L(g)$ is Lipschitz with constant $L(g) = B^2$ and then combining the Hölder constant and exponent derived in Ex. 2.5.2 with Thm. 2.5.5 to yield the Hölder exponent $p_h = p_g p_f = \frac{1}{2}$ and constant $L(h) = 1\sqrt{L(g)} = |B|$.

A note on Hölder continuity for optimisation bounds

In the presence of a known Lipschitz constant, optimisation of a Lipschitz continuous function can be done with Shupert's algorithm [230]. Of importance to us at various parts of the thesis is, that it also provides non-asymptotic error bounds. Unfortunately, it is limited to minimising functions on one-dimensional input domains. For multi-dimensional domains DIRECT [124] is a global optimiser that is popular, not least because no Lipschitz constant is required. On the flip side, it does not provide the error bounds afforded by Shupert's methods. Furthermore, we will sometimes be interested in bounds on the maxima and minima for the more general case, where the functions to be optimised are Hölder continuous with respect to normed spaces. We are currently working on such a method, as well as on a corresponding cubature algorithm. As it is work in progress and its exposition is beyond the scope of this work, we will merely sketch a naive version that provides some desired bounded optimisation for Hölder continuous functions:

Let $f : I \subset \mathbb{R}^d \to \mathbb{R}$ be a Hölder continuous function with Hölder constant L and exponent p given with respect to the maximum-norm $\|\cdot\|_\infty$.[6] Let $S = \{(s_i, f_i) | i = 1, ..., N\}$ be a sample of target function f with $f_i = f(s_i)$, $s_i \in I, \forall i$.

Assume I is a hyper-rectangle and let $J_1, ..., J_N \subset I$ be a partition of the domain I such that each sub-domain J_j is a hyperrectangle containing s_j. (More generally, to avoid I having to be a hyperrectangle itself, one could alternatively assume that the $J_1, \ldots, J_N$ are a covering of I. That is, $I \subset \cup_i J_i$.)

By Hölder continuity $\forall x \in J_j : |f_j - f(x)| \leq L \|x - s_j\|_\infty^p$ for some $p \in (0, 1], L \geq 0$. Let $a_1, \ldots, a_d > 0$, $b_1, \ldots, b_d > 0$ be defined such that $J_j = s_j + ([-a_1, b_1] \times ... \times [-a_d, b_d])$ where the addition is defined point-wise. Furthermore, let

$$D_j := \max\{a_1, \ldots, a_d, b_1, \ldots, b_d\} \tag{2.44}$$

be the maximal distance of s_j to the boundary of J_j.

Then, $\forall x \in J_j : f(x) \leq f_j + L \max_{x \in J_j} \|x - s_j\|_\infty^p = f_j + L D_j^p =: q_j$.

Since, the hyperrectangles formed a covering of domain I, the desired upper bound on the maximum is

[6] Due to norm-equivalence relationships we can bound each norm by each other norm by multiplication with a well-known constant (see e.g. [125]). Therefore, knowing a Hölder-constant with respect to one norm immediately yields a constant with respect to another, e.g. $\|\cdot\|_\infty \leq \|\cdot\|_2 \leq \sqrt{d}\|\cdot\|_\infty$ where d is the dimension of the input space.

$\max_{j=1}^{N} q_j \geq \max_{x \in I} f(x)$. By the same argument, a lower bound on the maximum based on the finite sample can be obtained as: $\min_{j=1}^{N} f_j - L D_j^p \leq \min_{x \in I} f(x)$.

In conclusion, we have

$$\underline{M} := \min_{j=1\,...,N} f_j - L\, D_j^p \leq \min_{x \in I} f(x) \tag{2.45}$$

$$\overline{M} := \max_{j=1,...,N} f_j + L\, D_j^p \geq \max_{x \in I} f(x). \tag{2.46}$$

A *batch* algorithm would get a sample, construct a partition $J_1, ..., J_N$ for that sample and then compute the minimum and maximum bounds $\underline{M}, \overline{M}$ given above.

An *adaptive* algorithm for bounded optimisation would be able to incrementally expand the given sample. That is, it would incrementally partition domain I into increasing small sub-hyperrectangles until the computational budget is expended or the bounds on the maxima and minima have shrunk satisfactorily.

The exploration criterion of where to obtain the next function sample could be tailored to finding $\overline{M}$ or $\underline{M}$, depending on the task at hand.

For instance, in a minimisation problem, one might choose to obtain a new sample point in the hyperrectangle J_{j_*} where our present bounds allow for the lowest function value to be. Its index is $j_* = \arg\min_{j=1\,...,N} f_j - L\, D_j^p$.

2.6. Inference with Gaussian random fields - a Bayesian nonparametric machine learning perspective

Non-deductive inference is the process of drawing conclusions about the general on the basis of the particular. Machine learning is about the automated construction of a mathematical model that facilitates non-deductive inference. These days, supervised machine learning essentially is function estimation. That is, given a (noisy) sample of a target function, artificial learning is the process of constructing a computable model of a function that generalises beyond the observed sample. While parametric machine learning subsumes all the information contained in the sample in a parameter of fixed size, nonparametric machine learning typically takes the entire sample into account when making inference. That is, it creates models for non-deductive inference whose complexity grows with the sample size. This conceptual difference typically leads to complementary advantages and disadvantages [171]: Parametric methods typically can take longer to aggregate all information into the finite parameter, have stronger inductive bias, but once a model is learned, inference speed is determined by the size of the parameter, not the size of the data and thus, can be very fast. For instance, in artificial neural networks, the parameter is given by the weights of the network links. Once the weights are determined from the training data, the required computation only depends on the number of

weights, not on the size of the training data.

By contrast, Gaussian random fields (GRFs) have been acknowledged as a nonparametric machine learning method in artificial intelligence since about the 1990s [264]. Being referred to as Kriging in geostatistics [238], in the machine learning community inference over functions with GRFs is commonly referred to as *Gaussian process (GP) regression (GPR)* – regardless of the input space dimensionality. As is the case with most nonparametric learning methods, the computational complexity for training and prediction grows with number of training examples. In fact, the computational complexity for training is cubic and the computational complexity for inference (prediction) is quadratic in the number of data points [201] (although the former can be reduced to quadratic effort either approximately [104] or exactly for specific cases [243] and the latter effort can be reduced to linear growth by recent SDE conversion techniques [203, 215]). As most non-parametric methods, the increased computational complexity is rewarded with a reduced inductive bias. That is, choosing the right covariance function, Gaussian processes can learn any smooth function.

In what is to follow, we will briefly rehearse the algorithmics of inference and learning with Gaussian processes. For an extensive treatment, the reader is referred to the standard textbook by Williams and Rasmussen [201].

2.6.1. Finite-index sets – Gaussian distributions

For now, consider the Hilbert space $(\mathbb{R}^d, \langle\cdot,\cdot\rangle_2)$ where $\langle\cdot,\cdot\rangle_2 : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ denotes the canonical Euclidean scalar product.

All central moments of a d-dimensional *normal* or *Gaussian* distribution $\mathcal{N}(\mu, K)$ are uniquely determined by its mean $\mu \in \mathbb{R}^d$ and covariance matrix $K \in \mathbb{R}^{d\times d}$. As the moments uniquely determine the entire distribution the Gaussian is uniquely determined by μ and K, hence the notation $\mathcal{N}(\mu, K)$. The Gaussian distribution is a probability distribution with well-defined density on $\mathbb{R}^d$

$$p_{\mathcal{N}}(x;\mu,K) = (2\pi)^{\frac{-d}{2}}\,|K|^{\frac{-1}{2}}\exp(-\frac{1}{2}(x-\mu)^T K^{-1}(x-\mu)). \tag{2.47}$$

The Gaussian distribution has many analytic advantages. Any linear or affine transformation of a Gaussian is a Gaussian, as are the conditionals and marginals.

Let $x : \Omega \to \mathbb{R}^q, y : \Omega \to \mathbb{R}^r, q, r > 0, q + r = d$ two random vectors. Consider the joint random vector $(x, y)^T$ with joint Gaussian distribution $\begin{bmatrix} x \\ y \end{bmatrix} \sim \mathcal{N}(\begin{bmatrix} \mu_x \\ \mu_y \end{bmatrix}, \begin{bmatrix} K_{xx} & K_{xy} \\ K_{yx} & K_{yy} \end{bmatrix})$ where $K_{xy} \in \mathbb{R}^{q\times r}$ is the covariance block matrix whose (i, j)th element contains the covariance $k(x(i), y(j))$ between the ith component of x and the jth component of y and where $K_{yx} \in \mathbb{R}^{r\times q}, K_{xx} \in \mathbb{R}^{q\times q}, K_{yy} \in \mathbb{R}^{r\times r}$ are defined analogously.

Then the marginal distributions can be read-off the joint moments: $x \sim \mathcal{N}(\mu_x, K_{xx}), \quad y \sim \mathcal{N}(\mu_y, K_{yy})$.

The conditional of x given y is

$$x|y \sim \mathcal{N}\big(\mu_x + K_{xy}K_{yy}^{-1}(y-\mu_y), K_{xx} - K_{xy}K_{yy}^{-1}K_{yx}\big). \tag{2.48}$$

Of course we can interpret a vector $f \in \mathbb{R}^d$ as a function $f : I \to \mathbb{R}$ where I is a set of d elements. In this light, $\Sigma \in \mathbb{R}^{d\times d}$ can be interpreted as a bounded linear operator K on $\mathbb{R}^d$ and $f^T = f^* : \mathbb{R}^d \to \mathbb{R}$ is the dual of f mapping vector $x \in \mathbb{R}^d$ to $\langle x,f\rangle_2$. From this point of view, $f^T K^{-1}$ is a concatenation of linear operators, yielding $f^T K^{-1} f = \langle K^{-1}f,f\rangle$. Therefore, it becomes obvious we can rewrite our normal density as

$$p_{\mathcal{N}}(f|\mu,K) \propto \exp(-\frac{1}{2}\langle K^{-1}(f-\mu),(f-\mu)\rangle_2). \tag{2.49}$$

2.6.2. Possibly infinite index sets –Gaussian Random Fields

Gaussian random fields generalize the notion of Gaussian distributions from finite index sets to potentially infinite index sets. Looked at at in another way, if a Gaussian distribution places a probabilistic measure of a finite-dimensional, Euclidean vector space, a Gaussian process (GPs) represents a distribution over a potentially infinite dimensional measurable Hilbert space. Since the latter can be rich function spaces and due to the attractive analytic properties inherented from Gaussianty, GPs have been extensively applied to Bayesian machine learning.

In the context of machine learning, GRFs have been proposed to solve problems in domains as diverse as learning inverse dynamics [179], signal processing [195], sensor networks [190], computational zoology [163], global optimization and quadrature [1, 189, 191].

In general, a stochastic process is a family $(X_t)_{t\in I}$ of random variables with index set I. In alternative notation it is also customary to write the stochastic process as $X : I \times \Omega \to \mathbb{R}$ where Ω denotes the joint sample space of the underlying Σ-algebra of events. Finite-dimensional distributions are stochastic processes with finite index sets.

While a Gaussian *distribution* $\mathcal{N}(\mu,K)$ ($\mu : d \to \mathbb{R}, K : d \times d \to \mathbb{R}$) is a probability distribution over d-dimensional vectors a *Gaussian Random Field* (GRF) is a random field interpretable as a distribution over functions $f : I \to \mathbb{R}$, typically $I \subset \mathbb{R}^d$. In machine learning, GRFs are typically referred to as *Gaussian processes (GPs)*, regardless of the dimension of the index set. Therefore, we will also sometimes use these terms interchangeably.

The name is inspired by the property of a GP that its marginal distributions of a finite number of vectors is a consistent Gaussian. The existence of such a construction in the limit of infinite finite-dimensional distributions is asserted by Kolmogorov's extension theorem that applies to all (projective) collections of finite distributions with symmetric and consistent measures (see e.g. [21]).

(Interestingly, this fact is a theoretical justification for conditions on inference over unobserved points in

an infinite index set on the basis of a finite number of observations.)

Similar to a Gaussian distribution, that is parameterized by a finite dimensional mean *vector* μ and a covariance *matrix* K, a GP is determined by a mean $function\ m : \mathbb{R}^d \to \mathbb{R}$ and a covariance *function* or *kernel (function)* $k : I^2 \to \mathbb{R}$. We write $f \sim \mathcal{GP}(m, k)$ where $m(x) = \mathbb{E}[f(x)]$ and $k(x, x') = \mathbb{E}[(f(x) - m(x))(f(x') - m(x'))]$.

Note, $k : \mathbb{R}^d \to [0, 1]$ is a proper covariance iff it is a proper kernel, i.e. iff it is positive definite and symmetric. If it is then it gives rise to a reproducing kernel Hilbert space (*RKHS*) $\mathcal{H}_k$, with a positive definite and symmetric scalar product using k (or k^{-1} as its integral kernel, e.g. $\langle f,g\rangle_{\mathcal{H}_k} = \int_{I^2} f(x)\, k(x, y)\, \bar{g}(y)\, d\mu(x, y)$ [11]. We see that $\langle \cdot\,, \cdot \rangle_{\mathcal{H}_k}$ is a scalar product iff k is a positive definite function (or, equivalently, if the pertaining kernel operator is positive definite). From the perspective of marginals, notice that a Gaussian density cannot be defined if k is not at least nonnegative definite, since then a Gaussian density $p_{\mathcal{N}}(f|\mu, K) \propto \exp(-\frac{1}{2}\langle K^{-1}(f - \mu), (f - \mu)\rangle_2)$ with covariance matrix defined by $K_{ij} = k(x_i, x_j)$ would be able to attain values greater than one.

As an example, one of the most basic but popular kernel functions is the *Squared Exponential (SE)* $k(x, x') = h\, \exp(-\|x - x'\|^2 / l)$. Another kernel we will use later on is the Ornstein-Uhlenbeck (OU)-kernel $k(x, x') = \frac{h}{2l} \exp(-l|x - x'|)$. Here $\theta := (l, h)$ are *hyperparameters* that are inferred automatically during training either with the ML-principle or are marginalized out with respect to a prior density over the hyperparameter space [201].

A kernel we will make use of in Ch. 3 is the *rational quadratic kernel (with automated relevance detection) RQ-ARD*. It is given by $k(x, x') = h\left(1 + \sum_{i=1}^{d} \frac{|x_i - x'_i|}{2\,\alpha\, l_i^2}\right)^{-\alpha}$ where parameter $\theta = (l_1, \ldots, l_d, h, \alpha)$. Note the separate length scale parameter l_i for each dimension i. It allows one to regulate the degree of influence each dimension has on the covariance, that is, how "relevant" the components are. The fact that these hyperparameters often are determined automatically via optimisation of the marginal log-likelihood [201] explains the expression *automated relevance determination*.

Intuitively, it is often possible to choose some pos. def. similarity measure where large $k(x, y)$ indicates similar x and y in the light of domain knowledge [201].

We will need the following well-known fact that allows us to construct GPs over X from observations of another process that is connected to X via a linear relationship:

Theorem 2.6.1. *Let X be a stochastic process with mean m_x and covariance $k_{x,x}$. Let L be a linear operator and $L_i k_{x,x}$ denote the application of L to the ith input of multi-input function $k_{x,x}$. We have: $Y := L\,X$ is a stochastic process with mean function Lm and covariance function $k_{y,y} = L_1 L_2 k$. The cross-covariances between Y and X are given by $k_{y,x} = L_1 k_{x,x}$.*

Probabilistic Regression

As stated above, a GRF can be construed as a prior over functions.

In GRF regression, we desire to compute the posterior conditional on training points. The posterior can then be used for inferring the function values at unseen test inputs. The posterior's variance even quantify the associated uncertainty of the predictions. The training data consists of a set of sampling points $X = \{x_1, \ldots, x_n\} \subset \mathbb{R}^d$ and sample values $y_1, \ldots, y_n \in \mathbb{R}$ which collectively constitute the training set D. The sample values y_i represent noisy measurements – a superposition of function values at the x_i and iid Gaussian noise. That is, $y_i = f(x_i) + e_i$ where $e_i \sim \mathcal{N}(0, \sigma_i^2)$ independently, for some (potentially non-stationary) variances σ_i^2. For notational convenience we write the samples values as a vector $y := (y_1, \ldots, y_n)^T \in \mathbb{R}^n$.

As is commonly done in practice, one could maximize the likelihood to adjust the covariance's hyper-parameters in the light of the training data $\mathcal{D} = (X, y)$ and then compute the posterior GRF or better yet, we could marginalize over the hyper-parameters with respect to our belief density. This posterior GRF can be utilized to make predictions of function values $f_* := f(x_*) \in \mathbb{R}$ of unobserved test points $x_* \in \mathbb{R}^d$ simply by applying the posterior mean function to it. In case the predictions serve to make a decision, the Bayesian alternative would be to integrate according to the distribution of function values at the test point. If the integrals are non-analytic we would have to resort to approximative integration techniques such as Monte-Carlo [9] or Bayesian quadrature [186, 191] to approximate the resulting posteriors.

For a given test point $x_* \in \mathbb{R}^d$, let $k(x_*, x_\mathcal{D}) := \kappa_* := (k(x_*, x_i))_{i=1,\ldots,n} \in \mathbb{R}^n$. That is, vector κ_*'s ith component $\kappa^i(x_*)$ quantifies the test input's similarity with the ith training example. As another definition we need below, let $K = ((\, k(x_i, x_j)\,))_{i,j=1,\ldots,n} \in \mathbb{R}^{n\times n}$ be the matrix resulting from computing the covariances between all training inputs in X. Assume a prior GRF $\mathcal{GP}(\mu, k)$ with prior mean $\mu : I \to \mathbb{R}$ and covariance function k. Now, in the light of finite available data $D = (X, y)$ what is the posterior distribution? Due to Kolmogorov's extension theorem we know the GRF needs to be consistent with its finite-dimensional projections. Hence, we can apply the well known formula for the Gaussian posterior conditioned on the data D yielding $f_*|x_*, y, X \sim \mathcal{N}\big(\bar{f}_*, \text{var}[f_*]\big)$where the posterior mean is given by

$$m(f_*|x_*, D) := \bar{f}_* = \mu(x_*) + k(x_*, x_\mathcal{D})^\top (K + \sigma^2 I)^{-1}(y - \mu(X)) \tag{2.50}$$

where $\mu(X) := (\mu(x_1), \ldots, \mu(x_n))^T \in \mathbb{R}^n$. The posterior's variance is given by

$$\text{var}[f_*] = k(x_*, x_*) - k(x_*, x_\mathcal{D})^\top (K + \sigma^2 I)^{-1} k(x_*, x_\mathcal{D}). \tag{2.51}$$

Here, the first term quantifies the prior uncertainty and the second its reduction due to the information contained in the training data.

Owing to Gaussianity, $\bar{f}_*$ is the MAP estimate of f_*. So, the variance quantifies the uncertainty of this estimate when predicting the function value of test input x_* is the MAP estimate $\bar{f}_*$. As we will discuss below, this uncertainty increases with the dissimilarity of the test input to the previously observed training points in X (with respect to kernel similarity measure k).

Computational complexity

In practical terms, in GRF regression, training or learning is essentially pre-computing those terms in Eq. 2.50 and Eq. 2.51 which do not depend on query point x^* and hence, can be computed as soon as the data has become available. We see that, provided the complexity of evaluating the kernel function is relatively small, the computationally dominant part of training is the inversion of the data matrix $K \in \mathbb{R}^{n \times n}$. For numerical reasons it is advantageous to compute Cholesky factors instead. But nonetheless, the worst-case complexity is $\mathcal{O}(n^3)$.

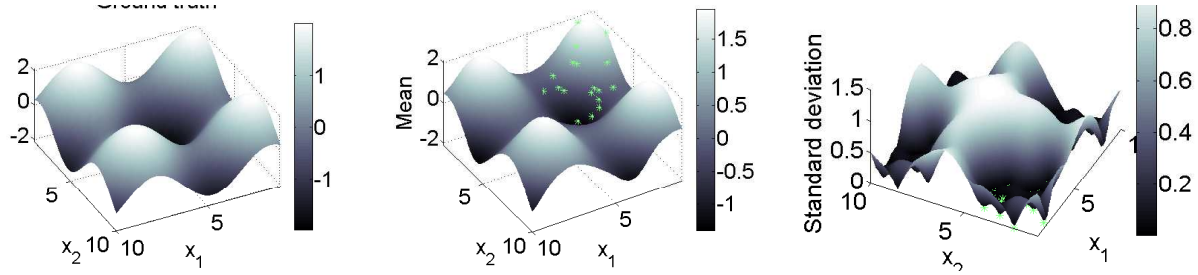
Furthermore, to perform *inference* about query point x^*, Eq. 2.50 and Eq. 2.51. Again, if the kernel function evaluation can be done with constant effort, the computations of posterior mean and variance can be performed with complexity $\mathcal{O}(n^2)$.

Given the relatively high complexity of GRF inference and learning, a large number of works have been dedicated to ameliorating the computational effort. As mentioned above, the training complexity can be reduced to quadratic effort either approximately [104] or exactly for specific cases [243]. Similarly, the computational effort for making predictions can be reduced to linear growth by recent SDE conversion techniques [203, 215]) provided the query points are known in advance. At various points in the thesis, we will employ GRFs in online learning. That is, data become available incrementally and there is a need to make predictions based on most recent information between these knowledge updates. While in principle, it would be possible to recompute the Cholesky factors from scratch each time a new training example arrives this is computational wasteful. Instead, all of our experiments where we utilised GRFs in online learning followed the recommendation of [182] and took advantage of an incremental Cholesky update rule by Seeger [226].

The inductive bias of GRF inference

While a few works have started to take advantage of the uncertainty quantifications afforded by GRF inference [78, 80], most works in applied machine learning focus on utilising the posterior mean for making inferences. In this case, inference is essentially equivalent to regularised regression where the regression function will always lie in the RKHS given by the kernel [260] .

In order to generalise beyond an observed sample, inductive inference needs to make assumptions about the underlying ground truth function. These assumptions, sometimes referred to as *inductive bias* [171], often



(a) Ground truth. (b) Posterior mean. (c) Posterior standard deviation.

Figure 2.1.: Periodic kernel of Gaussian random field on two-dimensional input space. Note, that due to periodicity, a small sample (green stars) confined to the region of $[1, 5] \times [1, 5]$ suffice to make accurate predictions in unobserved input space.

is implicit in the learning method. For instance, in Bayesian non-parametric machine learning, there is an explicit prior. However, there also is an implicit bias imposed by the necessity to choose a probability space the probability measure is defined on. For GRFs, this bias is a restriction of the hypothesis space imposed by the chosen covariance function– the posterior mean predictions are known to lie within the RKHS of the pertaining kernel operator and the sample paths are a.s. contained in its eigenfunction space. While for certain *universal kernels* these spaces cover wide function classes such as the space of smooth or continuous functions [231, 240], other choices of kernels can yield very restrictive hypothesis spaces.

As an example, consider the kernel $k(x, x') := \sum_{i=1}^{n} f_i(x) f_i(x')$. It is not hard to see that this function is a valid kernel whose RKHS $\mathcal{H}_k$ coincides with the span of the functions $f_1, ..., f_n$. For instance, if we desired to learn functions we know are constrained to be solutions to the ODE $\dot{x} = -ax\, dt$, we could choose the f_i to span the solution space of the differential equation. We could define a Gaussian process with this covariance function that would learn solutions to this differential equation. However, this GP is very restricted since the function space is finite-dimensional. Imposing this knowledge is advantageous if the ground-truth is indeed within the ODE's solution space since it will only take a few training examples to perfectly identify the trajectory. However, this strong knowledge also means that the GP will be unable to learn any function not in the finite-dimensional solution space. We will elaborate on these points briefly in what is to follow.

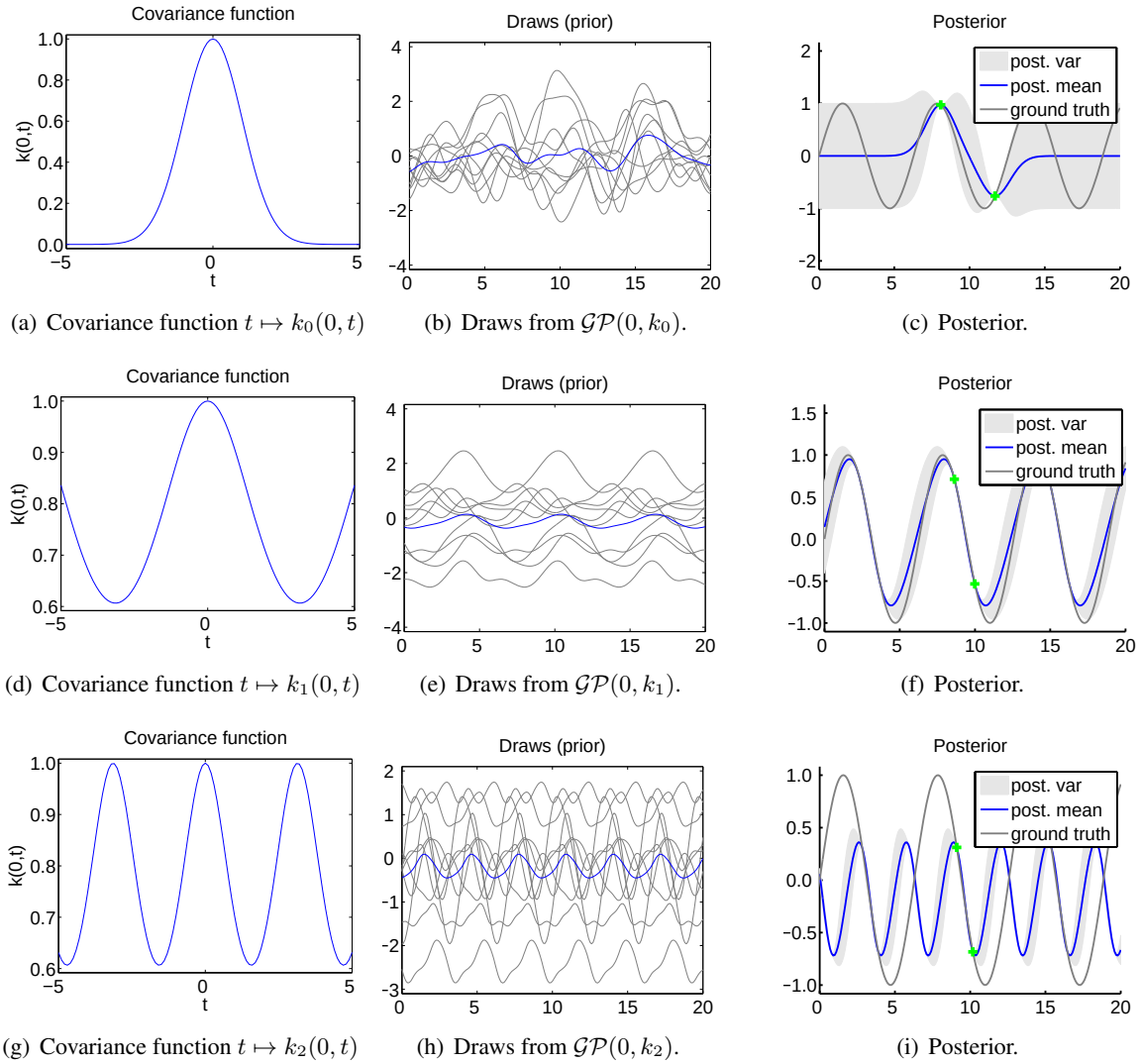
Kernel function value $k(x, x')$ quantifies the similarity between x and x'. Thus, an informed choice of the kernel is important. This intuition can be supported mathematically. For simplicity, assume a zero prior mean function and no observational noise. We can rewrite Eq. 2.50 as

$$m(f(x_*)|D) = \sum_{i=1}^{|D|} \alpha_i k(x_*, x_i), \quad \alpha_i = K^{-1} y. \tag{2.52}$$

If we furthermore acknowledge that the RKHS $\mathcal{H}_k$ can be defined as the closure of kernel mixtures, $\mathcal{H}_k = \text{cl}\left(\{\sum_{j=1}^{M} \gamma_j k(\cdot, x_j) | M \in \mathbb{N}, \gamma_j \in \mathbb{R}, x_j \in I\}\right)$ [130], we realize that the RKHS of our chosen kernel k restricts the space of functions we can infer with our GRFs. This is another way of looking at GRF regression

from the "weight space" point of view [201], as generalized linear regression in a *feature space*. Notice however, that the restriction to feature space may not always be too limiting if we choose a kernel whose RKHS is dense in the space of desirable functions. For instance the SE kernel belongs to the class of *universal* kernels [170, 201, 240]. While there are competing definitions of universality of an RKHS [236], it loosely refers to kernels whose RKHS is dense in rich function classes, for instance the space of all bounded continuous functions on compact domains [231]. As an example, GPR with a SE-kernel is equivalent to linear regression with infinitely many Gaussian basis functions which are known to be dense in the space of smooth functions. Hence, choosing a prior with an SE-kernel is merely imposing smoothness as an inductive bias. In contrast, as mentioned above, we can show that $k(x, x') := \exp(-a(x+x'))$ is a valid covariance. However, its RKHS coincides with the solution space of the ODE $\dot{f} = -af$ and hence, our posterior mean functions we could infer are all confined to this very restricted (finite-dimensional) space.

It should be noted that even for universal kernels, hyper-parameters have to be chosen in advance. In principle, the hypothesis space for GRFs endowed with such kernels is very broad. However, due to the computational and memory complexity of GRF inference, predictions will have to be based on limited data sets. As we will see in the context of our control applications, for any finite data set, the inference accuracy of the predictions is strongly dependent on the choice of kernel, hyper-parameters and the prior. Therefore, for practical purposes, GRF inference is much less "black-box", than the consistency guarantees might tempt one to believe.



(a) Covariance function $t \mapsto k_0(0,t)$ (b) Draws from $\mathcal{GP}(0,k_0)$. (c) Posterior.

(d) Covariance function $t \mapsto k_1(0,t)$ (e) Draws from $\mathcal{GP}(0,k_1)$. (f) Posterior.

(g) Covariance function $t \mapsto k_2(0,t)$ (h) Draws from $\mathcal{GP}(0,k_2)$. (i) Posterior.

Figure 2.2.: Top row: example with SE-kernel (covariance function) with length and output scale parameters set to one. Second row: example with periodic kernel (covariance function) with period length 2π. Bottom row: same example but for kernel with period length π. The ground truth 2nd col.: Draws from a GP with zero mean prior and the periodic kernel (covariance function). 3rd col.: Posterior GP conditioned on two data points(green crosses) drawn from the sine function with period length 2π. Referring to the variance of the posterior under the second model (bottom right plot), note how the ground truth is more unlikely than under the prior using the first kernel (top right plot) featuring the right periodicity. This property is taken advantage of in hyperparameter optimisation where fitting hyperparameters are determined by maximising the log-likelihood function of the data [201].

3. Random fields and feedback linearisation for Bayesian identification and control of control-affine dynamical systems

"I wanted to have a water jet in my garden: Euler calculated the force of the wheels necessary to raise the water to a reservoir, from where it should fall back through channels, finally spurting out in Sans Souci. My mill was carried out geometrically and could not raise a mouthful of water closer than fifty paces from the reservoir. Vanity of vanities! Vanity of geometry!"

Friederich II of Prussia, 1778

This chapter proposes a new method, *random field system identification and inversion control (RF-SIIC)*, for simultaneous probabilistic identification and control of control-affine systems. Identification is achieved by conditioning random field priors on observations of configurations and noisy estimates of configuration derivatives. In contrast to previous work that has utilised random fields for identification, we leverage the structural knowledge afforded by Lagrangian mechanics and learn the drift and control input matrix functions of the control-affine system separately. Our approach uses feedback-linearisation with the aim to reduce, in expectation, the uncertain nonlinear control problem to one that is easy to regulate in a desired manner. Our method combines the flexibility of nonparametric Bayesian learning with epistemological guarantees on the expected closed-loop trajectory. We illustrate the performance of our approach in the context of both fully actuated and underactuated mechanical systems. Our simulations show that our approach is able to very rapidly and accurately learn a dynamic model as well as a pertaining control policy online.

3.1. Introduction

Control may be regarded as decision making in a dynamic environment. Decisions have to be based on beliefs over the consequences of actions encoded by a model. Dealing with uncertain or changing dynamics is the

realm of adaptive control. In classical adaptive control, parametric approaches are used (e.g. [258]) and uncertainties are typically modelled by Brownian motion (yielding stochastic adaptive control [87, 146]) or via set-based considerations (an approach followed by robust adaptive control [192]). In contrast, we adopt an epistemological take on probabilistic control and bring to bear Bayesian nonparametric learning methods whose introspective qualities [116] can aide in addressing exploration-exploitation trade-offs in a principled manner [4].

In contrast to classical adaptive control where inference has to be restricted to finite-dimensional parameter space, the nonparametric approach affords the learning algorithms with greater flexibility to identify and control systems with very few model assumptions. This is possible because these methods grant the flexibility to perform Bayesian inference over rich, infinite-dimensional function spaces that could encode the dynamics. This property has led to a surge of interest in Bayesian nonparametrics; particularly benefiting their algorithmic advancement and application to a plethora of learning problems. Due to their favourable analytic properties, *Gaussian processes* (GPs) [21,201] have been the main choice of method in recent years. Among other domains, GPs have been applied to learning discrete-time dynamic systems in the context of model-predictive control [137,138,173,209], learning the error of inverse models [179,181], dual control [4] as well as reinforcement learning and dynamic programming [78, 79, 135, 212]. For articles surveying the field of learning for model learning and control, the reader is referred to [180].

The extent of flexibility inherent in these models can, however, lead to use in a black-box fashion, disregarding important structural knowledge in the underlying dynamics [135, 137, 138, 173, 212]. This can result in unnecessarily high-dimensional learning problems, slow convergence rates and often necessitates large training corpora, typically to be collected offline. In the extreme, the latter requirement can cause slow prediction and conditioning times. Moreover, they have been used in combination with computationally intensive planning methods such as dynamic programming [78,79,212] rendering online learning and tracking difficult.

In contrast to this body of work, we incorporate structural *a priori* knowledge of the dynamics, afforded by Lagrangian mechanics (without sacrificing the flexibility of nonparametrics). This requires, in some instances, a (partial) departure from Gaussianity but improves the detail with which the system is identified and can reduce the dimensionality of the identification problem. Furthermore, our method uses uncertainties inherent in the models to achieve active training example incorporation and decision making. As well learning, our method employs (partial) feedback-linearisation [233] in an outer-loop control law to reduce the complexity of the control problem. Thereby, the nominal problem is reduced to controlling a double-integrator via an inner-loop control law which can be set to any desired reference acceleration. If we combine the outer-loop controller with an inner-loop controller that has desirable guarantees (e.g. stability) for the double-integrator,

these properties can extend to the expected given non-linear closed-loop dynamics. The resulting approach enables rapid decision making and can be deployed in online learning and control.

Our approach can be seen as a generalisation of GP-MRAC [66]. In this paper, which is discussed in greater detail in Sec. 4.4.2, the authours utilise a Gaussian process on joint state-control space to learn the error of an inversion controller in model-reference adaptive control. The paper contains a theorem on stability based on several restrictive assumptions, including the assumption that the solution trajectory of the GP-driven model could be stated as a Markov process with orthogonal increments represented as an Ito-SDE of time. In addition, their method requires knowledge of the invertibility of the adaptive element. Translated to control-affine systems that means that the control input vector field has to be known and the uncertainty only is in the drift. In contrast, we consider actuated and underactuated systems where both the drift and control input matrix can be unknown a priori. Our method is capable of identifying the drift and control input vector fields constituting the underlying control-affine system individually, yielding a more fine-grained identification result. While this benefit requires the introduction of separating signals to the control during online learning, each of the coupled learning problems has state space dimensionality only. Moreover, our method is not limited to Gaussian processes. If the control-input vector fields are identified with a log-normal process, our controller will automatically be cautious in scarcely explored regions.

The remainder of this chapter is structured as follows: In Sec. 3.2, we describe assumptions on the underlying dynamics and the idea of partial feedback linearisation for control. Our exposition focusses on control-affine systems of second order. It should be noted however, that it also is applicable in dynamic systems of general order and non-control-affine systems whose control input vector fields have known nullspace.

Sec. 3.3.2 describes the online data generation and learning procedure of the system identification module. Utilising the concepts of *partial feedback-linearisation (PFL)*, Sec. 3.3.3 develops a control law that makes control decisions on the basis of the learned system model. In Sec. 3.4, we consider discrete-time systems and discuss conditions for which we can (and for which we cannot) prove closed-loop stability of the expected trajectory.

Sec. 3.5 contains simulations that illustrate our approach in the context of simple mechanical systems. We start considering the task of learning a single torque-actuated pendulum which we use to illustrate several properties of both our method in particular and properties common to Bayesian non-parametric methods in general. Before giving an illustration of collocated PFL in underactuated cartpole system, we examine the control of a torque-actuated undamped double pendulum. We show that our method manages to learn and control this system rapidly online.

3.2. Continuous dynamics and (partial) feedback-linearisation

Let $I \subseteq \mathbb{R}$ be a set of times, $\mathcal{Q} \subseteq \mathbb{R}^n$ denote the configuration space, $\mathcal{X} \subseteq \mathbb{R}^d$ the state space and $\mathcal{U} \subseteq \mathbb{R}^m$ the control space.

Via the principle of least action and the resulting Euler-Lagrange equation, Lagrangian mechanics constrains a large variety of mechanical systems to be *control-affine*. That is, the configuration trajectory satisfies the equation

$$\ddot{q} = f(q, \dot{q}) + g(q)u. \tag{3.1}$$

where $q \in \mathcal{Q}$ is a generalized coordinate of the configuration and $u \in \mathcal{U}$ is the control input and f is the *drift* vector field and g the *control input* vector field or matrix.

For example, in the cart-pole control domain we consider below, q will encode the cart position and pole angle, whereas u is an acceleration signal to the cart.

Defining $x_1 := q$, $x_2 := \dot{q} \in \mathcal{Q}$, we can write the state as $x := [x_1, x_2]$. The dynamics can be restated as the system of equations

$$\dot{x}_1 = x_2 \tag{3.2}$$

$$\dot{x}_2 = f(x_1, x_2) + g(x_1, x_2)u \tag{3.3}$$

$$= f(x_1, x_2) + \sum_{j=1}^{m} u_j \, g_j(x_1, x_2) \tag{3.4}$$

where $m = \dim \mathcal{U}$ and $g_j(x_1, x_2)$ is the jth row of matrix $g(x_1, x_2) \in \mathbb{R}^{n \times m}$.

Note, in many rigid body systems, we have

$$f(q, \dot{q}) = H^{-1}(q)\Big(-C(q, \dot{q})\dot{q} - G(q)\Big) \tag{3.5}$$

where $H(q)$ denotes the positive definite and symmetric inertial matrix and $C(q, \dot{q}), G(q)$ include a description of environmental forces such as friction and gravitational terms [248]. Furthermore, in this case, $g(q)$ is a diagonal matrix whose diagonal is given by the vector $H^{-1}(q)B(q)$. Here, vector $B(q)$ describes which configuration components the control can influence.

For instance, we might have $g(q)u = H^{-1}(q)\,[u_1; 0]$ where H is diagonal u_1 is the part of the control action vector u we can set to influence the dynamics. In this case, if we split up the configuration vector $q = [q_1; q_2]$ with $\dim q_1 = \dim u_1$ then q_1, is called the actuated part (i.e. the one we can manipulate directly) and q_2 the non-actuated part of the configuration.

In a *fully-actuated* system, the control can influence each configuration acceleration independently. Mathematically, this means that $g(q)$ is invertible and $\dim u_1 = \dim q$. Or, in the rigid body system, that vector

B comprises non-zero components only. In this case, we can employ *feedback-linearisation* to reduce the control problem to a double-integrator problem. That is, setting the control to $g^{-1}(q)[-f(q,\dot{q}) + u']$ yields closed-loop dynamics $\ddot{q} = u'$. Finding pseudo-control u' driving the state to a desirable goal state ξ is easy. For instance, it can be achieved by the linear control law $u'(t, [q;\dot{q}]) = -K(t)\,[q;\dot{q}]$ where $K(t)$ is a pos. def. matrix.

However, in many mechanical systems, only parts of the configuration space are actuated. In such *underactuated* systems, the row rank of $B(q)$ (and thus, of $g(q)$) is strictly less than the configuration space dimensionality n. While it is no longer possible to reduce all configuration component dynamics to a double-integrator problem, *partial feedback linearisation* (PFL) [233] can be employed to feedback-linearise at least rank (g) of the components. Depending on whether the actuated components or non-actuated ones are linearised, the method is called *collocated* or *non-collocated* linearisation. We will include a brief description of both approaches.

In summary, we will see that whether the part of configuration q_1 to be feedback-linearised is non-actuated or actuated and whether we have an underactuated system or not, we can express the acceleration in control-affine form:

$$\ddot{q}_1 = a(q,\dot{q}) + b(q)u \tag{3.6}$$

for some suitable choices of partial drift a and matrix b. The definitions of a and b depend on whether q_1 is actuated or non-actuated. Furthermore, we have also discussed the fully actuated case where $q = q_1$, $a = f$ and $g = b$ was invertible.

In any of these cases we can then attempt to feedback-linearise by setting the control to

$$u(t, [q,\dot{q}]; u') := b^{\dagger}(q)[-a(q,\dot{q}) + u'] \tag{3.7}$$

where $b^{\dagger}$ denotes the Moore-Penrose pseudoinverse. If b is invertible, $b^{\dagger} = b^{-1}$. This yields the simple closed-loop partial dynamics $\ddot{q}_1 = u'$ as desired. Free parameter u' is the *pseudo-* or *inner-loop control* and can be set at will to control q_1 in some desired manner.

Next, we will briefly go over the definitions of a and b for rigid body systems that are frequently found in robotics [248].

Collocated feedback linearisation. Let q_1 be the active sub-configuration, i.e. the projection of q onto the actuated sub-space of configuration space $\mathcal{Q}$. In non-collocated linearisation, we desire a control such that $\ddot{q}_1 = u'$ where u' is a desired reference acceleration of actuated configuration q_1. Without loss of generality, let $q = [q_1; q_2]$, q_1 be the actuated part and q_2 the non-actuated part of the configuration. For notational con-

venience, define $\phi(q,\dot{q}) := C(q,\dot{q})\dot{q}+G(q) = [\phi_1(q,\dot{q});\phi_2(q,\dot{q})]$ and write $H(q) = \begin{pmatrix} H_{11}(q) & H_{12}(q) \\ H_{21}(q) & H_{22}(q) \end{pmatrix}$ where the $H_{ii}(q)$ are pos. def. $n \times n$ block matrices. Finally, let $Q(q) := H_{11}(q) - H_{12}(q)\, H_{22}^{-1}(q)\, H_{21}(q)$ which is invertible [248]. With these definitions, rearranging Eq. 3.1, we find

$$\ddot{q}_1 = a(q,\dot{q}) + b(q)u \tag{3.8}$$

where

$$b(q) = Q^{-1}(q) \tag{3.9}$$

and the *drift* (of the actuated sub-configuration) is:

$$a(q,\dot{q}) = Q^{-1}(q)\Big(-\phi_1(q,\dot{q}) + H_{12}H_{22}^{-1}\phi_2(q,\dot{q})\Big). \tag{3.10}$$

We can linearise this part of configuration space by setting

$$u(t,[q,\dot{q}];u') := b^{-1}(q)[-a(q,\dot{q}) + u']$$

yielding the linearised partial dynamics $\ddot{q}_1 = u'$ as desired.

Non-collocated partial feedback linearisation. This time, let q_1 be the *passive* sub-configuration, i.e. the projection of q onto the non-actuated sub-space of configuration space $\mathcal{Q}$. And, let $q = [q_1; q_2]$, where q_1 is the non-actuated part and q_2 the actuated part of the configuration. In non-collocated linearisation, we desire a control such that $\ddot{q}_1 = u'$ where u' is a desired reference acceleration of non-actuated configuration q_1. Again, for notational convenience, let $\phi(q,\dot{q}) := C(q,\dot{q})\dot{q} + G(q) = [\phi_1(q,\dot{q});\phi_2(q,\dot{q})]$ and write $H(q) = \begin{pmatrix} H_{11}(q) & H_{12}(q) \\ H_{21}(q) & H_{22}(q) \end{pmatrix}$ where the $H_{ii}(q)$ are pos. def. $n \times n$ block matrices. While H is pos. def. the off diagonal block matrices may not be and may have to be inverted with a pseudo-inverse.

Finally, let $R(q) := H_{21}(q) - H_{22}(q)\, H_{12}^{\dagger}(q)\, H_{11}(q)$ where $H_{12}^{\dagger}(q)$ denotes the pseudo-inverse of block matrix $H_{12}(q)$. The latter provides a unique solution if rank $H_{12} = \dim q_1$. With these definitions, rearranging Eq. 3.1, we find

$$R(q)\ddot{q}_1 = \tilde{a}(q,\dot{q}) + u \tag{3.11}$$

where

$$\tilde{a}(q,\dot{q}) = \Big(H_{22}H_{12}^{\dagger}\phi_1(q,\dot{q}) - \phi_2(q,\dot{q})\Big). \tag{3.12}$$

Similar to the above, we can attempt to linearise this part of configuration space by setting

$$u(t, [q, \dot{q}]; u') := R(q)u' - \tilde{a}(q, \dot{q}).$$

To make this more consistent with our previous exposition we could also write

$$\ddot{q}_1 = a(q, \dot{q}) + b(q)u \tag{3.13}$$

where this time,

$$b(q) = R^{\dagger}(q) \tag{3.14}$$

and the *drift* (of the non-actuated sub-configuration) is

$$a(q, \dot{q}) = R^{\dagger}(q)\Big(H_{22}H_{12}^{\dagger}\phi_1(q, \dot{q}) - \phi_2(q, \dot{q})\Big). \tag{3.15}$$

With these definition, setting the control as per Eq. 3.7 yields the linearised partial dynamics $\ddot{q}_1 = u'$ as desired whenever inversion with the pseudo-inverse succeeds.

For more background on PFL and its generalisation, *task space linearisation*, the reader is referred to [248].

3.3. Learning and adaptive inversion control

3.3.1. Learning an uncertain drift vector field

Before considering the more general case where both b and the drift a are uncertain, we will start with the case where b is known a priori.

Epistemic uncertainty and learning. While partial feedback linearisation (PFL) is known to be a proficient tool in simplifying the control problem of q_1, it requires accurate knowledge of the dynamics. While knowledge of the inertial matrix (and hence, of b) usually entirely depends on the plant (which we can often model accurately), drift field a also depends on external forces (e.g. state-varying friction, wind, gravitational forces due to the slope of the terrain, e.t.c.) that may be subject to change or are unknown to the system designer a priori.

Therefore, we assume partial drift vector field a can be uncertain a priori. That is, a priori our uncertainty is modelled by the assumption that $a \sim \Pi^a$ where Π^a is a random field. Here the randomness is not some form of an "inherent stochasticity" property of the actual dynamics but an encoding of the subjective beliefs of the learner what the deterministic vector field a is. That is, the random field reflects an epistemic uncertainty

about the true underlying (deterministic) dynamics. Consequently, Eq. 3.8 gives rise to a random differential equation encoding our belief over the correct equations of motion.

If data (in the form of acceleration estimates) become available over the course of the state evolution, we can update our beliefs in a Bayesian fashion. That is, at time $t \in I$ we assume $a \sim \Pi^a|\mathcal{D}_t$ where $\mathcal{D}_t$ is the data recorded up to time t. The process of conditioning is often referred to as (Bayesian) *learning*.

Data collection. Let the full state be denoted by $x = [q, \dot{q}]$. We assume our controller can be called at an ordered set of times $I_u \subset I$. At each time $t \in I_u$, the controller is able to obtain a (potentially noisy) observation the state $x(t)$ and to set the control input $u_t = u(t, x_t)$. The controller may choose to evoke learning at an ordered subset $I_\lambda \subset I_u$ of times. To this end, at each time $\tau \in I_\lambda$, the controller evokes a procedure explicated in Sec. 3.3.2 if it decides to incorporate an additional data point into data set $\mathcal{D}_t$ $(t > \tau)$. Each data point is a record of the estimate of a state and the estimated drift at the recorded time. The decision on whether to update the data will be based on the belief over the data point's anticipated informativeness as approximated by its variance.[1]

We assume that learning can occur every Δ_λ seconds and the controller is called every $\Delta_u \leq \Delta_\lambda$ seconds. A continuous control takes place in the limit of infinitesimally small Δ_u. For simplicity, we assume the uncertain dynamics are well approximated (in the mean) by an Euler discretisation. And, we may assume that we can observe configurations, derivatives and accelerations. Of course, if we do not have physical means to measure velocities and accelerations, obtaining numerical estimates becomes necessary based on observations of configurations q. To estimate derivatives, we chose a first-order method. That is, our state derivative estimates are $\dot{x}(t_i) \approx \frac{x(t+\Delta_o)-x(t)}{\Delta_0}$ where Δ_o is a period length with which we can observe states. In this work, we assume $\Delta_o = \Delta_u$. Alternatively, we can obtain second-order numerical estimates as per $\dot{x}(t_i + \Delta_o) := \frac{x(t_i+2\Delta_o)-x(t_i)}{2\Delta_0}$.

Assuming online learning, the data sets $\mathcal{D}_t$ are found incrementally as follows: Assume we are at time $t \in I_\lambda$ in configuration q and that we decide to learn about a. This decision is made, whenever our uncertainty about $a_t := a\big(x(t)\big)$, encoded by $\mathrm{var}[a\big(x(t)\big)]$, is above a certain threshold θ^a_{var}. Next, we obtain (by measurement or computation) an estimate $\ddot{q}'_1 = \ddot{q}_1 + \nu$ of $\ddot{q}_1$ where ν represents sensor noise or numerical error. When working with Gaussian processes it is convenient to assume $\nu \sim \mathcal{N}(0, \sigma_o^2)$. Leveraging Eq. 3.8, we obtain a (noisy) estimate $\tilde{a}$ of drift a as per $\tilde{a} = \ddot{q}'_1 - b(q)u(t)$.

We then incorporate data point $(t, x, \tilde{a})$ into $\mathcal{D}_{t+\Delta_u} = \mathcal{D}_t \cup \{(t, x, \tilde{a})\}$ and retrain our probabilistic model over a by computing the posterior $\Pi^a|\mathcal{D}_{t+\Delta_u}$ available for predicting in subsequent time steps.

[1] Variance is known to approximate entropic measures of uncertainty (cf. [4]) and often easier to compute than entropy.

3.3.2. Learning under complete uncertainty

Above we have considered the case where the drift vector field was uncertain but the control input vector field was completely known. This allowed us to determine the drift value $a(x)$ based on knowing the control u having a (possibly noisy) acceleration estimate $\ddot{q}$. In this subsection, we will consider the more general case where also b has to be learned. To make this possible we will set up a cascade of learners, one for each of the vector fields. In order to separate the contribution of $b(x)u$ and $a(x)$ to an acceleration measure, we can attempt to eliminate the influence of b in order to learn about a (at least when b is uncertain). Fortunately, in a control-affine system this can be done by setting the control input signal to zero.[2] Conversely, learning about $b(x)$ can only be done successfully in states x where $a(x)$ is relatively certain. This suggests an interleaved online learning approach which we will describe next.

Epistemic uncertainty and learning. Both dynamics functions a and b can be uncertain a priori. That is, a priori our uncertainty is modelled by the assumption that $a \sim \Pi^a, b \sim \Pi^b$ where Π^a, Π^b are random fields. The processes reflect our epistemic uncertainty about the true underlying (deterministic) dynamics functions a and b. That is, all probabilities have to be interpreted in a Bayesian manner.

If data becomes available over the course of the state evolution, we can update our beliefs over the dynamics in a Bayesian fashion. That is, at time $t \in I$ we assume $a \sim \Pi^a|\mathcal{D}_t, b \sim \Pi^b|\mathcal{D}_t$ where $\mathcal{D}_t$ is the data recorded up to time t.

Data collection. We assume our controller can be called at an ordered set of times $I_u \subset I$. At each time $t \in I_u$, the controller is able to observe the state $x_t = x(t)$ [3] and to set the control input $u_t = u(t, x_t)$. The controller may choose to evoke learning at an ordered subset $I_\lambda \subset I_u$ of times. To this end, at each time $\tau \in I_\lambda$, the controller evokes a procedure explicated in Sec. 3.3.2 if it decides to incorporate an additional data point (t, x_t, u_t) into data set $\mathcal{D}_t$ $(t > \tau)$. The decision on whether to update the data will be based on the belief over the data point's anticipated informativeness as approximated by the variance (of the pertaining random vector the data point would instantiate).[4]

For simplicity, we assume that learning can occur every Δ_λ seconds and the controller is called every $\Delta_u \leq \Delta_\lambda$ seconds. A continuous control takes place in the limit of infinitesimal Δ_u.

Learning procedure

As before, the data sets $\mathcal{D}_t$ are found incrementally in an online learning mode. Since it is hard to use the data to infer a and b simultaneously, we will have to actively decide which one we desire to learn about (and set the control accordingly – which we will henceforth refer to as a *separating control*). To this end, we distinguish

[2] An alternative approach is suggested in the future work part at the end of this chapter.
[3] In fact, we can only observe q and have to obtain noisy observations of $\dot{q}$ as we will describe below.
[4] Variance is known to approximate entropic measures of uncertainty (cf. [4]) and often easier to compute than entropy.

between the following learning components:

- Learning $a(x)$: Assume we are at time $t \in I_\lambda$ and that we decide to learn about a. This decision is made, whenever our uncertainty about $a_t := a\big(x(t)\big)$, encoded by $\text{var}[a\big(x(t)\big)]$, is above a certain threshold θ^a_{var}. We assume derivatives are estimated with second-order numerical approximations. Therefore, when learning is initiated, we keep the control constant for two more time steps $t + \Delta_u$, $t + 2\Delta_u$ to obtain a good derivative estimate as described above. To remove additional uncertainty due to ignorance about b, we set separating control $u_t = u_{t+\Delta_u} = u_{t+2\Delta_u} = 0$ yielding dynamics $\dot{x}_2 = a(x)$ during time interval $[t, t+2\Delta_u)$. On the basis of a noisy estimate $\ddot{Q}_1(t+\Delta_u)$ of $\ddot{q}_1(t+\Delta_u)$, we can determine a noisy estimate $\tilde{a}_{t+\Delta_u}$ of unknown function value $a_{t+\Delta_u}$ for time $t + \Delta_u$ as per

$$\tilde{a}_{t+\Delta_u} = \ddot{Q}_1(t + \Delta_u).$$

  So, $(t + \Delta_u, \tilde{a}_{t+\Delta_u})$ is added to the data after time $t + 2\Delta_u$.

- Learning $b_j(x)$: At time $t \in I_\lambda$, we choose to learn about function b_j whenever our uncertainty about $a(x(t_i))$ is sufficiently small (i.e. $\text{var}[a(x_i)] \leq \theta^a_{\text{var}}$) and our uncertainty about b_j is sufficiently large ($\text{var}[b_j(x_{t_i})] > \theta^b_{\text{var}}$). When learning is initiated, we keep the control constant for two more time steps $t + \Delta_u$, $t + 2\Delta_u$ to obtain a good derivative estimate as described above.

  Let $e_j \in \mathbb{R}^m$ be the jth unit vector. To learn about $b_j(x)$ at state x, we apply a control action $u = u_j e_j$ where $u_j \in \mathbb{R}\backslash\{0\}$. Inspecting Eq. 3.6 we can then see that $b_j(x) = \frac{\ddot{q}_1 - a(x)}{u_j}$.

  Therefore, after time $t+2\Delta_u$ on the basis of (noisy) observation $\ddot{Q}(t+\Delta_u)$, we calculate $\tilde{b}_j\big(x(t + \Delta_u)\big) = \frac{\ddot{Q}_1(t+\Delta_u) - \tilde{a}\big(x(t+\Delta_u)\big)}{u_j(t+\Delta_u)}$ and add training point $\Big(x_{t+\Delta_u}, \tilde{b}_j(x_{t+\Delta_u})\Big)$ to the data set. Here $\tilde{b}_j(x_{t+\Delta_u}) =$. The additional variance (as per Eq. 3.17) is captured by setting observational noise levels for Π^b accordingly.

  Note, since $a(x)$ (and potentially, due to observational noise $\ddot{q}_1$) will generally be a random variable, so is $b_j(x)$. It has the mean

$$\langle b_j(x) \rangle = \frac{\langle \ddot{q}_1 \rangle - \langle a(x) \rangle}{u_j} \tag{3.16}$$

  and variance

$$\text{var}[b_j(x)] = \frac{\text{var}[\ddot{q}_1] + \text{var}[a(x)]}{u_j^2} \leq \frac{\text{var}[\ddot{q}_1] + \theta^a_{\text{var}}}{u_j^2}. \tag{3.17}$$

3.3.3. Control law

Unless the control actions are chosen to aid system identification (as described above), we will want to base our control on our probabilistic belief model over the dynamics. Given such an uncertain model, it remains to

define a control policy u with desirable properties. In this chapter, we combine the expected posterior beliefs over the dynamics with the feedback-linearising control laws of Sec. 3.2 in a certainty equivalent manner.

That is, depending on the situation at hand (e.g. whether we have an underactuated or fully-actuated system) we apply the corresponding linearising control laws substituting the posterior expectations $\langle a(x(t))|\mathcal{D}_t\rangle$ and $\langle b(x(t))|\mathcal{D}_t\rangle$ in place of the actual, uncertain functions a and b.

With reference to Eq. 3.7, this translates to an inversion control law:

$$u(t, [q, \dot{q}]; u') := \langle b^\dagger(q)|\mathcal{D}_t\rangle[-\langle a(q, \dot{q})|\mathcal{D}_t\rangle + u']. \tag{3.18}$$

In combination with the control actions injected to identify the system, we name this controller RF-SIIC (where *RF* stands for *random field* and *SIIC* for *system identification* and *inversion control*).

In case the expectations coincide with the ground truth dynamics (and if the pseudo-inverse coincides with the inverse) this controller yields the closed-loop partial dynamics $\ddot{q}_1 = u'$ as desired. Free parameter u' is the pseudo- or inner-loop control and can be set at will to steer to control q_1 in some desired manner. For instance, we can set it to some reference dynamics which are to be followed. A typical choice is the *linear control law*

$$u'(t, x; \xi, w) := w_1(\xi - q_1) + w_2(\dot{\xi} - \dot{q}_1) \tag{3.19}$$

causing q_1 to converge to target configuration ξ if feedback *gain* weights w_1, w_2 are chosen to be positive.

3.4. Discrete-time learning and control with an uncertain drift field

In this section, we consider a time-discretised version of a second-order system assuming the control input vector field b is known and can be inverted. With notation as before, assume $n = m, d = m + n$.

Let $q \in \mathbb{R}^m$ be a configuration and $x := [q, \dot{q}] \in \mathcal{X} = \mathbb{R}^d$ be the full state and as before $u \in \mathcal{U}$ denotes the control decision that has to lie in control space $\mathcal{U} = \mathbb{R}^m$. We assume the dynamics are given by the second-order equation

$$\ddot{q} = a(x) + b(x, u)$$

where $b : \mathbb{R}^d \times \mathcal{U} \to \mathbb{R}^m$ is a known, deterministic function that is control invertible, i.e. where we know there exists a function $b^- : \mathbb{R}^d \times \mathbb{R}^m \to \mathbb{R}^m$ such that $b\big(x, b^-(x, u')\big) = u', \forall x \in \mathcal{X}, u' \in \mathbb{R}^m$. Examples are fully-actuated, control-affine systems with invertible control input matrices as considered above. That is, where $b(x, u) = B_x u$ and $B_x \in \mathbb{R}^{m\times m}$ is invertible. Let $I_m \in \mathbb{R}^{m\times m}$ be the identity matrix, $O_m = 0I_m$ a matrix of zeros and $o_m \in \mathbb{R}^m$ denote the m-dimensional vector of zeros.

We can rewrite our dynamics as the first-order system:

$$\dot{x} = \begin{pmatrix} O_m & I_m \\ O_m & O_m \end{pmatrix} x + f(x) + g(x,u). \tag{3.20}$$

where $f(x) = \big(o_m, a(x)\big)^\top$ and $g(x) = \big(o_m, b(x,u)\big)^\top$.

An Euler-approximated, time-discrete version is:

$$x_{k+1} = E\, x_k + \Delta\, f(x_k) + \Delta\, g(x_k, u_k) \tag{3.21}$$

where Δ is a time-increment and $E = \begin{pmatrix} I_m & \Delta\, I_m \\ O_m & I_m \end{pmatrix}$. This is the system we desire to stabilise. That is we will design a control law $u : \mathcal{X} \to \mathcal{U}$ that drives the state towards goal state 0. This is without loss of generality, since the case of tracking a reference is a trivial extension of stabilisation at zero and therefore, the task of stabilising 0 is the canonical case most often considered in control. For further explanations of this point refer to Ch. 4.

Since we will assume f to be uncertain, we can only do so relative to our beliefs over f. Encoding our epistemic beliefs in terms of probabilities, any convergence guarantee on reaching the goal state will therefore have to be of a probabilistic nature. Before deriving a controller with such guarantees, we will next outline how to update one's prior beliefs in the light of data.

3.4.1. Bayesian non-parametric drift learning

Assuming we know the dynamics are given by Eq. 3.21 but are uncertain about drift vector field $f = (o_m, a)^\top$. The first step in Bayesian non-parametric learning is to model the uncertainty by assuming the drift is drawn from a prior process $f \sim \Pi^f = (o_m, \Pi^a)^\top$. The notation $\Pi^f = (o_m, \Pi^a)$ indicates that

$\forall x \in \mathcal{X}$, measurable sets $S = S_1 \times \ldots \times S_{2m} \subset \mathbb{R}^{2m}$:

$$\Pr_{\Pi^f}[f(x) \in S] = \begin{cases} 0, \exists i \leq m : S_i \neq \{0\} \\ \Pr_{\Pi^a}[f_{m+1}(x) \times \ldots \times f_{2m}(x) \in S_{m+1} \times \ldots \times S_{2m}], \text{ otherwise.} \end{cases} \tag{3.22}$$

Observing a sequence of (state, control, successor-state) triples (x_i, u_i, x_{i+1}) allows one to compute a sequence $\mathcal{D} = \{(x_i, f_i)\}$ where $f_i := f(x_i) = \frac{1}{\Delta}(x_{i+1} - E\, x_i) + g(x_i, u_i)$. Learning consists of computing the posterior belief process $\Pi^f|\mathcal{D} = (o_m, \Pi^a|\mathcal{D})^\top$.

3.4.2. Inversion control law

Assume we are at time step $k \in \mathbb{N}$ having collected data $\mathcal{D}_k$. Based on our posterior belief $\Pi^f|\mathcal{D}_k = (o_m, \Pi^a|\mathcal{D}_k)^\top$ we define an inversion- control law as follows:

$$u(x_k, u'_k) = b^-(x_k, -\mathfrak{m}_k + u'_k) \tag{3.23}$$

where

$$\mathfrak{m}_k = \langle a(x_k)|\mathcal{D}_k, x_k\rangle \tag{3.24}$$

is the expected value of the drift computed with respect to the posterior $\Pi^a|\mathcal{D}_k$ and u'_k is referred to as the *pseudo-control*.

The closed-loop dynamics degenerate to

$$x_{k+1} = E\, x_k + \Delta\, F_k + \Delta\, (O_m, I_m)^\top u'_k \tag{3.25}$$

where $(F_k)_{k\in\mathbb{N}}$, with $F_k = f(x_k) - (o_m, \mathfrak{m}_k)^\top$ is a random field. *Linear-feedback pseudo-control.* Let $K \in \mathbb{R}^{m\times d}$ be a feedback gain matrix with positive definite sub-matrices $K_1, K_2 \in \mathbb{R}^{m\times m}$, $K = [K_1, K_2]$. We assume the objective is to drive the state to goal state $\xi = 0$. We set the *pseudo control* to a linear feedback, $u'_k := -Kx_k$, the control law becomes

$$u(x; K) = b^-(x, -\mathfrak{m}_k - Kx) \tag{3.26}$$

yielding the closed-loop dynamics

$$x_{k+1} = Mx_k + \Delta\, F_k \tag{3.27}$$

where

$$M = \begin{pmatrix} I_m & \Delta\, I_m \\ -\Delta K_1 & I_m - \Delta K_2 \end{pmatrix}. \tag{3.28}$$

Here we normally devise K such that $\|\!|M|\!\|_2 \geq 1$ but $\rho(M) < 1$, in which case M is a *stable* or *Hurwitz* matrix.

3.4.3. Convergence guarantee for the expected trajectory

With this setup we can guarantee convergence of the expected state trajectory to the goal state $\xi = 0$.

Theorem 3.4.1. *Assume the closed-loop dynamics are given by Eq. 3.27 with known matrix M and time increment Δ. Moreover, assume the learner is capable of keeping the training data $\mathcal{D}_k$ up to date, containing the entire history of states up to time k. That is, $\mathcal{D}_k = \{x_0, ..., x_k\}$. Then, $\langle x_k \rangle = M^k \langle x_0 \rangle$. Stability of M ($\rho(M) < 1$) implies stability of the expected trajectory $\big(\langle x_k \rangle\big)_{k \in \mathbb{N}_0}$. That is, $\lim_{k\to\infty} \|\langle x_k \rangle\| = 0$.*

Proof. Since $x_{k+1} = Mx_k + \Delta\, F_k$ we have $\langle x_{k+1} \rangle = M\langle x_k \rangle + \Delta \langle F_k \rangle$.

Showing $\langle F_k \rangle = 0$ would allow us to conclude $\langle x_{k+1} \rangle = M\langle x_k \rangle$ and thus, we would have shown $\langle x_{k+1} \rangle = M^k \langle x_0 \rangle$. M being stable implies that the recurrence converges to zero as desired.

So, it remains to be shown that we have $\langle F_k \rangle = 0$: By definition of F_k, it suffices to show that $\langle a_k - \mathfrak{m}_k \rangle = 0$. Let $\mathcal{F}_k = \{F_0, \ldots, F_{k-1}\}$ denote the history of random increments. By the law of iterated expectations, we have $\langle a_k - \mathfrak{m}_k \rangle = \langle \langle a_k - \mathfrak{m}_k | \mathcal{F}_k \rangle_{a_k} \rangle_{\mathcal{F}_k} = \langle \langle a_k - \mathfrak{m}_k | \mathcal{F}_k \rangle_{a_k} \rangle_{\mathcal{F}_k}$ (here the subscripts next to the expectation brackets indicate which variables the expectations are taken over). We will show that the inner expectation $\langle a_k - \mathfrak{m}_k | \mathcal{F}_k \rangle_{a_k} = \langle a_k | \mathcal{F}_k \rangle_{a_k} - \langle \mathfrak{m}_k | \mathcal{F}_k \rangle_{a_k}$ is zero. By knowing $\mathcal{F}_k$, the state history $x_0, ..., x_k$ is deterministically determined via the recurrence of Eq. 3.27 (and vice versa).

Hence,

$$\langle \cdot | \mathcal{F}_k \rangle = \langle \cdot | \{x_0, ..., x_k, F_0, ..., F_{k-1}\} \rangle \tag{3.29}$$

$$= \langle \cdot | \{x_0, ..., x_k, a_0, ..., a_{k-1}\} \rangle \tag{3.30}$$

$$= \langle \cdot | \mathcal{D}_k \rangle \tag{3.31}$$

where the last step follows by our assumption of $\mathcal{D}_k = \{x_0, ..., x_k\}$ whose knowledge allows the reconstruction of the $a_0, ..., a_{k-1}$. Thus, $\langle a_k | \mathcal{F}_k \rangle_{a_k} - \langle \mathfrak{m}_k | \mathcal{F}_k \rangle_{a_k} = \langle a_k | \mathcal{D}_k \rangle_{a_k} - \mathfrak{m}_k \overset{Eq.3.24}{=} \mathfrak{m}_k - \mathfrak{m}_k = 0$.

□

The theorem tells us that, when we can keep our data always up-to-date, the control actions as per Eq. 3.26 guarantee that our subjective (i.e. Bayesian) expectation over the controlled closed-loop trajectory succeeds in converging to the goal state $\xi = 0$ as desired. Investigating stronger notions such as mean-square stability will have to be deferred to future work.

Remark 3.4.2. At a cursory glance, our result that the expected trajectory is stable may not seem surprising since we always subtract the mean. However, it should be emphasized that for establishing this result, the assumption that the data is always kept up to date proved key (cf. Eq. 3.31). Of course, for fast sampling rates this is unrealistic and the mean $\mathfrak{m}_k$ we subtract will be conditional on a subset of the actual history. That is, $\mathcal{D}_k \subsetneq \mathcal{F}_k$. In this case, one might attempt to establish the desired result by marginalising over the unobserved elements of the increment history. In the standard theory of Ito stochastic differential equations, the orthogonality of the increments a_i would make this approach successful in establishing the desired result. Unfortunately though, in our situation, the uncertainty arises from the draw $a \sim \Pi^a$ where Π^a may introduce

strong correlations. Developing a better understanding of the impact of these correlations on the expected closed-loop trajectory is something we consider an open problem and subject to work in progress.

3.5. Simulations

In this section, we illustrate our method by applying to online identification of simulated control-affine systems. In Sec. 3.5.1, we begin with basic fully actuated systems where full feedback-linearisation is possible but where both drift and control input vector fields are unknown a priori. To give a simple example of an underactuated system, in Sec. 3.5.2, we apply partial feedback-linearisation to a cart-pole with uncertain drift. Learning is done as described above where acceleration estimates were obtained numerically from state measurements obtained every Δ_u seconds.

3.5.1. Fully actuated pendula with completely unknown dynamics

To simulate a discrete 0th order sample-and-hold controller in a continuous environment, we simulated the dynamics between two consecutive controller calls (occurring every Δ_u seconds) employing standard integration techniques.

Before learning, we modelled our prior beliefs over the drift and control input vector fields by assuming $a \sim \mathcal{GP}(0, K_a)$ and $b \sim \log \mathcal{GP}(0, K_b)$ had been drawn from a normal and log-normal process, respectively. The latter assumption encodes a priori knowledge that control input function b can only assume positive values (but, to demonstrate the idea of cascading processes, we had discarded the information that b was a constant). During learning, the latter process was based on a standard normal process conditioned on $\log$-observations of $\tilde{b}$. To compute the control law, we need to convert the posterior mean over $\log b$ into the expected value over b. The required relationship is known to be as follows:

$$\langle b(x)|\mathcal{D}_t\rangle = \exp\Big(\langle \log b(x)|\mathcal{D}_t\rangle + \frac{1}{2}\text{var}[\log b(x)|\mathcal{D}_t]\Big). \tag{3.32}$$

If required, the posterior variance can be obtained as

$$\text{var}[b(x)|\mathcal{D}_t] = \Big((\exp 2\langle \log b(x)|\mathcal{D}_t\rangle) + \text{var}[\log b(x)|\mathcal{D}_t]\Big) \exp\Big(\text{var}[\log b(x)|\mathcal{D}_t] - 1\Big).$$

Note, the posterior mean over b increases with the variance of our normal process in log-space, and, the control law as per Eq. 3.18 is inversely proportional to the magnitude of this mean. Hence, the resulting controller is *cautious*, in the sense that control output magnitude is damped in regions of high uncertainty (variance). Depending on the situation, this can either be a curse or a blessing. If the system is stable under zero excitation the law has the advantage of gradually exploring the state space before moving to new

unexplored parts. On the other hand, if zero excitation leads to instability, this behaviour may of course be problematic or at least not help (for instance think of a UAV falling from the sky under zero control action). In this case, it might be recommended to couple the controller with a stabilising feedback controller in a hybrid control setup.

Single pendulum

We explored our method's properties in simulations of a rigid pendulum with (a priori unknown) drift $a(x) := -\frac{g}{l}\sin(x_1) - \frac{r(x_1)}{ml^2}x_2$ and constant input function $b(x) = \frac{1}{ml^2}$. Here, $x_1 = q, x_2 = \dot{q} \in \mathbb{R}$ are joint angle position and velocity, r denotes a friction coefficient, g is acceleration due to gravity l is the length and m the mass of the pendulum. The control input $u \in \mathbb{R}$ applies a torque to the joint that corresponds to joint-angle acceleration. The pendulum could be controlled by application of a torque u to its pivotal point. $q = 0$ encode the pendulum pointing downward and $q = 0$ denoted the position in which the pendulum is upward. Given an initial configuration $x_0 = [q_0, \dot{q}_0]$ we desired to steer the state to a terminal configuration $\xi = [q_f, 0]$.

To simulate a discrete 0th order sample-and-hold controller in a continuous environment, we simulated the dynamics between two consecutive controller calls (occurring every Δ_u seconds) employing standard integration techniques.

We illustrate the behaviour of our controllers in a sequence of four experiments. The parameter settings are provided in Tab. 3.1. Recorded control energies and errors (in comparison to continuous proportional controllers) are provided in Tab. 3.2.

Our Bayesian controller maintains an epistemic beliefs over the dynamics. These beliefs govern our control decisions (including those when to learn). Furthermore, to keep prediction times low, beliefs are only updated when the current variance indicated a sufficient of uncertainty. Therefore, one would expect to observe three properties of our controller:

(i) When the priors are chosen sensibly (could be indicated by the dynamic functions' likelihood under the probabilistic models), we expect good control performance.

(ii) Prior training improves control performance and, reduces learning, but is not necessary to reach the goal. Both properties can be observed in Exp. A and Exp. B.

(iii) When the controller is ignorant of the inaccuracy of its beliefs over the dynamics (i.e. the actual dynamics are unlikely but the variances are low), control may fail since the false beliefs are not updated. An example of this is provided in Exp. C.

(iv) We can overcome such problems practically, by employing the standard technique (see [201]) of optimising the prior hyper-parameters to maximise marginal likelihood. In Exp. D, this approach was success-

$Parameter(s):$	(l,r,m)	Δ_u	Δ_l	$(\theta^a_{var}, \theta^{\log b}_{var})$	x_0	ξ	(w_1, w_2)	t_f
Exp. A	(1,1,0.5)	.01	.5	(.001, .005)	(0,-2)	$(\pi, 0)$	(1,1)	20
Exp. B	(1,0.5,4)	.01	1	(.001, .005)	(0,-2)	$(\pi, 0)$	(2,2)	50
Exp. C	(1,0.5,4)	.01	1	(.001, .005)	(0,-2)	$(\pi, 0)$	(2,2)	15
Exp. D	(1,0.5,4)	.01	1	(.001, .005)	(0,-2)	$(\pi, 0)$	(2,2)	20

Table 3.1.: Parameter settings.

	$\int_I u^2_{adapt}(t)dt$				$\int_I (x(t)-\xi)^2 dt$				$(\left\|\mathcal{D}^a_{t_f}\right\|, \left\|\mathcal{D}^b_{t_f}\right\|)$	
$Controller:$	P1	P100	SP1	SP2	P1	P100	SP1	SP2	SP1	SP2
Exp. A	134	644	139	57	137	10	59	25	(18, 20)	(23, 53)
Exp. B	1981	11942	28808	2834	496	10	179	18	(37, 10)	(41, 10)
Exp. C	552	11942	14759	17029	139	10	82	72	(2,1)	(2,1)
Exp. D	730	11942	3753	1619	184	10	83	17	(12,2)	(12,2)

Table 3.2.: Cumulative control energies, squared errors and data sizes (rounded to integer values). P*k*: P-controller with feedback gain k. P1 failed to reach the goal state in all experiments. High-gain controller P100 succeeded in reaching the goal in all experiments but required a lot of energy. SP1: random field -based controller with empty data set to start with. SP2: reset SP1 with training data collected from the first run.

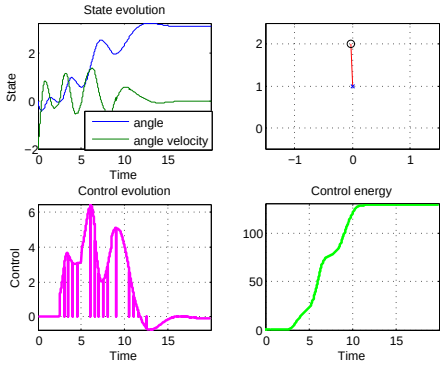
fully applied to the control problem of Exp. C.

Experiment A. We started with a zero-mean normal process prior over $a(\cdot)$ endowed with a rational quadratic kernel [201] with automated relevance detection (RQ-ARD). The kernel hyper-parameters were fixed. Observational noise variance was set to 0.01. The log-normal process over $b(\cdot)$ was implemented by placing a normal over $\log b(\cdot)$ with zero mean and RQ-ARD kernel with fixed hyper-parameters and observational noise level 0.1. Note, the latter was set higher to reflect the uncertainty due to Π^a. In the future, we will consider incorporating hetereoscedastic observational noise based on var$[a]$ and the sampling rate. Also, one could incorporate knowledge about periodicity in the kernel.

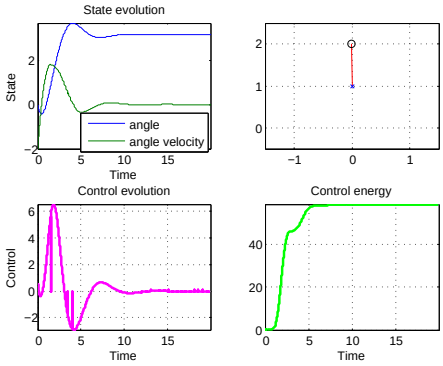
Results are depicted in Fig. 3.1 and 3.2. We see that the system was accurately identified by the random fields. When restarting the control task with random fields pre-trained from the first round, the task was solved with less learning, more swiftly and with less control energy.

Experiment B. Our control law was conceived under the assumption that the dynamic functions were drawn from processes coinciding with our priors. However, in practice Bayesian methods often suffer from the problem that a "good" prior often is hard to conceive. Since we consider online learning, the prior influences decision making from start. Therefore, we can expect that a poor prior might negatively affect our controller's performance. We will test our method under a prior that assigns low likelihood to our ground-truth dynamics.

The low length-scale parameters restrict generalisation. In combination with our observational noise-variances, we will expect our controller to be highly cautious and to slowly approaching the target (while

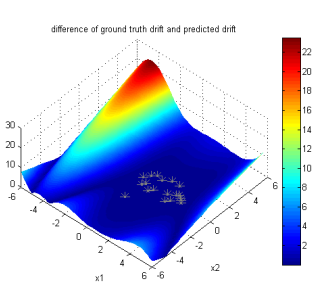


(a) Control with untrained prior.

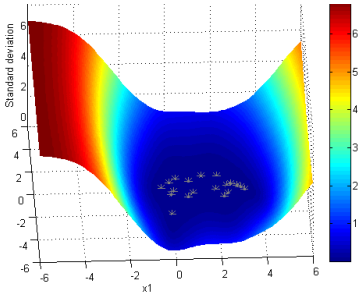


(b) Control evolution with trained prior from the first round.

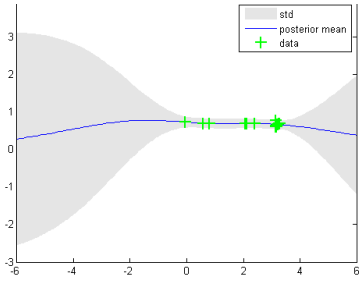
Figure 3.1.: Experiment A. Comparison of runs with untrained and pre-trained processes. The top-right image shows the final position of the pendulum having successfully reached the target angle $\xi_1 = \pi$. The dips in the control signal represent probing control actions arising during online learning.



(a) $\left|a(x) - \langle a(x)|D_{t_f}\rangle\right|$.

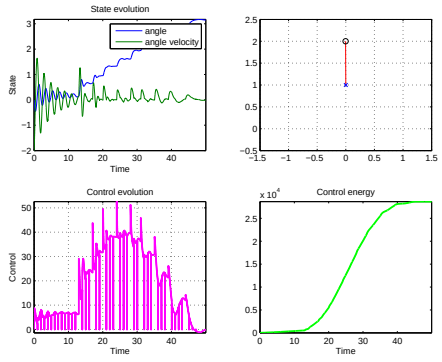


(b) Posterior standard deviation $\sqrt{\text{var}[a(x)|D_{t_f}]}$.

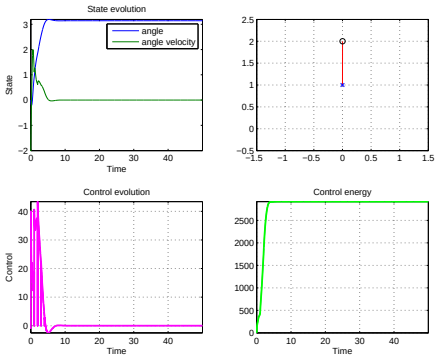


(c) Posterior over $\log b$.

Figure 3.2.: Experiment A. Posterior models of SP1. Stars indicate training examples. The random field has learned the dynamics functions in explored state space with sufficient accuracy.



(a) Control evolution of SP1, with untrained prior.



(b) Control evolution of controller SP2, benefiting from learning experience from the first round.

Figure 3.3.: Experiment B. Comparison of runs with untrained and pre-trained processes. The low-length scale processes benefit significantly from pre-training.

learning as much as possible). The results, depicted in Fig. 3.3 are consistent with this expectation.

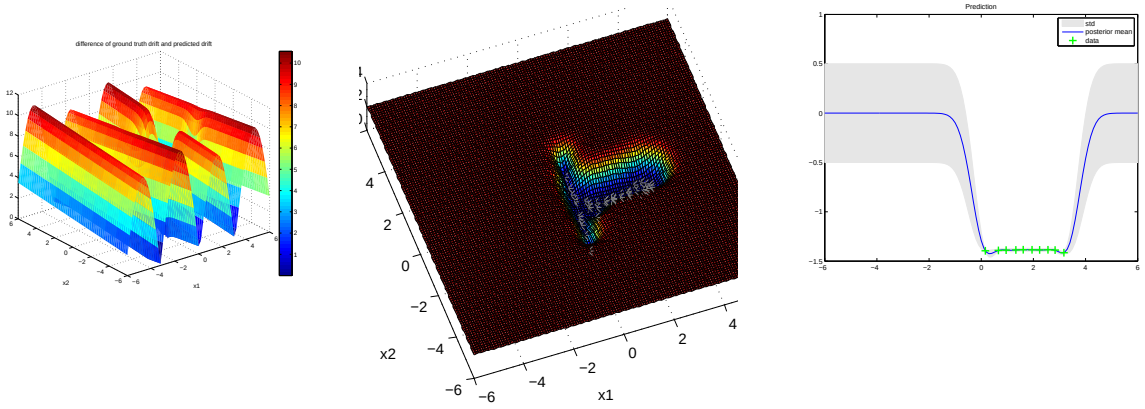
Experiment C. We investigated the impact of inappropriate magnitudes of confidence in a wrong model. We endowed the controller's priors with zero mean functions and RQ-ARD kernels [201]. Length scales of kernel K_a were set to 20 and the output scale to 0.5. In addition to the low output-scale, we set observational noise variance to a low value of 0.0001 suggesting (ill-founded) high confidence in the prior. The length scale of kernel K_b was set to 50 with low output scales and observational noise variance of 0.5 and 0.001, respectively.

The results, depicted in Fig. 3.5. As to be expected, the controller fails to realise the inadequacy of its beliefs. This results in a failure to update its beliefs and consequently, in a failure to converge to the target state.

Of course, this could be overcome with an actor-critic approach. Such solutions will be investigated in the context of future work.

Experiment D. Exp. C was repeated. This time, however, the kernel hyper-parameters were found by maximizing the marginal likelihood of the data. The automated identification of hyper-parameters is beneficial in practical scenarios where definition of a good prior for the underlying dynamics may be hard to conceive.

The optimiser succeeded in finding sensible parameters that allowed good control performance. As before, the method benefited from prior training yielding faster convergence and lower control effort. Both untrained and pre-trained methods outperformed the P-controllers either in terms of control energy or convergence. Finally, the SP controllers with hyper-parameter optimisation outperformed the SP controllers with fixed

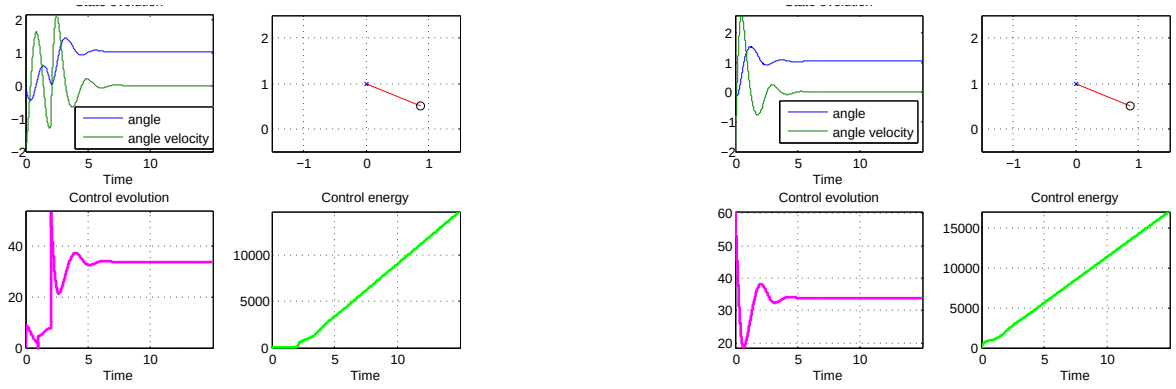


(a) $\left|a(x) - \langle a(x)|D_{t_f}\rangle\right|$.

(b) Posterior standard deviation $\sqrt{\text{var}[a(x)|D_{t_f}]}$.

(c) Posterior over $\log b$.

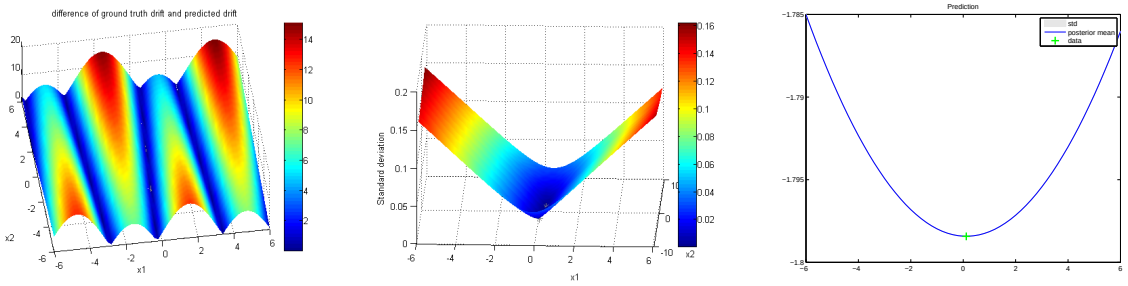
Figure 3.4.: Experiment B. Posterior models of SP1. Stars indicate training examples. We can see how the small length scales result in poor generalization performance in unexplored state space.



(a) Control evolution of SP1, with untrained prior.

(b) Control evolution of controller SP2, benefiting from learning experience from the first round.

Figure 3.5.: Experiment C. Comparison of runs with untrained and pre-trained processes. Neither run succeeds in arriving at the target state due to being overly confident.



(a) $\left|a(x) - \langle a(x)|D_{t_f}\rangle\right|$. (b) Posterior standard deviation $\sqrt{\text{var}[a(x)|D_{t_f}]}$. (c) Posterior over $\log b$.

Figure 3.6.: Experiment C. Posterior models of SP1. Stars indicate training examples. Note, the low posterior variance suggests misleading confidence in an inaccurate model.

hyper-parameters set in Exp. C (c.f. Tab. 3.2).

Torque-Actuated Double Pendulum

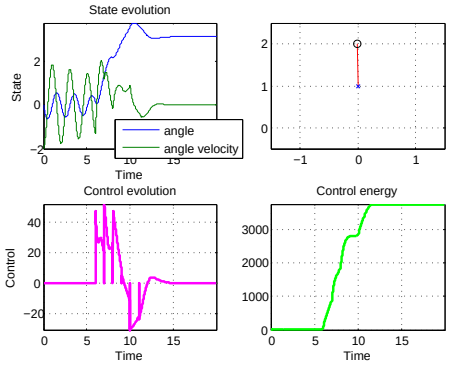
As a first non-trivial test case, we explored our controller's performance on a frictionless, torque-actuated double-pendulum as derived in [248]. The state $x \in \mathbb{R}^4$ consists of two joint angle positions and velocities. The uncontrolled system is known to exhibit chaotic behaviour and is unstable for all states except zero. Furthermore, the double pendulum has been used as model for a simple two-link robotic manipulator [234].

The system could be controlled by applying a torque to each joint. Given an initial state $x_0 = [0; 0; -1; -1]$ (downward position, with negative initial velocity), the task was to drive the double-pendulum upwards and stabilise the state at $x_f = [\pi; \pi, 0, 0]$ (motionless upward position).

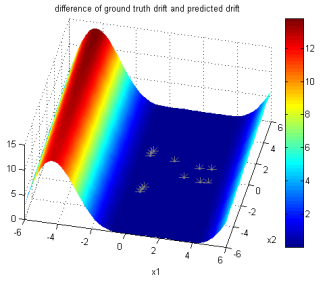
Our *RF-SIIC* controller was initialised with a the prior as described above. Kernels K_a and K_b were again chosen to be from the class of rational quadratic kernels with automated relevance detection (RQ-ARD) (cf. Sec. 2.6). The observational noise variance was set to 0.01. The log-normal process over $b(\cdot)$ was implemented by placing a normal over $\log b(\cdot)$ with zero mean and RQ-ARD kernel with fixed observational noise level 0.02. Note, the latter was set higher to reflect the uncertainty due to Π^a. In the future, we will consider incorporating hetereoscedastic observational noise based on $\text{var}[a]$ and the sampling rate. Also, one could incorporate knowledge about periodicity in the kernel.

To showcase the learning behaviour, we conducted the experiment in three stages.

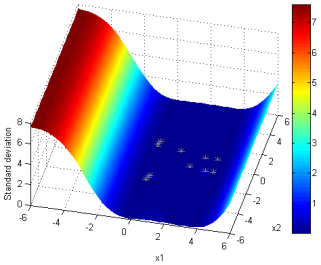
(I). As always, learning was done by conditioning on the observed training examples. However, in this first run, every third learning step, we allowed for full hyper-parameter training by optimising the marginal log-likelihood (see [201]). This hyper-parameter optimisation can make a significant difference if the ground truth dynamics are unlikely under the presupposed prior. Results are depicted in Fig. 3.8(a) and 3.8(b). Our



(a) Control evolution of SP1, with untrained prior.

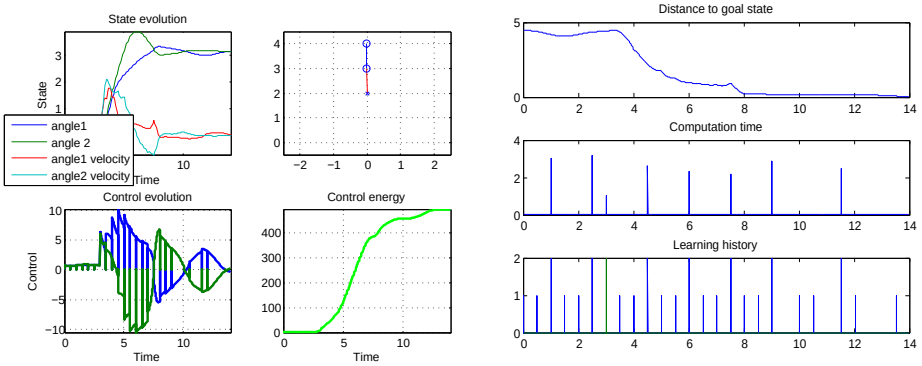


(b) $\left|a(x) - \langle a(x)|D_{t_f}\rangle\right|$.



(c) Posterior standard deviation $\sqrt{\text{var}[a(x)|D_{t_f}]}$.

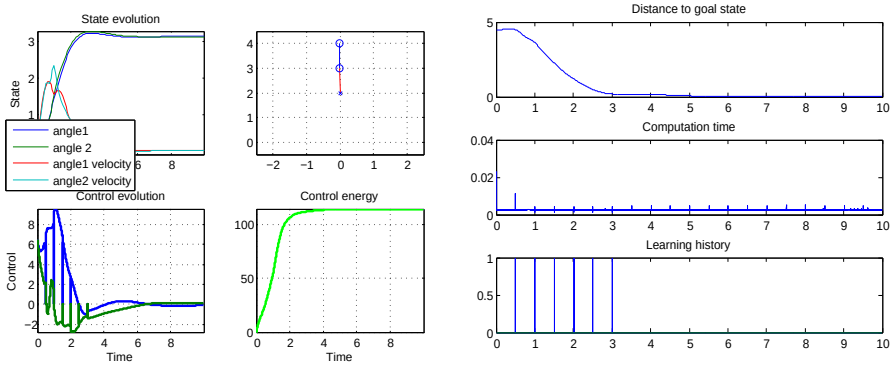
Figure 3.7.: Experiment D. Posterior models of SP1. Stars indicate training examples. The optimisation process succeeded in finding a sufficiently appropriate model.



