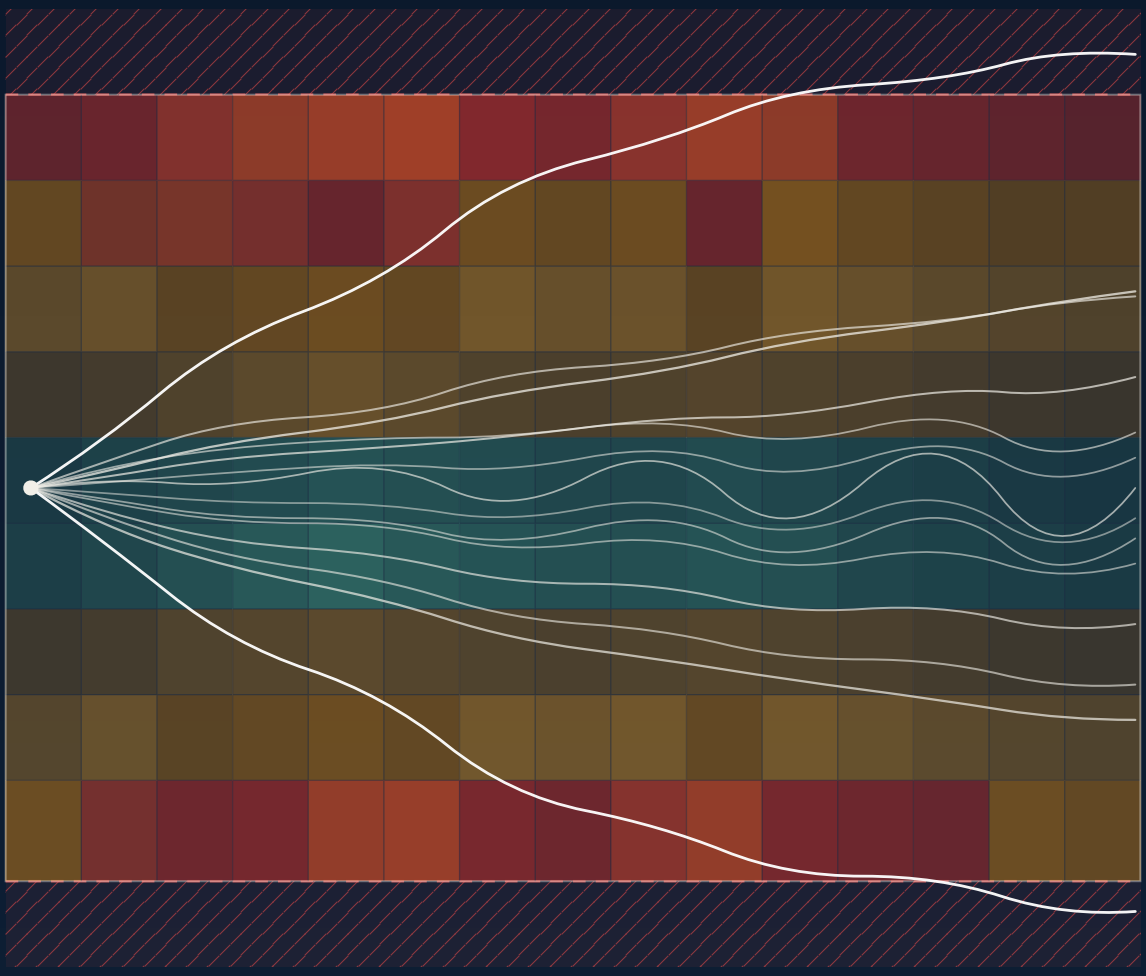
Safety verification of discrete-time stochastic systems



Frederik Baymler Mathiesen

TU Delft

SAFETY VERIFICATION OF DISCRETE-TIME STOCHASTIC SYSTEMS

Safety verification of discrete-time stochastic systems

Dissertation

for the purpose of obtaining the degree of doctor
at Delft University of Technology,
by the authority of the Rector Magnificus Prof. dr. ir. H. Bijl;
Chair of the Board of Doctorates,
to be defended publicly on
Thursday 25 June 2026 at 10:00

by

Frederik Baymler Mathiesen

This dissertation has been approved by the promotors.

Composition of the doctoral committee:

Rector Magnificus,	chairperson
Dr. M. Mazo Espinosa,	Delft University of Technology, promotor
Dr. ir. S.C. Calvert,	Delft University of Technology, promotor
Dr. L. Laurenti,	Delft University of Technology, copromotor
Independent members:	
Prof. dr. M.T.J. Spaan,	Delft University of Technology
Prof. dr. N. Jansen,	Ruhr-University Bochum
Prof. dr. R. Jungers,	Université catholique de Louvain
Dr. N. Paoletti,	Kings College London
Prof. dr. ir. J.W. van Wingerden,	Delft University of Technology, reserve member

This dissertation has been completed in fulfillment of the requirements of the Dutch Institute of Systems and Control (DISC) for graduate studies and the TU Delft Graduate School for the Doctoral Education Program. The research was supported by the TU Delft AI Labs programme.

Printed by:	Ridderprint
Front:	Conceptual safety field of a discrete-time stochastic system.
Back:	Parallel execution grid and contributor graph.

An electronic version of this dissertation is available at
http://repository.tudelft.nl/.

This dissertation is written with Oxford spelling.

A very large part of space-time must be investigated,
if reliable results are to be obtained.

Alan Turing

Contents

Summary

Cyber-physical systems integrate digital control and physical processes and often operate in complex, uncertain environments. The consequence of operational failures in such systems can be catastrophic and may include loss of life, environmental damage, irreparable system damage, and economic disruption. Therefore, safety is a critical concern and traditional engineering approaches that only rely on extensive testing and conservative design margins are insufficient to guarantee safety in the face of uncertainty. The issue is that testing only assesses the performance on a limited number of scenarios or samples, while in practice the collection of possible scenarios is uncountable due to the physical process and uncertainties therein. Formal methods provide a powerful alternative, offering mathematically rigorous verification that accounts for all possible behaviours subject to ranges of disturbances and uncertainties. A key challenge in applying formal methods to stochastic dynamical systems is a fundamental tension between computational tractability and conservatism. Existing approaches either scale poorly with the system dimension and complexity or produce loose bounds on safety probabilities that limit their practical applicability. This creates a critical gap between theory and practice: while formal verification methods exist, their practical applicability to real-world systems remains severely limited. Therefore, the core research question driving this work is: *How to efficiently compute tight bounds on the satisfaction probabilities for safety, reachability, and reach-avoid specifications of stochastic systems?*

To answer this question and address the gap, the present dissertation develops several complementary approaches for efficiently verifying properties of stochastic systems. The focus is on discrete-time, continuous-space stochastic systems and simple specifications over given sets. The approaches represent points along the spectrum of the scalability-conservatism trade-off and rely on different system assumptions. The methods developed belong to two families: stochastic barrier functions and finite-state abstractions.