(a) Control evolution and state with untrained prior.

(b) Performance history as a function of time (sec).

Figure 3.8.: Exp. I. Left-half: Control and state history. Bottom left plot: Blue curve: $u_1(x,t)$. Green curve: $u_2(x,t)$. Right-half: Evolution of distance to goal (top), computation time [s] of the controller and record of when learning took place (bottom). For the latter, values have the following meaning: 0: no learning took place, 1: learning by conditioning only, 2: full learning, including hyper-parameter optimisation.



(a) Control and state evolution.

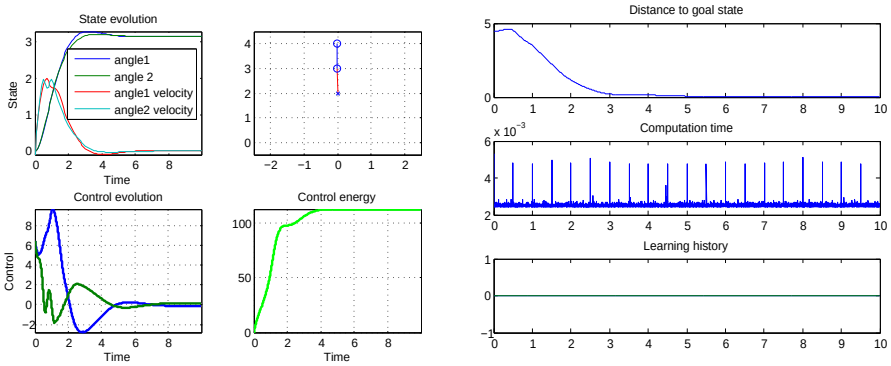
(b) Performance history as a function of time (s).

Figure 3.9.: Exp. II. Repetition of Exp. I with pre-trained controller and without hyper-parameter optimisation. Bottom left plot: Blue curve: $u_1(x,t)$. Green curve: $u_2(x,t)$.

RF-SIIC method managed to stabilise the system at the goal state after 14 seconds of online learning. Note, the drops in control signal (see Fig. 3.8(a)) are the probing actions performed to learn about the drift $a(\cdot)$.

(II). The previous experiment was restarted, however, with the learner pre-trained from Exp. 1 and without hyper-parameter optimisation. That is, learning was based on conditioning only. The results are depicted in Fig. 3.9(a) and 3.9(b). Observe, the controller benefits from the learning experience from the previous round evoking fewer learning steps yielding faster convergence to the goal. Furthermore, the processing time for controller calls is drastically reduced due to the absence of hyper-parameter optimisation.

(III). Once again, the experiment was restarted. This time, however the learner was switched off so that the controller had to rely on the posterior models trained during the previous two rounds. The learner controller



(a) Control evolution of RF-SIIC. (b) Performance history as a function of time (s).

Figure 3.10.: Exp. III. Repetition of Exp. I with pre-trained controller and without any further learning.

successfully drove the system to the goal (see Fig. 3.10(a) and 3.10(b)).

3.5.2. Non-collocated PFL for controlling a pendulum-driven cart-pole with uncertain drift

As a simple example of an underactuated control problem we explored our approach in simulations of a cart-pole with dynamics following [248] but having an uncertain drift term. Identifying and controlling in the presence of drift is a research area in its own right (see [162] and the references included therein). In our setting, friction merely serves as an example of a drift that can depend on the environment and would typically be uncertain but learnable. Other examples of drifts are manifold and depend on the application at hand. For instance, the terrain the cart is deployed in could have a slope which would induce location-dependent gravitational forces, an underwater vehicle could be exposed to currents and so on.

In our cartpole example, the configuration was $q = [x; \theta] \in \mathbb{R}^2$, where $q_1 = x$ was the cart position and $q_2 = \theta$ the joint angle of the pole.

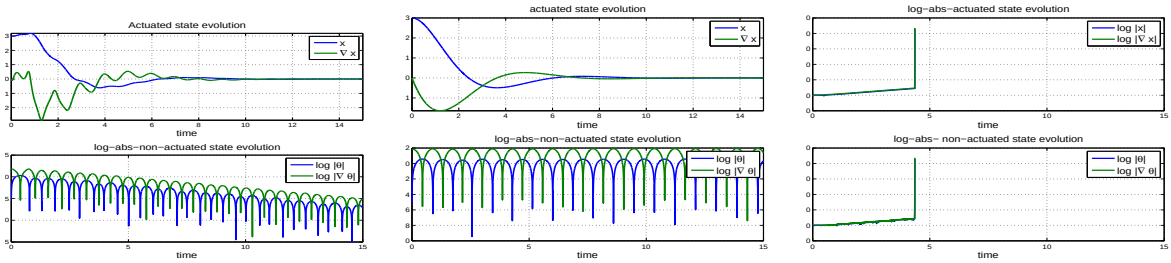
The ground-truth equations of motion were given by

$$H(q) = \begin{pmatrix} m_c + m_p & m_p l \cos q_2 \\ m_p l \cos q_2 & m_p l^2 \end{pmatrix}, C(q, \dot{q}) = \begin{pmatrix} r_1(q_1) & -m_p l \dot{q}_2^2 \sin q_2 \\ r_2(q_2) & 0 \end{pmatrix}, G(q) = \begin{pmatrix} 0 \\ m_p\, g\, l \sin q_2 \end{pmatrix}.$$

Here, $g = 9.81$ is acceleration due to gravity l is the pole length and m_p, m_c the masses of the pole and cart, respectively. The pole's joint was frictionless. Furthermore, $r_1, r_2 : \mathbb{R} \to \mathbb{R}$ denote friction maps of the ground the cart pole stood on and the of the pendulum, respectively. For our experiments, we chose a very "slippery" surface, i.e. $r_1 : x \mapsto 0$.

We assumed that the cart could be controlled by application of a force u pushing the cart to the left or right and that the pendulum was unactuated. Mathematically, this translates to control u only influencing q_1 and thus, $b(q) = \text{diag}(1, 0)$ (cf. Sec. 3.2).

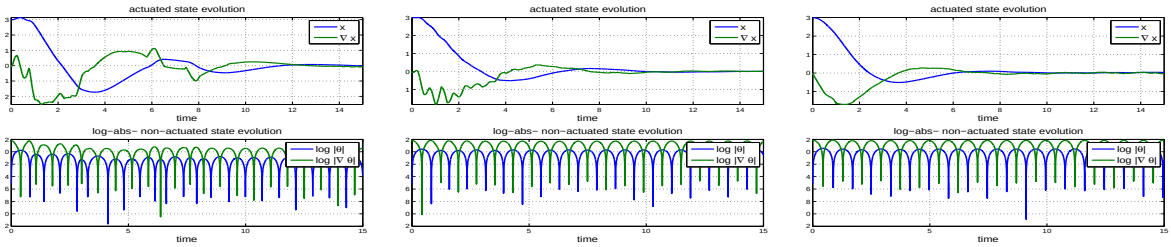
Given an initial state $x(0) = [x(0); \dot{x}(0); \theta(0); \dot{\theta}(0)] = [3; 0; 0; 6]$ the control task was to drive the state to



(a) Exp0. State trajectory when the cart is solely controlled by a proportional feedback law. The state converges but with marked deviation from the reference.

(b) Exp1. PFL succeeds following the reference exactly.

(c) Exp2. PFL fails. State divergent.



(d) Exp3. GP learner enables controller to successfully learn the PFL.

(e) Exp4. GP learner enables controller to successfully learn the PFL.

(f) Exp5. Benefiting from previous learning rounds, the fully trained SP-cPFL controller closely matches the reference (cf. Exp1).

Figure 3.11.: State trajectories of our experiments.

goal configuration $\xi = [x_{goal}; \dot{x}_{goal}] = [0; 0]$ following the linear reference dynamics $\ddot{x} = -w_1 x - w_2 \dot{x}$ for given gain weights $w_1, w_2 \geq 0$. The simulations of the controlled cart-pole were based on a first-order Euler approximation and 0th-order sample and hold control with time-discretisation step sizes $\Delta_t = \Delta_u = 0.001$.

An overview of all parameter settings is given Tab. 3.3.

Parameter(s)	Setting	Meaning
l	0.5	Pole length [m]
r	$\mapsto 0$	Friction map (floor)
(m_c, m_p)	$(.5, .5)$	Cart and pole masses [kg]
Δ_u	0.001	Control period
Δ_λ	0.25	Learning period
θ^a_{var}	0.01	Variance threshold
g	9.81	Gravitational acceleration $[m/s^2]$
(w_1, w_2)	(1,1)	Pseudo feedback gain weights
ξ	(0,0)	Target cart configuration
$s(0)$	(3,0,0,6)	Initial state
σ_o^2	0.01	Observational noise variance

Table 3.3.: Parameter settings.

With these settings, we conducted six experiments with varying controllers:

- Exp0: Simple proportional control.

- Exp1: Non-collocated PFL where the true dynamics were completely known.
- Exp2: Non-collocated PFL, but this time, the controller was mislead to believe the friction was $r : x \mapsto 9$.
- Exp3: Our SP-cPFL method as introduced above. We utilised a Gaussian process (GP) to learn the drift based on a zero-mean prior $a \sim \mathcal{GP}(0, K_a)$ with covariance function K_a being chosen to be an rational-quadratic kernel with automated relevance detection (RQ-ARD) [201]. For increased robustness against numerical errors in the acceleration estimates, we set the observational noise variance to a small but non-zero level of $\sigma_o^2 = 0.01$. The kernel hyper-parameters were adjusted online, maximising the marginal posterior log-likelihood (as in Exp. D above).
- Exp4: A repetition of Exp3, but on the basis of the posterior GP pre-trained in Exp3. Subsequent learning took place on the basis of conditioning only, i.e. without hyper-parameter optimisation.
- Exp5: A repetition of Exp4, but on the basis of the posterior GP pre-trained in Exp4 and without subsequent learning.

In all experiments, the inner-loop pseudo control was set as per Eq. 3.19 with feedback gains $w_1 = w_2 = 1$.

The results are depicted in (cf. Fig. 3.11). Fig. 3.11.a depicts the state trajectory controlled by a simple proportional controller. Note how the swinging pole affects the cart position and velocity. As expected, the PFL-based controller, knowing the correct dynamics, was successful in exhibiting reference behaviour in x (cf. Fig. 3.11.b) by completely removing the effects of the swinging pole on the cart. However, as demonstrated by Exp. 2, the PFL approach is sensitive to having a false friction model causing the state to diverge (cf. Fig. 3.11.c). Fortunately, our GP-based controller successfully learns to partially linearise the model in a correct manner even when being initialised with an uninformed prior over drift field a (cf. Fig. 3.11.d). Examining the results of Exp5 (see Fig. 3.11.f), observe how the approach is able to benefit from multiple learning episodes reproducing the desired behaviour exhibited by the PFL controller from Exp1 (cf. Fig. 3.11.b) after only two rounds of online learning.

3.5.3. Observations

We note that the majority of run-time was taken up with hyper-parameter optimisation. While the latter would benefit from an improved choice of optimiser, implementation and hardware, it might be appropriate to base hyper-parameter optimisation on collected offline data and restrict online learning to conditioning. In spite of not optimising the implementation for speed (Matlab, execution on a standard laptop), run-times with a learner based on conditioning suggest that learning can be done in real-time. The trained controller could

be called within a few microseconds and hence, already is feasible to be deployed in real-time already. This sets the method apart to other methods that rely on computationally expensive planning such as dynamic programming [79] or optimisation-based MPC. Also note, that, once the model is fully trained (see Exp. III), the control signal is smooth and was generated in micro-seconds.

Apart from fully actuated systems, we also considered a simple underactuated problem. While the setup was simplistic, it proved the point that our approach was successful in learning to correctly partially linearise an underactuated system and could learn to remove the influence of the non-linearised dynamics to induce the desired plant behaviour.

3.6. Discussion and concluding remarks

We have proposed a new framework, RF-SIIC, that combines learning with random fields and adaptive (partial and full) feedback linearisation to learn a control policy in a Bayesian fashion. In contrast to most related methods, our approach takes into account the structure of dynamics equation and can learn drift and control input vector fields in separation. This is interesting from a system identification point of view as it allows for a more detailed understanding of the physics of the underlying system. Furthermore, if the drift changes (e.g. due to change in the plant's environment) the identified control input function remains valid and the system does not have to be relearned from scratch. In addition the structural knowledge allowed the learners to learn forward models whose dimensionality equalled the dimensionality of state space. This is in contrast to many competing methods that require learning on the joint state-action space [77, 180].

For the control of discrete-time systems, we were able to leverage the structural knowledge to provide a guarantee of convergence of the expected closed-loop trajectory to the desired goal state. Here, it is to be emphasized that since all probabilities are degrees of subjective beliefs, the convergence guarantee also is a guarantee about epistemic beliefs over the (deterministic) dynamic system behaviour.

Our simulations have illustrated our controller's behaviour in the context of simple simulated rigid body systems and served as first demonstration of the viability of the approach. They show that our approach can be successful in simultaneous online learning and control and that it is fast enough to be applied at high sample rates. Furthermore, utilising the variance as a learning criterion the online learning process was able to keep the training corpora small.

The efficiency of our learning and control methods comes at a price. In particular, if both the drift and the input mappings are uncertain, the need to distinguish between the two motivated us to set the control input to zero for brief periods of time in order to learn about the drift. This can be very disadvantageous in settings where such "zero-spikes" can destabilise the system and future work will investigate ways to circumvent this. Furthermore, to learn about input mapping b, we need to be reasonably certain about the drift at the state

where b is to be learned. This can imply that data about b will often be much sparser than the data available for the model of a.

3.6.1. Future work.

So far, bounded control is not considered. While this could be modelled by squashing the control output through a bounded function (e.g. in lieu to [79]), the present absence of a planning method precludes the controller to solve tasks such as swing-ups under bounded control. The latter would involve forecasting and planning. Future work could address this and seek to combine our SIIC approach in combination with MPC. An idea would be to replace the pseudo controller by a predictive controller that is capable of avoiding obstacles in state space. We might then explore how to add virtual obstacles to state space preventing the controller to choose actions that steer the state into a region of state space that is likely to cause the actuators to saturate. A control sequence that connects a start state with a desired goal state in free-space should then solve the control problem of the closed-loop dynamic system under constrained control.

In addition we will investigate utilising our SIIC framework with alternative learning methods. We do so briefly in Ch. 4 where we replace the random fields by our *kinky inference* method (*KI*) introduced in that chapter. Both the random fields and our KI method provide uncertainty quantifications that can be utilised for deciding when to learn and when not. In the present chapter, we have used the variance as such a criterion. However, as we have explored in the context of the pendulum experiments, the variance is a subjective quantity that may be misleading if reality does not match the beliefs of the learner. Therefore, we will investigate alternative methods. As a simple first step for instance, we could imagine to evoke learning if and only if the observed state transitions do not match with the predictions made on the basis of the current model. Apart from possibly being a more effective criterion this would impose a "never changing a winning team" behaviour, keep the data sets sparse and limit the number of aforementioned zero control spikes.

In fact, one of the most immediate things to explore would be to extend our system identification approach to avoid the injection of the "zero-spikes" in the control signal for the purpose of training example generation. Remember, at present, the control is set to zero in order to obtain a sample of $a(x)$ on the basis of a measurement of $\ddot{q}$ even when $b(x)$ is very uncertain. Instead we could investigate to what extend one could compute (or approximate) the posterior joint distributions over a and b given all the collected data. Alternatively, if we have access to data (or can generate it) where the same state is visited twice (at least approximately), we could investigate the following procedure to obtain the required training examples: Assume we have access to data containing records of states, control actions pertaining acceleration measurements. Moreover, assume the data contains records for the same state x twice, but where two different control actions $u, v, u \neq v$ were taken and acceleration estimates $\ddot{q}_1, \ddot{Q}_1$ were recorded. From these two records we can generate an estimate

of $a(x)$ and $b(x)$ for further use in Bayesian learning as follows: From the acceleration measurements we can compute $D := \ddot{q}_1 - \ddot{Q}_1$. Knowing about the control-affine structure of the system we obtain the equations $\ddot{q}_1 = a(x) + b(x)u$ and $\ddot{Q}_1 = a(x) + b(x)v$. This yields $D = b(x)(u - v)$. This is a linear equation which, provided $u_j \neq v_j$ for all dimensions j, we can solve for $b(x)$ to obtain a training example of b at x. To obtain a training example of drift a at x, we could now substitute $b(x)$ into our equations and solve for $a(x)$, e.g. $a(x) = \ddot{q}_1 - b(x)u$.

This procedure removes the necessity of injecting $u = 0$ into the control signal and allows us to generate observations about a and b simultaneously on the basis of two distinct visitations of the same state. In the context of Bayesian nonparametrics, conditioning on these data give posterior random fields over a and b which we can employ for control as described in this chapter. Investigations of the general viability of this approach, as well as of its merits and shortcomings in different situations, will be conducted in the course of future work.

4. Kinky inference for learning, prediction and control with bounded set uncertainty

"We have to have error bars around all our predictions. That is something that's missing in much of the current machine learning literature."

Michael Jordan (IEEE-Spectrum, 2014)

4.1. Introduction

In order to generalise beyond an observed sample, inductive inference needs to make assumptions about the underlying ground truth function. In this chapter, we will derive non-parametric learning and inference methods that fold in knowledge about boundedness, Hölder continuity and error bounds on the observations and inputs. Since these types of a priori knowledge impose weak forms of regularity, the method will be applicable to non-smooth functions that contain kinks- a property that is reflected in the predictions of the inference method and that gave rise to the catch-phrase *kinky inference*. While weak enough an assumption to encompass rich function classes, the approach has the advantage of yielding bounds on the inference.

The approach yields a non-parametric machine learning method which is applied to system identification and control. The provided error bounds are an attractive property since they offer information on bounded stability and provide guidelines for adjusting the control to take the remaining uncertainty of a partially identified system into account. This can be beneficial in settings where we desire to impose robustness constraints on collision avoidance (see Sec. 4.4.2) or stability and can help to guide state space exploration in active learning. Based on our nonparametric inference rule, we construct a variety of learning-based controllers for some of which we provide guarantees of stability and construct robust tubes. In addition to our theoretical guarantees, our simulations suggest that kinky inference is a very flexible and fast learning approach that is well suited for learning and controlling dynamical systems and that it can outperform state-of-the-art methods.

4.2. Kinky inference over function values - nonparametric machine learning

Supervised machine learning methods are algorithms for inductive inference. On the basis of a sample, they construct (learn) a generative model of a data generating process that facilitates inference over the underlying ground truth function and aims to predict function values at unobserved inputs. If the inferences (i.e. predictions of function values) are utilised in a decision-making process whose outcome involves risk, information about the prediction uncertainty can be vital. For instance, a robot that has a poor model about its dynamics would have to keep greater distance to obstacles than one that has a good model. Having such applications in mind, we are especially interested in learning algorithms that allow for conservative inference. That is, the predictions come with uncertainty quantifications that never underestimate the true uncertainty.

Typically there are infinitely many explanations that could explain any finite data set. Therefore, one needs to impose assumptions a priori in order to be able to generalise beyond the observed data and to establish the desired uncertainty quantifications.

These assumptions, sometimes referred to as inductive bias [171], can take various forms and are an essential distinguishing factor between different machine learning algorithms. The inductive bias implicitly or explicitly restricts the hypothesis space. That is, the space of target functions that can be learned. In this work, we devise a learning method for ground truth functions that are assumed to be contained in the a priori hypothesis space $\mathcal{K}_{\text{prior}}$ whose members are known to be Hölder continuous functions. Moreover, the hypotheses may be known to be contained within certain predefined bounds.

In the case in which the parameters of the Hölder class the target function is contained in are known (or at least an upper bound on the Hölder constant and bounds) we can give robust error estimates that give worst-case guarantees on the predictions. In the event the presumed parameters turn out to be wrong, Sec. 4.3 contains proposals on how to update the belief over the constants in the light of new data.

Conservative learning and inference in a nutshell

The general approach we will consider is as follows. We desire to learn a target function f based on prior knowledge of the form that $f \in \mathcal{K}_{prior}$ and some (noisy) function sample $\mathcal{D}$. The step of forming a posterior belief of the form $f \in \mathcal{K}_{post} \supseteq \mathcal{K}_{prior} \cap \mathcal{K}(\mathcal{D})$ will be referred to as *conservative inference over functions* (i.e. learning). Here, $\mathcal{K}(\mathcal{D})$ is the set of all functions that could have generated the observed data. The conservatism lies in the fact that we have not ruled out any candidate functions that are consistent with our information. For simplicity, for now, assume a one-dimensional target $f : \mathcal{X} \to \mathbb{R}$. To perform inference over a function *value* at input x in a conservative manner, we merely infer that $f(x) \in \hat{H}(x) = [\mathfrak{l}(x), \mathfrak{u}(x)]$ where *floor function* $\mathfrak{l}(x) = \inf_{\phi \in \mathcal{K}_{post}} \phi(x)$ and *ceiling function* $\mathfrak{u}(x) = \sup_{\phi \in \mathcal{K}_{post}} \phi(x)$ delimit the values

that any function in the posterior class $\mathcal{K}_{post}$ could give rise to. If a point prediction of a function value is required, we could choose any function value $\hat{\mathfrak{f}}(x) \in [\mathfrak{l}(x), \mathfrak{u}(x)]$ between the values given by the floor and ceiling which also give rise to the bounds on the point prediction.

Note that generally, computation of $\mathfrak{u}(x)$ and $\mathfrak{l}(x)$ will have to take all data points into account. For large data sets $\mathcal{D}$, this may become computationally intractable. Instead, we might to limit our computations of the delimiting functions $\mathfrak{u}, \mathfrak{l}$ to a subset of data points. On the other hand, if computation is not an issue then we will see how we can compute $\mathfrak{u}, \mathfrak{l}$ optimally. In this case, conservatism is maximally reduced and we can guarantee that $\mathcal{K}_{post} = \mathcal{K}_{prior} \cap \mathcal{K}(\mathcal{D})$ and that the magnitude of uncertainty measured by $\frac{1}{2}|\mathfrak{u}(x) - \mathfrak{l}(x)|$ is minimal.

Of course, the type of inference we get from this general procedure depends on the chosen prior class $\mathcal{K}_{prior}$ and the specifics of the data. We will develop *kinky inference* which is an approach to conservative inference that considers prior classes of multi-output functions over (pseudo-) metric input spaces that are optionally constrained by boundedness and by knowledge of Hölder regularity as well as knowledge on uncertainty bounds on the observations and function inputs. For this class of inference mechanisms we prove conservatism and convergence guarantees to the learning target. What is more, we will consider classes where the imposed regularity assumptions are uncertain or inferred from data to adjust for overly restrictive prior assumptions that do not fit the observations.

This capability of imposing boundedness assumptions, accommodating for noise and being able to adjust the regularity assumptions will be of importance in the control applications of our method that we will develop in later parts of this work.

4.2.1. The framework of conservative inference – definitions, setting and desiderata

Let $f : \mathcal{X} \to \mathcal{Y}$ be a function where $\mathcal{X}$, $\mathcal{Y}$ are two spaces endowed with (pseudo-) metrics $\mathfrak{d}_{\mathcal{X}} : \mathcal{X}^2 \to \mathbb{R}_{\geq 0}$, $\mathfrak{d}_{\mathcal{Y}} : \mathcal{Y}^2 \to \mathbb{R}_{\geq 0}$, respectively. Spaces $\mathcal{X}, \mathcal{Y}$ will be referred to as *input space* and *output space*, respectively.

For simplicity, in this work, we will assume $\mathcal{Y} \subseteq \mathbb{R}^m$ and $\mathfrak{d}_{\mathcal{Y}}(y, y') = \|y - y'\|$ for some standard norm that is equivalent to the maximum norm.[1] In contrast, we do not impose restrictions on the input space (pseudo-) metric. Since the output metric is defined via a norm, we will often drop the subscript of the input metric. That is we may on occasion write $\mathfrak{d}$ instead of $\mathfrak{d}_{\mathcal{X}}$.

Assume we have access to a *sample* or *data set* $\mathcal{D}_n := \{(s_i, \tilde{f}_i, \varepsilon(s_i)) | i = 1, \ldots, N_n\}$ containing $N_n \in \mathbb{N}$ sample vectors $\tilde{f}_i \in \mathcal{Y}$ of function f at sample input $s_i \in \mathcal{X}$. To aide our discussion, we define $G_n = \{s_i | i = 1, \ldots, N_n\}$ to be the *grid* of sample inputs contained in $\mathcal{D}_n$. Subscript n aides our exposition when

[1] Extensions to more general output spaces seem very possible, albeit, instead of learning a function, we would then learn a discrepancy from a nominal reference in output space. We will leave this to future work.

we consider sequences of data sets indexed by n.

The sampled function values are allowed to have interval-bounded *observational error* given by $\varepsilon : \mathcal{X} \to \mathbb{R}^m_{\geq 0}$. That is, all we know is that $f(s_i) \in \mathbf{O}_i := [\underline{f}_1(s_i), \overline{f}_1(s_i)] \times \ldots \times [\underline{f}_d(s_i), \overline{f}_d(s_i)]$ where $\underline{f}_j(s_i) := \tilde{f}_{i,j} - \varepsilon_j(s_i)$, $\overline{f}(s_i) := \tilde{f}_{i,j} + \varepsilon_j(s_i)$ and $\tilde{f}_{i,j}$ denotes the jth component of vector $\tilde{f}_i$.

The interpretation of these errors depends on the given application. For instance, in the context of system identification, the sample might be based on noisy measurements of velocities. The noise may be due to sensor noise or may represent numerical approximation error.

As we will see below, ε may also quantify *input* uncertainty (that is when predicting $f(x)$, x is uncertain). This is an important case to consider, especially in the context of control applications where the ground truth might represent a vector field or an observer model. Finally, the observational error can also model error around the computation of the metric. This might be of interest if the metric has to be approximated numerically or estimated with statistical methods that provide confidence intervals (in the latter case our predictions also only hold within pertaining confidence bounds).

It is our aim to learn function f in the sense that, combining prior knowledge about f with the observed data $\mathcal{D}_n$, we infer *predictions* $\hat{\mathfrak{f}}_{\mathfrak{n}}(x)$ of $f(x)$ at unobserved *query inputs* $x \notin G_n$. Being the target of learning, we will refer to f as the *target* or *ground-truth* function. In our context, the evaluation of $\hat{\mathfrak{f}}_n$ is what we refer to as *(inductive) inference*. For a discussion of the competing definitions of non-deductive inference in a philosophical context, the reader is referred to [98]. The entire function $\hat{\mathfrak{f}}_{\mathfrak{n}}$ that is learned to facilitate predictions is referred to as the *predictor*. Typically, the predictor lies in the space that coincides with (or is at least dense in) the a priori hypothesis space $\mathcal{K}_{prior}$ of conceivable target functions. In this case the predictor can be referred to as a *candidate function* or *hypothesis* [171].

In addition to the predictions themselves, we are also interested in *conservative bounds* on the error of the inferred predictions. Depending on the hypothesis space under consideration, these bounds can be of a probabilistic or deterministic nature. In this chapter, we restrict ourselves to the latter.

That is, we desire to provide a computable *prediction uncertainty function* $\hat{\mathfrak{v}}_{\mathfrak{n}} : \mathcal{X} \to \mathcal{Y}$ such that we *believe* that for any input query $x \in \mathcal{X}$, the ground truth value $f(x)$ lies somewhere within the *uncertainty hyperrectangle*

$$\hat{H}_n(x) := \left\{ y \in \mathcal{Y} \,|\, \forall j \in \{1, ..., \dim \mathcal{Y}\} : y_j \in \hat{H}_{n,j}(x) \right\} \tag{4.1}$$

around the prediction $\hat{\mathfrak{f}}_{\mathfrak{n}}(x)$ where

$$\hat{H}_{n,j}(x) := [\,\hat{\mathfrak{f}}_{\mathfrak{n},\mathfrak{j}}(x) - \hat{\mathfrak{v}}_{\mathfrak{n},\mathfrak{j}}(x),\, \hat{\mathfrak{f}}_{\mathfrak{n},\mathfrak{j}}(x) + \hat{\mathfrak{v}}_{\mathfrak{n},\mathfrak{j}}(x)] \tag{4.2}$$

is referred to as the jth *prediction uncertainty* interval. Here, $\hat{\mathfrak{f}}_{\mathfrak{n},\mathfrak{j}}(x)$, $\hat{\mathfrak{v}}_{\mathfrak{n},\mathfrak{j}}(x)$ denote the jth components of

vectors $\hat{\mathfrak{f}}_{\mathfrak{n}}(x)$, $\hat{\mathfrak{v}}_{\mathfrak{n}}(x)$, respectively.

In this chapter, will understand a *machine learning mechanism* to implement a computable function that maps a data set $\mathcal{D}_n$ to a prediction function $\hat{\mathfrak{f}}_n$ and an uncertainty estimate function $\hat{\mathfrak{v}}_n$. The question remains to be addressed which properties these functions are desired to exhibit.

Mitchell [171] proposed to classify an algorithm as a machine learning algorithm, if it improves its performance with respect to some specified cost function with increasing data. We are interested in learning a function that *converges* to the ground truth. That is, in the limit of infinite, informative data, the point-wise distance, measured by a norm, between the prediction and the ground-truth should shrink *monotonically*. That is, the cost in Mitchell's definition might be quantified by this distance measuring the discrepancy between predictor and target.

Being interested in *conservative* inference, we desire to give conditions under which the uncertainty beliefs $\hat{\mathfrak{v}}_n(x)$ around the predictions $\hat{\mathfrak{f}}_n(x)$ are never overoptimistic. That is, for any input $x \in \mathcal{X}$, we desire to give a guarantee that the target $f(x)$ cannot assume values outside the uncertainty hyperrectangle $\hat{H}_n(x)$.

In order to conceive an inference mechanism that can generalise beyond the sample and to establish its desired properties, we need to make a priori assumptions. That is, it is important to impose a priori restrictions on the class of possible targets (the *hypothesis space*). As mentioned above, we will denote this space by $\mathcal{K}_{prior}$.

Once sample $\mathcal{D}_n$ is known, we can combine the information contained therein with the prior. At the very least, we can form a *posterior hypothesis space* $\mathcal{K}_{post}$ by falsification. That is, we eliminate all hypotheses from $\mathcal{K}_{prior}$ that could not have generated the sample.

To formalise this, we define

$$\mathcal{K}(\mathcal{D}_n) = \left\{ f : \mathcal{X} \to \mathcal{Y} | \, \forall i \in \{1, ..., N_n\} : \, f(s_i) \in \mathbf{O}_i \right\} \tag{4.3}$$

to be the set of *sample-consistent* functions.

To perform inference over functions we combine the observed data with a priori knowledge about the hypothesis class of possible candidate target functions and construct a posterior hypothesis set $\mathcal{K}_{post}(\mathcal{D}_n)$. Normally, we would expect *monotonicity* of our inference. That is, the additional information afforded by the data should not make us more uncertain, i.e. $\mathcal{K}_{post}(\mathcal{D}_n) \subset \mathcal{K}_{prior}$. Disregarding potential computational restrictions, we should take the available data into account and require consistency. That is, ideally, we desire the satisfaction of the requirement $\mathcal{K}_{post}(\mathcal{D}_n) \subset \mathcal{K}(\mathcal{D}_n)$. In this case, we then have $\mathcal{K}_{post}(\mathcal{D}_n) \subseteq \mathcal{K}_{prior} \cap \mathcal{K}(\mathcal{D}_n)$. Since the requirement of *conservatism* demands that no function that could have generated the available data should be ruled out (i.e. $\mathcal{K}_{post}(\mathcal{D}_n) \supseteq \mathcal{K}(\mathcal{D}_n)$) the conservative optimal inference over

functions would be:[2]

$$\mathcal{K}_{post}(\mathcal{D}_n) = \mathcal{K}_{prior} \cap \mathcal{K}(\mathcal{D}_n). \tag{4.4}$$

As mentioned above, computing a prediction and bounds on the basis of $\mathcal{K}_{prior} \cap \mathcal{K}(\mathcal{D}_n)$ might involve computation that grows linearly with the data set size $N_n = |\mathcal{D}_n|$. Therefore, computational limitations may impose the requirement to base our inferences $\hat{\mathfrak{f}}_{\mathfrak{n}}(x)$, $\hat{\mathfrak{v}}_{\mathfrak{n}}(x)$ over function values on a subset or reduced summary of the available data. Retaining the requirement of conservatism, from the function inference point of view, this translates to having

$$\mathcal{K}_{post}(\mathcal{D}_n) = \mathcal{K}_{prior} \cap \tilde{\mathcal{K}}(\mathcal{D}_n) \text{ where } \tilde{\mathcal{K}}(\mathcal{D}_n) \supset \mathcal{K}(\mathcal{D}_n). \tag{4.5}$$

Having learned, i.e. performed inference over functions, we can use this to predict function values conservatively by considering all the function values within the posterior class which could be assumed at a given query input $x \in \mathcal{X}$. In Sec. 4.2.6, we will consider two examples of such reduced data inference approaches. Apart from a simple finite subset approach ($\tilde{\mathcal{K}}(\mathcal{D}_n) = \mathcal{K}(S), S \subset \mathcal{D}_n, |S| \leq N_{max}$), we will consider the case where a prediction of a function value at each input will take into account those data points that correspond to the k nearest neighbours of the input in the grid. For the time being however, our main focus will be on inference based on the full data set $\mathcal{D}_n$, i.e. on the case where $\tilde{\mathcal{K}}(\mathcal{D}_n) = \mathcal{K}(\mathcal{D}_n)$.

Remark 4.2.1 (False prior hypotheses, lazy relaxations and search bias). Furthermore, note that it might happen that $\mathcal{K}_{prior} \cap \mathcal{K}(\mathcal{D}_n) = \emptyset$. Presuming there was no problem with the data set, this might be indicative of $\mathcal{K}_{prior}$ being too restrictive. One approach to deal with such a situation would be to relax our prior assumptions. If we do so just enough to ensure that $\mathcal{K}_{post}(\mathcal{D}_n) \neq \emptyset$, we will call this relaxation *lazy*. Below, we will consider a priori hypothesis spaces constrained by Hölder regularity under the assumption that the parameters of the Hölder condition are known. Since this is often difficult to achieve in practice, Sec. 4.3 contains a lazy adaptation rule of the Hölder constant. This is an example of a lazy relaxation of the prior hypothesis space. From a wider point of view, one could interpret the relaxation procedure as being part of the learning algorithm. From this vantage point, the actual prior hypothesis space is the maximally relaxed space on which the algorithm imposes a *search bias* [171] that prefers hypotheses in more constrained sets over those that lie in more relaxed ones. Since more constrained spaces have lower learning capacity, this search bias imposes a form of regularization found in many machine learning approaches [201]. Interestingly, in the context of the relaxations of the Hölder constant considered in Sec. 4.3.2, preferring lower Hölder constants over larger ones seems to establish a strong link to regularisation in kernel methods [260]. Investigating these links is something we have not done yet, but which might be undertaken in future work.

After these preparations we are now in a position to give a formal summary of properties we would ideally

[2] As an aside, note that the introduction of conservatism to the inference renders the spirit of the approach consistent with Popper's falsification paradigm to epistemology.

like an inference mechanism to exhibit:

Definition 4.2.2 (Desiderata)**.** *With definitions as above, we desire a machine learning algorithm to implement a mapping $\mathcal{D}_n \mapsto (\hat{\mathfrak{f}}_{\mathfrak{n}}, \hat{\mathfrak{v}}_{\mathfrak{n}})$ such that, for all data sets $\mathcal{D}_n$, the resulting inductive inference satisfies the following **desiderata**:*

1. *Conservatism: $\forall x \in \mathcal{X}\, \forall \phi \in \mathcal{K}_{prior} \cap \mathcal{K}(\mathcal{D}_n) : \phi(x) \in \hat{H}_n(x)$. In particular, the ground-truth f is always contained in the posterior set and its function values always are contained in the corresponding uncertainty hyperrectangles.*

2. *Monotonicity: Additional data cannot increase the uncertainty.*

   That is, $\mathcal{D}_n \subseteq \mathcal{D}_{n+1}$ implies $\hat{H}_n(x) \supseteq \hat{H}_{n+1}(x)$ or equivalently, $\hat{\mathfrak{v}}_n(x) \geq \hat{\mathfrak{v}}_{n+1}(x), \forall x \in \mathcal{X}$ (where the inequality holds for each component).

   Ideally, the following other two desiderata should hold as well:

3. *Convergence: If $(G_n)_{n\in\mathbb{N}}$ is a sample grid sequence of inputs converging to a dense subset of $\mathcal{X}$ (as $n \to \infty$) then $\hat{\mathfrak{v}}_{\mathfrak{n}}(x) \overset{n\to\infty}{\longrightarrow} \varepsilon(x), \forall x \in \mathcal{X}$. If convergence of the grid to the dense subset is uniform (cf. Def. C.1.3) and the observational error is zero everywhere ($\varepsilon(x) = 0$, $\forall x$) then the convergence $\hat{\mathfrak{v}}_{\mathfrak{n}} \overset{n\to\infty}{\longrightarrow} 0$ is uniform.*

4. *Minimality (optimality): There is no conservative uncertainty bound that is tighter than $\hat{\mathfrak{v}}_n$. That is, if for some hyperrectangle H, $x \in \mathcal{X}$ we have $H \subsetneq \hat{H}_n(x)$ then we have: $\exists x \in \mathcal{X}, \phi \in \mathcal{K}_{prior} \cap \mathcal{K}(\mathcal{D}_n) : \phi(x) \notin H$. In particular, it might be possible that the function value of the ground-truth function at x is not contained in H.*

Remark 4.2.3. Note the statement of uniform convergence in the absence of observational error is interesting from a machine learning theoretic point of view. It tells us that if we can construct a grid sequence $(G_n)_{n\in\mathbb{N}}$ that uniformly converges to a dense set of the domain $\mathcal{X}$ then the worst-case approximation error $\sup_{x\in\mathcal{X}} \left\| \hat{\mathfrak{f}}_{\mathfrak{n}}(x) - f(x) \right\|_\infty$ vanishes as $n \to \infty$. This can also be of interest in quadrature: if we construct a closed-form expression of the integral of the functions $\mathfrak{u}_n : x \mapsto \hat{\mathfrak{f}}_{\mathfrak{n}}(x) + \hat{\mathfrak{v}}_{\mathfrak{n}}(x), \mathfrak{l}_n : x \mapsto \hat{\mathfrak{f}}_{\mathfrak{n}}(x) - \hat{\mathfrak{v}}_{\mathfrak{n}}(x)$ then, due to uniform convergence, $\int_{\mathcal{X}} f(x)\, \mathrm{d}x = \int_{\mathcal{X}} \lim_{n\to\infty} \mathfrak{u}_n(x)\, \mathrm{d}x = \lim_{n\to\infty} \int_{\mathcal{X}} \mathfrak{u}_n(x)\, \mathrm{d}x$ assuming the integrals exists. Past work, not included in the thesis, has utilised this result for conservative quadrature of Hölder continuous functions.

4.2.2. Kinky inference and nonparametric machine learning

Having specified general desirable properties on conservative inference we will now define a class of inference rules that have special cases that satisfy these desiderata under certain conditions. This class of inference rules, which we will call *kinky inference* rules, can be stated as follows:

Definition 4.2.4 (The class of kinky inference rules)**.** *Let $\mathbb{R}_\infty := \mathbb{R} \cup \{-\infty, \infty\}$ and $\mathcal{X}$ be some space endowed with a pseudo-metric $\mathfrak{d}_\mathcal{X}$. Let $\underline{B}, \bar{B} : \mathcal{X} \to \mathcal{Y} \subseteq \mathbb{R}_\infty^m$ denote lower- and upper bound functions that can be specified in advance and assume $\underline{B}(x) \leq \bar{B}(x), \forall x \in I \subset \mathcal{X}$ component-wise. Assume we are given a finite data set $\mathcal{D}_n := \{(s_i, \tilde{f}_i, \varepsilon(s_i)) | i = 1, \ldots, N_n\}$. For query input $x \in \mathcal{X}$, we define the set of data point indices $\mathcal{I}_n(x) \subseteq \{1, \ldots, N_n\}$. Furthermore, we define $w_\mathfrak{u}(x), w_\mathfrak{l}(x) \in [0, 1]$ such that $w_\mathfrak{u}(x) + w_\mathfrak{l}(x) = 1$ to be some weights. Unless explicitly stated otherwise, we will choose $w_\mathfrak{u}(x) = w_\mathfrak{l}(x) = \frac{1}{2}, \forall x \in \mathcal{X}$.*

Furthermore, we define the predictor $\hat{\mathfrak{f}}_\mathfrak{n} : \mathcal{X} \to \mathcal{Y}$ and uncertainty quantifier $\hat{\mathfrak{v}}_\mathfrak{n} : \mathcal{X} \to \mathbb{R}_{\geq 0}^m$ to perform inference over function values such that for query input $x \in \mathcal{X}$ and for $j = 1, \ldots, m$, their jth output components are given by:

$$\hat{\mathfrak{f}}_{\mathfrak{n},\mathfrak{j}}(x) := w_\mathfrak{u}(x)\ \min\{\bar{B}_j(x), \mathfrak{u}_{n,j}(x)\} + w_\mathfrak{l}(x)\ \max\{\underline{B}_j, \mathfrak{l}_{n,j}(x)\} \tag{4.6}$$

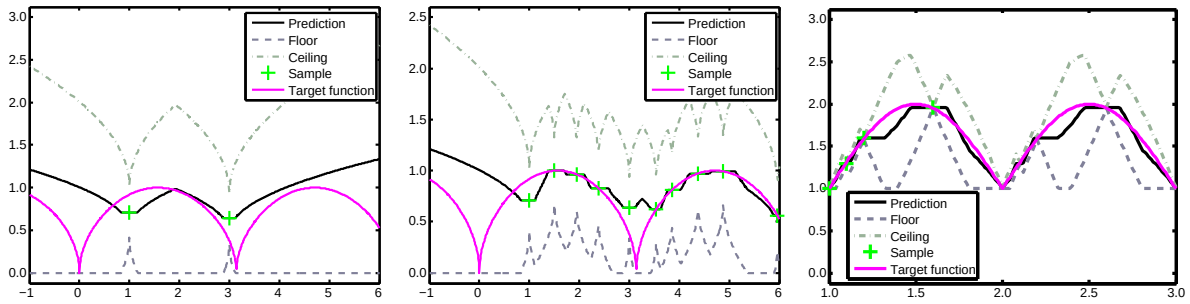
$$\hat{\mathfrak{v}}_{\mathfrak{n},\mathfrak{j}}(x) := \max\{w_\mathfrak{u}(x), w_\mathfrak{l}(x)\} \big(\min\{\bar{B}_j(x), \mathfrak{u}_{n,j}(x)\} - \max\{\underline{B}_j, \mathfrak{l}_{n,j}(x)\}\big). \tag{4.7}$$

Here, $\mathfrak{u}_n, \mathfrak{l}_n : \mathcal{X} \to \mathbb{R}^m$ are called ceiling and floor functions, respectively. Their jth component functions are given by $\mathfrak{u}_{n,j}(x) := \min_{i \in \mathcal{I}_n(x)} \tilde{f}_{i,j} + L_j\, \mathfrak{d}_\mathcal{X}^p(x, s_i) + \varepsilon_j(x)$ and $\mathfrak{l}_{n,j}(x) := \max_{i \in \mathcal{I}_n(x)} \tilde{f}_{i,j} - L_j\, \mathfrak{d}_\mathcal{X}^p(x, s_i) - \varepsilon_j(x)$, respectively. Here $p \in \mathbb{R}, L \in \mathbb{R}^m$ and functions $\underline{B}, \bar{B}, \varepsilon$ are parameters that have to be specified in advance. To disable restrictions of boundedness, it is allowed to specify the upper and lower bound functions to constant ∞ or $-\infty$, respectively. Function $\hat{\mathfrak{f}}_\mathfrak{n}$ is the predictor that is to be utilised for predicting/inferring function values at unseen inputs. Function $\hat{\mathfrak{v}}_\mathfrak{n}(x)$ quantifies the uncertainty of prediction $\hat{\mathfrak{f}}_\mathfrak{n}(x)$. An inference rule over function values with predictor $\hat{\mathfrak{f}}_\mathfrak{n}$, with component functions as per Eq. 4.6, and uncertainty quantification $\hat{\mathfrak{v}}_\mathfrak{n}$, whose component functions are given by Eq. 4.7, will be called a kinky inference rule.

Remark 4.2.5 (Choice of weights). The for each output component j, the prediction about a function value $\hat{\mathfrak{f}}_\mathfrak{n}(x)_j$ is formed by choosing a value between $\mathfrak{l}_{n,j}(x) = \inf_{\phi \in \mathcal{K}_{post}} \phi_j(x)$ and $\mathfrak{u}_{n,j}(x) = \sup_{\phi \in \mathcal{K}_{post}} \phi_j(x)$. The weights $w_\mathfrak{u}(x), w_\mathfrak{l}(x)$ determine where in the interval the prediction value is picked. Remember, $\hat{\mathfrak{v}}_{n,j}(x)$ quantifies the radius of the worst-case symmetric error interval around this prediction. From Eq. 4.7 one can see that this error interval is smallest for $w_\mathfrak{u}(x) = w_\mathfrak{l}(x) = \frac{1}{2}, \forall x \in \mathcal{X}$. This fact is the motivation for choosing this constant as the default value for the weights.

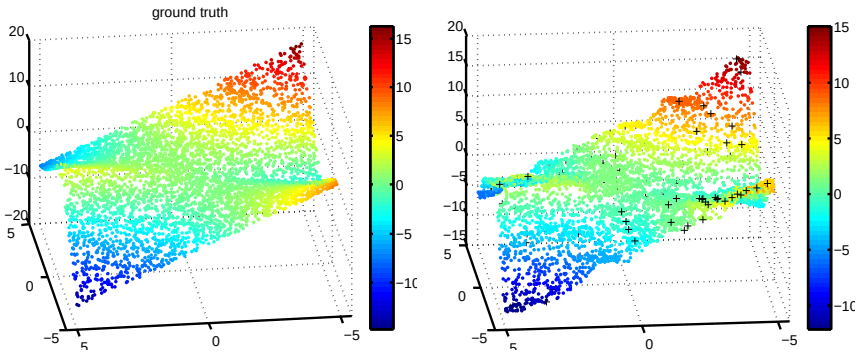
Above, we have mentioned that, for the time being, our emphasize is on inference that takes into account all available data. Translated to the kinky inference framework, this corresponds to choosing $\mathcal{I}_n(x) := \{1, \ldots, N_n\}, \forall x \in \mathcal{X}$. Such a kinky inference rule that takes into account the full data set when making inferences shall henceforth be referred to as a *full kinky inference* rule (*KI*).

To develop a first feel for this kinky inference rule, we plotted some examples of ground-truth functions and pertaining predictions in Fig. 4.1 and Fig. 4.2. Firstly, we can see that the predictions involve kinks. This is due to the minimisations and maximisations that occur in the computation of the ceiling and floor



(a) KI prediction with two examples. (b) Refined prediction. (c) Using a periodic pseudo-metric.

Figure 4.1.: Examples of different target functions and full kinky inference predictions. **Fig. 4.1(a) and Fig. 4.1(b)** show kinky inference performed for two and ten noisy sample inputs of target function $x \mapsto \sqrt{|\sin(x)|}$ with observational noise level $\varepsilon = 0.3$. The floor function folds in knowledge of the non-negativity of the target function ($\underline{B} = 0$). The predictions were made based on the assumption of $p = \frac{1}{2}$ and $L = 1$ and employed the standard metric $\mathfrak{d}_{\mathcal{X}}(x, x') = |x - x'|$. As can be seen from the plots, the function is well interpolated between sample points with small perturbations. Furthermore, the target is within the bounds predicted by the ceiling and floor functions. **Fig. 4.1(c)** depicts a prediction folding in knowledge about the periodicity of the target function by choosing a pseudo-metric $\mathfrak{d}_{\mathcal{X}} = \left|\sin(\pi\,|x - x'|\right|$. The prediction performs well even in unexplored parts of input space taking advantage of the periodicity of the target. This time, there was no observational error and the target function was $x \mapsto |\sin(\pi x)| + 1$.



(a) Wingrock dynamics. (b) Inferred model.

Figure 4.2.: Example of a prediction of the KI rule on two-dimensional input space. *Left plot:* the target function being a patch of the wingrock dynamics we will study in Sec. 4.4.2. *Right plot:* the predictions inferred on the basis of 100 sample points using metric $\mathfrak{d}_{\mathcal{X}}(x, x') = \|x - x'\|_{\infty}$ and $p = 1, L = 2.5$.

functions which introduce kinks into the prediction signal. This property provided the motivation behind the term "kinky inference".

Secondly, we note that in all the examples, the uncertainty bounds given by the ceiling and floor functions were conservative. That is, the target remained contained in them. An example where we see how the bounds shrink and allow the predictor to approach to the ground-truth (up to the observational error tube) will be given in Sec. 4.4.1. There we consider an agent that utilises kinky inference to learn a drift vector field it is immersed in. Also, note the choice of distance (pseudo-) metric $\mathfrak{d}_{\mathcal{X}}$ can affect the predictions. In the example depicted in Fig. 4.1(c), a metric was chosen that took into account a priori knowledge about periodicity of the target function. This allowed predictions to be accurate even in unexplored parts of the input space.

Remark 4.2.6 (Remarks on the relationship to kernels and distance metric learning). Unless altered by $\bar{B}$, $\underline{B}$, the kinky inference predictor assigns similar function values to inputs that are similar (i.e. close) with respect to the chosen input metric. Therefore, the choice of metric is important. Metrics are measures of dissimilarity and therefore, are closely related to similarity measures such as kernels. In much of the recent machine learning literature, kernel functions are utilised to perform inference. Note, a kernel function is a measure of similarity often based on quantities inversely related to metrics (which are dissimilarity measures). In order to gain inspiration about metrics or pseudo-metrics suitable for an application at hand, we can draw on the parts of the kernel learning literature that are concerned with the tailoring of kernels to specific applications (for a recent overview of *kernel engineering*, including periodic ones, refer to [88]). As we have seen in the example depicted in Fig. 4.1(c), we might consider folding in knowledge of periodicity of the underlying function class. However, as a future direction, one might consider exploring to harness more sophisticated automated kernel construction techniques (e.g. [10,88,134]) and distance metric learning methods (see [145]) that have been developed throughout the machine learning community in recent years.

4.2.3. Theoretical guarantees

The examples depicted in the plots of Fig. 4.1 and Fig. 4.2 seem to be consistent with what we would expect from an inference mechanism that is conservative and monotonically convergent and hence, might satisfy the desiderata we stated in Def. 4.2.2. We will now give conditions under which we can guarantee the desiderata to be fulfilled.

Of course, the validity of our guarantees depends on the validity of the assumption that the target is contained in the a priori hypothesis space $\mathcal{K}_{prior}$. Inspection of the prediction rules reveals that the minimisation and maximisations eliminate all those function values that do not adhere to boundedness conditions (imposed by $\underline{B}$, $\bar{B}$) and conditions on Hölder continuity up to a margin of error given by ε.

That this is exactly the right class for which our desiderata hold is asserted by the following theorem:

Theorem 4.2.7. *Assume all (a priori) possible targets $f : \mathcal{X} \to \mathcal{Y}$ are given by the class*

$$\mathcal{K}_{prior} = \left\{ f : \mathcal{X} \to \mathcal{Y} \subseteq \mathbb{R}^m | f \in \mathfrak{H}_{\mathfrak{d}_\mathcal{X}}(L, p) \cap \mathcal{B} \right\} \tag{4.8}$$

where $\mathfrak{H}_{\mathfrak{d}_\mathcal{X}}(L,p) = \{f : \mathcal{X} \to \mathcal{Y} \subset \mathbb{R}^m \,|\forall j \in \{1,\dots,m\} \forall x, x' \in \mathcal{X} : |f_j(x) - f_j(x')| \leq L_j \, \mathfrak{d}_\mathcal{X}^p(x,x')\}$ denotes the class of $L-p$- Hölder continuous functions with respect to metric $\mathfrak{d}_\mathcal{X}$. Let $\mathcal{B} := \{\phi : \mathcal{X} \to \mathcal{Y} \,|\, \forall x \in \mathcal{X}, j \in \{1,...,m\} : \underline{B}_j(x) \leq \phi_j(x) \leq \bar{B}_j(x)\}$ be the set of all functions bounded component-wise between functions $\underline{B}$, $\bar{B} : \mathcal{X} \to \mathbb{R}_\infty^m$, where we will always define $\mathbb{R}_\infty := \mathbb{R} \cup \{-\infty, \infty\}$. Furthermore, we assume $w_\mathfrak{u} \equiv w_\mathfrak{l} \equiv \frac{1}{2}$ and $\mathcal{I}_n(x) = \{1,...,N_n\}, \forall x \in \mathcal{X}$. Then we have:

Then the full kinky inference rule as per Def. 4.2.4 is conservative, monotonically convergent (in the limit of dense sample grids) and optimal in the sense of Def. 4.2.2. That is, it satisfies Desiderata 1-4 as per Def. 4.2.2.

Proof. The statement is derived in the appendix, Sec. C.2.1. □

Computational effort

We can see from Def. 4.2.4 that the full kinky inference method can be classified as a supervised nonparametric learning method. That is, the computational effort for predicting grows with the data set size N_n. If the computational effort for computing the metric for a given input is in the class $\mathcal{O}(\delta)$ and the computational effort for computing the bounding functions $\underline{B}$ and $\bar{B}$ is in $\mathcal{O}(\beta)$ then the computational effort for computing the prediction functions is in $\mathcal{O}(N_n\delta + \beta)$. For example, in case the simple metric $\mathfrak{d}_\mathcal{X}(x,x') = \|x - x'\|$ is utilised and the bounding functions are constant we have $\mathcal{O}(\delta) = \mathcal{O}(d)$ and $\mathcal{O}(\beta) = \mathcal{O}(1)$, yielding a total computational effort of $\mathcal{O}(d\, N_n)$ for performing inference. Finally, as we have derived in Sec. C.4, the effort reduces to logarithmic asymptotic complexity $\mathcal{O}(\log N_n)$ in the one-dimensional case where the metric $\mathfrak{d}_\mathcal{X}(x,x') = |x - x'|$ is used and if $p = 1$ (cf. Rem. C.4.7). This can make a significant difference in terms of computational speed of the inference in the presence of large data sets. For higher dimensions, our *k*-nearest neighbour based method we will introduce in Sec. 4.2.6 might help to bring down computation to at least expected log-time as well.

4.2.4. Uncertain inputs

So far, we have considered output uncertainty but not input uncertainty. That is, we have assumed that the function values $f(s_i)$ at the sample inputs are uncertain but that query inputs x as well as sample inputs s_i are known exactly. However, in many applications this assumption typically is not met. For example, in Sec. 4.4, we will employ KI for learning and controlling dynamic systems. Here the target functions are state-dependent vector fields and the state measurements often are corrupted by noise (e.g. due to sensor

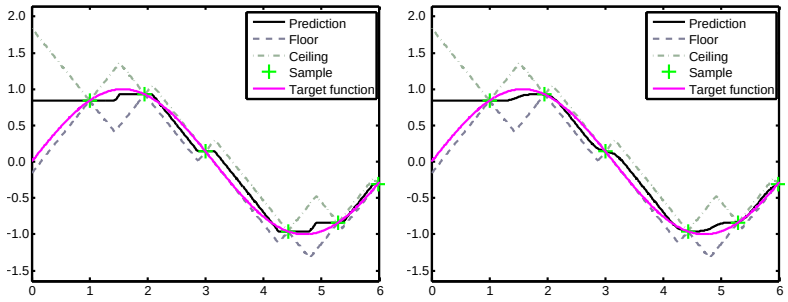
noise or delays) which can affect both training data and query inputs. In the latter case, the controller might query the KI method to predict a function value $f(x)$ based on the assumption of the plant being in state x while the plant may in fact be in a different state $x + \delta_x$. We would like to guarantee that our inference over the actual function value still is within the predicted bounds provided the input estimation error δ_x is known or known to be bounded. Fortunately, the Hölder property of the target can help to address the problem of quantifying the prediction error due to input uncertainty. For instance, consider $\mathfrak{d}(x, x') = \|x - x'\|$. By Hölder continuity, the error of the prediction of the jth output component of the target function f is bounded by $|f_j(x + \delta_x) - f_j(x)| \leq \|\delta_x\|^p$.

Firstly, assume the training inputs s_i were certain but the query inputs x during prediction are uncertain. Therefore, assuming we know an upper bound $\varepsilon_{\text{inp}}(x)$ on the input uncertainty, i.e. $\|\delta_x\|^p \leq \varepsilon_{\text{inp}}(x), \forall x$, we can add this term to the uncertainty estimate $\hat{\mathfrak{v}}(x)$ as per Eq. 4.7.

Secondly, consider the case were both the sample inputs s_i and the query inputs x are uncertain with the uncertainty being quantified by the function $\varepsilon_{\text{inp}}(\cdot)$. In this case, the uncertainty of the sample inputs affects the posterior hypothesis (i.e. the enclosure) over f and hence, the inferences over function values at all inputs. By the same argument as for the case of query noise, we accommodate for this fact leveraging the Hölder property. Instead of adjusting just the prediction uncertainty, we simply add $\varepsilon_{\text{inp}}(\cdot)$ to the error function $\varepsilon(\cdot)$ (that we have previously used to model observational error of the function values). That is, the observational uncertainty model has absorbed the input uncertainty. Predictions with uncertain query points can then be done as per Def. 4.2.4 (but with $\varepsilon_{\text{inp}}(\cdot) + \varepsilon(\cdot)$ in place of $\varepsilon(\cdot)$).

4.2.5. Ironing out the kinks - a smoothed inference rule

The inferred predictor of kinky inference as per Eq. 4.6 contains kinks. In certain applications, this can be disadvantageous. For instance, below, we will feed the predictor into a control signal. However, in applications such as robotics, smooth controllers are typically preferred over non-smooth ones since the latter give rise to increase wear and tear of the actuators. With this in mind we will consider a modification of the predictor which filters the prediction output through a smoothing filter as known from computer vision [17,99]. The idea is to convolve the predictor with a function that implements a low-pass filter and abates high frequencies thereby smoothing out rough transitions in the signal. For now, we will restrict the exposition to one-dimensional input and output spaces. However, extensions to multiple output dimensions result from following our steps through for each component function. Extensions to multi-dimensional input spaces follow by taking advantage of the corresponding filters developed for multi-dimensional signal processing (see e.g [198, 269]). As an example, for some $n \in \mathbb{N}, w_i \geq 0\,(i = 1, ..., n)$ and (typically small) step size



(a) KI on a sine curve. (b) SKI on a sine curve.

Figure 4.3.: Comparison of the kinky inference rule (left) versus the smoothed kinky inference rule (right) when predicting a sine curve. Note the smoothed out kinks of the SKI rule due to the application of the smoothing filter. Here we chose $\sigma = 0.1$.

$\sigma \in \mathbb{R} \geq 0$, consider the modified prediction rule (*smoothed kinky inference (SKI) rule*):

$$f_n^*(x) = \frac{1}{\sum_{i=-q}^{q} w_i} \sum_{i=-q}^{q} w_i \hat{\mathfrak{f}}_{\mathfrak{n}}(x + i\sigma). \tag{4.9}$$

Either $w_0 = 2$ and $w_{-1} = w_2 = 1$ or $w_{-2} = w_2 = 7, w_{-1} = w_1 = 26, w_0 = 41$ are common choices for discrete approximations of a Gaussian filter [17].

A comparison of the predictions made by the smoothed kinky inference rule (SKI) against the prediction function of the standard kinky inference rule is depicted in Fig. 4.2.5. As can be seen from the plot, in the SKI prediction the kinks have been smoothed out while all the predictions still are contained within the prediction bounds given by the floor and ceiling functions.

The question arises how the smoothing might affect the theoretical properties. Given that $\hat{\mathfrak{f}}_{\mathfrak{n}}$ is Hölder continuous with constant $L(\hat{\mathfrak{f}}_{\mathfrak{n}})$ and exponent p, we note that due to Lem. 2.5.6, the Hölder exponent and constant remain unchanged. For the constant, we apply point 4 and 9 of the lemma to see that $L(f_n^*) = \frac{1}{\sum_{i=-q}^{q} w_i} \sum_{i=-q}^{q} w_i L(\hat{\mathfrak{f}}_{\mathfrak{n}}) = L(\hat{\mathfrak{f}}_{\mathfrak{n}}) \frac{1}{\sum_{i=-q}^{q} w_i} \sum_{i=-q}^{q} w_i = L(\hat{\mathfrak{f}}_{\mathfrak{n}})$.

Secondly, the question arises what the maximal discrepancy between $\hat{\mathfrak{f}}_{\mathfrak{n}}$ and f_n^* is. This question is of importance to make sure that the prediction still is within the bounds given by the ceiling and floor functions. To this end, assume $\hat{\mathfrak{f}}_{\mathfrak{n}}$ is Hölder with constant L and exponent p. For simplicity assume we utilise Gaussian smoothing for the SKI rule with $n = 1$, $w_{-1} = w_1 = 1, w_0 = 2$. The deviation of this resulting smoothed

predictor f_n^* from the kinky inference predictor $\hat{\mathfrak{f}}_\mathfrak{n}$ can be bounded as follows

$$\left|\hat{\mathfrak{f}}_\mathfrak{n}(x) - f_n^*(x)\right| = \left|\hat{\mathfrak{f}}_\mathfrak{n}(x) - \frac{1}{w_{-1}+w_0+w_1}(w_{-1}\,\hat{\mathfrak{f}}_\mathfrak{n}(x-\sigma) + w_0\,\hat{\mathfrak{f}}_\mathfrak{n}(x) + w_1\,\hat{\mathfrak{f}}_\mathfrak{n}(x+\sigma)\right| \tag{4.10}$$

$$= \frac{1}{4}\left|2\,\hat{\mathfrak{f}}_\mathfrak{n}(x) - \hat{\mathfrak{f}}_\mathfrak{n}(x-\sigma) - \hat{\mathfrak{f}}_\mathfrak{n}(x+\sigma)\right| \tag{4.11}$$

$$\leq \frac{1}{4}\left|\hat{\mathfrak{f}}_\mathfrak{n}(x) - \hat{\mathfrak{f}}_\mathfrak{n}(x-\sigma)\right| + \frac{1}{4}\left|\hat{\mathfrak{f}}_\mathfrak{n}(x) - \hat{\mathfrak{f}}_\mathfrak{n}(x+\sigma)\right| \tag{4.12}$$

$$\leq \frac{L}{4}\,\mathfrak{d}_\mathcal{X}^p(x, x-\sigma) + \frac{L}{4}\,\mathfrak{d}_\mathcal{X}^p(x, x+\sigma). \tag{4.13}$$

If the metric is induced by a norm $\|\cdot\|$ the last expression simplifies to $\frac{L}{2}\,\|\sigma\|^p$. In order to ensure that uncertainty bound takes this error into account, we can add this error to the uncertainty quantification $\hat{\mathfrak{v}}_\mathfrak{n}$. This guarantees that the ground truth is contained within the error bounds around predictions f_n^*. Furthermore, since the bound is valid for all inputs $x \in \mathcal{X}$, the convergence guarantees established for kinky inference also hold for the smoothed version up to supremum norm error of at most $\sup_{x\in\mathcal{X}} \frac{L}{4}\,\mathfrak{d}_\mathcal{X}^p(x, x-\sigma) + \frac{L}{4}\,\mathfrak{d}_\mathcal{X}^p(x, x+\sigma)$ (which in the norm-based metric case equals $\frac{L}{2}\,\|\sigma\|^p$).

4.2.6. Kinky inference based on sample subsets

In Sec. 4.2.1, we have stated the optimal conservative inference over functions as the computation of the posterior set

$$\mathcal{K}_{post}(\mathcal{D}_n) = \mathcal{K}_{prior} \cap \mathcal{K}(\mathcal{D}_n). \tag{4.14}$$

As mentioned above, computing a prediction and bounds on the basis of $\mathcal{K}_{prior} \cap \mathcal{K}(\mathcal{D}_n)$ might involve computation that grows linearly with the data set size $N_n = |\mathcal{D}_n|$. Therefore, computational limitations may impose the requirement to base our inferences $\hat{\mathfrak{f}}_\mathfrak{n}(x)$, $\hat{\mathfrak{v}}_\mathfrak{n}(x)$ over function values on a subset of the available data. Retaining the requirement of conservatism, from the function inference point of view, this translates to having

$$\mathcal{K}_{post}(\mathcal{D}_n) = \mathcal{K}_{prior} \cap \tilde{\mathcal{K}}(\mathcal{D}_n) \text{ where } \tilde{\mathcal{K}}(\mathcal{D}_n) \supset \mathcal{K}(\mathcal{D}_n). \tag{4.15}$$

Having learned, i.e. performed inference over functions, we can use this to predict function values conservatively by considering all the function values within the posterior class which could be assumed at a given query input $x \in \mathcal{X}$.

In the remainder of this section, we will consider two examples of such reduced data approaches in the kinky inference context. The first will make inferences on the basis of a budget of maximally $\bar{N}$ sample points. That is, $\tilde{\mathcal{K}}(\mathcal{D}_n) = \mathcal{K}(\bar{\mathcal{D}})$ for some $\bar{\mathcal{D}} \subset \mathcal{D}_n$ of size $\left|\bar{\mathcal{D}}\right| \leq \bar{N}$. Basing inference on a fixed finite sample subset, this method provides an upper bound on the maximal prediction run-time. On the flip-side,

drawing inferences about the function value for each query based on a fixed finite subset of the data of maximal size imposes the limitation that the maximal uncertainty in general will not vanish even when total available data $\mathcal{D}_n$ samples the target function on an increasingly dense grid as $n \to \infty$. The latter issue is avoided by the second example of a data subset inference method. Here, we will compute the predictions $\hat{\mathfrak{f}}_{\mathfrak{n}}(x)$ and uncertainty quantifications $\hat{\mathfrak{v}}_{\mathfrak{n}}(x)$ based on a query-dependent subset $\mathcal{S}_n^{\text{knn}}(x) \subseteq \mathcal{D}_n$ that contains exactly those sample points $(s_i, \tilde{f}_i, \varepsilon_i) \in \mathcal{D}_n$ whose inputs s_i are the up to k nearest neighbours (*kNN*) of the query $x \in \mathcal{X}$ in grid G_n. That is, for a given query $x \in \mathcal{X}$, we perform inference about the target function value based on the function inference method with $\tilde{\mathcal{K}}(\mathcal{D}_n) = \mathcal{K}\big(\mathcal{S}_n^{\text{knn}}(x)\big)$.

We can define the *kNN* subset $\mathcal{S}_n^{\text{knn}} = \{p_i \in \mathcal{D}_n | i \in \mathcal{I}_n^{\text{knn}}(x)\}$ in terms of the index set mapping $\mathcal{I}_n^{\text{knn}}(\cdot) : \mathcal{X} \to 2^{\{1,\dots,N_n\}}$ of an input to the set of indices in data set $\mathcal{D}_n$ pertaining to the k nearest neighbours of the input in grid G_n.

Definition 4.2.8 (Formal definition of kNN index subsets $\mathcal{I}_n^{\text{knn}}(x)$)**.** *Formally, for each $x \in \mathcal{X}, \mathcal{D}_n \subset \mathcal{X}$, $k \geq 1$, the subset of indices $\mathcal{I}_n^{knn}(x) \subseteq \{1, ..., N_n\}$ of data points in $\mathcal{D}_n$ can be recursively defined as follows: Let $\pi : \{1, ..., N_n\} \to \{1, ..., N_n\}$ be a permutation such that $\forall i \leq j : \mathfrak{d}_{\mathcal{X}}(x, s_{\pi(i)}) \leq \mathfrak{d}_{\mathcal{X}}(x, s_{\pi(j)})$. That is, π orders the indexes according to the distance of the corresponding grid input points to query x. This allows us to define a set of the up to k nearest neighbour indices as $\mathcal{I}_n^{knn}(x) = \big\{\pi(1), \dots, \pi(\min\{k, N_n\})\big\}$.*

In spite of the fact that inference will be based on a finite sample of size $|\mathcal{S}_n| \leq k < \infty$, the necessity to find the k nearest neighbours for each query causes computational prediction effort to grow with the data size $\mathcal{D}_n$. However, by employing efficient (approximate or exact) k-nearest neighbour search methods, we expect that this complexity can be significantly reduced in practice. Note, the *full* inference method that utilises the entire data set $\mathcal{D}_n$ regardless of its size (i.e. $\tilde{\mathcal{K}}(\mathcal{D}_n) = \mathcal{K}(\mathcal{D}_n)$) is a limit case of both data subset inference methods: the *full* method has the input-output behaviour of the *kNN*-based method with $k = \infty$ or of a "finite sample size" approach with maximal sample size $\bar{N} = \infty$.

Above, we have discussed the full, finite data subset and k-nearest neighbour based approaches to conservative inference over functions. In the context of the kinky inference class over function values, these approaches find their manifestations by choosing the index sets $\mathcal{I}_n(x)$ appropriately as follows:

Definition 4.2.9 (Full, finite-sample and kNN kinky inference rules)**.** *The choice of index set mapping $\mathcal{I}_n : \mathcal{X} \to 2^{\{1,\dots,N_n\}}$ can influence the inference and computational effort. We consider the following choices:*

1. *If $\mathcal{I}_n(x) = \{1, \dots, N_n\}, \forall x \in \mathcal{X}$, we refer to the inference rule as a **(full) kinky inference (KI)** rule.*
2. *If the index set is a fixed set of maximally $\bar{N}$ elements, i.e. for $\mathcal{D}_n, |\mathcal{D}_n| = N_n \in \mathbb{N}$ we have $\exists \mathcal{I} \subset \{1, \dots, N_n\} \forall x \in \mathcal{X} : \mathcal{I}_n(x) = \mathcal{I} \wedge |\mathcal{I}| = \min\{N_n, \bar{N}\}$, we refer to the inference rule as a **budgeted kinky inference (b-KI)** rule.*

3. *For a given data set $\mathcal{D}_n$, natural number $k \geq 1$ and query input $x \in \mathcal{X}$, we define $\mathcal{I}_{knn}(x; \mathcal{D}_n) \subset \{1, \ldots, N_n\}$ to be the set of indices of the up to k-nearest neighbours of x in grid G_n (cf. Def. 4.2.8). If $\mathcal{I}_n(x) = \mathcal{I}_n^{knn}(x)$, then we refer to the resulting kinky inference rule as a* ***k-nearest neighbour kinky inference (kNN-KI)*** *rule.*

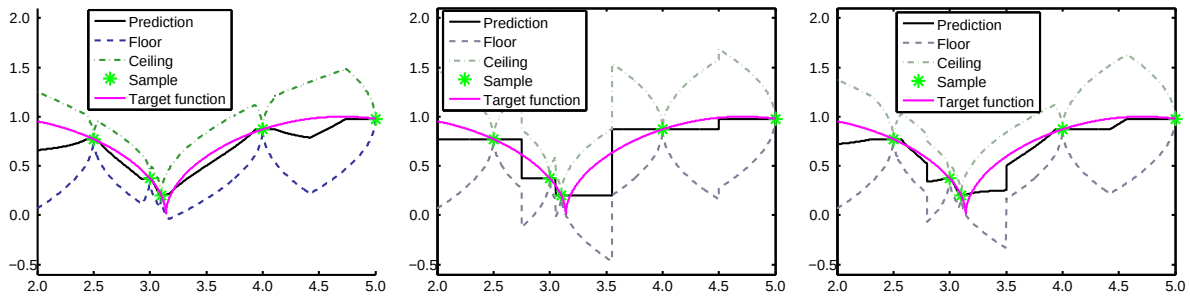
The difference between these three types of kinky inference rules is how much and which of the available data is utilised in the inference over function values. The full method has been in the centre of attention thus far. In the full method (KI), the inference benefits from the entire sample. This guarantees the lowest uncertainty quantification bound (cf. Thm. 4.2.7) but also means that the computational prediction effort inevitably grows linearly with the data. Throughout the subsequent sections of this work, this approach will be the default method.

When the necessity arises to make predictions under a computational budget, we may choose to select a fixed sub-sample of at most $\bar{N}$ data points on the ground of which all subsequent inferences about the target's values will be made (giving rise to a *b-KI* rule). Here, of course the problem arises which data points to select. Considering this data selection problem will be done in the course of future work.

A more flexible choice is to select the sub-sample the prediction is based upon for each query point individually at runtime. In *kNN-KI*, the inferences are based on the sub-sample that contains exactly the up to k nearest neighbours. The intuition behind this choice is that, in the full method (KI), data points with inputs that are closer to a query input will tend to have a higher chance of influencing the ceiling and floor functions (and hence the prediction and uncertainty quantification) while remote data points will tend to exert no influence at all. In combination with efficient methods for computing k-nearest neighbours [12, 93, 100, 213], we believe that the *kNN-KI* approach will be able to be able to compute predictions efficiently even with large data sets. However, investigating the properties and tradeoffs of this method in combination with different *kNN* search methods will have to be deferred to future work. For the time being, all of the results presented in the subsequent sections of this thesis will be based on the full KI method.

As a first illustration of the kNN-KI method consider Fig. 4.4. Depicted are the predictions of the full KI method (Fig. 4.4(a)) in comparison to the prediction results obtained from the *kNN-KI* approach for $k = 1$ (Fig. 4.4(b)) and $k = 2$ (Fig. 4.4(c)). The implementation of nearest neighbour search was based on a k-d tree [28]. The parameter settings of the kinky inference rules were $L = 1, p = \frac{1}{2}$ and, as usual, $w_{\mathfrak{u}} = w_{\mathfrak{l}} = \frac{1}{2}$.

Despite its simplicity, the examples expose some fundamental differences between the full and the nearest-neighbour based kinky inferences. While, appealing to Lem. 2.5.6, we can show that the former always results in a Hölder continuous prediction and has minimal uncertainty bounds $\hat{\mathfrak{v}}_{n,j}$, the abrupt switching between nearest neighbour zones can introduce discontinuities in the predictor of the kNN-based kinky inference methods. Furthermore, the kNN approach may neglect to take into account sample points that can reduce the



(a) KI inference. (b) 1NN-KI inference. (c) 2NN-KI inference.

Figure 4.4.: Full kinky inference (KI) in comparison to 1NN-KI and 2NN-KI for ground truth target function $x \mapsto \sqrt{|\sin x|}$ based on a sample on grid $G_n = \{2.5, 3, 3.1, 4, 5\}$. In comparison to the kNN-KI methods, the full kinky inference approach (KI) has the benefit of tighter uncertainty bounds (given by the floor and ceiling functions) as well as offering a (Hölder) continuous predictor.

uncertainty quantifications of a prediction. Therefore, the uncertainty bounds generally can be looser than those provided by full kinky inference. Both facts can be observed in the plots. As mentioned above, future work will investigate whether these downsides can be compensated by the lower expected prediction effort which would allow kNN-KI to entertain larger data sets than the full KI method even when predictions have to be made under real-time requirements. However, throughout the remainder of this thesis, we will solely focus on the full KI method.

4.3. Uncertain Hölder constants

Throughout most of the previous sections we have assumed that the Hölder constant was known to us. That is, we assumed $f \in \mathcal{K}_{prior} \subseteq \mathfrak{H}_{\mathfrak{d}_\mathcal{X}}(L, p)$ for some known constant $L > 0$. However, in many cases, we may be uncertain about the best Hölder constant. There are many ways to quantify one's uncertainty and how to update ones belief over such constant in the light of newly available data. We will adopt two different approaches. The first approach, which we outline in Sec. 4.3.1, is a Bayesian one where subjective probabilistic beliefs are entertained and updated employing probabilistic calculus. Inference on bounds only hold true with given probabilities relative to the posterior distributions over the actual Hölder constants. A drawback is, as typical in the Bayesian framework, that it is not always practicable to specify a priori beliefs in distributional form and that complete "ignorance priors" may not exist within the strict framework of measure theory [123]. Furthermore, the Bayesian approach is known to fall prey to prejudice in that it cannot recover from false beliefs of impossibility. That is, if an observed data point provides evidence for the validity of an event that has zero measure under the prior, the posterior will be unable to adjust and also ascribes zero-measure to the event, even if most strongly supported by the evidence.

The second method, which we develop in Sec. 4.3.2, side-steps any such issues by adopting a lazy approach. Once a Hölder constant is declared by the user, the constant is assumed to be true, unless new data

arrives that is detected to be in conflict with the previously entertained Hölder constant. As pointed out in Rem. 4.2.1, this procedure can be seen as a lazy relaxation of the prior hypothesis space.

For simplicity we will restrict our exposition to real-valued target functions. However, in the case of multi-output functions ($m > 1$), the methods presented in this section can simply be applied to each component function of the target f.

4.3.1. A Bayesian treatment of probabilistically uncertain Hölder constants

Throughout the previous section we have assumed that the Hölder constants were known to us. However, in many cases, we may be uncertain about the best Hölder constant. We assume the uncertainty is expressed as a distribution. Following the Bayesian point of view, we interpret probability densities over events as subjective degrees of beliefs that these events hold true. In the following subsection, we describe how to utilize the densities over the best Hölder constant $L^* \in \mathbb{R}_+$ to yield a (once again, conservative) bound on the probability of our functions estimates and integrands not being conservative. After that, we will address the question of how to update our beliefs over L^* in the light of new function evaluations in a Bayesian manner.

For simplicity, we will assume that $\mathfrak{d}_{\mathcal{X}}$ is a metric on the input space, that the target function f is real-valued, that there are no observational errors and that there are no a priori bounds. That is, $\varepsilon_i = 0$, $\forall i$ and $\underline{B} = -\infty$, $\bar{B} = \infty$. Furthermore, we tacitly assume a Hölder exponent p is known and fixed. The update of uncertain beliefs over both L and p simultaneously is deferred to future work.

Let $\pi : \mathbb{R}_+ \to \mathbb{R}_+$ be a density encoding our belief over best Hölder constant L^* of target function f. In the absence of bounds and observational error, we define the Hölder enclosure $\mathcal{E}^{\mathfrak{u}_n}_{\mathfrak{l}_n}(\ell) = \{\phi \,|\, \phi(x) \in \hat{H}_n(x)\}$ where $\hat{H}_n$ is the uncertainty hyperrectangle defined above (see Eq. 4.2), computed on the basis of Hölder constant $L = \ell$. Assume we construct a Hölder enclosure $\mathcal{E}^{\mathfrak{u}_n}_{\mathfrak{l}_n}(\ell)$ based on choosing $\ell \in \mathbb{R}_+$ as a Hölder constant. How large do we need to choose ℓ to guarantee that f is completely contained in the enclosure? Since we are uncertain about the true Hölder constant, this question can only be framed probabilistically. That is, for a given certainty threshold $\theta \in (0, 1)$ we desire to find $\ell > 0$ such that

$$\theta \leq \Pr[f \in \mathcal{E}^{\mathfrak{u}_n}_{\mathfrak{l}_n}(\ell)].$$

Theorem 4.3.1. *Let $\mathfrak{u}_n, \mathfrak{l}_n$ be a valid ceiling and floor function, respectively. Let $P : t \mapsto \int_{x \leq t} \pi(x)\, dx$ our density's cumulative distribution function (cdf). We have,*

$$P(\ell) \geq \theta \Rightarrow \theta \leq \Pr[f \in \mathcal{E}^{\mathfrak{u}_n}_{\mathfrak{l}_n}(\ell)].$$

Proof. Let L^* denote the best Hölder constant of function f.

$\forall \ell \geq L \geq 0 : \Pr[f \in \mathcal{E}^{\mathfrak{u}_n}_{\mathfrak{l}_n}(\ell) | L^* = L] = 1$. Hence, $\Pr\big[f \in \mathcal{E}^{\mathfrak{u}_n}_{\mathfrak{l}_n}(\ell)\big] = \int_0^{\ell} \Pr\big[f \in \mathcal{E}^{\mathfrak{u}_n}_{\mathfrak{l}_n}(\ell) | L^* =$

$L\big]\ \mathrm{d}P(L) + \int_\ell^\infty \Pr\big[f \in \mathcal{E}^{\mathfrak{u}_n}_{\mathfrak{l}_n}(\ell) | L^* = L\big]\ \mathrm{d}P(L) = \int_0^\ell\ \mathrm{d}P(L) + \int_\ell^\infty \Pr\big[f \in \mathcal{E}^{\mathfrak{u}_n}_{\mathfrak{l}_n}(\ell) | L^* = L\big]\ \mathrm{d}P(L) \geq \int_0^\ell\ \mathrm{d}P(L) = P(\ell).$ □

That is to say, in order to guarantee conservativeness of our enclosure, all we need to do is to compute it on the basis of a fixed Hölder constant that is just large enough such that $P(\ell) = \int_0^l \pi(L)\, dL \geq \theta$. Notice, if the integral cannot easily be determined in closed-form, but a Hölder constant for density π is known, we can employ our Hölder quadrature method described previously (based on a known constant), to evaluate $P(\ell)$ conservatively by finding a lower bound on it.

A Bayesian update rule

Define $\mathfrak{d}_\mathcal{Y}(f_1, f_2) = |f_1 - f_2|$ and let $\mathfrak{d}_\mathcal{X}(x, y)$ be some input space metric.

Assume we hold a prior belief over the best Hölder constant L^* encoded as a density $\pi_0 : I \subset \mathbb{R}_+ \to [0, 1]$. Assume we are given a sample $D = \{(s_i, f_i)\}_{i=1,\ldots,N}$ of input-function value pairs, $x_i \in \mathcal{X}$, $f_i = f(s_i) \in \mathcal{Y}, \forall i$, the question arises of to calculate a posterior in the light of the new data.

Since we assume $f : \mathcal{X} \to \mathcal{Y} \subseteq \mathbb{R}$ is Hölder we have $\frac{\mathfrak{d}_\mathcal{Y}(f_i, f_j)}{\mathfrak{d}_\mathcal{X}(s_i, s_j)} \leq L^*, (i, j = 1, ..., N, s_i \neq s_j)$ where L^* is the best Hölder constant of f. So, our observations allow us to infer that the *empirical Hölder constant*

$$L_D := \max_{i,j,i\neq j} \frac{\mathfrak{d}_\mathcal{Y}(f_i, f_j)}{\mathfrak{d}^p_\mathcal{X}(s_i, s_j)} \tag{4.16}$$

is a lower bound on the best Hölder constant.[3] Note, the computation of an update of this quantity can be done with computational effort linear in the number of pre-existing quantities. That is, assuming a pre-existing data set D and that we make an additional observation s_{N+1}, f_{N+1} which is incorporated into the updated data set $D' = D \cup \{(s_{N+1}, f_{N+1})\}$. Instead of computing $L_{D'}$ from scratch as per Eq. 4.16, we can leverage:

$$L_{D'} = \max\{L_D, L'\}, \text{ where } L' := \max_{i=1,\ldots,N} \frac{\mathfrak{d}_\mathcal{Y}(f_{N+1}, f_i)}{\mathfrak{d}^p_\mathcal{X}(s_{N+1}, s_i)} \tag{4.17}$$

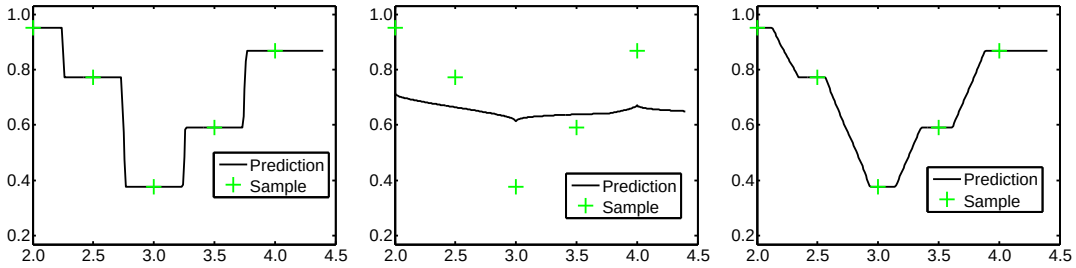
which can be computed in $\mathcal{O}(dN)$ where as before, $d = \dim \mathcal{X}$. The remaining question is how to derive a posterior over L^* based on this newly observed lower bound.

By Bayes theorem, we can write

$$\pi(L|L_{D'}) = \frac{\pi(L_{D'}|L)\, \pi_0(L)}{\int_I \pi(L_{D'}|L)\, \pi_0(L)\ \mathrm{d}L}.$$

If we are uncertain about which empirical Hölder constant we observe given the real Hölder constant

[3] In the context of a special Lipschitz interpolation rule, Beliakov [24] considered this quantity as an estimate of the Lipschitz constant.



(a) Inferred model with $L = 10$. (b) Inferred model with $L = 0.1$. (c) Inferred model with $L = 1$.

Figure 4.5.: Example of the predictions of the values of ground-truth function $x \mapsto \sqrt{|\sin(x)|}$ using varying Hölder constants. For all predictions, a standard metric $\mathfrak{d}_{\mathcal{X}}(x, x') = |x - x'|$ and a Hölder exponent of $p = \frac{1}{2}$ was employed. The global Hölder constant (and hence, the optimal constant) was $L = 1$.

$L^* = L$ we set the Likelihood function

$$\pi(L_{D'}|L) = \begin{cases} 0, L_{D'} > L \\ \\ \frac{1}{L}, \text{otherwise.} \end{cases} .$$

Of course, if the definite integrals of prior π_0 are not known in closed form, we need to approximate numerically. Depending on knowledge of smoothness properties we can either employ standard methods such as Gaussian quadrature or, if necessary, utilise our Hölder quadrature methods to that effect.

4.3.2. A lazy update rule

We assume there is a global Hölder constant that is uncertain. In the presence of uncertainty it is important to be able to update the Hölder constant L in the light of the data. After all, if chosen too conservatively high, the regularity properties of the actual ground truth will be lost and yield predictions that are too steep and, for large L, yield predictors resembling step functions (cf. Fig 4.5(a)). And, if decision making takes the error bounds provided into account, the unnecessary uncertainty may result in overly cautious behaviour and hence, poor performance. On the other hand, underestimating L may even be worse. If L is chosen too small then the error bounds are off and predictions may be unable to accurately infer the true shape of the function (cf. Fig 4.5(b)).

Above we have outlined a Bayesian treatment in the face of probabilistic uncertainty. This is an adequate choice if one's uncertainty can be specified by a prior and if the necessary integrals can be computed efficiently.

In practice, both can be difficult and we might resort to a more pragmatic approach which we will outline in this subsection. Furthermore, the Bayesian approach can fall prey to stubbornness and cannot recover from false beliefs of impossibility. That is, if an observed data point provides evidence for the validity of an event

that has zero measure under the prior the posterior will be unable to adjust and also ascribe zero-measure to the event, even if most strongly supported by the evidence.

To avoid underestimating the Hölder constant without resorting to overly conservative choices of L it might be possible to start with a good estimate of L that is not overly conservative and to update this lazily, that is, only if required. Owing to this lazy, reactive update behaviour we will refer to the resulting approach as a *lazy* update rule.

Assume we are given a sample $D_n := \{(s_i, \tilde{f}_i, \varepsilon_i) | i = 1, \ldots, N_n\}$ of the real-valued target $f : \mathcal{X} \to \mathbb{R}$ and, for simplicity, assume a uniform observational error bound $\varepsilon = \varepsilon(s_i) \in \mathbb{R}_{\geq 0}, \forall i$ and that $D_{n+1} = D_n \cup \{(s_{N_n+1}, f_{N_n+1}, \varepsilon)\}$. For ease of notation, we assume that $\mathfrak{d}^p_{\mathcal{X}}(s_i, s_j) > 0$ for $i \neq j$. This is guaranteed to be the case if $\mathfrak{d}_{\mathcal{X}}$ is a metric and the sample does not contain multiple sample instances of the target function at the same input and consequently, $s_i \neq s_j$ for $i \neq j$. Furthermore, we assume to entertain a deterministic belief in $L_n > 0$ that is consistent with the data. That is, $L_n \geq L_{D_n} = \max_{i,j=1,\ldots,N_n, i \neq j} \frac{\mathfrak{d}_{\mathcal{Y}}(f_i, f_j) - 2\varepsilon}{\mathfrak{d}^p_{\mathcal{X}}(s_i, s_j)}$ (cf. Eq. 4.16). The term 2ε imposes some tolerance margin that is necessary because a failure of consistency between the empirical constant L_{D_n} and one's belief L could be explained away by error in the observations.

Now assume that a new sample observation $(s_{N_n+1}, f_{N_n+1}, \varepsilon)$ arrives.

This new data point is evidence for constant L_n to be unduly low if it falls outside of the optimal enclosure $\mathcal{E}^*_n$. This is the case if

$\tilde{f}_{N_n+1} + \varepsilon < \mathfrak{l}^*_{N_n}(s_{N_n+1}) \vee \tilde{f}_{N_n+1} - \varepsilon > \mathfrak{u}^*_{N_n}(s')$ or equivalently, if $\tilde{f}_{N_n+1} + \varepsilon < \max_i \tilde{f}_i - \varepsilon - L_n \, \mathfrak{d}^p_{\mathcal{X}}(s_{N_n+1}, s_i) \vee \tilde{f}_{N_n+1} - \varepsilon > \min_i \tilde{f}_i + \varepsilon + L_n \, \mathfrak{d}^p_{\mathcal{X}}(s_{N_n+1}, s_i)$.

Equivalently, this means that

$$\left| \tilde{f}_{N_n+1} - \tilde{f}_i \right| > 2\varepsilon + L_n \, \mathfrak{d}^p_{\mathcal{X}}(s_{N_n+1}, s_i) \, \forall i \in \{1, ..., N_n\}. \tag{4.18}$$

This is a condition one could test for in $\mathcal{O}(d \, N_n)$ steps, provided the metric can be evaluated in $\mathcal{O}(d)$ steps. If the condition is met, we can change L to assume a value such that the condition is no longer met. To this end, we can choose the updated Hölder constant L_{n+1} to be

$$L_{n+1} := \max \left\{ L_n, \max_{i=1,\ldots,N_n} \frac{\left| \tilde{f}_{N_n+1} - \tilde{f}_i \right| - 2\varepsilon}{\mathfrak{d}^p_{\mathcal{X}}(s_{N_n+1}, s_i)} \right\} \tag{4.19}$$

which can be computed in $\mathcal{O}(d \, N_n)$ assuming the metrics can be evaluated in $\mathcal{O}(d)$.

Note, this result is consistent with the following alternative:

To compute $L_{D_{n+1}} = \max_{i,j=1,\ldots,N_{n+1}, i \neq j} \frac{\mathfrak{d}_{\mathcal{Y}}(f_i, f_j) - 2\varepsilon}{\mathfrak{d}^p_{\mathcal{X}}(s_i, s_j)}$ with the incremental update rule as per Eq. 4.17 and then to update L_n to the smallest value that is at least as large as the belief in the previous constant but also consistent with the empirical constant. That is, to update the constant to the smallest number L_{n+1} such

that $L_{n+1} \geq L_{D_{n+1}} \wedge L_{n+1} \geq L_n$. This coincides with the updated constant as per Eq. 4.19.

4.4. Kinky inference and model reference adaptive control

In Sec. 4.2, we have presented *kinky inference* (KI), a non-parametric machine learning method that entertains assumptions about Hölder continuity and boundedness and can handle interval-bounded noise. In this section, we illustrate the utility of the approach in the context of model learning of nonlinear dynamical systems and (tracking) control.

As a first test and introduction to the control applications, Sec. 4.4.1 considers an application of KI for model learning and control in a first-order system where the control can directly influence the velocity of a continuous-time plant. The bounds afforded by the KI rule are employed to provide a stability guarantee for a controlled uncertain system that meets the prior assumptions of the KI learner. In Sec. 4.4.2, we translate the controller to a more realistic second-order system. Simulations conducted in the context of wingrock tracking control demonstrate that our KI learning method is well suited for learning dynamic system models both offline and online and can outperform established machine learning methods such as Gaussian processes or radial basis function neural networks in model-reference adaptive control.

4.4.1. A kinky inference velocity controller with stability guarantees

Consider an agent that desires to control a plant with the first-order dynamics $\dot{x} = f(x) + g(x)u$ where $x \in \mathcal{X} \subset \mathbb{R}^d$ is the state ("position") and $u \in \mathbb{R}^d$ a control that can be chosen by the agent. Set $\mathcal{X}$ might be some bounded space. For instance, the bounds could be imposed by the confines of a room a robot moves in or, in heating control, might be a range of temperatures. For simplicity, we assume that $g(x) \in \mathbb{R}^{d\times d}$ is invertible and known for all $x \in \mathcal{X}$. By contrast, drift vector field f is assumed to be uncertain a priori and will be identified and inverted utilising our kinky inference rule.