Stochastic barrier functions are Lyapunov-like functions that provide certificates of safety by imposing conditions on the expected value of the barrier function along system trajectories. The core idea is that if the composition of the barrier function with the dynamics of the system forms a c-martingale, then the probability of safety can be bounded using martingale inequalities. The main challenge is to construct a barrier that is optimal with respect to the martingale inequalities. Handcrafting such functions is difficult and time-consuming, and existing synthesis methods are often limited to low-dimensional and simple systems for non-trivial results. To enable efficient synthesis of stochastic barrier functions, we develop multiple synthesis techniques, including a neural network-based method that offers flexibility but requires post hoc verification to confirm correctness. More significantly, we introduce piecewise-constant stochastic barrier function theory and synthesis

methods that are guaranteed to asymptotically approach optimality. The synthesis methods include a dual linear programming formulation, a counterexample-guided inductive synthesis with linear programming solvers, and gradient descent optimization; their trade-off is between scalability and required parameter tuning. The theoretical analysis of piece-wise constant barriers reveals deep insights into the relationship between barrier functions and system dynamics, illuminates fundamental sources of conservatism inherent to the approach, and establishes clear connections to Interval Markov Decision Process (IMDP)-based finite-state abstractions. Additionally, we develop a data-driven scenario-theoretic approach for systems with partially unknown dynamics, leveraging scenario theory to handle uncertainty in system models.

Finite-state abstractions, on the other hand, reduce the original continuous-space system to a finite-state model that can be analysed using probabilistic model checking techniques. While abstraction-based methods are exceedingly flexible – they have been successfully applied to a wide range of systems, including partially-unknown systems, and specifications – they often suffer from high computational complexity of using probabilistic model checking and scalability issues due to the pervasive state-space explosion problem. To address the first issue, we develop hardware-aware algorithmic innovations for faster model checking of IMDPs via dynamic programming. Dynamic programming over IMDPs relies for efficiency on an algorithm named O-maximization, or order-maximization, which by theoretical analysis is revealed to be composed of two phases: a sorting phase and a cumulative summation phase. We introduce parallel algorithms to both phases, which allows us to exploit modern highly parallel computing architectures to achieve significant speedups in verifying IMDPs. To address the second issue, we introduce a novel finite-state model called factored Interval Markov Decision Processes (fIMDPs) that exploits structural properties of the system dynamics to significantly reduce memory requirements while maintaining formal guarantees. Factored models encode data-dependencies more fine-grained than flat models, which is the key driver for the reduction in memory. Moreover, factored models have been successfully used in the context of abstraction to Markov Decision Processes (MDPs), but have not been applied until now to IMDPs. An insight of abstracting to factored models is that the structural exploitation inadvertently tightens the ambiguity sets that characterize IMDPs conservatism, thereby reducing the pessimism of the bounds.

The methods developed in this dissertation represent significant algorithmic and theoretical advances to address the scalability-conservatism trade-off and enable more efficient computation of tighter safety probability bounds, advancing formal verification of stochastic dynamical systems. By advancing stochastic barrier function synthesis and IMDP-based finite-state abstractions, this work pushes the frontier of formal verification for stochastic systems, providing new tools and insights to bridge the gap between theory and practice. The findings suggest that future progress in scalable safety verification for stochastic systems depends critically on designing algorithms that respect and exploit inherent problem structure, offering a promising, albeit challenging path toward making formal methods practical for real-world stochastic cyber-physical systems.

Samenvatting

Cyber-physical systemen integreren digitale besturing met fysieke processen en opereren vaak in complexe, onzekere omgevingen. De gevolgen van operationele storingen in deze systemen kunnen rampzalig zijn zoals verlies van mensenlevens, milieuschade, onherstelbare systeemschade en economische verstoring omvatten. Daarom is veiligheid een grote zorg en traditionele technische benaderingen die uitsluitend steunen op uitgebreid testen en conservatieve ontwerpmarges zijn ontoereikend om de veiligheid te garanderen bij onzekerheid. Het probleem is dat traditionele testen slechts het resultaat beoordeelt op een beperkt aantal scenario's of steekproeven, terwijl in de praktijk de verzameling van mogelijke scenario's ontelbaar is vanwege het fysieke proces en de onzekerheden daarin. Formele methoden bieden een krachtig alternatief en leveren wiskundig rigoureuze verificatie die rekening houdt met al mogelijk gedrag onder reeksen van verstoringen en onzekerheden. Een belangrijke uitdaging bij het toepassen van formele methoden op stochastic dynamical systemen is een fundamentele spanning tussen computational tractability en conservatism. Bestaande benaderingen schalen ofwel slecht met de systeemdimensie en -complexiteit, ofwel produceren ruime grenzen op veiligheidswaarschijnlijkheden die hun praktische toepasbaarheid beperken. Dit creëert een grote kloof tussen theorie en praktijk: hoewel formele verificatiemethoden bestaan, blijft hun praktische toepasbaarheid op reële systemen ernstig beperkt. Daarom is de onderzoeksvraag die dit werk aandrijft: *Hoe kunnen we strakke grenzen efficiënt berekenen op de statisfaction probability's voor safety-, reachability- en reach-avoid-specificaties van stochastic systemen?*

Om deze vraag te beantwoorden, ontwikkelt dit proefschrift meerdere complementaire benaderingen voor het efficiënt verifiëren van eigenschappen van stochastic systemen. De focus ligt op discrete-time, continuous-state stochastic systemen en eenvoudige specificaties over gegeven verzamelingen. De benaderingen vertegenwoordigen punten langs het spectrum van de schaalbaarheid-conservatism-afweging en berusten op verschillende systeemaannames. De ontwikkelde methoden behoren tot twee families: stochastic barrier functies en finite-state-abstractions

stochastic barrier functies zijn Ljapunov-achtige functies die veiligheidscertificaten bieden door voorwaarden op te leggen aan de verwachte waarde van de barrier functie langs systeemtrajecten. Het idee is dat als de samenstelling van de barrier functie met de dynamica van het systeem samen een c-martingale vormen, dan kan de veiligheidswaarschijnlijkheid begrensd worden met behulp van martingaalongelijkheden. De belangrijkste uitdaging is het construeren van een barrier die optimaal is ten opzichte van de martingaalongelijkheden. Het handmatig ontwerpen van dergelijke functies is moeilijk en tijdrovend. Bestaande berekeningsmethoden zijn vaak beperkt tot laagdimensionale en eenvoudige systemen voor

niet-triviale resultaten. Om efficiënte berekeningen van stochastic barrier functies mogelijk te maken, ontwikkelen we meerdere technieken, waaronder een op neural network gebaseerde methode die flexibiliteit biedt maar post hoc verificatie vereist om correctheid te bevestigen. Bovendien introduceren we piece-wise constant stochastic barrier functietheorie en berekeningsmethoden die gegarandeerd asympototically benaderen optimaliteit. De berekeningsmethoden omvatten een dual linear programmingsformulering, een counter-example guided inductive synthesis met linear programming oplossers en gradient descent optimalisatie; hun afweging ligt tussen schaalbaarheid en vereiste parameterafstemming. De theoretische analyse van piece-wise constant barriers onthult diepe inzichten in de relatie tussen barrier functies en systeemdynamica, belicht fundamentele bronnen van conservatism, en legt duidelijke verbanden met Interval Markov Decision Process (IMDP)-gebaseerde finite-state abstractions. Daarnaast ontwikkelen we een datagedreven scenariotheorie-gebaseerde benadering voor systemen met gedeeltelijk onbekende dynamica, waarbij scenariotheorie wordt ingezet om onzekerheid in systeemmodellen te behandelen.

Finite-state abstractions daarentegen reduceren het oorspronkelijke continuous-state systeem tot een finite-state model dat geanalyseerd kan worden met probabilistic model checking technieken. Hoewel abstraction-gebaseerde methoden buitengewoon flexibel zijn – ze zijn met succes toegepast op een breed scala aan systemen, waaronder gedeeltelijk onbekende systemen, en specificaties – lijden ze vaak onder de hoge computational complexity van probabilistic model checking en schaalbaarheidsproblemen door het state-space explosion probleem. Om het eerste probleem aan te pakken, ontwikkelen we hardware-bewuste algoritmische innovaties voor snellere model checking van IMDPs via dynamic programming. Dynamic programming over IMDPs is afhankelijk van efficiëntie op een algoritme genaamd O-maximalisatie, dat door theoretische analyse blijkt te bestaan uit twee fasen: een sorteerfase en een cumulative summation fase. We introduceren parallelle algoritmen voor beide fasen, waardoor we moderne sterk parallelle computerarchitecturen kunnen benutten om aanzienlijke versnellingen te bereiken bij het verifiëren van IMDPs. Om het tweede probleem aan te pakken, introduceren we een nieuw finite-state model genaamd factored Interval Markov Decision Processes (fIMDPs) dat structurele eigenschappen van de systeemdynamica benut om geheugenvereisten te verminderen met behoud van formele garanties. Gefactoriseerde modellen coderen data-afhankelijkheden gedetailleerder dan vlakke modellen, wat de belangrijkste drijfveer is voor de geheugenreductie. Bovendien zijn gefactoriseerde modellen met succes gebruikt in de context van abstraction naar Markov Decision Processes (MDPs), maar zijn tot nu toe niet toegepast op IMDPs. Een inzicht van het abstraction naar gefactoriseerde modellen is dat de structurele benutting onbedoeld de ambiguity set aanscherpt die het conservatism van IMDPs karakteriseren, waardoor het conservatism van de grenzen wordt verminderd.

De methoden die in dit proefschrift zijn ontwikkeld, vertegenwoordigen significante algoritmische en theoretische vooruitgangen om de schaalbaarheid-conservatism-afweging aan te pakken en efficiëntere berekening van strakkere veiligheids-

waarschijnlijkheidsgrenzen mogelijk te maken, waarmee de formele verificatie van stochastic dynamical systemen wordt bevorderd. Door de berekeningen van stochastic barrier functies en IMDP-gebaseerde finite-state abstractions te bevorderen, verlegt dit werk de grenzen van formele verificatie voor stochastic systemen en biedt het nieuwe instrumenten en inzichten om het gat tussen theorie en praktijk te overbruggen. De bevindingen suggereren dat toekomstige vooruitgang in schaalbare veiligheidsverificatie voor stochastic systemen afhangt van het ontwerpen van algoritmen die de probleemstructuur respecteren en benutten. Dit biedt een veelbelovend en uitdagend pad naar het praktisch maken van formele methoden voor reële stochastic cyber-physical systemen.

1

Introduction

In safety-critical systems, the cost of being wrong is high,
and the benefit of being right is that nothing happens.

John Rushby

1

1.1. Motivation

Cyber-Physical Systems (CPSs) are autonomous systems that integrate computation and physical processes [1]. CPSs are becoming more ubiquitous with examples including infrastructure (e.g., electrical grid [2] and nuclear reactor control [3]), autonomous and robotic devices (e.g., medical devices [4] and human-robot interaction [5]), and transportation systems (e.g., avionics [6], air traffic control [7], railway systems [8], and automated vehicles [9]). Importantly, many CPSs operate in uncertain environments and are subject to stochastic disturbances and unpredictable interactions with humans and other systems [10].

The autonomous nature and the critical roles that many CPSs fulfil implies that they are often safety-critical systems. That is, systems where any failure or malfunction can have catastrophic consequences, including significant economic, environmental, and social impact, severe injuries, and/or fatalities [11]. Due to the nature of the consequences, regulatory bodies often mandate rigorous verification and validation processes to ensure system safety and reliability [12]. One family of methods for providing such assurances is Formal Methods, where the idea is to use rigorous mathematical frameworks to model systems, specify requirements, and (formally) verify satisfaction [13], [14]. Unlike simulation-based testing, which can only explore a (small) finite set of scenarios, formal verification aims to prove properties over all possible states and disturbances. For stochastic systems, this translates to probabilistic safety guarantees, e.g., ensuring that the probability of entering unsafe states remains below a specified threshold.

The general idea of formal methods is to reason about all possible system behaviours mathematically such as via enumeration of all states or trajectories, by constructing mathematical objects that capture system behaviour (e.g., Lyapunov functions or forward-invariant sets), or by abstracting the system to a simpler representation that is more amenable to analysis with the previous two methods. For studying safety of stochastic systems, two prominent families of formal methods have emerged: Stochastic Barrier Functions (SBFs) and finite-state abstractions [15], [16]. SBFs provide sufficient conditions for probabilistic safety through Lyapunov-like value functions that reason about the continuous state space directly [15], [17], [18]. In particular, the expected increase of the function along system trajectories is bounded, which can then be used to derive probabilistic safety guarantees via martingale theory [15], [17]. The primary challenges for stochastic barrier functions lie in efficiently constructing these functions for complex systems with non-conservative associated safety probabilities. Finite-state abstractions, on the other hand, reduce complex systems to finite-state representations, enabling the application of existing probabilistic model checking techniques [19], [20]. The guarantees for the finite-state model can subsequently be lifted to the concrete system via simulation relations. The challenges of abstraction-based methods primarily revolve around constructing abstractions that balance computational tractability with the preservation of essential system properties. Furthermore, understanding the connections between these two families of methods and what the advantages and limitations of each are remains an open problem; in particular, what method is preferable for which class of systems and problems.

This dissertation addresses the challenges faced within both families, SBFs and finite-state abstractions, by developing improved computational methods and novel theoretical advances grounded in the latest developments of optimization theory, neural network verification, and symbolic-numeric computational techniques.

1.2. Existing Work

1.2.1. Formal Methods for Cyber-Physical Systems

Formal methods for CPSs encompass a spectrum of modelling frameworks ranging from linear to non-linear dynamics, discrete to continuous time, and deterministic to stochastic systems. Common formalizations include difference and differential equations, differential inclusion, switched models, and hybrid automata [21]. The choice of model dictates both tractability and fidelity, with models of higher fidelity incurring higher computational costs. In this dissertation, the focus is on stochastic discrete-time systems, formalized as stochastic difference equations (see Section 2.3). This choice is due to the fact that many CPSs are stochastic by nature and digital computers have become the standard for control of these autonomous systems.