It is our aim to design a controller such that state trajectory $x(\cdot)$ follows *reference* trajectory $x_{\text{ref}}(\cdot)$. For instance, in the one-dimensional case, if we desire the state to reach setpoint ξ we could set the reference derivative to $\dot{x}_{\text{ref}}(t) = w(\xi - x)$, i.e. $x_{\text{ref}}(t) = x(0)\exp(-wt) - \xi(1 - \exp(1 - Kt))$. That is, if the devised control law were successful in causing the state to follow the reference, the state would converge to the target ξ as $t \to \infty$. Control with a reference model is also referred to as *tracking*, since the objective is to follow (i.e. track) the reference signal x_{ref}. A special case, which we also have encountered in Ch. 3, is the task of *stabilisation*. Here, the reference signal x_{ref} is set to a constant setpoint $\xi \in \mathcal{X}$. That is, the only control objective is to reach, and stay close to, the one point ξ. Note, in setpoint control we have $\dot{x}_{\text{ref}} = 0$.

Having chosen a reference, we can attempt to solve our control task by utilising feedback linearisation in a model-reference adaptive control fashion [177] similar to the approach in Ch. 3 and Sec. 4.4.2. That is, we

define a control law:

$$u := g^{-1}(x)(\nu_{\text{fb}} - \nu_{ad} + \nu_{\text{ref}}). \tag{4.20}$$

Here, $\nu_{\text{ref}} := \dot{x}_{\text{ref}}(t)$ defines the reference derivative and ν_{fb} is a *feedback element* we will define below. Furthermore, intended to remove the drift f, *adaptive element* $\nu_{ad} := \hat{\mathfrak{f}}_{\mathfrak{n}}(x)$ is the output of the kinky inference rule as per Def. 4.2.4 after having observed data set $\mathcal{D}_n = \left\{ \left(x_j, f(x_j), \varepsilon(x_j)\right) \,|\, j = 1, ..., N_n \right\}$ containing a (noisy) sample of the drift at N_n inputs $x_1, \ldots, x_{N_n}$.

Closing the control-loop changes the plant dynamics to $\dot{x} = F(x) + \nu_{\text{fb}} + \nu_{\text{ref}} = F(x) + \nu_{\text{fb}} + \dot{x}_{\text{ref}}$ where $F(x) = f(x) - \hat{\mathfrak{f}}_{\mathfrak{n}}(x)$. Consequently the *error dynamics* are

$$\dot{e} = F(x) + \nu_{\text{fb}} \tag{4.21}$$

where the *state error trajectory* $e(\cdot) = x(\cdot) - x_{\text{ref}}(\cdot)$ represents the deviation of the state trajectory from the nominal trajectory given by the reference.

Since the drift f is uncertain, $F_n(x)$ will generally not be zero. Assuming f satisfies the prior assumptions inserted into the kinky inference learner ($f \in \mathcal{K}_{prior}$), Thm. 4.2.7 guarantees that the prediction bounds $\hat{\mathfrak{v}}$ provided by our KI rule (cf. Def. 4.2.4) hold and F is bounded. In particular, $F(x) \in E_n := [-\hat{\mathfrak{v}}_{n,1}(x),\, \hat{\mathfrak{v}}_{n,1}(x)] \times [-\hat{\mathfrak{v}}_{n,d}(x), ...,\, \hat{\mathfrak{v}}_{n,d}(x)]$ where $\hat{\mathfrak{v}}_{\mathfrak{n}}$ is defined as per Def. 4.2.4.

One way of driving the error to zero and to supress the influence of the remaining uncertainty is to set the feedback term to $\nu_{\text{fb}} = -Ke$ with $K := \text{diag}(w_1, ..., w_d)$ for some gain weights $w_1, ..., w_d \geq 0$. This yields error dynamics given by the differential equation:

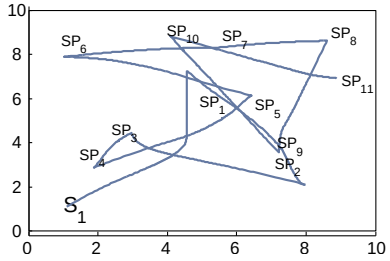
$$\dot{e}(t) = F(x(t)) - Ke(t). \tag{4.22}$$

We will now analyse the impact of both K and the maximal prediction uncertainty on stability.

Remember, we can restate the differential equation in integral form as follows (cf. Lem. 2.4.3): $e(t) = e(0) + \int_0^t -Ke(\tau)\, \mathrm{d}\tau + \int_0^t F(x(\tau))\, \mathrm{d}\tau$. Since $F = f - \hat{\mathfrak{f}}_{\mathfrak{n}}$, by Thm. 4.2.7, $\|F(x)\|_\infty \leq \bar{\mathfrak{N}} := \sup_{x \in \mathcal{X}} \| \hat{\mathfrak{v}}_{\mathfrak{n}}(x)\|_\infty$. Thus, $\|e(t)\|_\infty \leq \left\| e(0) + \int_0^t -Ke(\tau)\, \mathrm{d}\tau \right\|_\infty + \left\| \int_0^t F(x(\tau))\, \mathrm{d}\tau \right\|_\infty \leq \left\| e(0) + \int_0^t -Ke(\tau)\, \mathrm{d}\tau \right\|_\infty + t\, \bar{\mathfrak{N}}$. Observe, $e(0) + \int_0^t -Ke(\tau)\, \mathrm{d}\tau$ is the solution to the ODE $\dot{e} = -Ke$ with initial value $e(0)$. The solution is well-known to be: $e(0)\exp(-Kt)$ (cf. Sec. 2.4.6). Therefore, the following bound always holds:

$$\|e(t)\|_\infty \leq \|e(0)\exp(-Kt)\|_\infty + t\, \bar{\mathfrak{N}}. \tag{4.23}$$

Note, the term $\bar{\mathfrak{N}}$ depends on the learning experience. As we know from Thm. 4.2.7, the uncertainty of the learner (and hence $\bar{\mathfrak{N}}$ vanishes with increasing data density). Hence, the state error bound can be reduced if



(a) Path taken by the exploring agent.

Figure 4.6.: Path of an agent exploring the map. The agent obtains a new drift training example every 7 seconds. The start position is denoted by S_1. The subsequent waypoints (given by the setpoints), denoted by $SP1, ..., SP11$, were determined incrementally as the positions of maximum posterior uncertainty. That is, on the basis of data set $\mathcal{D}_n$ the agent chose a new setpoint ξ with components $\xi_j \in \arg\max_{x\in\mathcal{X}} \hat{\mathfrak{v}}_{n,j}(x)$. Note how the agent became increasingly successful in reaching his waypoints. With increasing learning experience, the trajectories become increasingly straight. This is a result of the adaptive element becoming more successful over time in cancelling the nonlinearity of the uncertain drift.

the environment of the agent is uniformly explored.

An agent exploration scenario

As an illustration, consider the a situation in which an agent is deployed in a confined state space, say an "arena" $\mathcal{X} = [0, 10] \times [0, 10]$. The agent's plant adheres to a dynamic model as above with uncertain drift field $f(x) = \begin{cases} [\sin(x_1); 2\cos(x_2)], & x \in \mathcal{X} \\ [0; 0], & \text{otherwise.} \end{cases}$. Initially the agent is uncertain about the drift. However, we assume it can acquire a sample point of the drift at its current location once every $\Delta = 7$ seconds at times $n\Delta$ ($n \in \mathbb{N}_0$). The observations are obtained by taking finite differences with sample rate 0.01. To account for numerical error we allow for a small observational noise level of $\varepsilon = 0.01$. At the initial learning step $n = 0$, we initialise the controller's adaptive element $\nu_{ad} = \hat{\mathfrak{f}}_{\mathfrak{n}}$ with a kinky inference predictor (cf. Def. 4.2.4). The KI learner is given the correct parameters $L = 2, p = 1$ and we set the upper and lower bound functions $\underline{B}$ and $\bar{B}$ to zero outside the arena $\mathcal{X}$. This folds in the knowledge about the state of affairs outside of the map. The agent aims to explore the state space in a manner suitable to decrease the maximal uncertainty $\bar{\mathfrak{N}}$ over time. Starting out with an estimated initial maximal uncertainty of 4.8, the agent is only allowed to stop the exploration process when $\bar{\mathfrak{N}} < 0.5$. For simplicity, we let the agent adopt a heuristic strategy and define a setpoint at the location $\xi_n \in \mathcal{X}$ where the current uncertainty is highest. This can be determined by setting $\xi_n \in \arg\max_{x\in\mathcal{X}} \hat{\mathfrak{v}}_{\mathfrak{n}}(x)$. This point, as well as the pertaining maximum uncertainty $\bar{\mathfrak{N}}$, required for computing the stopping criterion of the exploration process, is found by optimising each component uncertainty function $\hat{\mathfrak{v}}_{\mathfrak{n}}$. As we have noted before, Lem. 2.5.6 allows us to obtain a Hölder constant and

exponent for this function. This renders it amenable to the bounded optimisation approach discussed in Sec. 2.5. In our specific case, this is even easier, since the $\hat{\mathfrak{v}}_{\mathfrak{n}}$ is even Lipschitz. Therefore, it can be optimised (and the optimum be bounded) using Shupert's algorithm [230]. Starting at position $S_1 = [1; 1]$, the agent computes new setpoints with this optimisation procedure every $\Delta = 7$ seconds. The agent then observes a new drift sample, updates the data set, computes a new setpoint and starts over. A simulation of the path taken by the agent is depicted in Fig. 4.6. Notice how the controller becomes increasingly successful in reaching the waypoints and in cancelling out the nonlinearity of the dynamics. An excerpt of the inferred beliefs over the drift at three different stages of the online learning process is depicted in Fig. 4.7. It shows how a new training example is inserted at the point of maximum uncertainty (Fig. 4.7(a) vs Fig. 4.7(b)), decreasing the overall maximum uncertainty given by the floor and ceiling functions. The last plot (Fig. 4.7(c)) shows the inferred model over the drift at the end of the exploration process.

Remark 4.4.1. We can see that the maximum uncertainty is mainly due to the belief over the first component $\hat{\mathfrak{f}}_{n,1}$. This can be attributed to the fact that the maximum slope of the ground truth of this component function is 1. Therefore, the assumed Lipschitz constant $L = 2$ is a conservative choice for this function, which means the bounds are not as tight as possible. By contrast, for the second component function, this constant is tight. We can see this where the slope is steepest, few training examples are needed to shrink the uncertainty down to negligible amounts (that is, down to the level of observational noise). As a way out, it would have been possible to choose a separate constant for both component functions. For the first component function, $L_1 = 1$ would have been the best choice.

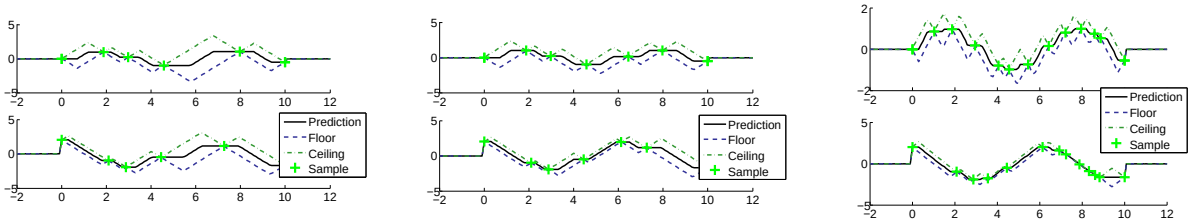
Stability and safety bounds

The attractive property of the bound of Eq. 4.23 is that it holds for all times (not just in the limit) and that K would not even have to be stable (i.e. Hurwitz). The clear disadvantage is that the bound grows with time and hence, does not provide a guarantee of stability of the state trajectory. To overcome this limitation, we will now provide a simple stability guarantee for the closed-loop trajectory of the kinky inference controlled plant. Since F_n is bounded (due to the guaranteed conservatism of the KI rule), this is a situation where our learning guarantees can be combined with standard results from robust control [140]. To this end, we remind the reader of the following fact:

Lemma 4.4.2 (from [140])**.** *Consider the system $\dot{x} = Ax + v$ where matrix A is stable and has the Jordan canonical form $J = V^{-1}AV$. Suppose that $v(t) \leq \bar{v}, \forall t \in [0, T]$. Finally for a matrix M and vector x let $|M|$ and $|x|$ denote their respective component-wise moduli. Then, we have:*

(I) $|x(t)| \leq |V| \left|Re(J)^{-1}\right| \left|V^{-1}\right| \bar{v}$, provided $\left|V^{-1}x(0)\right| \leq \left|Re(J)^{-1}\right| \left|V^{-1}\right| \bar{v}$.

(II) Furthermore, $\forall \epsilon > 0 \exists \bar{t} \forall t \geq \bar{t} : |x(t)| \leq |V| \left|Re(J)^{-1}\right| \left|V^{-1}\right| \bar{v} + |V| \epsilon$ (regardless of the initial condition).



(a) KI predictions after 6 learning steps. (b) KI predictions after one additional learning step. (c) Final prediction.

Figure 4.7.: Excerpt of the evolution of the posterior beliefs during the exploration phase. The agent starts out with a kinky inference learner that has initial knowledge of the fact that velocity outside the arena $\mathcal{X}$ is zero. The agent utilises the uncertainty bounds (floor and ceiling function) to guide exploration. That is, it puts a new setpoint at the position of the maximum uncertainty. The agent stops exploring after the maximal uncertainty threshold $\bar{\mathfrak{v}}$ is below 0.5. The top row shows the posteriors over the first output component of the learned drift vector field. The bottom row depicts the posterior beliefs over the second output dimension of the drift vector field. Plots Fig. 4.7(a) and Fig. 4.7(b) show how the exploration heuristic manages (after some learning) to explore the parts of state space with high uncertainty.

Harnessed with our Thm. 4.2.7 about the KI learner, we can immediately apply the lemma to yield the following simple stability guarantee:

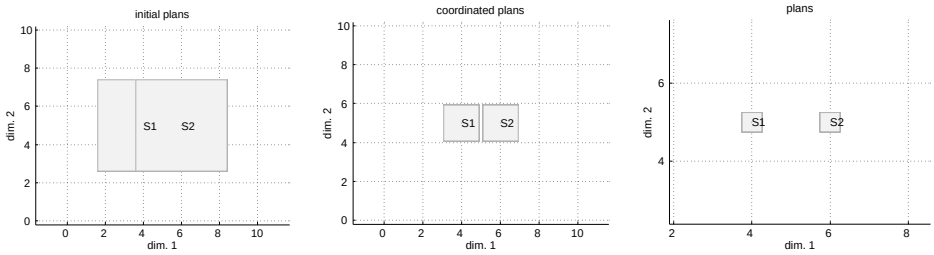
Theorem 4.4.3. *Assume we are given a sample* $\mathcal{D} = \left\{ \left(x_j, f(x_j), \varepsilon(x_j)\right) \,|\, j = 1, ..., N \right\}$ *of uncertain drift* $f \in \mathcal{K}_{prior}$ *where* $\mathcal{K}_{prior}$ *is defined as in Sec. 4.2. For each output dimension* i *we train a kinky inference learner on the data yielding predictors* $\hat{\mathfrak{f}}_{n,i}$ *with error function* $\hat{\mathfrak{v}}_{n,i}$ *(as per Def. 4.2.4) for components* $i = 1, ..., d$. *Define* $\bar{\mathfrak{v}}_i := \sup_{x \in \mathcal{X} \subset \mathbb{R}^d} \hat{\mathfrak{v}}_i(x)$ *to be the maximum prediction error in any dimension on the state space. Desiring to stabilise setpoint* ξ, *we apply the control* $u := g^{-1}(x)(-Ke - \hat{\mathfrak{f}} + \dot{x}_{ref})$ *where as before,* $K = diag(w_1, ..., w_d) \geq 0$. *This causes the plant to follow trajectory* $x(t) = x_{ref}(t) + e(t)$ *where* $x_{ref} = \xi$. *We have:*

1. *For dimensions* $(i = 1, ..., d)$, *define the interval* $S_i := [-\frac{\bar{\mathfrak{v}}_i}{w_i}, \frac{\bar{\mathfrak{v}}_i}{w_i}]$. $S = S_1 \times \ldots \times S_d$ *is an invariant set for the error dynamics. That is,* $e(t) \in S$ *implies* $e(t') \in S, \forall t' \geq t$.

2. *For any initial error* $e(0)$, $e(t)$ *converges to the largest invariant set in* S.

Proof. One approach of showing the result would be to construct a piece-wise Lyapunov type function that is zero on S and assumes values proportional to the distance of the error to the set S otherwise. We can then apply one of the more modern versions of La Salle's invariance theorem [34, 115] to conclude the statement. However, the statement also follows directly from Lem. 4.4.2. To see this, we choose $A = -K$, $v(t) = F_n(x(t))$ where A and v assume the roles as specified in the lemma.

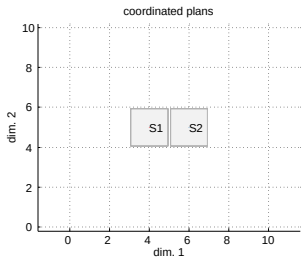
Firstly, we show that S is an invariant set. This is equivalent to saying that the component error (satisfying $\dot{e}_i(t) = Ae_i(t) + v_i(t)$) remains confined to S_i, provided $e_i(0) \in S_i$. So, assume $e_i(0) \in S_i$. By definition, $A = -K$ is stable. Since in our simple case, K is diagonal, $J = -K = A$ and $V = I_d$ is the identity



(a) Uncoordinated plans (prior learning).

(b) Coordinated plans (prior learning).

(c) Uncoordinated plans (after learning).



(d) Coord. plans (after learning).

Figure 4.8.: The grey boxes represent the agents' safety regions (invariant sets plus a margin taking into account the agent radius of 0.1) for various choices of maximal uncertainties $\bar{\mathfrak{v}}_i$ and feedback gains w_i for dimensions $i \in \{1, 2\}$. Fig. 4.8(a): $w_i = 2, \bar{\mathfrak{v}}_i = 4.8$. Fig. 4.8(b): $w_i = 5.3, \bar{\mathfrak{v}}_i = 4.8$. Fig. 4.8(c): $w_i = 2, \bar{\mathfrak{v}}_i = 0.5$. Fig. 4.8(c): $w_i = 0.55, \bar{\mathfrak{v}}_i = 0.5$.

matrix. Hence, $|V| = \left|V^{-1}\right| = I_d$. The assumption $e_i(0) \in S_i$ implies $\left|V^{-1}e_i(0)\right| = |e_i(0)| \leq \frac{\bar{\mathfrak{v}}_i}{w_i} = \left(\left|\mathrm{Re}(J)^{-1}\right| \left|V^{-1}\right| \bar{\mathfrak{v}}\right)_i$ for $i = 1, ..., d$. Hence, the condition of Lem. 4.4.2 (I) is satisfied.

Because of the diagonal form of K, we have the simple situation of no component interactions. Hence, for each component $i \in \{1, ..., d\}$, we have $|e_i(t)| \leq |V_{ii}| \left|A_{ii}^{-1}\right| \left|V_{ii}^{-1}\right| \bar{\mathfrak{v}}_i = |V_{ii}| \left|w_i^{-1}\right| \left|V_{ii}^{-1}\right| \bar{\mathfrak{v}}_i = \frac{1}{w_i}\bar{\mathfrak{v}}_i$ where we also have utilised $|F_i(x)| \leq \bar{\mathfrak{v}}_i$ (by Thm. 4.2.7). This shows that S is invariant.

The second statement about convergence to S follows directly from Lemma 4.4.2 (II). □

The theorem tells us something interesting about the connection between learning success (measured by the $\bar{\mathfrak{v}}_i$) and the magnitude of the feedback gain. That is, in order to stabilise the system up to a desired error bound, we can either increase the gain or reduce the uncertainty by learning. To illustrate this point, assume we have two agents 1 and 2 tasked to stabilise their plants at setpoints $S1 = [4; 5]$ and $S2 = [6; 5]$, respectively. Having a diameter of 0.2, and being immersed in an uncertain drift field as considered in the exploration scenario above, the agents desire to find a "plan" of how to set their feedback gains w_i such that it is guaranteed that they stabilise their setpoints (up to a certain error) while being guaranteed to avoid a collision with the other agent. Thm. 4.4.3 helps us to achieve this. That is, the agents can "coordinate" their plans (gain weights) such that their invariant sets are sufficiently small not to overlap. Consider Fig. 4.8. Fig. 4.8(a) depicts a situation the agents would encounter if they were initialised with the configuration the agent in our exploration scenario had prior to learning. In the depicted situation , their uncertainties were $\bar{\mathfrak{v}}_i = 4.8$

and the feedback gain was set to a diagonal matrix with diagonal entries $w_i = 2$. As the figure shows, the safety regions (invariant sets, plus some margin representing the agents' diameter) overlapped. Therefore, a collision could not be ruled out. As one way out, we can employ Thm. 4.4.3 to adjust the gain such that the regions will be disjoint. The result of doing so is depicted in Fig. 4.8(b).

In order to improve control success without having to resort to higher feedback gain, the agent can alternatively attempt to bring down the uncertainty by learning. That is, we can incorporate the learning experience gained during the agent exploration phase (see above) to bring down the prediction uncertainties $\bar{\mathfrak{v}}_i$ in each dimension i, while leaving the gains unaltered (Fig. 4.8(c)). As we can see from the plots, the reduced uncertainties ensured that the safety regions were disjoint, without further need to increase the gain. If all we were interested in was collision avoidance (or if we wanted to save control energy), we now could decrease the gains w_i as much as possible without making the safety regions overlap again (Fig. 4.8(d)). Once, again Thm. 4.4.3 tells us how to set the gains accordingly.

Note that so far, we have assumed that the agents are homogenous and have not taken (differing) notions of cost into account. For instance, an agent who experiences a lower loss when expending control energy might be assigned a higher gain than one with a lower notion of control cost. Automating such coordination methods in continuous time is addressed in Ch. 5. Our bounds for the kinky inference controllers could be combined with the coordination methods developed therein. This would yield a method for automated coordination with guaranteed continuous time-collision avoidance of learning agents.

4.4.2. Kinky inference and model reference adaptive control for online learning and tracking control in the presence of wing rock dynamics

As pointed out in [67], modern fighter aircraft designs are susceptible to lightly damped oscillations in roll known as "wing rock". Commonly occurring during landing [214], removing wing rock from the dynamics is crucial for precision control of such aircraft. Precision tracking control in the presence of wing rock is a nonlinear problem of practical importance and has served as a test bed for a number nonlinear adaptive control methods [66, 67, 172].

For comparison, we replicate the experiments of the recent work of Chowdhary et. al. [63, 66].[4] Here the authors have compared their Gaussian process based approach, called *GP-MRAC*, to the more established adaptive model-reference control approach based on RBF networks [129, 218], referred to as *RBFN-MRAC*. Replacing the Gaussian process learner by our kinky inference learner, we readily obtain an analogous approach which we will refer to as *kinky inference model reference adaptive control* (*KI-MRAC*). As an additional baseline, we also examine the performance of a simple P-controller.

[4] We are grateful to the authors for kindly providing the code.

While with the exact same parameters settings of the experiments in [66], performance of our KI-MRAC method comes second to GP-MRAC, we also evaluate the performance of all controllers over a range of 555 random parameter settings and initial conditions. As we will see, across this range of problem instances and parameter settings, KI-MRAC markedly outperforms all other methods.

Model reference adaptive control

Before proceeding with the wing rock application we will commence with (i) outlining model reference adaptive control (MRAC) [14] as considered in [66] and (ii) describe the deployment of kinky inference to this framework. In analogy to our exposition for velocity-controlled first order systems offered above, we will now rehearse the description of MRAC for second-order systems following [66].

Assume $m \in \mathbb{N}$ to be the dimensionality of a configuration of the system in question and define $d = 2m$ to be the dimensionality of the pertaining state space $\mathcal{X}$.

Let $x = [x_1; x_2] \in \mathcal{X}$ denote the state of the plant to be controlled. Given the control-affine system

$$\dot{x}_1 = x_2 \tag{4.24}$$

$$\dot{x}_2 = a(x) + b(x)\, u(x) \tag{4.25}$$

it is desired to find a control law $u(x)$ such that the closed-loop dynamics exhibit a desired reference behaviour:

$$\dot{\xi}_1 = \xi_2 \tag{4.26}$$

$$\dot{\xi}_2 = f_r(\xi, r) \tag{4.27}$$

where r is a reference command, f_r some desired response and $t \mapsto \xi(t)$ is the reference trajectory.

If a priori a and b are believed to coincide with $\hat{a}_0, \hat{b}_0$ respectively, the inversion control $u = \hat{b}_0^{-1}(-\hat{a}_0 + u')$ is applied. This reduces the closed-loop dynamics to $\dot{x}_1 = x_2, \dot{x}_2 = u' + \tilde{a}(x, u)$ where $\tilde{a}(x, u)$ captures the modelling error of the dynamics:

$$\tilde{a}(x, u) = a(x) - \hat{a}_0(x) + \big(b(x) - \hat{b}_0(x)\big)u. \tag{4.28}$$

Let $I_d \in \mathbb{R}^{d \times d}$ denote the identity matrix. If b was perfectly known, then $b - \hat{b}_0^{-1} = 0$ and the model error can be written as $\tilde{a}(x) = a(x) - \hat{a}_0(x)$. In particular, $\tilde{a}$ has lost its dependence on the control input.

In this situation [63, 66] propose to set the pseudo control as follows: $u'(x) := \nu_r + \nu_{pd} - \nu_{ad}$ where

$\nu_r = f_r(\xi, r)$ is a feed-forward reference term, ν_{ad} is a yet to be defined output of a learning module *adaptive element* and $\nu_{pd} = [K_1 K_2]e$ is a feedback error term designed to decrease the *tracking error* $e(t) = \xi(t) - x(t)$ by defining $K_1, K_2 \in \mathbb{R}^{m \times m}$ as in described in what is to follow.

Inserting these components, we see that the resulting *error dynamics* are:

$$\dot{e} = \dot{\xi} - [x_2; \nu_r + \nu_{pd} + \tilde{a}(x)] = Me + B\big(\nu_{ad}(x) - \tilde{a}(x)\big) \tag{4.29}$$

where $M = \begin{pmatrix} O_m & I_m \\ -K_1 & -K_2 \end{pmatrix}$ and $B = \begin{pmatrix} O_m \\ I_m \end{pmatrix}$. If the feedback gain matrices K_1, K_2 parametrising ν_{pd} are chosen such that M is stable then the error dynamics converge to zero as desired provided the learning error E_λ vanishes: $E_\lambda(x(t)) = \|\nu_{ad}(x(t)) - a(x(t))\| \overset{t\to\infty}{\longrightarrow} 0$.

It is assumed that the adaptive element is the output of a learning algorithm that is tasked to learn $\tilde{a}$ online. This is done by continuously feeding it training examples of the form $\big(x(t_i), \tilde{a}(x(t_i)) + \varepsilon_i\big)$ where ε_i is observational noise.

Intuitively, assuming the learning algorithm is suitable to learn target $\tilde{a}$ (i.e. $\tilde{a}$ is close to some element in the hypothesis space [171] of the learner) and that the controller manages to keep the visited state space bounded, the learning error (as a function of time t) should vanish.

Substituting different learning algorithms yields different adaptive controllers. *RBFN-MRAC* [129] utilises radial basis function neural networks for this purpose whereas *GP-MRAC* employs Gaussian process learning [201] to learn $\tilde{a}$ [63, 66].

Note, this setting of GP-MRAC is a special case of the SP-SIIC approach we introduced in Ch. 3. Here $\hat{a}_0$ is the prior mean function of the s.p. we place over the drift. Our exposition in Sec. 3 is more general in that it allows for uncertain b that is learned from data, considers under-actuated systems and makes no distributional restrictions. As an additional feature, GP-MRAC assumes that the data sets can be updated continuously. While of course this is not tractable the authors maintain a data corpus of a fixed size by a range of methods including simple FIFO queues as well as sparsification methods [72, 131]. In contrast, we utilise learning criteria to only include informative data points. In addition, we employ hyper-parameter optimisation which we show to be highly beneficial. While hyper-parameter training cannot be done at high frequency, it would be possible to run it in the background on a separate CPU (in parallel to the CPU on which the controller runs).

In what is to follow, we utilise our kinky inference learning method as the adaptive element. Following the nomenclature of the previous methods we name the resulting adaptive controller *KI-MRAC* (where *KI* stands for *kinky inference*).

The wing rock control problem

The wing rock dynamics control problem considers an aircraft in flight. Denoting x_1 to be the roll attitude (angle of the aircraft wings) and x_2 the roll rate (measured in angles per second), the controller can set the aileron control input u to influence the state $x := [x_1; x_2]$.

Based on [172], Chowdhary et. al. [63, 66] consider the following model of the wing rock dynamics:

$$\dot{x}_1 = x_2 \tag{4.30}$$

$$\dot{x}_2 = a(x) + b\,u \tag{4.31}$$

where $b = 3$ is a known constant and $a(x) = W_0^* + W_1^* x_1 + W_2^* x_2 + W_3^* |x_1|\, x_2 + W_4^* |x_2|\, x_2 + W_5^* x_2^3$ is an priori unknown nonlinear drift.

Note, the drift is non-smooth but, employing Lem. 2.5.6, it would be easy to derive a Lipschitz constant on any bounded subset of state space if the parameters $W := (W_0^*, \ldots, W_5^*)$ were known.

To control the system we employ KI-MRAC as the adaptive element ν_{ad}. In the absence of the knowledge of a Lipschitz constant, we start with a guess of $L = 1$ (which will turn out to be too low) and update it following the procedure described in Sec. 4.3.2.

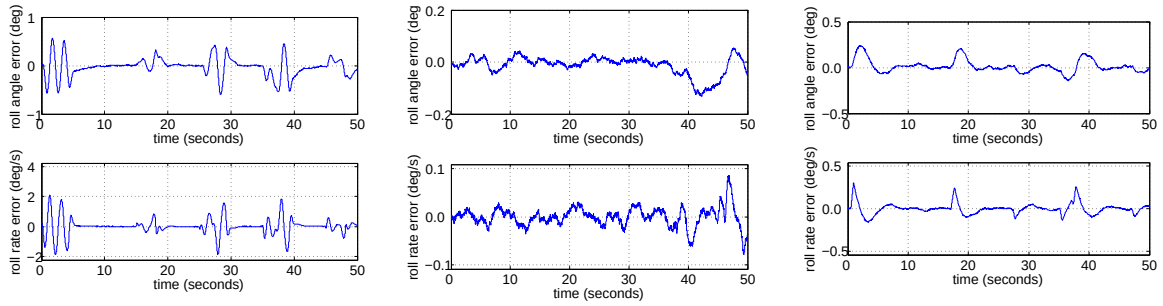
In a first instance, we replicated the experiments conducted in [66, 67] with the exact same parameter settings. That is, we chose $W_0^* = 0.8, W_1^* = 0.2314, W_2^* = 0.6918, W_3^* = -0.6245, W_4^* = 0.0095, W_5^* = 0.0214$.

The simulation initialised with start state $x = (3, 6)^\top$ and simulated forward with a first-order Euler approximation with time increment $\Delta = 0.005[s]$ over a time interval $\mathcal{I}_t = [t_0, t_f]$ with $t_0 = 0[s]$ and $t_f = 50[s]$. Training examples and control signal were continuously updated every $\Delta_u = \Delta_o = \Delta[s]$. The RBF and GP learning algorithms were initialised with fixed length scales of 0.3 units. The GP was given a training example budget of a maximum of 100 training data points to condition the posterior model on. Our kinky inference learner was initialised with $L = p = 1$ and updated online following the lazy update method described in Sec. 4.3.2.

The test runs also exemplify the working of the lazy update rule. The initial guess $L = 1$ was too low. However, our lazy update rule successfully picked up on this and had ended up increasing constant to $L = 2.6014$ by the end of the online learning process.

The results are plotted in Fig. 4.9. We can see that in terms of tracking error of the reference our KI-MRAC outperformed RBF-MRAC and was a close runner-up to GP-MRAC which had the lowest tracking errors.

To obtain an impression of the learning performance of the three learning algorithms we also recorded the



(a) Tracking error (RBF-MRAC). (b) Tracking error (GP-MRAC). (c) Tracking error (KI-MRAC).

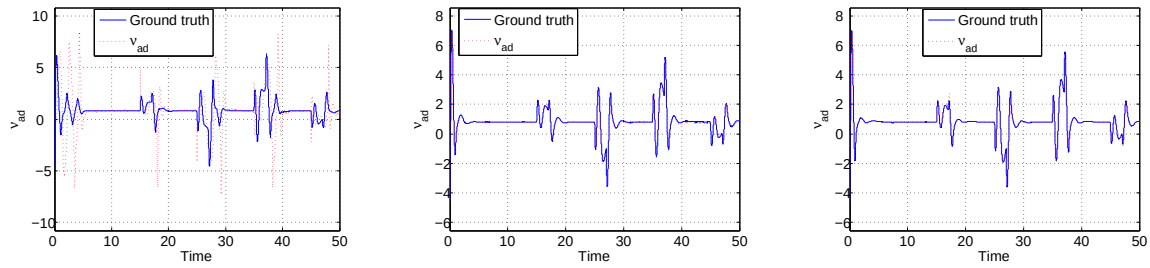
Figure 4.9.: Tracking error comparison of first example.

prediction error histories for this example problem. The results are depicted in Fig. 4.10. We can see that our kinky inference method and the GP method both succeeded in predicting the drift quite accurately while the RBFN method was somewhat lagging behind. This is consistent with the observations made in [63, 66]. The authors explain the relatively poor performance of the radial basis function network method by the fact that the reference trajectory on occasion led outside the region of state space where the centres of the basis function were placed in advance. By contrast, due to the non-parametric nature of the GP, GP-MRAC does not suffer from such a priori limitations. In fact, it can be seen as an RBF method that flexibly places basis functions around all observed data points [201]. We would add that, as a non-parametric method, KI-MRAC shares this kind of flexibility, which might explain the fairly similar performance.

However, being an online method, the authors of GP-MRAC explicitly avoided hyperparameter training via optimising the marginal log-likelihood. The latter is commonly done in GP learning [201] to avoid the impact of an unadjusted prior (also see Ch. 3 for a discussion in the context of system identification) but is often a computational bottle neck. Therefore, avoiding such hyperparameter optimisation greatly enhances learning and prediction speed in an online setting. However, we would expect the performance of the prediction to be dependent upon the hyperparameter settings. As we have noted above, the Lipschitz constant depends on the part of state space visited at runtime. Similarly, we might expect length scale changes depending on the part of state space the trajectory is in. Unfortunately, [63, 66, 67] provide no discussion of the length scale parameter setting and also called the choice of the maximum training corpus "arbitrary".

Since the point of learning-based and adaptive control is to be able to adapt to various settings, we test the controllers across a range of randomised problem settings, initial conditions and parameter settings.

We created 555 randomised test runs of the wingrock tracking problems and tested each algorithm on each one of them. The initial state $x(t_0)$ was drawn uniformly at random from $[0, 7] \times [0, 7]$, the initial kernel length scales were drawn uniformly at random from $[0.05, 2]$, and used both for RBF-MRAC and GP-MRAC. The Hölder constant L for the KI-MRAC was initialised at random from the same interval but allowed to be adapted as part of the online learning process. The parameter weights W of the system dynamics



(a) Prediction v.s. ground truth (RBF-MRAC).

(b) Prediction v.s. ground truth (GP-MRAC).

(c) Prediction v.s. ground truth (KI-MRAC).

Figure 4.10.: Prediction vs ground truth comparisons for the first example. Both nonparametric methods accurately predict the true drift and clearly outperform the RBFN learner.

specified above were multiplied by a constant drawn uniformly at random from the interval $[0, 2]$. To allow for better predictive performance of GP-MRAC we doubled the maximal budget to 200 training examples. The feedback gains were chosen to be $K_1 = K_2 = 1$.

In addition to the three adaptive controllers we also tested the performance of a simple P controller with just these feedback gains (i.e. we executed x-MRAC with adaptive element $\nu_{ad} = 0$). This served as a baseline comparison to highlight the benefits of the adaptive element over simple feedback control.

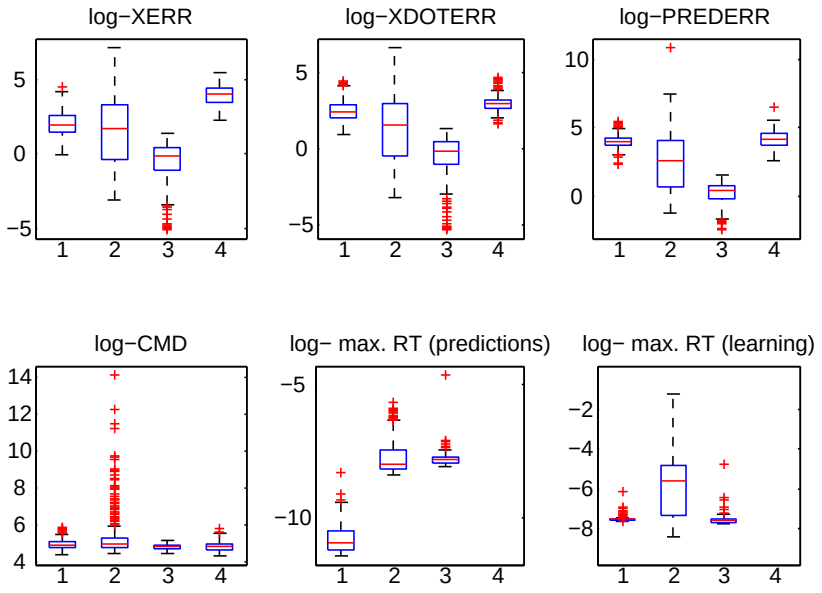
The performance of all controllers across these randomised trials is depicted in Fig. 4.11. Each data point of each boxplot represent a performance measurement for one particular trial.

For each method, the figures show the boxplots of the following recorded quantities:

- *log-XERR*: cummulative angular position error (log-deg), i.e. $\log(\int_{t_0}^{t_f} \|\xi_1(t) - x_1(t)\| \, dt)$.
- *log-XDOTERR*: cummulative roll rate error (log-deg/sec.), i.e. $\log(\int_{t_0}^{t_f} \|\xi_2(t) - x_2(t)\| \, dt)$.
- *log-PREDERR*: log-prediction error, i.e. $\log(\int_{t_0}^{t_f} \|\nu_{ad}(x(t)) - \tilde{a}(x(t))\| \, dt)$.
- *log-CMD*: cummulative control magnitude (log-scale), i.e. $\log(\int_{t_0}^{t_f} \|u(t)\| \, dt)$.
- *log-max. RT (predictions)*: the log of the maximal runtime (within time span $[t_0, t_f]$) each method took to generate a prediction ν_{ad} within the time span.
- *log-max. RT (learning)*: the log of the maximal runtime (within time span $[t_0, t_f]$) it took each method to incorporate a new training example of the drift $\tilde{a}$.

As can be seen from Fig. 4.11, all three adaptive methods outperformed the simple P controller in terms of tracking error.

In terms of prediction runtime, the RBF-MRAC outperformed both GP-MRAC and KI-MRAC. This is hardly surprising. After all, RBF-MRAC is a parametric method with constant prediction time. By contrast,



(a) Results over 555 randomised examples.

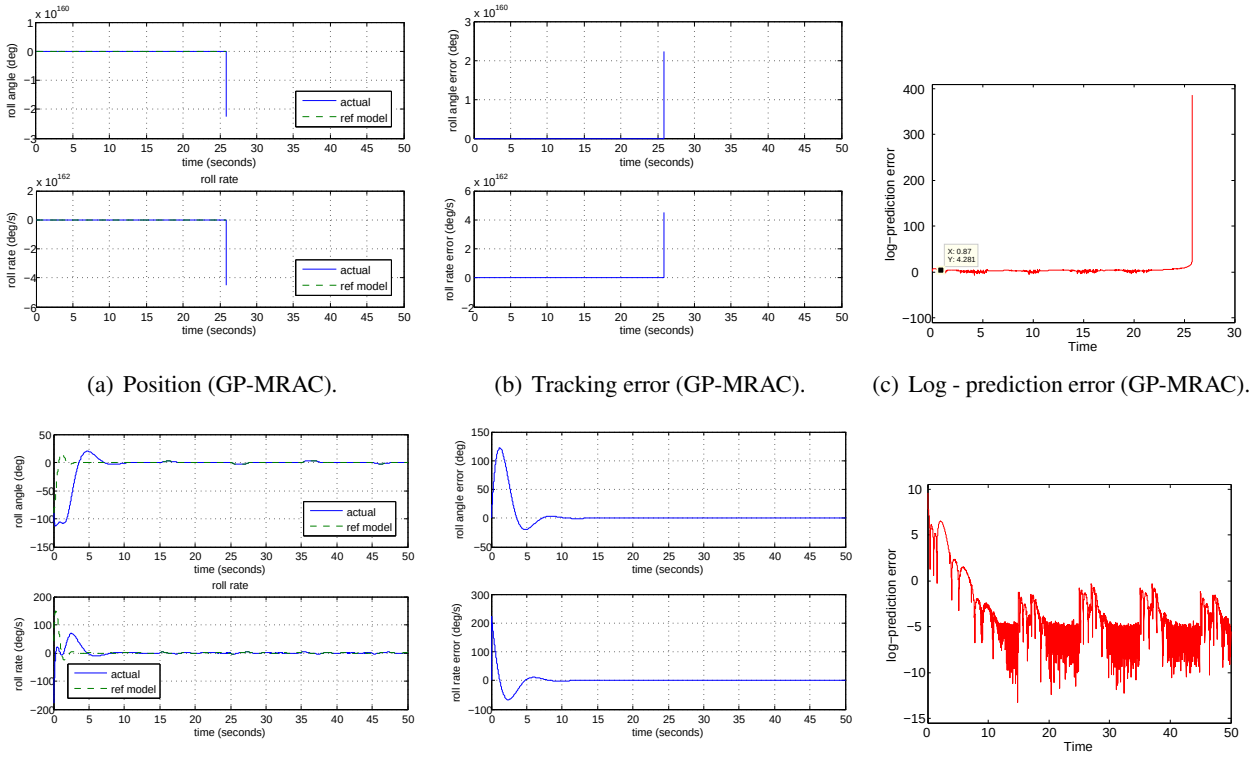
Figure 4.11.: Performance of the different online controllers over a range of 555 trials with randomised parameter settings and initial conditions. 1: RBF-MRAC, 2: GP-MRAC, 3: KI-MRAC, 4: P-Controller. KI-MRAC outperforms all other methods with respect to all performance measures, except for prediction runtime (where the parametric learner RBF-MRAC performs best).

both non-parametric methods will have prediction times growing with the number of training examples. That is, it would be the case if GP-MRAC were given an infinite training size budget. Indeed one might argue whether GP-MRAC, if operated with a finite budget, actually is a parametric approximation where the parameter consists of the hyperparameters along with the fixed-size training data matrix X (cf. Sec. 2.6). When comparing the (maximum) prediction and learning runtimes one should also bear in mind that GP-MRAC predicted with up to 200 examples in the training data set. By contrast, KI-MRAC undiscerningly had incorporated all 10001 training points by the end of each trial.

Across the remaining metrics, KI-MRAC markedly outperformed all other methods.

Note, we have also attempted to test all methods across a greater range of problem settings, including larger initial states, more varied hyper-parameter settings, lower feedback gains and more varied choices of dynamics coefficients W. However, this resulted in GP-MRAC to often run into conditioning problems. This is a common issue in GP learning due to the necessity of matrix inversion or Cholesky decompositions of the covariance matrix. Similar behaviour ensued when setting the training size budget to large values. All these changes often resulted in long learning runtimes, spiky control outputs and thus, poor overall performance. Similarly, code execution of our RBF-MRAC implementation was frequently interrupted with error messages when the state was initialised to positions outside of the rectangle $[0, 7] \times [0, 7]$.

We have not investigated the root cause of these issues in greater detail yet. However, it might be worth



(a) Position (GP-MRAC). (b) Tracking error (GP-MRAC). (c) Log - prediction error (GP-MRAC).

(d) Position (KI-MRAC). (e) Tracking error (KI-MRAC). (f) Log - prediction error (KI-MRAC).

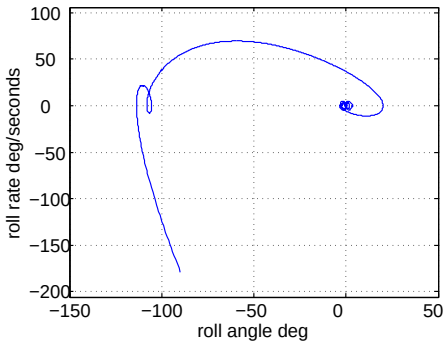
Figure 4.12.: Example where GP-MRAC fails. By contrast, KI-MRAC manages to adapt and direct the system back to the desired trajectory.

exploring whether the great robustness of kinky inference might be an additional selling point that sets it apart from other recent adaptive control methods. Such robustness is of course important in control settings such as flight control where failure or erratic behaviour of the adaptive element may result in critical incidents.

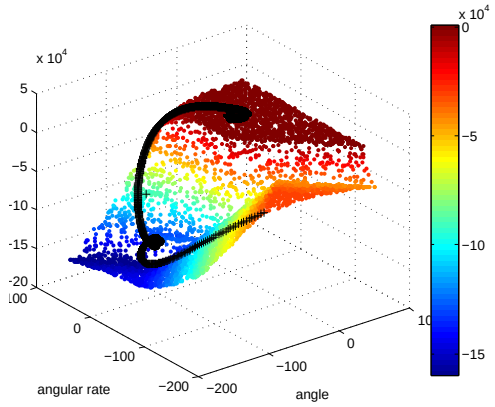
An example where GP-MRAC failed to track the reference occurred when repeating our first experiment with the following modifications: The initial state was chosen to be $x(t_0) = (-90, 40)^\top$ corresponding to a rapidly rotating aircraft. Furthermore, the wing rock coefficients W were multiplied by a factor of 5, amplifying the non-linearities of the drift field.

When initialised with a length scale parameter of 0.3, the GP ran into conditioning problems and caused the output of the adaptive element in GP-MRAC to produce spikes of very large magnitude and thus, further destabilised the system. We tried the problem with various kernel length scale settings ranging from 0.3 to 20. Increasing the length scale parameter to length scale of at least 1 seemed to fix the conditioning problem. Nonetheless, GP-MRAC still did not manage to learn and stabilise the system in any of these settings. A record of GP-MRAC's performance in this example (for length scale of 1) is depicted in Fig. 4.12(a) - 4.12(c). As the plots show, GP-MRAC starts with relatively high tracking and prediction error from which it could not recover. At about 26 seconds into the simulation the state rapidly diverged.

For comparison, we also tried KI-MRAC on the same problem, starting with initial $L = 1$ as before.



(a) State path (KI-MRAC).



(b) Learned drift model (KI-MRAC).

Figure 4.13.: Depicted are the state path and the drift model learned online by KI-MRAC.

Starting out with a relatively large tracking and prediction error, KI-MRAC nonetheless managed to recover and successfully track the system (see Fig. 4.12(d) - 4.12(f)). The state path and learned drift model obtained by KI-MRAC are depicted in Fig. 4.13.

4.4.3. Boundedness in discrete-time systems

In the previous subsection, we gave an illustration KI-MRAC – a combination of a feedback-linearising controller with our KI learning method. The results are encouraging – our adaptive control law managed to learn a dynamic system online and to track a reference in the presence of wingrock dynamics where other state of the art methods failed.

However, KI-MRAC is an online learning approach and assumes that the predictions can be made on the basis of the entire history of past observations. This might work for short time horizons. Unfortunately, such assumptions are often unrealistic in practice. Despite being a fast nonparametric method, eventually the size of the training corpus will grow to an extent where the required prediction response time exceeds the desired control refresh frequency.

Furthermore, the controller in KI-MRAC did not take the uncertainty into account when making decisions on the control action.

In this section, we assume that predictions are made on the basis of a fixed offline training corpus. This means that the uncertainty will always be bounded from below. For this setting, we are interested in the following questions:

1. Given a linearising adaptive control law in GP-MRAC in offline learning mode, can we quantify a bound on the error dynamics ? That is, folding in the prediction bounds of our KI learner, how do we construct a robust tube around the nominal trajectory that is guaranteed to contain the real trajectory?

2. How can we utilise the tubes for collision avoidance in a multi-agent setting?

The second question will be addressed in Sec. 6.4 in the context of predictive (open-loop) control. For now, we will lay some of the ground work and address the first question. However, the derivation of bounds is also of relevance for collision avoidance since it can be harnessed to bound the trajectory within a safety region (tube). The latter can be converted into collision avoidance constraints or a criterion function (cf. Ch. 5, Ch. 6 as well as references in Sec. 2.2).

The exposition in this section will focus on KI-MRAC in discrete-time. Here, we assume the dynamics are given as first-order Euler approximations of the second-order control-affine dynamics of Sec. 4.4.2. This means that the error dynamics as per Eq. 4.29 translate to the recurrence relation:

$$e_{k+1} = Me_k + \Delta F_k \tag{4.32}$$

where $\Delta \in \mathbb{R}_+$ is a positive time increment, k is the time step index. Furthermore,

$$F_k := F(x_k) = f(x_k) - \hat{\mathfrak{f}}_{\mathfrak{n}}(x_k) = B\big(\nu_{ad}(x_k) - \tilde{a}(x_k)\big) \tag{4.33}$$

is an uncertain increment due to the model error of the learner (cf. Sec. 4.4.1 or Eq. 4.29), $B = \begin{pmatrix} O_m \\ \Delta I_m \end{pmatrix}$ and

$$M = \begin{pmatrix} I_m & \Delta I_m \\ -\Delta K_1 & I_m - \Delta K_2 \end{pmatrix} \tag{4.34}$$

is the (error state) transition matrix.

This is completely consistent with the uncertain recurrence considered in Eq. 2.26, with the exception, that the uncertainty of increment F_k is not probabilistic. Instead it is an uncertain quantity reflecting the prediction error of the trained KI learner. Assuming its assumptions are met, Thm. 4.2.7 tells us that F_k is interval-bounded. Employing the same type of argument as in Lem. 2.4.35 we see that the solution of the recurrence is given by:

$$e_k = Me_{k-1} + \Delta F_{k-1} = \ldots = M^k\, e_0 + \Delta \sum_{i=0}^{k-1} M^{k-1-i}\, F_i. \tag{4.35}$$

We desire to bound the norm of the error. To this end, we leverage that the norms are sub-additive and sub-multiplicative to deduce:

$$\|e_k\| \leq \left|\!\left|\!\left| M^k \right|\!\right|\!\right| \, \|e_0\| + \Delta \sum_{i=0}^{k-1} \left|\!\left|\!\left| M^{k-1-i} \right|\!\right|\!\right| \, \|F_i\| \leq \left|\!\left|\!\left| M^k \right|\!\right|\!\right| \, \|e_0\| + \Delta \bar{\mathfrak{N}}_k \sum_{i=0}^{k-1} \left|\!\left|\!\left| M^{k-1-i} \right|\!\right|\!\right| =: \varrho[k] \tag{4.36}$$

where $\bar{\mathfrak{N}}_k$ us chosen such that we can guarantee that $\bar{\mathfrak{N}}_k \geq \max_{i=1,\dots,k-1} \|F_i\|$. For instance, we could choose $\bar{\mathfrak{N}}_k := \sup_{s\in\mathcal{S}_k} \|F(s)\| \leq c \sup_{s\in\mathcal{S}_k} \|\hat{\mathfrak{v}}_{\mathfrak{n}}(s)\|_\infty$ where $\mathcal{S}_k = \bigcup_{t<k}\{x \in \mathcal{S} | \, \|x - \xi[t]\| \leq \varrho[t]\}$ is the union of the possible states around the nominal trajectory $\xi[\cdot]$ at previous time steps. As in Sec. 4.4.1, we assume there exists a maximum model error norm $\bar{\mathfrak{N}}$, i.e. $\forall i : \|F_i\| \leq \bar{\mathfrak{N}}$. For instance, if we construct the tube for collision avoidance within a confined part of state space (such as an arena), if we know the drift is periodic or bounded, we can set $\bar{\mathfrak{N}} = \sup_x \|F(x)\|_\infty$. Then we have:

$$\|e_k\| \leq \left|\!\left|\!\left| M^k \right|\!\right|\!\right| \, \|e_0\| + \Delta\bar{\mathfrak{N}} \sum_{i=0}^{k-1} \left|\!\left|\!\left| M^{k-1-i} \right|\!\right|\!\right|. \tag{4.37}$$

As shown in Sec. A.3.2, the right hand side is convergent provided the gains K_1, K_2 are chosen such that M is stable, i.e. $\rho(M) < 1$, and provided $\bar{\mathfrak{N}}_k$ is is bounded.

Whether or not M is stable, in low dimensions, the sums can be computed offline and in advance. This is of great benefit if the controller that is building on the error bounds is utilising optimisation-based control with a finite time-horizon (for a related setup, see cf. Sec. 6.4).

To get a (conservative) closed-form bound on the error norms, Sec. A.3.2 contains a derivation and discussion of the following result:

Theorem 4.4.4. *Let $(F_k)_{k\in\mathbb{N}_0}$ be a norm-bounded sequence of vectors with $\bar{\mathfrak{N}}_k := \max_{i\in\{0,\dots,k-1\}} \|F_i\| \leq \bar{\mathfrak{N}} \in \mathbb{R}$. For error sequence $(e_k)_{k\in\mathbb{N}_0}$ defined by the linear recurrence $e_{k+1} = Me_k + \Delta F_k \;\; (k \in \mathbb{N}_0)$, we can define the following bounds:*

1. $\|e_k\| \leq \left|\!\left|\!\left| M^k \right|\!\right|\!\right| \, \|x_0\| + \Delta\bar{\mathfrak{N}}_k \sum_{i=0}^{k-1} \left|\!\left|\!\left| M^i \right|\!\right|\!\right|$. *If $\rho(M) < 1$ and $\exists\bar{\mathfrak{N}} : \bar{\mathfrak{N}} \geq \bar{\mathfrak{N}}_k \geq 0, \forall k$, the right-hand side converges as $k \to \infty$.*

2. *Let $k_0 \in \mathbb{N}, k_0 > 1$ such that $\left|\!\left|\!\left| M^k \right|\!\right|\!\right| < 1, \forall k \geq k_0$ and let $\varphi := \left|\!\left|\!\left| M^{k_0} \right|\!\right|\!\right| < 1$ and let $\delta_k := \lfloor k/k_0 \rfloor$. If $r := \rho(M) < 1$, for $k > k_0$, we also have:*

$$\|e_k\| \leq c\,\varphi^{\delta_k} \, \|e_0\| + \Delta\bar{\mathfrak{N}}_k \Big(\sum_{j=0}^{k_0-1} \left|\!\left|\!\left| M^j \right|\!\right|\!\right| + c\,k_0 \frac{\varphi - \varphi^{\left\lfloor \frac{k}{k_0} \right\rfloor + 1}}{1-\varphi} \Big) \overset{k\to\infty}{\longrightarrow} C \leq \Delta\bar{\mathfrak{N}} \sum_{j=0}^{k_0-1} \left|\!\left|\!\left| M^j \right|\!\right|\!\right| + \frac{\Delta\bar{\mathfrak{N}}\, c\, k_0\, \varphi}{1-\varphi} \tag{4.38}$$

   for some constants $C, c \in \mathbb{R}$. Here, possible choices for $c \in \mathbb{R}$ are:

   (i) $c = \max\{\left|\!\left|\!\left| M^i \right|\!\right|\!\right| | i = 1, \dots, k_0 - 1\}$

   or (ii) $c = \frac{1}{(d-1)!}\Big(\frac{1-d}{\log r}\Big)^{d-1} |\!|\!| M |\!|\!|^{d-1} \; r^{\frac{1-d}{\log r} - d + 1}$. Since $|\!|\!| M |\!|\!| \neq 1$, one can also choose (iii) $c := |\!|\!| M |\!|\!|^{k_0}$.

3. *If $|\!|\!| M |\!|\!| \neq 1$, the following holds: $\|e_k\| \leq |\!|\!| M |\!|\!|^k \, \|e_0\| + \Delta\bar{\mathfrak{N}}_k \frac{1 - |\!|\!| M |\!|\!|^k}{1 - |\!|\!| M |\!|\!|}$.*

Proof. The theorem is proven in the appendix (cf. Thm. A.3.3). □

Note, the bound provided in 2. is of particular interest in high-dimensional systems and for large time steps where computation of the bound according to 1. may be computationally demanding or intractable. In Ch. A, we derive a fast, conservative algorithm that can calculate a sequence of improving upper bounds on the "critical matrix exponent" k_0. This is of interest in cases where its naive computation is expensive (e.g. in high-dimensional systems). Furthermore, the chapter contains an illustration of how the error norm bounds decrease with decreasing spectral radius $\rho(M)$.

Finally, Sec. 6.4 considers an application of a bound similar to 1) in Thm. 4.4.4 to the different setting where there is no feedback gain and where the reference control is determined via mixed-integer programming in the context of MPC with collision avoidance constraints. However, we point out that in such a setting, one could also use feedback control in combination with predictive control. For this situation, the bounds of Thm. 4.4.4 would be of immediate utility.

4.5. Outlook on additional learning-based control methods

In the following we will briefly discuss applications of KI to our system identification and inversion control approach (SIIC), followed by an application to inversion dynamics model learning and control (IMLC) [180]. While providing rigorous examinations and comparisons was beyond the time limit and scope of this work, we have included the latter application to highlight our method's applicability in control that is based on black-box learning [157].

4.5.1. Kinky inference for system identification and inversion control (KI-SIIC)

In this section, we will apply kinky inference to our SIIC framework introduced in Ch. 3. Let q be a generalised coordinate (a configuration) and $x = [q; \dot{q}]$ be the state. Given the second-order control-affine dynamics $\ddot{q} = a(x) + b(x)u$, we infer both a and b with a separate KI learner. The online learning process otherwise proceeds just as described in Ch. 3. Note, in the case of random field learning, we placed a log-normal prior over the components of b to learn it. The motivation was to encode the knowledge that b was positive and bounded away from zero. Fortunately, in our KI framework, we can encode knowledge of boundedness quite explicitly by setting the lower bound function $\underline{B}$ to zero. As before, we set the pseudo-control to linear feedback with gain weights $w_1 = w_2 = 2$. This weight is large enough to drive the linearised dynamics to the goal, but sufficiently weak to ensure that learning success was important to achieve reaching and stabilising a goal state.

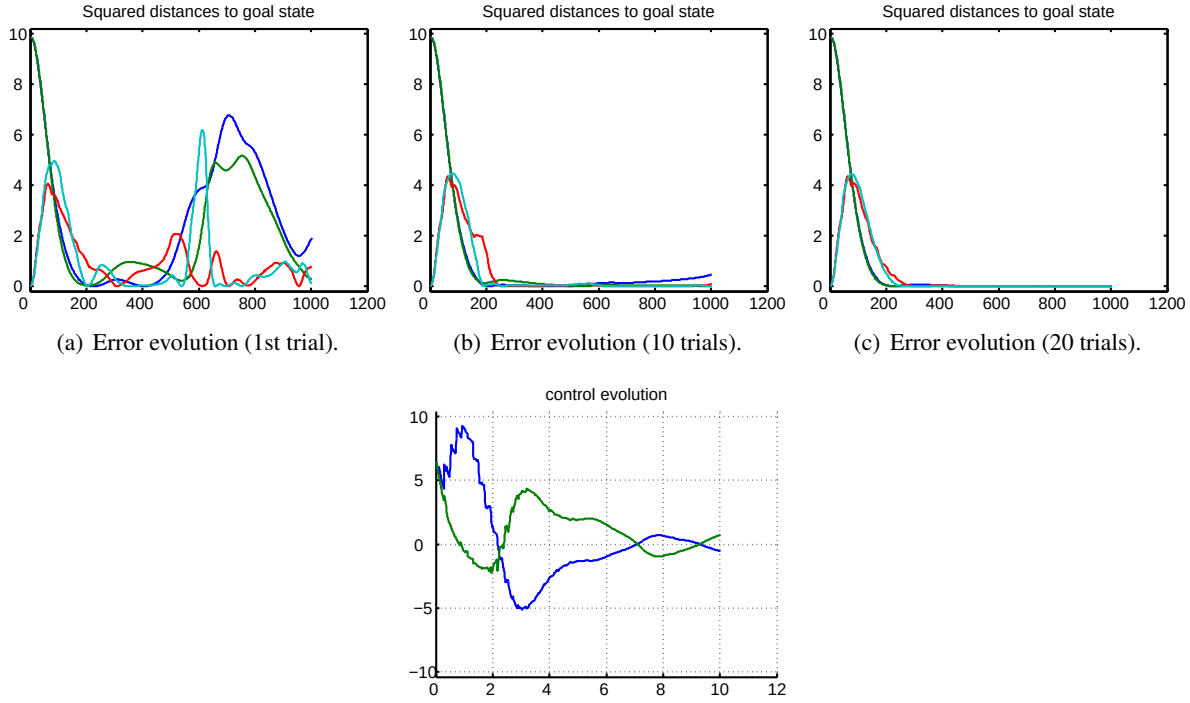
As an illustration that our SIIC framework also works well with the KI learner in place of the random fields, we have repeated a double pendulum simulation. The results of the learning process are depicted in Fig. 4.14. In this simulation, we have repeated the stabilisation task of moving the pendulum to the goal state

$\xi = [\pi; \pi; 0; 0]$. Firstly, we recorded the state and control trajectories for an episode of ten seconds of online learning. The resulting squared control error evolution (measuring the squared distances of the state variables to the goal state ξ at each time step) is depicted in Fig. 4.14(a). During online learning, we allowed the controller to update the training examples to be updated every $\Delta_\lambda = 0.3$ seconds and set the observational uncertainty level to $\varepsilon \equiv 0.01$. In the initial round, the controller was unsuccessful in stabilising the double pendulum. Next, we repeated online learning for another nine episodes. Then, we restarted the trained controller with learning switched off. This time, the controller was more successful in achieving the task, without further learning, although having some remaining error that seemed to become worse again towards the end of the recorded time span (see Fig. 4.14(b)). As expected, this remaining error was removed when we restarted the simulation on the bases of additional learning experience (refer to Fig. 4.14(c)). The control signal generated by KI-SIIC in the last run is depicted in Fig. 4.14(d). Unsurprisingly, the control signal generated by kinky inference can indeed be "kinky", i.e. non-smooth. This in contrast to RF-SIIC which produced smooth control signals once learning was switched off. We attribute this behaviour to the kinks in the predictions of the kinky inference rule. However, in applications where this ought to be a problem, we expect to be able to alleviate this issue by replacing the vanilla KI rule by the smoothed version introduced in Sec. 4.2.5.

4.5.2. Kinky inference in inverse dynamics model learning (KI-IMLC)

So far, we have considered the learning of *forward* dynamic models of the form $\dot{x} = \psi(x, u)$ which we identified and then feedback linearised. An alternative that has become popular in robotics is *inverse (dynamics) model learning (IMLC)*. The inverse dynamics model is the mapping $\psi^- : \mathcal{X} \times \mathcal{X} \to \mathcal{U}$, $(x, \dot{x}_{\text{ref}}) \mapsto u$ where $\dot{x}_{\text{ref}}$ is a desired reference state derivative. In a configuration space formulation this translates to a mapping $(q, \dot{q}, \ddot{q}) \mapsto u$ which can be learned by observing data of the form $\mathcal{D} = \{\big((q_i, \dot{q}_i, \ddot{q}_i), u_i\big) | i = 1, ..., N\}$. While having the disadvantage of having to learn over a markedly higher dimensional space, the advantage is that being a black-box model, the IMLC approach has a great degree of flexibility. Also, because it does not linearise the dynamics, it might be able to benefit from the intrinsic properties of the dynamics rather than cancelling them.

Learning inverse dynamic models with standard Gaussian process regression has been considered in [60, 201]. Analogous approaches have considered the application of nonparametric learning methods to learning inverse kinematics models [86] or inverse operational space models [197]. To speed up computation of the IMLC problem, Tuong et. al. [182] propose to combine the predictions of many GPs (each having small data sets that contain localised information) to make local predictions at different parts of state space. A similar idea, but with linear prediction models, had been proposed in LWPR [221,257]. In their paper, Tuong



(a) Error evolution (1st trial).

(b) Error evolution (10 trials).

(c) Error evolution (20 trials).

(d) Control evolution (after learning).

Figure 4.14.: Fig. 4.14(a)- Fig. 4.14(d): Depicted are the squared distances to the goal as a function of time steps. The abscissa marks the time steps the squared distances were recorded. The step size was 0.01 seconds. The blue and green lines are the squared distances to the goal (i.e. errors) of the configurations q_1, q_2 (angles in radians), respectively. The red and cyan lines are the pertaining squared velocity errors. Fig. 4.14(a) shows the squared errors during the first episode of online learning. Fig. 4.14(b) shows the squared error evolution during control (with online learning switched off) on the basis of learning experience gathered over the course of 10 preceding learning episodes of 10 seconds each. Fig. 4.14(c) depicts this error evolution on the basis of 20 preceding episodes. Not surprisingly, the controller becomes better at controlling the plant with increased learning experience. The last plot depicts the control evolution (as a function of time in seconds) of the last episode (with learning switched off).

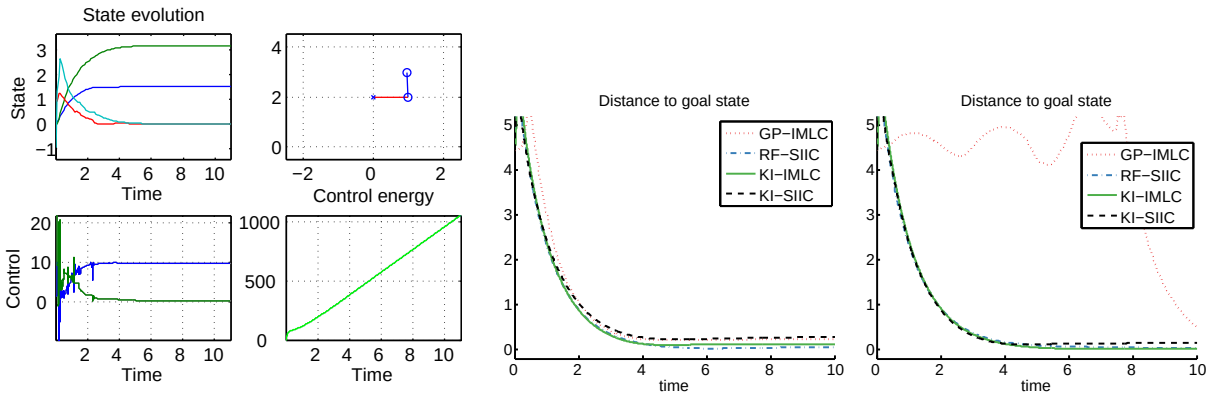
et. al. [182] compare a variety of learning methods against standard Gaussian process regression (GPR) for learning inverse dynamic models of real robots. Their data suggests that GPR requires high computational cost for learning and prediction but outperforms other methods such as LWPR [257] and LGP [182] in terms of prediction accuracy and tracking performance. Therefore, we use the standard method of GPR as a baseline method. We will refer to the resulting controller as *GP-IMLC* and compare it to a controller that substitutes the GP by our kinky inference learner. This new controller will be referred to as *kinky inference inverse model learning based control (KI-IMLC)*.

Note, in order to speed up computation, we could just as easily apply the localisation methods proposed in [182] to the KI-IMLC approach. This may also have the advantage that a variety of localised KI-IMLC models would benefit from local estimates of the Hölder constant. For now however, we restrict ourselves to the standard method. To test the viability of kinky inference for inverse dynamics model learning and control, we repeated our double-pendulum experiment from the previous subsection. This time, we replaced the KI-SSIC controller by KI-IMLC.

Note, without further modifications, KI-IMLC per se does not lend itself to online learning. To see this, consider a situation where at time t, the system is in a state $x(t) = [q(t); \dot{q}(t)]$ where the KI rule is very uncertain about the reference acceleration $\ddot{q}_{ref}$. Furthermore, if the present acceleration $\ddot{q}$ is very distinct from the desired one $\ddot{q}_{ref}$ (as measured by metric $\mathfrak{d}_{\mathcal{X}}$) then incorporation of a new sample point $\Big(\big(q(t), \dot{q}(t), \ddot{q}(t)\big), u\big(q(t), \dot{q}(t), \ddot{q}(t)\big)\Big)$ provides little information about the control $u\big((q(t), \dot{q}(t), \ddot{q}_{ref}\big)\big)$ for the reference acceleration $\ddot{q}_{ref}$. This may mean that the controller may fail to reach the goal. As a solution, it might be conceivable to feed the controller a reference trajectory that smoothly connects the true reference with the current state and acceleration or have a planner to find a trajectory that is close to explored state space while meeting the control objective. However, we will not consider this any further. Instead, we will confine our exposition to offline learning.

To this end, we generate an offline sample set $\mathcal{D}$ containing tuples of the form $\Big(\big(q, \dot{q}, \ddot{q}\big), u\Big)$. The sample inputs $\big(q, \dot{q}, \ddot{q}\big)$ were drawn uniformly at random from the hyperrectangle $[-10, 10]^4 \times [-50, 50]$. That is, the state training example inputs were allowed to assume values between -10 and 10 while we drew accelerations from the larger interval [-50,50]. The Hölder constant was adapted to the random sample utilising the lazy update rule outlined in Sec. 4.3.2. On the basis of the accordingly adapted KI rule, the pertaining control was determined by inverting the ground-truth dynamics with these inputs (and adding a bit of noise of magnitude up to 0.01).

We tested the control performance of KI-IMLC in our double-pendulum test-bed. Here, KI-IMLC made inference about the correct control on the basis of the sample $\mathcal{D}$. As a reference acceleration, we chose $\ddot{q}_{ref} = w(\xi_1 - q) + w(\xi_2 - \dot{q})$ where ξ_1 was the goal angle setpoint and ξ_2 the goal angular velocity.



(a) Double-pendulum controlled by KI-IMLC. (b) Error evolutions (1st try). (c) Error evolutions (2nd try).

Figure 4.15.: Fig. 4.15(b): Example of KI-IMLC being successful at driving a learned double-pendulum to goal state $\xi = [\pi; \pi/2; 0; 0]$. Fig. 4.15(b): Depicted is a comparison of the error evolutions, i.e. the distances to the goal as a function of time (seconds), of the control task solved by different methods. All methods managed to stabilise the system well on the basis of the randomly generated data set. Fig. 4.15(c) exposes a recurring issue we encountered with GP-IML. Depending on the randomly generated data set, GP-IMLC sometimes failed. We attribute this to the outcome of hyperparameter optimisation.

This choice means that in the limit of dense data, a perfectly trained KI-IMLC controller would control the system such that the closed-loop dynamics would mimic a double-integrator controlled by a P-controller with setpoint ξ.

As a first illustration, consider Fig. 4.15(a). On the basis of a random data set of size 20, and with reference parameter $w = 20$, we tasked KI-IMLC to control the double-pendulum to reach the goal state $\xi = [\pi; \pi/2, 0; 0]$. In this situation, KI-IMLC had no difficulty. As a first comparison, we repeated the experiment with the different the different algorithms GP-IMLC, RF-SIIC (from Ch. 3), KI-IMLC and KI-SIIC. For the purposes of the comparison, the SIIC methods were trained offline on the same randomly generated data set as the other methods. As in previous experiments, the initial state was $x_0 = [0; 0; -1; -1]$ and the goal state was chosen to be $\xi = [\pi; \pi/2, 0; 0]$. The random field methods, were allowed to perform hyperparameter optimisation. In the first run (Fig. 4.15(b)), all methods accomplished the task with little rest error. However, repeating experiments like this several times, we frequently encountered GP-IMLC struggling on a variety of instances. Inspecting the trained hyperparameters gives us reason to conjecture that this might be due to the fact that the optimisation hyperparameter optimisation procedure failed to produce good results and got stuck in a local minimum of the marginal posterior log-likelihood. Note, we did not register this problem with any of the other methods. One explanation might be the fact that, owing to the increased dimensionality of the input space in the IMLC setup, GP-IMLC had to optimise over a higher-dimensional space than RF-SIIC. As a note of caution though, in the present setup, both methods are not entirely comparable on a level playing field. This results from the fact that in the current setup, GP-IMLC

generated the control by imitating a feedback controller via pure learning. By contrast, the feedback pseudo-control employed in the SIIC method may be able the compensate for some learning error (provided the model over $b(x)$, as defined above, is not profoundly off).

While time did not permit this, in future work, we would like to make a more extensive comparison on higher-dimensional robotic tracking tasks. Here, we would endow the methods with forward references (instead of error feedback). We anticipate that this will increase comparability and shed light on the respective inherent benefits and drawbacks more clearly.

4.6. Conclusions

This chapter introduced the kinky inference framework as an approach to inductive inference in dynamical systems. Folding in knowledge about boundedness and Hölder continuity (with respect to an arbitrary pseudo-metric) the method has been demonstrated to be able to infer unobserved function values successfully. Assuming all observational or input uncertainties are interval-bounded, the method is capable of providing bounds around its predictions. That is, for any finite sample we have shown that the inferred function value is correct up to lying within a given error hyperrectangle. We have shown that this uncertainty vanishes (uniformly) in the limit of data sets that (uniformly) converge to a dense subset of input space. Provided no additional a priori knowledge about the target function is known, our bounds are provably tight. We have discussed several variations of the kinky inference rule, including a smooth version, the capability of handling input uncertainty and we have provided two methods for updating beliefs over the Hölder constant.

We have discussed applications to system identification and control in uncertain dynamical systems highlighting the benefits (and some of its shortcomings) of kinky inference in this application domain. In particular, the benefits are rapid learning and prediction, the error bounds around the predictions as well as the flexibility to learn rich function classes.

In the design of learning-based controllers, the uncertainty bounds afforded by our KI rule are a particularly useful feature. These allowed us to convert the prediction bounds into bounds on the error dynamics. That is, assuming the prior assumptions are correct (and no Hölder constant updates are necessary), we have obtained stability bounds for the closed-loop dynamics.

To highlight the applicability of KI in control, we have provided simulations for the different controllers in the model-reference adaptive control framework, in combination with our SIIC control approach of Ch. 3 and in inverse dynamics model learning and control. Apart from simple illustrations in the control of double-pendula, we have also provided simulations of KI-MRAC in tracking control of the rudder of an unstable fighter-aircraft in the presence of wingrock. Here, KI-MRAC proved competitive with recent state-of-the art algorithms in the test scenario considered in [63, 66, 67]. Furthermore, across the board of several hundred of

randomized problem instances, KI-MRAC largely outperformed all competing controllers.

In Ch. 6, we will present an additional application in the context of multi-agent control. There we will utilise the uncertainty bounds afforded by the kinky inference rule to construct robust tubes. These will be converted into robust collision avoidance constraints that are incorporated into the optimisation problem of a predictive controller. This construction will yield a multi-agent controller for KI learners that folds in the posterior knowledge about an uncertain drift.

Future work will be invested in extended tests on realistic, higher-dimensional systems and explore the benefits and weaknesses of the different approaches. To this end, we will employ results on input and observational noise to a greater extent than we had to in our proof-of-concept simulations. Other extensions of interest pertain to the Hölder constants. At the moment, these are global estimates. In a spirit similar to localised model learning [182, 257], it may be beneficial to entertain a number of local KI models, each basing their inferences and constant estimates on local information. This might be beneficial not only from a computational point of view, but also provide improved inferences if there is a large variation of the Hölder constant across state space. Furthermore, for estimating Hölder constants, we would like to compare our approaches to alternatives that are attempting to find Lipschitz constants on the basis of data by mathematical programming [25]. A related direction to this effort would be the adaptation of the Hölder exponent in addition to the Hölder constant. This is important in situations where a valid Hölder exponent is not known a priori and might certainly extend the black-box learning capabilities of our approach.

Another avenue would be to address the problem of data sparsification. That is, given a sample size budget, which data points are most informative to keep in the budgeted kinky inference approach? With regard to kNN-KI, we would like to test the practical benefits of this approach. In particular, we would like to try the impact of approximate nearest neighbour methods that offer guarantees of logarithmic search effort without needing prohibitively large effort for incremental learning which is of utmost importance in online learning scenarios such as the wingrock control application discussed in Sec. 4.4.2. To this end, we might consider modifying *locally sensitive hashing* [120] methods for fast nearest neighbour search. In this context, it might be of particular interest to build on ideas raised by Beliakov [24] and reduce the problem of computing the full KI rule to a variant of an additively weighted nearest neighbour problem.

With regard to control, we would like to work on more extensive guarantees for our KI-based controllers. For example we believe to be able to prove bounds on the error dynamics of KI-IMLC. As the treatment of [180] indicates, this would be the first time a guarantee on an inversion model learning-based controller would be given. Furthermore, we would like to test the viability of planning under constraints of bounded control.

Finally, we will investigate additional theoretical properties both of the full and the reduced sample KI

methods. For instance, we believe that, with the exception of minimality of the error bounds, the proof of Thm. 4.2.7 will go through for *kNN-KI* without much adaptation. In addition, we believe both *KI* and *kNN-KI* can yield *universal approximators*: Remember, in Sec. 4.3, we pointed out that in the limit $L \to \infty$, the predictor $\hat{f}_n(\cdot)$ becomes a step function. Since it is well-known that step functions are dense in the space of absolutely integrable functions and any continuous function with compact support is absolutely integrable, the set of all predictor functions that could be generated from data (with $L \to \infty$) is dense in the set of continuous functions with compact support. This means that in the limit case $L \to \infty$, KI becomes a universal approximator. We believe that in conjunction with our constant adaptation techniques of Sec. 4.3 we can give such universal approximation guarantees even for the case of finite initial Hölder constant that gets adapted to the data with our lazy or Bayesian update rules. Deriving this rigorously will be done in the context of future work.

5. Collision prediction and policy-search for multi-agent systems with uncertain continuous-time dynamics

"I can calculate the motions of heavenly bodies, but not the madness of people."

Isaac Newton

Existing work in multi-agent collision prediction and avoidance typically assumes discrete-time trajectories that are deterministic or have Gaussian or interval-bounded uncertainty due to an orthogonal increments process. In this chapter, we propose an approach that allows detection of collisions even between continuous, stochastic trajectories with the only restriction that means and upper bounds on the variance-covariances can be computed. To this end, we employ probabilistic bounds to derive criterion functions whose negative sign provably is indicative of probable collisions. For criterion functions that are Hölder continuous, algorithms are provided to find negative values or prove their absence. We propose an iterative policy-search approach that avoids prior discretisations and, upon termination, yields collision-free trajectories with adjustably high certainty. We test our method with both fixed-priority and auction-based protocols for coordinating the iterative planning process. Results are provided in collision-avoidance simulations of feedback controlled plants.

5.1. Introduction

Due to their practical importance, multi-agent collision avoidance and control have been extensively studied across different communities including AI, robotics and control. Considering continuous stochastic trajectories, reflecting each agent's uncertainty about its neighbours' time-indexed locations in an environment space, we exploit a distribution-independent bound on collision probabilities to develop a conservative collision-prediction module. It avoids temporal discretisation by stating collision-prediction as a one-dimensional

optimization problem. If mean and standard deviation are computable Lipschitz functions of time, one can derive Lipschitz constants that allow us to guarantee collision prediction success with low computational effort. This is often the case, for instance, when dynamic knowledge of the involved trajectories is available (e.g. maximum velocities or even the stochastic differential equations).

To avoid collisions detected by the prediction module, we let an agent re-plan repeatedly until no more collisions occur with a definable probability. Here, re-planning refers to modifying a control signal (influencing the basin of attraction and equilibrium point of the agent's stochastic dynamics) so as to bound the collision probability while seeking low plan execution cost in expectation. To keep the exposition concrete, we focus our descriptions on an example scenario where the plans correspond to sequences of setpoints of a feedback controller regulating an agent's noisy state trajectory. However, one can apply our method in the context of more general policy search problems.

In order to foster low social cost across the entire agent collective, we compare two different coordination mechanisms. Firstly, we consider a simple fixed-priority scheme [94], and secondly, we modify an auction-based coordination protocol [49] to work in our continuous setting. In contrast to pre-existing work in auction-style multi-agent planning (e.g. [49, 151]) and multi-agent collision avoidance (e.g. [15, 20, 142]), we avoid *a priori* discretizations of space and time. Instead, we recast the coordination problem as one of incremental open-loop policy search. That is, as a succession of continuous optimisation or root-finding problems that can be efficiently and reliably solved by modern optimisation and root-finding techniques (e.g. [124, 230]).

While our current experiments were conducted with linear stochastic differential equation (SDE) models with state-independent noise (yielding Gaussian processes), our method is also applicable to any situation where mean and (upper bounds on variances) can be evaluated. This encompasses non-linear, non-Gaussian cases that may have state-dependent uncertainties (cf. [101]).

In addition we discuss how our criterion function approach also is suitable to be applied in robust control. Here, we need to be able to compute the nominal trajectory as well as the radii of the robust tube at any given point in time. This renders our approach applicable to coordinating agents which entertain bounded beliefs over their dynamics informed by our kinky inference learning approach developed in Ch. 4.

Related Work

Multi-agent trajectory planning and task allocation methods have been related to auction mechanisms by identifying locations in state space with atomic goods to be auctioned in a sequence of repeated coordination rounds (e.g. [49, 151, 250]). Unfortunately, even in finite domains the coordination is known to be intractable – for instance the sequential allocation problem is known to be NP-hard in the number of goods and agents [139, 217]. Furthermore, collision avoidance corresponds to non-convex interactions.

This renders the coordination problem inapplicable to standard optimization techniques that rely on convexity of the joint state space. In recent years, several works have investigated the use of mixed-integer programming techniques for single- and multi-agent model-predictive control with collision avoidance both in deterministic and stochastic settings [49, 158]. To connect the problem to pre-existing mixed-integer optimization tools these works had to limit the models to dynamics governed by linear, time-discrete difference equations with state-independent state noise. The resulting plans were finite sequences of control inputs that could be chosen freely from a convex set. The controls gained from optimization are open-loop – to obtain closed-loop policies the optimization problems have to be successively re-solved on-line in a receding horizon fashion. However, computational effort may prohibit such an approach in multi-agent systems with rapidly evolving states.

Furthermore, prior time-discretisation comes with a natural trade-off. On the one hand, one would desire a high temporal resolution in order to limit the chance of missing a collision predictably occurring between consecutive time steps. On the other hand, communication restrictions, as well as poor scalability of mixed-integer programming techniques in the dimensionality of the input vectors, impose severe restrictions on this resolution. To address this trade-off, [89] proposed to interpolate between the optimized time steps in order to detect collisions occurring between the discrete time-steps. Whenever a collision was detected they proposed to augment the temporal resolution by the time-step of the detected collision thereby growing the state-vectors incrementally as needed. A detected conflict, at time t, is then resolved by solving a new mixed-integer linear programme over an augmented state space, now including the state at t. This approach can result in a succession of solution attempts of optimization problems of increasing complexity, but can nonetheless prove relatively computationally efficient. Unfortunately, their method is limited to linear, deterministic state-dynamics.

Another thread of works relies on dividing space into polytopes [15,154], while still others [61,84,142,164] adopt a potential field. In not accommodating uncertainty and stochasticity, these approaches are forced to be overly conservative in order to prevent collisions in real systems.

In contrast to all these works, we will consider a different scenario. Our exposition focuses on the assumption that each agent is regulated by exerting influence over its continuous stochastic dynamics. For instance, we might have a given feedback controller with which one can interact by providing a sequence of setpoints constituting the agent's plan. While this restricts the choice of control action, it also simplifies computation as the feedback law is fixed. The controller can generate a continuous, state-dependent control signal based on a discrete number of control decisions, embodied by the setpoints. Moreover, it renders our method applicable in settings where the agents' plants are controlled by standard off-the-shelf controllers (such as the omnipresent PID-controllers) rather than by more sophisticated customized ones. Instead of

imposing discreteness, we make the often more realistic assumption that agents follow continuous time-state trajectories within a given continuous time interval. Unlike most work [15, 164, 242, 253] in this field, we allow for stochastic dynamics, where each agent cannot be certain about the location of its team-members. This is crucial for many real-world multi-agent systems. The uncertainties are modelled as state-disturbances which can reflect physical disturbances or merely model inaccuracies. Of particular interest to this thesis is the the case where the uncertainty quantifies the agent's bounded epistemic belief over a trajectory. Here the uncertainty arises from a posterior belief over the governing physical law of the agent's plant. In Ch. 4 we discuss how estimates robust tubes can be obtained which can be used with the collision detection method described in what is to follow.

This chapter is an extended version of work that has been published in the proceedings of AAMAS'14 [52] and an earlier stage of this work was presented at an ICML [50] workshop.

5.2. Predictive probabilistic collision prediction with criterion functions

Task. Our aim is to design a collision-detection module that can decide whether a set of (predictive) stochastic trajectories is collision-free (in the sense defined below). It is designed to trigger an alarm whenever there exists a time such that the probability of a collision might be undesirably high. The module we will derive is guaranteed to make this decision conservatively, based on knowledge of the first and second order moments of the trajectories alone. In particular, no assumptions are made about the family of stochastic processes the trajectories belong to. As the required collision probabilities will generally have to be expressed as non-analytic integrals, we will content ourselves with a fast, *conservative* approach. That is, we are willing to tolerate a non-zero false-alarm-rate as long as decisions can be made rapidly and with zero false-negative rate. Of course, for certain distributions and plant shapes, one may derive closed-form solutions for the collision probability that may be less conservative and hence, lead to faster termination and shorter paths. In such cases, our derivations can serve as a template for the construction of criterion functions on the basis of the tighter probabilistic bounds.

Problem Formalization. Formally, a collision between two objects (or agents) $\mathfrak{a}, \mathfrak{r}$ at time $t \in I := [t_0, t_f] \subset \mathbb{R}$ can be described by the event

$$\mathfrak{C}^{\mathfrak{a},\mathfrak{r}}(t) = \{(s^{\mathfrak{a}}(t), s^{\mathfrak{r}}(t)) | \, \|s^{\mathfrak{a}}(t) - s^{\mathfrak{r}}(t)\|_2 \leq \frac{\Lambda^{\mathfrak{a}} + \Lambda^{\mathfrak{r}}}{2}\}. \tag{5.1}$$

Here, $\Lambda^{\mathfrak{a}}, \Lambda^{\mathfrak{r}}$ denote the objects' diameters, and $s^{\mathfrak{a}}, s^{\mathfrak{r}} : I \to \mathbb{R}^D$ are two (possibly uncertain) trajectories in a common, D-dimensional *interaction* space. The latter is a transformation of the state space of the underlying

plant dynamics where conflicts can arise. For instance, in a multi-UAV collision avoidance scenario, this could be the three-dimensional space of UAV locations.

In a stochastic setting, we desires to bound the collision probability below a threshold $\delta \in (0, 1)$ at any given time in I. We loosely say that, the trajectories are *(instantaneously) collision-free* if $\Pr[\mathfrak{C}^{\mathfrak{a},\mathfrak{r}}(t)] < \delta, \forall t \in I$.

An alternative notion would be the collision of entire trajectories. Here, the trajectories are called *(trajectory-wise) collision-free* if the trajectory constraint $\Pr[\bigcup_{t\in I} \mathfrak{C}^{\mathfrak{a},\mathfrak{r}}(t)] < \delta$.

For now, our exposition starts with focussing on the instantaneous notion of probabilistic collisions. However, later we will also discuss cases for which we can solve collision avoidance prediction and avoidance for entire trajectories, for both stochastic and interval-bounded uncertainties.

For point-wise case, we desire a procedure that, with finite computation, allows the agent $\mathfrak{a}$ to either verify conservatively that none of the point-wise probabilistic constraints in the continuously infinite set $\{\Pr[\mathfrak{C}^{\mathfrak{a},\mathfrak{r}}(t)] < \delta | t \in I\}$ are violated or to declare that some might be violated (triggering an "alarm"). Here, the word conservatively means that a declaration of the absence of any collisions need to be correct, whereas it might still be the case that none of the probabilistic constraints are violated despite an alarm being triggered. Of course ideally, the method should seek to reduce conservatism (that is the number of false-alarms) as much as possible, without becoming non-conservative or computationally intractable.

Approach. Assuming time index set $I \subset \mathbb{R}$ is infinite and computation is finite, checking the condition $\Pr[\mathfrak{C}^{\mathfrak{a},\mathfrak{r}}(t)] < \delta$ for every $t \in I \subset \mathbb{R}$ clearly is out of the question. To overcome this dilemma, we require inference that takes a model of the agents' dynamics into account. Smoothness conditions of the dynamics connect the constraints between different points in time and allow us to rule out . The idea is to convert the constraints into a negativity checking problem of a function. This *so-called* criterion function will be constructed from the probabilistic constraints such that it is provably positive if none of the constraints are violated. And, as we will see below, the regularity of the dynamics of the agents can often be translated into regularity conditions on the criterion function that allow the provable solution of the pertaining negativity checking problem with finite computation.

For conservative collision detection between two agents' stochastic trajectories $s^{\mathfrak{a}}, s^{\mathfrak{r}}$, we construct a *criterion function* $\gamma^{\mathfrak{a},\mathfrak{r}} : I \to \mathbb{R}$ (eq. as per Eq. 5.3 below). A conservative criterion function has the property $\gamma^{\mathfrak{a},\mathfrak{r}}(t) > 0 \Rightarrow \Pr[\mathfrak{C}^{\mathfrak{a},\mathfrak{r}}(t)] < \delta$. That is, a collision between the trajectories with probability above δ at any time can be ruled-out if $\gamma^{\mathfrak{a},\mathfrak{r}}$ attains only positive values. If one could evaluate the function $t \mapsto \Pr[\mathfrak{C}^{\mathfrak{a},\mathfrak{r}}(t)]$, an ideal criterion function would be

$$\gamma^{\mathfrak{a},\mathfrak{r}}_{\text{ideal}}(t) := \delta - \Pr[\mathfrak{C}^{\mathfrak{a},\mathfrak{r}}(t)]. \tag{5.2}$$

It is ideal in the sense that $\gamma^{\mathfrak{a},\mathfrak{r}}_{\text{ideal}}(t) > 0 \Leftrightarrow \Pr[\mathfrak{C}^{\mathfrak{a},\mathfrak{r}}(t)] < \delta$. However, in most cases, evaluating the criterion function in closed form will not be feasible. Therefore, we adopt a conservative approach: That is, we determine a criterion function $\gamma^{\mathfrak{a},\mathfrak{r}}(t)$ such that provably, we have $\gamma^{\mathfrak{a},\mathfrak{r}}(t) \leq \gamma^{\mathfrak{a},\mathfrak{r}}_{\text{ideal}}(t), \forall t$, including the possibility of false-alarms. That is, it is possible that for some times t, $\gamma^{\mathfrak{a},\mathfrak{r}}(t) \leq 0$, in spite of $\gamma^{\mathfrak{a},\mathfrak{r}}_{\text{ideal}}(t) > 0$.

Utilising the conservative criterion functions for collision-prediction, we assume a collision occurs unless $\min_{t\in I} \gamma^{\mathfrak{a},\mathfrak{r}}(t) > 0, \forall \mathfrak{r} \neq \mathfrak{a}$. If the trajectories' means and standard deviations are Hölder continuous functions of time then one can often show that $\gamma^{\mathfrak{a},\mathfrak{r}}$ is Hölder as well. In such cases negative values of $\gamma^{\mathfrak{a},\mathfrak{r}}$ can be found or ruled out provably based on a finite number of function evaluations. For the case of Lipschitz continuous criterion functions we will modify Shupert's optimization method [230] to provide a more rapid online algorithm for this purpose.

In situations where the criterion function is Lipschitz but its Lipschitz constant is unavailable or hard to determine, we could base our detection on the output of a global minimization method such as DIRECT [124]. Alternatively, a function of time with a known Lipschitz constant may be derivable whose values are an upper bound on the standard deviations of the stochastic trajectories under examination.

5.2.1. Proving the existence or absence of negative function values of a Hölder continuous function

Let $t_0, t_f \in \mathbb{R}, t_0 \leq t_f, I := [t_0, t_f] \subset \mathbb{R}$. Assume we are given a Hölder continuous *target function* $\gamma : I \to \mathbb{R}$ with Hölder constant $L \geq 0$ and exponent $p \in (0, 1]$. That is, $\forall S \subset I \, \exists L_S \leq L \, \forall x, x' \in S : |\gamma(x) - \gamma(x')| \leq L_S \, |x - x'|^p$. Let $t_0 < t_1 < t_2 < ... < t_N < t_f$ and define $G_N = (t_0, \ldots, t_{N+1})$ to be the *sample grid* of size $N + 2 \geq 2$ consisting of the inputs at which we choose to evaluate the target γ.

Our goal is to select function evaluations so as to prove or disprove the existence of a negative function value of target γ.

A naive algorithm

As a first, naive method, Alg. 1 leverages Hölder continuity to answer the question of positivity correctly after a finite number of function evaluations.

The algorithm evaluates the function values on a finite grid assuming a global Hölder constant L and exponent p. The grid is iteratively refined until either a negative function value is found or, the Hölder continuity of function γ allows us to infer that no negative function values can exist. The latter is the case whenever $\min_{t\in G_N} \gamma(t) > L \, \Delta^p$ where $G_N = (t_0, ..., t_{N+1})$ is the grid of function input (time) samples, $\Delta^p = |t_{i+1} - t_i| \, (i = 0, ..., N - 1)$ and $L > 0$ is a Hölder constant of the function $\gamma : (t_0, t_f) \to \mathbb{R}$ which is to be evaluated.

input : Domain boundaries $t_0, t_f \in \mathbb{R}$, function $\gamma : (t_0, t_f) \to \mathbb{R}$, Hölder constant $\mathsf{L} > 0$ and exponent $p \geq 0$.
output: Flag flag indicating presence of a non-positive function value (flag = 1 indicates existence of a non-positive function value; flag =0 indicates it has been ruled out that a negative function value can exist). Variable criticalTime contains the time of a non-positive function value if such exists (criticalTime $= t_0 - 1$, iff $\gamma((t_0, t_f)) \subset \mathbb{R}_+$).

flag $\leftarrow -1$; criticalTime $\leftarrow t_0 - 1$;
TimeGrid $\leftarrow \{t_0, t_f\}$;
$r \leftarrow -1$;
repeat
  $r \leftarrow r + 1$; $\Delta \leftarrow \frac{t_f - t_0}{2^r}$; $N \leftarrow (t_f - t_0)/\Delta$;
  TimeGrid $\leftarrow \cup_{i=0}^{N} \{t_0 + i\Delta\}$;
  minVal $\leftarrow \min_{t \in \text{TimeGrid}} \gamma(t)$;
  if minVal ≤ 0 **then**
    flag $\leftarrow 1$; criticalTime $\leftarrow \arg\min_{t \in \text{TimeGrid}} \gamma(t)$;
  else if minVal $>$ L Δ^p **then**
    flag $\leftarrow 0$;
until flag $= 1$ OR flag $= 0$;

Algorithm 1: Naive algorithm deciding whether a Lipschitz continuous function γ has a non-positive value on a compact domain. Note, if minVal $>$ L Δ^p the function is guaranteed to map into the positive real numbers exclusively.

The claim is established by the following Lemma:

Lemma 5.2.1. *Let $\gamma : [t_0, t_f] \subset \mathbb{R} \to \mathbb{R}$ be a Hölder continuous function with constant $L \geq 0$ and exponent $p \geq 0$. Furthermore, let $G_N = (t_0, t_1, \ldots, t_{N+1})$ be an equidistant grid with $\Delta = |t_{i+1} - t_i|$ $(i = 0, ..., N-1)$.*

We have, $\gamma(t) > 0, \forall t \in (t_0, t_f)$ if $\forall t \in G_N : \gamma(t) > L\,\Delta^p$.

Proof. By constructing a partition, this statement could follow from Eq. 2.45. However, we will prove the claim from first principles as follows: Since L is a Hölder constant of γ we have $|\gamma(t) - \gamma(t')| \leq L|t - t'|^p, \forall t, t' \in (t_0, t_f)$. Now, let $t^* \in (t_0, t_f)$ and $t_i, t_{i+1} \in G_N$ such that $t^* \in [t_i, t_{i+1}]$. Consistent with the premise of the implication we aim to show, we assume $\gamma(t_i), \gamma(t_{i+1}) > L\Delta^p$ and, without loss of generality, we assume $\gamma(t_i) \leq \gamma(t_{i+1})$. Let $\delta := |t_i - t^*|$. Since $t_i \leq t^* \leq t_{i+1}$ we have $\delta \leq \Delta$. Due to monotonocity of $t \mapsto t^p$, $\Delta^p \geq \delta^p$. Finally, $0 < L\Delta^p < |\gamma(t_i)|$ implies $\gamma(t^*) \geq \gamma(t_i) - |\gamma(t_i) - \gamma(t^*)| \geq \gamma(t_i) - L|t_i - t^*|^p > L\Delta^p - L\delta^p = L(\Delta^p - \delta^p) \geq 0$. □

Apart from a termination criterion, the lemma shows that larger Hölder constants will generally cause longer run-times of the algorithm as finer resolutions Δ will be required to ensure non-negativity of the function under investigation.

An improved adaptive algorithm for Lipschitz functions

Next, we will present an improved version of the algorithm provided above. At present, this improved version is restricted to the Lipschitz case, that is, $p = 1$. But, we have developed a similar method that works for general p. Not having tested it yet in simulation, its presentation will have to be deferred to future work.

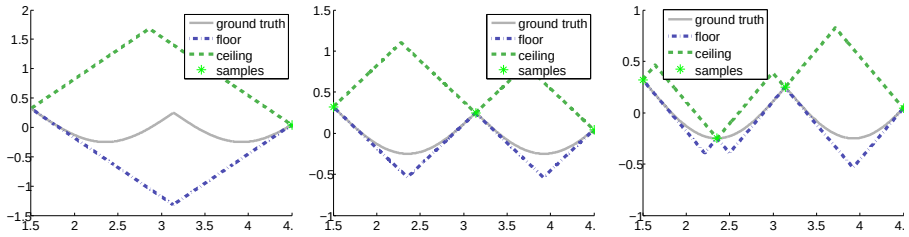


Figure 5.1.: Proving the existence of a negative value of function $x \mapsto |\sin(x)|\cos(x) + \frac{1}{4}$. Left: Initial condition. Centre: First refinement. Right: The second refinement has revealed the existence of a negative value.

We can define two functions, *ceiling* $\mathfrak{u}_N$ and *floor* $\mathfrak{l}_N$, such that (i) they bound the target $\forall t \in I : \mathfrak{l}_N(t) \leq \gamma(t) \leq \mathfrak{u}_N(t)$, and (ii) the bounds get tighter for denser grids. In particular, one can show that $\mathfrak{l}_N, \mathfrak{u}_N \overset{N\to\infty}{\longrightarrow} f$ uniformly if G_N converges to a dense subset of $[a, b]$. Define $\xi_N^{\mathfrak{l}} := \arg\min_{x\in I} \mathfrak{l}_N(x)$. It has been shown that $\xi_N^{\mathfrak{l}} = \min_{i=1}^{N-1} \frac{t_{i+1}+t_i}{2} - \frac{\gamma(t_{i+1})-\gamma(t_i)}{2L}$ and $\mathfrak{l}_N(\xi_N^{\mathfrak{l}}) = \min_i \frac{\gamma(t_{i+1})+\gamma(t_i)}{2} - L\frac{t_{i+1}-t_i}{2}$ (see Sec. C.4 as well as [124, 230]). It is trivial to refine this to take localised Lipschitz constants into account: $\xi_N^{\mathfrak{l}} = \min_i \frac{\gamma(t_{i+1})+\gamma(t_i)}{2} - L_{J_i}\frac{t_{i+1}-t_i}{2}$ where L_{J_i} is a Lipschitz number valid on interval $J_i = (t_i, t_{i+1})$.

This suggests the following algorithm: *We refine the grid G_N to grid G_{N+1}, by including $\xi_N^{\mathfrak{l}}, f(\xi_N^{\mathfrak{l}})$ as a new sample. This process is repeated until either of the following stopping conditions are met: (i) a negative function value of γ is discovered ($f(\xi_N^{\mathfrak{l}}) < 0$), or (ii) $\mathfrak{l}_N(\xi_N^{\mathfrak{l}}) \geq 0$ (in which case we are guaranteed that no negative function values can exist).*

input : Domain boundaries $t_0, t_f \in \mathbb{R}$, function $\gamma : (t_0, t_f) \to \mathbb{R}$, Lipschitz constant $\mathsf{L} > 0$.
output: Flag flag indicating presence of a non-positive function value (flag = 1 indicates existence of a non-positive function value; flag =0 indicates it has been ruled out that a negative function value can exist). Variable criticalTime contains the time of a non-positive function value if such exists (criticalTime $= t_0 - 1$, iff $\gamma((t_0, t_f)) \subset \mathbb{R}_+$).

flag $\leftarrow -1$; criticalTime $\leftarrow t_0 - 1$;
$G_N \leftarrow \{t_0, t_f\}$;
$N = 0$;
repeat
  $\xi^{\mathfrak{l}} \leftarrow \min_{i=1}^{N} \frac{t_{i+1}+t_i}{2} - \frac{\gamma(t_{i+1})-\gamma(t_i)}{2L}$;
  $\mathfrak{l}_N(\xi_N^{\mathfrak{l}}) \leftarrow \min_{i=1}^{N} \frac{\gamma(t_{i+1})+\gamma(t_i)}{2} - L\frac{t_{i+1}-t_i}{2}$;
  minVal $\leftarrow \gamma(\xi^{\mathfrak{l}})$;
  if minVal ≤ 0 **then**
    flag $\leftarrow 1$; criticalTime $\leftarrow \xi^{\mathfrak{l}}$;
  else if $\mathfrak{l}_N(\xi_N^{\mathfrak{l}}) > 0$ **then**
    flag $\leftarrow 0$;
  else
    $N \leftarrow N + 1$; $G_N \leftarrow G_N \cup \{\xi^{\mathfrak{l}}\}$;
  end
until flag $= 1$ OR flag $= 0$;

Algorithm 2: Adaptive algorithm based on Shubert's method to prove whether a Lipschitz continuous function γ has a non-positive value on a compact domain. Note, if $\mathfrak{l}_N(\xi_N^{\mathfrak{l}}) > 0$ the function is guaranteed to map into the positive reals exclusively.

For pseudo-code refer to Alg. 2.

An example run is depicted in Fig. 5.1. Note, without our stopping criteria, our algorithm degenerates to

Shubert's minimization method [230]. The stopping criteria are important to save computation, especially in the absence of negative function values.

5.2.2. Deriving collision criterion functions

This subsection is dedicated to the derivation of a criterion function. The idea is to define balls $H^{\mathfrak{a}}, H^{\mathfrak{r}}$ with respect to some norm, that are sufficiently large to contain a large enough proportion of each agent's probability mass to ensure that no collision occurs (with sufficient confidence) as long as the balls do not overlap. The balls will be defined by head radii of sufficient size (cf. Sec. 2.4.7). We then define a criterion function that assumes negative values whenever the balls do overlap.

For the most part of our discussion, it is convenient to consider the maximum-norm $\|\cdot\|_\infty$ causing the balls to be hyperrectangles. This allows us to employ tail inequalities to define the necessary radii.

For ease of notation, we omit the time index t. For instance, in this subsection, $x^{\mathfrak{a}}$ now denotes random vector $x^{\mathfrak{a}}(t)$ rather than the full stochastic trajectory.

The next thing we will do is to derive sufficient conditions for absence of collisions, i.e. for $\Pr[\mathfrak{C}^{\mathfrak{a},\mathfrak{r}}] < \delta$.

To this end, we make an intermediate step: For each agent $\mathfrak{q} \in \{\mathfrak{a}, \mathfrak{r}\}$ we define an open hyperrectangle $H^{\mathfrak{q}}$ centred around mean $\mu^{\mathfrak{q}} = \langle s^{\mathfrak{q}}(t) \rangle$. As a D-dimensional hyperrectangle, $H^{\mathfrak{q}}$ is completely determined by its centre point $\mu^{\mathfrak{q}}$ and its edge lengths $l_1^{\mathfrak{q}}, ..., l_D^{\mathfrak{q}}$. Let $O^{\mathfrak{q}}$ denote the event that $x^{\mathfrak{q}} \notin H^{\mathfrak{q}}$ and $P^{\mathfrak{q}} := \Pr[O^{\mathfrak{q}}]$. We derive a simple disjunctive constraint on the component distances of the means under which we can guarantee that the collision probability is not greater than the probability of at least one object being outside its hyperrectangle. This is the case if the hyperrectangles do not overlap. That is, their max-norm distance is at least $\Lambda^{\mathfrak{a},\mathfrak{r}} := \frac{\Lambda^{\mathfrak{a}}+\Lambda^{\mathfrak{r}}}{2}$.

Before engaging in a formal discussion we need to establish a preparatory fact:

Lemma 5.2.2. *Let $\mu_j^{\mathfrak{q}}$ denote the jth component of object $\mathfrak{q}$'s mean and $r_j^{\mathfrak{q}} = \frac{1}{2} l_j^{\mathfrak{q}}$. Furthermore, let $\mathfrak{F}^{\mathfrak{a},\mathfrak{r}} := \overline{\mathfrak{C}^{\mathfrak{a},\mathfrak{r}}}$ be the event that no collision occurs and $\mathfrak{B}^{\mathfrak{a},\mathfrak{r}} := H^{\mathfrak{a}} \times H^{\mathfrak{r}}$ the event that $x^{\mathfrak{a}} \in H^{\mathfrak{a}}$ and $x^{\mathfrak{r}} \in H^{\mathfrak{r}}$. Assume the component-wise distance between the hyperrectangles $H^{\mathfrak{a}}, H^{\mathfrak{r}}$ is at least $\Lambda^{\mathfrak{a},\mathfrak{r}}$, which is expressed by the following disjunctive constraint:*

$$\exists j \in \{1, ..., D\} : \left|\mu_j^{\mathfrak{a}} - \mu_j^{\mathfrak{r}}\right| > \Lambda^{\mathfrak{a},\mathfrak{r}} + r_j^{\mathfrak{a}} + r_j^{\mathfrak{r}}.$$

Then, we have : $\mathfrak{B}^{\mathfrak{a},\mathfrak{r}} \subset \mathfrak{F}^{\mathfrak{a},\mathfrak{r}}$.

Proof. Since $\|x\|_\infty \leq \|x\|_2, \forall x$ we have

$\mathfrak{F}_\infty := \{(x^{\mathfrak{a}}, x^{\mathfrak{r}}) | \, \|x^{\mathfrak{a}} - x^{\mathfrak{r}}\|_\infty > \Lambda^{\mathfrak{a},\mathfrak{r}}\} \subset \{(x^{\mathfrak{a}}, x^{\mathfrak{r}}) | \, \|x^{\mathfrak{a}} - x^{\mathfrak{r}}\|_2 > \Lambda^{\mathfrak{a},\mathfrak{r}}\} = \mathfrak{F}^{\mathfrak{a},\mathfrak{r}}$. It remains to be shown that $\mathfrak{B}^{\mathfrak{a},\mathfrak{r}} \subset \mathfrak{F}_\infty$: Let $(x^{\mathfrak{a}}, x^{\mathfrak{r}}) \in \mathfrak{B}^{\mathfrak{a},\mathfrak{r}} = H^{\mathfrak{a}} \times H^{\mathfrak{a}}$. Thus, $\forall j \in \{1, ..., D\}, \mathfrak{q} \in \{\mathfrak{a}, \mathfrak{r}\} : \left|x_j^{\mathfrak{q}} - \mu_j^{\mathfrak{q}}\right| \leq r_j^{\mathfrak{q}}$. For contradiction, assume $(x^{\mathfrak{a}}, x^{\mathfrak{r}}) \notin \mathfrak{F}_\infty$. Then, $|x_i^{\mathfrak{a}} - x_i^{\mathfrak{r}}| \leq \Lambda^{\mathfrak{a},\mathfrak{r}}$ for all $i \in \{1, ..., D\}$.

Hence, $|\mu_i^{\mathfrak{a}} - \mu_i^{\mathfrak{r}}| = |\mu_i^{\mathfrak{a}} - x_i^{\mathfrak{a}} + x_i^{\mathfrak{a}} - x_i^{\mathfrak{r}} + x_i^{\mathfrak{r}} - \mu_i^{\mathfrak{r}}| \leq |\mu_i^{\mathfrak{a}} - x_i^{\mathfrak{a}}| + |x_i^{\mathfrak{a}} - x_i^{\mathfrak{r}}| + |x_i^{\mathfrak{r}} - \mu_i^{\mathfrak{r}}| \leq r_i^{\mathfrak{a}} + \Lambda^{\mathfrak{a},\mathfrak{r}} + r_i^{\mathfrak{r}}, \forall i \in \{1, ..., D\}$ which contradicts our disjunctive constraint in the premise of the lemma. q.e.d.

□

Theorem 5.2.3. *Let $\mu_j^{\mathfrak{q}}$ denote the jth component of object $\mathfrak{q}$'s mean and $r_j^{\mathfrak{q}} = \frac{1}{2} l_j^{\mathfrak{q}}$. Assume, $s^{\mathfrak{a}}, s^{\mathfrak{r}}$ are random variables with means $\mu^{\mathfrak{a}} = \langle s^{\mathfrak{a}} \rangle, \mu^{\mathfrak{r}} = \langle s^{\mathfrak{r}} \rangle$, respectively. Assume the max-norm distance between hyperrectangles $H^{\mathfrak{a}}, H^{\mathfrak{r}}$ is at least $\Lambda^{\mathfrak{a},\mathfrak{r}} > 0$ (i.e. the hyperrectangles do not overlap), which is expressed by the following disjunctive constraint:*

$$\exists j \in \{1, ..., D\} : \left|\mu_j^{\mathfrak{a}} - \mu_j^{\mathfrak{r}}\right| > \Lambda^{\mathfrak{a},\mathfrak{r}} + r_j^{\mathfrak{a}} + r_j^{\mathfrak{r}}.$$

Then, we have :

$$\Pr[\mathfrak{C}^{\mathfrak{a},\mathfrak{r}}] \leq P^{\mathfrak{a}} + P^{\mathfrak{r}} - P^{\mathfrak{r}} P^{\mathfrak{a}} \leq P^{\mathfrak{a}} + P^{\mathfrak{r}}$$

where $P^{\mathfrak{q}} = \Pr[x^{\mathfrak{q}} \notin H^{\mathfrak{q}}], (\mathfrak{q} \in \{\mathfrak{a}, \mathfrak{r}\})$ and, as before, $\mathfrak{C}^{\mathfrak{a},\mathfrak{r}}$ is the event of an instantaneous collision at the time in question (cf. Eq. 5.1).

Proof. As in Lem. 5.2.2, let $\mathfrak{F}^{\mathfrak{a},\mathfrak{r}} := \overline{\mathfrak{C}^{\mathfrak{a},\mathfrak{r}}}$ be the event that no collision occurs and let $\mathfrak{B}^{\mathfrak{a},\mathfrak{r}} := H^{\mathfrak{a}} \times H^{\mathfrak{r}}$. We have $\Pr[\mathfrak{C}^{\mathfrak{a},\mathfrak{r}}] \leq 1 - \Pr[\overline{\mathfrak{C}^{\mathfrak{a},\mathfrak{r}}}] = 1 - \Pr[\mathfrak{F}^{\mathfrak{a},\mathfrak{r}}]$. By Lem. 5.2.2 we have $\mathfrak{B}^{\mathfrak{a},\mathfrak{r}} \subset \mathfrak{F}^{\mathfrak{a},\mathfrak{r}}$ and thus, $1 - \Pr[\mathfrak{F}^{\mathfrak{a},\mathfrak{r}}] \leq 1 - \Pr[\mathfrak{B}^{\mathfrak{a},\mathfrak{r}}] = \Pr[\overline{\mathfrak{B}^{\mathfrak{a},\mathfrak{r}}}]$. Now, $\Pr[\overline{\mathfrak{B}^{\mathfrak{a},\mathfrak{r}}}] = \Pr[x^{\mathfrak{a}} \notin H^{\mathfrak{a}} \vee x^{\mathfrak{r}} \notin H^{\mathfrak{r}}] = P^{\mathfrak{a}} + P^{\mathfrak{r}} - P^{\mathfrak{a}} P^{\mathfrak{r}} \leq P^{\mathfrak{a}} + P^{\mathfrak{r}}$. q.e.d. □

One way to define a criterion function is as follows:

$$\gamma^{\mathfrak{a},\mathfrak{r}}(t; \varrho(t)) := \max_{i=1,...,D} \{|\mu_i^{\mathfrak{a}}(t) - \mu_i^{\mathfrak{r}}(t)| - \Lambda^{\mathfrak{a},\mathfrak{r}} - r_i^{\mathfrak{a}}(t) - r_i^{\mathfrak{r}}(t)\} \tag{5.3}$$

where $\varrho = (r_1^{\mathfrak{a}}, \ldots, r_D^{\mathfrak{a}}, r_1^{\mathfrak{r}}, \ldots, r_D^{\mathfrak{r}})$ is the parameter vector of radii. (For notational convenience, we will often omit explicit mention of parameter ϱ in the function argument.)

For more than two agents, agent $\mathfrak{a}'s$ overall criterion function is $\Gamma^{\mathfrak{a}}(t) := \min_{\mathfrak{r} \in \mathfrak{A} \setminus \{\mathfrak{a}\}} \gamma^{\mathfrak{a},\mathfrak{r}}(t)$.

Thm. 5.2.3 tells us that the collision probability is bounded from above by the desired threshold δ if $\gamma^{\mathfrak{a},\mathfrak{r}}(t) > 0$, provided we chose the radii $r_j^{\mathfrak{q}}, r_j^{\mathfrak{r}}$ $(j = 1, ..., D)$ such that $P^{\mathfrak{q}} \leq \frac{\delta}{d^{\mathfrak{q}}}$ where $d^{\mathfrak{a}}, d^{\mathfrak{r}} \in [0, 1]$ are coefficients dividing up probability bound δ between the two agents. That is, they are set such that $d^{\mathfrak{a}} + d^{\mathfrak{r}} = 1$.

For instance, one could partition the probability mass in proportion to the relative uncertainty as measured by a variance matrix norm. That is, one could set $d^{\mathfrak{q}} = \frac{\|C^{\mathfrak{a}}\|}{\|C^{\mathfrak{a}}\| + \|C^{\mathfrak{r}}\|}$ where $C^{\mathfrak{q}}$ is agent $\mathfrak{q}$'s variance matrix. Such a split is advantageous if the uncertainties of $\mathfrak{a}$ and $\mathfrak{r}$ differ much at time t. Of course, we can always divvy up the tolerable probability bound δ evenly throughout the remainder this chapter, that is $d^{\mathfrak{q}} = \frac{\delta}{2}$.

Let $\mathfrak{q} \in \{\mathfrak{a}, \mathfrak{r}\}$. Probability theory provides several distribution-independent bounds relating the radii of a (possibly partly unbounded) hyperrectangle to the probability of not falling into it (cf. Sec. 2.4.7). In general,

these are bounds of the form

$$P^{\mathfrak{q}} \leq \beta(r_1^{\mathfrak{q}}, ..., r_D^{\mathfrak{q}}; \Theta)$$

where β is a continuous function that decreases monotonically with increasing radii and Θ represents additional information. In the case of Chebyshev-type bounds information about the first two moments are folded in, i.e. $\Theta = (\mu^{\mathfrak{q}}, C^{\mathfrak{q}})$ where $C^{\mathfrak{q}}(t) \in \mathbb{R}^{D \times D}$ is the variance (-covariance) matrix of $x^{\mathfrak{q}}(t)$ pertaining to agent $\mathfrak{q}$. We then solve for radii that fulfil the inequality $\frac{\delta}{d^{\mathfrak{q}}} \geq \beta(r_1^{\mathfrak{q}}, ..., r_D^{\mathfrak{q}}; \Theta)$ while simultaneously ensuring collision avoidance with the desired probability.

Inspecting Eq. 5.3, it becomes clear that, in order to maximally diminish conservatism of the criterion function, it would be ideal to choose the radii in ϱ such that

$\varrho = \text{argmax}_{\varrho} \gamma^{\mathfrak{a},\mathfrak{r}}(t; \varrho) = \text{argmax}_{r_1^{\mathfrak{a}},...,r_D^{\mathfrak{a}} r_1^{\mathfrak{r}},...,r_D^{\mathfrak{r}}} \max_{i=1,...,D} \{|\mu_i^{\mathfrak{a}} - \mu_i^{\mathfrak{r}}| - \Lambda^{\mathfrak{a},\mathfrak{r}} - r_i^{\mathfrak{a}} - r_i^{\mathfrak{r}}\}$ subject to the constraints $\frac{\delta}{d^{\mathfrak{q}}} \geq \beta(r_1^{\mathfrak{q}}, ..., r_D^{\mathfrak{q}}; \Theta), (\mathfrak{q} \in \{\mathfrak{a}, \mathfrak{r}\})$.

In situations where β is derived from a Chebyshev-type bound, we propose to set as many radii as large as possible (in order to decrease β in the constraint) while setting the radii $r_i^{\mathfrak{a}}, r_i^{\mathfrak{r}}$ in one dimension i as small as possible without violating the constraint. That is, we define the radii as follows: Set $r_j^{\mathfrak{q}} := \infty, \forall j \neq i$. The remaining unknown variable, $r_i^{\mathfrak{q}}$, then is defined as the solution to the equation $\frac{\delta}{d^{\mathfrak{q}}} = \beta(r_1^{\mathfrak{q}}, ..., r_D^{\mathfrak{q}}; \Theta)$. The resulting criterion function, denoted by $\gamma_i^{\mathfrak{a},\mathfrak{r}}$, we obtain with this procedure of course depends on the arbitrary choice of dimension i. Therefore, we obtain a less conservative criterion function by repeating this process for each dimension i and then constructing a new criterion function as the point-wise maximum: $\gamma^{\mathfrak{a},\mathfrak{r}}(t) := \max_i \gamma_i^{\mathfrak{a},\mathfrak{r}}(t)$.

A concrete example of this procedure is provided below.

Example constructions of distribution-independent criterion functions

We can use the above derivation as a template for generating criterion functions.

Consider the following concrete example. Combining union bound and the standard (one-dim.) Chebyshev bound yields $P^{\mathfrak{q}} = \Pr[s^{\mathfrak{q}} \notin H^{\mathfrak{q}}] \leq \sum_{j=1}^{D} \frac{C_{jj}^{\mathfrak{q}}}{r_j^{\mathfrak{q}} r_j^{\mathfrak{q}}} =: \beta(r_1^{\mathfrak{q}}, \ldots, r_D^{\mathfrak{q}}; C^{\mathfrak{q}})$. Setting every radius, except $r_i^{\mathfrak{q}}$, to infinitely large values and β equal to $\frac{\delta}{d^{\mathfrak{q}}}$ yields $\frac{\delta}{d^{\mathfrak{q}}} = \frac{C_{ii}^{\mathfrak{q}}}{r_i^{\mathfrak{q}} r_i^{\mathfrak{q}}}$, i.e. $r_i^{\mathfrak{q}} = \sqrt{\frac{d^{\mathfrak{q}} C_{ii}^{\mathfrak{q}}}{\delta}}$. Finally, inserting these radii (for $\mathfrak{q} = \mathfrak{a}, \mathfrak{r}$) into Eq. 5.3 yields our first collision criterion function:

$\gamma^{\mathfrak{a},\mathfrak{r}}(t) := |\mu_i^{\mathfrak{a}}(t) - \mu_i^{\mathfrak{r}}(t)| - \Lambda^{\mathfrak{a},\mathfrak{r}} - \sqrt{\frac{d^{\mathfrak{a}}(t)\, C_{ii}^{\mathfrak{a}}(t)}{\delta}} - \sqrt{\frac{d^{\mathfrak{r}}(t)\, C_{ii}^{\mathfrak{r}}(t)}{\delta}}$.

Of course, this argument can be made for any choice of dimension i. Hence, a less conservative, yet valid, choice is

$$\gamma^{\mathfrak{a},\mathfrak{r}}(t) := \max_{i=1,...,D} |\mu_i^{\mathfrak{a}}(t) - \mu_i^{\mathfrak{r}}(t)| - \Lambda^{\mathfrak{a},\mathfrak{r}} - \sqrt{\frac{d^{\mathfrak{a}}(t)\, C_{ii}^{\mathfrak{a}}(t)}{\delta}} - \sqrt{\frac{d^{\mathfrak{r}}(t)\, C_{ii}^{\mathfrak{r}}(t)}{\delta}}. \tag{5.4}$$

For simplicity we can always choose $d^{\mathfrak{q}}(t) = \frac{1}{2}$ in which case the criterion function reduces to:

$$\gamma^{\mathfrak{a},\mathfrak{r}}(t) := \max_{i=1,\ldots,D} |\mu_i^{\mathfrak{a}}(t) - \mu_i^{\mathfrak{r}}(t)| - \Lambda^{\mathfrak{a},\mathfrak{r}} - \sqrt{\frac{2C_{ii}^{\mathfrak{a}}(t)}{\delta}} - \sqrt{\frac{2C_{ii}^{\mathfrak{r}}(t)}{\delta}}. \tag{5.5}$$

A discussion of the derivation of Lipschitz numbers as well as Hölder constants and exponents for this function refer is contained in Sec. 5.2.3 below.

For the special case of two dimensions, we can derive a less conservative alternative criterion function based on a tighter two-dimensional Chebyshev-type bound [263]:

Theorem 5.2.4 (Alternative collision criterion function)**.** *Let spatial dimensionality be $D = 2$. Choosing*
$r_i^{\mathfrak{q}}(t) := \sqrt{\frac{1}{2\delta}} \sqrt{C_{ii}^{\mathfrak{q}}(t) + \frac{\sqrt{C_{ii}^{\mathfrak{q}}(t)C_{jj}^{\mathfrak{q}}(t)(C_{ii}^{\mathfrak{q}}(t)C_{jj}^{\mathfrak{q}}(t) - (C_{ij}^{\mathfrak{q}}(t))^2)}}{C_{jj}^{\mathfrak{q}}(t)}}$
($\mathfrak{q} \in \{\mathfrak{a},\mathfrak{r}\}, i \in \{1,2\}, j \in \{1,2\} - \{i\}$) in Eq. 5.3 yields a valid distribution-independent criterion function. That is, $\gamma^{\mathfrak{a},\mathfrak{r}}(t) > 0 \Rightarrow \Pr[\mathfrak{C}^{\mathfrak{a},\mathfrak{r}}(t)] < \delta$.

We provide a proof sketch and a Lipschitz constant (for non-zero uncertainty) in [55] and we have given a full derivation in the context of discrete-time SMPC in [159]. Note, the Lipschitz constant we have derived therein becomes infinite in the limit of vanishing variance. This can either be solved by deriving a Hölder exponent and constant or circumventent by replacing the function around its roots by a suitable upper bound. For more details on both approaches refer to Sec. 5.2.3.

A criterion function for Gaussian trajectories

The idea of our derivations above was to define hyperrectangles $H^{\mathfrak{a}}, H^{\mathfrak{r}}$ sufficiently large to contain a large enough proportion of each agent's probability mass to ensure that no collision occurs (with sufficient confidence) as long as the rectangles do not overlap. We then defined the criterion function that attain negative values whenever the hyperrectangles do overlap. The size of the rectangles were determined on the basis of distribution-independent Chebyshev bounds. Of course, these trajectories can be quite conservative. For instance, if the trajectories $x^{\mathfrak{a}}$ are Gaussian processes, the derivations made thus far can be made on the basis of the tighter tail bounds. Assign $d^{\mathfrak{q}} = \frac{1}{2}$ for all agents $\mathfrak{q}$. Following through with the same type of argument as above (but for Euclidean balls instead of hyperrectangles) we derive a criterion function of the form $\gamma^{\mathfrak{a},\mathfrak{r}}(t;\varrho^{\mathfrak{a}},\varrho^{\mathfrak{r}}) := \{\|\mu^{\mathfrak{a}}(t) - \mu^{\mathfrak{r}}(t)\|_2 - \Lambda^{\mathfrak{a},\mathfrak{r}} - \varrho^{\mathfrak{a}}(t) - \varrho^{\mathfrak{r}}(t)\}$. Here, $\varrho^{\mathfrak{q}}$ is a radius such that $\Pr[s^{\mathfrak{q}} \notin \mathfrak{B}_{\varrho^{\mathfrak{q}}}(\langle s^{\mathfrak{q}} \rangle)] \leq \frac{\delta}{2}$ (and where $\mathfrak{B}_{\varrho^{\mathfrak{q}}}(s)$ denotes a Euclidean ball with radius $\varrho^{\mathfrak{q}}$ centred at s).

Leveraging Cor. 2.4.33 and rearringing for the radius yields $\varrho^{\mathfrak{q}} \geq \sqrt{2\|\!|C^{\mathfrak{q}}|\!\|_2 \log(\frac{4}{\delta})}$.

To render the criterion function as large as possible (to reduce false detections), the radius is chosen as small as possible. That is,

$$\varrho^{\mathfrak{q}} = \sqrt{2\|\!|C^{\mathfrak{q}}|\!\|_2 \log(\frac{4}{\delta})}. \tag{5.6}$$

Inserting this into the definition of $\gamma^{\mathfrak{a},\mathfrak{r}}$ yields the criterion function:

$$\gamma^{\mathfrak{a},\mathfrak{r}}_{\mathcal{N}}(t) := \|\mu^{\mathfrak{a}}(t) - \mu^{\mathfrak{r}}(t)\|_2 - \Lambda^{\mathfrak{a},\mathfrak{r}} - \sqrt{2|||C^{\mathfrak{a}}(t)|||_2 \log(\frac{4}{\delta})} - \sqrt{2|||C^{\mathfrak{r}}(t)|||_2 \log(\frac{4}{\delta})}. \tag{5.7}$$

This is a criterion function for Gaussian trajectories as desired.

Being based on the tighter (sub-) Gaussian tail bound, it can be tighter than the Chebyshev-type bounds if the differences between the largest and smallest variance are small. In case that difference is large however, the fact that the criterion funtion only takes into account the largest Eigenvalue of the variance-covariance matrix $C^{\mathfrak{a}}$ can mean that this bound is unnecessarily conservative in directions of lower variance. In such cases, we recommend using the generic criterion function (based on hyperrectangles) as per Eq. 5.3 but in conjunction with the Gaussian tail radii as per Eq. 5.6 for each different dimension. This yields a criterion function

$$\gamma^{\mathfrak{a},\mathfrak{r}}_{\mathcal{N}}(t; \varrho(t)) := \max_{i=1,\dots,D} \Big\{ |\mu^{\mathfrak{a}}_i(t) - \mu^{\mathfrak{r}}_i(t)| - \Lambda^{\mathfrak{a},\mathfrak{r}} - \sqrt{2C^{\mathfrak{a}}_{ii} \log(\frac{4}{\delta})} - \sqrt{2C^{\mathfrak{r}}_{ii} \log(\frac{4}{\delta})} \Big\}. \tag{5.8}$$

Criterion functions for entire trajectories based on martingale considerations

So far, we have derived collision criterion functions that allow us to rule out violations of the *instantaneous* collision constraints $\Pr[\mathfrak{C}^{\mathfrak{a},\mathfrak{r}}(t)] < \delta$ for all t. However, in continuous time, that does not necessarily tell us much about the probability of the trajectories ever getting to close to each other within the given time interval $\mathcal{I}_t = [t_0, t_f]$. That is, we might be interested in

$$\Pr[\mathfrak{C}^{\mathfrak{a},\mathfrak{r}}(\mathcal{I}_t)] < \delta \tag{5.9}$$

where $\mathfrak{C}^{\mathfrak{a},\mathfrak{r}}(\mathcal{I}_t) := \{(s^{\mathfrak{a}}(t), s^{\mathfrak{r}}(t)) | \exists t \in \mathcal{I}_t : \|s^{\mathfrak{a}}(t) - s^{\mathfrak{r}}(t)\|_2 \leq \Lambda^{\mathfrak{a},\mathfrak{r}}\}$ is the event that the two agents $\mathfrak{a}, \mathfrak{r}$ collide at *some* point within the time span of interest $\mathcal{I}_t$. Bounding this probability could be attempted by investigating first-exit times or maximal inequalities. Unfortunately, to the best of our knowledge, both are areas of active research and results are usually limited to one-dimensional Gaussian or sub-Gaussian processes of certain types [2, 16, 23, 31, 64, 246, 247]. For instance, bounds on the supremum of a a class of one-dimensional diffusion processes on a compact time interval are given in [31], Thm. 5.4.1. Unfortunately, the bound is one-sided and not directly applicable for our purposes. In stochastic control (e.g. [147]), Lyapunov arguments have been utilised to show m.s. convergence of Ito stochastic processes with multiplicative noise. Typically these methods leverage martingale type arguments of one form or another. In future work, we might explore in greater detail how to harness such existing approaches to derive tubes.

For now, we will confine our exposition to a special class of process to illustrate the construction of

continuous-time tubes that yield a desired criterion function for the constraint in (5.9). Linking to our considerations of Sec. 2.4.10, we consider the stochastic interaction-space dynamics of agent $\mathfrak{q}$ given by the Ito stochastic initial value problem (cf. Eq. 2.41):

$$\mathrm{d}s^{\mathfrak{q}}(t) = \mu^{\mathfrak{q}}(t)\,dt\; + g(t)\,\mathrm{d}W(t) \tag{5.10}$$

$$s^{\mathfrak{q}}(0) = s_0^{\mathfrak{q}} \tag{5.11}$$

where $g(t)$ is a square integrable, non-anticipating stochastic process with a.s. continuous sample paths. Moreover, we assume $\mu^{\mathfrak{q}}(\cdot)$ to be a deterministic trajectory. For instance, it could represent some open-loop control that is determined in advance to direct the state along a desired nominal trajectory in expectation.

Remember, by Thm. 2.4.42, for $r > 0$, $\mathfrak{q} \in \mathfrak{A}$ we have

$$\Pr[\max_{k\in\mathcal{I}_t} \|s^{\mathfrak{q}}(t) - \langle s^{\mathfrak{q}}(t)\rangle\| \geq r] \leq \frac{\mathrm{tr}\big(\mathrm{Var}[s^{\mathfrak{q}}(t_f)]\big)}{r^2}.$$

Therefore, with probability of at least $1-\delta^{\mathfrak{q}}$ agent $\mathfrak{q}$'s stochastic trajectory does not leave tube $\mathbb{T}^{\mathfrak{q}} = \{s^{\mathfrak{a}}(\cdot)|\forall t \in \mathcal{I}_t : \|s^{\mathfrak{q}}(t) - \langle s^{\mathfrak{q}}(t)\rangle\| \leq \varrho^{\mathfrak{q}}\}$ where

$$\varrho^{\mathfrak{q}} = \sqrt{\frac{\mathrm{tr}\big(\mathrm{Var}[s^{\mathfrak{q}}(t_f)]\big)}{\delta^{\mathfrak{q}}}}. \tag{5.12}$$

Since $\|\cdot\|_\infty \leq \|\cdot\|_2$, $\|\cdot\|$ can denote the Euclidean or maximum norm.

That is, with probability at least $1-\delta^{\mathfrak{q}}$ we have $s^{\mathfrak{q}}(t) \in \mathbb{T}^{\mathfrak{q}}, \forall t \in \mathcal{I}_t$.

Between two tubes $\mathbb{T}^{\mathfrak{a}}, \mathbb{T}^{\mathfrak{r}}$ we can define a "distance" $\tilde{\mathfrak{d}}(\mathbb{T}^{\mathfrak{a}}, \mathbb{T}^{\mathfrak{r}})$ as the minimal distance between any two interaction space points within the time interval $\mathcal{I}_t$. That is, $\tilde{\mathfrak{d}}(\mathbb{T}^{\mathfrak{a}}, \mathbb{T}^{\mathfrak{r}}) = \inf_{t\in\mathcal{I}_t} \mathfrak{d}(s^{\mathfrak{a}}(t), s^{\mathfrak{r}}(t))$ where $\mathfrak{d}$ is a metric on the interaction space induced by a norm such as the Euclidean or the maximum norm.

Following the same type of reasoning as above, we see that if the agents choose controls such that their tubes have a distance of at least $\Lambda^{\mathfrak{a},\mathfrak{r}}$ and such that the probability of the event $s^{\mathfrak{a}} \in \mathbb{T}^{\mathfrak{a}}_{\varrho^{\mathfrak{a}}} \wedge s^{\mathfrak{r}} \in \mathbb{T}^{\mathfrak{r}}_{\varrho^{\mathfrak{r}}}$ is at least $1-\delta$ then the probability of a collision is less than δ and thus, our collision avoidance constraint as per Eq. 5.9 is guaranteed to be satisfied.

As above, it is easy to see that $\Pr[s^{\mathfrak{a}} \in \mathbb{T}^{\mathfrak{a}}_{\varrho^{\mathfrak{a}}} \wedge s^{\mathfrak{r}} \in \mathbb{T}^{\mathfrak{r}}_{\varrho^{\mathfrak{r}}}] \geq 1-\delta$ holds whenever $\Pr[s^{\mathfrak{a}} \notin \mathbb{T}^{\mathfrak{a}}_{\varrho^{\mathfrak{a}}}] < \delta^{\mathfrak{a}} \wedge \Pr[s^{\mathfrak{r}} \notin \mathbb{T}^{\mathfrak{r}}_{\varrho^{\mathfrak{r}}}] < \delta^{\mathfrak{r}}$ such that $\delta^{\mathfrak{a}} + \delta^{\mathfrak{r}} = \delta$.

So, if the agent dynamics are of the type given in Eq. 5.10, picking the tube radii as per Eq. 5.12 with choices $\delta^{\mathfrak{a}} + \delta^{\mathfrak{r}} = \delta$ guarantees collision avoidance with probability at least $1-\delta$ as desired.

That is, a valid criterion function is:

$$\gamma^{\mathfrak{a},\mathfrak{r}}(t) = \|\langle s^{\mathfrak{a}}(t)\rangle - \langle s^{\mathfrak{r}}(t)\rangle\| - \Lambda^{\mathfrak{a},\mathfrak{r}} - \sqrt{\frac{\mathrm{tr}\big(\mathrm{Var}[s^{\mathfrak{a}}(t_f)]\big)}{\delta^{\mathfrak{a}}}} - \sqrt{\frac{\mathrm{tr}\big(\mathrm{Var}[s^{\mathfrak{r}}(t_f)]\big)}{\delta^{\mathfrak{r}}}}. \tag{5.13}$$

Similarly, to our argument for the point-wise case above, it is easy to see that it suffices to pick $\delta^{\mathfrak{a}}, \delta^{\mathfrak{r}}$, which in turn define the radii of the tubes (cf Eq. 5.12), such that $\delta^{\mathfrak{a}} + \delta^{\mathfrak{r}} = \delta$.

One might choose to divide up the allowable exit probability mass δ according to the relative uncertainty: $\delta^{\mathfrak{a}} = \frac{\mathrm{tr}\big(\mathrm{Var}[s^{\mathfrak{a}}(t_f)]\big)}{\mathrm{tr}\big(\mathrm{Var}[s^{\mathfrak{a}}(t_f)]\big) + \mathrm{tr}\big(\mathrm{Var}[s^{\mathfrak{r}}(t_f)]\big)}$. Or, for simplicity, it might be possible to simply divvy it up evenly, that is $\delta^{\mathfrak{a}} = \frac{\delta}{2}$.

A criterion function for robust control

In this chapter, our main focus has been on stochastic control. That is, the trajectories' uncertainties are given by probability density functions and controls are constrained by the desire to satisfy a probabilistic belief over the collision event. Robust control pertains to the situation where the uncertainties are known to be bounded but otherwise the distribution may not necessarily be known. In the absence of further distributional knowledge we desire to find control actions that guarantee $\Pr[\mathfrak{C}^{\mathfrak{a},\mathfrak{r}}(t)] = 0, \forall t$ regardless of the distribution on the bounded domain the trajectory is known to lie within. (Note, the trajectory will follow some probability distribution. In the deterministic case, its density at each point in time is a dirac distribution.) This can often be guaranteed when the distributions are confined to regions of compact support.

The hard collision constraint is satisfied for any distributions with compact support if the support maintain a minimum clearance of $\Lambda^{\mathfrak{a},\mathfrak{r}}$. Therefore, the collision constraint is enforced by demanding that the supports maintain this clearance at all times.

To give a example of a resulting criterion function, assume that we know that a.s. we have $s^{\mathfrak{q}} \in H^{\mathfrak{q}}(t)$ where $H^{\mathfrak{q}}(t)$ is the hyperrectangle $H^{\mathfrak{q}}(t) = \mu^{\mathfrak{q}}(t) + ([-r_1^{\mathfrak{q}}(t), r_1^{\mathfrak{q}}(t)] \times \ldots \times [-r_d^{\mathfrak{q}}(t), r_d^{\mathfrak{q}}(t)])$ centred around the nominal position $\mu^{\mathfrak{q}}(t)$.

Clearly, the trajectories of agents $\mathfrak{a}$ and $\mathfrak{r}$ (enlarged by the agents' physical radii) cannot collide if the convex distance between $H^{\mathfrak{a}}$ and $H^{\mathfrak{r}}$ is at least $\Lambda^{\mathfrak{a},\mathfrak{r}}$. Formally this is the case if $\gamma^{\mathfrak{a},\mathfrak{r}}(t) := \max_{i=1,\ldots,d} |\mu_i^{\mathfrak{a}} - \mu_i^{\mathfrak{r}}| - \Lambda^{\mathfrak{a},\mathfrak{r}} - r_i^{\mathfrak{a}}(t) - r_i^{\mathfrak{r}}(t) > 0$. Therefore, $\gamma^{\mathfrak{a},\mathfrak{r}}(\cdot)$ is a suitable criterion function. It goes without saying that if additional knowledge about the distributions on the domains is available and if non-zero collision probabilities are tolerable, conservatism can be significantly reduced. This insight is the motivation behind a great many SMPC approaches that assume uncertainties with bounded domains (cf. e.g. [144]).

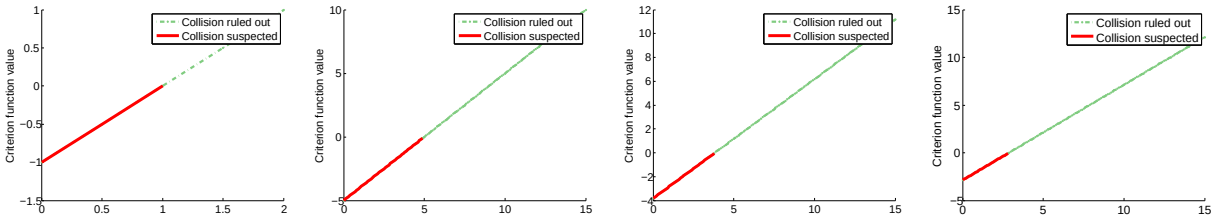


Figure 5.2.: Criterion function values (as per Eq. 5.4) as a function of $\|\langle s^{\mathfrak{r}}\rangle - \langle s^{\mathfrak{a}}\rangle\|_\infty$ and with $\delta = 0.05$, $\Lambda^{\mathfrak{a},\mathfrak{r}} = 1$. Plots from left to right: 1) variances $C^{\mathfrak{a}} = C^{\mathfrak{r}} = \text{diag}(.00001, .00001)$. 2) variances $C^{\mathfrak{a}} = C^{\mathfrak{r}} = \text{diag}(.1, .1)$. 3) variances $C^{\mathfrak{a}} = C^{\mathfrak{r}} = \text{diag}(.1, .1)$ and with improved criterion function (as per Thm. 5.2.4). 4) With criterion function tailored to Gaussian trajectories as per Eq. 5.7. As expected, the latter criterion function is the least conservative.

Multi-agent case.

So far our exposition has focussed on collision prediction of one agent $\mathfrak{a}$ with one other agent $\mathfrak{r}$. Next, we examine collision prediction of $\mathfrak{a}$ with a set $\mathfrak{A}'$ of other agents. Let $\mathfrak{a} \in \mathfrak{A}$, $\mathfrak{A}' \subset \mathfrak{A}$ such that $\mathfrak{a} \notin \mathfrak{A}'$ a subset of agents. We define the event that $\mathfrak{a}$ collides with at least one of the agents in $\mathfrak{A}$' at time t as $\mathfrak{C}^{\mathfrak{a},\mathfrak{A}'}(t) := \{(s^{\mathfrak{a}}(t), s^{\mathfrak{r}}(t)) | \exists \mathfrak{r} \in \mathfrak{A}' : \|s^{\mathfrak{a}}(t) - s^{\mathfrak{r}}(t)\|_2 \leq \Lambda^{\mathfrak{a},\mathfrak{r}}\} = \bigcup_{\mathfrak{r}\in\mathfrak{A}'} \mathfrak{C}^{\mathfrak{a},\mathfrak{r}}$. By union bound, $\Pr[\mathfrak{C}^{\mathfrak{a},\mathfrak{A}'}(t)] \leq \sum_{\mathfrak{r}\in\mathfrak{A}'} \Pr[\mathfrak{C}^{\mathfrak{a},\mathfrak{r}}(t)]$.

Theorem 5.2.5 (Multi-Agent Criterion)**.** *Let $\gamma^{\mathfrak{a},\mathfrak{r}}$ be valid criterion functions defined w.r.t. collision bound $\delta^{\mathfrak{a}}$. We define* multi-agent collision criterion function $\Gamma^{\mathfrak{a},\mathfrak{A}'}(t) := \min_{\mathfrak{r}\in\mathfrak{A}'} \gamma^{\mathfrak{a},\mathfrak{r}}(t)$. *If $\Gamma^{\mathfrak{a},\mathfrak{A}'}(t) > 0$ then the collision probability with $\mathfrak{A}$' is bounded below $\delta^{\mathfrak{a}}|\mathfrak{A}'|$. That is,* $\Pr[\mathfrak{C}^{\mathfrak{a},\mathfrak{A}'}(t)] < \delta^{\mathfrak{a}}|\mathfrak{A}'|$.

Proof. Let $\mathfrak{a} \in \mathfrak{A}$, $\mathfrak{A}' \subset \mathfrak{A}$ such that $\mathfrak{a} \notin \mathfrak{A}'$ a subset of agents. We define the event that $\mathfrak{a}$ collides with at least one of the agents in $\mathfrak{A}$' at time t as $\mathfrak{C}^{\mathfrak{a},\mathfrak{A}'}(t) := \{(s^{\mathfrak{a}}(t), s^{\mathfrak{r}}(t)) | \exists \mathfrak{r} \in \mathfrak{A}' : \|s^{\mathfrak{a}}(t) - s^{\mathfrak{r}}(t)\|_2 \leq \Lambda^{\mathfrak{a},\mathfrak{r}}\} = \bigcup_{\mathfrak{r}\in\mathfrak{A}'} \mathfrak{C}^{\mathfrak{a},\mathfrak{r}}$.

We have established that if $\forall \mathfrak{r} \in \mathfrak{A}' : \gamma^{\mathfrak{a},\mathfrak{r}}(t) > 0$ then $\Pr[\mathfrak{C}^{\mathfrak{a},\mathfrak{r}}(t)] < \delta^{\mathfrak{a}}, \forall \mathfrak{r} \in \mathfrak{A}'$. Now, let $\Gamma^{\mathfrak{a},\mathfrak{A}'} < \delta^{\mathfrak{a}}$. Hence,$\forall \mathfrak{r} \in \mathfrak{A}' : \gamma^{\mathfrak{a},\mathfrak{r}}(t) > 0$. Thus, $\forall \mathfrak{r} \in \mathfrak{A}' : \Pr[\mathfrak{C}^{\mathfrak{a},\mathfrak{r}}(t)] < \delta^{\mathfrak{a}})$ Therefore, $\sum_{\mathfrak{r}\in\mathfrak{A}'} \Pr[\mathfrak{C}^{\mathfrak{a},\mathfrak{r}}(t)] \leq |\mathfrak{A}'|\,\delta^{\mathfrak{a}}$. By union bound, $\Pr[\mathfrak{C}^{\mathfrak{a},\mathfrak{A}'}(t)] \leq \sum_{\mathfrak{r}\in\mathfrak{A}'} \Pr[\mathfrak{C}^{\mathfrak{a},\mathfrak{r}}(t)]$. Consequently, we have $\Pr[\mathfrak{C}^{\mathfrak{a},\mathfrak{A}'}(t)] \leq |\mathfrak{A}'|\,\delta^{\mathfrak{a}}$. q.e.d.

□

Moreover with Lem. 2.5.6 it is easy to establish that $\Gamma^{\mathfrak{a},\mathfrak{A}'}$ is Hölder if the constituent functions $\gamma^{\mathfrak{a},\mathfrak{r}}$ are.

Our distribution-independent collision criterion functions have the virtue that they work for all distributions – not only the omnipresent Gaussian. Unfortunately, distribution-independence is gained at the price of conservativeness (ref. to Fig. 5.2). In our experiments in Sec. 5.4, the collision criterion function as per Thm. 5.2.4 is utilized as an integral component of our collision avoidance mechanisms. The results suggest that the conservativeness of our detection module does not entail prohibitively high-false-alarm rates for the distribution-independent approach to be considered impractical. That said, whenever distributional knowledge can be converted into a criterion function. One could then use our derivations as a template to generate

refined criterion functions using Eq. 5.3 with adjusted radii r_i, r_j, reflecting the distribution at hand. We already have considered the Gaussian case; the sub-Gaussian case follows easily by employing the appropriate sub-Gaussian tail inequalities and also encompasses stochastic uncertainty with given by densities with bounded-support (cf. Sec. 2.4.7).

5.2.3. Lipschitz and Hölder properties

To render our criterion functions amenable to the algorithms of Sec. 5.2.1, we need to derive Lipschitz constants or a Hölder constant and exponent.

As an example, consider our generally applicable criterion function as per Eq. 5.5:

$$\gamma^{\mathfrak{a},\mathfrak{r}}(t) := \max_{i=1,\dots,D} |\mu_i^{\mathfrak{a}}(t) - \mu_i^{\mathfrak{r}}(t)| - \Lambda^{\mathfrak{a},\mathfrak{r}} - \sqrt{\frac{2C_{ii}^{\mathfrak{a}}(t)}{\delta}} - \sqrt{\frac{2C_{ii}^{\mathfrak{r}}(t)}{\delta}}.$$

Note, this function has the desirable property of being Lipschitz continuous, provided the mean $\mu_i^{\mathfrak{q}} : I \to \mathbb{R}$ and standard deviation functions $\sigma_{ii}^{\mathfrak{q}} = \sqrt{C_{ii}^{\mathfrak{q}}} : I \to \mathbb{R}_+$ are. In particular, on the basis of Lem. 2.5.6, it is easy to show $L(\gamma^{\mathfrak{a},\mathfrak{r}}) \leq \max_{i=1,\dots,D} L(\mu_i^{\mathfrak{a}}) + L(\mu_i^{\mathfrak{r}}) + \sqrt{\frac{2}{\delta}}\big(L(\sigma_{ii}^{\mathfrak{a}}) + L(\sigma_{ii}^{\mathfrak{r}})\big)$ where, as before, $L(f)$ denotes a Lipschitz constant of function f.

In cases where the standard deviations, as a function of time, are not Lipschitz (or, it seems hard to determine a Lipschitz constant), it may often be possible to replace the variances by an upper bound that is. For instance, the variance of an Ornstein-Uhlenbeck process $\mathrm{d}X = K(\xi - X)\,\mathrm{d}t + B\,\mathrm{d}W$ is (cf. Sec. 2.4.6).

$$C(t) = \mathrm{var}[X(t)] = \frac{B^2}{2K}\Big(1 - \exp\big(-2Kt\big)\Big) \tag{5.14}$$

on time interval $\mathcal{I}_t = [0, t_f]$.

Here, the standard deviation function is $\sigma(t) = \sqrt{\frac{B^2}{2K}}\sqrt{1 - \exp\big(-2Kt\big)}$. Owing to the square root, a Lipschitz constant that is globally valid on time interval $[0, t_f]$ cannot be given, since the derivative of the square root tends to infinity as the argument tends to zero. However, being content with a conservative approach, we might be happy to only derive a Lipschitz number for $[0 + \varepsilon, t_f)$ and base collision detection on $[0, \varepsilon]$ on an upper bound of the real variance on this time interval. Since in our example, the variance is monotonically increasing, we could pretend that $\sigma(t) = \sigma(\varepsilon), \forall t \in [0, \varepsilon]$. In the extreme case, we can do this for all times. That is, we bound $\sigma(\cdot)$ by its stationary value of $\frac{B^2}{2K}$ [101]. Being variance-conservative, this bound can be used instead of the real variance in the definition of the criterion function. While increasing conservatism of the detection process (decreasing $\gamma^{\mathfrak{a},\mathfrak{r}}(t)$), this can also have computational advantages since the Lipschitz number of a constant is zero and hence, the overall Lipschitz number of the criterion function is reduced. As we have noted above, reducing the Lipschitz number can speed up detection run-time if our

Lipschitz algorithms of Sec. 5.2.1 are employed.

As an alternative to these approaches, we could base negative value checking on Hölder properties rather than Lipschitz properties in situations where the standard deviation is not Lipschitz but Hölder continuous. For instance, while not being Lipschitz, we saw in Ex. 2.5.7 that our OU standard deviation $C(t)$ as per Eq. 5.14 is Hölder continuous with constant $L(C) = |B|$ and exponent $p_C = \frac{1}{2}$. For standard deviation trajectories like this we can employ an improved version of Alg. 1 for negative sign detection in lieu to the one we provided for Lipschitz continuity in Sec. 5.2.1.

5.3. Collision Avoidance

In this section we outline the core ideas of our proposed approach to multi-agent collision avoidance. After specifying the agent's dynamics and formalizing the notion of a single-agent plan, we define the multi-agent planning task. Then we describe how conflicts, picked-up by our collision prediction method, can be resolved. In Sec. 5.3.1 we describe the two coordination approaches we consider utilizing to generate conflict-free plans.

I) Model (example). We assume the system contains a set $\mathfrak{A}$ of agents indexed by $\mathfrak{a} \in \{1, ..., |\mathfrak{A}|\}$. Each agent $\mathfrak{a}$'s associated plant has a probabilistic state trajectory following stochastic controlled D-dimensional state dynamics in the continuous interval of (future) time $I = (t_0, t_f]$. That is, for simplicity of exposition, we assume that the state space coincides with the interaction space.

Remark 5.3.1 (Collisions in transformed state space). More generally, one might instead consider an interaction space $\mathcal{S}$ that is a linear transformation T of state space $\mathcal{X}$. For instance in a second order system it might be of interest to consider $\mathcal{S} = T\mathcal{X}$ where T is a linear projection. Since T is linear, computing the means and (co)-variances in $\mathcal{S}$ on the basis of means and covariances in $\mathcal{X}$ is simple.

We desire to ask agents to adjust their policies to avoid collisions. Each policy gives rise to a stochastic belief over the trajectory resulting from executing the policy. For our method to work, all we require is that the trajectory's mean function $m : I \to \mathbb{R}^D$ and covariance matrix function $\Sigma : I \to \mathbb{R}^{D\times D}$ are evaluable for all times $t \in I$.

A prominent class for which closed-form moments can be easily derived are linear stochastic differential equations (*SDE*s). For instance, in connection to control applications, it is of interest to consider Ornstein Uhlenbeck trajectories given by the first-order Ito-SDE

$$dx^{\mathfrak{a}}(t) = K\big(\xi^{\mathfrak{a}}(t) - x^{\mathfrak{a}}(t)\big)dt + B\,dW \tag{5.15}$$

where $K, B \in \mathbb{R}^{D\times D}$ are matrices $x^{\mathfrak{a}} : I \to \mathbb{R}^D$ is the state trajectory and W is a vector-valued Wiener pro-

cess. Here, $u(x^{\mathfrak{a}};\xi^{\mathfrak{a}}) := K(\xi^{\mathfrak{a}} - x^{\mathfrak{a}})$ could be interpreted as the control policy of a linear feedback-controller parametrised by $\xi^{\mathfrak{a}}$. It regulates the state to track a desired trajectory $\xi^{\mathfrak{a}}(t) = \zeta_0^{\mathfrak{a}}\chi_{\{0\}}(t) + \sum_{i=1}^{H^{\mathfrak{a}}} \zeta_i^{\mathfrak{a}}\chi_{\tau_i^{\mathfrak{a}}}(t)$ where $\chi_{\tau_i} : \mathbb{R} \to \{0,1\}$ denotes the indicator function of the half-open interval $\tau_i^{\mathfrak{a}} = (t_{i-1}^{\mathfrak{a}}, t_i^{\mathfrak{a}}] \subset [0, T^{\mathfrak{a}}]$ and each $\zeta_i^{\mathfrak{a}} \in \mathbb{R}^D$ is a *setpoint*. If K is positive definite the agent's state trajectory is determined by setpoint sequence $p^{\mathfrak{a}} = (t_i^a, \zeta_i^{\mathfrak{a}})_{i=0}^{H^{\mathfrak{a}}}$ (aside from the random disturbances) which we will refer to as the agent's *plan*. For example, plan $p^{\mathfrak{a}} := \big((t_0, x_0^{\mathfrak{a}}), (t_f, x_f^{\mathfrak{a}})\big)$ could be used to regulate agent $\mathfrak{a}$'s *start state* $x_0^{\mathfrak{a}}$ to a given *goal state* $x_f^{\mathfrak{a}}$ between times t_0 and t_f. For simplicity, we assume the agents are always initialized with plans of this form before coordination commences. However, this is not a necessary restriction and our method works with other initialisations

One may interpret a setpoint as some way to alter the stochastic trajectory. Below, we will determine setpoints that modify a stochastic trajectory to reduce collision probability while maintaining low expected cost. From the vantage point of policy search, $\xi^{\mathfrak{a}}$ is agent $\mathfrak{a}$'s policy parameter that has to be adjusted to avoid collisions.

Remark 5.3.2. Of course a simple alternative would be to replace K by a time-dependent matrix $K(t)$ as a policy parameter to be optimised. This policy space is implicitly searched by many model-predictive control approaches that update the gain matrix of a linear feedback-controller [143, 161]. While being simple and yielding a computationally less demanding optimisation task, the policy space that can be explored is more restrictive. In particular, it could not change the spatial location of the equilibrium point and hence, the policy space would not be expressive enough to avoid static obstacles. An alteration that is also common in MPC [223] is to instead search the policy space $\mathcal{P}_u = \{u(x,t;K,c) | u(x,t) = K(t)x + c(t), c \in \mathbb{R}^D, \forall t\}$. If K is invertible this space coincides with the set-point policy space we consider (except that MPC usually is concerned with discrete-time systems and bounded controls [161]). It would be unproblematic to employ our methods to search this policy space.

II) Task. Each agent $\mathfrak{a}$ desires to find a sequence of setpoints $(p^{\mathfrak{a}})$ such that (i) it moves from its start state $x_0^{\mathfrak{a}}$ to its goal state $x_f^{\mathfrak{a}}$ along a low-cost trajectory and (ii) such that along the trajectory its plant (with diameter Δ) does not collide with any other agents' plant in interaction space with at least a given probability $1 - \delta \in (0,1)$.

III) Collision resolution. An agent seeks to avoid collisions by adding new setpoints to its plan until the collision probability of the resulting trajectory drops below threshold δ. For choosing these new setpoints we consider two methods **WAIT** and **FREE**. In the first method the agents insert a time-setpoint pair $(t, x_0^{\mathfrak{a}})$ into the previous plan $p^{\mathfrak{a}}$. Since this aims to cause the agent to wait at its start location $x_0^{\mathfrak{a}}$ we will call the method WAIT. It is possible that multiple such insertions are necessary to make the agent wait sufficiently long to ensure all collisions are avoided. Of course, if a higher-priority agent decides to traverse through $x_0^{\mathfrak{a}}$, this method is too rigid to resolve a conflict. In the second method the agent optimizes for the time and

location of the new setpoint. Let $p^{\mathfrak{a}}_{\uparrow(t,s)}$ be the plan updated by insertion of time-setpoint pair $(t, s) \in I \times \mathbb{R}^D$. We propose to choose the candidate setpoint (t, s) that minimizes a function being a weighted sum of the expected cost entailed by executing updated plan $p^{\mathfrak{a}}_{\uparrow(t,s)}$ and a hinge-loss collision penalty $c^{\mathfrak{a}}_{coll}(p^{\mathfrak{a}}_{\uparrow(t,s)}) := \lambda \max\{0, -\min_t \Gamma^{\mathfrak{a}}(t)\}$. Here, $\Gamma^{\mathfrak{a}}$ is computed based on the assumption we were to execute $p^{\mathfrak{a}}_{\uparrow(t,s)}$ and $\lambda >> 0$ determines the extent to which collisions are penalized. Since the new setpoint can be chosen freely in time and state-space we refer to the method as FREE.

5.3.1. Coordination

We will now consider how to integrate our collision detection and avoidance methods into a coordination framework that determines who needs to avoid whom and at what stage of the coordination process. Such decisions are known to significantly impact the *social cost* (i.e. the sum of all agents' individual costs) of the agent collective.

Fixed-priorities (FP). As a baseline method for coordination we consider a basic fixed-priority method (e.g. [26, 94]). Here, each agent has a unique ranking (or priority) according to its index $\mathfrak{a}$ (i.e. agent 1 has highest priority, agent $|\mathfrak{A}|$ lowest). When all higher-ranking agents are done planning, agent $\mathfrak{a}$ is informed of their planned trajectories which it has to avoid with a probability greater than $1 - \delta$. This can be done by repeatedly invoking for collision detection and resolution methods described above until no further collision with higher-ranking agents are found.

Lazy Auction Protocol (AUC). While the FP method is simple and fast the rigidity of the fixed ranking can lead to sub-optimal social cost and coordination success. Furthermore, its sequential nature does not take advantage of possible parallelization a distributed method could. To alleviate this we propose to revert the ranking flexibly on a case-by-case basis. In particular, the agents are allowed to compete for the right to gain passage (e.g. across a region where a collision was detected) by submitting bids in the course of an auction. The structure of the approach is outlined in Alg. 3.

Assume an agent $\mathfrak{a}$ detects a collision at a particular time step t_{coll} and invites the set of agents $\mathcal{C}^{\mathfrak{a}} = \{\mathfrak{r} \in \mathfrak{A} | \gamma^{\mathfrak{a},\mathfrak{r}}(t_{\text{coll}}) \leq 0\}$ to join an auction to decide who needs to avoid whom. In particular, the auction determines a winner who is not required to alter his plan. The losing agents need to insert a new setpoint into their respective plans designed to avoid all other agents in $\mathcal{C}^{\mathfrak{a}}$ while keeping the plan cost function low.

The idea is to design the auction rules as a heuristic method to minimize the social cost of the ensuing solution. To this end, we define the bids such that their magnitude is proportional to a heuristic magnitude of the expected regret for losing and not gaining passage. That is agent $\mathfrak{a}$ submits a bid $b^{\mathfrak{a}} = \mathfrak{l}^{\mathfrak{a}} - \mathfrak{s}^{\mathfrak{a}}$. Magnitude $\mathfrak{l}^{\mathfrak{a}}$ is defined as $\mathfrak{a}$'s anticipated cost $c^{\mathfrak{a}}_{plan}(p^{\mathfrak{a}}_{\uparrow(t,s)})$ for the event that the agent will not secure "the right of

input : Agents $\mathfrak{a} \in \mathfrak{A}$, cost functions $c^{\mathfrak{a}}$, dynamics, initial start and goal states, initial plans $p^1, ..., p^{|\mathfrak{A}|}$.
output: collision-free plans $p^1, ..., p^{|\mathfrak{A}|}$.

repeat
  for $\mathfrak{a} \in \mathfrak{A}$ **do**
    $[\ \mathsf{flag}^{\mathfrak{a}}, \mathcal{C}^{\mathfrak{a}}, t_{\mathsf{coll}}] \leftarrow \texttt{CollDetect}^{\mathfrak{a}}(\mathfrak{a}, \mathfrak{A} - \{\mathfrak{a}\})$
    if $\mathsf{flag}^{\mathfrak{a}} = 1$ **then**
      winner $\leftarrow \texttt{Auction}(\mathcal{C}^{\mathfrak{a}} \cup \{\mathfrak{a}\}, t_{\mathsf{coll}})$
      foreach $\mathfrak{r} \in (\mathcal{C}^{\mathfrak{a}} \cup \{\mathfrak{a}\}) - \{\mathsf{winner}\}$ **do**
        $p^{\mathfrak{r}} \leftarrow \texttt{Avoid}^{\mathfrak{r}}((\mathcal{C}^{\mathfrak{a}} \cup \{\mathfrak{a}\}) - \{\mathfrak{r}\}, t_{\mathsf{coll}})$
        $\texttt{Broadcast}^{\mathfrak{r}}(p^{\mathfrak{r}})$
      end
    end
  end
until $\forall \mathfrak{a} \in \mathfrak{A} : \mathsf{flag}^{\mathfrak{a}} = 0$;

Algorithm 3: Lazy auction coordination method (AUC) (written in a sequentialized form). Collisions are resolved by choosing new setpoints to enforce collision avoidance. $\mathcal{C}^{\mathfrak{a}}$: set of agents detected to be in conflict with agent $\mathfrak{a}$. $\mathsf{flag}^{\mathfrak{a}}$: collision detection flag (=0, iff no collision detected). t_{coll}: earliest time where a collision was detected. Avoid: collision resolution method updating the plan by a single new setpoint according to WAIT or FREE.

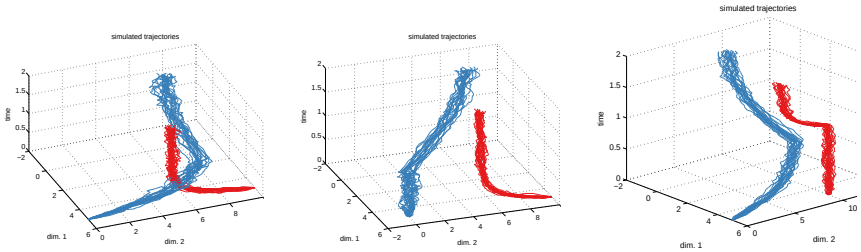


Figure 5.3.: EXP1. Draws from uncoordinated agents' plans (left), after coordination and collision resolution with methods FP-WAIT (centre) and AUC-WAIT (right).

passage" and has to create a new setpoint (t, s) (according to (III)) tailored to avoid all other agents engaged in the current auction. On the other hand, $\mathfrak{s}^{\mathfrak{a}} := c^{\mathfrak{a}}_{plan}(p^{\mathfrak{a}})$ is the cost of the unchanged plan $p^{\mathfrak{a}}$. If there is a tie among multiple agents the agent with the lowest index among the highest bidders wins.

Acknowledging that $\mathfrak{s}^{winner} + \sum_{\mathfrak{a} \neq winner} \mathfrak{l}^{\mathfrak{a}}$ is an estimated social cost (based on current beliefs of trajectories) after the auction, we see that the winner determination rule greedily attempts to minimize social cost:$\forall \mathfrak{r} \in \mathfrak{A} \backslash \{winner\} : b^{winner} \geq b^{\mathfrak{r}} \Leftrightarrow \mathfrak{s}^{\mathfrak{r}} + \sum_{\mathfrak{a} \neq \mathfrak{r}} \mathfrak{l}^{\mathfrak{a}} \geq \mathfrak{s}^{winner} + \sum_{\mathfrak{a} \neq winner} \mathfrak{l}^{\mathfrak{a}}$.

This bidding rule is an adaptation of the bidding rule of the discrete-time version of our auction approach described in Ch. 6.

5.4. Simulations

As a first test, we simulated three simple multi-agent scenarios, *EXP1*, *EXP2* and *EXP3*. Each agent's dynamics were an instantiation of an SDE of the form of Eq. 5.15. We set δ to achieve collision avoidance with certainty greater than 95%. Collision prediction was based on the improved criterion function as per Thm.

	Experiment 1			Experiment 2		
Quantity	NONE	AUC-WAIT	FP-WAIT	NONE	AUC-FREE	FP-FREE
A	78	0	0	51	0	0
B	13.15	13.57	12.57	14.94	16.22	18.13
C	0.05	0.04	25.8	0.05	0.05	0.05
D	0	6	3	0	4	4

Table 5.1.: Quantities estimated based on 100 draws from SDEs simulating executions of the different plans in EXP1 and EXP2. **A**: estimated collision probability [%]; **B**: averaged path length away from goal; **C**: averaged sqr. dist. of final state to goal; **D**: number of collision resolution rounds. Notice, our collision avoidance methods succeed in preventing collisions. In EXP1 the FP-WAIT method failed to reach its first goal in time which is reflected in the *sqr. distance to goal* measure.

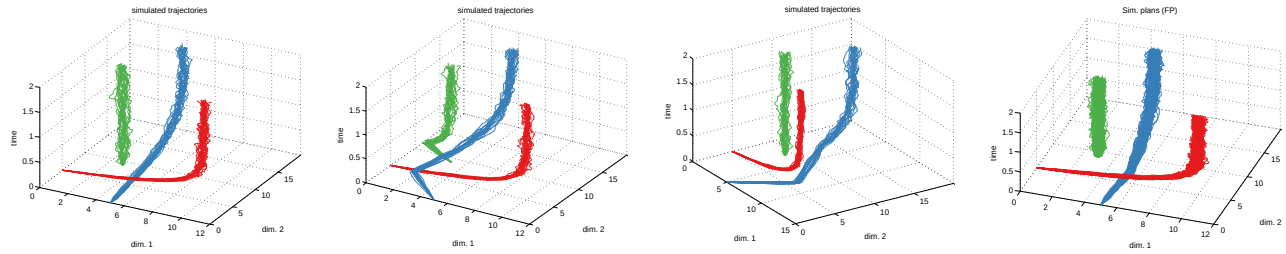


Figure 5.4.: EXP2. From left to right: Draws from uncoordinated agents' plans (1st image), after coordination and collision resolution with methods FP-FREE (2nd image) and AUC-FREE (3rd image). Collision detection was based on Chebyshev-type criterion function. Finally, we repeated the experiment with Gaussian collision criterion function $\gamma^{\mathfrak{a},\mathfrak{r}}_{\mathcal{N}}$ of Eq. 5.7.

5.2.4. During collision resolution with the FREE method each agent $\mathfrak{a}$ assessed a candidate plan $p^{\mathfrak{a}}$ according to cost function $c^{\mathfrak{a}}_{plan}(p^{\mathfrak{a}}) = w_1\, c^{\mathfrak{a}}_{traj}(p^{\mathfrak{a}}) + w_2\, c^{\mathfrak{a}}_{miss}(p^{\mathfrak{a}}) + w_3\, c^{\mathfrak{a}}_{coll}(p^{\mathfrak{a}})$. Here $c^{\mathfrak{a}}_{traj}$ is a heuristic to penalize expected control energy or path length; in the second summand, $c^{\mathfrak{a}}_{miss}(p^{\mathfrak{a}}) = \left\| x^{\mathfrak{a}}(t_f) - x^{\mathfrak{a}}_f \right\|^2$ penalizes deviation of the mean from the goal state; the third term $c^{\mathfrak{a}}_{coll}(p^{\mathfrak{a}})$ penalizes collisions (cf. III). The weights are design parameters which we set to $w_1 = 10, w_2 = 10^3$ and $w_3 = 10^6$, emphasizing avoidance of mission failure and collisions. Note, if our method was to be deployed in a receding horizon fashion, the parameters could also be adapted online using standard learning techniques such as no-regret algorithms [155, 235].

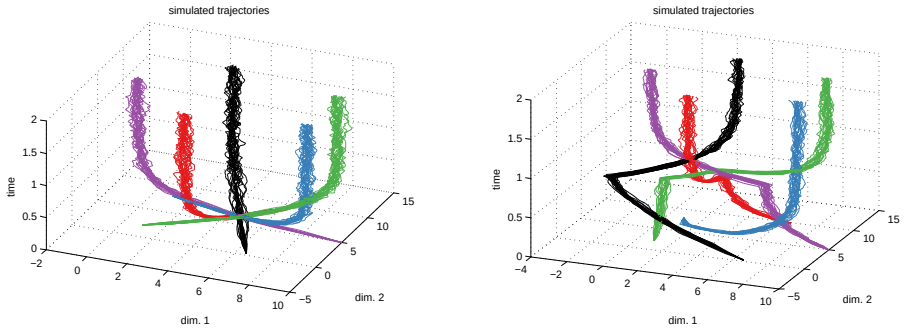


Figure 5.5.: Ex. of EXP3 with 5 agents. Draws from uncoordinated agents' plans (left), after coordination and collision resolution with methods AUC-FREE (right).

EXP1. Collision resolution was done with the WAIT method to update plans. Draws from the SDEs with

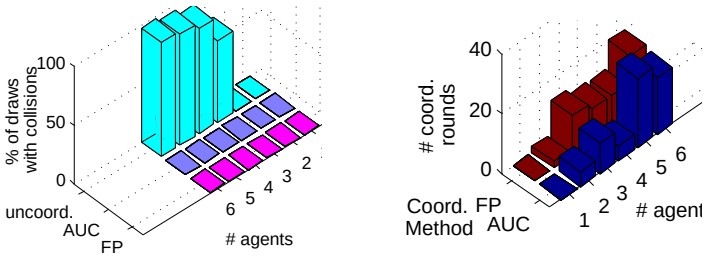


Figure 5.6.: Recorded results for EXP3 with 1 to 6 agents. Note, all collisions were successfully avoided.

the initial plans of the agents are depicted in Fig. 5.3 (left). The curves represent 20 noisy trajectories of agents 1 (red) and 2 (blue). Each curve is a draw from the stochastic differential dynamics obtained by simulating the execution of the given initial plan. The trajectories were simulated with the Euler-Maruyama method for a time interval of $I = [0s, 2s]$. The spread of the families of curves is due to the random disturbances each agent's controller had to compensate for during runtime.

Agent 1 desired to control the state from start state $x_0^1 = (5, 10)$ to goal $x_f^1 = (5, 5)$. Agent 2 desired to move from start state $x_0^2 = (5, 0)$ via intermediate goal $x_{f_1}^2 = (5, 7)$ (at 1s) to final goal state $x_{f_2}^2 = (0, 7)$. While the agents meet their goals under the initial plans, their execution would imply a high probability of colliding around state $(5, 6)$ (cf. Fig. 5.3 (left), Tab. 5.1). Coordination with fixed priorities (1 (red) $>$ 2 (blue)) yields conflict-free plans (Fig. 5.3 (centre)). However, agent 2 is forced to wait too long at its start location to be able to reach intermediate waypoint $x_{f,1}^2$ in time and therefore, decides to move directly to its second goal. This could spawn high social cost due to missing one of the designated goals (Tab. 5.1). By contrast, the auction method is flexible enough to reverse the ranking at the detected collision point causing agent 1 to wait instead of 2 (Fig. 5.3 (right)). Thereby, agent 2 is able to reach both of its goal states in time. This success is reflected by low social cost (see Tab. 5.1).

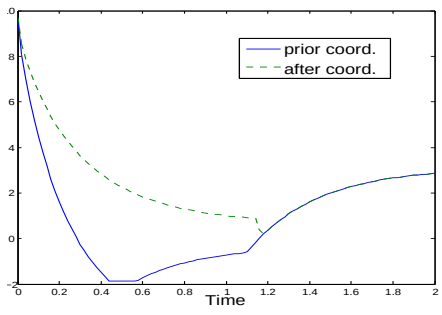


Figure 5.7.: EXP1. Criterion functions for collision detection of agent 2 before and after coordination. The first graph accurately warns of a collision before coordination (as indicated by negative values), whereas the one corresponding to collision-free trajectories is in the positive-half space as desired.

EXP2. The setup was analogous to EXP1 but with three agents and different start and goal states as depicted in Fig. 5.4. Furthermore, collisions were avoided with the FREE method with 10 random initializations

of the local optimizer. Coordination of plans with fixed priorities (1 (red) $>$ 2 (blue) $>$ 3 (green)) caused 2 to avoid agent 1 by moving to the left. Consequently, 3 now had to temporarily leave its start and goal state to get out of the way (see Fig. 5.4, 2nd from the left). With two agents moving to avoid collisions social cost was relatively high (see Tab. 5.1). During coordination with the auction-based method agent 2 first chose to avoid agent 1 (as in the FP method). However, losing the auction to agent 3 at a later stage of coordination, agent 2 decided to finally circumvent 1 by arcing to the right instead of to the left. This allowed 3 to stay in place (see Fig. 5.4, 3rd from the left).

EXP3. Next, we conducted a sequence of experiments for varying numbers of agents ranging from $|\mathfrak{A}| = 1, .., 7$. In each experiment all agents' start locations were placed on a circle. Their respective goals were placed on the opposite ends of the circle. The eigenvalues of the feedback gain matrices of each agent were drawn at random from a uniform distribution on the range [2,7]. An example situation for an experiment with 5 agents is depicted in Fig. 5.5. Collision avoidance was achieved.

Note, that despite this setting being close to worst case (i.e. almost all agents try to traverse a common, narrow corridor) the coordination overhead is moderate (see Fig. 5.6, right). Also, all collisions were successfully avoided (see Fig. 5.6, left).

5.5. Conclusions

This work considered multi-agent planning under stochastic (as well as under interval-bounded) uncertainty and non-convex chance-constraints for collision avoidance. In contrast to pre-existing work, we did not need to rely on prior space or time-discretisation. This was achieved by deriving criterion functions with the property that the collision probability is guaranteed to be below a freely definable threshold $\delta \in (0, 1)$ if the criterion function attains no negative values. Thereby, collision detection was reduced to deciding whether such negative values exist. Based on Hölder continuity of the criterion functions, we have provided a simple algorithm that can conservatively make the decision whether such values exist based on a finite number of function evaluations. For Lipschitz continuous criterion functions, we provided an algorithm for making this decision more rapidly and with reduced conservatism.

We described a general procedure for deriving criterion functions. Two such functions based on Chebyshev-type bounds were derived. Since these bounds are distribution-independent, this approach is universally applicable. As we have shown by deriving criterion functions valid for the Gaussian case, folding in distributional knowledge can lead to reduced conservatism and can be done within our framework as well. Here conservatism means that the algorithm is guaranteed to never certify absence of a criterion function negative value when such a value can exist. This is an important property that allows as to guarantee that our coordination approach never overlooks collisions that are undesirably probable. On the flip side, the larger degree of con-

servatism might result in overestimation of collision probabilities and consequently, induce unnecessary plan modifications in the course of coordination. Where possible, the degree of conservatism should be reduced without ever giving it up entirely. That is, without ever underestimating the risk of collisions.

To enforce collision avoidance, our method modified the agent's plans until no collisions could be detected. To coordinate the detection and avoidance efforts of the agents, we employed an auction-based as well as a fixed-priority method.

Our experiments are a first indication that our approach can succeed in finding collision-free plans with high-certainty with the number of required coordination rounds scaling mildly in the number of agents. While in its present form, our coordination mechanisms do not come with a termination guarantee, in none of our simulations have we encountered an infinite loop. A modified version of the auction method will also be examined in greater detail in Ch. 6. Future work might consider whether our approach can be combined with a simple stopping criterion that terminates the coordination attempt when a computational budget is expended or an infinite loop is detected.

The computation time within each coordination round depends heavily on the time required for finding a new setpoint and for collision detection. This involves minimizing $(t, s) \mapsto c^{\mathfrak{a}}_{plan}(p^{\mathfrak{a}}_{\uparrow(t,s)})$ and $c^{\mathfrak{a}}_{coll}$, respectively. The worst-case complexity depends on the choice of cost functions, their domains and the chosen optimizer. Fortunately, we can draw on a plethora of highly advanced global optimisation methods (e.g. [124, 230]) guaranteeing rapid optimization success. In terms of execution time, we can expect considerable alleviations from implementation in a compiled language. Furthermore, the collision detection and avoidance methods are based on global optimization and thus, would be highly amenable to parallel processing – this could especially benefit the auction approach.

While our exposition was focussed on the task of defining setpoints of feedback-controlled agents, the developed methods can be readily applied to other policy search settings, where the first two moments of the probabilistic beliefs over the trajectories (that would result from applying the found policies) can be computed.

6. Multi-agent optimisation for coordination and collision avoidance in discrete-time systems

"The only thing that will redeem mankind is cooperation."

Bertrand Russel (1872-1970)

We consider multi-agent planning problems that can be modelled by constrained optimisation where the agent-interactions are modelled via the constraints and the objective function quantifies social cost. Due to their expressiveness on the one hand, and the existence of highly potent optimisers on the other [119], our focus will be on problems that can be solved with mixed-integer programming (*MIP*). Being able to solve NP-hard problems as well as expressing logic constraints [222], this framework is very general and encompasses a large number of varied multi-agent planning problems that often are considered in complete separation in different fields, including artificial intelligence, control and operations research. While one could solve the coordination problem by combining the individual agents' problems into one large joint centralized MIP (e.g. [222]), this approach becomes rapidly intractable for growing numbers of agents and large problem domains. And, from a distributed artificial intelligence perspective, it is desirable to decentralize computation and reduce the necessity of global communication.

To address these issues, we propose three different generic methods based on a combination of conflict detection and constraint-generation. Planning is done repeatedly until all conflicts are resolved. During the repeated planning trials the agents communicate with agents they share constraints with (or with the environment they share with other agents) with the aim to gradually uncover a set of constraints that is just as large as necessary to achieve coordination. Two of our methods are complete and optimal. That is, they guarantee to find an optimal solution that satisfies all inter-action constraints, provided such a solution exists. Unfortunately, this highly desirable worst-case guarantee is achieved by merging those optimisation problems that cannot be solved independently. Therefore, in the worst case, these methods end up solving the completely centralised optimisation problem and hence, cannot be considered to be fully distributed.

As an alternative method, we also propose an auction-flavoured approach. When conflicts (that is inter-agent constraint violations) are detected, some agents add local constraints to their local constraint sets to resolve the conflict while keeping their individual planning problems separate otherwise. The local con-

straints, which we refer to as *virtual obstacles*, only pertain to the actions of the corresponding agent and hence, add no coupling of the agents' problems. In our auction approach, the question who needs to add constraints to add such a virtual obstacle (and hence reduce its feasible set and potentially increase his cost), and who does not, is determined by an auction with bids designed to locally optimise social welfare. In all stages of the interaction, each agent is responsible for optimising its local optimisation problem independently. Furthermore, communication generally is low-bandwidth and only required between agents that are affecting the same constraints, which could be managed via proxy agents.

Therefore, while not offering a guarantee of optimality, the auction method offers the benefits of a more distributed method. Furthermore, as we demonstrate in a sequence of simulations in a graph planning context, it can drastically reduce overall computation while finding solutions of lower overall social cost than fixed priority methods [26, 27].

After defining the coordination methods in full generality in Sec. 6.1, we illustrate their performance in the context of graph routing with indivisible flows. We then explain how our the coordination methods can also be employed in the context of multi-agent stochastic MPC with probabilistic collision avoidance constraints. First, we give an example how the auction method can be combined with single-agent SMPC methods where collision avoidance is enforced by particle-based chance-constraints [37, 38].

Connecting to the theoretical results for collision detection presented in Ch. 5, we then move on to derive our own MIP representation of multi-agent stochastic model-predictive control (MA-SMPC) with probabilistic collision avoidance constraints based on probabilistic tail inequalities. This alternative approach offers significant computational improvements and suggests applicability to receding horizon control where the optimiser has to be evoked at higher frequencies. We present simulations of multi-robot SMPC with stochastic collision avoidance constraints suggesting the viability of our methods in this application domain.

The chapter concludes by combining the multi-agent predictive control approach with our kinky inference learning approach of Ch. 4 for coordinated collision avoidance in the presence of an uncertain drift field. Assuming the kinky inference learner entertains partial knowledge about the uncertain drift, the constraints we need to impose to enforce collision avoidance now fold in the prediction error bounds afforded by the kinky inference rule. Consistent with our theoretical guarantees of Ch. 4, our simulations exemplify the fact that the constraints relax when additional learning experience becomes available.

6.1. Multi-agent planning as optimisation with interaction constraints

In this section, we will state the abstract model assumptions and coordination problem as well as our proposed solutions in full generality. To keep the exposition concrete enough to be followed, the abstract definitions will be illustrated by reference to the graph routing scenario we will later employ to test the methods in. This

multi-agent graph planning problem is defined as follows:

Definition 6.1.1 (MATPP)**.** *A group of agents $\mathfrak{A} = \{1, ..., A\}$ desires to find individual plans $p^1, ..., p^A$, respectively, that translate to collision-free trajectories in a graph such that each agent $\mathfrak{a}$'s trajectory leads from its start $S^{\mathfrak{a}}$ to its destination $D^{\mathfrak{a}}$ and the topology of the graph is respected. That is, agents can move from one node to another in one time step if and only if the nodes are connected by an edge.*

With this simple scenario in mind, we will proceed to the more abstract definitions and problem statement.

6.1.1. Definitions and objective

Consider a multi-agent system with a set of A agents indexed by $\mathfrak{A} = \{1, ..., A\}$. Each agent $\mathfrak{a} \in \mathfrak{A}$ desires to compute a plan $p^{\mathfrak{a}} \in F^{\mathfrak{a}}$ where $F^{\mathfrak{a}}$ is a local feasible set encoding what a viable plan is.

The interpretation of a plan depends on the application at hand. For instance, a plan could be a control policy or a set of resources to be consumed. In the graph planning scenario, $p^{\mathfrak{a}}$ *is* a time-indexed sequence $(p_t^{\mathfrak{a}})_{t\in\mathbb{N}}$ where $p_t^{\mathfrak{a}}$ that corresponds to a decision specifying which node to visit at time t. Local feasible set $F^{\mathfrak{a}}$ could then be specified by the set of plans that satisfy constraints on plans having to connect start and goal node and to be consistent with the topology of the graph. That is, we might impose constraints ensuring that two consecutive nodes should be linked by an edge in the graph.

Furthermore, we assume that an agent has preferences over different plans. The preferences are assumed to be modelled by a cost function $c^{\mathfrak{a}} : F^{\mathfrak{a}} \to \mathbb{R}$ such that a perfectly rational agent would ideally plan by choosing

$$p^{\mathfrak{a}} \in \arg\min_{p^{\mathfrak{a}} \in F^{\mathfrak{a}}} c^{\mathfrak{a}}(p^{\mathfrak{a}}). \tag{6.1}$$

In other words, planning is done by solving an optimisation problem (OP).

Of course, in multi-agent systems, agents interact. Therefore, the feasibility or the cost of one agent's feasible actions might depend on the actions of another agent. This renders the planning problems inter-dependent.

Commonly, these dependencies are modelled either as inter-dependence of costs (soft-constraints) as commonly done in the DCOP literature, or as inter-dependence of the feasible sets (hard-constraints). Note, a soft-constraint is a penalty cost added to an agent's objective function used to discourage violation of hard constraints by assigning high costs to such plans that violate the constraints. This was an approach we have undertaken in Ch. 5 and which is at the heart of DCOP methods such as maxsum [96] or [47]. In the limit of infinite penalty costs, the soft-constraints are equivalent to hard-constraints, a fact that is exhibited by Lagrangian dualisation of constrained optimisation (see e.g. [43]).

In this chapter, we will assume all inter-agent problems are coupled via *hard*-constraints which are typically called either *coupling*, *interaction* or *inter-agent* constraints. In the graph planning application, we could have a collision constraint for each node, stating that no more than one agent is allowed to occupy that node at any given point in time. Assume there are $m^{\mathfrak{a}}$ such coupling constraints of the form $C_1^{\mathfrak{a}}(p^{\mathfrak{a}}, p^{\neg\mathfrak{a}}) \leq 0 \ldots, C_{m^{\mathfrak{a}}}^{\mathfrak{a}}(p^{\mathfrak{a}}, p^{\neg\mathfrak{a}}) \leq 0$ for each agent $\mathfrak{a} \in \mathfrak{A}$ and let $\mathcal{C}^{\mathfrak{a}}$ denote this constraint set. Moreover, let $R_j^{\mathfrak{a}}(p^{\neg\mathfrak{a}}) = \{p^{\mathfrak{a}} | C_j^{\mathfrak{a}}(p^{\mathfrak{a}}, p^{\neg\mathfrak{a}}) \leq 0\}$ be the set of all plans $p^{\mathfrak{a}}$ of agent $\mathfrak{a}$ that satisfy the jth coupling constraint, given that all other agents' plans are fixed to $p^{\neg\mathfrak{a}}$.

From agent $\mathfrak{a}$'s perspective this means that if the plans $p^{\neg\mathfrak{a}} = \left(p^{\mathfrak{r}}\right)_{\mathfrak{r}\in\mathfrak{A}-\{\mathfrak{a}\}}$ of all the other agents $\mathfrak{r} \in \neg\mathfrak{a} := \mathfrak{A} - \{\mathfrak{a}\}$ were fixed, its individual optimisation problem to be solved in reaction to the other agents would have to be the more restricted problem

$$p^{\mathfrak{a}} \leftarrow \min_{p^{\mathfrak{a}} \in F^{\mathfrak{a}} \cap R^{\mathfrak{a}}(p^{\neg\mathfrak{a}})} c^{\mathfrak{a}}(p^{\mathfrak{a}}) \tag{6.2}$$

where $R(p^{\neg\mathfrak{a}}) = \cap_{j=1}^{m} R_j^{\mathfrak{a}}(p^{\neg\mathfrak{a}})$ is the set of all *conflict-free* plans $p^{\mathfrak{a}}$. By this we refer to plans $\mathfrak{a}$ could execute without violating any of the coupling constraints.