The focus of this dissertation is primarily on safety specifications, which are often described as simple properties but are fundamental to many safety-critical CPSs. Importantly, safety and reachability are dual, i.e., safety can be expressed as the complement of reachability with respect to (wrt.) given unsafe sets. Furthermore, complex, temporal specifications can often be reduced to reachability or safety properties [13]. Temporal logics provide a precise language to express and capture desired behaviours and constraints over time including sequences of events, timing constraints, and (probabilistic) thresholds [13], [22]. The specifications for stochastic systems are inherently probabilistic, which necessitates the use of quantitative and probabilistic formal methods [13]. The subsequent subsections describe probabilistic formal methods in greater detail. Although questions of numeric representations and accuracy arise in this context [23], we ignore these questions – the main contribution is novel methods for safety verification of stochastic systems; not formally verified implementations of the algorithms.

Verification methods for CPSs can broadly be classified along three axes: design-time vs. runtime verification, system classes, and methodology (model checking, certificate functions, reachability analysis). For the first axis, while runtime monitoring has a rich history (monitor synthesis, online conformance checking) [24], the focus here is design-time verification, as the primary objective of this dissertation is to verify safety properties *prior* to deployment. The second axis has been discussed above: the dissertation is restricted to studying formal methods for stochastic discrete-time systems. The third axis, methodology, broadly fall into three categories:

1. model checking and symbolic methods that explore the state space exhaustively or symbolically [13];
2. certificate functions, e.g., Lyapunov and barrier functions, that construct value functions, which encode properties of the system evolution [25]; and

3. reachability analysis techniques that compute over-approximations of reachable sets over time [26].

The following subsections review existing work for stochastic systems in each of these categories.

Before proceeding, it is important to stress that a core challenge irrespective of method family is scalability and state-space explosion [27]. For each category, we will discuss existing techniques for mitigating these challenges.

1.2.2. Stochastic Barrier Functions

Barrier functions, including Stochastic Barrier Function (SBF), are Lyapunov-like scalar functions that prove that the system trajectories stay within certain sets and thus provide a powerful framework to verify safety properties [25]. In the context of stochastic discrete-time systems, this is typically achieved by enforcing supermartingale conditions on the composition of the barrier function with the system dynamics. This bounds the expected increase in value along system trajectories [15], [28]. Using this fact, in addition to constraints on the value of the function in the initial and unsafe sets, it is possible to upper bound the probability of entering unsafe sets [15], [17]. For a formalization of SBFs and the resulting safety guarantees, see Chapter 3.

While the underpinning theory of SBFs was established in the 1960s [17], their use as a formal verification method for stochastic systems has only emerged recently [15], [18], [28]. The main reason is that the synthesis of suitable barrier functions for complex systems is a non-trivial challenge, which requires computational synthesis methods. Existing synthesis methods are focused solely on optimization-based approaches, and in particular, Sum-of-Squares (SoS) for polynomial systems [15], [28]. The idea is that if both the barrier template and system dynamics are polynomial then the expectation of the supermartingale constraint can be evaluated analytically via the computation of moments of the noise distribution. Thus, the synthesis problem can be formulated as a polynomial program, which is efficient to solve using Sum-of-Squares Programming (SoSP) [15], [29]. However, this approach is limited to polynomial systems and noise distributions with computable moments, and suffers from scalability issues due to both the multinomial number of terms in the basis and the quadratic nature of the Gramian matrix in the Semi-definite Programming (SDP) reformulation of the SoS program [29]. Chapter 4 aims to address this problem with a neural network barrier template.

Subsequent work, parallel to this dissertation, has explored compositional techniques for SBFs to address scalability challenges [30], [31]. These techniques leverage system structure to decompose large systems into smaller subsystems and synthesize a barrier function for each subsystem. These are subsequently composed into a joint barrier using a small-gain argument to obtain safety guarantees for the overall system. Although promising, these methods are still relatively new and restricted to immediately decomposable systems. Furthermore, the safety specifications must be locally decomposable to each subsystem, i.e., the specification cannot reason about interactions between subsystems.

Data-driven methods, again parallel to this work, have been proposed in settings

where the system dynamics and noise distribution are unknown [32], [33]. These methods typically rely on statistical methods such as concentration inequalities and Scenario Approach theory to construct barrier functions from data, with a formal confidence attached. While these methods open up new application domains, they often require prohibitively large amounts of data sampled under very restrictive assumptions such as sampling states and noise independently or sampling initial states from uniform distributions. Consequently, the resulting probability tends to be weaker than their model-based counterparts.

Finally, some works have explored alternative formulations of stochastic barrier functions, with the aim of reducing conservatism and improving applicability. These include k-inductive SBFs [34], [35] and stochastic control barrier functions [36], [37]. k-inductive SBFs reason about multiple (k) time steps simultaneously at the cost of increased computational complexity. Stochastic control barrier functions are a more traditional control barrier function formulation extended to stochastic systems, which is less computationally expensive but with the limitation that it potentially requires excessive control efforts. Additionally, barriers are assumed given, not synthesized, thus, requiring engineering efforts.

1.2.3. Abstraction and Model Checking-based Verification

Probabilistic model checking performs exhaustive or symbolic exploration of stochastic finite-state models to compute probabilities of satisfaction. Common model classes include Markov Decision Processes (MDPs) [38], [39], Interval Markov Decision Processes (IMDPs) [16], [40], and various other Robust Markov Decision Process (RMDP) formulations [41], [42] that trade between fidelity and tractability. For continuous-state stochastic systems, it is, however, generally infeasible to apply probabilistic model checking directly as the state spaces are not finite or enumerable. Instead, finite-state abstractions are constructed that approximate the concrete system while preserving relevant properties and allowing for easier analysis [43]. The formal characterization of the relationship between the concrete system and its abstraction is typically established via simulation relations (e.g., approximate, alternating, or probabilistic; their applicability depends on system models). These relations guarantee that the properties verified on the abstract model hold for, or can be lifted to, the concrete system [16], [38]. For verifying the abstract model, standard probabilistic model checking techniques and tools, e.g., **PRISM** [19] or **Storm** [20], may be employed.

A primary source of conservatism in abstraction-based methods is the discretization error introduced when constructing the finite-state model [43]. As a consequence, to achieve sufficient accuracy, a fine partitioning is typically required, which is computationally expensive. Numerous techniques have been proposed to mitigate this error, including

- adaptive gridding [40], [44], [45], which refines only regions where the discretization error is the largest;
- embedding *local* discretization errors into the model rather than establishing

a *global* error [16];

- corrective feedback controllers [46], which reduce the effect of discretization induced disturbances and thus the required partition resolution; and
- leveraging system structure such as additive noise [39] and sparse dependencies [47], enabling better scalability and thus finer partitioning without incurring a computational penality.

To further improve scalability, similar to SBFs, a family of methods revolve around decomposing the system into smaller subsystems, constructing abstractions for and verifying each subsystem individually, and then composing these abstractions to obtain an overall guarantee [48]. These composition techniques generally rely on small-gain [49]–[53] or dissipativity arguments [48], [54]–[56]. Similar to compositional techniques for SBFs, these methods require being able to decompose the specification to subsystems; that is, the specifications cannot describe about interactions between subsystems and the guarantees rely on the assumption that the interactions are sufficiently small.