Note, a constraint $C_j(p) \leq 0$ may only be influenced by the plans of a subset $\mathfrak{S}_j$ of agents. That is, $\exists \mathfrak{S}_j \subsetneq \mathfrak{A} \forall p^{\mathfrak{S}_j} \forall p', p'' \in \times_{\mathfrak{a}\in\mathfrak{A}-\mathfrak{S}_j} F^{\mathfrak{a}} : C_j(p^{\mathfrak{S}_j}, p') = C_j(p^{\mathfrak{S}_j}, p'')$. We say that the plans $p^{\mathfrak{a}}, p^{\mathfrak{r}}$ of agents $\mathfrak{a}$ and $\mathfrak{r}$ are *conflicting* if a constraint they both influence is violated, i.e. if $\exists j : \mathfrak{a}, \mathfrak{r} \in \mathfrak{S}_j \wedge C_j(p) > 0$. If two agents $\mathfrak{a}$ and $\mathfrak{r}$ have conflicting plans, we refer to the agents as *being in conflict*.

Since the agents' plans are inter-dependent, coordination is required to avoid conflicts. The outcome of the coordination process is called a *joint*, *global* or *overall* plan $p := p^{\mathfrak{A}} := (p^{\mathfrak{a}})_{\mathfrak{a}\in\mathfrak{A}} = (p^1; \ldots; p^A)$ and is the collection of individual plans. We define $F_L := F^1 \times \ldots \times F^A$ to be the set of all locally feasible overall plans.

It will be convenient to merge all local coupling constraints into one large set $\mathcal{C} = \bigcup_{\mathfrak{a}\in\mathfrak{A}} \mathcal{C}^{\mathfrak{a}} = \{C_1(\cdot) \leq 0, ..., C_m(\cdot) \leq 0\}$ consisting of $m = \sum_{\mathfrak{a}\in\mathfrak{A}} m^{\mathfrak{a}}$ constraints, each of which corresponds exactly to one local interaction constraint of some agent. That is, $\forall j \in \{1, ..., m\} \exists \mathfrak{a} \in \mathfrak{A} \exists i \in \{1, ..., m^{\mathfrak{a}}\} \forall p : C_j(p) \leq 0 \Leftrightarrow C_i^{\mathfrak{a}}(p) \leq 0$. The set of all joint plans that satisfy all coupling constraints will be denoted by $F_{\mathcal{C}}$. We have $F_{\mathcal{C}} = \{p | \forall j = 1, ..., m : C_j(p) \leq 0\}$.

Coordination will attempt to ensure the overall plan is jointly feasible. That is, each constituent plan should be locally feasible and all interaction constraints need to be satisfied: $p \in F_L \cap F_C$. A coordination method that guarantees to generate a jointly feasible overall plan (if such exists) is sometimes referred to as being *complete*.

Beyond feasibility, a joint plan's quality is assessed by a *social choice* function $s(p)$ which can vary depending on the coordination method designer's aims [75]. For instance, if one was interested in minimizing

the maximum cost of any individual, the goal would be to find a joint plan p such that $s(p)$ was small with $s(p) = \max_{\mathfrak{a}\in\mathfrak{A}} c(p^{\mathfrak{a}})$. In this chapter, we will be most interested in minimizing *social cost* defined as the sum of the individual costs: $s(p) = \sum_{\mathfrak{a}\in\mathfrak{A}} c^{\mathfrak{a}}(p^{\mathfrak{a}})$. A coordination method that can guarantee to yield a jointly feasible plan with minimal social cost is called *optimal*.

6.1.2. Coordination methods

We will proceed with discussing coordination methods. The first, *OPT*, is a standard centralised approach. It has the advantage of being complete and optimal. On the flip side, it has the disadvantage of incorporating all constraints into the description of the feasible set. While this reduces the size of the search space it usually also increases its search complexity. For instance in integer programming, the computational effort grows poorly with the number of constraints. This is because the integer programming solver might have to search an optimal sub-solution for each combination of assignments of the variables that satisfy the constraint. Especially in situations where the inter-actions are sparse, this is wasteful. Here, only a small number of interaction constraints will be required to enforce overall feasibility of the solution. In that case, it is wasteful to incorporate all constraints into a centralised optimisation problem.

The following two methods we propose, *CCG and SDCG*, both capitalize on this intuition. They aim to keep the number of constraints low by delaying their addition to the description of the feasible set until they become violated. In addition $SDCG$ also keeps the dimensionality of the optimisation problems down by only merging two problems if they are connected by an active constraint. Both CCG and SDCG are complete and optimal.

The fourth method, *AUC*, is a heuristic approach based on a bidding rule that was designed to implement some for of local distributed gradient descent. While it is not optimal we can show its completeness under certain assumptions. And, we have not come across infeasibility issues in the course of our experiments discussed below.

Optimal centralised coordination (OPT)

The *socially optimal* solution could be obtained in a straight-forward manner as the solution of the *centralized* optimization problem :

$$p_{\text{opt}} \leftarrow \min_{p\in F_L\cap F_C} \sum_{\mathfrak{a}\in\mathfrak{A}} c^{\mathfrak{a}}(p^{\mathfrak{a}}). \tag{6.3}$$

Of course, any algorithm that can solve this optimisation problem in finite time provides a complete and optimal planning method. For instance, the optimal solution can be found by exhaustively searching over all integer variable assignments each of which gives rise to a linear programme. The latter can be solved

optimally with the simplex method or an interior point method.

Unfortunately, such centralized approaches often scale poorly in the number of agents. For instance, if the optimisation problem can be stated as a mixed-integer programme, the problem is NP-hard the worst-case. That said, modern MIP solvers have been applied to such centralized problems and reported their optimal solutions with hundreds of agents within seconds [273, 274].

Nonetheless, centralization of the planning process constitutes a computational and communication choke-point as well as a single points of failure [75]. And, we can expect that iterative decomposition approaches developed to work with modern MIP solvers for each sub-problem will greatly enhance scalability, both in the number of agents and constraints.

Therefore, we will seek to replace the centralized OP with iterative approaches. The common idea is that they start out with as few coupling constraints as possible and incrementally constrain their feasible sets as necessary until all conflicts are avoided. All these methods will have to compromise between the degree of desired distributivity, complexity and guarantees of completeness and optimality. We will start out with two methods, *CCG* and *SDCG*, which we proposed in [54] in the context of mixed-integer linear programming. They guarantee to be both complete and optimal, but which might (in the worst-case) end up having to solve the centralised optimisation problem and thus, can invoke even more computation than the centralised approach in the worst case. After briefly describing the well-known class of fixed-priority (FP) methods, we will then describe our lazy auction approach (AUC). Note, the latter is a discrete-optimisation variant of the one described in Ch. 5 which we, in a more simplified version, first presented in [48].

Optimal and complete constraint generation methods – CCG and SDCG approaches

The first method we propose, *Centralised Constraint Generation (CCG)*, is a centralised approach. It aims to reduce the number of interaction constraints of a centralized OP to generate the optimal joint solution. On top of that, the second method, *Semi-Decentralised Constraint Generation (SDCG)*, also attempts to reduce the dimensionality of the domain by uncovering a decomposition into cliques of independent sub-problems that can be solved locally by groups of agents. For both methods, we assume all agent interactions are modelled as constraints. As before, we call solutions violating these constraints to be in *conflict*. For instance, in the graph routing scenario considered below, such constraints will prohibit collisions – i.e. no node is allowed to be simultaneously visited by two agents.

CCG: The approach aims to solve the centralized OP for the joint solution with a small number of interaction constraints. In an iterative process, CCG solves a sequence of centralized OPs. Each OP jointly optimises the agents' solutions all at once in (high-dimensional) joint planning space. That is, in iteration i, CCG solves an OP of the form

$$p_{ccg}(i) \leftarrow \min_{p \in F_L \cap F_{\mathcal{C}(i)}} \sum_{\mathfrak{a} \in \mathfrak{A}} c^{\mathfrak{a}}(p^{\mathfrak{a}}). \tag{6.4}$$

where $p_{ccg}(i)$ denotes the joint outcome after the iteration, $F_{\mathcal{C}(i)} = \{p | C_j(p) \leq 0, \forall j \in I(i)\}$ and $I(i) \subset \{1, \ldots, m\}$ is the set of indices of coupling constraints to be included in the optimisation in iteration i.

The initial OP is started with a set $\mathcal{C}(0)$ of initial interaction constraints. Typically this set is empty so that no interactions are considered at first. After a joint solution is generated it is examined for conflicts. Any interaction constraint that is violated is then added to the definition of the OP before re-planning begins. That is, if after iteration i, conflict detection determines a set $I_{\text{confl}}(i)$ of constraint violations: $\forall j \in I_{\text{confl}}(i) : C_j\big(p(i)\big) > 0$. If conflicts are detected and $I_{\text{confl}}(i) \neq \emptyset$, these constraints are added to the set of coupling constraints to be taken into consideration during the next iteration. That is,

$$I(i+1) := I(i) \cup I_{\text{confl}}(i). \tag{6.5}$$

The whole process is repeated until a conflict-free joint solution is found.

SDCG: At the outset, each agent solves its own (low-dimensional) OP independently, without any interaction constraints, as per (6.1). The agents then exchange their solutions and detect *conflicts*. Agents that detect to be in conflict with each other merge into clusters of agents. Within each cluster, they plan centrally by combining their OPs into one larger joint OP that is solved with our CCG method. That is, each of these cluster OPs contains the constraints of the individual OPs. In addition it is initialised with those interaction constraints that depend on the agents in the cluster and that were violated before and prohibit the previously detected conflicts among the agents within the cluster. The CCG method then is applied to this sub-problem and may incorporate additional interaction constraints between the agents in the cluster as required. Whenever the sub-solutions of different clusters emerge to be in conflict, the corresponding clusters are merged. The whole process of merging and re-planning repeats until all conflicts are resolved. (In the worst-case, this happens when all agents have merged into one large cluster which reduces the situation to the CCG method solving the overall agent problem.)

Remarks on how many conflicting constraints to merge. In CCG (and by extension, also in SDCG), the question of how many of the violated constraints to merge (cf. Eq. 6.5) is a design parameter whose optimal choice may depend on the application at hand. In our implementation, we opted to merge all violated coupling constraints. However, it would be conceivable to only merge one conflicting constraint at a time. This can be beneficial in situations where (e.g. due to the structure of the local feasible set) the addition of some interaction constraints automatically rules out violations of others. Investigating the merits of either choice will have to be deferred to future work. For the time being, we provide two examples that illustrate

either choice can trump the other, depending on the scenario.

A simple scenario where merging the constraints with *all* conflicting interaction-constraints is beneficial is given in the following example:

Example 6.1.2. Consider a two-agent resource consumption scenario with agent set $\mathfrak{A} = \{1, 2\}$ and two indivisible resources that can be consumed by the agents. Agent $\mathfrak{a}$'s plan $p^{\mathfrak{a}} \in \{0, 1\}^2$ is a two-dimensional binary vector where $p_j^{\mathfrak{a}} = \begin{cases} 1, \text{if resource j is consumed by agent } \mathfrak{a}, \\ 0, \text{ otherwise.} \end{cases}$. We assume agent $\mathfrak{a}$ receives a reward of $\mathfrak{a}\, p_j^{\mathfrak{a}}$ for consuming resource j. That is, agent 2 receives twice the reward for consuming a resource agent 1 receives. This gives rise to an individual cost function $c^{\mathfrak{a}}(p^{\mathfrak{a}}) = -\mathfrak{a} \sum_{j=1}^{2} p_j^{\mathfrak{a}}$. Therefore, both agents ideally would like to consume both resources. That is, starting with $\mathcal{C}(0) = \mathbb{R}^2, I(0) = \emptyset$ and local feasible sets $F^{\mathfrak{a}} = \{p^{\mathfrak{a}} \in \{0, 1\}^2\}$ ($\mathfrak{a} \in \mathfrak{A}$), the jointly optimal solution at the end of the 0th iteration is $p(0) = (p^1(0), p^2(0))$ with $p^1(0) = p^2(0) = (1, 1)$. Having only one unit of each resource $j \in \{1, 2\}$ available, we impose the inter-agent constraints $C_j(p) = \sum_{\mathfrak{a} \in \mathfrak{A}} p_j^{\mathfrak{a}} - 1 \leq 0$ which together ensure that no resource can be consumed by two agents simultaneously. Clearly, the initial joint solution $p(0)$ violates both the first and the second interaction constraint that preclude consumption of one resource by two agents. So, in iteration $i = 1$ the constraint set $\mathcal{C}(1)$ needs to be enlarged. Now, if we were to only incorporate the first constraint (i.e. $I(1) = \{1\}$, $C(1) = \{C_1(p) \leq 0\}$) only overconsumption of the first resource would be prohibited. This would cause both agents to over-consume the second resource which in turn necessitates a third round $i = 2$ that also imposes the second resource interaction constraint: with $I(2) = \{1, 2\}, C(2) = \{C_1(p) \leq 0, C_2(p) \leq 0\}$ the final jointly optimal feasible solution becomes $p(2) = (p^1(2), p^2(2))$ where $p^1(2) = (0, 0)$, indicating that agent 1 gets no resources and where $p^2(2) = (1, 1)$, indicating that agent 2 gets to consume both resources. Note, we could have avoided the computation incurred in iteration $i = 1$ had we followed the approach to incorporate both interaction constraints in the first iteration (rather than just one of them) after both had been violated after the 0th iteration.

In the previous example, both interaction constraints could be violated simultaneously and so, incorporating both of them after the first optimization round was beneficial. However, in some cases, the addition of some interaction constraints automatically rules out violations of others or removes the cost incentives for such violations. An example of the latter arises simply by modifying the individual cost functions of the previous example:

Example 6.1.3. Assume we repeat Ex. 6.1.2 but with the modification of the individual cost functions to $c^{\mathfrak{a}}(p^{\mathfrak{a}}) = -\mathfrak{a}\, p_1^{\mathfrak{a}}\, p_2^{\mathfrak{a}} + \frac{1}{4}(p_1^{\mathfrak{a}} + p_2^{\mathfrak{a}})$, ($\mathfrak{a} \in \mathfrak{A} = \{1, 2\}$). Since the plan components are binary, the cost function expresses that an agent only benefits from a resource if he is also allowed to consume the other resource – otherwise the agents prefer not consuming any resource at all. Going through the iterations as before we find that the socially optimal jointly feasible solution $p = (p^1, p^2)$ with $p^1 = (0, 0)$ and $p^2 = (1, 1)$ is found after the first iteration, even if only the first interaction constraint $C_1(p) \leq 0$ is added to the constraint set after the

0th iteration. This is because the imposition of $C_1(p) \leq 0$ denies access of the first resource to agent 1 (since not both agents can use the first resource now and agent 2 would benefit more from consuming it than agent 1). Now, due to the nature of the cost function, this means that the minimal cost is incurred if agent 1 consumes no resources ($p^1(1) = (0,0)$) and agent 2 gets to consume all resources ($p^2(1) = (1,1)$). Subsequently re-optimising with the second constraint $C_2(p) \leq 0$ added would not alter the plans and could only increase computational effort of the joint optimisation problem in comparison to the more relaxed problem where the second constraint is not imposed.

Properties. CCG and SDCG have the following properties:

(i) *Completeness and optimality.* It is fairly obvious that both methods are optimal and complete, provided there is an algorithm that can solve each sub-problem optimally and in finite time.

The proof relies on the fact that in each iteration, a relaxed version of the optimal centralized OP is solved. In the iterative process, conflict detection then uncovers a superset of those interaction constraints that would be active for an optimal plan vector under the full global OP.

We formalise the result in the following theorem:

Theorem 6.1.4. *Assume problem OPT (as per Eq. 6.3) has an optimal jointly feasible solution. Moreover, assume that for any subset of agents $\mathfrak{S} \subseteq \mathfrak{A}$ and subset of constraints $\mathcal{C}' \subseteq \mathcal{C}$ there is an algorithm to optimally solve the sub-problem $p^{\mathfrak{S}} \longleftarrow \min_{p^{\mathfrak{S}} \in F_L \cap F_{\mathcal{C}'}} \sum_{\mathfrak{a} \in \mathfrak{S}} c^{\mathfrak{a}}(p^{\mathfrak{a}})$ in finite time. Then using the same algorithm with the CCG and SDCG methods ensures that both CCG and SDCG are complete and optimal.*

Proof. (i) CCG: Termination: By assumption, in each iteration i, the sub-problems as per (6.4) are feasible and can be solved in finite time. Therefore, each iteration terminates in finite time. Furthermore, CCG terminates after iteration i, iff $I_{\text{confl}}(i) = \emptyset$. If CCG does not terminate after iteration i, this means that $I_{\text{confl}}(i) \neq \emptyset$. Hence, $I(i-1) \subsetneq I(i) \subseteq \mathcal{I}$, $\forall i \geq 1$ where $\mathcal{I} = \{1, \ldots, m\}$ is the finite index set of all coupling constraints. Therefore, the sequence $(|I(i)|)_i$ is strictly monotonously increasing and upper bounded by $m < \infty$. Since the sequence $(I(i))_i$ is confined to assume subsets of from finite set {1,...,m}, it has to terminate after T steps, for some $T \leq m$.

Optimality: Since $I(i-1) \subsetneq I(i) \subseteq \mathcal{I}$, $\forall i \in \{1, ..., T\}$ obviously $F_{\mathcal{C}} \subseteq F_{\mathcal{C}(i)} \subsetneq F_{\mathcal{C}(i-1)}$, $\forall i \in \{1, ..., T\}$. Thus, each optimisation problem in iteration i as per (6.4) solves a relaxed version of OPT (cf. expression (6.3)). Hence, $s(p_{ccg}(1)) < \ldots < s(p_{ccg}(T)) \leq s(p_{opt}) < \infty$ where, as before, s denotes social cost and $p_{opt} \in \mathcal{P}_{opt} := \text{argmin}_{p \in F_L \cup F_{\mathcal{C}}} s(p)$ is a socially optimal jointly feasible solution.

It remains to be shown that $s(p_{ccg}(T)) = s(p_{opt})$. For contradiction, assume that $s(p_{ccg}(T)) < s(p_{opt})$. But this implies that p_{ccg} is not jointly feasible – otherwise $s(p_{opt})$ would not be a socially optimal jointly feasible cost. Hence, the solution $p_{ccg}(T)$ violates at least one coupling constraint. But, this is a contradiction to T being the terminal time step, for which necessarily $I_{\text{confl}}(T) = \emptyset$ (otherwise the conflict detection module would not allow termination).

(ii) SDCG: We have shown that CCG is optimal and complete. Hence, every iteration of SDCG returns an optimal solution to each of its sub-problem in finite time. The rest follows in complete analogy to (i). By the same argument of monotonous sequences of constraint indices and relaxed feasible sets we can show that the social costs of the joint solutions SDCG produces after each iteration is approaching the optimal jointly feasible social cost from below and has to terminate after $T \leq m$ iterations. In (i) we have shown that collision detection eventually uncovers a superset of a constraint set sufficiently restrictive to render the joint solution jointly feasible. By the same argument, the terminal joint solution of SDCG is jointly feasible. Thus, $s(p_{sdcg}(T)) = s(p_{opt})$. □

(ii) *Improved average-case tractability*. One can conceive problem instances where our methods eventually solves the full centralized problem (OPT). Since computation is expended before this final OP is created, they would be slower than OPT in such cases. Often however, agent plan interactions occur on a small subset of plan dimensions (between few agents) and resources (e.g. nodes). In such cases, optimal solutions only involve few active interaction constraints that may involve only a fraction of plan vector components each. For instance, in large graphs where agents will never (or only sparsely) encounter, it would be computationally wasteful to generate plans based on mixed-integer problems that even search over those potential interactions that will never influence the optimal solution anyway. Consequently, a lot can be gained by our method in such scenarios. Our simulations we will describe below confirm this intuition.

(iii) *Semi-distributivity (SDCG)*. In cases, where the full centralized OP does not have to be solved, SDCG may succeed in decomposing the large joint problem into low-dimensional sub-problems that could be solved more locally and in parallel. However, since the approach may eventually lead to the necessity to solve the full central OP in the worst case, we cannot claim that SDCG is fully distributed. Furthermore, the decomposition is likely to be of greatest benefit in comparison to CCG, if the coupling constraints only depend on small subsets of agents. Otherwise, few conflicts would rapidly result in joining all agents. Therefore, if given a choice between stating coupling constraints based on large sets of agents, or small sets, it can be beneficial to opt for the small sets if SDCG is to be employed. Once again, our graph planning problem serves as an example: For each vertex in the graph one might add a constraint stating that the sum of occupancies of that node of all agents should be less or equal to one. Of course, this one constraint involves all agents. So, with this constraint, SDCG would merge all agents as soon as two agents would plan on using that node – even those whose plans played no part in the constraint violation. Of course, this is unnecessary. Instead one could replace that one coupling constraint by many coupling constraints, one for each agent pair, stating that the sum of occupancies of that node by that agent pair is not allowed to exceed one. This has the effect, that if our two agents plan on visiting the node at the same time, only their two problems would be merged rather than those of all agents.

Fixed-Priority Methods (FP)

In the SDCG method, problems are kept separate as much as possible but are combined into a larger joint OP whenever conflicts are detected. This allowed the method to provide an optimality guarantee. However, it has the disadvantage that a fully centralised problem with a high-dimensional planning space may emerge in the course of coordination. This means that the method is neither fully distributed and may become intractable. As a way out, a plethora of distributed methods have been proposed that keep the problems separate at all times and achieve coordination by injecting additional constraints into the descriptions of the feasible sets of some of the agents. A simple but widely used methods are fixed-priority methods (e.g. [26]). Here each agent has a ranking or priority. Whenever a higher ranking agent $\mathfrak{a}$ is in conflict with a lower ranking agent $\mathfrak{r}$, the lower ranking agent inserts an *avoidance* constraint that alters its own plan such that the detected conflict is avoided. The higher ranking agent does not have to alter his plan. In a collision-avoidance scenario, this essentially means that the trajectories of higher ranking agents are obstacle trajectories which the lower ranking agents have to avoid. For an illustration of the method's deployment in a continuous-time collision avoidance setting the reader is referred to Sec. 5.

Lazy Auctions (AUC, AUC2)

While the FP method is simple and distributed (or, could even be decentralised without requiring communication), it can be highly sub-optimal and often generate infeasible solutions due to the rigidity of the approach. To alleviate the situation a variety of modifications to the fixed ranking have been proposed that search for a good ranking (eg. [27, 174, 175]).

By contrast, we propose a flexible auction-based approach that decides on resource allocation on a case-by-case basis. (This method is a discrete optimisation version of the auction method we proposed in Ch. 5). If a conflict between agents is detected the agents bid for the right to keep their feasible sets unaltered by an avoidance constraint.

With the aim to heuristically optimise for social cost, we define the bids such that the social cost after the auction is greedily minimized. That means that the winner is determined to be the agent who would be most adversely impacted if he lost the auction and had to change his plan by adding an avoidance constraint. Having an auction scheme where the highest bid wins, the bids will thus be defined to quantify the additional cost resulting from incorporation of the avoidance constraint.

We will now formalise these intuitive definitions. In order to allow our idea of conflict-avoidance by insertion of local constraints to be viable, we need to assume that this is always possible and that such avoidance constraints can be computed in finite time. We formalise this assumption as follows:

Assumption 6.1.5. We assume that for each coupling constraint $C_j(\cdot) \leq 0$ involving exactly plans of agents

$\mathfrak{S} \subset \mathfrak{A}$ and for each $\mathfrak{a} \in \mathfrak{S}$ there exists a computable representation of a set of plans $V_j^{\mathfrak{a}}$ such that any $p^{\mathfrak{a}} \in V_j^{\mathfrak{a}}$ "does not contribute" to the violation of the jth coupling constraint.

What this assumption means and how $V_j^{\mathfrak{a}}$ has to be defined depends on the application at hand. For example, let us revisit the graph planning scenario. Assume the j th component $p_j^{\mathfrak{a}}$ of plan $p^{\mathfrak{a}}$ specifies agent $\mathfrak{a}$'s intent to occupy node j, where $p_j^{\mathfrak{a}} \in \{0, 1\}$ and $p_j^{\mathfrak{a}} = 1$ means the agents does intent to occupy node j. The coupling constraint might express that the node is an exclusive resource which no more than two agents can simultaneously claim: $C_j(p) = \sum_{\mathfrak{a}\in\mathfrak{A}} p_j^{\mathfrak{a}} - 1 \leq 0$. Here, $V_j^{\mathfrak{a}}$ could be the set of all plans $p^{\mathfrak{a}}$ that avoid occupying node j: $V_j^{\mathfrak{a}} = \{p^{\mathfrak{a}} \,|\, p_j^{\mathfrak{a}} \leq 0\}$. From a collision avoidance perspective, $V_j^{\mathfrak{a}}$ could be interpreted as the complement of a *virtual obstacle* (given by the set of plans contained in $\bar{V}_j^{\mathfrak{a}}$) around that node. The introduction of the avoidance constraint $p_j^{\mathfrak{a}} \leq 0$ (or $p \in V_j^{\mathfrak{a}}$) enforces that $\mathfrak{a}$ avoids this virtual obstacle (and hence, no longer contributes to the violation of the jth coupling constraint).

With this assumption and the intuitive interpretation of a virtual obstacle in mind, we can proceed with the formal description of the auction method:

The core bidding protocol. In iteration i, agents $\mathfrak{a} \in \mathfrak{S} \subset \mathfrak{A}$ have come up with plans

$p^{\mathfrak{a}}(i) \leftarrow \min_{p^{\mathfrak{a}}\in F^{\mathfrak{a}}\cap F^{\mathfrak{a}}_{avoid}(i)} c^{\mathfrak{a}}(p^{\mathfrak{a}})$ where $F^{\mathfrak{a}}_{avoid}(i)$ is the set of all local plans satisfying all avoidance constraints $\mathfrak{a}$ has incorporated up until iteration i.

Assume that their plans are in conflict with each other over the jth constraint. That is, $C_j(p^{\mathfrak{S}}(i)) > 0$ where $p^{\mathfrak{S}}(i) = (p^{\mathfrak{a}}(i))_{\mathfrak{a}\in\mathfrak{S}}$.

Each agent $\mathfrak{a} \in \mathfrak{S}$ enters the auction and computes a bid $b^{\mathfrak{a}}$ as per:

$$b^{\mathfrak{a}} = l^{\mathfrak{a}}(i) - s^{\mathfrak{a}}(i). \tag{6.6}$$

Here $s^{\mathfrak{a}}(i) = c^{\mathfrak{a}}(p^{\mathfrak{a}}(i))$ is the cost of the "short way", i.e. the cost of the unaltered plan. Its counter-part, $l^{\mathfrak{a}}(i)$, quantifies the cost of the "long way around". That is, we have $l^{\mathfrak{a}}(i) = c^{\mathfrak{a}}(\tilde{p}^{\mathfrak{a}}(i))$

where $\tilde{p}^{\mathfrak{a}}(i) \leftarrow \min_{p^{\mathfrak{a}}\in F^{\mathfrak{a}}\cap F^{\mathfrak{a}}_{avoid}(i)\cap V_j^{\mathfrak{a}}} c^{\mathfrak{a}}(p^{\mathfrak{a}})$ is the altered plan resulting from updating the set of collision avoidance constraints to avoid the new conflict. If the updated feasible set is empty (i.e. $F^{\mathfrak{a}}\cap F^{\mathfrak{a}}_{avoid}(i)\cap V_j^{\mathfrak{a}} = \emptyset$) and no solution can be found, we set $l^{\mathfrak{a}}(i) = \infty$.

The winner is determined to be the agent who submits the highest bid. If there is a tie between multiple agents, the agent with the highest index wins. The winner can keep his plan unaltered. By contrast, each agent $\mathfrak{a}$ who lost the auction has to replan in the following iteration, setting $F^{\mathfrak{a}}_{avoid}(i+1) := F^{\mathfrak{a}}_{avoid}(i) \cap V_j^{\mathfrak{a}}$. The procedure repeats until all conflicts are avoided.

To gain an intuitive motivation for the bidding rule, notice the bid quantifies the regret an agent expects to have for losing the auction (given its current belief of the availability of resources). Let $\mathfrak{q}$ be the winner of the auction after iteration i. Acknowledging that $s^{\mathfrak{q}}(i) + \sum_{\mathfrak{a}\in\mathfrak{S}\setminus\{\mathfrak{q}\}} l^{\mathfrak{a}}(i)$ is the tentative social cost after

the auction, we see that the winner determination rule greedily attempts to minimize social cost: $\forall \mathfrak{r} \in \mathfrak{S}$: $b^{\mathfrak{q}}(i) \geq b^{\mathfrak{r}}(i) \Leftrightarrow \forall \mathfrak{r} \in \mathfrak{S} : s^{\mathfrak{r}}(i) + \sum_{\mathfrak{a} \in \mathfrak{S} \setminus \{\mathfrak{r}\}} l^{\mathfrak{a}}(i) \geq s^{\mathfrak{q}}(i) + \sum_{\mathfrak{a} \in \mathfrak{S} \setminus \{\mathfrak{q}\}} l^{\mathfrak{a}}(i)$.

Note, that the bidding rule is myopic in the sense that only the current conflict is taken into account when computing the bids as per Eq. 6.6. Obviously, the method therefore is not optimal. Furthermore, it is not always complete – although we have provided a termination guarantee of a modified bidding rule in the specific context of collision avoidance in graphs [48, 121].

We have found two modifications of the generic auction method to be useful to foster both properties:

Release of resources: Note, we may interpret the obstacle $V_j^{\mathfrak{a}}$ as a resource the agent $\mathfrak{a}$ loses when incorporating the avoidance constraint $p^{\mathfrak{a}} \in V_j^{\mathfrak{a}}$. Whenever an agent who won an auction over a constraint j in a previous iteration, no longer has a plan that would have triggered a conflict on constraint j, it communicates this fact to all agents he has won the pertaining auction against. Each of those agents $\mathfrak{a}$ may then decide to relax their feasible sets $F_{avoid}^{\mathfrak{a}}$ by removing the corresponding virtual obstacle constraints. To avoid cycles, we have found it useful to limit the number of releases of the same virtual obstacles. In our implementations, an agent who has released a resource before will never release it a second time.

Bids that take anticipated conflicts into account (AUC2). The bids as per Eq. 6.6 only depend on local plan cost. To avoid deadlocks and reduce the number of iterations, we have found that an effective heuristic is to penalize anticipated conflicts as well. That is, if due to loss of an auction and consequential inclusion of a virtual obstacle, the agent would be directed into a part of solution space with a larger number of conflicts with other agents, this certainly would an undesirable outcome – chances are it will lose even more auctions further down the line of coordination and this will spawn more coordination effort and probably further increased cost. Without the necessity of looking too far into the future, the agents could modify their bids to

$$b^{\mathfrak{a}}(i) = l^{\mathfrak{a}}(i) - s^{\mathfrak{a}}(i) + w_{confl}\Delta_{confl}. \tag{6.7}$$

Here, $w_{confl}\Delta_{confl}$ is a weighted difference between the number of conflicts with other agents' plans under the altered plan and the conflict count under the current plan. The weight $w_{confl} \geq 0$ is a design parameter.

Communication. As with all the other coordination methods there are several degrees of freedom regarding the architectural implementation of the mechanism. For instance, to detect a conflict, all agents communicate their current plans to all other agents. If directed sink-source communication is employed this results in a communication effort of $\mathcal{O}(A^2)$ messages per coordination round, where A is the number of agents. Utilisation of broadcast messages can reduce the communication effort to $\mathcal{O}(A)$. In either case, each agent would be responsible to detect the next conflict and arrange an auction with the other agents. Alternatively, the mechanism designer could set up a number of additional dedicated conflict detectors and auctioneers (e.g. one for a set of time steps or coupling constraints). That is, if an agent conceives a plan affecting the cou-

pling constraint j they tell the corresponding constraint agent. This constraint agent in turn detects possible conflicts and invites all involved agent to join the auction (and may function as an auctioneer).

6.2. Illustrations in multi-agent graph planning

Our general descriptions made so far were occasionally accompanied by explanations set in a graph planning problem as per Def. 6.1.1. For us, this problem will merely serve as simple test domain to further elucidate our coordination approaches. However, a slightly modified version, often called multi-agent pathfinding, is an active research area in its own right and has spawned a variety of specialist algorithms dedicated to its solution (cf. Sec. 2.3 and references therein).

Graphs are mathematical abstractions that are simple but reflect the essence of many real-world planning scenarios ranging from multi-robot path-finding and computer games to operations research applications such as supply chain management [13].

A graph $\mathcal{G} = (V, E)$ is a pair consisting of a set V of $vertices$ or $nodes$ and a set $E \subset V^2$ of edges. The edges impose a relation structure on the vertices. In a robot trajectory planning scenario the vertices could correspond to locations of a spatial graph. Assuming discretised time we could construct $\mathcal{G}$ such that $(v, v') \in E$ iff a robot can travel from location v to v' in one time step. Finally, we assume the robot incurs a cost $c^e > 0$ for traversing an edge $e \in E$. Depending on the objective, such a cost can model the time delay (e.g. $\gamma(e) = 1$ [sec]) for moving from v to v' (where the vertices are chosen such that $e = (v, v')$).

For a given graph, a binary flow is a function $f : E \to \{0, 1\}$ adhering to the network topology. That is, it obeys the Kirchhoff rules which states that the sum of all flows assigned to edges entering a node has to equal the sum of the flows of all edges pointing away from that node. Furthermore, normally there is a source node and a sink node that are an exception to the rule. That is, the source node has more flow leaving it to meet a demand level of flow. Conversely, the sink node absorbs all the flow entering it, up to the demand level the pertaining source node has created.

Flow problems are fairly general models for many planning problems. They can model interdependent processes on a production line or network traffic in routing. In aforementioned-pathfinding applications, a flow translates to a route through a map whose topology is given by a graph. In that case additional constraints may be imposed that guarantee that two robots never travel through each other, i.e. plan a flow that uses an undirected edge or node simultaneously [273, 274].

Connecting to our coordination methods, in what is to follow, we will formalise multi-agent flow and routing problems as optimisation with mixed-integer constraints. We will then use these problems as a test bed for further elucidation and comparison of the different coordination approaches.

6.2.1. Mixed-integer linear programming formulations

In this subsection, we will formalize the graph planning problems as optimisation problems adhering to our notation and state them as mixed-integer programmes. Firstly, we state the binary optimal multi-commodity flow problem (BOMCFP) as such, which we imagine should be standard. Then we explain how MTAPP can be formulated as such by unrolling a spatial graph into a spatio-temporal one. We proposed this method in [121]. Independently, [273] proposed a very similar reduction a year later. The only difference seems to be that they, being interested in MAPF for physical robots, also introduced additional constraints that prevented agents from simultaneously traversing the same edge (in opposite directions) within a spatial graph. By contrast we are content with only preventing simultaneous occupation of the same spatial nodes.

Finding disjoint binary flows – BOMCFP

Let $\mathcal{G} = (V, E)$ where V is a set of vertices and $E \subset V^2$ a set of edges. For convenience, we introduce the following notation: For a vertex $v \in V$ we define $(v, *) := \{(v, n) | n \in V, (v, n) \in E\}$ to be the set of all edges leaving v and $(*, v) := \{(n, v) | n \in V, (n, v) \in E\}$ as the set of all edges entering v.

A plan $p^{\mathfrak{a}} \in \mathcal{P} = \{0, 1\}^{|E|}$ of agent $\mathfrak{a}$ is a binary vector whose ith component $p_i^{\mathfrak{a}}$ equals 1 iff the agent desires to use the ith edge $e_i \in E$. We desire a joint plan $p := (p^1, \dots, p^A)$ that encodes that (I) each agent $\mathfrak{a}$ leaves his start node $S^{\mathfrak{a}}$ and reaches his goal; (II) that this path leading to $D^{\mathfrak{r}}$ is consistent with the graph's topology; and, (III) that the path is not in collision with any other agent's plan. The latter condition is to say that no two agents can use an edge that leads into the same node.

We will translate these desiderata into constraints on the plans. For each individual plan $p^{\mathfrak{a}}$ ($\mathfrak{a} \in \mathfrak{A}$) with source node $S^{\mathfrak{a}}$ and sink node $D^{\mathfrak{a}}$ we introduce the following constraints:

- (0.i) To restrict solutions to binary flows, we impose the constraint $p^{\mathfrak{a}} \in \{0, 1\}^{|E|}$.
- (I.i) Define the vector σ by its components $\sigma_i = \begin{cases} 1, & \text{if } e_i \in (S^{\mathfrak{a}}, *) \\ 0, & \text{otherwise} \end{cases}$ for $i = 1, ..., |E|$. Introducing the constraint $\sigma^\top p^{\mathfrak{a}} = 1$ ensures that 1 unit of flow leaves start node $S^{\mathfrak{a}}$.
- (I.ii) We introduce a constraint $b^\top p^{\mathfrak{a}} \leq 0$ enforcing that all the flow that agent $\mathfrak{a}$ plans on sending from source node $S^{\mathfrak{a}}$ reaches sink node $D^{\mathfrak{a}}$. This is accomplished by defining its components as follows:

$$b_i := \begin{cases} -1, & \text{if } e_i \in (S^{\mathfrak{a}}, *) \backslash \{(S^{\mathfrak{a}}, D^{\mathfrak{a}})\} \\ 1, & \text{if } e_i \in (*, D^{\mathfrak{a}}) \backslash \{(S^{\mathfrak{a}}, D^{\mathfrak{a}})\} \\ 0, & \text{otherwise} \end{cases}.$$

- The constraint $q^\top p^{\mathfrak{a}} \leq 0$ where $q \in \mathcal{P}, q(i) = \begin{cases} 1, & \text{if } e_i \in (D^{\mathfrak{a}}, *) \\ 0, & \text{otherwise} \end{cases}$ $(i = 1, ..., |E|)$ makes sure that none of $\mathfrak{a}$'s flow leaves the sink.

- (II.i) Next we need to make sure that all plans are consistent with the graph topology. In particular, Kirchhoff's rule ("flow that enters a transient node exits a transient node") is met. For every agent $\mathfrak{a} \in \mathfrak{A} = \{1, \ldots, A\}$ and node $n \in V \backslash \{S^{\mathfrak{a}}, D^{\mathfrak{a}}\}$ we introduce a constraint: $k_n^\top p^{\mathfrak{a}} \leq \underline{0}$ where the components $k_{n,i}$ of the k_n are defined by:

$$\forall i \in \{1, ..., |E|\} : k_{n,i} = \begin{cases} -1, & \text{if } e_i \in (*, n) \\ 1, & \text{if } e_i \in (n, *) \\ 0, & \text{otherwise.} \end{cases}$$

To ensure coordination, we have yet to define the interaction constraint guaranteeing that no two plans should involve edges entering the same node. To this end, for each node $n \in V$, we define the vector $\kappa_n \in \mathbb{R}^{|V|}$ whose i component is $\kappa_{n,i} = \begin{cases} 1, & \text{if } e_i \in (*, n) \\ 0, & \text{otherwise} \end{cases}$. Defining the A-fold Cartesian vector product $\bar{\kappa}_n := \prod_{i=1}^{A} \kappa_n = \underbrace{\kappa_n \times ... \times \kappa_n}_{A \text{ times}}$ the collision avoidance constraints become

- (III) $\forall n \in V : \bar{\kappa}_n^\top p - 1 \leq 0$.

Quantity $\bar{\kappa}_n^\top p$ essentially counts the total number of flow leading into node n and the constraint ensures collision avoidance by restricting this total number to be not greater than one.

Linking up to our definitions in the preceding sections, we summarize:

- The local feasible set of agent $\mathfrak{a}$ can be defined as:

$$F^{\mathfrak{a}} := \left\{ p^{\mathfrak{a}} \in \{0,1\}^{|E|} \,\middle|\, \sigma^\top p^{\mathfrak{a}} = 1, b^\top p^{\mathfrak{a}} \leq 0, q^\top p \leq 0, \forall n \in \{1, ..., |V|\} \backslash \{S^{\mathfrak{a}}, D^{\mathfrak{a}}\} : k_n^\top p^{\mathfrak{a}} \leq 0 \right\}.$$

- The set of jointly conflict-free plans is: $F_{\mathcal{C}} = \{p \in \{0,1\}^{A|E|} \,|\, \forall n \in V : \bar{\kappa}_n^\top p \leq 1\}$.

Notice, all constraints are linear with restrictions of the variables to assume binary values.

The coordination problem could now be solved centrally as per optimisation problem (6.3). The cost function definitions depend on the application. For instance, in a weighted shortest paths problem with edge transition costs, we could simply define $c^{\mathfrak{a}}(p^{\mathfrak{a}}) = (\gamma^{\mathfrak{a}})^\top p^{\mathfrak{a}}$ where the ith component of vector $\gamma^{\mathfrak{a}}$ denotes the cost the agent incurs when routing a flow that uses edge e_i. This is a linear function. In this case, the OP is a binary mixed-integer linear programme (BLP) and can be solved with solvers such as *CPLEX* or Matlab's inbuilt function *bintprog*.

If the auction method is to be employed, we still have to specify a virtual obstacles via an avoidance constraint. This can be done by simply denying access to a node that was lost in an auction. For instance, if $\mathfrak{a}$ lost node n, it would intersect future plans with $V_n^{\mathfrak{a}} = \{p^{\mathfrak{a}} \mid \bar{\kappa}_n^\top p^{\mathfrak{a}} \leq 0\}$ the set of all plans that do not route any flow into the lost node.

Collision-free trajectories – MATPP

Up until now, we have shown how to solve a multi-agent graph planning problem that enforces disjoint *paths*. In applications such as multi-robot routing or control of discrete dynamic systems, *trajectory* planning often is of greater interest. For instance, here the nodes in a spatial graph may correspond to resources such as locations in task space.

Finite-time horizon trajectory planning. We restate the trajectory planning problem with a finite time horizon:

Definition 6.2.1 (Finite-time horizon MATPP)**.** *A group of agents $\mathfrak{A} = \{1, ..., A\}$ desires to find individual plans $p^1, ..., p^A$, respectively, that translate to collision-free trajectories (i.e. sequences of nodes or edges indexed by time) of length $T + 1$ in a graph such that each agent $\mathfrak{a}$'s trajectory leads from its start $S^{\mathfrak{a}}$ to its destination $D^{\mathfrak{a}}$ and the topology of the graph is respected. That is, agents can move from one node to another in one time step if and only iff the nodes are connected by an edge.*

We assume the following model: Let $R = \{R_1, ..., R_M\}$ be the set of resources (states /locations/nodes) in the system that agents $1, ..., A$ can claim in a discrete sequence of time steps $t = 0, 1, ..., T$. Here $T \in \mathbb{N}$ is a *time horizon*. Remember, conflict-avoidance dictates that resource claims are mutually exclusive (i.e. no two agents can simultaneously use the same resource). Therefore, at the very least, one should be able to assume that the number of resources is at least as high as the number of agents, $0 < A \leq M$.

Assuming not all states are reachable from all other states in one time step, we can express the reachability (within one time step) as a sequence of relations $E_t = (R \times R)_t \, (t \in \mathbb{N}_0)$ where $(R_u, R_v) \in E_t$ iff a agent can reach R_v at time step $t + 1$ when it is in R_u at time step t. (For simplicity, we tacitly assume homogeneous agent capabilities that are known in advance.) Let Z_t^a denote the resource agent a plans to occupy at time t. In this environment, a *legal* plan corresponds to a sequence of states $(Z_t^a)_{t \in \mathbb{N}_0}$ such that $\forall t \in \mathbb{N}_0 : (Z_t^a, Z_{t+1}^a) \in E_t$. Furthermore, the *overall plan* now corresponds to the joint sequence $(Z_t)_{t \in \mathbb{N}_0}$ where $Z_t = Z_t^1 \times \ldots \times Z_t^A$.

We can express this model as a graph planning problem. Spatio-temporal graph $\mathcal{G} = (V, E)$ consists of vertices $V = \{v_{t,m} | t \in \mathbb{N}_0, m \in \{1, ..., M\}\}$ where vertex $v_{t,m}$ corresponds to resource R_m at time t. So the graph consists of T time layers ("rows") indexed by t and each of the M "columns" corresponds to one state each. The set of edges reflects the reachability conditions defined above. That is, we assume

$(v_{t,l}, v_{t+1,m}) \in E$ iff $(R_l, R_m) \in E_t$. For an illustration refer to Fig. B.1.

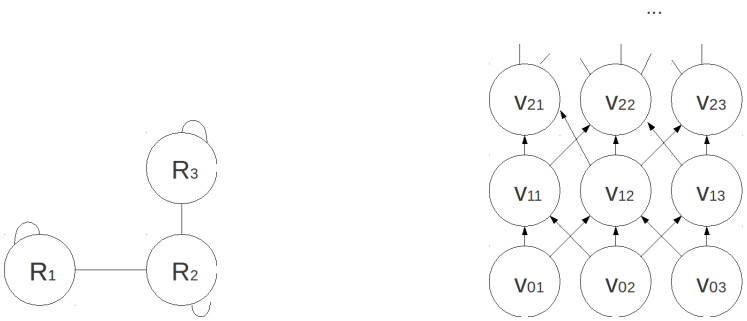


Figure 6.1.: Left: Spatial graph with $M = 3$ spatial nodes. Right: First three time layers (rows) of the corresponding spatio-temporal graph.

With these notions in mind, agent $\mathfrak{a}$'s plan in coordination iteration i can be represented as a sequence of vertices or edges in spatio-temporal graph $\mathcal{G}$. We denote this sequence as $(z_t^{\mathfrak{a}}(i))$ where $z_t^{\mathfrak{a}}(i) \in \{v_{t,1}, ..., v_{t,M}\}$ is the vertex agent $\mathfrak{a}$ intends to visit in time layer t. Notice the relationship $z_t^{\mathfrak{a}}(i) = v_{t,m} \Leftrightarrow Z_t^{\mathfrak{a}}(i) = R_m$.

When executing a plan that involves a state transition corresponding to using an edge in E the agent receives a cost specified by edge cost function $\gamma^{\mathfrak{a}} : E \to \mathbb{R}_+$. (Note, above-mentioned cost function $c^{\mathfrak{a}}$ on the set of feasible plans can now be defined as the sum of all costs of edges (acc. to $\gamma^{\mathfrak{a}}$) the plan would have to use to generate the implied sequence of nodes.)

At initial time $t = 1$, agents $1, ..., A$ are in their start states $S^1 \in R, ..., S^A \in R$, respectively ($S^a \neq S^r, \forall a \neq r$). That is they are all at distinct nodes in (time-) *layer* 0. The trajectory planning problem can now be expressed as finding a collection of low-social cost plans corresponding to legal sequences of nodes (or edges) $(z_t^{\mathfrak{a}})_{t\in\{1,...,T\}}$ ($a \in \mathcal{A}$) in the spatio-temporal graph $\mathcal{G}$ such that the local constraints are met, they are collision-free (i.e. $z_t^{\mathfrak{a}} \neq z_t^r, \forall t \in \mathbb{N}_0 \forall a \neq r$) and the agents reach their destinations from their start states.

Of course each legal sequence of nodes has a one-to-one relation to a sequence of edges connecting the nodes. Therefore, we can either state a plan $p^{\mathfrak{a}}$ as a sequence of spatio-temporal nodes or a sequences of edges in the spatio-temporal graph. Since collision-free trajectories in the spatial graph coincide with disjoint paths in the spatio-temporal graph, we have found a reduction of the trajectory planning problem to the multi-commodity flow problem (BOMCFP) – which we already know how to solve.

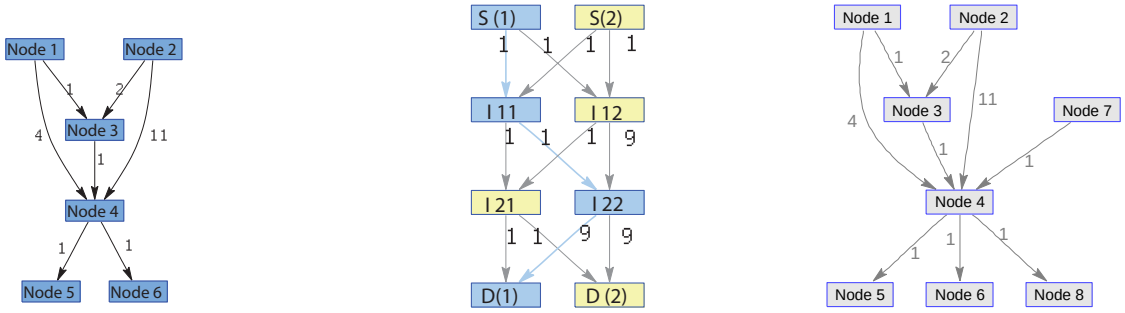
Unbounded time-horizon problems. So far, we have assumed that planning was over a finite time horizon T. In many cases however, that time horizon can render the problem infeasible. And, it might be possible that one is interested in finding any trajectories that connect all starts and goals without a time limit. We refer to this setting as the unbounded time-horizon trajectory planning problem. Unfortunately, the smallest horizon $\underline{T}$ such that a feasible solution to the trajectory exists depends on many factors such as the number

of agents and the topology of the graph. This problem is similar to runtime prediction and is known to be hard. And, setting the horizon to a conservatively large number increases number of constraints and plan dimensionality and will therefore substantially slow down the optimisation. An obvious solution would be to follow an incremental approach where the time-horizon is increased successively as needed. For instance, looking at a very similar problem, Yu and Lavalle [273] proposed to start out with horizon $T = 1$ which then is doubled whenever the solution is not feasible. As opposed to this approach, in the simulations given in [121], we started with a relatively conservative horizon, increasing it by the number of agents $\mathfrak{A}$ in each time step.

6.2.2. Lazy auction properties illustrated in graph planning

In this subsection, we will illustrate our lazy auction bidding rule in the context of the graph trajectory planning scenario introduced above. In particular, we consider a few specific simple trajectory planning examples.

Ex. A. Two agents desire to find low-cost trajectories in a graph with transition costs as depicted in Fig. 6.2(a). Agent 1 desires to find a trajectory from Node 1 to 5, Agent 2 from Node 2 to 6.



(a) Ex. A. (b) Ex. B. (c) Ex. C.

Figure 6.2.: Three examples. Numbers next to the edges denote the transition costs. Ex. B : S(a)/D(a): start/destination of agent a. Coordinated plans depicted in blue (Agent 1) and cream (Agent 2) which happens to be socially optimal.

In the first iteration ($i = 1$), Agent 1 and Agent 2 both assume they can freely use all resources (nodes). Solving a binary linear program they generate their shortest trajectories as $p^1 = (1\,3\,4\,5\,5...)$ and $p^2 = (2\,3\,4\,6\,6...)$, respectively. Detecting a conflict at time step 1 and 2, the agents enter an auction for contested Node 3. Agent 1's estimated "detour cost" for not winning Node 3 (assuming he will be allowed to use all other nodes in consecutive time steps) is 2 which he places as a bid $b^1(i) = 2$. On the other hand, Agent 2's detour cost ist $b^2(i) = l^2(i) - s^2(i) = 12 - 4 = 8$ and hence, she wins the auction. Having lost, Agent 1 adds a constraint to his description of his feasible set (more precisely to R) that from now on prevents it from using Node 3 in time step 1. Replanning results in updated plans $p^1(i+1) = (1\,4\,5\,5\,...)$ and

$p^2(i+1) = (2\,3\,4\,6\,6...)$. Being conflict-free now, these plans can be executed by both agents.

Notice how the laziness of our method protected us from unnecessary computational effort: the initial conflict at time 2 (Node 4) was implicity resolved by the first auction without the need to set up an explicit auction for Node 4 or bidding on all combinations of availability of Nodes 3 and 4.

Ex. B. Of, course, this positive effect of laziness may not always bear fruit - in several situations resolving a collision at one node may not prevent collisions from happening (or, trigger new ones) at other nodes. As an example consider Ex. B in Fig. 6.2(b) and assume Agent 2's initial plan visits Vertex I11 - after this conflict is resolved there will be a second at Vertex I21.

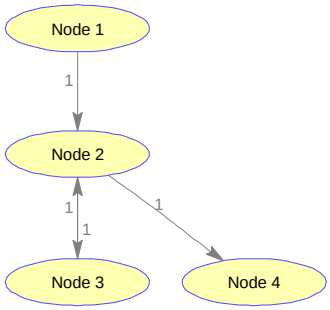
Nonetheless, Ex. A was designed to provide an intuition that it often can lead to favourable coordination outcomes.

Ex. C. In its simple form, the bidding rule AUC is not complete in general. That is, there are problems where, without further modifications, the auction mechanism can fail to generate a feasible solution. An example for this can be generated by modifying the graph planning problem of Ex. 1 as follows: The graph from Ex. A is augmented by adding two nodes Node 7 and Node 8. Node 7 leads into Node 4 which in turn leads into Node 8. The augmented graph is depicted in Fig. 6.2(c). The start and goal nodes of Agents 1 and 2 are as in Ex. A. However, this time, we add a third agent, Agent 3, whose start and goal nodes are Node 7 and Node 8, respectively. Inevitably, the only feasible plan of Agent 3 is $p^3 = (7\,4\,8\,8\,...)$. Going through the iterations as in Ex. A, Agents 1 and Agent 2 enter an auction for Node 3 yielding updated plans $p^1 = (1\,4\,5\,5\,...)$ and $p^2 = (2\,3\,4\,6\,...)$. Of course, this causes Agent 1's plan p^1 to be in conflict with Agent 3's plan p^3 due to the common usage of Node 4 in the second time step. Now we have a problem: Having lost the right to use Node 3 in the second time step 2, Agent 1 cannot deviate from using Node 4 in the second time step. But due to the network topology, Agent 3 has to use Node 4 at the second time step, too. The result is that both agents submit infinite bids, causing the higher-priority agent to get his way and leaving the lower-priority agent with an infeasible planning problem. That is, the conflict cannot be resolved, unless we modify the auction method.

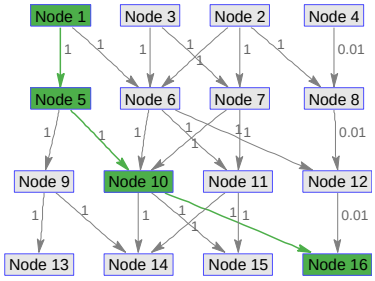
Note, the issue could have been avoided by allowing agents to always have a place to go (e.g. by introducing reflexive edges to all nodes). Alternatively, AUC2 with suitably large w_{confl} (cf. Eq. 6.7) would have avoided the problem since it would have caused Agent 1 to win the first auction against Agent 2.

As a possible modification of AUC1 that might promise a more principled fix to completeness issues than AUC2 does, we may consider to allow the agents to rechallenge other agents for the right to use nodes (i.e. to remove virtual obstacles) that were lost in the past. Testing this idea however will have to be left to future work.

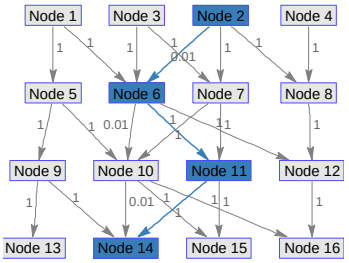
Ex. D. Next we will consider a situation where the solution obtained by the simple bidding rule (AUC)



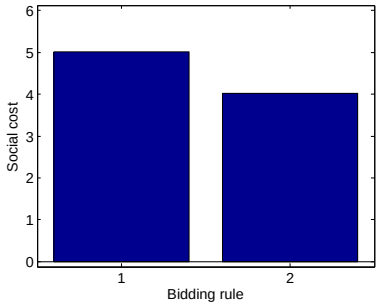
(a) Spatial graph.



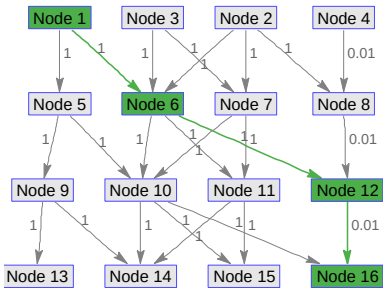
(b) Agent 1's plan in st-graph.



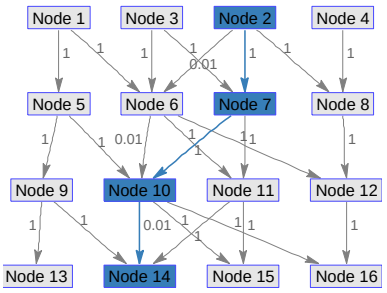
(c) Agent 2's plan in st-graph.



(d) Social cost comparison.



(e) Agent 1's plan in st-graph.



(f) Agent 2's plan in st-graph.

Figure 6.3.: Ex. D. Comparison of the basic bidding rule and the modified one. Note, the node number in the spatio-temporal graph (Fig. 6.3(b), 6.3(c), 6.3(f), 6.3(f)) is $n = 4t + s$, where s is the pertaining spatial node and t is the time step of that node. For instance, consider Fig. 6.3(b). The green path means that agent 1 plans to occupy spatial node 1 at time 0, and again at time 1, node 2 at time 2 and node 4 at time 3. In node sequence notation we can write his plan as $p^2 = (1\,1\,2\,4)$. Figures 6.3(b) and 6.3(c) show the plans resulting from coordiantion with the basic bidding rule (AUC). Figures 6.3(e) and 6.3(e) depict the coordination result utilising the modified bidding rule (AUC2). The latter incurs lower social cost (cf. Fig. 6.3(d)).

as per Eq. 6.6 is sub-optimal and where the modification that penalizes collisions (AUC2, cf. Sec. 6.1.2) fixes the problem. It also illustrates how the auction method can avoid infeasibility by reversing the priorities flexibly in a situation where the FP approach fails to generate a feasible solution.

Consider a trajectory planning problem with time horizon $T = 3$. At initial time $t = 0$, the spatial graph, depicted in Fig. 6.3(a), contains two agents, 1 and 2, at start nodes $S^1 = 1$ and $S^2 = 2$, respectively. The respective goal locations are $D^1 = 4$, $D^2 = 2$. We assume that per time step, each agent incurs a cost of 1 for transitioning between nodes, a cost of 1 for staying at a non-goal node and a cost of 0.01 for staying at her goal node.

In a fixed priority scheme, Agent 2 outranks Agent 1. In that case, Agent 2 is better off not moving out of the way to let Agent 1 pass through to reach her goal. Therefore, with these fixed priorities, Agent 2 blocks the way of Agent 1 and the FP method would fail to generate a feasible plan.

In contrast, our lazy auction method AUC with the basic bidding rule as per Eq. 6.6 manages to invert the priorities in time step $t = 2$ – here, Agent 1 outbids Agent 2 for the right to access spatial node 2 at time step

$t = 2$ in order to be able to reach her goal by terminal time $T = 3$. The result is a collision-free plan (see Fig. 6.3(b), 6.3(c)). However, the myopic nature of the bids results in sub-optimality. Agent 1 had already lost the right to pass through spatial node 2 at time $t = 1$. Had Agent 1 realised that the situation would not change in the next time step, she could have outbid Agent 2 during the first auction. This would have allowed her to arrive at her goal node one time step earlier, saving her 0.99 units of cost. (For Agent 2 it would not have made a difference in the end, since he eventually had to make way regardless without affecting his extra cost of 2* 0.99 for manoeuvring out of the way for one time step.)

In this situation, the modified bid as per Eq. 6.7 helps since it folds in the change of collisions into the magnitude of the bid. This causes Agent 1 to outbid Agent 2 for spatial node 2 already in the first auction and allows agent 1 to pass through spatial node 2 without delay. This results in plans $p^1 = (1\,2\,4\,4)$ and $p^2 = (2\,3\,2\,2)$ which coincide with the socially optimal jointly feasible solution of this planning problem. For its depiction in the spatial-temporal domain refer to Fig. 6.3(e) and Fig. 6.3(f).

6.2.3. Comparisons on randomized graph planning problems

In Sec. 5.3.1, we have discussed several methods for optimal and suboptimal multi-agent planning by optimisation in the presence of hard constraints. Our own methods, CCG, SDCG, AUC and AUC2 aim to alleviate computation via decomposition and lazy constraint generation. While CCG and SDCG are provably complete and optimal, our distributed auction method is not and uses bids as a local heuristic to optimise social cost.

In the tension between optimality and tractability, our methods compromise between the optimal but slow centralised method OPT on the one hand, and the heuristic but decentralised and heuristic method FP on the other. In this subsection we will compare the different methods' performances along the metrics of social cost and relative runtime for the generation of a solution. In addition to the methods discussed so far, we will also examine the use of Max-Sum [96, 208] as a representant of a modern DCOP method.

Comparisons on graph trajectory problems We conducted three experiments on randomized spatial graphs with varying number of agents, start and goal locations, edge costs and topologies. To facilitate trajectory planning, the spatial graphs were converted into spatio-temporal graphs as described above. The performance of the different coordination methods was recorded in terms of runtime and social cost. The results are depicted in Fig. 6.4 and Fig. 6.5. The upper and lower quartiles of the results are also given in Tab. 6.1.

Exp. 1: In the first experiment, we tested the various approaches on 250 randomized graph planning problems with random but non-conflicting start and goal locations in a spatial graph. The number of agents varyied between 2 and 7. Each spatial graph ensured that each node had a connection to two neighbours as well as reflexive edges. Edge transition costs were drawn uniformly at random from the interval [1,100].

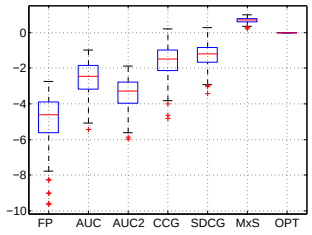
Reflexive edge transitions were set uniformly to a cost of 50. runtime comparisons are depicted in Fig. 6.4(a) and a comparison of the social costs of the different methods are depicted in Fig. 6.5(a). The boxplots depicted in the figures show the median as well as the quartiles and outliers of the algorithm's runtimes relative to the runtime of the centralised solution. Because statistics of runtime ratios were computed, the quantities are dimensionless. All algorithms were implemented in Matlab R2013a, with *bintprog* as the optimiser. (Bintprog essentially is a combination of branch-and-bound and a linear-programming solver similar to the approach described in [30]) As an exception, the implementation of Max-Sum called a Java sub-routine from Matlab.

As expected, FP was the fastest method of all. Unfortunately, FP failed to produce feasible solutions on 11 % of the graph planning problems. In contrast, our auction methods always produced feasible solutions. Furthermore, they outperformed FP in terms of social cost while requiring only a fraction of the planning runtime. AUC is the standard auction bidding rule, where AUC2 denotes the modified bidding rule as per Eq. 6.7. As we had hoped when conceiving the modified bids of AUC2, the additional computation entailed by folding in the collision count heuristic to steer the plans away from conflicts evidently paid off. That is, AUC2 sometimes required fewer coordination rounds, which in turn shows in improved runtime performance in comparison to the basic bidding rule (AUC). Furthermore, it is noteworthy that all heuristic methods exhibited faster runtimes than the optimal methods OPT, CCG, SDCG.

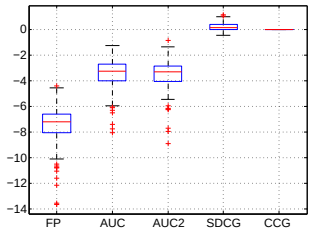
The latter two managed to solve the problems faster than the centralised approach (OPT). Comparing CCG and SDCG in terms of runtime it appears at first glance, that the dimensionality reduction SDCG attempts has not paid off on the relatively low-dimensional tests. However, it should be noted that the simulations were run sequentially. If run on a parallel architecture in large graphs with relatively sparse interaction, we would expect SDCG to outperform CCG due to the decomposition. In such situations, we would also expect our auction methods to improve their relative performance both in comparison to the optimal methods as well as to FP.

Similarly, Max-Sum should improve its runtime performance on a parallel architecture as it was designed as a distributed message-passing approach. However, as can be seen from Fig. 6.5(a), the DCOP method often failed to produce feasible solutions (indicated by super-optimal social cost) and when a feasible solution was found, it was not always optimal. Owing to this poor performance both in terms of solution quality and runtime performance, we have excluded the method from subsequent investigation.

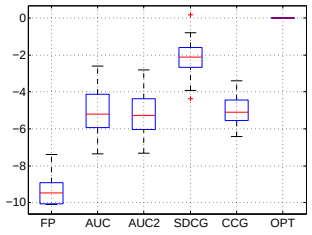
Exp. 2: In this experiment, we only tested the faster methods FP, AUC, AUC2, CCG and SDCG on 150 randomised graph planning problems with 3-10 agents. The spatial graphs contained nodes with random connections to 2-3 other nodes. This time, we assigned a maximum cost of 100 to reflexive edges. In the absence of OPT, the relative runtimes depicted in Fig. 6.4(b) were computed relative to the runtime of



(a) Exp.1: runtime ratios relative to OPT (log-scale).



(b) Exp.2: runtime ratios relative to CCG (log-scale).

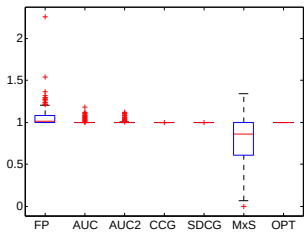


(c) Exp.3: runtime ratios relative to OPT (log-scale).

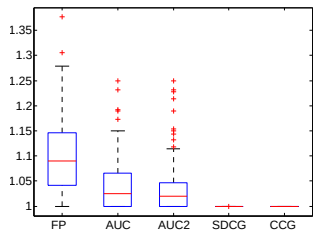
Figure 6.4.: Comparison of coordination methods on randomised graph planning problems. Compared are relative runtimes (on a log-scale) incurred by the different methods to deliver a solution. For example, consider Fig. 6.4(a). Here, the ith data point in the third boxplot was computed as $\log(r(AUC2)/r(OPT))$ where $r(AUC2), r(OPT)$ are the runtimes of algorithms AUC2, OPT for solving the ith planning problem, respectively. In Fig. 6.4(b), the runtime ratios are computed relative to CCG rather than OPT. For example, each data point of the first boxplot represents $\log(r(FP)/r(CCG))$ for the pertaining randomised graph planning problem.

CCG. We found that for a larger number of agents, FP was more likely to fail finding a feasible solution. In total FP produced infeasible solutions on 14 % of the problem instances. By contrast, all other methods always achieved coordination. Overall the relative performances of the algorithms were quite similar to those recorded in the first experiment.

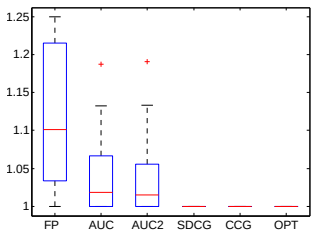
Exp. 3: In the previous experiments, Matlab's bintprog was utilised to solve the optimisation problems. One advantage of this solver is that it is easy to understand and not proprietory. Therefore, computation of different methods that utilise this solver as a subroutine can be compared fairly and without unknown influences from a sophisticated heuristic internally employed by the solver. However, in practice CPLEX [119] is a more widely employed comercially available optimisation package. Therefore, in the third experiment we tested performance of our methods when using CPLEX instead of Matlab's binary programming solver on 100 randomised graph planning problems with 1-10 agents. The runtime results are depicted in Fig. 6.4(c). Interestingly, CCG seems to connect particularly well with CPLEX, exhibiting runtime competetive with that of the auction-based approaches. However, this comparison was not entirely fair: the CCG implementation was based on a static object that maintained the solver from the previous iteration during iterative replanning. In contrast, the other methods' implementations conducted replanning of the (updated) problems largely from scratch in each iteration. We suspect that CPLEX takes advantage of solutions from previous runs thereby accelerating CCG's runtime disproportionally to the other methods. In future work, we would like to investigate this further. We believe by implementing the auction methods with CPLEX in a similar manner to CCG (e.g. by maintaining the solver object in memory between plan updates), we can also speed up these competing methods further in a similar manner to the speedup observed for CCG.



(a) Exp.1: soc. cost relative to OPT.



(b) Exp.2: soc. cost relative to CCG.



(c) Exp.3: soc. cost relative to OPT.

Figure 6.5.: Comparison of coordination methods on randomised graph planning problems with regard to social costs incurred by the different methods. For example, consider Fig. 6.5(a). Here, the ith data point in the third boxplot was computed as $s(AUC2)/s(OPT)$ where $s(AUC2), s(OPT)$ are the social costs of incurred by the solutions for ith planning problem generated by AUC2, OPT, respectively. In Fig. 6.5(b), these ratios are computed relative to CCG rather than OPT.

	Relative social cost (75%, 25 % quartiles)			*Relative runtime (75%, 25 % quartiles)*		
Experiment:	Exp. 1	Exp. 2	Exp. 3	Exp. 1	Exp. 2	Exp. 3
FP	1.08, 1	1.15, 1.04	1.22, 1.04	0.02, 0.003	0.0013, 0.0003	0.00013, 0.00004
AUC	1, 1	1.07,1	1.07, 1	0.15, 0.04	0.065, 0.017	0.016, 0.0026
AUC2	1, 1	1.04, 1	1.06, 1	0.06, 0.02	0.055, 0.017	0.0126, 0.0024
CCG	1, 1	1, 1	1, 1	0.38, 0.12	1, 1	0.013, 0.0038
SDCG	1, 1,1	1, 1	1, 1	0.44, 0.18	1.49, 1.007	0.203, 0.068
MxS	1, 0.6	-	-	2.2, 1.85	-	-
OPT	1, 1	-	1, 1	1, 1	-	1, 1

Table 6.1.: Upper and lower quartiles of the performance measures for the different methods across the randomised planning problems.

Comparison of social cost in multi-commodity flow problems in simple randomized forwardly directed graphs. In the trajectory planning problem, FP could become infeasible due to a lack of a sufficient number of paths to the goal if all trajectories were blocked by higher ranking agents. When comparing social cost, those planning problems were effectively removed from the comparison of social cost. In this subsection, we discuss simulations that investigated social cost on simpler multi-commodity flow problems on graphs of fixed topological structure (but with randomized edge cost) where feasibility could always be guaranteed. We compared social cost conducted on 10000 randomized graph planning problems in the multi-commodity flow setup. In each randomized trial, the planning environment was a forward directed graph similar in structure to the one in Fig. 6.2(b) (b). Each graph had a random number of vertices ($L \times N$ - graphs where number of layers $L \sim \text{Unif}(\{3, ..., 11\})$, number of nodes per layer $N \sim \text{Unif}(\{3, ..., 11\}$) and randomized vertex-transition costs drawn from $\text{Unif}([1, ..., 200])$. The coordination task was to have each agent find a cost optimal trajectory through the randomized graph where the agents had a randomized start location in the first layer and a destination vertex in the last layer.

By design, the problems were chosen to be simple enough such that an optimal method could solve them in reasonable amount of time and such that no problems had to be discarded due to FP not being able to find

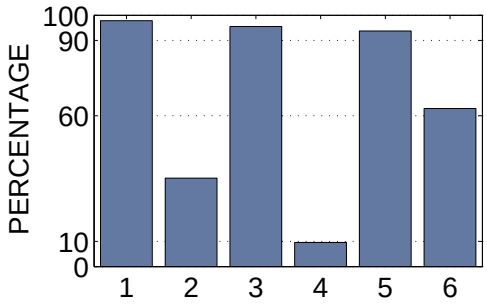


Figure 6.6.: Results of comparison between different methods over 10000 randomized problem instances. The bars represent the percentages of the trials where... 1: $s(AUC) \leq s(FP)$ 2: $s(AUC) < s(FP)$ 3: $s(AUC) \leq s(BEST-FP)$ 4: $s(AUC) < s(BEST-FP)$ 5: $s(AUC) = s(OPT)$ 6: $s(FP) = s(OPT)$.

a feasible solution.

Discarding Max-Sum and knowing that CCG, SDCG and OPT all are guaranteed to produce socially optimal solutions, for each trial we compared the social costs only on a subset of our coordination methods. In addition to the trajectory planning simulations, we also included a comparison with $FP - BEST$. This method was exhaustively computing the best solution that could be obtained by the best fixed ranking of the FP method. It serves as a best case for methods that search for a fixed ranking with low social cost (e.g. [27]).

The results are depicted in Fig. 6.6.

For a given problem instance, let $s(AUC), s(FP), s(OPT)$ denote the social cost of the coordinated plan generated by our method, the fixed priority methods FP, and the optimal social cost, respectively. Finally, let $s(BEST - FP)$ be the social cost the plans of the fixed priority method with the best choice of priorities in hindsight would have incurred.

The data show that our auction method performed optimally on 93 % of the problems (5) while the fixed priority method did so on only 62.2 %. Conversely, the fixed priority method outperformed our method only on 2.1 % (see bar (1)) while it was strictly outperformed on 35.1 % of the randomized trials (2).

6.2.4. Modified bids and termination guarantee

Many iterative coordination methods suffer from the limitation that they cannot be guaranteed to terminate in a finite number of coordination steps which can for instance happen due to running into deadlocks or cycles.

In this section, we will present a slight modification of the auction-based method and discuss conditions under which it is guaranteed to terminate with a valid coordination solution when applied in environments with a finite number of resources (nodes in a spatial graph) but where we do allow an infinite planning horizon.

In lieu to the graph planning problems introduced in the previous section, we assume that the planning environment is a graph $G = (V, E)$. However, in order to prove termination of our mechanism we make

some additional assumptions. **In particular, we assume** G is connected, finite, edge relation E is reflexive, the edge costs are uniformly 1, $\forall v \in V : deg(v) \geq |\mathcal{A}|$, i.e. each vertex has at least as many children as there are agents. Finally, we assume that an agent that reaches its goal node is effectively removed from the system in subsequent time steps, i.e. seizes claiming any further resources (nodes) other agents might want to visit after it has reached its goal.

In the modified version of our mechanism bidding for resources (nodes) will usually proceed as the standard bidding method (AUC) does. As a new exception to the normal rule, whenever an agent $\mathfrak{r}$ bids for the right to visit a node $v \in V$ in time step t but has lost the right to use some node for some later time step $t + \tau$ ($\tau \geq 1$) or has lost $v \in V$ for time step t to some other agent in a previous coordination iteration (who may have released it again in the mean time) then $\mathfrak{r}$ will submit the modified bid: $b^{\mathfrak{r}}(i) = 1 - \delta\big(l^{\mathfrak{r}}(i) - s^{\mathfrak{r}}(i)\big)$ where $\delta : \mathbb{R} \to \{0, 1\}$ denotes the Kronecker delta function and l, s are defined as in Eq. 6.6.

Theorem 6.2.2 (Termination Guarantee)**.** *Under the assumptions listed above, our modified auction mechanism generates a joint plan in a finite number of coordination iterations that is legal (i.e. two consecutive nodes in the plan sequence of nodes are always connected by an edge in E), collision-free and allows all agents to reach their destinations, regardless of the (mutually exclusive) start and goal nodes.*

The proof is lengthy and rather technical. The interested reader can find it in Appendix B.

6.2.5. Discussion

In this section, we have tested our coordination methods on randomised graph planning problems. When considering the quality of the joint solution found by the different algorithms, we see that Max-Sum frequently did not manage to find the optimal or feasible solutions to the collision avoidance problem. As predicted the FP method was the fastest method but also the least optimal one. Furthermore, it did not manage to produce a feasible solution on a significant proportion of the problem instances. By contrast, the auction -based methods provided a compromise between social cost and fast runtime performance and managed to avoid infeasibility. In situations where optimality matters, CCG and SDCG have proved to be methods that combine optimality with improved runtime performance. While CCG seemed to outperform SDCG (probably due to the additional overhead for merging the sub-problems), we would expect SDCG to outperform CCG in situations where a large number of agents have very few interactions in large graphs and when computation is performed on a parallel architecture. We would like to investigate this further in future work. All of our methods were employing a binary integer solver as a black-box sub-routine. Our experiments suggest that tailoring implementation details towards the underlying solver can make a noticeable difference. From a practical perspective, exploring this interplay between method and available solver further might also be worthwhile if a coordination method is to be deployed in practical applications with a large number of agents.

6.3. Multi-agent SMPC with probabilistic collision-avoidance constraints

Within the field of model-predictive control (MPC), multi-agent motion planning and control problems in continuous state space have been addressed with mixed-integer linear programming (MILP) techniques [222].

We stated our coordination methods in the framework of constrained optimisation. The generality of this framework renders our methods applicable to a large number of planning contexts. One application that can be stated very naturally as an optimisation problem is *stochastic model-predictive control* (SMPC). Remember from Sec. 2.2 that in SMPC, at each stage of the repeated planning process one desires to solve an optimisation problem as per Eq. 2.4.

We will now translate this control problem into our multi-agent optimisation framework introduced in Sec. 6.1. This will make it evident how our coordination methods can be employed. With notation as before, consider a multi-agent SMPC problem with agents $\mathfrak{a} \in \mathfrak{A} = \{1, \dots, A\}$. In the SMPC setting, each agent's plan $p^{\mathfrak{a}} = u^{\mathfrak{a}}_{0:T-1}$ is a sequence $u^{\mathfrak{a}} := u_{0:T-1} = \left(u^{\mathfrak{a}}_t\right)_{t\in\mathcal{I}_t}$ ($\mathcal{I} = \{0, \dots, T-1\}$) of control inputs up to time horizon T. The plant dynamics of agent $\mathfrak{a}$ are given by the transition function $\psi^{\mathfrak{a}}$ connecting the states $x^{\mathfrak{a}}_t$ via the recursion $x^{\mathfrak{a}}_{t+1} = \psi^{\mathfrak{a}}\big(t, x^{\mathfrak{a}}_t, u^{\mathfrak{a}}_t, \nu_t\big)$. Typically, ψ is a known linear function and ν an o.i. process. In addition, the agents may be confined to some free-space $\mathfrak{F}$ which the state has to be contained in with a given probability of at least $\tilde{\delta} \in (0, 1)$. Probability bound $\tilde{\delta}$ is an objective parameter defining risk-averseness and is specified in advance.

Consequently, in our notation of Sec. 2.2, the locally feasible set of agent $\mathfrak{a}$ is

$$F^{\mathfrak{a}} = \{u^{\mathfrak{a}}_{0:T-1} \in \mathcal{U} \,|\, \forall t : x^{\mathfrak{a}}_{t+1} = \psi^{\mathfrak{a}}\big(t, x^{\mathfrak{a}}_t, u^{\mathfrak{a}}_{0:T-1}, \nu_t\big), \Pr[x^{\mathfrak{a}}_t \notin \mathfrak{F}] < \tilde{\delta}\} \tag{6.8}$$

where $\mathcal{U}$ is a set of admissible controls. For instance, in bounded control, one might have $\mathcal{U} = \{u_{0:T-1} | u_t \in [0, \bar{u}], \forall t\}$ for some control bound $\bar{u}$.

Existing work (e.g. [37–39]) has shown how to state the single-agent stage optimisation problem

$$p^{\mathfrak{a}} \leftarrow \min_{p^{\mathfrak{a}} \in F^{\mathfrak{a}}} c^{\mathfrak{a}}(p^{\mathfrak{a}}). \tag{6.9}$$

as a mixed-integer linear programme under the assumption of linear dynamics, a piece-wise linear cost function $c^{\mathfrak{a}}$ (e.g. a 1-norm) and free-space $\mathfrak{F}$ being the complement of $k \in \mathbb{N}$ polygonal obstacles $O_i = \{x \in \mathcal{X} | \langle x, n_{i1}\rangle \leq b_{i1}, \dots, \langle x, n_{ik}\rangle \leq b_{ik}\}$ $(i = 1, ..., k)$ (cf. Sec . 2.2).

To convert the probabilistic (chance-) constraint $\Pr[x^{\mathfrak{a}}_t \notin \mathfrak{F}] < \tilde{\delta}$ into a deterministic one, previous work either assumed bounded-support uncertainty [205, 206, 251] in the framework of stochastic or robust control, or, Gaussian uncertainty was assumed [37]. Later work yielded distribution independence by approximating

the chance constraint with a particle method [38]. However, no consideration was given to the impact of the number of samples on the approximation quality of the chance constraint.

Our own work, which part of this section is based on, has focussed on multi-agent collision-avoidance MPC utilising particle methods [48] as well as distribution-independent tail bounds [158, 159].

In this section, we bring to bear our coordination methods in multi-agent collision avoidance problems where the agents' stage optimisation problems are coupled via probabilistic collision avoidance constraints of the form

$$\forall t \in \mathcal{I}_t : \Pr[\mathfrak{C}^{\mathfrak{a},\mathfrak{r}}(t)] < \delta_t^{\mathfrak{a},\mathfrak{r}} \tag{6.10}$$

pertaining to the instantaneous collision probabilities or

$$\Pr[\cup_{t\in\mathcal{I}_t}\mathfrak{C}^{\mathfrak{a},\mathfrak{r}}(t)] < \delta^{\mathfrak{a},\mathfrak{r}} \tag{6.11}$$

bounding the collision probability of the entire trajectories.

Here, the event $\mathfrak{C}^{\mathfrak{a},\mathfrak{r}}(t)$ denotes a collision between the agents $\mathfrak{a},\mathfrak{r}$ at time step t. Therefore, the coupling constraints are to bound the probability of a collision occurring within the set of times $\mathcal{I}_t$.

Note, the trajectory constraint in Eq. 6.11 is the more stringent one. That is, if it holds then automatically the instantaneous constraint holds with the same bound $\delta_t^{\mathfrak{a},\mathfrak{r}} = \delta^{\mathfrak{a},\mathfrak{r}}$. Conversely, in finite-time horizon problems, $|\mathcal{I}_t| < \infty$, the trajectory collision constraint as per Eq. 6.11 can be enforced by a set of instantaneous constraints as per Eq. 6.10. For instance, we can always harness the union bound giving

$$\Pr[\cup_{t\in\mathcal{I}_t}\mathfrak{C}^{\mathfrak{a},\mathfrak{r}}(t)] \leq \sum_{t\in\mathcal{I}_t}\mathfrak{C}^{\mathfrak{a},\mathfrak{r}}(t). \tag{6.12}$$

Hence, by choosing $\delta_t^{\mathfrak{a},\mathfrak{r}} := \delta^{\mathfrak{a},\mathfrak{r}}/\left|\mathcal{I}_t\right|$, the constraint as per Eq. 6.10 conservatively ensures that the trajectory constraint holds. That is, any trajectory satisfying the former is guaranteed to satisfy the latter.

As in Ch. 5, we formally define the collision event as $\mathfrak{C}^{\mathfrak{a},\mathfrak{r}}(t) = \{(s^{\mathfrak{a}}(t), s^{\mathfrak{r}}(t))|\, \|s^{\mathfrak{a}}(t) - s^{\mathfrak{r}}(t)\|_2 \leq \frac{\Lambda^{\mathfrak{a}}+\Lambda^{\mathfrak{r}}}{2}\}$. Here, $\Lambda^{\mathfrak{a}}, \Lambda^{\mathfrak{r}}$ denote the agents' plant diameters, and $s^{\mathfrak{a}}, s^{\mathfrak{r}} : \mathcal{I}_t \to \mathbb{R}^D$ are two (possibly uncertain) trajectories in a common *interaction* space. The latter is a transformation of the state space of the underlying plant dynamics where conflicts can arise. That is, there exists a transformation $\mathcal{T} : \mathcal{X} \to \mathcal{S}$ from state space $\mathcal{X}$ to interaction space $\mathcal{S}$ such that $s = \mathcal{T}x, \forall x \in \mathcal{X}, s \in \mathcal{S}$.

In the multi-robot coordination scenarios we will consider, interaction space simply is the set of spatial locations where agents can collide. Therefore, it will be a strict sub-space of state-space (in mechanical systems, states not just contain positions but also include velocities).

In this setting, the resulting optimal centralised multi-agent planning problem (cf. Eq. 6.3) is:

$$p_{\text{opt}} \leftarrow \min_{p \in F_L \cap F_{\mathcal{C}}} \sum_{\mathfrak{a} \in \mathfrak{A}} c^{\mathfrak{a}}(p^{\mathfrak{a}}). \tag{6.13}$$

where $F_{\mathcal{C}} := \{(p^1, \ldots, p^A) \in \mathcal{U}^1 \times \ldots \times \mathcal{U}^A | \Pr[\cup_{t \in \mathcal{I}_t} \mathfrak{C}^{\mathfrak{a},\mathfrak{r}}(t)] < \delta^{\mathfrak{a},\mathfrak{r}}, \forall \mathfrak{a}, \mathfrak{r} \in \mathfrak{A}\}$ enforces collision avoidance between the agents and $F_L = F^1 \times \ldots \times F^A$, where

$F^{\mathfrak{a}} = \{u^{\mathfrak{a}}_{0:T-1} \in \mathcal{U} \,|\, \forall t : x^{\mathfrak{a}}_{t+1} = \psi^{\mathfrak{a}}\big(t, x^{\mathfrak{a}}_t, u^{\mathfrak{a}}_{0:T-1}, \nu_t\big), \Pr[x^{\mathfrak{a}}_t \notin \mathfrak{F}] < \tilde{\delta}\}$ ensures local feasibility.

In addition to exhibiting coordination with sampling methods, we focus on the development of conservative deterministic constraints that are based in tail inequalities. As we explained in [159], the tail bounds have computational advantages over particle-based collision probability approximations in multi-agent settings: in the sampling-based approach the number of particles scales poorly in the number of agents (if centralised approaches were utilised) and can cause computational blow-up since the optimisation problem's complexity scales poorly with the number of particles. While in our earlier work [158, 159] the tail inequalities restricted applicability to two-dimensional interaction space, in this section, we can harness our results from Ch. 5 to derive conservative approximations that work in multiple dimensions by converting our criterion functions into mixed-integer constraints. Since we derived both distribution-independent criterion functions and criterion functions based on (sub-) Gaussian tail inequalities, we can choose to either work without any distributional assumptions or to benefit from reduced conservatism if we know that our uncertainties follow (sub-) Gaussian distributions.

The remainder of this section is structured as follows. Firstly, we will sketch the particle-based collision avoidance method. We will then briefly rehearse the computational issues with its application to the multi-agent setting. Keeping the individual agents' problems decoupled at all times, employing our auction-based method turns out to alleviate the issue. We describe how our lazy auction-methods can be brought to bear in the SMPC collision avoidance problem and provide simulations illustrating the viability of this approach. Subsequently, we discuss how the probabilistic coupling constraints can be replaced by deterministic ones by virtue of converting our criterion functions back into constraints based on tail inequalities rather than on sample approximations. Our simulations of this approach in a multi-robot collision avoidance scenario illustrate that the method even is tractable without the auction method and provides one example where leveraging Gaussianity reduces conservatism and thereby planning time and social cost.

6.3.1. Lazy auctions in distributed collision avoidance with sampling-based SMPC

Preliminaries- Sampling-based SMPC with probabilistic obstacle avoidance

Multi-agent motion planning and control problems in continuous state space have been addressed with mixed-integer linear programming (MILP) techniques [222]. Typically they rely on time-discretization only, without

prior space-discretization. If being able to cope with uncertainty, they typically are solve with a centralized planner, are confined to particular distributional assumptions or assume interval-bounded uncertainty and cannot cope with non-convexities.

Recently, stochastic MPC methods have been suggested for single-agent planning that accommodate for uncertainty in the effect of control signals to avoid collisions. For instance, Blackmore et. al. [38] discuss a particle-based MPC method that can be used to generate a low-cost trajectory for a vehicle that avoids obstacles with adjustably high confidence. Being a sampling-based method the guarantees hold in the limit of large samples and do not have to rely on distributional assumptions. In their model, the plans $p^{\mathfrak{a}}$ are time-discrete sequences of control inputs. The spatial location $x_t^{\mathfrak{a}}$ of agent $\mathfrak{a}$ at time t is assumed to be a linear function of all previous control inputs plus some random perturbations $\nu_0, ..., \nu_{t-1} \sim \mathcal{D}$. So, given plan $p^{\mathfrak{a}}$, drawing n samples of perturbations for all time steps generates N possible sequences of locations (*particles*) $(x_t^{\mathfrak{a},(j)})_t$ $(j = 1, ..., N)$ agent $\mathfrak{a}$ could end up in when executing his plan.

Note, the sample of the uncertain state trajectory gives rise to a sample of the interaction space trajectory. That is, from $(x_t^{\mathfrak{a},(j)})_t$ we can obtain $(s_t^{\mathfrak{a},(j)})_t$ where $s_t^{\mathfrak{a},(j)} = \mathcal{T} x_t^{\mathfrak{a},(j)}$.

Formally, the uncertain dynamics model is encoded by $x_t^{\mathfrak{a},(j)} = f_{t-1}(x_0^{\mathfrak{a},(j)}, u_0^{\mathfrak{a}}, ..., u_{t-1}^{\mathfrak{a}}, \nu_0^{(j)}, ..., \nu_{t-1}^{(j)})$ $(j = 1, ..., N)$ where f_{t-1} typically is a linear function and $u_0^{\mathfrak{a}}, ..., u_{t-1}^{\mathfrak{a}}$ is a sequence of control inputs as specified by agent $\mathfrak{a}$'s plan. Due to this direct functional relationship we can constrain agent $\mathfrak{a}$'s MILP's search for optimal control inputs by adding constraints on the particles.

Let T be the number of time steps given by the time horizon and temporal resolution. That is, $t \in \mathcal{I}_t = \{1, ..., T\}$. Furthermore, let $\mathfrak{F}$ be the free-space, i.e. the set of all locations that do not belong to an obstacle. Obstacle avoidance is realized by specifying a chance constraint $\Pr((x_t^{\mathfrak{a}})_{t \in \mathcal{I}_t} \notin \mathfrak{F}) \leq \tilde{\delta}$ on the actual location of the agent. For practical purposes, $\Pr((x_t^{\mathfrak{a}})_{t \in \mathcal{I}_t} \notin \mathfrak{F})$ is estimated by sampling which leads to the approximated chance constraint $\frac{1}{N} |\{j = 1, ..., N : (x_t^{\mathfrak{a},(j)})_{t \in \mathcal{I}_t} \notin \mathfrak{F}\}| \leq \tilde{\delta}$ which is added to agent $\mathfrak{a}$'s individual MILP [38].

To particle-approximate the coupling constraints as per Eq. 6.11, note that the probability of a collision between agent $\mathfrak{a}$ and agent $\mathfrak{r}$ at time step t is

$$\Pr(\|s_t^{\mathfrak{a}} - s_t^{\mathfrak{r}}\| < \Lambda^{\mathfrak{a},\mathfrak{r}}) = \mathbb{E}_{s_t^{\mathfrak{a}},\, s_t^{\mathfrak{r}}}\{\mathbf{1}_{\mathfrak{C}(t)}\} = \int\int \mathbf{1}_{\mathfrak{C}^{\mathfrak{a},\mathfrak{r}}(t)}(s_t^{\mathfrak{a}}, s_t^{\mathfrak{r}})\, \mathrm{d}\mathbb{P}^{\mathfrak{a}}(s_t^{\mathfrak{a}})\, \mathrm{d}\mathbb{P}^{\mathfrak{r}}(s_t^{\mathfrak{r}}) \tag{6.14}$$

$$\approx \frac{1}{N^2} \sum_{k=1}^{N} \sum_{j=1}^{N} \mathbf{1}_{\mathfrak{C}^{\mathfrak{a},\mathfrak{r}}(t)}(s_t^{\mathfrak{a},(k)}, s_t^{\mathfrak{r},(j)}) \tag{6.15}$$

where $\mathbb{P}^{\mathfrak{a}}$ and $\mathbb{P}^{\mathfrak{r}}$ are the probability measures representing the uncertainty regarding agent $\mathfrak{a}$'s and agent $\mathfrak{r}$'s locations respectively (given the histories of their control inputs) and where $\mathbf{1}_{\mathfrak{C}(t)}(s_t^{\mathfrak{a}}, s_t^{\mathfrak{r}}) := \begin{cases} 1 & \text{, for } \|s_t^{\mathfrak{a}} - s_t^{\mathfrak{r}}\| < \Lambda^{\mathfrak{a},\mathfrak{r}} \\ 0 & \text{, otherwise.} \end{cases}$

The probability can be approximated with arbitrary accuracy in the limit of an infinite number of samples ($N \to \infty$). Finite sample-bounds could be attempted based on quadrature bounds. However, such considerations are beyond the scope of this work. (Instead, below, we will provide some remarks on utilising sample-based tail inequalities to obtain finite-sample bounds on the collision probabilities.)

As we explain in detail in [159], we could utilise this particle representation of the collision probability to enforce collision avoidance in a MIP representation of centralised OP 6.13. However, as pointed out in [158,159], the statement as a set of MIP constraints requires the introduction of $\mathcal{O}(N^2A^2TD)$ extra binary variables to the centralised MIP if all collisions between all agents are to be avoided. Since the worst-case complexity of MIP scales exponentially in the number of binary variables, unfortunately, this approach can render the optimisation problem intractable if N is set to high values to ensure good approximation quality of the probabilistic constraint or if the number of agents is large. As we will see below, the auction-based method can alleviate some of these issues by replacing these coupling constraints with constraints for probabilistic avoidance of virtual obstacles in order to achieve coordination. The obstacle avoidance constraints follow [38] and only require $\mathcal{O}(N)$ binary constraints per virtual obstacle. Nonetheless, in order to determine whether coordination is required at a time step, the collision avoidance constraint still has to be checked. In this subsection, this is done on the basis of Eq. 6.15. An alternative would be to do collision detection on the basis of the criterion functions developed in Ch. 5.