Data-driven abstraction techniques fall into two categories: direct and indirect methods. Direct methods construct abstractions directly from data via statistical techniques, whereas indirect methods first learn system models from data, with statistical guarantees, and subsequently construct abstractions from the learned models. Direct methods for finite-state abstractions include Wasserstein balls [41], [57], [58], statistical hypothesis testing [59], and scenario approach-based techniques [60]–[62]. Indirect methods include Gaussian processes [63]–[66] and Bayesian estimation of credible parameter sets [67]. It is important to stress that despite their versatility, data-driven abstraction-based methods generally require large amounts of data. Furthermore, both memory and computational requirements for model checking the abstract model remain significant bottlenecks.

1.2.4. Probabilistic Reachability Analysis

Probabilistic reachability analysis is another prominent formal method to verify stochastic systems [68]. The key idea is to compute sets of states, i.e. flowpipes, with a guaranteed *minimum* probability of reaching or avoiding certain sets over a time horizon. Methods for computing these sets include dynamic programming formulations [68]–[70], chance-constrained formulations [71], sampling/particle-based techniques [72], [73], and Lagrangian methods [74]. A limitation of these methods is that they are largely restricted to linear systems [71], and for non-linear systems, they rely on unsound approximations [69], [73], although convergent in the limit.

1.3. Contributions

With the existing literature on formal methods for stochastic systems, discussed in the previous section, serving as a foundation, this dissertation makes two key contributions to the field:

- **Theoretical advances and novel synthesis methods for stochastic barrier functions:** Building upon existing techniques, this dissertation proposes new theoretical frameworks for Stochastic Barrier Functions (SBFs) and methods for synthesizing SBFs tailored to different classes of stochastic systems. These methods enhance the ability to construct barrier functions that bound the probabilistic safety properties of stochastic systems tighter existing synthesis methods.

- **Improved finite-state abstraction techniques:** This dissertation introduces advanced techniques for constructing finite-state abstractions of stochastic systems that preserve safety properties more accurately. By leveraging structural information about system dynamics and noise characteristics, these techniques yield abstractions that facilitate more efficient safety verification.

Together, these contributions push both the theoretical understanding of the verification of discrete-time stochastic systems and available computational methods.

1.4. OUTLINE

First, notation, background theory, and formalization of the problem settings considered are presented in Chapter 2. Proofs of key theoretical results are provided in this chapter to ensure that the dissertation is self-contained.

Subsequent chapters are divided into two main parts, each focussing on one of the two families of formal methods for stochastic systems as introduced in previous sections: SBFs and finite-state abstractions. Each part includes an introduction to the respective family.

Part I, which focuses on *Stochastic Barrier Functions*, includes Chapters 4, 5, and 6.

- Chapter 4 addresses a key limitation of existing synthesis methods for SBFs: scalability to complex systems. To this end, this chapter presents a method for synthesizing SBFs parameterized as neural networks along with a verification framework based on state-of-the-art neural network verification techniques. This chapter is based on [75].

- Chapter 5 further develops the theoretical foundations of SBFs by extending to the class of piecewise candidates. This allows for more flexible barrier functions to capture complex system dynamics and set geometries. Moreover, this chapter presents three synthesis methods for piecewise constant SBFs with different trade-offs between computational complexity and reliability of the computation. This chapter is based on [76] and [77].

- Chapter 6 presents a novel theoretical framework for SBFs synthesis based on the scenario approach from optimization theory for use in contexts where the noise distribution is unavailable. This chapter is based on [78] and [79].

Part II, which focuses on finite-state abstractions to *Robust Markov Decision Processes*, includes Chapters 8 and 9.

- Chapter 8 introduces a tool, **IntervalMDP.jl**, for modelling of IMDPs and model checking over this class, which includes algorithmic improvements targeting specifically General-Purpose Graphics Processing Units (GPUs) for significant acceleration. This chapter is based on [80],
- Chapter 9 presents a novel subclass of Robust Markov Decision Processes called factored Interval Markov Decision Processes (fIMDPs). In addition, this chapter proposes a symbolic-numeric abstraction technique to fIMDPs that leverages structural information about the concrete system. This structural information allows for less conservative abstractions that better preserve safety properties. This chapter is based on [81].

Chapters 4-6 and 8-9 all contain independent numerical experiments that illustrate the theoretical developments and computational methods proposed in each chapter. Furthermore, since each of these chapters considers a slightly different problem and this context is important for formal methods, their introduction contains a formal problem statement.

Finally, Chapter 10 concludes the dissertation by summarizing the key contributions, discussing their implications, and outlining potential directions for future research in the field of formal methods for stochastic systems.

2

Preliminaries

2.1. INTRODUCTION

In this chapter, we define the mathematical preliminaries and notation of the dissertation. This includes the general class of systems studied along with special subclasses in Section 2.3 to be studied in subsequent chapters. First, we will review convex analysis in Section 2.2 as it is a recurring theme throughout the methods presented in this dissertation including convex computational geometry, convex relaxations of non-linear functions, and convex and scenario optimization. Specific background on Stochastic Barrier Functions (SBFs) and Interval Markov Decision Process (IMDP)-based abstractions are given Chapters 3 and 7, respectively.

2.1.1. NOTATION

$\mathbb{R}$ and $\mathbb{N}$ represent the set of real and natural numbers with $\mathbb{N}_0 = \mathbb{N} \cup \{0\}$ and $\mathbb{R}_{\geq 0} = \{x \in \mathbb{R} : x \geq 0\}$. We denote vectors in $\mathbb{R}^n$ by italics, e.g. $x \in \mathbb{R}^n$. For any two vectors $x, y \in \mathbb{R}^n$, we denote the Hadamard product by $(x \odot y)_i = x_i y_i$. The set $\mathbb{S}_+^n$ is the set of symmetric positive semi-definite matrices of size $n \in \mathbb{N}$. For a finite set S, $|S|$ is its cardinality. Set-valued functions are denoted as $f : A \rightrightarrows B$ for any two sets A, B, i.e. $f(a) \subseteq B$ for any $a \in A$. For a set $X \subset \mathbb{R}^n$ and matrices $A \in \mathbb{R}^{n \times n}$ and $b \in \mathbb{R}^n$, the set $AX + b = \{Ax + b : x \in X\}$ is the affine transformation of X through A and b. The Minkowski sum of two (compact) sets A and B is denoted by $A \oplus B = \{a + b : a \in A, b \in B\}$. $\mathbf{1}_X(x)$ denotes the indicator function for the set X, that is, $\mathbf{1}_X(x) = 1$ if $x \in X$ and 0 otherwise. $\text{diag}(\cdot)$ denotes a square diagonal matrix with $\cdot$ on the diagonal. A partition of a set $X \subset \mathbb{R}^n$ is a set of regions $Q = \{q_1, \ldots, q_\ell\}$ with $q_i \subset X$ such that $\cup_{q \in Q} q = X$ and $q_i \cap q_j$ has zero Lebesgue measure for any two $i \neq j$. Given a partition Q of a set X, we say that a function $f : X \to Y$, where $Y \subset \mathbb{R}^m$, is a piecewise affine (PWA) if $f(x) = \max\{f_1(x), \ldots, f_\ell(x)\}$ where $f_i(x) = A_i x + b_i$ with matrices $A_i \in \mathbb{R}^{m \times n}$ and $b_i \in \mathbb{R}^m$ for $x \in q_i$ and $-\infty$ otherwise. $\text{ReLU}(x) = \max(0, x)$ is the rectified linear unit function, which is piecewise linear. The supremum norm $\|\cdot\|_\infty$ on a function $f : \mathbb{R}^n \to \mathbb{R}_{\geq 0}$ is defined as $\|f\|_\infty = \sup_{x \in \mathbb{R}^n} |f(x)|$.