Employing lazy auctions for the coordination of stochastic collision avoidance

In what is to follow, we will describe how to employ our auction mechanism to achieve probabilistic collision-avoidance: Each agent solves its local MILP to find a plan that corresponds to sequences of n particle trajectories. When two (or more) agents $a, r, ..$ detect their particle clusters $\{s_t^{\mathfrak{a},(1)}, ..., s_t^{\mathfrak{a},(N)}\}$, $\{s_t^{\mathfrak{r},(1)}, ..., s_t^{\mathfrak{r},(N)}\}$,.. 'get too close', they suspect a conflict and participate in an auction. The winner gets to use the contested region, while the losers receive constraints that correspond to a *virtual obstacle* (that is valid for time step t) and re-plan. Notice, for notational convenience, we omit the explicit mention of the coordination iteration in our notation throughout the rest of the section.

Next, we will explain the application of our the auction-based coordination method to the trajectory planning problem in greater detail. Every agent a draws disturbance sample (particles) and uses them to solve an approximated version of OP 6.9 to generate an approximately feasible and optimal plan. As explained above, the mechanism requires the agents to exchange their plans in every coordination iteration. However, assuming they know each other's dynamics, the agents do not need to exchange all particles constituting their trajectories – it suffices only to exchange the optimal control inputs that lead to the particle trajectories (alongside the state or seed of their own pseudo-random-generator with which they drew their disturbance

parameters).

With this knowledge, all the other agents are able to exactly reconstruct each others' particle trajectories. Now each agent locally carries out a test for collision by calculating the probability of a collision for each plan of every other agent.

Let $\{s_t^{\mathfrak{a},(1)}, \ldots, s_t^{\mathfrak{a},(N)}\}$ be the particle cluster that is a sample of the belief over agent $\mathfrak{a}$'s position at time step t. Furthermore, let $\{s_t^{\mathfrak{r},(1)}, \ldots, s_t^{\mathfrak{r},(N)}\}$ be the particle cluster of agent $\mathfrak{r}$.

The probability of a collision of agent $\mathfrak{a}$ and agent $\mathfrak{r}$ at time step t is approximated by their respective particle representations as per Eq. 6.15. So, collision detection could be done by checking whether these approximated probabilities are above a predefined threshold $\delta_t^{\mathfrak{a},\mathfrak{r}}$. As a computationally efficient alternative, one could construct a conservative collision criterion function and use it for collision detection (see Sec. 5.2). (Note that depending on the bounds used to construct the criterion function, utilising the latter may result in an unnecessarily high false-alarm rate. In that case it may be advisable to use the criterion function as a first filter and only check for collisions via Eq. 6.15 at time steps that are flagged as collision candidates by the criterion function approach.)

If a time step with a collision probability above the tolerated threshold is found, the agents engage in an auction for the contested spatial resource at that time, as described in previous sections. The resource in this case corresponds to the right to pass through. We propose its denial to be embodied by a new *virtual obstacle* the loser of the auction, say agent $\mathfrak{r}$, will have to avoid (but only at time t). By placing the virtual obstacle around the winner's location sample mean estimate at time step t, we limit the chance of a collision. We represent the new obstacle by a hyper-square $B_{\varrho+\epsilon}(\bar{s}_t^{\mathfrak{a}})$ with side length $\varrho + \epsilon$ and centred at the mean $\bar{x}_t^{\mathfrak{a}}$ of agent $\mathfrak{a}$ at time step t. This choice is made for simplicity of exposition. More generally (and less conservatively) one could place any polygonal obstacle of sufficient size around the mean. The limitation to polygonal obstacles is motivated by the possibility to connect to existing work for chance-constrained obstacles avoidance with MILP [38], for which the authors derive the appropriate integer constraints aiming to guarantee that chance constraint $\Pr[x_t^{\mathfrak{a}} \notin \mathfrak{F}] < \tilde{\delta}$ is satisfied.

Determination of virtual obstacles. Before one can introduce the avoidance constraint $\Pr[p_t^{\mathfrak{a}} \in \bar{V}_t^{\mathfrak{a}}] < \delta_t^{\mathfrak{a},\mathfrak{r}}$ the virtual obstacle $\bar{V}_t^{\mathfrak{a}}$ has to be defined. While there are many ways of defining such a suitable set, we will limit our exposition to box-shaped virtual obstacles centred at the other agent's nominal position at the time of conflict. The question arises how large to set its radius (or edge length).

Obviously, the larger the virtual obstacle, the lower the probability of a collision between the agents. On the other hand, an overly large additional obstacle shrinks the free-space and may unsuitably increase trajectory costs or even lead to deadlocks. Next, we will derive coarse mathematical guidelines for how to set the size of the virtual obstacle in order to avoid a collision with a predefined probability.

Let t be a fixed time step. Omitting the time index, $\mathfrak{C} := \{(s_t^{\mathfrak{a}}, s_t^{\mathfrak{r}}) | (\|s_t^{\mathfrak{a}} - s_t^{\mathfrak{r}}\| < \Lambda^{\mathfrak{a},\mathfrak{r}})\}$ is the event of a collision at time t. Let $E := \{(s_t^{\mathfrak{a}}, s_t^{\mathfrak{r}}) | \|s_t^{\mathfrak{a}} - \bar{s}_t^{\mathfrak{a}}\|_2 \leq \varrho\}$ be the event that the true position of agent $\mathfrak{a}$ at time step t deviates no more than ϱ from the mean of its position given by $\bar{s}_t^{\mathfrak{a}}$. By introducing a chance constraint with threshold $\frac{\delta}{2}$,

$$\Pr[s_t^{\mathfrak{r}} \in B_{\epsilon+\varrho}(\bar{s}^{\mathfrak{a}})] < \frac{\delta}{2} \tag{6.16}$$

we enforce a bound on the collision probability. Introduction of the virtual obstacle to agent $\mathfrak{r}$'s constraints induces his planner to adjust the control inputs such that the fraction of particles $(s_t^{\mathfrak{r},(j)})_{i=1,\dots,N}$ that are inside the square box $B_{\epsilon+\varrho}(\bar{s}^{\mathfrak{a}})$ with edge length $\varrho + \epsilon$ around mean $\bar{s}_t^{\mathfrak{a}}$ is bounded (and by particle approximation of the chance constraint, hence also the (approximated) probability that agent $\mathfrak{r}$ is inside the box). Parameter ϱ needs to be specified after the desired δ is defined and we will now discuss a proposal how this can be done.

Let K be the event $\{(s_t^{\mathfrak{a}}, s_t^{\mathfrak{r}}) | s_t^{\mathfrak{r}} \in B_{\epsilon+\varrho}(\bar{s}^{\mathfrak{a}}))\}$ that agent r is inside the box around $\bar{s}^{\mathfrak{a}}$.

We have $\Pr(\mathfrak{C}) = \Pr(\mathfrak{C} \cap E) + \Pr(\mathfrak{C} \cap \neg E) = \Pr(\mathfrak{C} \cap E \cap K) + \Pr(\mathfrak{C} \cap E \cap \neg K) + \Pr(\mathfrak{C} \cap \neg E) = \Pr(\mathfrak{C} \cap E \cap K) + \Pr(\mathfrak{C} \cap \neg E)$ where the last equality holds since $\Pr(\mathfrak{C} \cap E \cap \neg K) = 0$. Furthermore, $\Pr(\mathfrak{C} \cap E \cap K) \leq \Pr(K)$ and $\Pr(\mathfrak{C} \cap \neg E) \leq \Pr(\neg E)$. Hence,

$$\Pr(\mathfrak{C}) \leq \Pr(K) + \Pr(\neg E). \tag{6.17}$$

This statement is not surprising since it should be intuitively clear that the event of a collision occurring is a subset of the disjunctive event that $s^{\mathfrak{r}}$ is in the box around the mean $\bar{s}^{\mathfrak{a}}$ or that $s^{\mathfrak{a}}$ is not in the box.

Due to chance constraint (6.16) we know that control inputs are found that (for sufficiently large N) ensure that $\Pr(K) < \frac{\delta}{2}$. Hence, all we are left to do is to determine box parameter ϱ such that

$$\Pr(\neg E) = \Pr(\|s_t^{\mathfrak{a}} - \bar{s}_t^{\mathfrak{a}}\|_2 > \varrho) \leq \frac{\delta}{2}. \tag{6.18}$$

In principle, this inequality could be solved by numerical integration. Alternatively, we recognize the inequality to be a tail inequality which can be bounded by the inequalities introduced in Sec. 2.4.7. For instance, in the case of Gaussian distributions, we have seen that $\Pr[\|s^{\mathfrak{a}} - \langle s^{\mathfrak{a}} \rangle\|_2 \geq \varrho] \leq 2\exp\left(-\frac{\varrho^2}{2\|\!|C^{\mathfrak{a}}|\!\|_2}\right)$ where $C^{\mathfrak{a}} = \mathrm{Var}[s^{\mathfrak{a}}]$ denotes the variance-covariance matrix. Setting the right hand side less or equal to $\frac{\delta}{2}$ and solving for ϱ shows that it suffices to choose $\varrho = \sqrt{2\|\!|C^{\mathfrak{a}}|\!\|_2 \log\left(\frac{4}{\delta}\right)}$.

Distribution-independent collision bounds. If dimension $d = 2$ we have $||\cdot||_2 \leq \sqrt{2}||\cdot||_\infty$. Hence, we have $\Pr(\neg E) = \Pr(\|s_t^{\mathfrak{a}} - \bar{s}_t^{\mathfrak{a}}\|_2 \geq \varrho) \leq \Pr(\sqrt{2}\,\|s_t^{\mathfrak{a}} - \bar{s}_t^{\mathfrak{a}}\|_\infty \geq \varrho) = 1 - \Pr(\|s_t^{\mathfrak{a}} - \bar{s}_t^{\mathfrak{a}}\|_\infty \leq \frac{1}{\sqrt{2}}\varrho) =: P_t^{\mathfrak{a}}(\varrho)$. Utilizing Whittle's generalization of Chebyshev's inequality [263] yields an upper bound $\beta(\varrho)$ on $P_t^{\mathfrak{a}}(\varrho)$. For the two-dimensional case we have $\Pr(\neg E) \leq P_t^{\mathfrak{a}}(\varrho) \leq \beta$ where $\beta(\varrho) = \frac{1}{\varrho^2}(c_{t,11}^{\mathfrak{a}} + c_{t,22}^{\mathfrak{a}}) + \frac{1}{\varrho^2}\sqrt{(c_{t,11}^{\mathfrak{a}} + c_{t,22}^{\mathfrak{a}})^2 - 4(c_{t,12}^{\mathfrak{a}})^2}$ and $c_{t,ij}^{\mathfrak{a}}$ denotes the covariance of $s_t^{\mathfrak{a}}$ between dimensions i and j (we provide an unabridged derivation in [159]).

Finally, by referring to Eq. 6.17 and Eq. 6.16 we see that we have

$$\Pr(\mathfrak{C}) \leq \frac{\delta}{2} + \Pr(\neg E) \leq \frac{\delta}{2} + \beta(\varrho) \tag{6.19}$$

which provides a recipe that allows us to bound collision probability $\Pr(C)$ below free parameter δ by adjusting virtual obstacle parameter ϱ accordingly.

Note, since approximate bound β is distribution-independent the bound holds for any noise distribution that governs our uncertainty.

Conservatism in the face of unknown moments. The derivations above were made based on the assumption that the true means (and possibly variances) were known. For instance, under linear dynamics with centred noise, this is realistic since then the mean trajectory just is the sequence of states for zero noise. However, particle methods have the advantage that the probabilistic constraints could be approximated directly, e.g. as per Eq. 6.15. And, they also work when the exact moments are unknown or expensive to compute. This can be important especially for the means because they always depend on the sequence of control actions and cannot be computed beforehand offline. Under such circumstances, one could substitute the true mean by the sample mean $\tilde{s}_t^{\mathfrak{a}} = \frac{1}{N}\sum_{j=1}^{N} s_t^{\mathfrak{a},(j)}$ and construct the virtual obstacle around this estimate. If the number N of particles is sufficiently large, the law of large numbers guarantees that this is a good estimate and the results will be identical for practical purposes. Unfortunately, increasing N rapidly increases computational effort. Therefore, it may be worthwhile to attempt keeping N down. To ensure the resulting method still offers conservative guarantees, the belief in the approximation error should be folded in.

To account for the belief over the disparity between sample and population mean, it might also be possible to enlarge the radius (edge length) of the virtual obstacle by a sufficient amount r_s. The offset radius s depends on the sample size N and should be large enough to raise the probabilistic belief in $s^{\mathfrak{a}}$ being inside the obstacle around the sample mean $\tilde{s}$ to a sufficient extent.

Let $E_0 := \{s^{\mathfrak{a}} | \, \|s^{\mathfrak{a}} - \tilde{s}^{\mathfrak{a}}\| < r + r_s\}$ be the event that the position in interaction space of agent $\mathfrak{a}$ is inside the $r + r_s$-ball around the sample mean. In analogy of our argument revolving around Eq. 6.18, we desire to determine r, r_s such that the event probability of its complement $\neg E_0$ is bounded from above by $\frac{\delta}{2}$. To this end, we define two related events for whose complements we know how to determine probability bounds.

Firstly, let $E_1 := \{s^{\mathfrak{a}} | \, \|s^{\mathfrak{a}} - \langle s^{\mathfrak{a}} \rangle\| < r\}$ and secondly, we define $E_2 := \{\tilde{s} | \, \|\tilde{s}^{\mathfrak{a}} - \langle s^{\mathfrak{a}} \rangle\| < r_s\}$ to denote the event that the sample mean $\tilde{s}^{\mathfrak{a}}$ deviates from the true mean $\langle s^{\mathfrak{a}} \rangle$ by at most r_s.

Owing to the sub-additivity of the norm, we see that $\|s^{\mathfrak{a}} - \tilde{s}^{\mathfrak{a}}\| \leq \|s^{\mathfrak{a}} - \langle s^{\mathfrak{a}} \rangle\| + \|\tilde{s}^{\mathfrak{a}} - \langle s^{\mathfrak{a}} \rangle\|$. Hence, $E_0 \supseteq E_1 \cap E_2$ or equivalently, $\neg E_0 \subseteq \neg E_1 \cup \neg E_2$. Thus, by union bound $\Pr[\neg E_0] \leq \Pr[\neg E_1] + \Pr[\neg E_2]$ which is what we desire to bound from above by $\frac{\delta}{2}$. Now, let $d_1, d_2 \in [0, 1], d_1 + d_2 = 1$. As discussed above, we can bound $\Pr[\neg E_1]$ from above by a suitable tail-inequality and simply choose r to be the corresponding

$d_1 \frac{\delta}{2}$-tail radius (cf. Sec. 2.4.7). Similarly, to bound $\Pr[\neg E_2]$, inequalities such as Hoeffding bounds (cf. Thm. 2.4.34) or Bernstein inequalities might be employable to yield a $d_2 \frac{\delta}{2}$-tail radius around the true mean.

Of course, the approach discussed so far can only cope with sample means and had to assume either boundedness or that the variances can be computed or bounded in closed form. If that is not the case, one might plug a sample estimate of the variance into the tail inequalities to compute the tail radii. Especially since the tail inequalities normally are somewhat conservative to begin with, in practice one might not lose conservatism in this approach.

However, the guarantee of conservatism can no longer be given that way. Therefore, we would suggest one of two avenues. One option would be to also fold in a tail bound on the variance estimate and take its error probability into account.

As an alternative, we would suggest to consider employing a purely sample-based tail inequality (e.g. [219] and [126], Eq. 18, Eq. 19). They offer a distribution-independent bound of the form $\Pr[|s - \tilde{s}| > \varrho] \leq \beta(\varrho, N, \hat{\sigma})$ where $\hat{\sigma}$ is a sample variance. This bound could then be utilised to find a conservative radius such that $\beta \leq \frac{\delta}{2}$, which bounds $\Pr[\neg E]$ with E defined on the basis of the sample mean rather than the true mean (cf. Eq. 6.18). However, to date, we have not tried this avenue so far and would have to defer this to potential future work.

Simulations

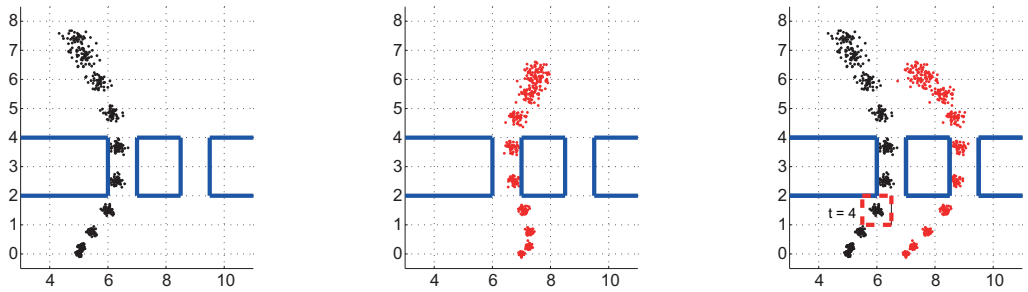
In the following simulations, we consider three different trajectory planning scenarios, all with planning horizon of length ten:

i) A simple example with only two agents to illustrate the very basic functionality of the mechanism. ii) A quantitative evaluation of the average runtime behaviour for an increasing number of agents in an environment with a fixed number of obstacles. iii) A quantitative evaluation of the average number of conflicts to be resolved by the mechanism in an increasingly complex environment for a fixed number of agents.

In all simulations the sample distribution for the agents was chosen as isotropic zero-mean white Gaussian noise with standard deviation $\sigma = 0.001$.

For the first illustration, consider the simulations of a two-agent planning scenario depicted in Fig 6.7. Here two agents 1 and 2 started at locations at the bottom of a map. When generating trajectories to destinations at the far side of the map, they desired to avoid the obstacles (blue rectangles). Their control inputs were accelerations and their state space consisted of locations and velocities. Each agent's cost function quantified the sum of expected ℓ_1 distances to the agent's destination of the generated trajectory.

Planning independently with the particle-control method, the agents found their individually cost-optimal trajectories as depicted in Figs. 6.7(a) and 6.7(b). Note, how the spread of their particle clusters increases as



(a) Agent 1's initial plan.

(b) Agent 2's initial plan.

(c) Updated conflict-free plans.

Figure 6.7.: Simple example. Blue box: obstacle. Dashed box: virtual obstacle for agent 2 for time step 4 (after he lost an auction against agent 1).

the uncertainties accumulate over time. Getting too close to each other at time step four (i.e. causing our coll. probability estimate to exceed our threshold δ) and auction was invoked where agent 1 was determined to be the winner. Hence, agent 2 got a constraint corresponding to a virtual obstacle (dashed box) for time step 4 denying access through the left gap for $t = 4$ and inducing him to instead take the (originally costlier) way around through the right gap (Fig. 6.7(c)).

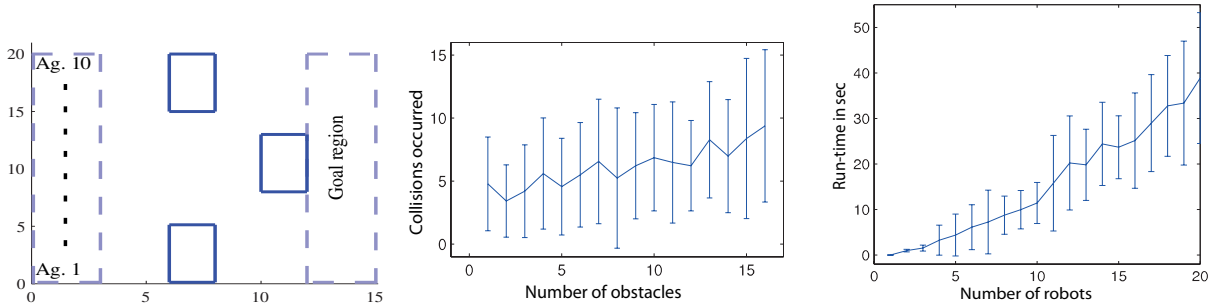
It should be expected that the number of iterations of our mechanism depends on the number of collisions during coordination, which in turn, should increase with the number (and size) of obstacles (or decrease with available free-space) and the number of agents in the system. To develop an intuition for the dependence of run-time on these factors we conducted randomized experiments (with varying agent destinations and obstacle placements) in which run-time and number of collisions were recorded. The results for ten agents with varying destinations and obstacles are depicted in Fig. 6.8(b).

In a third round of simulations, the obstacles were placed at fixed positions together with fixed, equally spaced, starting positions for the agents. In order to provoke potential conflicts, the agents' goals were drawn at random from a uniform distribution. We iteratively added more agents to the planning scenario and set up the mechanism to calculate conflict-free plans for varying numbers of agents. The results are depicted in Fig. 6.8(c).

The simulations were implemented in MATLAB, with no particular emphasis on run-time optimization and all experiments were executed on a standard desktop computer. In summary, Fig. 6.8 illustrates that both the number of coordination iterations (collisions occurred during planning) and run-time increased moderately with increasing problem complexity.

6.3.2. SMPC with collision avoidance constraints on the basis of tail inequalities

In the previous subsection, we have discussed how the auction-method in conjunction with the concept of virtual obstacle can ameliorate the computational blow-up due to approximating the inter-agent collision



(a) An example test environment with three obstacles and ten agents.

(b) Collisions varying with obstacles.

(c) Run-time for planning as a function of the number of agents.

Figure 6.8.: Left arena with three obstacles and ten agents. Centre: Number of arising conflicts vs. varying number of obstacles. Right: Runtime in seconds vs. number of agents. Plots show averages and standard deviations over 50 Monte-Carlo runs of randomized problems.

constraints with constraints between particles. However, if we desire to deploy any of the optimal coordination methods OPT, CCG or SDCG in SMPC collision avoidance, incorporation of inter-agent collision avoidance constraints is unavoidable. To improve tractability, one attempt might be to base estimation of the double integral approximation as per Eq. 6.15 on a sub-sample of the particles. Unfortunately, this would reduce approximation accuracy and certainly, the resulting approach would not guarantee to be conservative and might underestimate the collision risk.

In what is to follow, we will present an alternative approach. Instead on relying on sampling we employ tail inequalities, or equivalently, our collision criterion functions $\gamma^{\mathfrak{a},\mathfrak{r}}$ derived in Ch. 5, to bound the collision probabilities. Coming from the latter angle, remember for collision bound $\delta \in (0,1)$ we have constructed our criterion functions such that the collision probability is at most δ if the criterion function $\gamma^{\mathfrak{a},\mathfrak{r}}$ is positive:

$$\Pr[\mathfrak{C}^{\mathfrak{a},\mathfrak{r}}(t)] < \delta_t \text{ if } \gamma^{\mathfrak{a},\mathfrak{r}}(t) > 0. \tag{6.20}$$

As before, let $s^{\mathfrak{a}}$ denote the uncertain trajectory of agent $\mathfrak{a} \in \mathfrak{A}$ in D-dimensional interaction space. Remember, interaction space is isomorphic to a subspace of state space. That is to say, if $x^{\mathfrak{a}}$ is the state of agent $\mathfrak{a}$, there exists a matrix $\mathcal{T}$ such that $s^{\mathfrak{a}} = \mathcal{T} x^{\mathfrak{a}}$. Assume linear state-transition dynamics $x(t+1) = M_t\, x(t) + B\, u(t) + w(t)$ with zero-mean o.i. noise $w(t)$. Leveraging the assumption of independence of the stochastic increments and application of Lem. 2.4.35 yields the solutions for the first two central moments as per:

$$\langle x(t+1)\rangle = M_t\langle x(t)\rangle + B_t u(t) \tag{6.21}$$

$$= P_{t,0}\langle x(0)\rangle + \sum_{i=0}^{t} P_{t,i+1}\, B_i u(i) \tag{6.22}$$

$$\text{Var}[x(t+1)] = M_t\, \text{Var}[x(t)]\, M_t^\top + \text{Var}[w(t)] \tag{6.23}$$

$$= \sum_{i=0}^{t} P_{t,i+1} \text{Var}[w(i)]\, P_{t,i+1}^\top \tag{6.24}$$

where as before we have defined

$$P_{j,i} = \begin{cases} M_j\, M_{j-1}\, \dots M_i & , i \leq j \\ I_d, & , i > j. \end{cases} \tag{6.25}$$

If we know the matrices $M_t, B_t, \text{Var}[w(t)]$ $(t = 1, \dots, T-1)$, we can leverage these equations to solve for the variance-covariance matrices offline before planning begins. The mean trajectories linearly depend on the open-loop control. Hence, linear state constraints directly translate to linear constraints on the control. This is useful to us since it allows us to find the control inputs via a MILP under state constraints.

Since interaction space was assumed to be a subspace of state space, constraints on the expected interaction space trajectories $(\mu^{\mathfrak{a}}(t))$ can be obtained via linear projection operators $\mathcal{T}$. That is,

$$\mu^{\mathfrak{a}}(t) = \Pi_{\text{proj}}\, \langle x^{\mathfrak{a}}(t)\rangle. \tag{6.26}$$

Similarly, the variance-covariance in interaction space is given by $C^{\mathfrak{a}}(t) = \text{Var}[\mathcal{T} x^{\mathfrak{a}}(t)] = \mathcal{T}\text{Var}[x^{\mathfrak{a}}(t)]\mathcal{T}^\top$. Since the expected state trajectories are linear functions of the control inputs and the nominal interaction space trajectories in turn are linear functions of the expected state trajectories, the expected interaction space trajectories are linear functions of the control inputs as well. Therefore, linear constraints on the nominal interaction space trajectories are equivalent to linear constraints on the control. By contrast, the covariances do not depend on the control inputs and can be computed offline before planning.

Remark 6.3.1. In case the dynamics are non-linear or if the uncertainties $\big(w(t)\big)_{t\in\mathbb{N}_0}$ are of a complicated nature (e.g. due to reflecting boundaries in the environment or correlations) we could still obtain sample estimates of the covariances and means with particle methods as described above. In that case, the constraints would impose restriction on the sample estimates of the moments rather then on the moments themselves. Of course, this approach would computationally more involved and suffers from inaccuracy due to the sampling approximation. But, its greater generality can make it interesting to consider, especially when non-linear optimisation tools become increasingly powerful.

Remember from Eq. 5.3, that one possibility for defining a criterion function is

$$\gamma^{\mathfrak{a},\mathfrak{r}}(t;\varrho(t)) := \max_{i=1,\dots,D}\{|\mu_i^{\mathfrak{a}}(t) - \mu_i^{\mathfrak{r}}(t)| - \Lambda^{\mathfrak{a},\mathfrak{r}} - r_i^{\mathfrak{a}}(t) - r_i^{\mathfrak{r}}(t)\} \tag{6.27}$$

where the radii $r_i^{\mathfrak{a}}(t), r_i^{\mathfrak{r}}(t)$ are the δ_t-tail radii (cf. Sec. 2.4.7) in the ith dimension of the uncertain trajectories $s^{\mathfrak{a}}, s^{\mathfrak{r}}$ at time t, respectively.

For instance, among other choices, in Ch. 5 we have looked at the distribution-independent tail-radii given in Thm. 5.2.4 and the tail radii for the Gaussian case, $r_i^{\mathfrak{q}}(t) = \sqrt{2C_{ii}^{\mathfrak{q}}(t)\log\left(\frac{4}{\delta_t}\right)}$, as per Eq. 5.8. Here, as before, C_{ij} is the covariance between the ith and jth dimension of the interaction space trajectory. That is, $C^{\mathfrak{q}}(t) = \text{Var}[s^{\mathfrak{q}}(t)]$.

The resulting criterion function can be used for conflict detection. To prevent such conflicts in subsequent iterations, collision avoidance constraints will have to be introduced. Given we have calculated the δ_t- tail radii, we can ensure collision avoidance conservatively in a MIP. This can be done by translating the constraint $\gamma^{\mathfrak{a},\mathfrak{r}}(t) > 0$ into a mixed-integer constraint. To this end, note $\gamma^{\mathfrak{a},\mathfrak{r}}(t) > 0$ is equivalent to the disjunctive constraint

$\bigvee_{i=1}^{D} |\mu_i^{\mathfrak{a}}(t) - \mu_i^{\mathfrak{r}}(t)| - \Lambda^{\mathfrak{a},\mathfrak{r}} - r_i^{\mathfrak{a}}(t) - r_i^{\mathfrak{r}}(t) > 0$. In turn, this constraint is equivalent to the disjunction

$$\bigvee_{i=1}^{D} \big(\mu_i^{\mathfrak{a}}(t) - \mu_i^{\mathfrak{r}}(t) - \Lambda^{\mathfrak{a},\mathfrak{r}} - r_i^{\mathfrak{a}}(t) - r_i^{\mathfrak{r}}(t) > 0\big) \vee \big(\mu_i^{\mathfrak{r}}(t) - \mu_i^{\mathfrak{a}}(t) - \Lambda^{\mathfrak{a},\mathfrak{r}} - r_i^{\mathfrak{a}}(t) - r_i^{\mathfrak{r}}(t) > 0\big). \tag{6.28}$$

To express this disjunctive constraint in mixed-integer linear form, we can readily employ the "Big-M" method discussed in Sec. 2.2.1 and add the resulting constraints to the coupling constraint set $\mathcal{C}$ of the mixed-integer constraint representation of the multi-agent planning problem of Eq. 6.13.

The resulting conjunctive constraints with slack variables for avoiding a collision at time t are:

$$\forall i \in \{1,\dots,D\}, \forall j \in \{1,2\} : \mu_i^{\mathfrak{a}}(t) - \mu_i^{\mathfrak{r}}(t) - \Lambda^{\mathfrak{a},\mathfrak{r}} - r_i^{\mathfrak{a}}(t) - r_i^{\mathfrak{r}}(t) + \bar{M}e_{i1}^{\mathfrak{a},\mathfrak{r}}(t) > 0 \tag{6.29}$$

$$\mu_i^{\mathfrak{r}}(t) - \mu_i^{\mathfrak{a}}(t) - \Lambda^{\mathfrak{a},\mathfrak{r}} - r_i^{\mathfrak{a}}(t) - r_i^{\mathfrak{r}}(t) + \bar{M}e_{i2}^{\mathfrak{a},\mathfrak{r}}(t) > 0 \tag{6.30}$$

$$\sum_{j=1}^{2}\sum_{i=1}^{D} e_{ij}^{\mathfrak{a},\mathfrak{r}}(t) < 2D \tag{6.31}$$

$$e_{i,j}^{\mathfrak{a},\mathfrak{r}} \in \{0,1\}. \tag{6.32}$$

where $\bar{M}$ is the "big M" constant chosen to assume a large value and the $e_{ij}^{\mathfrak{a},\mathfrak{r}}$ $(\mathfrak{a},\mathfrak{r} \in \mathfrak{A}, i = 1,\dots,D, j \in \{1,2\})$ are the binary slack variables. The constraint in (6.31) ensures that at least one of the disjunctive terms in (6.28) is satisfied.

With reference to the centralised optimisation problem (6.13), we could define the full jointly feasible set $F_{\mathcal{C}}$ as the set of all control input sequences $(u^{\mathfrak{a}}_{0:T-1})_{\mathfrak{a}\in\mathfrak{A}}$ that satisfy the collision avoidance constraints (6.29) - (6.32) for all time steps and all combinations of agents. Due to due introduction of the binary slack variables this may require the solver to check search over $\mathcal{O}(A^2DT)$ binary variables. Therefore, delaying the introduction of these constraints and variables employing our CCG or SDCG method promises to be highly beneficial and will be investigated in future work. For now we will be satisfied with noticing that in comparison to the centralised particle method where the number of binary constraints was $\mathcal{O}(N^2A^2TD)$ growing quadratically in N the number of particles. Therefore, our new tail bound-based approximation provides significant computational savings. We have provided a more detailed comparison between both approaches in [158, 159].

As mentioned in the previous section, for linear systems, it is of course possible to also conservatively enforce the local collision avoidance constraints $\Pr[s^{\mathfrak{a}}(t) \notin \mathfrak{F}] < \bar{\delta}$ by the tail inequality approach. The intuition of course is that a static obstacle essentially is a high ranking agent that will not move. In section, 2.2.1, we already have discussed how free-space can be stated as a conjunction of disjunction of constraints (cf. Eq. 2.8):

$$\mathfrak{F} = \{s \in \mathcal{X} | \bigwedge_{q=2}^{N_o} (\bigvee_{j=1}^{k_q} \langle s,n_{qj}\rangle > b_{qj})\} \tag{6.33}$$

where each polytope obstacle $O_q = \{s|(\bigwedge_{j=1}^{k_q} \langle s,n_{ij}\rangle \leq b_{ij})\}$ is a set encompassed by k_q hyperplanes.

Let $r^{\mathfrak{a}}_q(t)$ denote a δ_q- tail radius of uncertain position $s^{\mathfrak{a}}(t)$ and $\Lambda^{\mathfrak{a}}$ denote the physical diameter of the agent's plant as defined above. May it be in continuous or discrete time, for collision detection with this obstacle, we can define the criterion function

$$\gamma^{\mathfrak{a}}_{O_q}(t) := \max_{j=1,\ldots,k_q} \langle \mu^{\mathfrak{a}}(t),n_{qj}\rangle - b_{qj} - \frac{\Lambda^{\mathfrak{a}}}{2} - r^{\mathfrak{a}}_q \tag{6.34}$$

which is positive if the distance to some hyperplane of the obstacle is at least $r^{\mathfrak{a}}_q + \Lambda^{\mathfrak{a}}$. Following the same type of argument as above we see that by definition of the δ_q-radius, $\gamma^{\mathfrak{a}}_{O_q}(t) > 0$ implies that $\Pr[s^{\mathfrak{a}}(t) \in O_q] < \delta_q$.

As for the inter-agent collision avoidance constraints, we can convert the condition $\Pr[s \notin O_q] < \delta_q$ into integer constraints. For agent $\mathfrak{a}$'s nominal interaction space position $\mu^{\mathfrak{a}}(t)$, these constraints are:

$$\forall j \in \{1,\ldots,k_q\} : \langle \mu^{\mathfrak{a}}(t), n_{qj} \rangle + \bar{M} \mathfrak{e}^{\mathfrak{a}}_{qj} > b_{qj} + r^{\mathfrak{a}}_q(t) + \frac{\Lambda^{\mathfrak{a}}}{2} \tag{6.35}$$

$$\sum_{j=1}^{k_q} e^{\mathfrak{a},\mathfrak{r}}_{qj}(t) < k_q \tag{6.36}$$

$$e^{\mathfrak{a},\mathfrak{r}}_{qj} \in \{0,1\}. \tag{6.37}$$

To ensure $\Pr[s^{\mathfrak{a}}(t) \notin \mathfrak{F}] < \bar{\delta}$ we impose these constraints for all q. For choosing δ_q we can leverage union bound and define the δ_q such that $\bar{\delta} \geq \sum_{q=1}^{N_o} \delta_q$ (for details refer to Sec. 5.2.2).

Next, we will illustrate our approach in the context of multi-robot collision avoidance.

An example: SMPC for collision avoidance of mobile robot teams

As a test case, we simulate a team of mobile robots of Cornell's 2000 RoboCup team [8]. The mobile robots are holonomic and move in a two-dimensional interaction space of locations. To render the nonlinear dynamics amenable to MILP, we base our simulations on the linearised version derived in [165,227]. The state $x^{\mathfrak{a}} \in \mathbb{R}^4$ of the robot $\mathfrak{a}$ is assumed to be governed by the linear stochastic dynamics $x^{\mathfrak{a}}_{t+1} = Mx^{\mathfrak{a}}_t + Bu^{\mathfrak{a}}_t + w^{\mathfrak{a}}_t$ where the $w^{\mathfrak{a}}_k$ are i.i.d. centred Gaussian random vectors and

$$M = \begin{pmatrix} 1 & 0 & 1-\exp(-\Delta) & 0 \\ 0 & 1 & 0 & 1-\exp(-\Delta) \\ 0 & 0 & \exp(-\Delta) & 0 \\ 0 & 0 & 0 & \exp(-\Delta) \end{pmatrix}, B = \begin{pmatrix} \Delta - 1 + \exp(-\Delta) & 0 \\ 0 & \Delta - 1 + \exp(-\Delta) \\ 1-\exp(-\Delta) & 0 \\ 0 & 1-\exp(-\Delta) \end{pmatrix} \tag{6.38}$$

are the transition and control input matrices, respectively. Parameter Δ is the discretisation step size in seconds. We assumed the control inputs and velocities are bounded: $x^{\mathfrak{a}}_3(t), x^{\mathfrak{a}}_4(t) \in [\mathfrak{v}_{min}, \mathfrak{v}_{max}], u^{\mathfrak{a}}(t) \in [\mathfrak{u}_{min}, \mathfrak{u}_{max}]$ where we chose $\mathfrak{v}_{min}, \mathfrak{u}_{min} = -20, \mathfrak{v}_{max}\mathfrak{u}_{max} = 20$. Each robot $\mathfrak{a}$ was assigned a start $S^{\mathfrak{a}}$ and goal location $G^{\mathfrak{a}}$ as well as an initial velocity of zero. To encourage them to reach their goal positions rapidly, we defined stage costs penalising the $1-$norm distance of their nominal state locations from their goal positions. That is, agent $\mathfrak{a}$'s objective function was $c^{\mathfrak{a}}(u^{\mathfrak{a}}_{0:T-1}) = \sum_{t=0}^{T-1} q^\top |\mu^{\mathfrak{a}}(t) - G^{\mathfrak{a}}|$ where $q = (1,...,1)^\top$ is a vector of ones and $\mu^{\mathfrak{a}}(t) \in \mathbb{R}^2$ is the nominal location at time t. Formulating these in mixed-integer linear form was done by introducing slack variables $y_j(t)$ as discussed in Sec. 2.2.1. Combining this objective with the collision-avoidance constraints would yield a centralised optimisation problem (OPT) of the form:

$$\min_{u^1_{0:T-1},\dots,u^A_{0:T-1}} \sum_{\mathfrak{a}=1}^{A}\sum_{t=0}^{T-1} q^\top y^\mathfrak{a}(t) \tag{6.39}$$

$$\text{s.t.} : \forall \mathfrak{a}, \mathfrak{r} \in \mathfrak{A} = \{1,\dots,A\}, t \in \{1,\dots,T-1\}, j \in \{1,2\}, i \in \{1,\dots,D\} : \tag{6.40}$$

$$\bar{x}^\mathfrak{a}(t+1) = M\bar{x}^\mathfrak{a}(t) + Bu^\mathfrak{a}(t) \tag{6.41}$$

$$\mu^\mathfrak{a}_j(t) - G^\mathfrak{a}_j \leq y^\mathfrak{a}_j(t),\ -\mu^\mathfrak{a}_j(t) + G^\mathfrak{a}_j \leq y_j(t)^\mathfrak{a} \tag{6.42}$$

$$\mu^\mathfrak{a}_i(t) - \mu^\mathfrak{r}_i(t) - \Lambda^{\mathfrak{a},\mathfrak{r}} - r^\mathfrak{a}_i(t) - r^\mathfrak{r}_i(t) + \bar{M}e^{\mathfrak{a},\mathfrak{r}}_{i1}(t) > 0 \tag{6.43}$$

$$\mu^\mathfrak{r}_i(t) - \mu^\mathfrak{a}_i(t) - \Lambda^{\mathfrak{a},\mathfrak{r}} - r^\mathfrak{a}_i(t) - r^\mathfrak{r}_i(t) + \bar{M}e^{\mathfrak{a},\mathfrak{r}}_{i2}(t) > 0 \tag{6.44}$$

$$\sum_{j=1}^{2}\sum_{i=1}^{D} e^{\mathfrak{a},\mathfrak{r}}_{ij}(t) < 2D \tag{6.45}$$

$$e^{\mathfrak{a},\mathfrak{r}}_{i,j}(t) \in \{0,1\}, x^\mathfrak{a}_3(t), x^\mathfrak{a}_4(t) \in [\mathfrak{v}_{min}, \mathfrak{v}_{max}], u^\mathfrak{a}(t) \in [\mathfrak{u}_{min}, \mathfrak{u}_{max}] \tag{6.46}$$

$$\Pr[s^\mathfrak{a}(t) \notin \mathfrak{F}] < \bar{\delta}. \tag{6.47}$$

Note, the nominal state trajectory $\bar{x}^\mathfrak{a}(\cdot)$ and interaction space trajectories $\mu^\mathfrak{a}(\cdot)$ were only used for improved readability. In an actual implementation, one would substitute them by linear functions of the control as per Eq. 6.26 and Eq. 6.22. So, the only variables optimised over were the control and slack variables.

The chance constraint in 6.47 could either be enforced by

- **(i)** particle constraints as described in the previous subsection, or
- **(ii)** by particle-free constraints as per (6.35) - (6.37).

We tested both variants.

Exp. 1. In the first experiment, we chose the particle approach for avoidance of the static obstacles with $N = 80$ particles per times step.

An example of and open-loop coordination outcome is depicted in Fig. 6.9. Here two robots 1 and 2 plan over a time horizon of $T = 13$ time steps. The time step size was chosen to be $\Delta = 1[s]$. The per-time collision probability bound was set to $\delta^{1,2} = 0.05$. We ran the open-loop planner with collision avoidance constraints based on two different tail radii. The first utilised the radii derived from Whittle's distribution-independent inequality given in Thm. 5.2.4. The second run imposed constraints on the basis of Gaussian tail inequalities as per Eq. 5.6. The boxes around the nominal trajectories reflect the size of these radii. As can be seen from the sizes of the plots, the Gaussian tail radii reduce conservatism of our collision avoidance constraints. This in turn gives rise to reduced conservatism of the planning outcome: the trajectories under the assumptions of Gaussianity are allowed to pass by each other more closely than when using the more conservative, distribution-independent bounds. This can have multiple beneficial consequences.

Firstly, as exemplified by the plot, the open-loop plans can have reduced social cost due to larger joint free-space than with the more conservative tail bounds. Secondly, over the course of repeated optimisation during planning, in all likelihood, fewer collision constraints might become active. This might translate to reduced runtime, especially when harnessing the CCG or SDCG methods. Investigating such effects on randomised problem instances may be done in future work.

(a) Coordinated plans on the basis of Whittle's bounds.

(b) Coordinated plans on the basis of Gaussian tail bounds.

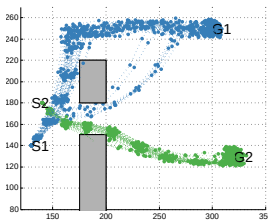
Figure 6.9.: Open-loop coordination result in the multi-robot scenario based on two different tail bounds. The dots are sample points from the uncertain trajectories. There are two static obstacles (grey rectangles) that are avoided with the particle-based method. The black boxes around the trajectory means demarcate the stochastic $\delta-$ tubes (for $\delta = 0.05$). When defining the tube on the basis of the distribution-independent bound as per Thm. 5.2.4 the conservatism of the bounds causes Robot 2 having to take the longer way around and consequently miss his goal at the finite time step. Observe how folding in knowledge of the Gaussian nature of the uncertainty results in reduced conservatism and hence better plans.

Exp. 2. So far, our discussion has focussed on open-loop SMPC. Indeed, there seems to be no work in SMPC with unbounded support distributions that can provide closed-loop stability guarantees for receding horizon control (RHC) with collision-avoidance constraints. This is hardly surprising. After all, due to the unbounded support there will be a non-zero chance of ending up anywhere in state space (including in an obstacle) at any given point in time [178]. Furthermore, due to the necessity to resort to integer programming, open-loop planning is computationally demanding which might limit its applicability to RHC in domains where high frequency control is required.

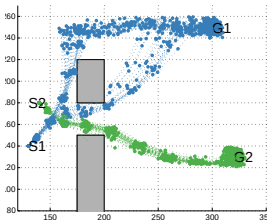
In order to gain a first impression on the performance of our approach in RHC, we generated 40 closed-loop paths with our planner embedded in the loop in a receding horizon fashion. As before, Each open-loop optimisation problem was evoked with a horizon of 12 time steps. The simulation was conducted for a duration of 16 simulated seconds. The results are depicted in Fig. 6.10.

We made four observations. Firstly, we note that the constraints based on Gaussian tail inequalities led to more trajectories of robot 1 passing through the gap between the obstacles in close proximity to robot 2's trajectory. Secondly, the closed-loop trajectories all arrive at the designated goal locations. Thirdly, collisions

are avoided in the vast majority of cases. Note, when we set $\delta = \delta^{1,2} = 0.05$ it is not surprising that on rare occasions, collisions with obstacles do occur (e.g. see in Fig. 6.10(a), agent 1's trajectory around time step 4). Finally, in most instances the optimisation problems remained feasible at all time steps. However, in a small fraction (ca. 1%) of the trials, the random disturbance of the dynamics carried the robots in a joint configuration where a feasible (i.e. collision free with sufficient probability) solution could no longer be found. In our simulations, we captured this error and simply restarted the trajectory simulation. Of course in a real-world deployment, other measures need to be taken (e.g. a temporary increase δ).



(a) Closed loop paths on the basis of Whittle's bounds.



(b) Closed loop paths on the basis of Gaussian tail bounds.

Figure 6.10.: Closed loop paths resulting from applying the open loop planner in a receding horizon fashion. Starting at $S1, S2$ respectively, robots 1 and 2 successfully reach their respective goal locations G1 and G2.

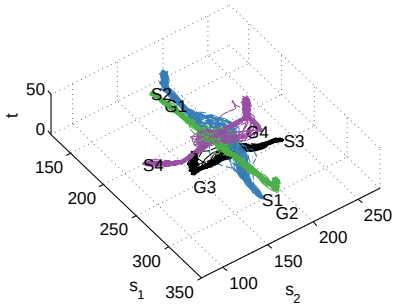
Exp. 3. Having verified that the closed-loop policy overall seems to be successful in reaching goals while avoiding collisions, it remains to be investigated whether our approaches are applicable in real time. In particular, if we desire to run our open-loop planners in a receding horizon control loop, runtimes of the optimisation-based planners need to be sufficiently rapid.

First, we tested the controller in the constraint setup employed in the previous two experiments– that is a hybrid of the particle-free inter-agent constraints for inter-agent collision avoidance and the particle-based method for avoidance of the static obstacles. This time, we reduced the number of particles from 80 to 30 per time step.

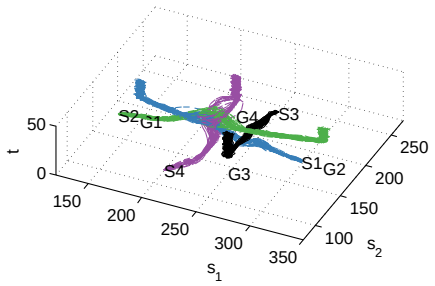
We removed all obstacles and set up a collision avoidance problem with four agents as depicted in Fig. 6.11. This time, we chose a temporal resolution of $\Delta = 0.5[s]$. That is, in a real-time situation the planner would have to be called with a frequency of at least $2[Hz]$. The horizon length was set to $T = 12$ and the the closed-loop dynamics were simulated for 20 simulated seconds, i.e. for 40 time steps. The simulations were run in Matlab on a Laptop with an i7-3630QM processor (2.4 Ghz) and 12 GB Ram.

The recorded run-times for each open-loop planning problem at the various stages are depicted in Fig. 6.11(c). The plots show that the maximum runtimes during early stages (i.e. when collisions occur) would be markedly above the temporal resolution of $\Delta = 0.5[s]$. Therefore, for this problem and in this implementation, the controller would be too slow to keep up with the required sample rate of $2[Hz]$.

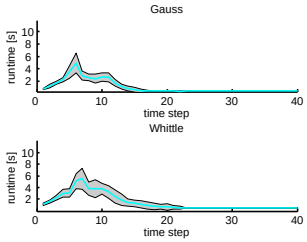
Comparing the solutions based on Whittle's and Gaussian tail bounds, we note that why the trajectories generated with the former are slightly more spread out (cf. Fig. 6.11(a) vs. Fig. 6.11(b)), there is otherwise little noticeable difference between the solutions in this particular problem. In a second round, we tested



(a) Closed-loop trajectories with RHC performed on the basis of Whittle's bounds.



(b) Closed-loop trajectories with RHC performed on the basis of Gaussian tail bounds.

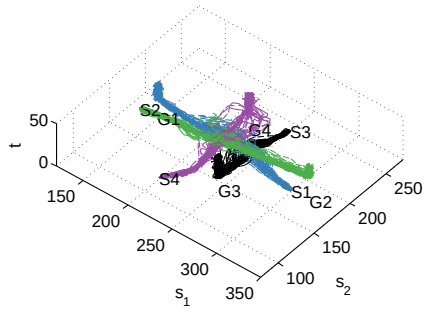


(c) Runtimes of open-loop planning in each stage. Shaded area encompasses two units of standard deviation around the mean.

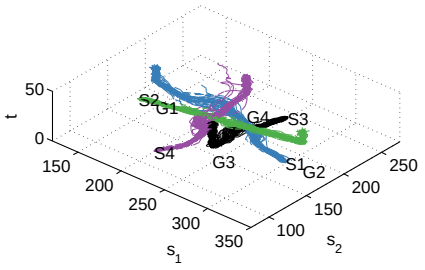
Figure 6.11.: 50 simulated closed-loop trajectories resulting from applying the open-loop planner (with particle-based obstacle avoidance) in a receding horizon fashion. The planner was evoked at 2 Hz. Agent $\mathfrak{a}$ is tasked with traversing the plane from start location $S^{\mathfrak{a}}$ to goal location $G^{\mathfrak{a}}$. In the vast majority of runs, all agents reached their goals. In no instance there was a collision between two trajectories. s_1, s_2 denote the first and second component of interaction space, t denotes the time step.

the open-loop planner in the completely particle-free configuration described above. The recorded results are shown in Fig. 6.12. Note exchanging the particle-based obstacle avoidance module by the completely particle-free constraints resulted in a per-stage speed-up of about 35 bringing the method within reach of RHC in real-time settings.

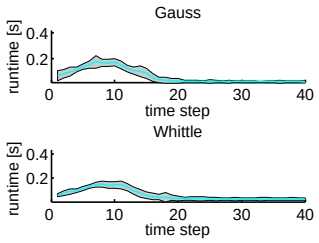
Exp. 4. As a final test, we wanted to get an impression of the impact of the number of agents and the horizon length to the tractability of coordination in the worst-case scenario of centralised coordination (OPT). In this setting, the number of interaction-action constraints (and hence, the number of binary variables) grows quadratically in the number of agents and linearly in the horizon. Unfortunately, as mentioned above, in integer-programming, the worst-case complexity is exponential in the number of binary variables. Therefore,



(a) Closed-loop trajectories with RHC performed on the basis of Whittle's bounds.



(b) Closed-loop trajectories with RHC performed on the basis of Gaussian tail bounds.



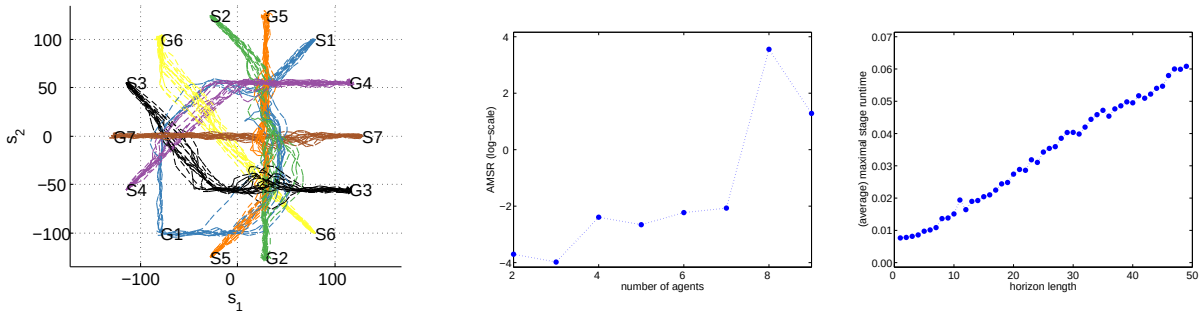
(c) Runtimes of open-loop planning. Shaded area encompasses two units of standard deviation around the mean.

Figure 6.12.: 50 simulated closed-loop trajectories resulting from applying the **particle-free** open-loop planner in a receding horizon fashion. The planner was evoked at 2 Hz. Agent $\mathfrak{a}$ is tasked with traversing the plane from start location $S^{\mathfrak{a}}$ to goal location $G^{\mathfrak{a}}$. In the vast majority of runs, all agents reached their goals. In no instance there was a collision between two trajectories. s_1, s_2 denote the first and second component of interaction space, t denotes the time step.

the worst-case complexity has to be estimated exponential in the number of agents and planning horizon. However, the average-case complexity, which depends on the specific problem at hand, may be much lower than this. To obtain a conservative sense of the computational limitations of the centralised approach, we choose the setting again where the agents are arranged in a circle which the agents need to traverse to reach their goals. An example outcome of the closed-loop trajectories for a problem with seven agents is depicted in Fig. 6.13(a).

This is a problem setup with high agent interactions and can be considered to be a hard coordination problem. Note, if circle size remains constant, the available free-space per agent shrinks as the number of agents grows. To somewhat adjust for this, we defined the circle radius such that the arc-length distance between two agents on the circle remained constant.

Note, problems with an even number of agents were harder than problems with an odd number of agents. This was the case since in problems with an even number of agents, the start locations of agents were the goal locations of others. This may have been the cause for somewhat stronger interaction and consequently,



(a) Coordinated closed-loop paths of seven agents arranged in a circle.

(b) Average maximum stage runtimes (AMSR) as a function of the number of agents A and fixed horizon $T = 10$.

(c) Average maximum stage runtimes as a function of planning horizon length T for problem with two agents.

Figure 6.13.: Averages of the maximum runtime of each open-loop optimisation of each stage as a function of the number of agents A (Fig. 6.13(b)) and as a function of planning horizon T (Fig. 6.13(b)). Averages were taken over the noisy closed-loop trajectories.

relatively longer runtimes (cf. Fig. 6.13(b)).

6.3.3. Discussion

In this section, we have derived both particle-based and particle-free methods for multi-agent collision avoidance in SMPC. We have discussed how tail inequalities can be harnessed to remove the dependence on sampling. This reduces the worst-case complexity of the pertaining optimisation problems. In addition, it also markedly improved measured runtimes in our simulations suggesting its applicability in RHC with probabilistic collision avoidance constraints. Our simulations suggest that in the particle-free setup, even the centralised method has the potential of being utilised in RHC coordination for sampling rates beyond one Hz. However, for larger numbers of agents the complexity renders an application to RHC intractable in problem domains that require high frequency control updates. Of course, it should be borne in mind that such runtime evaluations depend on details of implementation and should therefore be taken with a grain of salt. For instance, Yu and Lavalle [273] report coordination of large numbers of agents in graph planning with a centralized approach solved by Gurobi [113]. Considering a very similar setup, our optimisers were only capable of solving graph planning problems of up to ten agents within reasonable time in a centralised setting. Since different solvers use different search heuristics, choosing a solver that is well adjusted to the structure of the particular application at hand will probably have to be done on a case by case basis.

To improve scalability in the number of agents, we expect our coordination methods to be highly beneficial. We already have described how the lazy auction coordination method can be employed for distributed SMPC and tested it in the context of the computationally more demanding particle-based SMPC control approach. Testing AUC, as well as our other methods CCG and SDCG in combination with our particle-free SMPC control method seems very promising. In the context of graph routing, application of our coordination methods

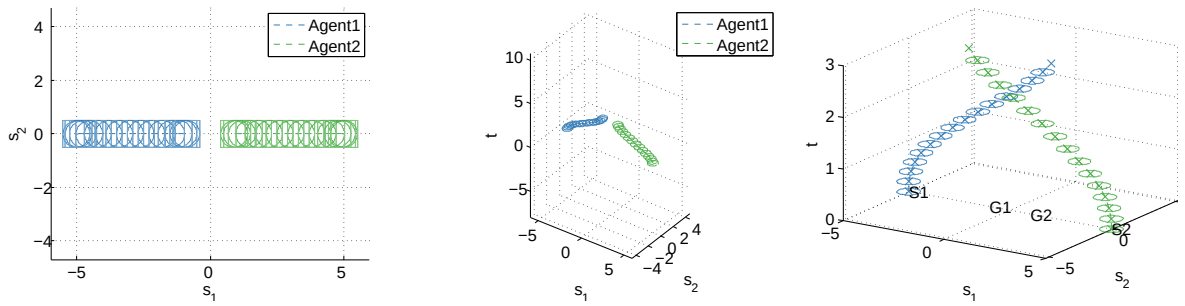
yielded speed-ups in the hundreds even for small agent numbers. On the other hand, the particle-free SMPC method gave rise to a speed-up in the tens. If these results are any indication, combining both approaches promises significant speed-up and improved scalability. We would hope that a merger of our coordination method with the particle-free SMPC approach will yield a potent method for multi-agent SMPC capable to be run in closed-loop RHC even for medium to large numbers of agents or smaller numbers of agents with high frequencies. While testing this has to be deferred to future work, the work presented in this chapter has laid out all the necessary components leaving the merger a mere matter of implementation.

Open issues. On occasion, the receding horizon implementation has generated trajectories that ended up being infeasible. That is, the system ended up in a joint state where the solver was unable to find a feasible joint control action that would result in trajectories with collision probabilities below the desired threshold. We have found that this could often be avoided by changing the planning horizon T of the open loop planner. Often this could be avoided by restarting the closed-loop trajectory simulations with an increase of planning horizon T. However, contrary to intuition, it was not always the case that an increase of T would necessarily ameliorate the problem. On the contrary, sometimes increasing it would trigger the problem in the first place and the solution to an existing feasibility problem lay in a decrease of the horizon.

6.4. Kinky inference in conservative collision avoidance of learning agents with uncertain drift

In Sec. 6.3, we considered predictive control for multi-agent collision avoidance where the agents' plants were governed by stochastic difference equations. In this section, we deploy similar methods but in a setting where the uncertainty is due to epistemic uncertainty over the governing dynamics. In particular, it is assumed that the drift of the system is uncertain. The drift then is (partially) inferred from observations employing kinky inference (cf. Ch. 4). Folding in information about the drift afforded from the partially identified model, the agents are controlled with our multi-agent MPC methods. Taking the uncertainty estimates around the KI predictions into account, collision avoidance is ensured by constraints on the tube distances around the uncertain trajectories. In contrast to the SMPC considerations of [158, 159], the size of the tubes is chiefly determined by the uncertainty quantification provided by the kinky inference rule. Assuming the prior assumptions are true and the drift is contained in $\mathcal{K}_{prior}$ (cf. Sec. 4.2), Thm. 4.2.7 asserts that the resulting controls will be conservative. That is, so long as we ensure the tubes do not intersect, the actual trajectories are guaranteed not to intersect. In addition, the theorem guarantees that the conservatism will shrink as additional data about the drift will become available.

Below, we will provide illustrations of the application of these results to the multi-agent collision avoidance problem. The simulations behave as predicted by our theory. There are numerous works in the control



(a) Open-loop plan. (b) Planned trajectory. (c) Actual trajectory after plan execution.

Figure 6.14.: *Exp1.* The leftmost plot shows the plans of agent 1 (blue) and agent 2 (green). The boxes depict the safety tube bounding the error whereas the circles indicate the physical expansion of the circular agents. Their start positions are $S1 = (-5, 0)^\top$ and $S2 = (5, 0)^\top$ while their goal locations are $G1 = (-1, 0)^\top$ and $G2 = (1, 0)^\top$, respectively. With the agents having a radius of $r = 0.5$ and not assuming any disturbances, the planner computes control actions that are anticipated to allow the agents to come to rest at their goal locations (see Fig. 6.14(a) and Fig 6.14(b)). However, when the plans are executed in the real system, the unforeseen drift f causes the trajectories to collide when the plans are executed (Fig. 6.14(c)).

literature that give guarantees on control success in a robust setting (cf. Sec. 2.2) as well as a vast number of works on agent coordination and collision avoidance (cf. Sec. 2.3). Nonetheless, we believe this to be the first work that can provide guarantees on the belief of collision avoidance that is due to bounds of learning algorithms that have identified an uncertain dynamic system.

6.4.1. Conservative open-loop control for collision avoidance of kinky inference learners

Since it is a stepping stone of RHC and to bring out more clearly the effects of learning and uncertainty, we focus on open-loop control. Suppose we have A agents with index set $\mathfrak{A}, |\mathfrak{A}| = A$. Each agent $\mathfrak{a} \in \mathfrak{A}$ has plant dynamics

$$x^{\mathfrak{a}}[k+1] = Ax^{\mathfrak{a}}[k] + Bu^{\mathfrak{a}}[k] + f(x^{\mathfrak{a}}[k]) \tag{6.48}$$

with matrices as considered in the double-integrator UAV model considered in [158]. That is, we chose

$$A = \begin{pmatrix} 1 & 0 & \Delta & 0 \\ 0 & 1 & 0 & \Delta \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}, B = \begin{pmatrix} 0 & 0 \\ 0 & 0 \\ \Delta & 0 \\ 0 & \Delta \end{pmatrix} \tag{6.49}$$

where $\Delta = 0.2$ was a predefined time increment representing the temporal discretisation resolution of the discrete-time approximation of the dynamics.

As before, f is an (uncertain or unknown) drift, $x^{\mathfrak{a}}[k] \in \mathcal{X} \subset \mathbb{R}^4$ denotes the state at time step k and $u^{\mathfrak{a}}[k]$

is the pertaining control input. Once more, we consider a collision avoidance scenario. Here, interactions between agents take place on a map of locations $\mathcal{S} = [-10, 10]^2 \subset \mathbb{R}^2$. A collision is the event where the plant locations of two agents get too close to each other, which is to be avoided. Formally, a collision is considered to occur if $\exists k \in \mathbb{N}, \mathfrak{a}, \mathfrak{r} \in \mathfrak{A} : \|s^{\mathfrak{a}}[k] - s^{\mathfrak{r}}[k]\| \leq \Lambda$ where $\Lambda \in \mathbb{R}_+$ is the agents' physical radius and $s^{\mathfrak{a}} = (x_1^{\mathfrak{a}}, x_2^{\mathfrak{a}})^\top \in \mathcal{S}$ is agent $\mathfrak{a}$'s interaction-space positions, i.e. its location in the environment given by a (bounded) set of attainable locations $\mathcal{S}$.

Exp. 1. In predictive control, a correct model can be crucial for control success [161]. This holds true especially in open-loop control where no feedback is provided to reduce the effects of model uncertainty. As an illustration, consider the plots of Fig. 6.14. Here, two agents were tasked to move from their start locations to their respective goal locations. Assuming our linear double-integrator model in the absence of a nonlinear drift ($f \equiv 0$), they plan (using MILP as in Sec. 2.2) to generate control sequences that are anticipated to lead the agents' plants to settle at their respective goal locations $G1 = (-1, 0)^\top$ and $G2 = (1, 0)^\top$.

We then simulated the trajectories resulting from executing the planned control sequences. However, deviating from the model utilised during planning, we set the drift function

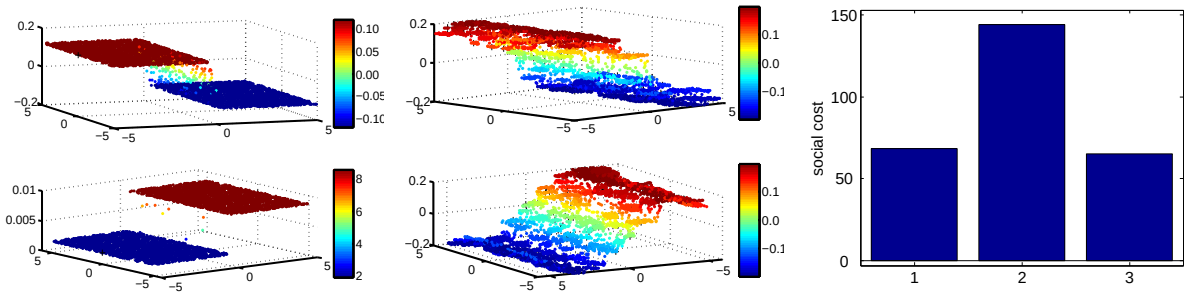
$$f(x) := \Delta\big(0, 0, -\sin(0.5\, x_1), -\sin(0.5\, x_2)\big)$$

for all agents $\mathfrak{a} \in \mathfrak{A} = \{1, 2\}$. In the context of mobile robot control, the drift function might simulate gravitational terms due to an a priori uncertain hilly terrain that has a valley around the origin. (That is, the gravitational forces are proportional to the slope of the terrain. So, with this choice of drift forces pushing towards the origin in a neighbourhood of the origin, this corresponds to a valley being located at the origin.)

Since the influence of the drift f was unaccounted for during planning, the actual trajectories (Fig. 6.14(c)), resulting from executing the plans, markedly deviated from the anticipated ones (Fig. 6.14(b)), yielding a collision around time step $k = 12$ (refer to Fig. 6.14(c)).

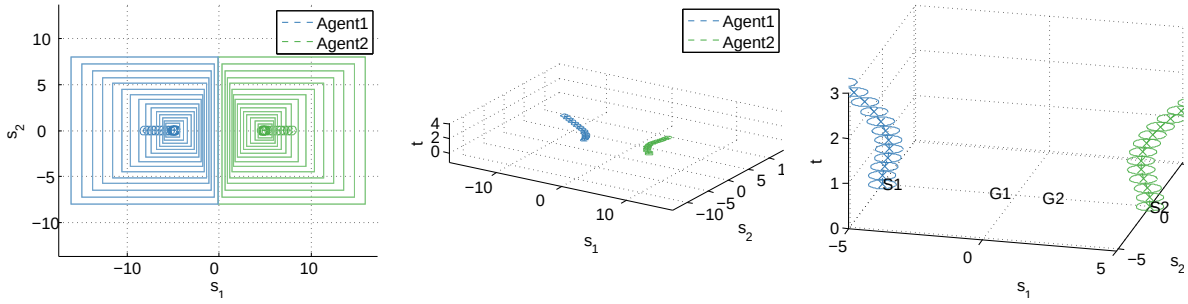
Exp. 2. To compensate for this, we employed a kinky inference learner to infer the drift based on a noisy sample. The latter could be obtained via observing state differences $\mathrm{d}x[k] = x[k+1] - x[k]$ and then solving for $f(x[k]) = \mathrm{d}x[k] - Ax[k] + Bu[k]$. Assuming the state is observable up to error ε_x this allows us to generate a data set of the form $\mathcal{D}_n = \big\{x[k_i], f(x[k_i]), \varepsilon \big| i = 1, \ldots, N_n\big\}$ where ε captures the effects of input and output noise as discussed above.

We initialised the KI learner with one training point for the drift value at each agent's start location in interaction space $\mathcal{S} = [-10, 10]^2$, perturbed by some noise of $\varepsilon = 0.01$. That is, with notation of Sec. 4.2, we generated a data set $\mathcal{D}_1 = \{(S1, f(S1), 0.01), (S2, f(S2), 0.01)\}$. Folding in knowledge that the drift was only dependent on interaction space inputs, the KI learner determined a maximum error of $\bar{\mathfrak{v}} =$



(a) Initial prediction. (b) Improved prediction. (c) Social cost after each experiment.

Figure 6.15.: Belief over drift models based on $\mathcal{D}_1$ and $\mathcal{D}_2 \supset \mathcal{D}_1$ with $|\mathcal{D}_1| = 2, |\mathcal{D}_2| = 100$. Here, the top figures show the predictions of $\hat{\mathrm{f}}_{n,3}(s)$ and the bottom plots depict the predictions of $\hat{\mathrm{f}}_{n,4}(s)$ for $n = 1$ and $n = 2$, respectively. The ground-truth drift model was $f(x) = \big(0, 0, -\sin(0.5\,x_1), -\sin(0.5\,x_2)\big)^\top$. Rightmost plot: Social cost after each experiment. Note how the reduced uncertainty translated to reduced social cost of the last experiment (bar plot 3) vs the cost in the second (bar plot 2). While the first experiment (bar plot 1) also accumulated low social cost, a collision occurred.



(a) Open-loop plan. (b) Planned trajectory. (c) Actual trajectory after plan execution.

Figure 6.16.: Exp2. Taking into account the conservative uncertainty bounds of the kinky inference predictions, the decisions (control actions) are conservative and hence, collisions are avoided during plan execution (Fig. 6.16(c)) according to plan (Fig. 6.16(a) and Fig. 6.16(b)). However, since the KI learner is initialised with only two training examples, the uncertainty is large and hence, the plans very conservative. In particular, the goals are never reached. Moreover, to compensate for the ever growing uncertainty bounds, the control makes the trajectories move further apart after a while to maintain the collision avoidance guarantee.

$\sup_{s\in\mathcal{S}} \| \hat{\mathfrak{v}}_1(s)\|_\infty = 1.27$. The resulting (coarse) prediction model $\hat{\mathfrak{f}}_1(\cdot)$ is depicted in Fig. 6.15(a) for various query points on a subset of the interaction space $\mathcal{S}$.

In lieu to feedback linearisation and MRAC control approaches [67] discussed above, we chose a control $u^{\mathfrak{a}}[k] := (-\nu^{\mathfrak{a}}_{ad} + \bar{u}^{\mathfrak{a}}[k])$ with $\nu^{\mathfrak{a}}_{ad} = \hat{\mathfrak{f}}(x)$ defined by the kinky inference prediction rule as per Def. 4.2.4. Here, $\bar{u}^{\mathfrak{a}}$ denotes a pseudo control yet to be defined. In contrast, to MRAC [67] we did not ascribe a fixed feedback policy to the pseudo-control. Instead, we intended to determine the pseudo control trajectory with MILP-based open-loop planning.

Applying the control yielded the closed-loop dynamics,

$$x^{\mathfrak{a}}[k+1] = Ax^{\mathfrak{a}}[k] + B\bar{u}^{\mathfrak{a}}[k] + F(x^{\mathfrak{a}}[k]) \tag{6.50}$$

where $F(x) = f(x) - \hat{\mathfrak{f}}(x)$ denotes an uncertain increment determining the deviation of the actual trajectory $x^{\mathfrak{a}}[\cdot]$ from the nominal trajectory given by the *nominal dynamics*

$$\bar{x}^{\mathfrak{a}}[k+1] = A\bar{x}^{\mathfrak{a}}[k] + B\bar{u}^{\mathfrak{a}}[k]. \tag{6.51}$$

Note, in contrast to our considerations in Sec. 6.3, the discrepancy is not modelled in a stochastic manner, but is assumed to be due to rest uncertainty of the KI rule. Since the latter provides an error estimate around its predictions, we can utilise these uncertainty quantifications to bound the discrepancy between nominal and actual dynamics. Since we aim to do planning on the basis of the nominal model, we need to determine how much extra space to leave between the agents to enforce collision avoidance in the presence of error $e^{\mathfrak{a}}[k] = x^{\mathfrak{a}}[k] - \bar{x}^{\mathfrak{a}}[k]$. In other words, how large do we have to choose the tube radius around the nominal trajectory?

In pursuit of an answer, we once again consider the error dynamics:

$$e^{\mathfrak{a}}[k+1] = Ae^{\mathfrak{a}}[k] + F(x^{\mathfrak{a}}[k]). \tag{6.52}$$

Going through analogous steps as in Sec. 4.4.3 (with A in place of M), we see that

$e^{\mathfrak{a}}[k] = A^k\, e^{\mathfrak{a}}[0] + \sum_{i=0}^{k-1} A^{k-1-i}\, F(x^{\mathfrak{a}}[i])$ and hence, $\|e^{\mathfrak{a}}[k]\| \leq \left|\!\left|\!\left| A^k \right|\!\right|\!\right| \, \|e^{\mathfrak{a}}[0]\| + \bar{\mathfrak{v}}_k \sum_{i=0}^{k-1} \left|\!\left|\!\left| A^{k-1-i} \right|\!\right|\!\right|$ where $\bar{\mathfrak{v}}_k = \max_{i=1,\dots,k-1} \|F(x^{\mathfrak{a}}[i])\|$. Since F is nonlinear, we would rather allow our control decisions to be conservative than having to incorporate the nonlinear term $F(x)$ in the constraints of our optimisation problem. (Note, that for certain input norms, the KI rule generates piece-wise linear prediction uncertainty functions. In future work, we would like to investigate using this property to convert constraints on F into linear constraints. We hope to be able to do so employing techniques akin to LP conversion techniques found in piece-wise linear optimisation [43].)

Therefore, we replace $\bar{\mathfrak{v}}_k$ by the more conservative constant

$$\bar{\mathfrak{v}} := \sup_{s\in\mathcal{S}} \|F(s)\| \leq c \sup_{s\in\mathcal{S}} \|\hat{\mathfrak{v}}(s)\|_\infty . \tag{6.53}$$

Here, c is a conversion constant given by the *equivalent norm inequalities* (cf. Eq. 2.20) that ensure $\|\cdot\| \leq c\,\|\cdot\|_\infty$. For instance, if $\|\cdot\|$ denotes the Euclidean norm, then $c = \sqrt{2}$ (cf. Sec. 2.4.2).

This allows us to obtain the state-independent bound

$$\|e^{\mathfrak{a}}[k]\| \leq \left|\left|\left|A^k\right|\right|\right| \, \|e^{\mathfrak{a}}[0]\| + \bar{\mathfrak{v}} \sum_{i=0}^{k-1} \left|\left|\left|A^{k-1-i}\right|\right|\right| =: r^{\mathfrak{a}}[k]. \tag{6.54}$$

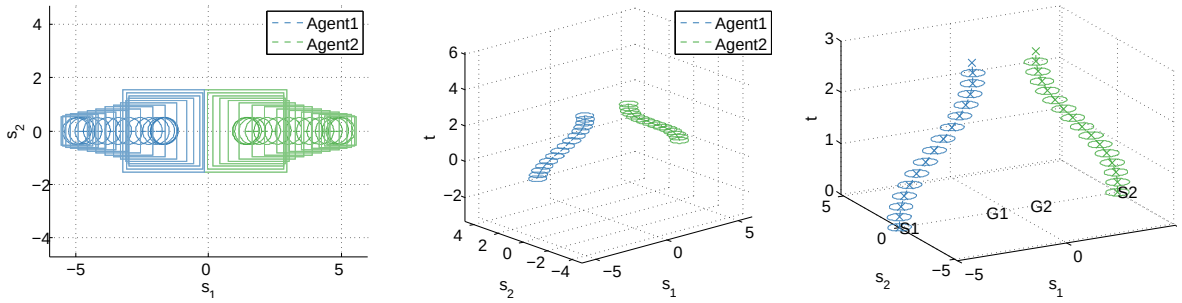
This bound $r_k^{\mathfrak{a}}$ is more conservative than if we had based it upon $\bar{\mathfrak{v}}_k$ instead of on $\bar{\mathfrak{v}}$. Nonetheless, it has the advantage of being state-independent and hence, can be pre-computed for all time steps within a given planning horizon.

In the absence of a nonlinear dependence on the state, open-loop planning could be conducted based on mixed-integer linear programming, analogously to the approach described in Sec. 6.3. The difference between the approaches is that in this section's simulations, the radii $r^{\mathfrak{a}}[k]$ of the tube were computed as per Eq. 6.54. In particular, they were based on the uncertainty due to the KI predictions quantified by $\hat{\mathfrak{v}}$ rather than on the basis of probabilistic tail radii. To enforce collision avoidance, we then introduced tube constraints of the form $\gamma^{\mathfrak{a},\mathfrak{r}}[k] > 0, \forall k \forall \mathfrak{a}, \mathfrak{r} \in \mathfrak{A}$ (to the constraint set of the multi-agent optimisation problem) on the basis of a robust criterion function (cf. Sec. 5.2.2) $\gamma^{\mathfrak{a},\mathfrak{r}}(k) = \|\bar{s}^{\mathfrak{a}}[k] - \bar{s}^{\mathfrak{r}}[k]\| - \Lambda - r^{\mathfrak{a}}[k] - r^{\mathfrak{r}}[k]$ where the tube radii $r^{\mathfrak{a}}[k], r^{\mathfrak{r}}[k]$ were defined as in Eq. 6.54, ruling out the possibility of any collisions.

The results of the planning process with these conservative constraints are depicted in Fig. 6.16.

This time, the planner had successfully taken into account the uncertainty, resulting in successful collision avoidance. However, due to the fact that the drift model had been scarcely identified, the resulting large uncertainty estimates, in conjunction with the conservatism of the decision making process due to our bounds, caused the planner to conceive control actions that resulted in trajectories that kept a rather larger safety distance between the agents. While this did prevent collisions, the goal locations could not be reached (Fig. 6.16(c)).

Exp. 3. Since all uncertainty is due to epistemic uncertainty of the inductive inference rule over the deterministic drift field, we expect the conservatism to reduce with increasing learning experience. To test this, we updated the data set from $\mathcal{D}_1$ to $\mathcal{D}_2$ the latter of which now contained 100 samples of the drift drawn uniformly at random from the interaction space. The results are shown in Fig. 6.17. As expected, the reduced uncertainty bound ($\bar{\mathfrak{v}} = 0.31$) yielded less conservative plans (see Fig. 6.17(a)). Furthermore, the actual trajectories observed from plan execution (Fig. 6.17(c)) closely matched the planned trajectories



(a) Open-loop plan. (b) Planned trajectory. (c) Actual trajectory after plan execution.

Figure 6.17.: *Exp3.* The simulation is restarted, but with a KI learner that has received 100 randomised, noisy observations of the drift resulting in reduced uncertainty bounds (compare the box size in Fig. 6.17(a) against those in Fig. 6.16(a)). This reduces the conservatism of the plans markedly (Fig. 6.17(a) and Fig. 6.17(b)). With the adaptive element being well trained to compensate for the drift, the executed trajectories match the planned trajectories closely. The reduced conservatism allows the trajectories to arrive close to their goals before eventually (beyond the plotted horizon) slowly parting again.

(Fig. 6.17(b)). This probably was a result of the improved identification result (see Fig. 6.15) afforded by the larger data set. Note, the reduced uncertainty allowed the trajectories to closely approach their goal positions (Fig. 6.17(c)). Since we have tied the agents' cost to the distances to their goal state, this translates to reduced social stage cost.

In conclusion, our simulations have demonstrated that the uncertainty quantifications of our kinky inference method can be successfully employed to facilitate conservative decision making in the context of multi-agent collision avoidance such that guarantees on collision avoidance can be given (whose veracity rests on the prior assumption $f \in \mathcal{K}_{prior}$). As expected, the simulations have illustrated the positive effect increased certainty (via learning) has on the conservatism of the decision-making process and hence, on the social cost (measuring distances to the goals) of the resulting trajectories (see Fig. 6.15(c)).

6.5. Conclusions and future work

In this chapter we have considered multi-agent coordination in discrete-time systems. We have proposed new coordination methods for multi-agent problems: the provably optimal methods CCG and SDCG, as well as our two heuristic auction-flavoured AUC1 and AUC2. While they share the general idea of breaking the large problem into a sequence of tractable pieces with many distributed algorithms across different communities (see references in Sec. 2.3 and Sec. 2.2), our methods provide a viable alternative. Furthermore, we have stated them in the very general framework of optimisation with hard interaction constraints. This generality renders them applicable to a wide range of problem domains, including discrete state problems such as graph planning and multi-commodity flow problems, as well as to continuous state problems such as

model-predictive control. Furthermore, via our general problem statement, we have shown how these problems are examples of a common multi-agent optimisation problem that is normally approached in complete independence by a variety of communities without any recognition of the potential for cross-fertilisation.

Using the graph planning problems as test-bed examples, we went on to demonstrate how our methods significantly improve the performance over popular alternatives.

Connecting to results from Ch. 5, we have derived new stochastic tubes that can be used in MPC with probabilistic collision avoidance constraints. Via the link of optimisation (and in the case ouf our auction methods, via the introduction of the notion of a *virtual obstacle*), we have explained how to apply our coordination methods to such non-convex stochastic MPC problems. Our interest in these problems arose since these are examples of hard planning problems where only limited research on multi-agent coordination has been done. While at present, in RHC control, we can only establish collision avoidance if the the closed-loop trajectories happen to be feasible, our simulations suggest that this tends to be the case in most instances. We are not sure whether this issue is unavoidable, since there will always be a certain probability of ending up in a joint configuration where no feasible plan with bounded control action can be found. This seems to hold true especially for long planning horizons. While sometimes (depending on the size and topology of the freespace), this collision probability could be controlled by adjusting the constraints accordingly, such a modification will increase conservatism. Furthermore, we have found that in practice, the remaining cases of infeasibility could be handled by reducing the tightness of the probabilistic collision avoidance constraints, whenever necessary.

As a final contribution, Sec. 6.4 combines our optimisation-based collision avoidance approach with kinky inference learning (Ch. 4). The result of this merger is a multi-agent collision avoidance method that converts the uncertainty quantifications of kinky inference into robust tubes. The guarantees about conservatism of the learning algorithm (cf. Thm. 4.2.7) translated to guarantees on collision avoidance, provided feasible plans (i.e. control action sequences) can be found. Furthermore, our simulations have illustrated how increased learning experience reduces the conservatism of the coordinated plans. We believe that the merger of bounded learning on the one hand and guaranteed multi-agent collision avoidance on the other, makes this learning-based controller the first of its kind.

Currently the focus of the latter method was on open-loop control. In future work, we would like to test our approach in receding horizon control. Here, we would like to modify our optimisation problem to incorporate results from the robust MPC literature (e.g. [148–150, 206]) that derive constraints with the explicit aim to maintain recursive feasibility.

7. Conclusions

"Anything that has a beginning and an end cannot simultaneously be infinite and everlasting."

Anaximander of Miletus

This work was motivated by the prospect of contributing to the advancement of theoretical underpinnings and methods for (multi-agent) decision-making, learning and inference under uncertainty. In the course of this endeavour, we have addressed a number of hard problems that arise at the intersection of a range of diverse but interconnected disciplines.

The driving factor behind the research was the desire to develop concepts that will help yielding decision making and inference methods that are flexible to learn to adapt to a rich variety of dynamics, while performing inference conservatively in a manner that facilitates safe decision making.

To this end, we have developed several machine-learning frameworks that are sufficiently flexible to identify broad classes of dynamical systems and have been demonstrated to be deployable in learning-based control. In addition to the employment of Bayesian nonparametric machine learning methods, we have also introduced a class of inference rules over function values, which we referred to as *kinky inference* rules, and provided several extensions important to our dynamical system learning applications. Furthermore, we were able to prove a variety of theoretical guarantees. These include (with qualifiers on the noise and bounding functions) (uniform) convergence in the limit of dense data sets as well as conservatism. In the context of control, we have converted the bounded-set quantifications of the uncertainties into robustness and stability guarantees for some of our learning-based control laws. Furthermore, we presented simulations highlighting the efficacy of the kinky inference based controllers. In comparison to learning-based control with Gaussian processes [180, 201], our new controllers proved to offer competitive and more reliable prediction and control performance, while being faster and offering uncertainty quantifications that were easier to translate into closed-loop guarantees.

Another set of contributions was made in multi-agent decision making and coordination which we stated in the general framework of optimisation with interaction constraints. We defined the notion of a collision in general terms. That is, as the violation of an interaction constraint. This means that our methods are very widely applicable beyond any of the scenarios considered in our simulations.

As a particularly challenging application domain, our exposition was consistently accompanied by examples of the non-convex problem of collision avoidance of (physical) agents. Here, we considered coordination both in discrete-time and continuous-time dynamical systems and proposed several solutions. As we have seen, in the continuous-time case, inference helped to ensure that the decisions that were made were guaranteed to avoid collisions with adjustably high certainty even when based on a finite number of computational steps. In both discrete-time and finite-time settings, decision making was intended to be conservative and with safety in mind. That is, even with finite computation any reached joint decision could be guaranteed to satisfy collision-avoidance constraints with adjustably high confidence relative to the given model. To test the methods, we provided simulations in the context of collision avoidance in graphs, multi-commodity flow problems, distributed stochastic model-predictive control as well as in collision-prediction and avoidance in stochastic differential systems.

We concluded the last chapter by showcasing a combination of the different methods developed in the thesis into a multi-agent predictive controller that coordinated learning agents with uncertain beliefs over their dynamics. Utilising the guarantees established for our learning algorithms, the resulting mechanism could provide collision avoidance guarantees relative to the a posteriori epistemic beliefs over the agents' dynamics. Therefore, the result exhibited many of the characteristics of a solution for decision making and inference under uncertainty we set out to achieve: The proved conservatism of the kinky inference rule guaranteed safe decision making, while the learning capabilities of the kinky inference approach granted the method the flexibility to adapt to the uncertain dynamics. Thereby, the level of conservatism could be reduced and yielded better coordination outcomes with increasing learning experience.

8. Future Work

"The secret of being a bore is to tell everything."

Voltaire (1694-1778)

As most theses, this work is an account of a more than three year long journey whose conclusion merely marks the end of a beginning. Every answer we found seemed to spawn a dozen new questions, ideas and opportunities. Throughout the document, we have hinted at numerous extensions, open problems and directions for further exploration. At this point, we have collected a small selection of conceivable extensions of this work that we currently consider of particular interest.

- While this is not a control thesis, we have devised a number of model-learning based control approaches over the course of this work and documented their behaviour in simulations. Most of the problems were chosen to be simple and to serve illustration purposes. In future work, we would find it interesting to investigate some of the our learning based controllers on higher dimensional and possibly real systems. This would require extensions of practical importance such as partial observability of the states and dealing with bounded control. We believe this thesis has derived much of the necessary foundations to this end. For instance, we believe that the problem of bounded control could be attempted via our KI-IMLC approach in combination with planning methods. Here the upper and lower bound functions $\underline{B}$, $\bar{B}$ could be of help. Since inverse models typically require a lot of data, data subset approaches such as improved versions of kNN-KI might become valuable in such cases, especially if we can furnish them with efficient online-learning capabilities. Exploring this and other data subset approaches rigorously is something we certainly intend to do in the future.

  To benefit from inherent non-linearity of the underlying system and to facilitate bounded control, another avenue would be to explore the performance of nonlinear MPC in conjunction with a suitable kinky inference rule.

  In addition, we already have described how to update the basic KI rule in the presence of input uncertainty. This might help in cases where the state is not directly observed (or is noisy). Furthermore, it would be possible to place a separate KI rule over an observational model whose (uncertain outputs) would be fed into the inputs of the original KI state-space inference rule.

- Based on ideas in lieu to our kinky inference method, we already have developed bounded quadrature quadrature and cubature methods. Outstanding is the issue of testing them against alternatives in the context of suitable applications and to place them in the context of related work.

- In optimisation, we have touched upon how Hölder regularity of an objective function yields bounds on optimisation success. Leveraging such bounds might be fruitful in neural network regression where the network's weights are trained via optimisation of a (Hölder continuous) function. In setups where these neural networks are used as the adaptive element of a feedback-controller, can we convert the optimisation bounds to stability guarantees for the resulting closed-loop system ?

- Hölder regularity has been utilised throughout this thesis to draw inferences about a function in the basis of a finite sample. Furthermore, we observe that most interpolation and machine learning rules for regression are based on sub-cases of Hölder continuous functions. It begs the question whether we can derive hardness results. For instance, in our conservative inference framework of Ch. 4, we have already seen that Hölder functions with a large constant L are harder to learn in the sense that they need more training examples to shrink the uncertainty bounds below a given worst-case error. In terms of inference complexity, what can we say about uniformly continuous functions that are not Hölder continuous? We believe we will be able to derive universal approximation guarantees for our KI rule when used in concert with Hölder constant (and Hölder exponent) update rules. However, we would expect that non-Hölder functions might be harder to learn in general.

- As we mentioned at various parts of this work, we would like to understand how the quality of decision making is affected by a computational budget. And, as a related matter, how can we sparsify the data sets optimally ?

- With regard to multi-agent planning, it would be interesting to consider several modifications of our auction-based coordination method. For instance, in Sec. 6.2.2 we gave an example where our lazy auction method in its basic form ended up being incapable of producing a feasible solution. As we pointed out we could investigate extensions that could seek to avoid such situations by allowing agents to re-enter auctions for resources they had lost to other agents in the past.

  In the context of continuous time and space control, it might be possible to combine our ideas of introducing virtual obstacles with path integral control methods [127]. In conjunction with our lazy auction methods, this could give rise to an alternative, fast approach to multi-agent collision avoidance and control in continuous settings.

Bibliography

[1] A. O'Hagan. Some Bayesian Numerical Analysis. *Bayesian Statistics*, 4:345–363, 1992.

[2] R. J. Adler. On excursion sets, tube formulas and maxima of random fields. *Ann. Appl. Prob.*, 2000.

[3] M. Alighanbari and J. P. How. Cooperative task assignment of unmanned aerial vehicles in adversarial environments. In *ACC*, 2005.

[4] T. Alpcan. Dual control with active learning using Gaussian process regression. *Arxiv preprint arXiv:1105.2211*, pages 1–29, 2011.

[5] M. Alvarez, David Luengo, and Neil. D. Lawrence. Latent force models. In *Twelfth International Workshop on Artificial Intelligence and Statistics, JMLR W&CP 5, Clearwater Beach*, 2009.

[6] A.L.Warren and T.E. Marlin. Improved output constraint-handling for MPC with disturbance uncertainty. In *American Control Conference (ACC)*, 2003.

[7] A.L.Warren and T.E. Marlin. Constrained MPC under closed-loop uncertainty. In *American Control Conference (ACC)*, 2004.

[8] R. D Andrea, T. Kalmar-Nagy, P. Ganguly, and M. Babish. The Cornell RoboCup team. *Robot Soccer WorldCup 4, Lecture Notes in Artificial Intelligence*, 2001.

[9] C. Andrieu, N. De Freitas, and A. Doucet. An introduction to MCMC for machine learning. *Machine learning*, pages 5–43, 2003.

[10] A. Argyriou, C. Micchelli, and M. Pontil. Learning convex combinations of continuously parameterized basic kernels. In *COLT*, 2005.

[11] N. Aronszajn. Theory of reproducing kernels. *Trans. of the Americ. Math. Soc.*, 68(3), 1950.

[12] Sunil Arya, David M. Mount, Nathan S. Netanyahu, Ruth Silverman, and Angela Y. Wu. An optimal algorithm for approximate nearest neighbor searching fixed dimensions. *J. ACM*, 45(6), November 1998.

[13] J. Ashayeri, A.J. Westerhof, and P. van Alst. Application of mixed integer programming to a large-scale logistics problem. *International Journal of Production Economics*, 1994.

[14] K. J. Astroem and B. Wittenmark. *Adaptive Control*. Addison-Wesley, 2nd edition, 2013.

[15] N. Ayanian and V. Kumar. Decentralized feedback controllers for multiagent teams in environments with obstacles. *IEEE Trans. on Robotics*, 26(5), 2010.

[16] J.-M. Azais and M. Wschebor. *Level sets and extrema of random processes and fields.* Wiley, 2009.

[17] H. Baessmann and J. Kreyss. *Bildverarbeitung Ad Oculos*. Springer, 4 edition, 2004.

[18] E. Balas. Disjunctive programming and a hierarchy of relaxations for discrete optimization problems. *SIAM J. Discr. Meth.*, 1985.

[19] B. Baran, E. D. Demaine, and D. A. Katz. Optimally adaptive integration of univariate Lipschitz functions. *Algorithmica*, 2008.

[20] D. Bareiss and J. van den Berg. Reciprocal collision avoidance for quadrotor helicopters using LQR-obstacles. In *AAAI-12 Workshop on Multiagent Pathfinding*, 2012.

[21] H. Bauer. *Wahrscheinlichkeitstheorie*. deGruyter, 2001.

[22] R. Becker, S. Zilberstein, V. Lesser, and C. Goldman. Solving transition independent decentralized markov decision processes. *Journal of Artificial Intelligence Research*, 2004.

[23] J. A. Beekman. Asymptotic distributions for the Ornstein-Uhlenbeck proccess. *J. Appl. Prob.*, 12:107–114, 1975.

[24] G. Beliakov. Interpolation of lipschitz functions. *Journal of Computational and Applied Mathematics*, 2006.

[25] G. Beliakov. Smoothing Lipschitz functions. *Optimization Methods and Software*, 2007.

[26] M. Bennewitz, W. Burgard, and S. Thrun. Exploiting constraints during prioritized path planning for teams of mobile robots. In *IROS*, 2001.

[27] M. Bennewitz, W. Burgard, and S. Thrun. Finding and optimizing solvable priority schemes for decoupled path planning techniques for teams of mobile robots. *Robotics and Autonomous Systems*, 2002.

[28] J. L. Bentley. Multidimensional binary search trees used for associative searching. *Com. of ACM*, 1975.

[29] C. Bererton, G. Gordon, S. Thrun, and P. Khosla. Auction mechanism design for multi-robot coordination. In *NIPS*, 2003.

[30] C. A. Bererton. *Multi-Robot Coordination and Competition Using Mixed Integer and Linear Programs*. PhD thesis, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, 2006.

[31] S. M. Berman. *Sojourns and Extremes of Stochastic Processes*. Wadsworth and Brooks / Cole Adv. Books and Software, 1992.

[32] D. Bertismas and J. Tsitsiklis. *Linear Optimization*. Athena Scientific, 1997.

[33] A. Beskos, O. Papaspiliopoulos, and G.O. Roberts. Monte-Carlo maximum likelihood estimation for discretely observed diffusion processes. *Annals of Statistics*, 2009.

[34] S. P. Bhat. Boundedness of orbits and stability of closed sets. *Nonlinear Analysis*, 2009.

[35] S. A. Billings. *Nonlinear System Identification - NARMAX Methods in the Time, Frequency and Spatio-Temporal Domains*. Wiley.

[36] C. M. Bishop. *Pattern recognition and machine learning*. Springer, 2006.

[37] L. Blackmore, H. Li, and B. Williams. A probabilistic approach to optimal robust path planning with obstacles. In *American Control Conference*, 2006.

[38] L. Blackmore, M. Ono, A. Bektassov, and B.C. Williams. A probabilistic particle approach to optimal, robust predictive control. *IEEE Trans. on Robotics*, 2010.

[39] L. Blackmore, M. Ono, and B. C. Williams. Chance-constrained optimal path planning with obstacles. *IEEE Trans. on Robotics*, 2011.

[40] Byron Boots. *Spectral Approaches to Learning Predictive Representations*. PhD thesis, Carnegie Mellon University, 2012.

[41] S. Boucheron, G. Lugosi, and P. Massart. *Concentration Inequalities: A Nonasymptotic Theory of Independence*. Oxford University Press, 2013.

[42] C. Boutilier, M. Goldszmidt, and B. Sabata. Sequential auctions for the allocation of resources with complementarities. In *IJCAI*, 1999.

[43] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[44] D. S. Broomhead and D. Lowe. Multivariable functional interpolation and adaptive networks. *Complex Systems*, 1988. First paper on RBF networks.

[45] V.V. Buldygin and E.D. Pechuk. Inequalities for the distributions of functionals of sub-Gaussian vectors. *Theor. Probability and Math. Statist.*, (80):25–36, 2010.

[46] J. Calliess and G. Gordon. No-regret learning and a mechanism for distributed multiagent planning. Technical report, Carnegie-Mellon University, 2008.

[47] J. Calliess and Geoffrey J. Gordon. No-regret learning and a mechanism for distributed multiagent planning. In *AAMAS*, 2008.

[48] J. Calliess, D. Lyons, and U. Hanebeck. Lazy auctions for multi-robot collision avoidance and motion control under uncertainty. *Springer, LNAI 7068*, 2011.

[49] J. Calliess, D. Lyons, and U. Hanebeck. Lazy auctions for multi-robot collision avoidance and motion control under uncertainty. Technical Report PARG-01-11, Dept. of Engineering Science, University of Oxford, 2011.

[50] J. Calliess, M. Osborne, and S. J. Roberts. Towards auction-based multi-agent collision-avoidance under continuous stochastic dynamics. In *ICML-2012, Workshop on Markets, Mechanisms, and Multi-Agent Models - Examining the Interaction of Machine Learning and Economics*, 2012.

[51] J. Calliess, M. Osborne, and S. J. Roberts. Towards optimization-based multi-agent collision-avoidance under continuous stochastic dynamics. In *AAAI Technical Report WS-12-10*, 2012.

[52] J. Calliess, M. Osborne, and S. J. Roberts. Conservative collision prediction and avoidance for stochastic trajectories in continuous time and space. In *Proc. of the 13th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS)*, 2014.

[53] J. Calliess, A. Papachristodoulou, M. Osborne, and S. J. Roberts. Stochastic processes and feedback-linearisation for online identification and Bayesian adaptive control of fully-actuated mechanical systems. In *WS- Advances in Machine Learning for Sensorimotor Control, NIPS. Also: arXiv:1311.4468*, 2013.

[54] J. Calliess and S. J. Roberts. Multi-agent planning with mixed-integer programming and adaptive interaction constraint generation (ext. abstr.). In *Sixth Annual Symposium on Combinatorial Search (SoCS)*, 2013.

[55] Jan-Peter Calliess, Michael A. Osborne, and Stephen J. Roberts. Conservative collision prediction and avoidance for stochastic trajectories in continuous time and space. *CoRR*, abs/1402.4157, 2014.

[56] M. Cannon, J. Buerger, B. Kouvaritakis, and S. Rakovic. Robust tubes in nonlinear model predictive control. In *IFAC Symp. on Nonlin. Control Sys,*, 2010.

[57] M. Cannon, B. Kouvaritakis, and X. Wu. Probabilistic constrained mpc for multiplicative and additive stochastic uncertainty. *Trans. on Automatic Control*, 2009.

[58] M Cap, P Novak, M Selecky, J. Faigl, and J. Vokrinek. Asynchronous decentralized prioritized planning for coordination in multi-robot system. In *IROS*, pages 3822–3829, 2013.

[59] D. M. Carpintero. *Strategies in Robust and Stochastic Model Predictive Control*. PhD thesis, University of Oxford, 2014.

[60] K. M. Chai, C. K. I. Williams, S. Klanke, and S. Vijayakumar. Multi-task Gaussian process learning of robot inverse dynamics. In *NIPS*, 2008.

[61] D.E. Chang, S.C. Shadden, J.E. Marsden, and R. Olfati-Saber. Collision avoidance for multiple agent systems. In *Decision and Control, 2003. Proceedings. 42nd IEEE Conference on*, volume 1, pages 539–543. IEEE, 2003.

[62] Q. Cheng. *Robust and Stochastic Model Predictive Control*. PhD thesis, University of Oxford, 2012.

[63] G. Cho, G. Chowdhary, A. Kingravi, J. P. . How, and A. Vela. A Bayesian nonparametric approach to adaptive control using Gaussian processes. In *CDC*, 2013.

[64] J. Choe and N. B. Schroff. On the supremum distribution of integrated stationary Gaussian processes with negative linear drift. *Advances in Applied Probability*, 1999.

[65] G. Chowdhary, M. Muhlegg, and J. P. How. Concurrent learning adaptive model predictive control. In *European Aerospace GNC Conference*, 2013.

[66] Girish Chowdhary, H.A. Kingravi, J.P. How, and P.A. Vela. Bayesian nonparametric adaptive control using Gaussian processes. Technical report, MIT, 2013.

[67] Girish Chowdhary, Hassan A. Kingravi, Jonathan How, and Patricio A. Vela. Nonparametric adaptive control of time-varying systems using Gaussian processes. In *American Control Conference (ACC)*, 2013.

[68] E. Clarke. Multipart pricing of public goods. *Public Choice*, 8:19–33, 1971.

[69] D. A. Cooper. Learning lipschitz functions. *Int. J. Computer Math.*, 59:15–26, 1995.

[70] D. A. Cooper. Learning C^2 and Hoelder functions. *International Journal of Pure and Applied Mathematics*, 2006.

[71] T.H Cormen, C.E. Leierson, R.L. Rivest, and C. Stein. *Introduction to Algorithms*. MIT Press, 2ns edition, 2001.

[72] L. Csato and M. Opper. Sparse on-line Gaussian processes. *Neural Computation*, 2002.

[73] F. Curbera. Optimal integration of lipschitz functions with a Gaussian weight. *Journal of Complexity*, 1998.

[74] G. B. Dantzig and P. Wolfe. Decomposition principle for linear programs. *Oper. Res.*, 8:101–111, 1960.

[75] R.K. Dash, N.R. Jennings, and D. C. Parkes. Computational mechanism design: A call to arms. *IEEE Int. Syst.*, 2003.

[76] B. de Wilde, A. W. ter Mors, and C. Witteveen. Push and rotate: a complete multi-agent pathfinding algorithm. *JAIR*, 2014.

[77] M. P. Deisenroth, G. Neumann, and J. Peters. A survey on policy search for robotics. *Foundations and Trends in Robotics*, 2013.

[78] MP Deisenroth, J. Peters, and C. E. Rasmussen. Approximate dynamic programming with Gaussian processes. *ACC*, June 2008.

[79] M.P. Deisenroth, C. E. Rasmussen, and J. Peters. Gaussian process dynamic programming. *Neurocomputing*, 2009.

[80] M.P. Deisenroth and C.E. Rasmussen. Pilco : A model-based and data-efficient approach to policy search. In *ICML*, 2011.

[81] S. Dereich, T. Mueller-Gronbach, and K. Ritter. Infinite-dimensional quadrature and quantization. *Arxiv:math/060124v1*, 2006.

[82] M. B. Dias, R. M. Zlot, N. Kalra, and A. Stentz. Market-based multirobot coordination: a survey and analysis. *Proceedings of the IEEE*, 94(7):1257 – 1270, 2006.

[83] E. W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1959.

[84] D.V. Dimarogonas, S.G. Loizou, K.J. Kyriakopoulos, and M.M. Zavlanos. A feedback stabilization and collision avoidance scheme for multiple independent non-point agents. *Automatica*, 42(2):229–243, 2006.

[85] J. L. Doob. *Stochastic Processes*. Wiley, 1953.

[86] Aaron D'Souza, Sethu Vijayakumar, and Stefan Schaal. Learning inverse kinematics. In *IROS*, 2001.

[87] T.E. Duncan and B.Pasik-Duncan. Adaptive control of a scalar linear stochastic system with a fractional brownian motion. In *FAC World Congress*, 2008.

[88] D. K. Duvenaud. *Automatic Model Construction with Gaussian Processes*. PhD thesis, Cambridge University, 2014.

[89] M. G. Earl and R. D'Andrea. Iterative MILP Methods for Vehicle-Control Problems. *IEEE Trans. on Robotics*, 2005.

[90] M. G. Earl and R. D'Andrea. Multi-Vehicle Cooperative Control Using Mixed Integer Linear Programming. *eprint arXiv:cs/0501092*, 2005.

[91] M.G. Earl and R. D'Andrea. Modeling and control of a multi-agent system using mixed integer linear programming. In *CDC*, 2002.

[92] C. Earls and G. Hooker. Bayesian covariance estimation and inference in latent Gaussian process models. *Journal of Statistical Methodology*, 18, 2014.

[93] J. Elseberg, S. Magnenat, R. Siegwart, and A. N$\acute{}$uchter. Comparison of nearest-neighbor-search strategies and implementations for efficient shape registration. *Journal of Software Engineering for Robotics*, 2012.

[94] M. A. Erdmann and T. Lozano-Perez. On multiple moving objects. *Algorithmica*, 1987.

[95] M. Evans, M. Cannon, and B. Kouvaritakis. Robust mpc for linear systems with bounded multiplicative uncertainty. In *Conf. on Decision and Control (CDC).*, 2012.

[96] A. Farinelli, A. Rogers, and N. Jennings. Decentralised coordination of low-power embedded devices using the max-sum algorithm. In *AAMAS*, 2008.

[97] A. Farinelli, A. Rogers, and N. Jennings. Coordination using the max-sum algorithm. In *IJCAI-09 Workshop on Distributed Constraint Reasoning (DCR)*, 2009.

[98] P. A. Flach and A. C. Kakas. On the relation between abduction and inductive learning. In J. Fagerberg, D.C. Mowery, and R.R. Nelson, editors, *Handbook of defeasible resaoning and uncertainty management systems. Abductive reasoning and learning*, volume 4, chapter 1, pages 1–33. Kluwer Academic Publishers, 2000.

[99] D. A. Forsyth and J. Ponce. *Computer Vision: A modern approach*. Prentice Hall, 2002.

[100] J. H. Friedman, J. L. Bentley, and R. A. Finkel. An algorithm for finding best matches in logarithmic expected time. *ACM Transactions on Mathematical Software*, 1976.

[101] C. Gardiner. *Stochastic Methods-A Handbook for the Natural and Social Sciences*. Springer, 4th edition, 2009.

[102] M. R. Garey and D. S. Johnson. *Computers and Intractability: A guide to the Theory of NP-Completeness*. Freeman and Company, 1979.

[103] B. Gerkey and M. Mataric. Sold!: Auction methods for multirobot coordination. *IEEE Transactions on Robotics and Automation*, 19(5):758–768, 2002.

[104] M. N. Gibbs and D. J. C. MacKay. Efficient implementation of Gaussian processes. Preprint, 1997.

[105] D. Gilat. Every nonnegative submartingale is the absolute value of a martingale. *Ann. Prob.*, 5(5):475–481, 1977.

[106] A. Girard, C.E. Rasmussen, J. Q. Candela, and R. Murray-Smith. Multi-step ahead prediction for non-linear dynamic systems- a Gaussian process treatment with propagation of the uncertainty. In *Advances in Neural Information Processing Systems*, 2003.

[107] C. Goldman and S. Zilberstein. Optimizing information exchange in cooperative multi-agent systems. In *AAMAS*, 2003.

[108] G. Goodwin, M. Seron, and J. de Donȧ. *Constrained Control and Estimation*. Springer, 1 edition, 2005.

[109] G. Gordon, S. A. Hong, and M. Dudik. First-order mixed integer linear programming. In *UAI*, 2009.

[110] G. Grimmet and D. Stirzaker. *Probaility and Random Processes*. Oxford University Press, 3rd edition, 2001.

[111] T. Groves. Incentives in teams. *Econometrica*, 41, 1973.

[112] C. Guestrin and G. Gordon. Distributed planning in hierarchical factored MDPs. In *UAI*, 2002.

[113] Gurobi. Gurobi solver online resources. *http://www.gurobi.com/*.

[114] D. Habib, H. Jamal, and S. A. Khan. Employing multiple unmanned aerial vehicles for co-operative path planning. *International Journal of Advanced Robotic Systems*, 2013.

[115] W. M. Haddad and V. Chellaboina. *Nonlinear Dynamical Systems and Control - A Lyapunov-Based Approach.* Princeton University Press., 2008.

[116] H.Grimmett, R.Paul, R. Triebel, and I.Posner. Knowing when we don't know: Introspective classification for mission-critical decision making. In *ICRA*, 2013.

[117] S.A. Hong and G. Gordon. Optimal distributed market-based planning for multi-agent systems with shared resources. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2011.

[118] A.S. Hurn and K.A. Lindsay. Estimating the parameters of stochastic differential equations by monte carlo methods. *Mathematics and Computers in Simulation.*, 1997.

[119] ILOG CPLEX IBM Software. *ILOG CPLEX User's Manual*, 2010.

[120] P. Indyk and R. Motwani. Approximate nearest neighbors: Towards removing the curse of dimensionality. In *30th Symposium on Theory of Computing*, 1998.

[121] D. Lyons J. Calliess and U. Hanebeck. Lazy auctions for multi-robot collision avoidance and motion control under uncterainty. Technical Report PARG-11-01, University of Oxford, 2011.

[122] M. Osborne J. Calliess and S. J. Roberts. Nonlinear adaptive hybrid control by combining Gaussian process system identification with classical control laws. In *WS -Novel Methods for Learning and Optimization of Control Policies and Trajectories for Robotics, ICRA*, 2013.

[123] E. T. Jaynes. *Probability Theory: The Logic of Science*. Cambridge U. Press, 2003.

[124] D.R. Jones, C.D. Peritunen, and B.E. Stuckman. Lipschitzian optimization without the Lipschitz constant. *JOTA*, 79(1), 1991.

[125] K. Königsberger. *Analysis 2*. Springer, 2000.

[126] A. Kaban. Non-parametric detection of meaningless distance in high dimensional data. *Stat. Comput.*, 2012.

[127] H. J. Kappen. A linear theory for control of non-linear stochastic systems. *Physical Review Letters*, 2005.

[128] M. M. Khorshid, R. C. Holte, and N. Sturtevant. A polynomial-time algorithm for non-optimal multi-agent pathfinding. In *Symposium on Combinatorial Search (SoCS)*, 2011.

[129] Y. H. Kim and F. Lewis. High-level feedback control with neural networks. *Robotics and Intelligent Systems*, 1998.

[130] G.S. Kimeldorf and G. Wahba. Some results on Tchebycheffian spline functions. *J. Math. Anal. Applic.*, 1971.

[131] H. A. Kingravi. *Reduced-Set Models for Improving the training and execution speed of kernel methods*. PhD thesis, Georgia Institute of Technology, 2014.

[132] Paul Klemperer. Auction Theory: A Guide to the Literature. *Journal of Economic Surveys*, 13(3):227–286, July 1999.

[133] P. E. Kloeden and E. Platen. *Numerical solution of stochastic differential equations.* Springer, 1992.

[134] M. Kloft, U. Brefeld, S. Sonnenburg adn P. Laskov, K-R Mueller, and A. Zien. Efficient and accurate and accurate lp-norm multiple kernel learning. In *NIPS*, 2010.

[135] J. Ko, D. Klein, D. Fox, and D. Haehnel. Gaussian Processes and Reinforcement Learning for Identification and Control of an Autonomous Blimp. In *ICRA*, 2007.

[136] J. Kocijan, A. Girard, B. Banko, and R. Murray-Smith. Dynamic systems identification with Gaussian process. *Mathematical and Computer Modelling of Dynamical Systems*, 11:411–424, 2005.

[137] J. Kocijan and R. Murray-Smith. Nonlinear Predictive Control with a Gaussian. *Lecture Notes in Computer Science 3355, Springer*, pages 185–200, 2005.

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

[139] Sven Koenig, Craig Tovey, X. Zheng, and I. Sungur. Sequential bundle-bid single-sale auction algorithms for decentralized control. In *IJCAI*, 2007.

[140] E. Kofman, H. Haimovich, and M. M. Seron. A systematic method to obtain ultimate bounds for perturbed systems. *Internation Journal of Control*, 2006.

[141] D. M. Kornhauser. Coordinating pebble motion on graphs, the diameter of permutation groups, and applications. Technical report, MIT, 1984.

[142] D. Kostic, S. Adinandra, J. Caarls, and H. Nijmeijer. Collision-free motion coordination of unicycle multi-agent systems. In *American Control Conference (ACC), 2010*, pages 3186–3191. IEEE, 2010.

[143] M.V. Kothare, V. Balakrishnan, and M. Morari. Robust constrained model predictive control using linear matrix inequalities. *Automatica*, 1996.

[144] B. Kouvaritakis and M. Cannon. *Encyclopedia of Systems and Control*, chapter Stochastic Model Predictive Control. Springer, 2014.

[145] B. Kulis. Metric learning: A survey. *Foundations and Trends in Machine Learning.*, 2012.

[146] P. R. Kumar. A survey of some results in stochastic adaptive control. *Siam J. Control and Optimization*, 23, 1985.

[147] H. J. Kushner. *Stochastic Stability and Control*. Academic Press, 1967.

[148] Y. Kuwata, T. Schouwenaars, A. Richards, and J. How. Robust constrained receding horizon control for trajectory planning. In *AIAA Guidance, Navigation and Control Conference Exhibit*, 2005.

[149] Yoshiaki Kuwata and Jonathan P. How. Cooperative distributed robust trajectory optimization using receding horizon MILP. *IEEE Trans. on Control Systems Technology*, 2011.

[150] Yoshiaki Kuwata, Arthur Richards, Tom Schouwenaars, and Jonathan P. How. Decentralized robust receding horizon control for multi-vehicle guidance. In *ACC*, 2006.

[151] M. Lagoudakis, V. Markakis, D. Kempe, P. Keskinocak, S. Koenig, A. Kleywegt, C. Tovey, A. Meyerson, and S. Jain. Auction-based multi-robot routing. In *Int. Conf. on Robotics: Science and Systems*, 2005.

[152] W. Langson. Robust model predictive control using tubes. Technical Report PCS 2000/9/1, Imperial College., 2000.

[153] N. D. Lawrence, G.. Sanguinetti, and M. Rattray. Modelling transcriptional regulation using Gaussian processes. In *Advances in Neural Information Processing Systems (NIPS)*, 2007.

[154] Y. Li and K. Gupta. Motion planning of multiple agents in virtual environments on parallel architectures. In *ICRA*, 2007.

[155] Nick Littlestone and Manfred K. Warmuth. The weighted majority algorithm. In *IEEE Symposium on Foundations of Computer Science*, pages 256–261, 1989.

[156] L. Ljung. *System identification - theory for the user*. Prentice Hall, 1999.

[157] Lennart Ljung. Perspectives on system identification. *Annual Reviews in Control*, 34(1):1 – 12, 2010.

[158] D. Lyons, J. Calliess, and U. Hanebeck. Chance constrained model predictive control for multi-agent systems with coupling constraints. In *American Control Conference (ACC)*, 2012.

[159] D. Lyons, J.P. Calliess, and U.D. Hanebeck. Chance-constrained Model Predictive Control for Multi-Agent Systems. *Arxiv preprint arXiv:1104.5384*, 2011.

[160] S. M. J. Lyons, S. Saerkkae, and A. Storkey. The coloured noise expansion and parameter estimation of diffusion processes. In *NIPS*, 2012.

[161] J. Maciejowski. *Predictive Control with Constraints*. Prentice Hall, 2002.

[162] C. Makkar, W. E. Dixon, W. G. Sawyer, and G. Hu. Lyapunov-based tracking control in the presence of uncertain nonlinear parameterizable friction. In *ACC*, 2005.

[163] R. Mann, R. Freeman, M. A. Osborne, R. Garnett, J. Meade, C. Armstrong, D. Biro, T. Guilford, and S. J. Roberts. Gaussian processes for prediction of homing pigeon flight trajectories. In *Bayesian Inference and Maximum Entropy Methods in Science and Engineering, AIP Conf.*, 2009.

[164] S. Mastellone, D.M. Stipanović, C.R. Graunke, K.A. Intlekofer, and M.W. Spong. Formation control and collision avoidance for multi-agent non-holonomic systems: Theory and experiments. *The International Journal of Robotics Research*, 27(1):107–126, 2008.

[165] Matthew G. Earl and Raffaello D'Andrea. Multi-vehicle cooperative control using mixed integer linear programming. *arXiv:cs/0501092v1 [cs.RO]*, 2005.

[166] D. Q. Mayne, M.M. Seron, and S.V. Rakovic. Robust model predictive control of constrained linear systems with bounded disturbances. *Automatica*, 2005.

[167] D.Q. Mayne and W. Langson. Robustifying model predictive control of constrained linear systems. *Electronics Letters*, 2001.

[168] P. Meyer. A decomposition theorem for supermartingales. *Illinois Journal of Mathematics*, 6:193–205, 1962.

[169] P. Meyer. Decomposition of supermartingales: the uniqueness theorem. *Illinois Journal of Mathematics*, 7:1–17, 1963.

[170] C. A. Micchelli. Universal kernels. *J. of Machine Learning Research*, 2006.

[171] T. Mitchell. *Machine Learning*. Mc Graw Hill, 1997.

[172] M.M. Monahemi and M. Krstic. Control of wingrock motion using adaptive feedback linearization. *J. of. Guidance Control and Dynamics.*, 1996.

[173] Roderick Murray-smith, Carl Edward Rasmussen, and Agathe Girard. Gaussian Process Model Based Predictive Control. In *IEEE Eurocon 2003: The International Conference on Computer as a Tool*, 2003.

[174] R. Nair, M. Tambe, M. Yokoo, D. Pynadath, and S. Marsella. Taming decentralized pomdps: Towards efficient policy computation for multiagent settings. In *IJCAI*, 2003.

[175] R. Nair, P. Varakantham, M. Tambe, and M. Yokoo. Network dis- tributed pomdps: A synthesis of distributed contstraint optimiza- tion and pomdps. In *Proc. of the 20th National Conference on Artificial Intelligence*, 2005.

[176] J. Nakanishi, J. Farrell, and S. Schaal. Composite adaptive control with locally weighted statistical learning. *Neural networks*, 2005.

[177] K. S. Narendra and A. M. Annaswamy. *Stable Adaptive Systems*. Prentice Hall, 1989.

[178] R. R. Negenborn and J.M. Maestre. Distributed model predicitive control. *IEEE Control Systems Magazine*, 2014.

[179] D. Nguyen-Tuong and J. Peters. Using model knowledge for learning inverse dynamics. In *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2010.

[180] D Nguyen-Tuong and J. Peters. Model learning for robot control: a survey. *Cognitive processing*, 2011.

[181] D. Nguyen-Tuong, J. Peters, M. Seeger, and B. Schölkopf. Learning inverse dynamics: a comparison. In *Europ. Symp. on Artif. Neural Netw.*, 2008.

[182] D. Nguyen-Tuong, M. Seeger, and J. Peters. Model learning with local Gaussian process regression. *Advanced Robotics*, 2009.

[183] Duy Nguyen-Tuong and Jan Peters. Online kernel-based learning for task-space tracking robot control. *IEEE Trans. on Neural Networks and Learning Systems.*, 2012.

[184] T. Nishi, M. Ando, and Masami Konishi. Distributed route planning for multiple robots using an augmented lagrangian decomposition and coordination technique. *IEEE Trans. on Robotics*, 2005.

[185] T. Nishi, M. Ando, and Masami Konishi. Experimental studies on a local rescheduling procedure for dynamic routing of autonomous decentralized agv systems. *Robotics and Computer-Integr. Manuf.*, 2006.

[186] A. O'Hagan. Bayes-hermite quadrature. *J. of Stat. Planning and Inference*, 29, 1991.

[187] S. C. W. Ong, S.W. Png, D. Hsu, and W. S. Lee. Planning under uncertainty for robotic tasks with mixed observability. *Int. J. of Robotics Research*, 2010.

[188] M. Ono, B. C. B. C. Williams, and L. Blackmore. Probabilistic planning for continuous dynamic systems under bounded risk. *Journal of Artificial Intelligence Research*, 2013.

[189] M.A. Osborne, R. Garnett, and S.J. Roberts. Gaussian processes for global optimization. *3rd International Conference on Learning and Intelligent Optimization (LION3)*, 2009.

[190] M.A. Osborne, S.J. Roberts, A. Rogers, and N. Jennings. Real-time information processing of environmental sensor network data using Bayesian Gaussian processes. *ACM Transactions on Sensor Networks*, 2011.

[191] Michael Osborne. *Bayesian Gaussian Processes for Sequential Prediction , Optimisation and Quadrature Sequential Prediction , Optimisation and Quadrature*. PhD thesis, Oxford University, 2010.

[192] Ioannou P. and J. Sun. *Robust Adaptive Control*. Prentice Hall, 1995.

[193] L. Pallottino, V. G. Scordio, E. Frazzoli, and A. Bicchi. Decentralized cooperative policy for conflict resolution in multi-vehicle systems. *IEEE Trans. on Robotics*, 23(6):1170–1183, 2007.

[194] D. Parsons and J. Canny. A motion planner for multiple mobile robots. In *ICRA*, 1990.

[195] F. Perez-Cruz, S. Van Vaerenbergh, J.J. Murillo-Fuentes, M. Lazaro-Gredilla, and I. Santamaria. Gaussian Processes for Nonlinear Signal Processing: An Overview of Recent Advances. *Signal Processing Magazine, IEEE*, 2013.

[196] Dejan Petelin and J. Kocijan. Control system with evolving Gaussian process models. In *Evolving and Adaptive Intelligent Systems (EAIS), 2011 IEEE Workshop on*. IEEE, 2011.

[197] J. Peters and S. Schaal. Learning opertational space control. In *Robotics: Science and Systems (RSS)*, 2006.

[198] J. G. Proakis and D. K Manolakis. *Digital Signal Processing*. Pearson, 4th edition, 2006.

[199] I. Prodan, S. Olaru, and C. Soica. Predicitive control for tight group formation of multi-agent systems. In *IFAC World Congress*, 2011.

[200] Vlastimil Ptak. Spectral Radius, Norms of Iterations, and the Critical Exponent. *Linear Algebra and its Applications*, 1:246–260, 1968.

[201] C.E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006.

[202] K. Rawlik, M. Toussaint, and S. Vijayakumar. On stochastic optimal control and reinforcement learning by approximate inference. In *RSS*, 2012.

[203] S. Reece and S. Roberts. An Introduction to Gaussian Processes for the Kalman Filter Expert. In *Fusion*, 2010.

[204] S. Reece, S. Roberts, S. Ghosh, A. Rogers, and N. Jennings. Efficient state-space inference of periodic latent force models. *Journal of Machine Learning Research*, 2014.

[205] A. Richards and J. How. Mixed-integer programming for control. In *ACC*, 2006.

[206] A. Richards and J. How. Robust variable horizon model predictive control for vehicle maneuvering. *Int. J. of Robust and Nonlinear Control*, 2007.

[207] O. Rivasplata. Subgaussian random variables: An expositionary note. 2012.

[208] A. Rogers, A. Farinelli, R. Stranders, and N.R. Jennings. Bounded approximate decentralised coordination via the max-sum algorithm. *Artificial Intelligence*, 175(2):730–759, February 2011.

[209] Alex Rogers, Sasan Maleki, Siddhartha Ghosh, and N.R. Jennings. Adaptive Home Heating Control Through Gaussian Process Prediction and Mathematical Programming. In *2nd Int. Workshop on Agent Technology for Energy Systems (ATES 2011)*, 2011.

[210] L.C.G. Rogers and D. Williams. *Diffusions, Markov Processes And Martingales*, volume 1. Cambridge U. Press, 2 edition, 2000.

[211] D. Roth and W. Yih. Integer linear programming inference for conditional random fields. In *ICML*, 2005.

[212] A. Rottmann and W. Burgard. Adaptive Autonomous Control using Online Value Iteration with Gaussian Processes. In *ICRA*, 2009.

[213] S. Russel and P. Norvig. *Artificial Intelligence - A Modern Approach*. Prentice Hall, 3rd edition, 2009.

[214] A. A. Saad. *Simulation and Analysis of wing rock physics for a generic fighter model with three degrees of freedom.* PhD thesis, Air Force Institute of Technology, Air University, 2000.

[215] S. Saerkkae and J. Hartikainen. Infinite-Diemnsional Kalman Filtering Approach to Spatio-Temporal Gaussian Process Regression. In *AISTATS*, 2012.

[216] N. Sandell, P. Varaiya, M. Athans, and M. Safonov. Survey of decentralized control methods for large scale systems. *Transactions on Automatic Control*, 1978.

[217] T. Sandholm. Algorithm for optimal winner determination on combinatorial auctions. *Artif. Int.*, 2002.

[218] R. Sanner and J.-J. Slotine. Gaussian networks for direct adaptive control. *Trans. on Neural Networks*, 1992.

[219] J. G. Saw, M.C. K. Yang, and T. C. Mo. Chebyshev inequality with estimated mean and variance. *Am. Statistn.*, 1984.

[220] D. Sbarbaro and R. Murray-Smith. Nonlinear adaptive control using non-parametric Gaussian Process prior models . In *15th Triennial World Congress of IFAC*, 2002.

[221] S. Schaal, C. G. Atkeson, and S. Vijayakumar. Scalable techniques from nonparametric statis- tics for real-time robot learning. *Applied Intelligence*, 2002.

[222] T. Schouwenaars, B. De Moor, E. Feron, and J. How. Mixed integer programming for multi-vehicle path planning. In *European Control Conference*, 2001.

[223] J. Schuurmans and J. A. Rossiter. Robust predictive control using tight sets of predicted states. In *IEEE Proceedings: Control Theory and Applications*, 2000.

[224] A.T. Schwarm and M. Nikolaou. Chance-constrained model predictive control. *AIChE Journal*, 1999.

[225] M. Seeger. *Bayesian Gaussian Process Models: PAC-Bayesian Generalisation Error Bounds and Sparse Approximations*. PhD thesis, University of Edinburgh, 2005.

[226] M. Seeger. Low rank update for the cholesky decomposition. Technical report, University of California at Berkeley, 2008.

[227] Jeff S. Shamma, editor. *Cooperative Control of Distributed Multi-Agent Systems*. Wiley, 2007.

[228] Y. Shoham and K. Leyton-Brown. *Multiagent Systems: Algorithmic, Game-Theoretic, and Logical Foundations*. Cambridge U. Press, 2009.

[229] Y. Shoham, R. Powers, and T. Grenager. If multi-agent learning is the answer, what is the question? *Artif. Intell.*, 171:365–377, 2007.

[230] B. Shubert. A sequential method seeking the global maximum of a function. *SIAM J. on Numerical Analysis*, 9, 1972.

[231] Alex Smola, Arthur Gretton, Le Song, and Bernhard Sch. A Hilbert Space Embedding for Distributions. In *Conf. on Algorithmic Learning Theory*, pages 1–20, 2007.

[232] T. T. Soong. *Random Differential Equations*. Academic Press, 1973.

[233] M. W. Spong. Partial feedback linearization of underactuated mechanical systems. In *Proc. IEEE Int. Conf. on Intel. Robots and Sys. (IROS)*, 1994.

[234] M.W. Spong, S. Hutchinson, and M. Vidyasagar. *Robot Dynamics and Control*. Wiley and Sons, 2006.

[235] N. Srinivas, A. Krause, S. Kakade, and M. Seeger. Gaussian Process Optimization in the Bandit Setting : No Regret and Experimental Design. In *ICML*, 2010.

[236] B. K. Sriperumbudur, K. Fukumizu, and G. R. G. Lanckriet. On the relation between universality, characteristic kern els and rkhs embedding of measures. In *AISTATS*, 2010.

[237] Trevor Standley. Finding Optimal Solutions to Cooperative Pathfinding Problems. In *AAAI*, 2010.

[238] M.L. Stein. *Statistical Interpolation of Spatial Data: Some Theory for Kriging.* Springer, 1999.

[239] F. Steinke and B. Schoelkopf. Kernels, regularization and differential equations. *Pattern Recognition*, 41:3271–3286, 2008.

[240] I. Steinwart. On the influence of the kernel on the consistency of support vector machines. *JMLR*, 2001.

[241] A. Stentz and M. B. Dias. A free market architecture for coordinating multiple robots. Technical Report CMU-RI-TR-99-42, Carnegie Mellon, 1999.

[242] D.M. Stipanović, P.F. Hokayem, M.W. Spong, D.D. Šiljak, et al. Cooperative avoidance control for multiagent systems. *Journal of Dynamic Systems, Measurement, and Control*, 129:699, 2007.

[243] A. J. Storkey. Truncated covariance matrices and Toeplitz methods in Gaussian processes. In *ICANN'99: Artificial Neural Networks*, 1999.

[244] P. Surynek. Multi-robot path planning. In *Multi-Robot Systems, Trends and Development*. InTech-Open Acess Publisher, 2011.

[245] R.S. Sutton and A.G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 1998.

[246] M. Talagrand. The supremum of some canonical processes. *Am. J. Math.*, 1994.

[247] M. Talagrand. *The Generic Chaining: Upper and Lower Bounds of Stochastic Processes*. Springer, 2005.

[248] Russ Tedrake. Underactuated robotics: Learning, planning, and control for efficient and agile machines. Course Notes for MIT 6.832, 2009.

[249] M. Toussaint, A.J. Storkey, and S. Harmeling. Expectation-maximisation methods for solving (PO) MDPs and optimal control problems. In *Bayesian Time Series Models*. Cambridge University Press, 2011.

[250] C. Tovey, M. Lagoudakis, S. Jain, and S. Koenig. The generation of bidding rules for auction-based robot coordination. In *Multi-Robot Systems: From Swarms to Intelligent Automata*, volume 3, pages 3–14, 2005.

[251] Paul Trodden and Arthur Richards. Robust distributed model predictive control using tubes. In *ACC*, 2006.

[252] L. Valiant. A theory of the learnable. *Communications of the ACM.*, 1984.

[253] J. Van den Berg, M. Lin, and D. Manocha. Reciprocal velocity obstacles for real-time multi-agent navigation. In *ICRA*, 2008.

[254] P. Velagapudi, K. Sycara, and P. Scerri. Decentralized prioritized planning in large multirobot teams. In *IROS'10*, 2010.

[255] P. Velagapudi, K. P. Sycara, and P. Scerri. Decentralized prioritized planning in large multirobot teams. In *IROS*, 2010.

[256] W. Vickrey. Counterspectulations, auctions and competitive sealed tenders. *J. of Finance*, 16:8–37, 1961.

[257] S. Vijayakumar and S. Schaal. Locally weighted projection regression : An O(n) algorithm for incremental real time learning in high dimensional space. In *ICML*, 2000.

[258] K.Y. Volyanskyy, M.M. Haddad, and A.J. Calise. A new neuroadaptive control architecture for nonlinear uncertain dynamical systems: Beyond sigma- and e-modifications. In *CDC*, 2008.

[259] Thomas Vossen, Michael Ball, and Robert H. Smith. On the use of integer programming models in ai planning. In *IJCAI*, 1999.

[260] G. Wahaba. Spline models for observational data. *SIAM*, 1990.

[261] L. Wasserman. *All of Nonparametric Statistics*. Springer, 2007.

[262] N. Weaver. *Lipschitz Algebras*. World Scientific, 1999.

[263] P. Whittle. A multivariate generalization of Tchebichev's inequality. *Quarterly Journal of Mathem.*, 1958.

[264] C. K. I. Williams and C. E. Rasmussen. Gaussian processes for regression. In *NIPS*, 1996.

[265] Z. Yan, N. Jouandeau, and A. A. Cherif. A survey and analysis of multi-robot coordination. *International Journal of Advanced Robotic Systems*, 2013.

[266] W. Yeoh, A. Felner, and S. Koenig. BnB-ADOPT: An asynchronous branch-and-bound DCOP algorithm. *Journal of Artificial Intelligence Research*, 2010.

[267] William Yeoh. *Speeding up distributed constraint optimization search algorithms*. PhD thesis, Univ. of Southern California, 2010.

[268] Makoto Yokoo, Edmund H. Durfee, Toru Ishida, and Kazuhiro Kuwabara. The distributed constraint satisfaction problem: Formalization and algorithms. *IEEE Trans. on Knowledge and Data Engineering.*, 1998.

[269] I. Young and L. J. van Vliet. Recursive implementation of the Gaussian filter. *Signal Processing*, 1995.

[270] N. Young. *An Introduction to Hilbert Space*. Cambridge U. Press, 1988.

[271] N. J. Young. Norm and spectral radius for algebraic elements of a Banach algebra. *Math. Proc. Camb. Phil. Soc.*, 88:129–133, 1980.

[272] N. J. Young. The rate of convergence of a matrix power series. *Linear Algebra and its Applications*, 1981.

[273] Jingjin Yu and Steven M Lavalle. Time optimal multi-agent path planning on graphs. *Multiagent Pathfinding AAAI Technical Report WS-12-10*, 1(0), 2012.

[274] Jingjin Yu and Steven M Lavalle. Planning optimal paths for multiple robots on graphs. *Arxiv preprint arXiv:204.3830v4*, 2013.

[275] Z. B. Zabinsky, R. L. Smith, and B. P. Kristinsdottir. Optimal estimation of univariate black-box Lipschitz functions with upper and lower bounds. *Computers and Operations Research*, 2003.

[276] Roie Zivan and Hilla Peled. Max/min-sum distributed constraint optimization through value propagation on an alternating DAG. In *AAMAS*, 2012.

A. Critical matrix exponents and norm bounds on states defined by linear recurrences

*L*et $\mathcal{X}$ be a finite-dimensional vector space over field $\mathbb{F} \in \{\mathbb{R}, \mathbb{C}\}$ and $A \in \mathcal{L}(\mathcal{X})$ be a linear operator on $\mathcal{X}$. Upper bounds on the critical operator exponent of a linear operator $A \in \mathcal{L}(\mathcal{X})$ with spectral norm $|||A||| \geq 1$ but spectral radius $\rho(A) < 1$ are derived. That is, we find k_0 such that $\left|\left|\left|A^k\right|\right|\right| < 1, \forall k \geq k_0$. In addition, we derive an anytime algorithm that refines these bounds. As a by-product, we give a conservative upper bound on $\sup_{k\in\mathbb{N}} \left|\left|\left|A^k\right|\right|\right|$. Our results are of interest especially in high-dimensional vector spaces where the computation of matrix powers is expensive and brute-force search infeasible. As an application, we utilise our results to find (finite-time) bounds for linear recurrence relations that can help to answer various questions in the design of controllers for Euler-approximations of second-order dynamical systems and facilitate the construction of tubes for robust control.

A.1. Introduction

The *critical matrix exponent (CME)* k_0 of matrix $A \in \mathbb{C}^{d\times d}$ is defined as the smallest number $k \in \mathbb{N}$ such that $\left|\left|\left|A^k\right|\right|\right| < 1$. This term is closely related to the notion of critical exponent of a Banach space of linear operators as used by $Pták$ [200].

Critical matrix exponents play an important role in matrix power series and numerical to solutions to Lyapunov matrix equations [272] and have been studied by a variety of authors [272]. It has been shown that the critical exponent for matrices with $|||A||| \leq 1$ equals d. Unfortunately, many applications give rise to difference equations where this assumption is violated. For instance, as discussed in Sec. A.3, recurrence relations arising as discretisations of second-order dynamic systems with stabilising feedback control typically allow one to set the feedback to ensure the transition matrix is discrete-stable, but the constraints of the physical system impose a spectral norm greater than one. As we will see, knowledge of an upper bound of the critical exponent of the transition matrix allows one to answer questions of finite-time stability– a goal that motivated the work presented in what is to follow.

In Sec. A.2, for matrix A, we derive a (concave) criterion function $\phi : \mathbb{R}_+ \to \mathbb{R}$ whose unique maximum is an upper bound of $\sup_{k\in\mathbb{N}} \left|\left|\left|A^k\right|\right|\right|$ and whose unique root yields an upper bound on the critical exponent of A. Due to favourable analytic properties of the criterion function, determination of the maximum can be done in closed-form, yielding the bound on $\sup_{k\in\mathbb{N}} \left|\left|\left|A^k\right|\right|\right|$. In addition, it is easy to construct Newton sequences that not only rapidly converge to the root, but whose elements all provably are upper bounds on the root of ϕ and thereby, on the critical exponent of A. The resulting construction yields an anytime algorithm for upper bounds on the critical exponent. Closed-form bounds are gained simply by choosing an element from a Newton sequence of finite length.

Sec. A.3 illustrates the utility of our results in the context of aforementioned stability analysis of a feedback-controlled discrete-time dynamic system followed by an extension bounding the error in of system dynamics in the context of robust control.

A.2. Determination of upper bounds on the CME and operator exponential extremum

It is our aim to find an upper bound $k_0 \in \mathbb{N}$ on the critical exponent of a matrix $A \in \mathbb{C}^{d\times d}$ with $\rho(A) < 1$. That is, we desire to find $k_0 \in \mathbb{N}$ such that $\forall k \geq k_0 : \left|\left|\left|A^k\right|\right|\right| < 1$. For notational convenience, define $\delta := d - 1, r := \rho(A)$.

It is well known [271] that for any operator (on Hilbert space) that is algebraic of degree d, we have $\left|\left|\left|A^k\right|\right|\right| \leq \binom{k}{\delta} |||A|||^\delta \, r^{k-\delta}, \forall k \geq d$. Utilising $\binom{k}{\delta} \leq \frac{k^\delta}{\delta!}$ and taking the log on both sides yields

$$\log \left|\!\left|\!\left| A^k \right|\!\right|\!\right| \leq \delta \log k - \log(\delta!) + \delta \log |\!|\!| A |\!|\!| + (k-\delta)\log(r) =: \phi(k). \tag{A.1}$$

Since the natural logarithm is a strictly monotonically growing function it does not change the ordering of the graph of a positive function. Hence, $\left|\!\left|\!\left| A^k \right|\!\right|\!\right| < 1$ iff $\log \left|\!\left|\!\left| A^k \right|\!\right|\!\right| < 0$. As shown, the latter is the case if $\phi(k) < 0$. Hence, our bound k_0 on the critical value can be found by choosing any $k_0 \in \mathcal{N} := \{\forall k \geq k_0 : \phi(k) < 0\}$.

Writing $\theta := -\log(\delta!) + \delta \log |\!|\!| A |\!|\!| - \delta \log(r)$ and extending ϕ onto the half-open interval $\mathbb{R}_{\geq 0}$, for $t \in \mathbb{R}_{\geq 0}$ we note

$$\phi(t) = \delta \log t + t \log r + \theta \tag{A.2}$$

$$\dot{\phi}(t) = \frac{\delta}{t} + \log r \tag{A.3}$$

$$\ddot{\phi}(t) = -\frac{\delta}{t^2}. \tag{A.4}$$

Remark A.2.1. Inspecting the derivatives, we observe

1. $\dot{\phi}$ decreases strictly monotonically to $\inf_t \dot{\phi}(t) = \log r$.
2. ϕ attains a global maximum at $t_* := -\frac{\delta}{\log r}$, $\max_{t \in \mathbb{R}_+} \phi(t) = \phi(t_*) =: \phi_*$.
3. So, ϕ increases strictly monotonically for $t < t_*$ and decreases strictly monotonically for $t > t_*$.
4. Hence, ϕ has a unique root $\varrho \geq t_*$.
5. The curvature of the graph rapidly vanishes. Consequently, ϕ approximately "behaves like a straight line" for sufficiently large inputs.

Example A.2.2. An illustration for these functions for a matrix $A \in \mathbb{R}^{4\times 4}$ with $|\!|\!| A |\!|\!| = 1.1731$ and $\rho(A) = 0.873$ is depicted in Fig. A.1. As this example illustrates, finding a bound based on $\phi(k) < 0$ can be quite conservative. That is, $\left|\!\left|\!\left| A^k \right|\!\right|\!\right| < 1$ for $k \geq 6$, whereas ϕ attains negative values as late as for values $k \geq 27$.

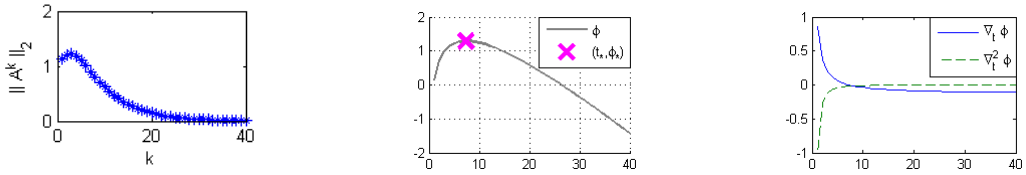


Figure A.1.: Left: Evolution of $k \mapsto \left|\!\left|\!\left| A^k \right|\!\right|\!\right|$. Centre: $\phi(x)$. Right: First and second derivatives of ϕ. Note, how the second derivative rapidly converges to zero causing ϕ to behave like a line after passing through the root. This means that our Newton-step based algorithm will rapidly converge.

As a by-product of our derivations so far, we have obtained the following result:

Theorem A.2.3. *With $r := \rho(A)$, we have*

$$\sup_{k\in\mathbb{N}} \left|\!\left|\!\left| A^k \right|\!\right|\!\right| = \max_{k\in\{1,\dots,k_0\}} \left|\!\left|\!\left| A^k \right|\!\right|\!\right| \leq \exp\big(\phi(\frac{1-d}{\log r})\big) = \frac{1}{(d-1)!}\Big(\frac{1-d}{\log r}\Big)^{d-1} |\!|\!| A |\!|\!|^{d-1} \; r^{\frac{1-d}{\log r}-d+1} < \infty.$$

Proof. (i) Firstly, we show $\sup_{k\in\mathbb{N}} \left|\!\left|\!\left| A^k \right|\!\right|\!\right| = \max_{k\in\{1,\dots,k_0\}} \left|\!\left|\!\left| A^k \right|\!\right|\!\right|$:

For contradiction, assume $k_* \in \arg\sup_{k\in\mathbb{N}} \left|\!\left|\!\left| A^k \right|\!\right|\!\right| \subset \mathbb{N}_0, k_* > k_0$. (Note, the assumption that $k_* \in \mathbb{N}_0$ is valid since ϕ is an upper bound and monotonically decreases, so eventually becomes lower than $|\!|\!| A |\!|\!|$). Hence, $\exists i > 0 : k_* = i + k_0$. Thus, $\left|\!\left|\!\left| A^{k_*} \right|\!\right|\!\right| \leq \left|\!\left|\!\left| A^i \right|\!\right|\!\right| \left|\!\left|\!\left| A^{k_0} \right|\!\right|\!\right| < \left|\!\left|\!\left| A^i \right|\!\right|\!\right| < \infty$. This is in contradiction to the assumption $k_* \in \arg\sup_{k\in\mathbb{N}} \left|\!\left|\!\left| A^k \right|\!\right|\!\right|$.

(ii) By Eq. A.1, we have $\forall k : \left|\left|\left|A^k\right|\right|\right| \leq \exp(\phi(k)) \leq \exp(\max_{k\in\mathbb{N}} \phi(k)) \leq \exp(\max_{t\in\mathbb{R}_+} \phi(k))$. By 2. in Rem. A.2.1, $\max_{t\in\mathbb{R}_+} \phi(t) = \phi(t_*) = \phi(\frac{1-d}{\log \rho(A)})$. Thus, $\forall k : \left|\left|\left|A^k\right|\right|\right| \leq \exp\big(\phi(\frac{1-d}{\log \rho(A)})\big)$. The RHS of this inequality is just the RHS of the inequality in the statement of the theorem. □

The idea, which we will substantiate in what is to follow, is that any sequence generated by Newton's root-finding method will be contained in $\mathcal{N}$ after one iteration, provided that it is initialised with a starting value $t_0 > t_*$. Newton's algorithm thus yields upper bounds on the critical exponent. We will also see that the sequence of bounds becomes increasingly tight.

Definition A.2.4. *Let $f : C^1(T)$ be a differentiable function with non-vanishing derivative on domain $T \subset \mathbb{R}$. A Newton sequence (for f) is a $T-$valued sequence $(t_i)_{i\in\mathbb{N}_0}$ such that $t_{i+1} = t_i - \frac{\phi(t_i)}{\dot\phi(t_i)}, \forall i \in \mathbb{N}_0$ with an increasing number of iterations.*

The next Lemma essentially implies that once a Newton sequence for ϕ is greater than the root of ϕ, it will remain so:

Lemma A.2.5. *Let $(t_i)_{i\in\mathbb{N}_0}$ be a Newton sequence for $\phi : \mathbb{R}_{>0} \to \mathbb{R}$ as defined above with unique root $\varrho \geq t_*$. For any $i \in \mathbb{N}_0$, $t_i > \varrho$ implies $t_i > t_{i+1} > \varrho$. In particular, $\dot\phi(t_j) < 0 \wedge \phi(t_j) < 0, \forall j \geq i$ if $t_i > \varrho$.*

Proof. Let $i \in \mathbb{N}_0$ with $t_i > \varrho$. Hence, $\dot\phi(t_i) < 0$ and $\phi(t_i) < 0$ (cf. Rem. A.2.1). Thus, $\frac{\phi(t_i)}{\dot\phi(t_i)} > 0$ and $t_{i+1} = t_i - \frac{\phi(t_i)}{\dot\phi(t_i)} < t_i$. Next, we show $\varrho < t_{i+1}$. Let $\epsilon > 0$ such that $t_i = \varrho + \epsilon$. As pointed out in Rem. A.2.1 $\dot\phi_{|(t_*,\infty)} \searrow$, which entails $\dot\phi(t) > \dot\phi(t_i), \forall t \in (\varrho, t_i)$. Hence, $\phi(t_i) = \phi(\varrho) + \int_\varrho^{t_i} \dot\phi(t)\,\mathrm{d}t < \int_\varrho^{t_i} \dot\phi(t_i)\,\mathrm{d}t = \epsilon\,\dot\phi(t_i)$. With $\dot\phi(t_i) < 0$ this implies $\epsilon > \frac{\phi(t_i)}{\dot\phi(t_i)}$. Substituting $t_i - \varrho$ for ϵ and rearranging terms yields $\varrho < t_i - \frac{\phi(t_i)}{\dot\phi(t_i)} = t_{i+1}$.

□

Note, the lemma assumes $t_0 > \varrho$. If we did know ϱ then our task of finding an upper bound k_0 would be completed: One could simply choose $k_0 = t_0$. Unfortunately, the root ϱ will in general be unknown to us. However, the next lemma guarantees that it suffices to choose any $t_0 > t_* = -\frac{\delta}{\log r}$ to guarantee that all consecutive t_i are greater than the root.

Lemma A.2.6. *Let $(t_i)_{i\in\mathbb{N}_0}$ be a Newton sequence with initial value $t_0 > t_*, \phi(t_0) \neq 0$ and ϱ the unique root of function ϕ. Then, $t_i \geq \varrho, \forall i \geq 1$. In particular, $\phi(t_i) < 0, \forall i \geq 1$.*

Proof. As before, let $\varrho > t_*$ denote the root of ϕ. Let $t_0 > \varrho$. Since $\dot\phi(t_*) = 0$ and $\dot\phi$ is strictly mon. decreasing on (t_*, ∞), $\dot\phi(t_0) < 0$. There are two sub-cases: If also $\phi(t_0) < 0$ then strict monotonicity implies $t_0 > \varrho$. Hence, Lem. A.2.5 applies to prove $t_i > \varrho, \forall i > 0$. Otherwise, if $\phi(t_0) > 0$, we have $\frac{\phi(t_0)}{\dot\phi(t_0)} < 0$ and hence, $t_1 = t_0 - \frac{\phi(t_0)}{\dot\phi(t_0)} > t_0$. Also, strict monotonicity of the derivative still implies $\dot\phi(t) < \dot\phi(t_0), \forall t > t_0$. Hence, $\phi(t_1) = \phi(t_0) + \int_{t_0}^{t_1} \dot\phi(t)\,\mathrm{d}t < \phi(t_0) + \int_{t_0}^{t_1} \dot\phi(t_0)\,\mathrm{d}t = \phi(t_0) + \dot\phi(t_0)[t_0 - \frac{\phi(t_0)}{\dot\phi(t_0)} - t_0] = 0$. So, $t_1 > \varrho$. Applying Lem. A.2.5 to the Newton sequence $(t_i)_{i\geq 1}$ shows that also $t_i > \varrho, \forall i \geq 1$.

□

We epitomise our findings in the following theorem:

Theorem A.2.7. *Let $A \in \mathbb{C}^{d\times d}$ with spectral radius $r := \rho(A) < 1$. Let $(t_i)_{i\in\mathbb{N}_0}$ be a Newton sequence with respect to function $\phi : t \mapsto (d-1)\log t + t\log r - \log\big((d-1)!\big) + (d-1)\log|||A||| - (d-1)\log(r)$. For any initial value $t_0 > -\frac{d-1}{\log\rho(A)}, \phi(t_0) \neq 0$, we have*
(i) $\forall i > 0 : \phi(t_i) < 0$ and
(ii) $\lim_{i\to\infty} \phi(t_i) = 0$ with $t_{i+1} < t_i, \forall i \geq 1$
(iii) $\forall i \geq 1 : \left|\left|\left|A^{\lceil t_i\rceil}\right|\right|\right| < 1$ where $\lceil t_i\rceil = \min\{k \in \mathbb{N} | k \geq t_i\}$ denotes the smallest natural number greater or equal than t_i.

input : Matrix $A \in \mathbb{C}^{d\times d}$, max. number of iterations $m \in \mathbb{N}$, initial offset $\epsilon > 0$.
output: Upper bound $k_0 \geq \min\{k \in \mathbb{N} : \left|\left|\left|A^k\right|\right|\right| < 1\}$ on the CME.

$N \leftarrow |||A|||$; // Compute the spectral norm
if $N < 1$ **then** $k_0 \leftarrow 1$
else
  $R \leftarrow \log \rho(A)$; $\delta \leftarrow d - 1$; $k_0 \leftarrow -\frac{\delta}{R} + \epsilon$
  for $i = 0 : m - 1$ **do**
    $\phi \leftarrow \delta \log k_0 + k_0 R - \log\big(\delta!\big) + \delta \log N - \delta R$; // $\phi(k_0)$
    $D_\phi \leftarrow \frac{\delta}{k_0} + R$; // $\dot{\phi}(k_0)$
    $k_0 \leftarrow k_0 - \phi/D_\phi$; // Newton step
  end
  $k_0 \leftarrow \lceil k_0 \rceil$; // Ensure the result is an integer
end

Algorithm 4: Newton-step based algorithm for finding an upper bound on the critical exponent of a matrix. Correctness (for any $m \in \mathbb{N}$) is guaranteed by Thm. A.2.7.

The theorem suggests an algorithm, based on Newton's method, for the determination of an upper bound k_0 on the critical exponent as given in Alg. 4.

Thm. A.2.7 implies that the algorithm generates a conservative sequence of upper bounds on the critical exponent. Therefore, in order to derive a closed-form bound k_0 on the latter, all we need to do is to write down the calculations performed of the algorithm for $m \geq 1$ iterations. The remaining questions are how to choose m and how to choose initial value t_0 so as to achieve a tight bound.

As we have established k_0 already is in the (ϱ, ∞) interval after one iteration, i.e. $t_1 > \varrho \geq t_*$. And, the curvature $\ddot{\phi}(t) = -\frac{\delta}{t^2}$ is monotonically increasing to zero from below. So, $\forall t > \varrho : |\ddot{\phi}(t)| < \left|\ddot{\phi}(\varrho)\right| \leq \left|\ddot{\phi}(t_*)\right| = \left|\log(r)^2\right|$. So, for spectral radius r close to 1, the curvature will be small on ϱ, t_1. Hence, a second Newton step, will bring t_2 close to the root. Of course, we have proven that further Newton steps will tighten our bound further, but there will be a trade-off with tightness of the bound and complexity of its formulaic expression. Therefore, we choose $m = 2$ yielding the bound $k_0 = t_2 = t_1 - \frac{\phi(t_1)}{\dot{\phi}(t_1)} = t_0 - \frac{\phi(t_0)}{\dot{\phi}(t_0)} - \frac{\phi(t_0 - \frac{\phi(t_0)}{\dot{\phi}(t_0)})}{\dot{\phi}(t_0 - \frac{\phi(t_0)}{\dot{\phi}(t_0)})} + 1$. The addition of $+1$ was made to ensure the bound still holds in a strict sense, if t_0 happened to be chosen to coincide with the root ϱ.

Theorem A.2.8. *For any $\epsilon > 0$, $t_0 = \epsilon + t_*$, $t_* = \frac{1-d}{\log r}$, both*

$$k_0(\epsilon) := \epsilon + t_* - \frac{\phi(\epsilon + t_*)}{\dot{\phi}(\epsilon + t_*)} + 1$$

and

$$\tilde{k}_0(\epsilon) := \epsilon + t_* - \frac{\phi(t_0)}{\dot{\phi}(\epsilon + t_*)} - \frac{\phi(\epsilon + t_* - \frac{\phi(\epsilon+t_*)}{\dot{\phi}(\epsilon+t_*)})}{\dot{\phi}(\epsilon + t_* - \frac{\phi(\epsilon+t_*)}{\dot{\phi}(\epsilon+t_*)})} + 1$$

are upper bounds on the critical exponent of matrix A. That is, $k_0 \geq \tilde{k}_0 \geq \min\{k \in \mathbb{N} : \left|\left|\left|A^k\right|\right|\right| < 1\}$.

The theorem essentially applies our algorithm for up to two steps and asserts that this suffices to find upper bounds on the critical exponent. The proof is an immediate consequence of Thm. A.2.7. The question of how to choose $\varepsilon > 0$ (optimally) is deferred to future work.

A.3. Application to proving finite-time stability in closed-loop discrete-time dynamical systems

A.3.1. Error-free dynamics

Critical exponents play an important role in a variety of applications. In this section, we illustrate how our results can be harnessed in answering finite-time stability guarantees of a closed-loop dynamic system.

A large number of important mechanical systems can be represented as a second-order control-affine system of the form:

$$\dot{x}_1 = x_2 \tag{A.5}$$
$$\dot{x}_2 = a(x) + b(x)u \tag{A.6}$$

where $x := [x_1; x_2] \in \mathbb{R}^m \times \mathbb{R}^m$ is the full state, b is typically bounded away from zero and u is a control input that can be freely chosen by the control designer. For simplicity, assume $m = 1$. A common control problem is that of stabilisation. That is, to design u as a function of state such that goal state $\xi = 0$. One way of achieving this is to set $u(x, t; w) := b^{-1}(x)(-a(x) - wx_1 - wx_2)$ where $w \in \mathbb{R}_+$ is a feedback gain parameter to be chosen at will. Substituting the control into the Euler-approximated matrix version of the dynamics equation, yields the following recurrent system representing the approximated closed-loop dynamics:

$$x_{k+1} = A_w\, x_k \qquad \text{where } A_w = \begin{pmatrix} 1 & \Delta \\ -\Delta w & 1 - \Delta w \end{pmatrix}. \tag{A.7}$$

Here, $\Delta \in \mathbb{R}_+$ is a time-increment representing the discretisation step size of the Euler-approximation and A_w is the transition matrix. By looking at the characteristic polynomial of transition matrix A_w it is easy to see that the system is stable ($\rho(A_w) < 1$) whenever the feedback gain is set to a positive number, while $|||A_w||| > 1$. Although this guarantees convergence of the state to zero eventually, it does not answer questions of finite-time errors, which may exhibit non-monotonous behaviour (for an example, see Fig. A.2).

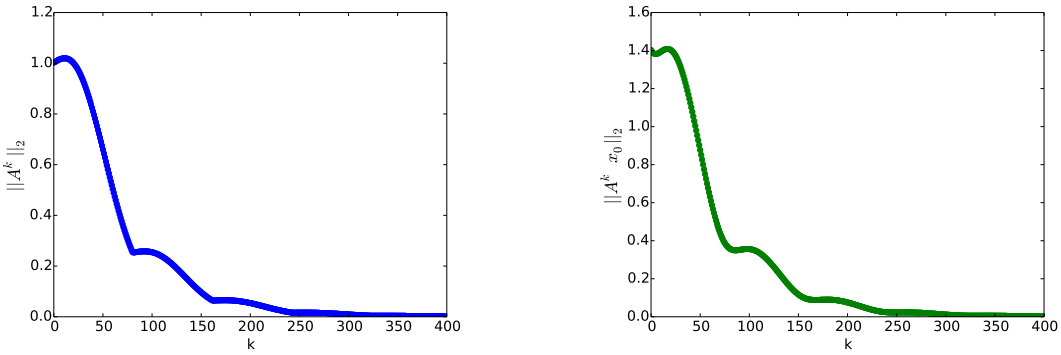


Figure A.2.: Power evolutions of dynamic system as per Eq. A.7, with $\Delta = 0.05$ and $w = 0.7$. Left: Evolution of $k \mapsto |||A^k|||$. Right: Evolution of $k \mapsto |||A^k x_0|||$ for $x_0 = (1, 1)^\top$. Note, the evolutions exhibit decreasing but non-monotonous behaviour.

While convergence in the limit of infinite time is of theoretical interest, the transient behaviour can be of great practical importance when studying a dynamic system.

In particular we may desire to answer the following questions:

1. Given $w > 0$, after what time step k_ϵ can we guarantee that the error $\|x_k\|$ is at most $\epsilon > 0$, i.e. $\|x_k\| \leq \epsilon, \forall k \geq k_\epsilon$?

2. Given maximal error ϵ and a time step k_f, how do we need to set control parameter w to ensure that the error remains below ϵ after time k_f ? That is, how to set w to ensure $k_\epsilon \leq k_f$?

1) Obviously, $\|x_k\|_2 = \left\|A_w^k x_0\right\|_2 \leq \left|\!\left|\!\left|A_w^k\right|\!\right|\!\right| \|x_0\|_2$. If $|\!|\!|A_w|\!|\!| < 1$ one can leverage that the norm is sub-multiplicative to see that $\|x_k\|_2 \leq |\!|\!|A_w|\!|\!|^k \|x_0\|_2 \overset{k\to\infty}{\longrightarrow} 0$ where convergence is strictly monotonous. Unfortunately, in situations such as ours, where $|\!|\!|A|\!|\!| \geq 1$, matters are more involved. Since the matrix is stable, Gelfand's lemma still guarantees monotonous convergence to zero in the long run. However, the dynamics may exhibit transient behaviour with state initially departing from equilibrium point 0 before eventually entering a phase of strict monotonous decreasing convergence. The time when the dynamics enter this phase of strict monotonous convergence often is of great practical interest and can be conservatively estimated by any bound on the critical exponent k_0 as follows:

By Thm. A.2.3, $\max_{r\in\{1,\ldots,k_0-1\}} |\!|\!|A_w^r|\!|\!| \leq \exp(\phi\Big(\frac{1-d}{\log\rho(A_w)}\Big)) := c_w < \infty$.

We have, $\forall k \exists i \in \mathbb{N}, r \in \{0,\ldots,k_0-1\} : k = ik_0 + r \geq k_0$ we have $\|x_k\|_2 = \left\|A_w^k x_0\right\|_2 = \left\|A_w^{k_0 i+r} x_0\right\|_2$ $\leq \left|\!\left|\!\left|A_w^{k_0}\right|\!\right|\!\right|^i |\!|\!|A_w^r|\!|\!| \|x_0\|_2 \leq \left|\!\left|\!\left|A_w^{k_0}\right|\!\right|\!\right|^i \|x_0\|_2 \max_{r\in\{0,\ldots,k_0-1\}} |\!|\!|A_w^r|\!|\!| = c_w \left|\!\left|\!\left|A_w^{k_0}\right|\!\right|\!\right|^i \|x_0\|_2$. Thus, a sufficient condition for $\|x_k\|_2 \leq \epsilon$ is $\left|\!\left|\!\left|A_w^{k_0}\right|\!\right|\!\right|^i \leq \frac{\epsilon}{c_w\|x_0\|_2}$. Taking the log on both sides and observing that $\log \left|\!\left|\!\left|A_w^{k_0}\right|\!\right|\!\right| < 0$ (since k_0 is an upper bound on the critical exponent), we obtain the equivalent sufficient condition:

$$k \text{ div } k_0 \geq \frac{\log \frac{\epsilon}{c_w\|x_0\|_2}}{\log \left|\!\left|\!\left|A_w^{k_0}\right|\!\right|\!\right|} =: R(w)$$

where we have also utilised $i = k \text{ div } k_0$. The last inequality is satisfied by choosing

$$k := \lceil R(w) \rceil \, k_0 \tag{A.8}$$

which can be seen as follows: For $r \in \mathbb{R}$, let $\lfloor r \rfloor := \max_{z\in\mathbb{Z}} z \leq r$ be the largest integer that is a lower bound on real number r. With $k := \lceil R(w) \rceil \, k_0$ we have $k \text{ div } k_0 = \left\lfloor \frac{k}{k_0} \right\rfloor = \left\lfloor \frac{\lceil R(w) \rceil k_0}{k_0} \right\rfloor = \lceil R(w) \rceil \geq R(w)$.

2) Given maximal error $\epsilon > 0$ and a time step k_f, how to set feedback constant w to ensure $\|x_k\|_2 \leq \epsilon, \forall k \geq k_f$?

Reminding ourselves of the preceding derivations in 1), Eq. A.8 tells as that it suffices to choose w such that $\frac{k}{k_0} = \lceil R(w) \rceil$. So, it is sufficient to choose $w > 0$ such that $0 \geq R(w) - \frac{k}{k_0}$.

Resubstitution of the constituent parts yields the equivalent expression: $\log c_w \geq \frac{\log \frac{\epsilon}{\|x_0\|_2}}{\log \left|\!\left|\!\left|A_w^{k_0}\right|\!\right|\!\right|} - \frac{k}{k_0} \Leftrightarrow$ $\phi\Big(\frac{1-d}{\log\rho(A_w)}\Big) \geq \frac{\log \frac{\epsilon}{\|x_0\|_2}}{\log \left|\!\left|\!\left|A_w^{k_0}\right|\!\right|\!\right|} - \frac{k}{k_0} \Leftrightarrow \delta \log\Big(\frac{1-d}{\log\rho(A_w)}\Big) - \log(\delta!) + \delta \log |\!|\!|A|\!|\!| + (\Big(\frac{1-d}{\log\rho(A_w)}\Big) - \delta) \log(r) \geq$ $\frac{\log \frac{\epsilon}{\|x_0\|_2}}{\log \left|\!\left|\!\left|A_w^{k_0}\right|\!\right|\!\right|} - \frac{k}{k_0}$. Ideally, this inequality would be converted into a closed-form bound on w. However, in the absence of a closed-form solution one can attempt numerical search for a w that satisfies the inequality.

A.3.2. Bounds with additive interval-bounded disturbances

So far, we have considered the special case of a two-dimensional dynamic system. Next, we will generalise this to more general discrete-time dynamics in higher dimensions and with interval-bounded disturbances.

We assume a $2m$-dimensional dynamic system of the form :

$$x_{k+1} = Mx_k + \Delta F_k \tag{A.9}$$

where $\Delta \in \mathbb{R}_+$ is a positive time increment, k is the time step index, each F_k is an (uncertain) *disturbance* such that $\|F_k\| \leq \bar{\mathfrak{N}}$ for some *disturbance norm bound* $\bar{\mathfrak{N}} \in \mathbb{R}_+$.

Matrix M is the transition matrix. As before, we will be chiefly interested in the case where $|\!|\!|M|\!|\!| \geq 1$ but $\rho(M) < 1$. This case is of interest in many control applications where $M = \begin{pmatrix} I_m & \Delta I_m \\ -\Delta K_1 & I_m - \Delta K_2 \end{pmatrix}$ and $K_1, K_2 > 0$ can be chosen to render M stable (we have discussed such an instance in the previous subsection).

By induction, it is easy to show that

$$x_k = M^k\, x_0 + \Delta \sum_{i=0}^{k-1} M^{k-1-i}\, F_i. \tag{A.10}$$

Hence,

$$\|x_k\| \leq \left|\!\left|\!\left| M^k \right|\!\right|\!\right| \, \|x_0\| + \Delta \sum_{i=0}^{k-1} \left|\!\left|\!\left| M^{k-1-i} \right|\!\right|\!\right| \, \|F_i\| \tag{A.11}$$

$$\leq \left|\!\left|\!\left| M^k \right|\!\right|\!\right| \, \|x_0\| + \Delta \bar{\mathfrak{N}}_k \sum_{i=0}^{k-1} \left|\!\left|\!\left| M^i \right|\!\right|\!\right| \tag{A.12}$$

where $\bar{\mathfrak{N}}_k := \max_{i\in\{0,\ldots,k-1\}} \|F_i\| \leq \bar{\mathfrak{N}} \in \mathbb{R}$.

Therefore, to bound $\|x_k\|$, we need to understand the behaviour of the matrix norm terms $\left|\!\left|\!\left| M^k \right|\!\right|\!\right|$ and $\sum_{i=0}^{k-1} \left|\!\left|\!\left| M^i \right|\!\right|\!\right|$.

For the former, we already have presented bounding and convergence arguments for the two-dimensional case. One approach to establish convergence of both terms is via Gelfand's formula. Remember, Gelfand's formula asserts $\rho(M) = \lim_{k\to\infty} \left|\!\left|\!\left| M^k \right|\!\right|\!\right|^{\frac{1}{k}}$. Hence, by the standard root test criterion for series convergence, $\sum_{k=0}^{\infty} \left|\!\left|\!\left| M^k \right|\!\right|\!\right| < \infty$ if $\rho(M) < 1$. In that case, necessarily $\left|\!\left|\!\left| M^k \right|\!\right|\!\right| \to 0$ as $k \to \infty$. The latter can also be seen as follows:

Let $\rho(M) < 1$. Then, we know that M^k will converge to the zero-matrix as $k \to \infty$:

$\forall \epsilon > 0, \exists k_0 \in \mathbb{N} \forall k \geq k_0 : \left| \left|\!\left|\!\left| M^k \right|\!\right|\!\right|^{1/k} - \rho(M) \right| < \epsilon$. In particular, we can choose $\epsilon_0 > 0$ and $k_0 \in \mathbb{N}$ such that $\left|\!\left|\!\left| M^k \right|\!\right|\!\right|^{1/k} < \varphi := \rho(M) + \epsilon_0 < 1, \forall k \geq k_0$. Thus, $\left|\!\left|\!\left| M^k \right|\!\right|\!\right| < \varphi^k < 1, \forall k \geq k_0$. Consequently, $\left|\!\left|\!\left| M^k \right|\!\right|\!\right| \overset{k\to\infty}{\longrightarrow} 0$.

Apart from these asymptotic convergence guarantee, we are interested in finite-time bounds. These will be derived next.

Lemma A.3.1. *Let $M \in \mathbb{C}^{d\times d}$ with spectral radius $r := \rho(M) < 1$. Let $k_0 \in \mathbb{N}, k_0 > 1$ such that $\forall k \geq k_0 : \left|\!\left|\!\left| M^k \right|\!\right|\!\right| < 1$. Define $\delta_k := k$ div $k_0 = \left\lfloor \frac{k}{k_0} \right\rfloor$ and $\varphi := \left|\!\left|\!\left| M^{k_0} \right|\!\right|\!\right| < 1$. We have:*

$$\left|\!\left|\!\left| M^k \right|\!\right|\!\right| \leq c\, \varphi^{\delta_k}, \;\; \forall k \in \mathbb{N}, k \geq k_0$$

for either choice:

1. $c := \max\{\left|\!\left|\!\left| M^i \right|\!\right|\!\right| | i = 1, \ldots, k_0 - 1\}$
2. *or, more conservatively,* $c := \frac{1}{(d-1)!}\left(\frac{1-d}{\log r}\right)^{d-1} |||M|||^{d-1} \;\; r^{\frac{1-d}{\log r} - d + 1}$.
3. *Provided* $|||M||| \geq 1$*, a simpler choice is* $c := |||M|||^{k_0}$.

Proof. Let $k \geq k_0$, $\delta_k := k$ div k_0, $\varrho_k := k$ mod $k_0 \leq k_0 - 1$. That is, $k = \delta_k k_0 + \varrho_k$. Define $\mu(k_0) := \arg\max\{\left|\!\left|\!\left| M^i \right|\!\right|\!\right| \,|\, i = 1, \ldots, k_0 - 1\}$.

1. $\left|\!\left|\!\left| M^k \right|\!\right|\!\right| = \left|\!\left|\!\left| M^{\delta_k k_0 + \varrho_k} \right|\!\right|\!\right| = \left|\!\left|\!\left| M^{\delta_k k_0} M^{\varrho_k} \right|\!\right|\!\right| \leq \left|\!\left|\!\left| M^{k_0} \right|\!\right|\!\right|^{\delta_k} |||M^{\varrho_k}||| \leq \left|\!\left|\!\left| M^{k_0} \right|\!\right|\!\right|^{\delta_k} \left|\!\left|\!\left| M^{\mu(k_0)} \right|\!\right|\!\right|$. This covers the choice $c := \left|\!\left|\!\left| M^{\mu(k_0)} \right|\!\right|\!\right|$.

2. The second choice of c is covered by Thm. A.2.3

which asserts $\left|\!\left|\!\left| M^{\mu(k_0)} \right|\!\right|\!\right| \leq \frac{1}{(d-1)!}\left(\frac{1-d}{\log r}\right)^{d-1} |||M|||^{d-1} \;\; r^{\frac{1-d}{\log r} - d + 1}$.

3. Since $\mu(k_0) < k_0$ and $|||M||| \geq 1$ we have $|||M|||^{\mu(k_0)} \leq |||M|||^{k_0}$. Hence, $\left|\!\left|\!\left| M^k \right|\!\right|\!\right| \overset{\text{see 1.}}{\leq} \left|\!\left|\!\left| M^{k_0} \right|\!\right|\!\right|^{\delta_k} \left|\!\left|\!\left| M^{\mu(k_0)} \right|\!\right|\!\right|$ $\leq \left|\!\left|\!\left| M^{k_0} \right|\!\right|\!\right|^{\delta_k} |||M|||^{\mu(k_0)} \leq \left|\!\left|\!\left| M^{k_0} \right|\!\right|\!\right|^{\delta_k} |||M|||^{k_0}$. □

Lemma A.3.2. *Let $k \in \mathbb{N}, k > k_0$, $q \in \mathbb{N}$ and $S_q := \sum_{j=0}^{k_0-1} \left|\!\left|\!\left| M^j \right|\!\right|\!\right|^q$.*

With definitions as in Lem. A.3.1, we have:

$$\sum_{j=0}^{k} \left|\!\left|\!\left| M^j \right|\!\right|\!\right|^q \leq S_q + c^q k_0 \frac{\varphi^q - \varphi^{q\left\lfloor \frac{k}{k_0} \right\rfloor + q}}{1-\varphi^q} \leq S_q + c^q k_0 \frac{\varphi^q}{1-\varphi^q}.$$

Proof. Let $\theta_q := \varphi^q$. As before, $\delta_k = \lfloor k/k_0 \rfloor = k \text{ div } k_0$, $\varrho_k := k \mod k_0$.

Then, $\sum_{j=k_0}^{k} \theta_q^{\delta_j} = (\varrho_k+1)\theta_q^{\delta_k} + \sum_{j=1}^{\delta_k - 1} k_0 \theta_q^j \leq \sum_{j=1}^{\delta_k} k_0 \theta_q^j$. Hence, $\sum_{j=0}^{k} \left|\!\left|\!\left| M^j \right|\!\right|\!\right|^q = S_q + \sum_{j=k_0}^{k} \left|\!\left|\!\left| M^j \right|\!\right|\!\right|^q \overset{Lem.A.3.1}{\leq} S_q + \sum_{j=k_0}^{k} c^q \varphi^{q\delta_j} = S_q + c^q \sum_{j=k_0}^{k} \theta_q^{\delta_j} \leq S_q + c^q k_0 \sum_{j=1}^{\delta_k} \theta_q^j = S_q + c^q k_0 \theta_q \sum_{j=0}^{\delta_k - 1} \theta_q^j = S_q + c^q k_0 \theta_q \frac{1-\theta_q^{\delta_k}}{1-\theta_q} = S_q + c^q k_0 \frac{\theta_q - \theta_q^{\delta_k+1}}{1-\theta_q} = S_q + c^q k_0 \frac{\varphi^q - \varphi^{q\left(\left\lfloor \frac{k}{k_0} \right\rfloor + 1\right)}}{1-\varphi^q} \leq S_q + c^q k_0 \frac{\varphi^q}{1-\varphi^q}$ where the last inequality follows from $0 \leq \varphi$.

□

We are in a position to wrap up the following conclusion:

Theorem A.3.3. *Let $(F_k)_{k\in\mathbb{N}_0}$ be a norm-bounded sequence of vectors with $\bar{\mathfrak{N}}_k := \max_{i\in\{0,\ldots,k-1\}} \|F_i\| \leq \bar{\mathfrak{N}} \in \mathbb{R}$. For sequence $(x_k)_{k\in\mathbb{N}_0}$ defined by the linear recurrence $x_{k+1} = Mx_k + \Delta F_k \;\; (k \in \mathbb{N}_0)$, we can define the following bounds:*

1. $\|x_k\| \leq \left|\!\left|\!\left| M^k \right|\!\right|\!\right| \, \|x_0\| + \Delta \bar{\mathfrak{N}}_k \sum_{i=0}^{k-1} \left|\!\left|\!\left| M^i \right|\!\right|\!\right|$. *The right-hand side converges as $k \to \infty$.*
2. *Let $k_0 \in \mathbb{N}, k_0 > 1$ such that $\left|\!\left|\!\left| M^k \right|\!\right|\!\right| < 1, \forall k \geq k_0$ and let $\varphi := \left|\!\left|\!\left| M^{k_0} \right|\!\right|\!\right| < 1$ and let $\delta_k := \lfloor k/k_0 \rfloor$. If $r := \rho(M) < 1$, for $k > k_0$, we also have:*

$$\|x_k\| \leq c\,\varphi^{\delta_k} \, \|x_0\| + \Delta \bar{\mathfrak{N}}_k \Big(\sum_{j=0}^{k_0-1} \left|\!\left|\!\left| M^j \right|\!\right|\!\right| + c\,k_0 \frac{\varphi - \varphi^{\left\lfloor \frac{k}{k_0} \right\rfloor + 1}}{1-\varphi} \Big) \overset{k\to\infty}{\longrightarrow} C \leq \Delta \bar{\mathfrak{N}} \sum_{j=0}^{k_0-1} \left|\!\left|\!\left| M^j \right|\!\right|\!\right| + \frac{\Delta \bar{\mathfrak{N}}\, c\, k_0\, \varphi}{1-\varphi} \tag{A.13}$$

   for some constant C. Here convergence is from below.

   Possible choices for $c \in \mathbb{R}$ are:

   (i) $c = \max\{ \left|\!\left|\!\left| M^i \right|\!\right|\!\right| | i = 1, \ldots, k_0 - 1 \}$

   or (ii) $c = \frac{1}{(d-1)!} \left(\frac{1-d}{\log r} \right)^{d-1} |\!|\!| M |\!|\!|^{d-1} \;\; r^{\frac{1-d}{\log r} - d + 1}$. *Since $|\!|\!| M |\!|\!| \geq 1$, one can also choose*

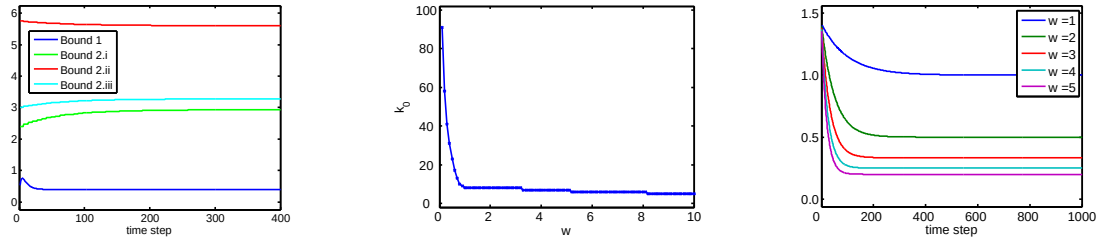
   (iii) $c := |\!|\!| M |\!|\!|^{k_0}$.

3. *If $|\!|\!| M |\!|\!| \neq 1$, the following also holds:* $\|x_k\| \leq |\!|\!| M |\!|\!|^k \, \|x_0\| + \Delta \bar{\mathfrak{N}}_k \frac{1 - |\!|\!| M |\!|\!|^k}{1 - |\!|\!| M |\!|\!|}$.

Proof. 1) is due to Eq. A.12. 3) Follows from the same equation in conjunction with the geometric sum formula and the fact that $\left|\!\left|\!\left| M^i \right|\!\right|\!\right| \leq |\!|\!| M |\!|\!|^i$. 2) is a consequence of Lem. A.3.1 and Lem. A.3.2. □

Note, these bounds are conservative. As an illustration of the relative degree of conservatism, refer to the left plot of Fig. A.3. As can be seen from the left-most plot, bound 1 has the lowest degree of conservatism while bound 2, with choice c as in 2.(ii) is the loosest. The bounds were computed using $M = A_w$ with A_w as defined in Eq. A.7, with $\Delta = .01$, $w = K_1 = K_2 = 5$ and with $\bar{\mathfrak{N}}_k = 1 \, (k \in \mathbb{N})$. The centre plot of Fig. A.3 shows the dependence of k_0 on w.

To get a feel for the third bound, we have also plotted the time evolution of bound 3 for $M = \text{diag}(1 - \Delta w, 1 - \Delta w)$ and varying w. As expected, increasing feedback gain w decreases the bound. This is most easy to explain for the simple case $d = 1$, $M = 1 - \Delta w < 1$. Then the bound becomes $M^k \, |x_0| + \Delta \bar{\mathfrak{N}}_k \frac{1-M^k}{1-M}$ $= M^k \, |x_0| + \frac{\bar{\mathfrak{N}}_k}{w} \overset{k\to\infty}{\longrightarrow} \frac{\bar{\mathfrak{N}}}{w} \overset{w\to\infty}{\longrightarrow} 0$.



(a) Error bound comparison (log-scale).

(b) Dependence of k_0 on gain w.

(c) Evolution of Bound 3 for various w ($|||M||| < 1$).

Figure A.3.: Left: Bounds 1 and Bound 2.i -iii of Thm. A.3.3 as a function of time step k. Since $|||M||| \geq 1, \forall w$ in the depicted setup, Bound 3 was not applicable. Centre: Decrease of k_0 with increasing gain w. Right: Evolution of Bound 3 for various w. Note how the bound decreases with increasing w. For all w we had $|||M||| < 1$. Hence, Bound 2 was not applicable in this setting.

B. Termination guarantee of the modified lazy auction mechanism

In this chapter, we provide a termination guarantee of our auction based coordination mechanism in the context of multi-agent trajectory planning in a shared graph environment. The expository backdrop is multi-robot collision avoidance in dynamic systems where the nodes are mutually exclusive projections of states and the edges correspond to (deterministic) state transition laws. The (control) actions are which edges to take and a legal plan (that is a feasble one) is one that avoids collisions. While we consider the case where a collision is defined as a situation where two robots occupy the same state, the situation can be further restricted to disallowing combinations of states.

B.1. Coordination in discrete graphs

If a coordination mechanism is to be applied in a time-critical scenario it can be important to consider the termination properties. That is, is the coordination method guaranteed to reach a coordination solution in a finite number of iterations? For a slight modification of our mechanism we can give an affirmative answer in environments with a finite number of states (i.e. resources or nodes) but a (countably) infinite planning horizon. Such planning problems can be formalized as path planning problems in graphs.

After formalizing the problem statement we provide a formal discussion of the circumstances under which we are indeed able to prove that our modified mechanism is guaranteed to achieve coordination in a finite number of iterations. The section concludes with a proof of this termination guarantee.

B.1.1. Problem statement and conventions

We assume the following model: Let $R = \{R_1, ..., R_M\}$ be the set of resources (states /locations) in the system that robots $1, ..., A$ can claim in a discrete sequence of time steps $t = 1, 2, ..., \infty$. Remember, safety dictates that resource claims are mutually exclusive (i.e. no two robots can simultaneously use the same resource). Therefore, we assume that the number of resources is at least as high as the number of robots, $0 < A \leq M$.

Assuming not all states are reachable from all other states in one time step, we can express the reachability (within one time step) as a sequence of relations $E_t = (R \times R)_t \, (t \in \mathbb{N}_0)$ where $(R_u, R_v) \in E_t$ iff a robot can reach R_v at time step $t + 1$ when it is in R_u at time step t.[1] `Insert`

[1] For simplicity, we tacitly assume homogeneous robot capabilities that are known in advance.

Let Z_t^a denote the state robot a plans to occupy at time t. In this environment, a *legal* open-loop plan corresponds to a sequence of states $(Z_t^a)_{t \in \mathbb{N}_0}$ such that $\forall t \in \mathbb{N}_0 : (Z_t^a, Z_{t+1}^a) \in E_t$. Furthermore, the *overall plan* now corresponds to the joint sequence $(Z_t)_{t \in \mathbb{N}_0}$ where $Z_t = Z_t^1 \times \ldots \times Z_t^A$.

We can express this model as a graph planning problem. Spatio-temporal graph $\mathcal{G} = (V, E)$ consists of vertices $V = \{v_{t,m} | t \in \mathbb{N}_0, m \in \{1, ..., M\}\}$ where vertex $v_{t,m}$ corresponds to state R_m at time t. So the graph consists of an infinite number of time layers ("rows") indexed by t and each of the M "columns" corresponds to one state each. The set of edges reflects the reachability conditions defined above. That is, we assume $(v_{t,l}, v_{t+1,m}) \in E$ iff $(R_l, R_m) \in E_t$. For an illustration refer to Fig. B.1.

With these notions in mind, robot a's plan in coordination iteration i can be represented as a sequence of vertices in spatio-temporal graph $\mathcal{G}$. We denote this sequence as $(z_t^a(i))$ where $z_t^a(i) \in \{v_{t,1}, ..., v_{t,M}\}$ is the vertex robot a intends to visit in time layer t. Notice the relationship $z_t^a(i) = v_{t,m} \Leftrightarrow Z_t^a(i) = R_m$.

When executing a plan that involves a state transition corresponding to using an edge in E the robot receives a cost specified by edge cost function $\gamma^a : E \to \mathbb{R}_+$. (Note, above-mentioned cost function c^a on the set of

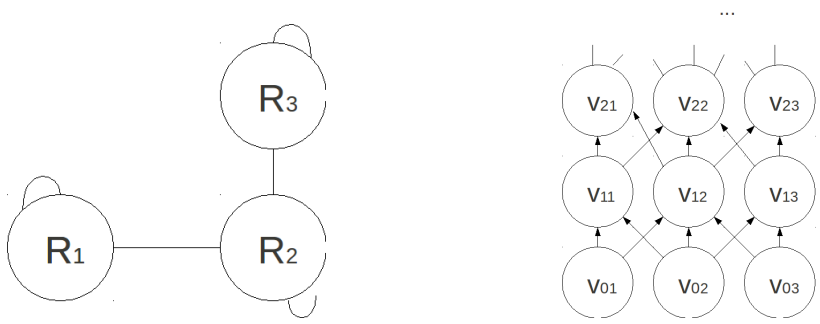


Figure B.1.: Left: Spatial graph with $M = 3$ resources. Right: First three time layers (rows) of corresponding spatio-temporal graph.

feasible plans can now be defined as the sum of all costs of edges (acc. to γ^a) the plan would have to use to generate the implied sequence of nodes.)

At time $t = 0$, robots $1, ..., A$ are in their start states $S(1) \in R, ..., S(A) \in R$, respectively ($S(a) \neq S(r), \forall a \neq r$). That is they are all at distinct nodes in (time-) *layer* 0. The planning problem can now be expressed as finding a collection of low-social cost open-loop plans corresponding to legal sequences of states (nodes) $(z_t^a)_{t\in\mathbb{N}_0}$ ($a \in \mathcal{A}$) such that the local constraints are met, they are collision-free (i.e. $z_t^a \neq z_t^r, \forall t \in \mathbb{N}_0 \forall a \neq r$) and the robots reach their destinations from their start states. Note, for convenience we will not always explicitly distinguish between a column $(v_{0,m}, v_{1,m}, ...)$ in the graph and the corresponding resource R_m.

B.1.2. Further Assumptions

To show that our mechanism is guaranteed to achieve coordination, we will consider the system dynamics. Notice, there are different facets we can consider: (I) the *coordination dynamics* which we define to govern the evolution of plans or global state assignments $(z_t)_{t\in\mathbb{N}_0}(i)$ as a function of iteration index i, and (II) the *execution dynamics* which we define to dictate the evolution of states $z_t(i)$ as a function of time index t for a given, fixed iteration i. Below, we will consider an intermediate level of dynamics. We will elucidate the execution dynamics until a finite time horizon for iterations that have established the absence of conflicts until that horizon.

Our mechanism can be demonstrated to empirically perform well in real-world settings. However, in order to be able to deliver a formal proof that establishes a guaranteed coordination success in a finite number of coordination iterations we introduce a number of additional assumptions:

- (A1) There exists a path from every vertex (location) to every other vertex in later time layers of the spatio-temporal graph.
- (A2) Every vertex has at least $|\mathcal{A}|$ children, where $\mathcal{A}$ is the set of robots competing for resources.
- (A3) State transition costs are uniformly positive. That is, all agents incur constant costs for using an edge: $\gamma^a(e) = k \in \mathbb{R}_+, \forall a \in \mathcal{A}, e \in E$
- (A4) Once a robot reaches its goal, it disappears from the system/planning problem (i.e. discontinues to claim any resources in subsequent time steps).
- (A5) The robots' individual planning algorithms are suitable and produce only legal plans. Furthermore, given the number of available resources each robot's planner is always capable of finding a cost optimal path between start state and destination state given the robots knowledge of available states.
- (A6) Robots can opt to remain where they are. That is, the transition relation is reflexive: $\forall t \in \mathbb{N}_0, m \in \{1, ..., M\} : (v_{t,m}, v_{t,m}) \in E$.
- (A7) The environment is static, i.e. $\forall t \in \mathbb{N}_0 : E_t = E_{t+1}$.

- (A8) The reachability relation is delayed symmetric, i.e. $\forall j, m \in \{1, ..., M\} \forall t : (R_m, R_j) \in E_t \Rightarrow (R_j, R_m) \in E_{t+1}$.

Assumption A1 is necessary to ensure all mutually exclusive configurations of start and destination locations correspond to a feasible plan resulting in a path that connects them. Positivity of costs required in A3 is reasonable in virtually all physical systems. If we lifted it (see below) this can result in simplifications guaranteeing optimal solutions of our mechanism for certain cost maps and in robots never having an incentive to reaching their destinations (also see the discussion below). The requirement that edge cost is a constant implies that a cost-optimal path is also a path with a minimal number of hops (vertex visitations). Assumption A2 will prove important to ensure that any robot that loses a resource actually can be guaranteed to be able to execute an alternative plan not involving this resource under all circumstances. Finally, A4 is an assumption mostly found in multi-robot coordination papers that provide theoretical guarantees. It ensures that no robot's destination is occluded by other (idle) robots that already reached their goals.