Let Ω be an abstract space, $\mathcal{F}$ be a σ-algebra defined on this set, and $\mathbb{P}$ be a probability measure; we denote by $(\Omega, \mathcal{F}, \mathbb{P})$ the associated complete probability space. A random variable with values in $\mathbb{R}^n$ is a measurable function $\mathbf{x} : \Omega \to \mathbb{R}^n$. To distinguish from vectors, we denote random variables in bold.

A probability distribution γ over a finite set S is a function $\gamma : S \to [0, 1]$ satisfying $\sum_{s \in S} \gamma(s) = 1$. $\mathcal{D}(S)$ is the set of all probability distributions over S, and a subset of $\mathcal{D}(S)$ is called an ambiguity set. For $\underline{\gamma}, \overline{\gamma} : S \to [0, 1]$ such that $\underline{\gamma}(s) \leq \overline{\gamma}(s)$ for each $s \in S$ and $\sum_{s \in S} \underline{\gamma}(s) \leq 1 \leq \sum_{s \in S} \overline{\gamma}(s)$, an interval ambiguity set $\Gamma \subset \mathcal{D}(S)$ is the set of distributions such that

$$\Gamma = \{\gamma \in \mathcal{D}(S) : \underline{\gamma}(s) \leq \gamma(s) \leq \overline{\gamma}(s) \text{ for each } s \in S\}.$$

$\underline{\gamma}, \overline{\gamma}$ are often referred to as the interval bounds of the interval ambiguity set. The set of all interval ambiguity sets over S is denoted by $\text{int amb}(S)$. For n finite sets $S_1, \ldots, S_n$ we denote by $S_1 \times \cdots \times S_n$ their Cartesian product.

Definition 2.1 (Product ambiguity set)**.** *Given $S = S_1 \times \cdots \times S_n$ and n ambiguity sets $\Gamma_i \in \mathcal{D}(S_i)$, $i = 1, \ldots, n$, the product ambiguity set $\Gamma \subseteq \mathcal{D}(S)$ is defined as*

$$\Gamma = \bigotimes_{i=1}^{n} \Gamma_i = \left\{ \gamma \in \mathcal{D}(S) \,:\, \gamma = \bigotimes_{i=1}^{n} \gamma^i,\ \gamma^i \in \Gamma_i \right\},$$

where $s = (s_1, \ldots, s_n) \in S$ and $\otimes$ denotes the Kronecker product. Each Γ_i is called a marginal or component ambiguity set.

2.1.2. MARTINGALES

A discrete-time martingale is a stochastic process $\{\mathbf{W}_k\}_{k\in\mathbb{N}}$ that satisfies $\mathbb{E}[|\mathbf{W}_k|] < \infty$ and $\mathbb{E}[\mathbf{W}_{k+1} \mid \mathbf{W}_1, \mathbf{W}_2, \ldots, \mathbf{W}_k] = \mathbf{W}_k$ for all $k \in \mathbb{N}$ [82]. If the stochastic process $\{\mathbf{W}_k\}_{k\in\mathbb{N}}$ instead satisfies $\mathbb{E}[|\mathbf{W}_k|] < \infty$ and $\mathbb{E}[\mathbf{W}_{k+1} \mid \mathbf{W}_1, \mathbf{W}_2, \ldots, \mathbf{W}_k] \leq \mathbf{W}_k$ for all $k \in \mathbb{N}$, then we call it a *supermartingale*. If a supermartingale additionally is non-negative, then Ville's inequality [83] holds, which states that,

$$\mathbb{P}\left[\sup_{k\in\{0,\ldots,K\}} \mathbf{W}_k \geq a\right] \leq \frac{\mathbb{E}[\mathbf{W}_0]}{a} \tag{2.1}$$

for any real number $a > 0$ and horizon $K \in \mathbb{N}$. Equation (2.1) is one way to prove the safety probability bound of SBFs. See Chapter 3 for more details.

2.2. CONVEX ANALYSIS

As described in the introduction, convex analysis is repeatedly used throughout the dissertation. Thus, for completeness, we include the following section that introduces key concepts and results from convex analysis.

Definition 2.2 (Convex set)**.** *A set $X \subset \mathbb{R}^n$ is a convex set if and only if for any $x, y \in X$ and any $\alpha \in [0, 1]$, it holds that $\alpha x + (1 - \alpha)y \in X$. In other words, every line segment between two points in the set X is contained in X.*

Definition 2.3 (Convex function)**.** *A function $f : \mathbb{R}^n \to \mathbb{R}$ is convex if and only if for all $x, y \in \mathbb{R}^n$ and any $\alpha \in [0, 1]$, it holds that*

$$f(\alpha x + (1 - \alpha)y) \leq \alpha f(x) + (1 - \alpha) f(y). \tag{2.2}$$

2.2.1. COMPUTATIONAL CONVEX GEOMETRY

Computational geometry is the study of geometric representations and algorithms for operations (e.g. convex hull, Minkowski sum, linear transformations, projections). We start with a handful of useful set definitions; including their representation.

Definition 2.4 (Polyhedron)**.** *A (convex) polyhedron X in representation is the set*

$$X = \{x \in \mathbb{R}^n : Hx \leq h\} \tag{2.3}$$

for some matrix $H \in \mathbb{R}^{m\times n}$ and vector $h \in \mathbb{R}^m$ and where $\leq$ is interpreted elementwise.

This definition of a polyhedron is also called the halfspace representation as each row of H and h defines a halfspace and X is the intersection of m halfspaces. Any convex polyhedron can be equivalently stated as being generated by two finite sets of points; specifically, a convex hull and a conic hull. This equivalence is stated in the Minkowski-Weyl theorem [84].