While assumption A4 obviously does restrict the applicability to some settings. Still, there many scenarios where A4 is met or which can be modified accordingly (eg by an external robot pick-up mechanism). For formal reasons, we model A4 by defining that if Robot r reaches its destination in iteration T_r along a conflict-free path then it plans on using a virtual dummy resource R_{m+r} (which no other robot ever claims) from then on: $Z^r_{T_r}(i) = D(r) \Rightarrow Z^r_{T_r+t} = R_{m+r}, \forall t \in \mathbb{N} - \{0\}$. Notice, it would be equally sufficient to lift A4 and replace it by another assumption stating that the destinations of the robots are never blocking the pathways of other robots to their destinations. This could be assured if the robots' destinations are dead ends (in which case we would adapt A2 to exclude destinations). For instance, in a multi-vehicle scenario the destinations could be distinct car ports.

A5 simply states that the robots are capable of making individually optimal decisions. In the graph planning scenario they could simply employ prominent shortest-path algorithms such as A^* or Dijkstra's algorithm [83]. It is obvious that this requirement is important since our mechanism relies on the cost estimations of the local planning algorithms to compute the bids and since all robots need to have working planning algorithms to compute the paths that lead them to their destinations.

B.2. Modified mechanism and termination guarantee

Definition B.2.1 ((Individual) Loss Horizon (ilh))**.** *For coordination iteration i, Robot r's current* individual loss horizon $ilh^r(i)$ *is defined as the largest time step where Robot r has lost a resource to a competing agent in iteration i or in previous coordination iterations. We say that the vertices $v_{t,m}$ where $t \in ilh^r(i)$ are* on the individual loss horizon *and define $ILH^r(i) := \{v_{t,m} \in V | t \in ilh^r(i)\}$ to be the set of these spatio-temporal vertices.*

Note, that in iteration i, every robot r believes that all spatio-temporal vertices $v_{t,m}$ with $t > ilh^r(i)$ are available (of course, it may turn out that r loses $v_{t,m}$ in a subsequent iteration $i' > i$ in which case $ilh^r(i) < ilh^r(i')$.

B.2.1. Modified bidding behaviour

As mentioned above we will modify our mechanism in order to establish provable convergence. In the modified mechanism each robot r maintains a list data structure L^r containing spatio-temporal vertices which he has lost in past iterations. We will call this list *loss horizon list* as it keeps track of r's individual loss horizon. Furthermore, it serves as a memory of the iteration order in which spatio-temporal vertices have been initially lost. The bidding mechanism will be as AUC described in Ch. 6 for the most recently lost vertex and vertices beyond a robot's loss horizon but will ensure that bids for all other vertices are restricted to $\{0, k\}$. While this property is a restriction in bidding flexibility (and possibly may result in higher social cost for certain problem instances) we leverage this restriction in our proof of the convergence theorem given below.

At the beginning of the coordination process each robot starts out with an empty list L^r. Whenever r loses a spatio-temporal vertex, say $v_{t,m}$, *which is not already in the list*, he inserts it to the first position of the list such that there is no vertex with larger time index $t' > t$ behind $v_{t,m}$ in the list. Hence, if L^r_u denotes the uth

element of the list then we have $\forall u < u' \in \mathbb{N} : ((L^r_u = v_{t,m} \wedge L^r_{u'} = v_{t',j} \Rightarrow t \geq t')$ and (if $t = t'$ then $v_{t,m}$ was lost in a later coordination iteration than $v_{t',j}$)). By construction, the first entry $top\,(L^r)$ of list L^r is the vertex of the highest time layer that r lost (for the first time) in the most recent iteration.

For reasons that are important for the proof of Thm. B.2.2 below, the loss horizon list is constructed to never shrink– although it is permissable to remove all nodes of lower time steps than $top(L^r(i))$ to save storage space.

The key difference between the generic auction method AUC described in Ch. 6 comes to bear when we consider bidding. Assume Robot r bids for vertex $v_{t,j}$ in iteration i and let $v_{t',m} = top\,(L^r)$. If $t > t'$ or $v_{t,j} = v_{t',m}$ then r submits the usual bid

$$b^r(i) = l^r(i) - s^r(i)$$

as in AUC described in Ch. 6. Otherwise however, r submits the bid

$$b^r(i) = k - k\,\delta(l^r(i) - s^r(i)) \tag{B.1}$$

where k is the constant transition cost acc. to A3 and $\delta : \mathbb{R} \to \{0, 1\}$ denotes the Kronecker delta function. Therefore, bids below the individual loss horizon are always in $\{0, k\}$ and can only qualitatively express whether losing a node is a detour ($b^r(i) = k$) or not ($b^r(i) = 0$). By contrast, bids for vertices beyond $ilh^r(i)$ express a quantification of the magnitude of the detour cost losing the auction is anticipated to entail. We will use this bid construction to show that the bids consequently always are in $\{0, k\}$ which in turn is a stepping stone in proving convergence Thm. B.2.2 stated below.

B.2.2. Termination guarantee

Theorem B.2.2 (Termination Guarantee)**.** *Provided the definitions and assumptions above hold, our modified mechanism generates a joint plan in a finite number of iterations that is legal, collision-free and allows all robots to reach their destinations, regardless of the (mutual exclusive) start states. That is:*

$\forall$ *injective* $S, D : \mathcal{A} \to R\, \exists T^1, \ldots, T^A, i \in \mathbb{N}_0 \forall a, r \in \mathcal{A}, a \neq r, t : z^a_t(i) \neq z^r_t(i) \wedge Z^a_{T^a}(i) = D(a) \wedge Z^a_0(i) = S(a)$.

Before attempting a formal proof we will give an informal outline as follows: The motivation behind augmenting the bidding rule by the one in Eq. B.1 was to ensure that we can show that all bids for a vertex whose loss would result in an increase in the path length of the bidders all equal k (see Lemma B.3.1 given below). Therefore, the robot A with the highest priority only loses bids for resources when there is an alternative route to his destination of equally short path length (avoiding the lost spatio-temporal vertex). Hence, this robot will always reach his destination in a finite number T^A of time steps, where T^A is the number of state transitions along a shortest path (cf. Lemma B.3.2 given below). Due to A4, Robot A is effectively removed from the path-planning process beyond time step T^A and we can show convergence of the remaining $A - 1$ robots' coordination problem with a recursive argument.

B.3. Derivation of Theorem B.2.2

Before commencing with our derivations we will need to establish a bit more notation and terminology that builds on the notions already defined above. For convenience a summary of the introduced notation is depicted in Tab.

We define the *min-hop-distance* $H(v_{t,s}, R_d)$ between a spatio-temporal vertex $v_{t,s}$ and a resource R_d as the minimum length of a path leading from $v_{t,m}$ into the column representing R_d. The min-hop-distance is a property of the graph. It could also be defined as the geodesic distance or shortest path length between R_s and R_d in the spatial graph.

By contrast, for robot r, we define his *conceived hop-distance* $h^r_i(v_{t,s}, R_d)$ between a spatio-temporal vertex $v_{t,s}$ and a resource R_d as the minimum length of a path leading from $v_{t,m}$ into the column representing R_d, given r's current belief about available vertices for time steps $\tau \geq t$, given the auction history of won and

Notation	Meaning
$\mathcal{A}$	set of robots
A	number of robots competing for resources ($A \leq \|\mathcal{A}\|$)
a, r	indices of robots r and a
t	ordinal time index
i	coordination iteration index
$R_1, ..., R_M$	resources / locations / nodes in spatial graph
$\mathcal{G} = (V, E)$	spatio-temporal graph
E_t	edges connecting temporal layers t and $t+1$.
$v_{t,m}$	spatio-temp. vertex in time layer t corresp. to resource R_m in spatial graph
$z_t^r(i)$	spatio-temp. vertex in time layer t r plans (in coordination iteration i) to use.
$Z_t^r(i)$	spatial vertex (resource) used by r at time t in iteration i.
δ	Kronecker delta
$S(r), D(r)$	start, goal vertex of robot r.
k	positive, uniform edge cost
$ilh^r(i)$	indiv. loss horizon of robot r in coord. iteration i (ref. Def. B.2.1)
$b^r(i)$	bid robot r submits in coord. iteration i
$H(v_{t,m}, R_s)$	min-hop-distance/ geodesic distance between R_m and R_s
$h_i^r(v_{t,m}, R_s)$	$r's$ hop-dist. between nodes in spatio-temp. graph given the history of vertices lost up to i
$V_t^r(i)$	set of reachable nodes in time layer t r has not lost up to coord. iteration i

Table B.1.: Reference table of frequently used notation.

lost resources up to iteration i. The term $hop - distance$ is motivated by the fact that the *length* of the path measures the number of vertex transitions ("hops"). A *shortest* path between two vertices or resources is one with minimum length.

Before proving our termination guarantee, we will need to establish the following two lemmata.

Lemma B.3.1. *Let k be the constant transition cost mentioned in Assumption A3. Assume, in iteration i, Robot r bids for a spatio-temporal vertex $v_{t,m}$ where $t \in \mathbb{N}, m \in \{1, ..., M\}$. Furthermore, assume A1-A5 hold and A6- A8 hold for all time layers of the spatio-temporal graph beyond the robot's loss horizon (i.e. $\forall E_\tau$ where $\tau \geq ilh^r(i)$).*

Then we have

$$b^r(i) = l^r(i) - s^r(i) = \begin{cases} k, & l^r(i) > s^r(i) \\ 0, & otherwise \end{cases}$$

where $b^r(i)$ is Robot r's bid in iteration i and $l^r(i)$, $s^r(i)$ are the path costs estimates under the assumptions that $v_{t,m}$ is not *or* is *available, respectively.*

PROOF. If contested node $v_{t,m}$ is below the robot's individual loss horizon, by construction of the loss list L^r, the bid is $b^r(i) = k - k\,\delta(l^r(i) - s^r(i)) \in \{0, k\}$ (cf. Sec. B.1, Eq. B.1). So, throughout the rest of the proof, we assume $v_{t,m}$ is on or beyond r's individual loss horizon, i.e. $t \geq ilh^r(i)$.

Due to Assumption A3, there is a directly proportional one-to-one correspondence between length of a path (number of time steps until destination) and path cost. That is, any path of hop-length λ incurs a cost of $k * \lambda$ where k is the uniform transition cost. Therefore, any min-cost path is also a shortest path (in terms of the number of hops).

Remember, $l^r(i)$ is the cost of the min-cost path connecting $S(r)$ and $D(r)$ given robot r's belief (in iteration i) about the available resources, under the condition that r will not be allowed to use $v_{t,m}$, i.e. $l^r(i) = k\,(t-1) + k\,\min_{v \in V_t^r(i) - \{v_{t,m}\}} h_i^r(v, D(r))$ where $V_t^r(i)$ is the set of all reachable (given the auction history) nodes in time layer t available to r (ie not lost). Similarly, $s^r(i)$ is the cost of r's min-cost path under the condition that he assumes $v_{t,m}$ to be available. That is, $s^r(i) = k\,(t-1) + k\,h_i^r(v_{t,m}, D(r))$. Notice, we can show with A2 that we have $V_t^r(i) - \{v_{t,m}\} \neq \emptyset$. Hence, $b^r(i) = l^r(i) - s^r(i) = [\min_{v \in V_t^r(i) - \{v_{t,m}\}} k\,h_i^r(v, D(r))] - k\,h_i^r(v_{t,m}, D(r)) \in k\mathbb{N}_0$. That is,

$$b^r(i) \in k\mathbb{N}_0.$$

Let $v^* \in \arg\min_{v \in V_t^r(i) - \{v_{t,m}\}} k\, h_i^r(v, D(r))$ such that v^* would be chosen by r's individual planner if $v_{t,m}$ were lost. Assumption A5 implies $v_{t,m} \in \arg\min_{v \in V_t^r(i)} k\, h_i^r(v, D(r))$, yielding

$$b^r(i) = [k\, h_i^r(v^*, D(r))] - k\, h_i^r(v_{t,m}, D(r)) \begin{cases} = 0, & h_i^r(v_{t,m}, D(r)) = h_i^r(v^*, D(r)) \\ \geq k, & otherwise. \end{cases}$$

Notice, for all vertices $\{v_{\tau,j} | \tau \geq ilh^r(i) = t, j \in \{1, ..., M\}\}$ on or beyond the individual loss horizon the min-hop-distance to the destination equals the conceived hop-distance. That is, we have $H(v_{\tau,j}, D(r)) = h_i^r(v_{\tau,j}, D(r)), \forall \tau \geq ilh^r(i) \forall j$. Hence,

$$b^r(i) = [k\, H(v^*, D(r))] - k\, H(v_{t,m}, D(r)) \begin{cases} = 0, & H(v_{t,m}, D(r)) = H(v^*, D(r)) \\ \geq k, & otherwise. \end{cases} \tag{B.2}$$

Remember, we assume Robot r enters an auction in iteration i to compete for resource (vertex) $v_{t,m}$ which is on or beyond its indiv. loss horizon. Compared to the resource $Z_{t-1}^r(i)$ occupied in the previous time step, visiting contested vertex $v_{t,m}$ would either result in

- (I) the same min-hop-distance ($H(v_{t,m}, D(r)) = H(z_{t-1}^r(i), D(r))$),
- (II) r having moved closer to $D(r)$ ($H(v_{t,m}, D(r)) < H(z_{t-1}^r(i), D(r))$) or,
- (III) r having increased the min-hop-distance ($H(v_{t,m}, D(r)) > H(z_{t-1}^r(i), D(r))$).

In the remainder of the proof we show $b^r(i) \in \{0, k\}$ for each of the three cases.

CASE I: We have two subcases. (I.i) There is an alternative node $v^* \in V_t(i)$ such that $H(v^*, D(r)) = H(v_{t,m}, D(r))$ and hence, $b^r(i) = 0$ (cf. Eq. B.2).

(I.ii) Due to A5 and since the robot had chosen $v_{t,m}$ over $v^* \in V_t(i)$, the only alternative to (I.i) is $H(v^*, D(r)) > H(v_{t,m}, D(r))$. Let $R_j := Z_{t-1}^r(i)$ be the resource occupied in time step $t-1$ according to the current plan. $(t+1) > ilh^r(i)$ implies that Robot r has not lost $v_{t+1,j}$ in an auction and that Assumptions A7 and A8 hold in time layer $t+1$. A7 and A8 yield that $v_{t+1,j}$ can be reached from v^* (incurring cost k). Now, $l^r(i) = k\,(t-1) + k + k\, H(v^*, D(r)) = k\,(t-1) + 2k + k\, H(v_{t+1,j}, D(r)) = k\,(t-1) + 2k + k\, H(v_{t,m}, D(r))$ where the last equality is implied by the fact that $H(v_{t+1,j}, D(r)) = H(v_{t,j}, D(r)) = H(v_{t-1,j}, D(r)) = H(v_{t,m}, D(r))$ and the previous one both from $H(v_{t+1,j}, D(r)) = H(v_{t,m}, D(r))$ and $H(v^*, D(r)) = k + H(v_{t,m}, D(r))$. On the other hand, $s^r(i) = k\,(t-1) + k + k\, H(v_{t,m}, D(r))$. Hence, $b^r(i) = l^r(i) - s^r(i) = k$.

CASE II: Obviously, r's previous column in the spatio-temporal graph changes (i.e. he uses a different resource), since otherwise, the min-hop-distance could not decrease between time steps $t-1$ and t. That is, $R_m \neq R_j$ where $z_{t-1}^r(i) = v_{t-1,j}$. Losing $v_{t,m}$ (ie R_m in time step t) can either mean that (II.i) there is an alternative available vertex in time layer t (Robot r can route a path through) that reduces the min-hop-distance to the destination or, (II.ii) there is not. As always, let v^* denote the vertex r's planning algorithm determines to be the second best to choose in time step t if $v_{t,m}$ is unavailable.

First, assume (II.i). Then $H(v_{t,m}, D(r)) = H(v^*, D(r))$ and thus (cf. Eq. B.2), $b^r(i) = 0$.

Conversely, assume (II.ii). Hence, $l^r(i) > s^r(r)$. Let resource index j be chosen such that $R_j = Z_{t-1}^r(i)$. We have $t \geq ilh^r(i)$.

Assume $t = ilh^r(i)$. Once again, we have multiple subcases: (a) In case $v_{t,j}$ has been lost before, it is listed in loss horizon list L^r (cf. Sec. B.2.1). In that case, due to A5, the robot must have lost $v_{t,m}$ in an even earlier iteration $v_{t,j}$ (since it is the "better node"). Hence, $v_{t,m} \in L^r, v_{t,m} \neq top(L^r)$ yielding $b^r(i) = k$ acc. to Eq. B.1, as explained in Sec. B.2.1. (b) Otherwise, if $v_{t,j}$ has not been lost it is available. That is, intuitively speaking, the robot can opt to keep using resource j also in time step t if beneficial (according to A6).

Likewise, if $t > ilh^r(i)$, $v_{t,j}$ has not been lost in an auction yet in previous iterations.

Since $t+1 > ilh^r(i)$, $v_{t+1,m}$ is also considered available by the robot. Since $v_{t,m}$ was considered the best node to be taken right after $v_{t-1,j}$ by r's planning algorithm and since we assumed our planner

to produce legal plans only, we have $(v_{t-1,j}, v_{t,m}) \in E_{t-1}$. Thus, by A8, we have $(v_{t,j}, v_{t+1,m}) \in E_t$. Hence (cf. Eq. B.2), $b^r(i) = k\,H(v^*, D(r)) - k\,H(v_{t,m}, D(r)) = k\,H(v_{t,j}, D(r)) - k\,H(v_{t,m}, D(r)) = k\,H(v_{t,m}, D(r)) + k - k\,H(v_{t,m}, D(r))$ where the last equality is a direct consequence of A8.

CASE III: In conjunction with Assumption A6, $H(v_{t,m}, D(r)) > H(z^r_{t-1}(i), D(r))$ implies $ilh^r(i) = t$ and that r has lost all nodes $v_t \in V_t(i)$ such that $H(v_t, D(r)) < H(z^r_{t-1}(i), D(r)) \vee H(v_t, D(r)) = H(z^r_{t-1}(i), D(r))$. Hence, $H(v^*, D(r)) > H(z^r_{t-1}(i), D(r))$ where, as always, v^* is the next best node to $v_{t,m}$ in terms of the number of remaining hops to the destination. Let $v_{t-1,j} := z^r_{t-1}(i)$. Since all nodes $v_{\tau,j}$ such that $\tau > t$ have not been lost (since $t \geq ilh^r(i)$), $v_{t+1,j}$ is available. Since $(v_{t-1,j}, v^*) \in E_{t-1} \wedge (v_{t-1,j}, v_{t,m}) \in E_{t-1}$, Assumptions A8 and A7 yield $(v^*, v_{t+1,j}) \in E_t \wedge (v_{t,m}, v_{t+1,j}) \in E_t$. Hence, $H(v^*, D(r)) \leq k + H(v_{t-1,j}, D(r))$ and $H(v_{t,m}, D(r)) \leq k + H(v_{t-1,j}, D(r))$. In conjunction with $H(v^*, D(r)) > H(v_{t-1,j}, D(r)) \wedge H(v_{t,m}, D(r)) > H(v_{t-1,j}, D(r))$ this yields $H(v^*, D(r)) = H(v_{t-1,j}, D(r)) + k = H(v_{t,m}, D(r))$. In conjunction with Eq. B.2 this yields $b^r(i) = 0$.

q.e.d.

Lemma B.3.2. *Assume a graph planning problem with some prior auction history (comprising $i' \in \mathbb{N}_0$ coordination iterations) consistent with our mechanism rules and where A1-A5 hold. Let A be the robot with the highest priority out of all resource-claiming robots $\mathcal{A} = \{1, ..., A\}$ currently in the system. Furthermore, assume A6- A8 are met for all time steps $t \geq \min_{a \in \mathcal{A}} ilh^a(i')$. Finally, for any $i \in \mathbb{N}_0$ let $T^A(i) \in \mathbb{N}_0$ such that $Z^A_{T^A}(i) = D(A)$.*
We have $\forall i > i' : T^A(i) \leq \tau + H(z^A_\tau(i'), D(A)) \in \mathbb{N}_0$ where $\tau := ilh^A(i')$.

PROOF.

Let $i > i'$. We show that, regardless of the spatial-temporal node $z^A_t(i)$ which A plans to occupy in time layer $t > \tau$, he will always be able to decrease his min-hop-distance to his destination in subsequent time layers. That is, $\forall t \geq \tau : H(z^A_t(i), D(a)) > H(z^A_{t+1}(i), D(a))$.

Due to Lemma B.3.1 we know that, in all iterations $i > i'$, all robots submit a bid of k when losing a contested node would entail a detour and a bid of 0 when not. Hence, Robot A is among the highest bidders in an auction for a contested vertex $v_{t,m}$ where losing $v_{t,m}$ would increase the path cost to his goal. In such cases he is guaranteed to win any such critical resources due to the fact that he has the highest priority over competitors with equally high bids. Due to the equivalence of cost and number of hops (A3), this means that for all iterations $i > i'$, A is able to make sure to gain any node beyond $\tau = ilh^A(i')$ and due to planner optimality (A5), we have $\forall t \geq \tau : H(z^A_t(i), D(a)) > H(z^A_{t+1}(i), D(a))$. Remember, all reachable nodes beyond $ilh^A(i')$ are available to Robot A. Since A is already in possession of spatio-temporal node $z^A_\tau(i')$ in iteration i' and since he only loses auctions when not increasing the number of hops on his path between his start and destination, his total path length (in hops) will remain below the number of hops to $z^A_\tau(i')$ (which equals τ) plus $H(z^A_\tau(i'), D(A))$ during all subsequent coordination iterations. **q.e.d.**

Based on these lemmata and our assumptions, we are prepared to show Thm. B.2.2.

Proof of Theorem B.2.2. For convenience we repeat the theorem before proving it:

Theorem B.2.2 *Provided the definitions and assumptions above hold, our modified mechanism generates a joint plan in a finite number of iterations that is legal, collision-free and allows all robots to reach their destinations, regardless of the (mutual exclusive) start states. That is:*

$\forall$ injective $S, D : \mathcal{A} \rightarrow R\, \exists T^1, \ldots, T^A, i \in \mathbb{N}_0 \forall a, r \in \mathcal{A}, a \neq r, t : z^a_t(i) \neq z^r_t(i) \wedge Z^a_{T^a}(i) = D(a) \wedge Z^a_0(i) = S(a)$.

PROOF.

Legality is enforced by Assumption A5. Since our coordination algorithm does not terminate until all plans are collision-free we only need to focus on termination in a finite number of coordination iterations. We prove this by induction on the number A of robots.

Base case. $|\mathcal{A}| = 1$: Is the single-robot case and here, the theorem holds due to Assumptions A1, A3 and A5.
Induction hypothesis: $\exists n \in \mathbb{N} \forall$ agent sets $\mathcal{A}', |\mathcal{A}'| \leq n$: the theorem holds regardless of

(a) the start and destination states of the participating agents and

(b) for any prior auction history (of $i' \in \mathbb{N}_0$ coordination iterations) on that graph that is consistent with the rules of the mechanism and

(c) for any spatio-temporal graph $\mathcal{G}' = (V', E')$ for which our Assumptions A1-A5 hold and such that we have $\forall t \geq \min_{a \in \mathcal{A}} ilh^a(i') : E'_t$ is consistent with Assumptions A6-A8.

Induction step $(n \rightarrow n+1)$: Let i be any current coordination iteration where Robot $A := n+1 \in \mathcal{A}$ is the highest priority robot in the system potentially competing for resources. Lemma B.3.2 implies the existence of $T^A \in \mathbb{N}_0$ such that $\forall \iota \geq i \exists t \leq T^A : Z_t^A(\iota) = D(A)$. By Assumption A4, Robot A hence discontinues claiming any vertices in current spatio-temporal graph $\mathcal{G}$ for time steps $\geq T^A$ and thus, is effectively removed from the system beyond time horizon T^A. Notice, that Lemma B.3.2 also tells us that even if collisions in later time steps among the remaining robots cause collisions in time steps $\leq T^A$, A is guaranteed to reach his destination before T^A and will be absent from claiming resources in later time steps.

Furthermore, the number of collisions (and hence, auctions) that can occur with Robot A are finite since (a) the length of his path is not exceeding T^A spatio-temporal vertices each corresponding to a representant chosen from a finite number of resources (b) the number of possible paths of length T^A is finite. Therefore, the other robots can collide with A only at a finite number of resources. Thus, there exists an iteration $\hat{i}$ from when on no more collisions occur with A.

Let $F = \{a \in \mathcal{A} | ilh^a(\iota) \leq T^A, \forall \iota \geq \hat{i}\}$ be the set of robots that reach their goals along collision free paths until at most time T^A. If $F = \mathcal{A}$ we are done since this implies that all robots manage to find collision-free paths in a finite number of iterations.

Otherwise, we define $\mathcal{A}' = \mathcal{A} - F$ to be the set of robots that require traveling durations to their destinations longer than T^A.

We choose $i' \geq \hat{i}$ such that $\min_{a \in \mathcal{A}'} ilh^a(i') > T^A$. Note, such an iteration exists since by construction, the individual loss horizons are monotonically increasing. Furthermore, notice that $A \in F$ and hence, $|\mathcal{A}'| < |\mathcal{A}| = n+1$.

Coordination of the remaining robots in $\mathcal{A}'$ reduces to a graph planning problem in graph $\mathcal{G}'$ which we construct from spatio-temporal graph $\mathcal{G}$ by pruning $z_0^A(i'), ..., z_{T^A}^A(i')$ from the latter.

Note, $\mathcal{G}'$ is consistent with our assumptions A1-A5. Also, since only nodes before T^A were pruned and $\min_{a \in \mathcal{A}'} ilh^a(i') > T^A$, A6- A8 are met for all time steps $t \geq \min_{a \in \mathcal{A}'} ilh^a(i')$ as well as required by the induction hypothesis and Lemma B.3.2. Since $|\mathcal{A}'| < n+1$, the induction hypothesis entails that this remaining coordination problem is solved in a finite number of coordination iterations. **q.e.d.**

C. Supplementary derivations for kinky inference

C.1. Convergence properties

We will construct function estimates that converge to the ground truth in the limit of large sample sets, provided the samples fulfil certain convergence properties themselves. To establish this in a rigorous manner, we need to review a few core convergence concepts.

Definition C.1.1 (Uniform convergence of function sequences)**.** *Let $(\mathcal{X}, \mathfrak{d}_{\mathcal{X}}), (\mathcal{Y}, \mathfrak{d}_{\mathcal{Y}})$ be two metric spaces. For $n \in \mathbb{N}$, let $f_n : I \to \mathbb{R}$. $f_n \overset{n\to\infty}{\longrightarrow} f$* uniformly *(with respect to $\mathfrak{d}_{\mathcal{Y}}$) if*

$$\forall \epsilon > 0 \exists N_0 \in \mathbb{N} \forall n \geq N_0 \forall x \in I : \mathfrak{d}_{\mathcal{Y}}(f_n(x), f(x)) < \epsilon.$$

The key distinction between both variants of convergence is that in point-wise convergence, the convergence rate can depend on x. By contrast, in uniform convergence the rate is independent of function input x.

For our purpose the following definition of a dense set suffices:

Definition C.1.2 (Dense)**.** *Let $(\mathcal{X}, \mathfrak{d})$ be a metric space. A set $D \subset \mathcal{X}$ is ($\mathfrak{d}$-) dense in a set $I \subset \mathcal{X}$ if every $\mathfrak{d}-$ open subset of I contains at least one point of D. That is: $\forall x \in I, \epsilon > 0 \exists \xi \in D : \mathfrak{d}(\xi, x) < \epsilon$.*

Definition C.1.3 (Uniform convergence of sequences of sets)**.** *Let $\left(\mathcal{X}, \mathfrak{d}\right)$ be a metric space. A sequence of sets $(Q_n)_{n\in\mathbb{N}}$, $Q_n \subset \mathcal{X}$, $|Q_n| < \infty$ converges to set $Q \subset \mathcal{X}$ uniformly if we have:*

$$\forall \epsilon > 0 \exists N_0 \in \mathbb{N} \forall n \geq N_0 \forall q \in Q \exists q_n \in Q_n : \mathfrak{d}(q_n, q) \leq \epsilon.$$

If the sequence consists of finite sets, we can rewrite the condition as

$$\forall \epsilon > 0 \exists N_0 \in \mathbb{N} \forall n \geq N_0 \forall q \in Q : \min_{q_n \in Q_n} \mathfrak{d}(q_n, q) \leq \epsilon.$$

In analogy to the intuition of uniform convergence of function sequences, uniform convergence of sets means that the rate of convergence does not depend on the location within the set.

In preparation for the derivations below, it will be useful to establish that uniform convergence to a dense subset of set I implies uniform convergence to I itself. Formally we express this as follows:

Lemma C.1.4. *Let Q be a dense subset of I and let Q_n be a sequence of sets uniformly converging to Q in the limit of $n \to \infty$. We have*

$$\forall e > 0 \exists N_0 \forall n \geq N_0 \forall x \in I \exists s \in Q_n : \mathfrak{d}(x, s) \leq e.$$

Proof. Let $e > 0$, $N_0 \in \mathbb{N}$ such that for all $n \geq N_0$ we have $\forall q \in Q \exists s_n(q) \in Q_n : \mathfrak{d}(s_n(q), q) \leq \frac{e}{2}$. Now, choose any $x \in I, n \geq N_0$. Since Q is dense in I we can choose $q \in Q$ such that $\mathfrak{d}(q, x) \leq \frac{e}{2}$. Hence, utilising the triangle inequality, we conclude $\mathfrak{d}(x, s_n(q)) \leq \mathfrak{d}(x, q) + \mathfrak{d}(q, s_n(q)) \leq \frac{e}{2} + \frac{e}{2} = e$. □

An alternative statement of the lemma is that, under the lemma's premise, we have

$$\forall e > 0 \exists N_0 \forall n \geq N_0 \forall x \in I : \inf_{s \in Q_n} \mathfrak{d}(x, s) \leq e.$$

Definition C.1.5 ("Eventually dense partition ")**.** *For each $n \in \mathbb{N}$ let $\mathcal{P}_n := \{P_{n,1}, ...P_{n,n}\}$ be a partition of I into subsets, each being a connected set and having the property that $P_{n,i}$ contains exactly one sample*

input s_i. Furthermore, let $r_{n,i} = \sup_{x,x' \in P_{n,i}} \frac{1}{2} \|x - x'\|$ be $P_{n,i}$'s "radius" w.r.t norm $\|\cdot\|$. We call $(\mathcal{P}_n)_{n\in\mathbb{N}}$ an eventually dense *partition sequence if the sequence (r_n^*) of maximum radii, $r_n^* := \max\{r_{n,i}\}$, converges to zero as $n \to \infty$.*

A sequence of inputs $(G_n)_{n\in\mathbb{N}}$ will be called "eventually dense" if it is impossible to define a partition sequence separating the points in G_n for each n such that G_n is not eventually dense.

It should be clear that a sample grid that converges to a dense subset in I is eventually dense (and vice versa).

It will become useful to remind ourselves of the standard fact that monotone sequences converge if they are bounded:

Theorem C.1.6 (Monotone convergence theorem)**.** *Let $(a_n)_{n\in\mathbb{N}}$ be a monotone sequence of real numbers. That is, $a_n \geq a_{n+1}, \forall n \in \mathbb{N}$ or $a_n \leq a_{n+1}, \forall n \in \mathbb{N}$. The sequence converges to a finite real number if and only if the sequence is bounded. In particular, if the sequence is monotonically increasing then $\lim_{n\to\infty} a_n = \sup a_n$. If it is decreasing, we have $\lim_{n\to\infty} a_n = \inf a_n$.*

This theorem can be directly applied to prove the following result:

Corollary C.1.7. *Let $(\phi_n)_{n\in\mathbb{N}}$, $\phi_n : I \to \mathbb{R}$ be a point-wise monotonically increasing sequence of functions and, let $(\gamma_n)_{n\in\mathbb{N}}$, $\gamma_n : I \to \mathbb{R}$ be a point-wise monotonously decreasing sequence of functions. That is, $\forall x \in I \forall n \in \mathbb{N} : \phi_n(x) \leq \phi_{n+1}(x) \wedge \gamma_n(x) \geq \gamma_{n+1}(x)$. Furthermore, we assume $\forall x \in I \forall n \in \mathbb{N} : \phi_n(x) \leq \gamma_n(x)$.*

Then, both sequences are point-wise convergent with point-wise limits $\lim_{n\to\infty} \phi_n(x) = \sup_{n\in\mathbb{N}} \phi_n \leq \inf_{n\in\mathbb{N}} = \lim_{n\to\infty} \gamma_n(x)$.

The corollary will be needed below when we establish convergence of our function estimate bounds.

C.1.1. Convergence with respect to the exponentiated metric

The purpose of the next two results is to show that it does not matter whether one establishes a convergence property with respect to a metric $\mathfrak{d}$ or whether one does so with respect to $\mathfrak{d}^p$.

Lemma C.1.8. *Let $p \in (0, 1]$, $\mathcal{X}$ be a metric space and $(x_n)_{n\in\mathbb{N}}$ be a sequence in $\mathcal{X}$ converging to $x \in \mathcal{X}$ with respect to metric $\mathfrak{d}$ iff the sequence converges to x with respect to $\mathfrak{d}^p$.*

Proof. $\Rightarrow$: Define the sequence $d_n := \mathfrak{d}(x_n, x)$. Convergence with respect to $\mathfrak{d}$ means that $d_n \to 0$ as $n \to \infty$. Since $d_n \geq$ mapping $\psi : \mathfrak{x} \mapsto \mathfrak{x}^p$ is well-defined for sequence inputs d_n. Furthermore, ψ is a continuous function and thus, $\lim_{n\to\infty} \mathfrak{d}^p(x_n, x) = \lim_{n\to\infty} \psi(d_n) = \psi(\lim_{n\to\infty} d_n) = \psi(0) = 0$. Hence, x_n converges to x with respect to $\mathfrak{d}^p$ as well. $\Leftarrow$: Completely analogous.

□

Lemma C.1.9. *Let $p \in (0, 1]$, $\mathcal{F}$ be a metric space of functions. Sequence of functions $(f_n)_{n\in\mathbb{N}}$ converges uniformly to f with respect to metric $\mathfrak{d}$ iff the sequence also converges uniformly to f with respect to $\mathfrak{d}^p$.*

Proof. $\Rightarrow$: Since $f_n \to f$ uniformly, we have $\forall e > 0 \exists N_e \geq 0 \forall n \geq N_e, x \in I : \mathfrak{d}(f_n, f) < e$. We show that also $\forall e > 0 \exists N_e \geq 0 \forall n \geq N_e, x \in I : \mathfrak{d}(f_n, f)^p < e$. Mapping $\psi : x \mapsto x^p$ invectively maps positive numbers onto the set of positive real numbers. Hence, $\mathfrak{d}(f_n, f) < e \Leftrightarrow \mathfrak{d}(f_n, f)^p < e^p$. Let $e > 0$ and $\varepsilon := \sqrt[p]{e}$. Due to uniform convergence, we can choose N_ε such that $\forall x \in I, n \geq N_\varepsilon : \mathfrak{d}(f_n, f) < \varepsilon$ which is equivalent to stating $\forall x \in I, n \geq N_\varepsilon : \mathfrak{d}(f_n, f)^p < \varepsilon^p = e$. $\Leftarrow$: Completely analogous. □

C.2. Theoretical guarantees of the enclosure

Remember, we defined the set of samples as $D_n = \{(s_i, \tilde{f}_i, \varepsilon(s_i))|i = 1, \ldots, N_n\}$. To aide our discussion, we define $G_n = \{s_i|i = 1, \ldots, N_n\}$ to be the grid of sample inputs contained in D_n. Our exposition of this subsection focusses on the case of one-dimensional output space. That is, $\mathcal{Y} = \mathbb{R}$ and we assume to be the target function f to be a mapping $f : I \subseteq \mathcal{X} \to \mathbb{R}$.

In this subsection, we assume $\mathfrak{d}_{\mathcal{X}}$ to be any suitable metric on the target function domain and $\tilde{d}(x, x') := \mathfrak{d}_{\mathcal{X}}^p(x, x')$. We have seen that (uniform) convergence with respect to $\tilde{d}$ is equivalent to (uniform) convergence with respect to metric $\mathfrak{d}$ (see Sec. C.1.1). Also note, that Lipschitz continuity with respect to $\tilde{d}$ is Hölder continuity with respect to metric $\mathfrak{d}_{\mathcal{X}}$. Furthermore, remember we are primarily interested in functions restricted to be Hölder with respect to $\mathfrak{d}$ and bounded by functions $\underline{B}$, $\bar{B}$. That is, we know the target is contained in $\mathcal{K}_{prior} = \text{Lip}_{\tilde{d}}(L) \cap \mathcal{B}$ where $\text{Lip}_{\mathfrak{d}}(L)$ denotes the L- Lipschitz functions with respect to $\tilde{d}$ and $\mathcal{B} := \{\phi : \mathcal{X} \to \mathbb{R} \,|\, \forall x \in \mathcal{X} : \underline{B}(x) \leq \phi(x) \leq \bar{B}(x)\}$.

As a final note, at no point will we utilise the definiteness property of the metric. Therefore, our definitions and results extend from metrics to the more general class of pseudo-metrics.

Definition C.2.1 (Sample-consistent functions)**.** *A function $\kappa : I \to \mathbb{R}$ is called consistent with sample D_n (shorthand:* sample-consistent*) if*

$$\forall i \in \{1, \dots, N_n\} : \kappa(s_i) \in [\tilde{f}_i - \varepsilon(s_i), \tilde{f}_i + \varepsilon(s_i)] = [\underline{f}(s_i), \overline{f}(s_i)].$$

The set of all sample-consistent functions on I will be denoted by $\mathcal{K}(D_n)$.

Informally speaking, $\mathcal{K}(D_n)$ contains all functions that could have created the sample set. Conversely, sample D_n rules out all functions that are not sample-consistent as candidates for being the target.

Definition C.2.2 (Enclosure)**.** *Let $\mathfrak{u}, \mathfrak{l} : I \to \mathbb{R}$ be two functions, such that $\mathfrak{u} \geq \mathfrak{l}$ pointwise on I. We define their enclosure on I as the compact interval $\mathcal{E}_{\mathfrak{l}}^{\mathfrak{u}}(I)$ of all functions between them. That is,*

$$\mathcal{E}_{\mathfrak{l}}^{\mathfrak{u}}(I) := \{\phi : I \to \mathbb{R} | \forall t \in I : \mathfrak{u}(t) \geq \phi(t) \geq \mathfrak{l}(t)\}.$$

For ease of notation we omit the domain argument whenever we refer to I. That is, we write $\mathcal{E}_{\mathfrak{l}}^{\mathfrak{u}} := \mathcal{E}_{\mathfrak{l}}^{\mathfrak{u}}(I)$.

An *enclosure* is sample-consistent if it is contained in the set $\mathcal{K}(D_n)$ of sample-consistent functions.

Based on the sample, we desire to find two *enclosing functions* $\mathfrak{u}_n : I \to \mathbb{R}$, $\mathfrak{l}_n : I \to \mathbb{R}$ bounding the target from above and below that are as tight as possible and converge to the target in the limit of infinite, sample set size $N \to \infty$. More formally, we would ideally like it to satisfy the following desiderata:

Definition C.2.3 (Desiderata)**.** *Based on an observed, erroneous data set D_n, we desire to define enclosing functions $\mathfrak{u}_n, \mathfrak{l}_n$ that give rise to an enclosure $\mathcal{E}_{\mathfrak{l}_n}^{\mathfrak{u}_n}$ with the following properties:*

1. ***Sample-Consistency**: The enclosure is sample-consistent, i.e. $\mathcal{E}_{\mathfrak{l}_n}^{\mathfrak{u}_n} \subseteq \mathcal{K}(D_n)$.*
2. ***Exhaustiveness (conservatism)**: Every ϕ, $\underline{B} \leq \phi \leq \bar{B}$ coinciding with target f on with grid G_n and having at most constant $L[J]$ on every subset $J \subset I$ is also contained in the enclosure: $\forall \phi : I \to \mathbb{R}, \phi \in \mathcal{K}_{prior} \cap \mathcal{K}(D_n) : \phi \in \mathcal{E}_{\mathfrak{l}_n}^{\mathfrak{u}_n}$. That is, $\mathcal{K}_{prior} \cap \mathcal{K}(D_n) \subseteq \mathcal{E}_{\mathfrak{l}_n}^{\mathfrak{u}_n}$. In particular, the target is always contained in the enclosure: $f \in \mathcal{E}_{\mathfrak{l}_n}^{\mathfrak{u}_n}$.*
3. ***Monotonicity:** Additional data does not increase uncertainy. That is, if $G_n \subseteq G_{n+1}$, we have $\mathcal{E}_{\mathfrak{l}_{n+1}}^{\mathfrak{u}_{n+1}} \subseteq \mathcal{E}_{\mathfrak{l}_n}^{\mathfrak{u}_n}, \forall n \in \mathbb{N}$.*
4. ***Convergence** to the target: Let the set of sample inputs G_n in D_n converge to a dense subset of domain I and for each input $x \in I$ let the sample error function $\varepsilon : I \to \mathbb{R}_{\geq 0}$ be bounded by $m_x \geq 0$ in some neighbourhood of x. Then, we have*
   - *(a) The sequence of enclosures converges point-wise. In particular,*

     $\forall x \in I : 0 \leq \lim_{n \to \infty} \mathfrak{u}_n(x) - \mathfrak{l}_n(x) \leq 2m_x$. *In particular, the enclosure converges to the target $f(x)$ at inputs x where $m_x = 0$.*
   - *(b) If convergence of G_n is uniform and the target observations are error-free, i.e. $m_x = 0, \forall x \in I$, then the enclosure converges to the target f uniformly.*
5. ***Minimality (optimality)**: If an enclosure $\mathcal{E}_a^b$ satisfies* Desideratum 2 *then it is a superset of our enclosure $\mathcal{E}_{\mathfrak{l}_n}^{\mathfrak{u}_n}$, i.e. $b \geq \mathfrak{u}_n, a \leq \mathfrak{l}_n$, $\mathcal{E}_{\mathfrak{l}_n}^{\mathfrak{u}_n} \subseteq \mathcal{E}_a^b$.*

Next, we will derive enclosing functions that give rise to such an enclosure.

Definition C.2.4 (Sample point ceiling and floor functions)**.** *Remember,* $\overline{f}(s_i) = \tilde{f}_i + \varepsilon(s_i)$ *and* $\underline{f}(s_i) = \tilde{f}_i - \varepsilon(s_i)$*. For the* i*th sample (*$i \in \{1, \dots, N_n\}$*,* $L \geq 0$*,* $x \in I$*, we define*

1. *The* i*th* sample point ceiling *function:*

$$\mathfrak{u}_n^i(x; L) : x \mapsto \overline{f}(s_i) + L\ \tilde{d}(x, s_i),$$

2. *and the* i*th* sample point floor *function:*

$$\mathfrak{l}_n^i(x; L) : x \mapsto \underline{f}(s_i) - L\ \tilde{d}(x, s_i).$$

If constant L *is clear from the context, its explicit mention shall be omitted.*

We show that these functions give rise to enclosures that satisfy the first three Desiderata of Def. C.2.3:

Lemma C.2.5. *Let* $i \in \{1, ..., N_n\}$*. Enclosure* $\mathcal{E}_{\mathfrak{l}_n^i}^{\mathfrak{u}_n^i}$ *is (i) consistent with the* i*th sample and (ii) conservative. More specifically:*

(i)

$$\forall i \in \{1, ..., N_n\}, \phi \in \mathcal{E}_{\mathfrak{l}_n^i}^{\mathfrak{u}_n^i} : \phi(s_i) \in [\underline{f}(s_i), \overline{f}(s_i)].$$

(ii) If target $f : I \to \mathbb{R}$ *is* L*-Lipschitz on* $J \subset I$ *with respect to* $\tilde{d}$ *then the corresponding sample ceilings are upper bounds and the sample floor functions are lower bounds. That is, for* $i = 1, \dots, N$*,* $\forall x \in J$ *we have*

$$\mathfrak{u}_n^i(x; L) \geq f(x) \geq \mathfrak{l}_n^i(x; L).$$

Proof. (i) Follows directly from the definitions. (ii) We show the first inequality, $\mathfrak{u}_n^i(x; L) \geq f(x)$. (The proof of the second inequality, $f(x) \geq \mathfrak{l}_n^i(x; L)$, is completely analogous.)

Due to the Hölder property, we have $\forall x, x' \in J : |f(x) - f(x')| \leq L\, \tilde{d}(x, x')$. In particular, $\forall x \in J : f(x) - f(s_i) \leq |f(x) - f(s_i)| \leq L\ \tilde{d}(x, s_i)$. Hence, $\forall x \in J : f(x) \leq f(s_i) + L\ \tilde{d}(x, s_i) \leq \tilde{f}(s_i) + \varepsilon(s_i) + L\ \tilde{d}(x, s_i) = \mathfrak{u}_n^i(x; L)$.

□

Definition C.2.6 (Hölder enclosure)**.** *On* $J \subset I$*, define the* optimal ceiling *as*

$$\mathfrak{u}_n^*(x; L) : x \mapsto \min_{i=1,\dots,N_n} \mathfrak{u}_n^i(x; L)$$

and the optimal floor function *as*

$$\mathfrak{l}_n^*(x; L) : x \mapsto \max_{i=1,\dots,N_n} \mathfrak{l}_n^i(x; L).$$

The enclosure $\mathcal{E}_{\mathfrak{l}_n}^{\mathfrak{u}_n}$ *with enclosing function choices* $\mathfrak{u}_n := \mathfrak{u}_n^*$ *and* $\mathfrak{l}_n := \mathfrak{l}_n^*$ *will be called "Hölder enclosure" and be denoted by* $\mathcal{E}_n^*$*. If constant* L *is fixed and clear from the context we may omit its explicit mention from the syntax.*

Before we can prove that the optimal functions are indeed optimal, we need to undergo some derivative preparations.

Lemma C.2.7. *The Hölder enclosure always contains the target. That is,* $\mathfrak{u}_n^*(x) \geq f(x) \geq \mathfrak{l}_n^*(x), \forall x \in I, n \in \mathbb{N}$.

Proof. This is a direct consequence of Lem. C.2.5. □

Theorem C.2.8. *Choosing* $\mathfrak{u}_n := \min\{\mathfrak{u}_n^*, \bar{B}\}$ *and* $\mathfrak{l}_n := \max\{\mathfrak{l}_n^*, \underline{B}\}$ *yields an enclosure* $\mathcal{E}_{\mathfrak{l}_n}^{\mathfrak{u}_n}$ *that satisfies Desiderata 1-5 specified in Def. C.2.3.*

Proof. We bear in mind that $\mathcal{E}^{\mathfrak{u}_n}_{\mathfrak{l}_n} = \mathcal{B} \cap \mathcal{E}^*_n$.

1) Consistency. By definition of $\mathfrak{u}^*_n$ and $\mathfrak{l}^*_n$ in conjunction with Lem. C.2.5.(i) we see that $\mathcal{E}^*_n = \cap_i \mathcal{E}^{\mathfrak{u}^i_n}_{\mathfrak{l}^i_n} \subseteq \mathcal{K}(D_n)$. Desideratum 1 follows from $\mathcal{E}^{\mathfrak{u}_n}_{\mathfrak{l}_n} = \mathcal{B} \cap \mathcal{E}^*_n \subseteq \mathcal{E}^*_n$.

2) Conservatism. By definition of $\mathfrak{u}^*_n$ and $\mathfrak{l}^*_n$ in conjunction with Lem. C.2.5.(ii) we have $\mathcal{K}(D_n) \cap \text{Lip}_{\tilde{d}}(L) \subseteq \mathcal{E}^*_n$. Hence, $\mathcal{K}_{prior} \cap \mathcal{K}(D_n) = \mathcal{B} \cap \text{Lip}_{\tilde{d}}(L) \cap \mathcal{K}(D_n) \subseteq \mathcal{E}^*_n \cap \mathcal{B} = \mathcal{E}^{\mathfrak{u}_n}_{\mathfrak{l}_n}$ proving Desideratum 2.

3) Monotonicity.

This follows directly from the way the ceiling and floor functions are defined. Considering the ceiling: We have $\mathfrak{u}_n(x) = \min\{\bar{B}, \min_{i=1,\dots,N_n} \mathfrak{u}^i_n(x)\}$. W.l.o.g., for $i = 1, \dots, N_n$, let $\mathfrak{u}^i_n = \mathfrak{u}^i_{n+1}$.

Then $\mathfrak{u}_{n+1}(x) = \min\{\bar{B}(x), \min\Big\{\min_{i=1,\dots,N_n} \mathfrak{u}^i_n(x), \min_{i=N_n+1,\dots,N_{n+1}} \mathfrak{u}^i_{n+1}(x)\Big\}\}$

$\leq \min\{\bar{B}(x), \min_{i=1,\dots,N_n} \mathfrak{u}^i_n(x)\} = \mathfrak{u}_n(x)$. Statement $\mathfrak{l}_{n+1}(x) \geq \mathfrak{l}_n(x)$ follows analogously.

4 a) Pointwise convergence. In absence of errors, one could show pointwise convergence by utilising Cor. C.1.7 –which is applicable due to Lem. C.2.5 in conjunction with 3). We will now prove the more general statement in the presence of observational errors ε.

Pick an arbitrary $x \in I$ such that the error $\varepsilon(\xi) \leq m_x$ for all ξ in some $\epsilon-$ball $\mathcal{U}_\epsilon(x) = \{\xi \in I | \, \tilde{d}(\xi, x) < \epsilon\}$ around x. It remains to be shown that $\lim_{n\to\infty} \mathfrak{u}_n(x) - \mathfrak{l}_n(x) \leq 2m_x$ for inputs x. Define $L := \sup_{J\subset I} L[J]$ to be a Hölder constant on the domain. Let $e > 0$ be arbitrary. We show that there exists $N_0 \in \mathbb{N}$ such that for all $n \geq N_0 : \mathfrak{u}_n(x; L) - \mathfrak{l}_n(x; L) \leq e + 2m_x$ as follows:

Since we assume G_n converges to a dense subset of I, Lem. C.1.9 applies. Hence, there is $N_0 \in \mathbb{N}$ such that $\forall n \geq N_0 \exists i \in \{1, ..., N_n\} : s_i \in \mathcal{U}_{e/2L}(x) = \{\xi \in I | \, \tilde{d}(x, \xi) < \frac{e}{2L}\}$.

So, choose $n \geq N_0$. Hence, $L\, \tilde{d}(x, s_q) < \frac{e}{2}$, $\varepsilon(s_q) \leq m_x$ (*)

where $q \in \arg\min_{i\in\{1,\dots,N_n\}} L\ \tilde{d}(x, s_i)$. We have:

$\mathfrak{u}^*_n(x; L) - \mathfrak{l}^*_n(x; L)$

$= \min_{i\in\{1,\dots,N_n\}} L\ \tilde{d}(x, s_i) + \tilde{f}(s_i) + \varepsilon(s_i) - \max_{i\in\{1,\dots,N_n\}} \tilde{f}(s_i) - \varepsilon(s_i) - L\, \tilde{d}(x, s_i)$

$= \min_{i\in\{1,\dots,N_n\}}\{L\ \tilde{d}(x, s_i) + \tilde{f}(s_i) + \varepsilon(s_i)\} + \min_{i\in\{1,\dots,N_n\}}\{-\tilde{f}(s_i) + \varepsilon(s_i) + L\ \tilde{d}(x, s_i)\}$

$\leq \min_{i\in\{1,\dots,N_n\}}\{L\ \tilde{d}(x, s_i) + \tilde{f}(s_i) + \varepsilon(s_i) - \tilde{f}(s_i) + \varepsilon(s_i) + L\, \tilde{d}(x, s_i)\}$

$= 2\min_{i\in\{1,\dots,N_n\}} L\ \tilde{d}(x, s_i) + \varepsilon(s_i) \leq 2L\ \tilde{d}(x, s_q) + 2\varepsilon(s_q) \overset{(*)}{<} e + 2m_x.$

As a special case, in the noise free setting with $m_x = 0$, this result implies that the enclosure shrinks to a single value as $n \to \infty$. That this value is $f(x)$ follows from Lem. C.2.7.

We have shown convergence of $\mathfrak{u}^*_n(x; L) - \mathfrak{l}^*_n(x; L)$. Convergence of $\mathfrak{u}_n(x) - \mathfrak{l}_n(x) = \min\{\bar{B}(x), \mathfrak{u}^*_n(x)\} - \max\{\underline{B}(x), \mathfrak{l}^*_n(x)\}$ follows by continuity of the max and min operation.

4 b) Uniform convergence. Let $G_n = \{s_1, \dots, s_{N_n}\}$ denote the N_n-sample input grid which we assume to converge uniformly to a dense set of domain I.

(I) Firstly, we show uniform convergence of $\mathfrak{u}^*_n, \mathfrak{l}^*_n$ to target f.

Assuming $m_x = 0$, we desire to show that $\forall e > 0 \exists N_0 \forall n \geq N_0, x \in I : |\mathfrak{u}^*_n(x) - f(x)| \leq e$ and $\forall e > 0 \exists N_0 \forall n \geq N_0, x \in I : |\mathfrak{l}^*_n(x) - f(x)| \leq e$. Let $e > 0$. Let G_n converge uniformly to a dense subset $G \subset I$. By Lem. C.1.4, we know that there exists N_0 such that for all $n \geq N_0$ and all $x \in I$ there is a $s_q \in G_n$ such that $\tilde{d}(x, s_q) < \frac{e}{2L}$. In particular, $L\ \tilde{d}(s_q, x) \leq \frac{e}{2}$ where we define $q \in \arg\min_i \tilde{d}(s_i, x)$. Hold such N_0 fixed, choose arbitrary $n \geq N_0$ and $x \in I$.

For starters, we show that each $\mathfrak{u}^*_n$ converges uniformly to f by showing $|\mathfrak{u}^*_n(x) - f(x)| \leq e$.

We have $|\mathfrak{u}^*_n(x) - f(x)| \overset{2)}{=} \mathfrak{u}^*_n(x) - f(x) \overset{def.}{\leq} \min_i \mathfrak{u}^i_n(x) - f(x) \leq \mathfrak{u}^q_n(x) - f(x) \overset{def.}{\leq} L\ \tilde{d}(x, s_q) + f(s_q) - f(x) \leq L\ \tilde{d}(x, s_q) + |f(s_q) - f(x)| \overset{f \text{ L-Lip.}}{\leq} L\ \tilde{d}(x, s_q) + L\ \tilde{d}(x, s_q) \leq e$ where in the last step we have leveraged $L\ \tilde{d}(x, s_q) \leq \frac{e}{2}$.

Next, we show the analogous for the floor. That is we show $|\mathfrak{l}^*_n(x) - f(x)| \leq e$:

$|\mathfrak{l}^*_n(x) - f(x)| \overset{2)}{=} f(x) - \mathfrak{l}^*_n(x) \overset{def.}{\leq} f(x) - \max_i \mathfrak{l}^i_n(x) \leq f(x) - \mathfrak{l}^q_n(x) \overset{def.}{\leq} f(x) - (f(s_q) - L\ \tilde{d}(x, s_q))$ $\leq L\ \tilde{d}(x, s_q) + |f(s_q) - f(x)| \leq L\ \tilde{d}(x, s_q) + L\ \tilde{d}(x, s_q) \leq 2\frac{e}{2} = e$ where, as before, in the last step we have utilised $L\ \tilde{d}(x, s_q) \leq \frac{e}{2}$.

(II) Secondly, we need to show that $\mathfrak{u}_n(\cdot) = \max\{\underline{B}(\cdot), \mathfrak{u}_n^*(\cdot)\}$ and $\mathfrak{l}_n = \min\{\underline{B}(\cdot), \mathfrak{l}_n^*(\cdot)\}$ converge uniformly to f. However, this presents no difficulty since we know that $f(x) \in [\underline{B}(x), \bar{B}(x)]$. Let $e \geq 0$ be arbitrary. We need to show $\exists \tilde{N}_0(e) \forall n \geq \tilde{N}_0(e), x \in I : |\mathfrak{u}_n(x) - f(x)| \leq e$. In (I), we already have established $\exists N_0(e) \forall n \geq N_0(e), x \in I : |\mathfrak{u}_n^*(x) - f(x)| \leq e$. We choose $\tilde{N}_0(e) := N_0(e)$. Since $f \leq \mathfrak{u}_n = \min\{\bar{B}(x), \mathfrak{u}_n^*\} \leq \mathfrak{u}_n^*$ pointwise, for all $x \in I$, we have $e \geq |\mathfrak{u}_n^*(x) - f(x)| = \mathfrak{u}_n^*(x) - f(x) \geq \mathfrak{u}_n(x) - f(x) = |\mathfrak{u}_n(x) - f(x)|$.

Uniform convergence of the floor $\mathfrak{l}_n$ follows completely analogously.

5) Optimality. Utilising concavity and definiteness of $h : x \mapsto x^p$ for $p \in (0, 1]$ it is easy to show that h is sub-additive. It follows that the mappings $\tilde{d}(\cdot, x)$ $(x \in I)$ are in $\text{Lip}_{\tilde{d}}(L)$. Based on this observation, Lem. 2.5.6 allows us to conclude that $\mathfrak{u}_n^i, \mathfrak{l}_n^i, \mathfrak{u}_n^*, \mathfrak{l}_n^* \in \text{Lip}_{\tilde{d}}(L)$. Hence, $\mathfrak{u}_n, \mathfrak{l}_n \in \text{Lip}_{\tilde{d}}(L) \cap \mathcal{B}$. Thus, if $\text{Lip}_{\tilde{d}}(L) \cap \mathcal{B} \subset \mathcal{E}_a^b$ then also $\mathfrak{u}_n, \mathfrak{l}_n \in \mathcal{E}_a^b$. This implies $\mathfrak{u}_n \leq b, \mathfrak{l}_n \geq a$, i.e. $\mathcal{E}_{\mathfrak{l}_n}^{\mathfrak{u}_n} \subseteq \mathcal{E}_a^b$. □

C.2.1. Proof of Thm. 4.2.7

We are now in a position to conclude the statement of Theorem 4.2.7 which is restated here for convenience:

Theorem 4.2.7:

Assuming all (a priori) possible targets $f : \mathcal{X} \to \mathcal{Y}$ are given by the class

$$\mathcal{K}_{prior} = \left\{ f : \mathcal{X} \to \mathcal{Y} \subseteq \mathbb{R}^m | f \in \mathfrak{H}_{\mathfrak{d}_\mathcal{X}}(L, p) \cap \mathcal{B} \right\} \tag{C.1}$$

where $\mathfrak{H}_{\mathfrak{d}_\mathcal{X}}(L, p) = \left\{ f : \mathcal{X} \to \mathcal{Y} \subset \mathbb{R}^m \,|\forall j \in \{1, \ldots, m\} \forall x, x' \in \mathcal{X} : |f_j(x) - f_j(x')| \leq L_j \, \mathfrak{d}_\mathcal{X}^p(x, x') \right\}$ denotes the class of $L - p$- Hölder continuous functions with respect to metric $\mathfrak{d}_\mathcal{X}$ and $\mathcal{B} := \{\phi : \mathcal{X} \to \mathcal{Y} \,|\, \forall x \in \mathcal{X}, j \in \{1, ..., m\} : \underline{B}_j(x) \leq \phi_j(x) \leq \bar{B}_j(x)\}$ is the set of all functions bounded component-wise between functions $\underline{B}, \bar{B} : \mathbb{R}^d \to \mathbb{R} \cup \{-\infty, \infty\}$, where we will always define $\mathbb{R}_\infty := \mathbb{R} \cup \{-\infty, \infty\}$.

Then we have: The full kinky inference rule (KI) as per Def. 4.2.4 (with $w_\mathfrak{u}(x) = w_\mathfrak{l}(x) = \frac{1}{2}$ and $\mathcal{I}_n(x) = \{1, \ldots, N_n\}, \forall x \in \mathcal{X}$) is conservative, monotonically convergent (in the limit of dense sample grids) and optimal in the sense of Def. 4.2.2. That is, it satisfies Desiderata 1-4 as per Def. 4.2.2.

Proof. We desire to apply Thm. C.2.8 to each component function. Inspecting the definitions, this turns out to establish the desired statement for each component function of the predictors and target. Since the desiderata as per Def. 4.2.2 are requirements for each output component, this will conclude the proof.

In detail: We prove the claim for each output component $j = 1, ..., m$. By assumption, the target's component function f_j is $L_j - p$ Hölder. Remember from Def. 4.2.4, $\hat{\mathfrak{f}}_{\mathfrak{n},\mathfrak{j}}(x) := \frac{1}{2} \min\{\bar{B}_j(x), \tilde{\mathfrak{u}}_{n,j}(x)\} + \frac{1}{2} \max\{\underline{B}_j, \tilde{\mathfrak{l}}_{n,j}(x)\}$ and $\hat{\mathfrak{v}}_{\mathfrak{n},\mathfrak{j}}(x) := \frac{1}{2}\big(\min\{\bar{B}_j(x), \tilde{\mathfrak{u}}_{n,j}(x)\} - \max\{\underline{B}_j, \tilde{\mathfrak{l}}_{n,j}(x)\}\big)$ with component functions given by $\tilde{\mathfrak{u}}_{n,j}(x) := \min_{i=1,\ldots,N_n} \tilde{f}_{i,j} + L_j \, \mathfrak{d}^p(x, s_i) + \varepsilon_j(x)$ and $\tilde{\mathfrak{l}}_{n,j}(x) := \max_{i=1,\ldots,N_n} \tilde{f}_{i,j} - L_j \, \mathfrak{d}^p(x, s_i) - \varepsilon_j(x)$, respectively. Connecting to Thm. C.2.8, for $x \in \mathcal{X}$, we define $\mathfrak{u}_n(x) := \hat{\mathfrak{f}}_{\mathfrak{n},\mathfrak{j}}(x) + \hat{\mathfrak{v}}_{\mathfrak{n},\mathfrak{j}}(x) = \min\{\bar{B}_j(x), \tilde{\mathfrak{u}}_{n,j}(x)\}$ and $\mathfrak{l}_n(x) := \hat{\mathfrak{f}}_{\mathfrak{n},\mathfrak{j}}(x) - \hat{\mathfrak{v}}_{\mathfrak{n},\mathfrak{j}}(x) = \max\{\underline{B}_j, \tilde{\mathfrak{l}}_{n,j}(x)\}$. Furthermore, we note $\mathfrak{u}_n - \mathfrak{l}_n = 2\,\hat{\mathfrak{v}}_{\mathfrak{n},\mathfrak{j}}(x)$. So, convergence of the enclosure $\mathcal{E}_{\mathfrak{l}_n}^{\mathfrak{u}_n}$ (i.e. convergence of $\mathfrak{u}_n - \mathfrak{l}_n$) is equivalent to convergence of $2\,\hat{\mathfrak{v}}_{\mathfrak{n},\mathfrak{j}}$. Finally, by definition, we see that $\mathfrak{u}_n^*(x) = \mathfrak{u}_{n,j}(x)$ and $\mathfrak{u}_n^*(x) = \mathfrak{l}_{n,j}(x)$ where $\mathfrak{u}_n^*, \mathfrak{l}_n^*$ match up the definitions of Def. C.2.6. Hence, Thm. C.2.8 is applicable. Therefore, Desideratum 1 -5 of Def. C.2.3 hold for the jth component of the prediction and target.

Note, the jth prediction uncertainty interval $\hat{H}_j(x)$ just coincides with $[\mathfrak{l}_n(x), \mathfrak{u}_n(x)]$ (cf. Eq. 4.2). Thus, By Desideratum 2 of Def. C.2.3 we conclude: $\forall \phi_j \in \mathcal{E}_{\mathfrak{l}_n}^{\mathfrak{u}_n} : \phi_j(x) \in \hat{H}_{n,j} = [\hat{\mathfrak{f}}_{\mathfrak{n},\mathfrak{j}}(x) - \hat{\mathfrak{v}}_{\mathfrak{n},\mathfrak{j}}(x), \hat{\mathfrak{f}}_{\mathfrak{n},\mathfrak{j}}(x) + \hat{\mathfrak{v}}_{\mathfrak{n},\mathfrak{j}}(x)]$. In particular, this means $f_j(x) \in \hat{H}_{n,j}$. Hence, Desideratum 1 of Def. 4.2.2 (conservatism) holds. The remaining desiderata as per Def. 4.2.7 follow from the validity of the corresponding desiderata as per Thm. C.2.8 on the enclosure for each component j.

By Desideratum 2 implies $\mathfrak{l}_n(x) \leq f_j(x) \leq \mathfrak{u}_n(x), \forall x$ which by our definition is equivalent to stating $f_j(x) \in [\mathfrak{l}_n(x), \mathfrak{u}_n(x)] = \hat{H}_j(x)$. Desiderata 3,4 of Def. C.2.3, establishes the desired properties of monotonocity and convergence, which extend to the corresponding properties to $\hat{H}_j$ and hence, to $\hat{\mathfrak{v}}_{\mathfrak{n},\mathfrak{j}}$. In turn, this establishes Desiderata 2-3 of Def. 4.2.2. Owing to the same link, $[\mathfrak{l}_n(x), \mathfrak{u}_n(x)] = \hat{H}_j(x)$, minimality of $\hat{H}_j(x)$ (i.e. of $\hat{\mathfrak{v}}_{\mathfrak{n},\mathfrak{j}}$) follows from the minimality of the enclosure (Desideratum 5 of Def. C.2.3). Thus, Desideratum 4 as per Def. 4.2.2 holds as well.

□

C.3. Pruning of uninformative sample points

The remaining derivations of this subsection show how the data can be pruned when new data points arrive the render old ones uninformative. They can be skipped by a reader not interested in the details.

We will focus our exposition on the ceiling functions. The derivations for the floor functions are entirely analogous (where one has to substitute $\leq$ by $\geq$ and $\min$ by $\max$).

Lemma C.3.1. *Let $\mathcal{N} := \{1, ..., N_n\}$, $q, i \in \mathcal{N}$ be the indices of a sample ceiling functions $\mathfrak{u}_n^q(\cdot; L_q)$ and $\mathfrak{u}_n^i(\cdot; L_i)$ where $L_q \leq L_i$ everywhere. We have*

$$\mathfrak{u}_n^q(s_i; L_q) \leq \mathfrak{u}_n^i(s_i; L_i) \Leftrightarrow \forall t \in I : \mathfrak{u}_n^q(t; L_q) \leq \mathfrak{u}_n^i(t; L_i).$$

Proof. $\Rightarrow$: $\mathfrak{u}_n^q(t; L_q) = \mathfrak{u}_n^q(s_q; L_q) + L_q\, \tilde{d}(t, s_q) \leq \mathfrak{u}_n^q(s_q; L_q) + L_q\, \tilde{d}(s_i, s_q) + L_q\, \tilde{d}(t, s_i) = \mathfrak{u}_n^q(s_i; L_q) + L_q\, \tilde{d}(t, s_i) \leq \mathfrak{u}_n^i(s_i; L_i) + L_q\, \tilde{d}(t, s_i) = \overline{f}(s_i) + L_q\, \tilde{d}(t, s_i) \leq \overline{f}(s_i) + L_i\, \tilde{d}(t, s_i) = \mathfrak{u}_n^i(t; L)$.
$\Leftarrow$: Trivial. □

The lemma implies that we can remove samples that are dominated by other samples' ceiling functions:

Theorem C.3.2. *Let $\mathcal{N} := \{1, ..., N_n\}$ and let $i \in \mathcal{N}$ be the index of sample ceiling function $\mathfrak{u}_n^i(\cdot; L_i) : t \mapsto \overline{f}(s_i) + L_i\, |t - s_i|$. If there exists a sample ceiling function $u^q(\cdot; L_q), q \in \mathcal{N}\backslash\{i\}$, $L_q \leq L_i$ everywhere, having a value below $\mathfrak{u}_n^i$ at the ith sample, then the ith sample plays no role in computing the optimal ceiling. That is:*
If there is $q \in \mathcal{N}\backslash\{i\}$ such that $\mathfrak{u}_n^q(s_i; L_q) \leq \mathfrak{u}_n^i(s_i; L_i)$, then we have:
$\mathfrak{u}^*(t; L_1, \ldots, L_n) = \min_{j \in \mathcal{N}} \mathfrak{u}^j(t; L_j) = \min_{j \in \mathcal{N}\backslash\{i\}} \mathfrak{u}^j(t; L_j)$.

Proof. The theorem is a direct consequence of Lem. C.3.1. □

Remark C.3.3. Normally, $L_i = L_q\, \forall q, j \in \mathcal{N}$ where each L_i is a function mapping t to a Hölder constant valid in some neighbourhood of t.

Theorem C.3.4. *Let $\mathcal{N} := \{1, ..., N_n\}$ and let $i \in \mathcal{N}$ be the index of sample ceiling function $\mathfrak{u}_n^i(\cdot; L_i) : t \mapsto \overline{f}(s_i) + L_i\, \tilde{d}(t, s_i)$. We have:*
if $\mathfrak{u}_n^q(s_i; L_q) \leq \mathfrak{u}_n^i(s_i; L_i)$ then

$$\mathfrak{u}^*(t; L_1, ..., L_n) \geq \min_{j \in \mathcal{N}} \mathfrak{u}^j(t; L_j) = \min\big\{ \min_{j \in \mathcal{N}\backslash\{i\}} \mathfrak{u}^j(t; L_j), \mathfrak{u}_n^q(s_i) + L_i\, \tilde{d}(t, s_i) \big\}.$$

Proof. Let $\mathfrak{u}_n^q(s_i; L_q) \leq \mathfrak{u}_n^i(s_i; L_i)$ and $L_q > L_i$ everywhere. Since $\mathfrak{u}_n^q(s_i) \leq \mathfrak{u}_n^i(s_i)$ and $L_i < L_q$ we have $\mathfrak{u}_n^i(t; L_i) = \overline{f}(s_i) + L_i\, \tilde{d}(t, s_i) = \mathfrak{u}_n^i(s_i; L_i) + L_i\, \tilde{d}(t, s_i) \geq \mathfrak{u}_n^q(s_i; L_i) + L_i\, \tilde{d}(t, s_i)$.

□

Remark C.3.5. The theorems allow us to assume, without loss of generality, that $\mathfrak{u}_n^i(s_i) = \min_j \mathfrak{u}_n^j(s_i)$ (as a preprocessing step, Thm. C.3.2 allows us to remove samples violating this assumption, without changing the optimal ceiling).

C.4. Special case: one-dimensional inputs, p=1

For functions on one-dimensional domains and exponent $p = 1$, matters simplify. We assume $\mathfrak{d}(x, y) := |x - y|$. This case has been considered in the context of optimisation [230] and quadrature [19] and the results therefore are not new. However, for our kinky inference method, the results rederived below have favourable practical implications (see Rem. C.4.7).

For ease of notation we will therefore often omit explicit mention of L in the statements of the ceiling and floor functions, unless it becomes necessary. Additionally, we assume, without loss of generality, that the sample inputs are ordered such that $s_1 \leq \ldots \leq s_n$. Let $b := \sup I$ and $a := \inf I$.

Under these assumptions, we next show that computing the optimal ceiling function only requires taking into account the up to two most adjacent samples :

Lemma C.4.1. *Let the assumptions of Rem. C.3.5 hold.*
We have:

1. $\forall i \in \mathcal{N}, t \geq s_i : \mathfrak{u}_n^*(t) = \min\{\mathfrak{u}_n^j(t) \,|\, j \in \mathcal{N}, j \geq i\}$,
2. $\forall i \in \mathcal{N}, t \leq s_i : \mathfrak{u}_n^*(t) = \min\{\mathfrak{u}_n^j(t) \,|\, j \in \mathcal{N}, j \leq i\}$.

Proof. 1.) Let $t \geq s_i$.
Showing $\leq$: This is trivial since $\mathfrak{u}_n^*(t) = \min_{j\in\mathcal{N}} \mathfrak{u}_n^j(t) \leq \min_{j\in\mathcal{N}, j\geq i} \mathfrak{u}_n^j(t)$.
Showing $\geq$: For contradiction, assume $\mathfrak{u}_n^*(t) < \min_{j\geq i} \mathfrak{u}_n^j(t)$. Hence, there is a $q \in \mathcal{N}, q < i$ such that

$$\forall t \geq s_i : \mathfrak{u}_n^q(t) < \min\{\mathfrak{u}_n^j(t) \,|\, j \in \mathcal{N}, j \geq i\}. \tag{C.2}$$

Choose such a $q \in \mathcal{N}, q < i$. By assumption, we have $s_q < s_i$. Hence, Condition C.2 implies in particular $\mathfrak{u}_n^q(t) = \overline{f}(s_q) + L(t - s_q) < \mathfrak{u}_n^i(t) = \overline{f}(s_i) + L(t - s_i)$. We have $\overline{f}(s_q) + L(t - s_q) = \overline{f}(s_q) + L(s_i - s_q) + L(t - s_i) = \mathfrak{u}_n^q(s_i) + L(t - s_i) \overset{!}{<} \overline{f}(s_i) + L(t - s_i) = \mathfrak{u}_n^i(s_i) + L(t - s_i)$ which holds, if and only if $\mathfrak{u}_n^q(s_i) < \mathfrak{u}_n^i(s_i)$. However, the last inequality contradicts our assumption of $\min_j \mathfrak{u}_n^j(s_i) = \mathfrak{u}_n^i(s_i)\, \forall i$ (Rem. C.3.5).

2.) The proof is analogous to 1.):
Let $t \leq s_i$.
Showing $\leq$: This is trivial since $\mathfrak{u}_n^*(t) = \min_{j\in\mathcal{N}} \mathfrak{u}_n^j(t) \leq \min_{j\in\mathcal{N}, j\leq i} \mathfrak{u}_n^j(t)$.
Showing $\geq$: For contradiction, assume $\mathfrak{u}_n^*(t) < \min_{j\leq i} \mathfrak{u}_n^j(t)$. Hence, there is a $q \in \mathcal{N}, q > i$ such that

$$\forall t \leq s_i : \mathfrak{u}_n^q(t) < \min\{\mathfrak{u}_n^j(t) \,|\, j \in \mathcal{N}, j \leq i\}. \tag{C.3}$$

Choose such a $q \in \mathcal{N}, q > i$ (thus, $s_q > s_i$).
The way we ordered the samples, $q > i$ implies $s_q > s_i$. Thus, Condition C.3 implies in particular: $\mathfrak{u}_n^q(t) = \overline{f}(s_q) + L(s_q - t) < \mathfrak{u}_n^i(t) = \overline{f}(s_i) + L(s_i - t) = \mathfrak{u}_n^i(s_i) + L(s_i - t)$. We have $\overline{f}(s_q) + L(s_q - t) = \overline{f}(s_q) + L(s_q - s_i) + L(s_i - t) = \mathfrak{u}_n^q(s_i) + L(s_i - t) \overset{!}{<} \mathfrak{u}_n^i(s_i) + L(s_i - t)$. However, the last inequality contradicts our assumption of $\min_j \mathfrak{u}_n^j(s_i) = \mathfrak{u}_n^i(s_i)\, \forall i$ (made in Rem. C.3.5).

□

The Lemma allows us to tremendously simplify computation of the optimal ceiling– it tells us that all samples, except the up to two neighbouring ones, can be discarded when computing the optimal ceiling at a given input:

Theorem C.4.2. *Let the assumptions of Rem. C.3.5 hold. Let $a := s_0 := \inf I, s_{n+1} := b := \sup I$ and for $i = 0, \ldots, N_n$ define $I_i := [s_i, s_{i+1}]$.*
We have:

$$\forall i, t \in I_i : \mathfrak{u}_n^*(t) = \begin{cases} \mathfrak{u}_n^{i+1}(t) & , i = 0 \\ \min\{\mathfrak{u}_n^i(t), \mathfrak{u}_n^{i+1}(t)\} & , i \in \{1, \ldots, N_n - 1\} \\ \mathfrak{u}_n^i(t) & , i = N. \end{cases}$$

The analogous statement for the Hölder floor function can be derived completely analogously and will therefore be given at this point without proof:

Theorem C.4.3. *Assume the assumptions of Rem. C.3.5 hold. Let $a := s_0 := \inf I, s_{n+1} := b := \sup I$ and for $i = 0, \ldots, N_n$ define $I_i := [s_i, s_{i+1}]$.*
We have:

$$\forall i, t \in I_i : \mathfrak{l}_n^*(t) = \begin{cases} \mathfrak{l}_n^{i+1}(t) & , i = 0 \\ \max\{\mathfrak{l}_n^i(t), \mathfrak{l}_n^{i+1}(t)\} & , i \in \{1, \ldots, N_n - 1\} \\ \mathfrak{l}_n^i(t) & , i = N. \end{cases}$$

In addition to saving computation by discarding irrelevant samples, the the theorems will play an important role when computing a closed-form the integral of the optimal ceiling and floor functions.

In preparation for the part on quadrature it will prove useful to determine the extrema of the enclosing functions on each subinterval I_i.

Lemma C.4.4. *We have:*

1. *If* $\mathfrak{u}^i(s_i) \leq \mathfrak{u}^{i+1}(s_i)$ *and* $\mathfrak{u}^{i+1}(s_{i+1}) \leq \mathfrak{u}^i(s_{i+1})$ *then we have* $\left|\overline{f}(s_{i+1}) - \overline{f}(s_i)\right| \leq L\,|s_{i+1} - s_i|$.
2. *If* $\mathfrak{l}^i(s_i) \geq \mathfrak{l}^{i+1}(s_i)$ *and* $\mathfrak{l}^{i+1}(s_{i+1}) \geq \mathfrak{l}^i(s_{i+1})$ *then we have* $\left|\underline{f}(s_{i+1}) - \underline{f}(s_i)\right| \leq L\,|s_{i+1} - s_i|$.

Proof. 1.) On the one hand, we have: $\mathfrak{u}^i(s_i) \leq \mathfrak{u}^{i+1}(s_i) \Leftrightarrow \mathfrak{u}^i(s_i) - \mathfrak{u}^{i+1}(s_i) \leq 0 \Leftrightarrow \overline{f}(s_i) + L\,|s_i - s_i| - \overline{f}(s_{i+1}) - L\,|s_i - s_{i+1}| \leq 0 \Leftrightarrow \overline{f}(s_i) - \overline{f}(s_{i+1}) - L\,|s_i - s_{i+1}| \leq 0 \Leftrightarrow \overline{f}(s_i) - \overline{f}(s_{i+1}) \leq L\,|s_i - s_{i+1}|$
$\Leftrightarrow \overline{f}(s_{i+1}) - \overline{f}(s_i) \geq -L\,|s_{i+1} - s_i|$.
On the other hand: $\mathfrak{u}^{i+1}(s_{i+1}) \leq \mathfrak{u}^i(s_{i+1}) \Leftrightarrow \mathfrak{u}^{i+1}(s_{i+1}) - \mathfrak{u}^i(s_{i+1}) \leq 0 \Leftrightarrow \overline{f}(s_{i+1}) - \overline{f}(s_i) - L\,|s_{i+1} - s_i| \leq 0 \Leftrightarrow \overline{f}(s_{i+1}) - \overline{f}(s_i) \leq L\,|s_{i+1} - s_i|$.

2.) The proof is completely analogous to 1.). □

Lemma C.4.5. *Let* $\mathfrak{d}(x, y) := |x - y|$. *For* $i \in \{1, \ldots, N_n - 1\}$ *let* $L_i > 0$, $I_i = [s_i, s_{i+1}]$. *Define* $\xi_i^{\mathfrak{u}} := \frac{s_i + s_{i+1}}{2} + \frac{\overline{f}(s_{i+1}) - \overline{f}(s_i)}{2L_i}$.
For $t \in I_i$ *we have*

1. $\xi_i^{\mathfrak{u}} \in I_i$ *and* $\mathfrak{u}_n^*(t) = \begin{cases} \mathfrak{u}_n^i(t), & t \leq \xi_i^{\mathfrak{u}} \\ \mathfrak{u}_n^{i+1}(t), & t \geq \xi_i^{\mathfrak{u}}. \end{cases}$
2. $\xi_i^{\mathfrak{u}} \in \arg\max_{\tau \in I_i} \mathfrak{u}_n^*(\tau)$.
3. $y_i^{\mathfrak{u}} := \mathfrak{u}_n^*(\xi_i^{\mathfrak{u}}) = \frac{\overline{f}(s_{i+1}) + \overline{f}(s_i)}{2} + L_i \frac{s_{i+1} - s_i}{2}$.

Proof. Let $1 \leq i \leq N - 1, t \in I_i$. Define $\alpha_i^{\mathfrak{u}}(t) = \overline{f}(a_i) + L_i(t - a_i) = \mathfrak{u}_n^i(t), \beta_i^{\mathfrak{u}}(t) = \overline{f}(b_i) + L_i(b_i - t) = \mathfrak{u}_n^{i+1}(t), \forall t \in I_i$.

Firstly, $\xi_i^{\mathfrak{u}} \in I_i$ is a direct consequence of Lem. C.4.4.

Secondly, we show $\mathfrak{u}_n^*(t) = \begin{cases} \mathfrak{u}_n^i(t), & t \leq \xi_i^{\mathfrak{u}} \\ \mathfrak{u}_n^{i+1}(t), & t \geq \xi_i^{\mathfrak{u}}. \end{cases}$:

By definition, $\mathfrak{u}_n^*(t) = \min\{\mathfrak{u}_n^i(t), \mathfrak{u}_n^{i+1}(t)\} = \min\{\alpha_n^i(t), \beta_n^i(t)\}$. We have $\dot{\alpha}_i^{\mathfrak{u}}(t) = L_i > 0$ and $\dot{\beta}_i^{\mathfrak{u}}(t) = -L_i < 0$. Hence, the functions are strictly monotonous. Let ξ be the input where the lines intersect, i.e. $\alpha_i^{\mathfrak{u}}(\xi) = \beta_i^{\mathfrak{u}}(\xi)$. Monotonicity implies $\forall t \leq \xi : \alpha_i^{\mathfrak{u}}(t) \leq \alpha_i^{\mathfrak{u}}(\xi) \wedge \beta_i^{\mathfrak{u}}(t) \geq \beta_i^{\mathfrak{u}}(\xi) = \alpha_i^{\mathfrak{u}}(\xi)$. Hence,

$$\forall t \leq \xi : \mathfrak{u}_n^*(t) = \min\{\alpha_i^{\mathfrak{u}}(t), \beta_i^{\mathfrak{u}}(t)\} = \alpha_i^{\mathfrak{u}}(t).$$

Analogously, monotonicity implies $\forall t \geq \xi : \beta_i^{\mathfrak{u}}(t) \leq \beta_i^{\mathfrak{u}}(\xi) \wedge \alpha_i^{\mathfrak{u}}(t) \geq \alpha_i^{\mathfrak{u}}(\xi) = \beta_i^{\mathfrak{u}}(\xi)$. Thus,

$$\forall t \geq \xi : \mathfrak{u}_n^*(t) = \min\{\alpha_i^{\mathfrak{u}}(t), \beta_i^{\mathfrak{u}}(t)\} = \beta_i^{\mathfrak{u}}(t).$$

In conjunction, we have $\mathfrak{u}_n^*(t) = \begin{cases} \alpha_i^{\mathfrak{u}}(t), & t \leq \xi \\ \beta_i^{\mathfrak{u}}(t), & t \geq \xi \end{cases}$. Again, using monotonicity of $\alpha_i^{\mathfrak{u}}$ and $\beta_i^{\mathfrak{u}}$ we infer that $\mathfrak{u}_n^*$ has a single maximum $y := \mathfrak{u}_n^*(\xi)$ at ξ.

Next, we show that $\xi = \xi_i^{\mathfrak{u}}$:
$\alpha_i^{\mathfrak{u}}(\xi) = \beta_i^{\mathfrak{u}}(\xi)$
$\Leftrightarrow \overline{f}(s_i) + L_i(\xi - s_i) = \overline{f}(s_{i+1}) - L_i(\xi - s_{i+1})$
$\Leftrightarrow L_i\big(2\xi - s_{i+1} - s_i\big) = \overline{f}(s_{i+1}) - \overline{f}(s_i)$
$\overset{L_i > 0}{\Leftrightarrow} \big(2\xi - s_{i+1} - s_i\big) = \frac{\overline{f}(s_{i+1}) - \overline{f}(s_i)}{L_i}$
$\Leftrightarrow \xi = \frac{\overline{f}(s_{i+1}) - \overline{f}(s_i)}{2L_i} + \frac{s_{i+1} + s_i}{2} = \xi_i^{\mathfrak{u}}$.

Finally, we need to evaluate $\mathfrak{u}_n^*(\xi_i^{\mathfrak{u}})$ to prove the last claim: We have
$\mathfrak{u}_n^*(\xi) = \alpha_i^{\mathfrak{u}}(\xi) = \overline{f}(s_i) + L_i(\xi - s_i)$
$= \overline{f}(s_i) + L_i\big(\frac{\overline{f}(s_{i+1}) - \overline{f}(s_i)}{2L_i} + \frac{s_{i+1}+s_i}{2} - s_i\big)$
$= \frac{\overline{f}(s_{i+1}) - \overline{f}(s_i) + 2\overline{f}(s_i)}{2} + L_i\frac{s_{i+1}-s_i}{2} = y_i^{\mathfrak{u}}$.

□

Completely analogously to the the proof of Lem. C.4.5 we can prove the corresponding statement for the Hölder floor:

Lemma C.4.6. *Let* $\mathfrak{d}(x,y) := |x-y|$. *For* $i \in \{1,\dots,N_n-1\}$ *let* $L_i > 0$, $I_i = [s_i, s_{i+1}]$. *Define* $\xi_i^{\mathfrak{l}} := \frac{s_i+s_{i+1}}{2} - \frac{\underline{f}(s_{i+1}) - \underline{f}(s_i)}{2L_i}$.
For $t \in I_i$ *we have*

1. $\xi_i^{\mathfrak{l}} \in I_i$ *and* $\mathfrak{l}_n^*(t) = \begin{cases} \mathfrak{l}_n^i(t), & t \le \xi_i^{\mathfrak{l}} \\ \mathfrak{l}_n^{i+1}(t), & t \ge \xi_i^{\mathfrak{l}}. \end{cases}$
2. $\xi_i^{\mathfrak{l}} \in \arg\min_{\tau \in I_i} \mathfrak{l}_n^*(\tau)$.
3. $y_i^{\mathfrak{l}} := \mathfrak{l}_n^*(\xi_i^{\mathfrak{l}}) = \frac{\underline{f}(s_{i+1}) + \underline{f}(s_i)}{2} - L_i\frac{s_{i+1}-s_i}{2}$.

Proof. The proof is analogous to the one provided for Lem. C.4.5.

□

Remark C.4.7 (Reduction of computational effort for KI and collision detection). While the results above are not new, their statement has practical benefits in terms of computational effort of the kinky inference rule in one-dimensions when the canonical metric $\mathfrak{d}(x,x') = |x-x'|$ is employed and $p=1$. Then Thm. C.4.2 and Thm. C.4.3 tell us that the ceiling and floor functions can be evaluated by only considering the two most adjacent sample ceiling and floor functions. If we maintain a search tree over the grid, we can search for those two entries in $\mathcal{O}(\log N_n)$. This reduces the computational effort for evaluating the prediction functions $\hat{\mathfrak{f}}_{\mathfrak{n}}$, $\hat{\mathfrak{v}}_{\mathfrak{n}}$ (cf. Def. 4.2.4) from linear to logarithmic asymptotic complexity. This can make a significant difference in terms of computational speed of the inference in the presence of large data sets.

As a second benefit of our derivations, Lem. C.4.6 and Lem. C.4.5 can be utilised in the collision detection method as presented in Sec. 5.2.1 where negative function values need to be proven or ruled out. The Lemmata tell us which finite number of points to check.

D. Notation and abbreviations

"In symbols one observes an advantage in discovery which is greatest when they express the exact nature of a thing briefly and, as it were, picture it; then indeed the labor of thought is wonderfully diminished."

Gottfried von Leibniz (1646 - 1716)

We list notation and abbreviations frequently used throughout the thesis. Some of the notation is context dependent. In what follows, X is a random variable or random vector, M a matrix, v a vector and T an operator.

Abbreviation	**Meaning**
DE	"Differential" or "difference" equation.
*R*DE, *S*DE, *U*DE, *O*DE	"*Random*", "*stochastic*", "*uncertain*", "ordinary" DE.
r.v.	"Random variable".
r.vec.	"Random vector".
s.p.	"Stochastic process".
r.f.	"Random field".
GP	"Gaussian process".
GRF	"Gaussian random field".
o.i.	"Orthogonal increments".
m.s.	"Mean square".
a.s.	"Almost sure".
w.l.o.g.	"Withouth loss of generality".
q.e.d.	"Quod erat demonstrandum". Alternative to the box marker for indicating the end of a proof.
pos.def., PD	Positive definite.
NND, PSD	Nonnegative definite.
kNN	"k-nearest neighbour".
IVP	"Initial value problem".

Table D.1.: Common abbreviations.

Symbol / Notation	Meaning
$\mathcal{GP}(m,k)$	GRF with mean function m and covariance function /kernel k.
$\log \mathcal{GP}(m,k)$	Log-normal r.f. with mean function m and covariance function /kernel k.
$\mathcal{N}(m,k)$	Gaussian distribution with mean vector m and covariance matrix C.
$\log \mathcal{N}(m,k)$	Log-normal distribution.
$\Pr$	Probability.
Ω	Sample or outcome space.
$\mathcal{B}, \Sigma$	A *sigma-algebra* (also known as *Borel-field*).
Π	Often used to denote a stochastic process.
$\mu, \mathbb{P}$	A measure and a probability measure.
$\mathrm{d}\mu$	Differential with respect to measure μ.
dt , $\mathrm{d}x$	Normally differential with respect to Riemann or Lebesgue measure.
$\int_{\mathcal{I}} f(t)\, dt$	Lebesgue / Riemann integral.
$\mathrm{d}W$	Wiener increment.
$\int_{\mathcal{I}} f(t)\, \mathrm{d}W$	Ito integral.
ω	Outcome.
$\mathcal{F}_t$	Filtration set for index ("time") t.
$X_t, X(t)$	Random variable/vector/function indexed by t
d	Dimension (natural number). Eg $v \in \mathbb{R}^d$.
$\mathcal{I}$	Index set. Typically, $\mathcal{I} \subset \mathbb{N}, \mathcal{I} \subset \mathbb{R}_+$ or $\mathcal{I} \subset \mathbb{R}^d$.
$\mathcal{L}_2$, L_2	Space of square-integrable r.v., functions.
$\langle X \rangle$	Expected value of X.
$\langle X \vert E \rangle$	Expected value of X conditioned on event E.
$\mathrm{var}[X]$	Variance of random variable X.
$\mathrm{cov}(X,Y)$	Covariance between random variables X and Y.
$k(\cdot,\cdot)$	Kernel function / covariance function.
$\mathrm{Var}[X]$	Variance (-covariance) matrix of random vector X (in "Feller notation").
$\mathrm{Cov}(X,Y)$	Cross-covariance matrix between random vectors X and Y.
I_d	Identity matrix, contained in $\mathbb{R}^{d\times d}$.
O_d	The zeror matrix, contained in $\mathbb{R}^{d\times d}$.
o_m	The zero vector in $\mathbb{R}^m$.
$\langle \cdot, \cdot \rangle$	Scalar product.
$\|\cdot\|$	Norm.
$\|v\|_2$	Euclidean Norm of vector v.
$\|v\|_\infty$	Maximum / supremum norm of vector v.
$\|\|\|M\|\|\|$	Matrix norm.
$\|\|\|M\|\|\|_2$	Spectral norm.
$\rho(M)$	Spectral radius.
$\|\|\|T\|\|\|$	Operator norm.
$T > 0$	T is strictly positive definite.
$T \geq 0$	T is nonnegative definite.
$T \geq \tilde{T}$	$T - \tilde{T}$ is nonnegative definite.
$T^*, T^\top$	Adjoint / transpose of operator T.
Δ	Time increment (duration between two time steps).
t	A (time) index.
$\mathcal{X}$	A state or input space.
$\mathfrak{d}(\cdot,\cdot)$, $\mathfrak{d}_{\mathcal{X}}(\cdot,\cdot)$	A metric or pseudo-metric on input space $\mathcal{X}$.
$\mathcal{U}$	A control input space.
$u / u(\cdot)$	A control input / law.
$\mathfrak{A}$	Set of agents.
$\mathfrak{a}, \mathfrak{r}, \mathfrak{q}$	Three particular agent indices.
$x^{\mathfrak{a}}(t)$	State trajectory of agent $\mathfrak{a}$ (possibly stochastic).
$\mathfrak{B}_\epsilon(x)$	Euclidean ϵ-ball centred at x.
$\mathcal{D}, \mathcal{D}_n$	A data (sample) set, the nth data set with .
N_n	$\vert\mathcal{D}_n\vert$ denoting the number of elements in $\mathcal{D}_n$.

Table D.2.: Notation and symbols commonly used throughout the thesis.