Proposition 2.1 (Minkowski-Weyl theorem)**.** *For any set $X \subset \mathbb{R}^n$, the following two statements are equivalent:*

1. *X is a polyhedron, that is, for some matrix $H \in \mathbb{R}^{m \times n}$ and vector $h \in \mathbb{R}^m$, $X = \{x \in \mathbb{R}^n : Hx \leq h\}$.*
2. *There exists a finite set of vertices $V = \{\{v_1, \dots, v_m\} \subset \mathbb{R}^n$ and a finite set of rays $R = \{r_1, \dots, r_l\} \subset \mathbb{R}^n$ such that $X = \text{conv}(V) \oplus \text{cone}(R)$ where*

$$\text{conv}(V) = \left\{ x \in \mathbb{R}^n : x = \sum_{i=1}^{m} \alpha_i v_i, \alpha \in \mathbb{R}^m_{\geq 0}, \sum_{i=1}^{m} \alpha_i = 1 \right\} \text{ and} \tag{2.4}$$

$$\text{cone}(R) = \left\{ x \in \mathbb{R}^n : x = \sum_{i=1}^{l} \mu_i r_i, \mu \in \mathbb{R}^l_{\geq 0} \right\}. \tag{2.5}$$

The latter is called the vertex representation of the polyhedron. $\text{conv}(V)$ is called the convex hull of V and $\text{cone}(R)$ is called the conic hull of R. One algorithm for converting between the two representations is the double description algorithm [84]. If a polyhedron is compact, we say that the set is a polytope, according to the convention of the literature on convex computational geometry [85]. A polytope in vertex representation is completely generated by V. We denote the set of vertices of any polytope X by $\text{vert}(X)$, i.e. $V = \text{vert}(X)$. Conveniently, the set of all polyhedra is closed under intersection, Minkowski summation, and affine transformations; in other words, the intersection of two polyhedra is a polyhedron and equivalently for the other operations.

A specialization of polytopes widely used throughout this dissertation is hyperrectangles.

Definition 2.5 (Hyperrectangle)**.** *For a given $\underline{x} \in \mathbb{R}^n$ and $\overline{x} \in \mathbb{R}^n$ with $\underline{x} \leq \overline{x}$, the corresponding hyperrectangle is*

$$X = \{x \in \mathbb{R}^n : \underline{x} \leq x \leq \overline{x}\} \tag{2.6}$$

where $\leq$ is interpreted elementwise.

A hyperrectangle can equivalently be stated by its center $c = (\overline{x} + \underline{x})/2$ and vector-valued radius $r = (\overline{x} - \underline{x})/2$. For convenience, we denote by $\mathcal{H}_r(c)$ the hyperrectangle specified by center $c \in \mathbb{R}^n$ and radius $r \in \mathbb{R}^n_{\geq 0}$.

Finally, a useful set due to the focus on stochastic systems is the probability simplex.

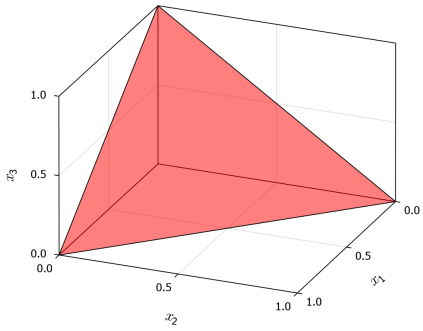


Figure 2.1: A probability simplex in 3 dimensions. It is a plane in 3D, thus its dimensionality is 2.

Definition 2.6 (Probability simplex)**.** *A probability simplex in n-dimensions is the set*

$$X = \left\{ x \in \mathbb{R}^n_{\geq 0} : \sum_{i=1}^{n} x_i = 1 \right\}. \tag{2.7}$$

In two dimensions, the probability simplex is the line between $(0, 1)$ and $(1, 0)$, and in three dimensions it is an equilateral triangle (see Figure 2.1).

In addition to the set definitions above, we also introduce uncertain affine transformations, which are particularly useful to the developments in the next Subsection 2.2.2, and state a result for a polyhedral over-approximations thereof.

Definition 2.7. *Let $\underline{A}, \overline{A} \in \mathbb{R}^{n \times m}$ and $\underline{b}, \overline{b} \in \mathbb{R}^n$ be given and let X be a set. Define $A(\alpha) = \alpha\underline{A} + (1 - \alpha)\overline{A}$ and $b(\alpha) = \alpha\underline{b} + (1 - \alpha)\overline{b}$ for any $\alpha \in [0, 1]$. Then the (linearly) uncertain affine transformation of X is $\cup_{\alpha \in [0,1]} A(\alpha)X + b(\alpha)$.*

Notice that the transformation $A(\alpha)X + b(\alpha)$ depends only linearly on α, which is either restrictive or conservative but also what enables efficient computation of an over-approximation (see the following Proposition 2.2). A pictorial example of an uncertain affine transformation and a concrete example of its power and limitation is shown in Figure 2.2, where the unit rectangle is rotated by the rotation matrix $R(\theta) = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix}$ between $\theta = 0$ and $\theta = 1$. Over-approximating an uncertain transformation with non-linear dependency on the uncertainty parameter α is hard, while if we consider the convex combination of $\underline{A} = R(0)$ and $\overline{A} = R(1)$ ($\underline{b} = \overline{b} = 0$), then the transformation can be over-approximated by the limits of the uncertainty parameter. This statement holds generally as we will show in the following proposition.

Proposition 2.2. *Let $\underline{A}, \overline{A} \in \mathbb{R}^{n \times m}$ and $\underline{b}, \overline{b} \in \mathbb{R}^n$ be given and let X be a convex set. Define $Y(\alpha) = A(\alpha)X + b(\alpha)$. Then $\cup_{\alpha \in [0,1]} Y(\alpha) \subset \text{conv}(Y(0) \cup Y(1))$.*

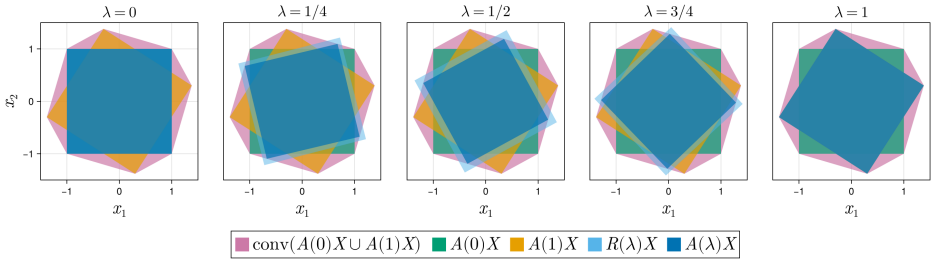


Figure 2.2: An example of a linearly uncertain affine transformation of a set $X = [-1, 1]^2$. The transformation is given by $\underline{A} = R(0)$, $\overline{A} = R(1)$, and $\underline{b} = \overline{b} = 0$ where $R(\theta) = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix}$ is the rotation matrix through an angle θ. Notice that while $\cup_{\alpha\in[0,1]} R(\alpha)X$ is not contained in $\mathrm{conv}(A(0)X \cup A(1)X)$, the linearly uncertain transformation $\cup_{\alpha\in[0,1]} A(\alpha)X$ is, since the transformation matrix is a *convex combination* between $\underline{A}$ and $\overline{A}$.

Proof. To prove the statement, we employ support functions $\rho_X(d) = \sup_{x\in X} d^\top x$. Support functions describe the (signed) distances from the origin to the supporting hyperplane of the set X in the direction d. We prove the subset relation by showing that the support function for $\mathrm{conv}(Y(0) \cup Y(1))$ is always larger than or equal the support function for $\cup_{\alpha\in[0,1]} A(\alpha)X + b(\alpha)$. That is, $\rho_{\cup_{\alpha\in[0,1]} A(\alpha)X+b(\alpha)}(d) \leq \rho_{\mathrm{conv}(Y(0)\cup Y(1))}(d)$ for any direction $d \in \mathbb{R}^n$.

$$\begin{aligned} \rho_{\cup_{\alpha\in[0,1]} Y(\alpha)}(d) &= \sup_{y\in\cup_{\alpha\in[0,1]} Y(\alpha)} d^\top y \\ &\leq \sup_{\alpha\in[0,1]} \sup_{y\in Y(\alpha)} d^\top y \\ &= \sup_{\alpha\in\{0,1\}} \sup_{y\in Y(\alpha)} d^\top y = \rho_{\mathrm{conv}(Y(0)\cup Y(1))}(d). \end{aligned} \tag{2.8}$$

Since $\rho_{\cup_{\alpha\in[0,1]} Y(\alpha)}(d) \leq \rho_{\mathrm{conv}(Y(0)\cup Y(1))}(d)$ for all $d \in \mathbb{R}^n$, it holds that $\mathrm{conv}(Y(0) \cup Y(1))$ is a (convex) over-approximation of $\cup_{\alpha\in[0,1]} Y(\alpha)$. □

Uncertain affine transformations generalize to any convex hull of affine transformation [86]. We limit ourselves to two affine transformations because, if we bound a non-linear function by two affine functions (see the following Subsection 2.2.2) in a given domain, then we can over-approximate reachable set of under the non-linear function in this domain. As a consequence, uncertain affine transformations are a powerful tool for analysing reachability and, as we shall see, can be constructed from diverse families of functions and non-restrictive assumptions.

2.2.2. LINEAR RELAXATIONS OF NON-LINEAR FUNCTIONS

In many applications involving non-linear functions such as optimisation, control, and formal verification, it is often useful to approximate these functions using simpler, more tractable representations. One common approach is to use (local) *linear relaxations*, which bound a non-linear function from above and below using affine

functions. These relaxations enable efficient analysis and computation by replacing complex non-linear behaviour with (conservative) linear approximations. The following definition formalizes this concept for functions defined over convex domains.

Definition 2.8 (Linear relaxations)**.** *Given a function $f : \mathbb{R}^n \to \mathbb{R}^m$ and a closed convex set $X \subset \mathbb{R}^n$, (over-approximating) linear relaxations in X are two linear functions $\underline{A}x + \underline{b}$ and $\overline{A}x + \overline{b}$ such that*

$$\underline{A}x + \underline{b} \leq f(x) \leq \overline{A}x + \overline{b}, \quad \text{for all } x \in X. \tag{2.9}$$

If $\underline{A} \neq \overline{A}$, we say that the linear relaxations are non-homogenous. Notice that we may treat these linear relaxations as an uncertain affine transformation $\cup_{\alpha\in[0,1]} A(\alpha)X + b(\alpha)$ such that $f(X) \subset \cup_{\alpha\in[0,1]} A(\alpha)X + b(\alpha)$, and by using Proposition 2.2, we can construct a polyhedral over-approximation of $f(X)$. This result is useful when studying verification of Neural Networks (NNs) (Chapter 4) and non-linear dynamical systems (Chapters 5, 6, and 9).

It is also useful to establish that linear relaxations always exist for a wide class of functions; namely (locally) Lipschitz continuous functions.

Proposition 2.3. *For any function f locally Lipschitz in a closed convex set $X \subset \mathbb{R}^n$ with the Lipschitz constant Lf, there exist linear relaxations $\underline{A}x + \underline{b}$ and $\overline{A}x + \overline{b}$ of f in X.*

Proof. We prove the statement by construction. To this end, assume without loss of generality (wlog.) that X is a hyperrectangle with center c and radius r and that Lf is the Lipschitz constant of f. That is,

$$\|f(x_1) - f(x_2)\|_\infty \leq Lf\|x_1 - x_2\|_\infty, \quad \text{for all } x_1, x_2 \in X. \tag{2.10}$$

This implies the component-wise bounds

$$f(c) - Lf\|x - c\|_\infty \cdot \mathbf{1} \leq f(x) \leq f(c) + Lf\|x - c\|_\infty \cdot \mathbf{1}, \quad \text{for all } x \in X, \tag{2.11}$$

where $\mathbf{1} \in \mathbb{R}^m$ is a vector of all ones. Let $M = \sup_{x\in X}\|x - c\|_\infty = \|r\|_\infty$. Then, we obtain the bounds

$$\underline{b} = f(c) - LfM \cdot \mathbf{1} \leq f(x) \leq f(c) + LfM \cdot \mathbf{1} = \overline{b}, \quad \text{for all } x \in X, \tag{2.12}$$

where $\underline{A} = \overline{A} = 0$. Thus, trivial linear relaxations always exist for any Lipschitz continuous function. □

While the constructive proof of Proposition 2.3 provides a method for computing linear relaxations, the method is often very conservative; in part due to the constant bounds computed. Instead, we can use certified first-order Taylor expansions $(f(c) + \nabla f(c)(x - c)) \oplus \mathcal{R}$ where $\mathcal{R} \subset \mathbb{R}^m$, called the residual, is a hyperrectangle such that

$$f(x) \in (f(c) + \nabla f(c)(x - c)) \oplus \mathcal{R}, \quad \text{for all } x \in X. \tag{2.13}$$

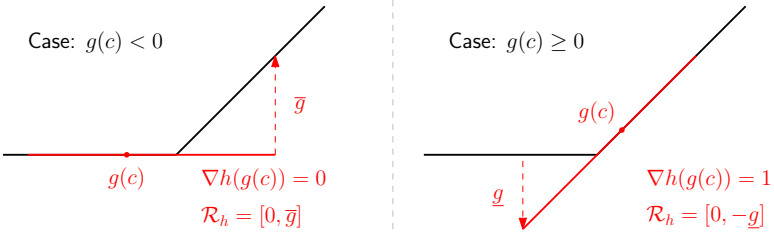


Figure 2.3: Computing the residual for ReLU in a certified first-order Taylor expansion can be split in two cases depending on $g(c) < 0$ or not. See Example 2.1 for the worked example.

Computing $\mathcal{R}$ can be done efficiently when f is twice continuously differentiable using the Lagrange error bound [38], [87]. However, a programmatic and more general approach is to forward propagate the certified Taylor expansion, (locally) expanding $\mathcal{R}$ as necessary [87]. We show this approach in the following worked example.

Example 2.1 (Propagating certified first-order Taylor expansions)**.** *Let $f = h \circ g$ where $h(x) = \max(0, x)$ is a ReLU function (notice ReLU is* not *twice continuously differentiable) and assume a certified first-order Taylor expansion of g*

$$g(x) \in (g(c) + \nabla g(c)(x - c)) \oplus \mathcal{R}_g, \quad \text{for all } x \in X. \tag{2.14}$$

Then, we trivially have $f(c) = h(g(c))$ and $\nabla f(c) = \nabla h(g(c)) \nabla g(c)$. Notice that $\nabla h(g(c)) = \mathbf{1}_{\{x \geq 0\}}(g(c))$. We can trivially compute an input interval $[\underline{g}, \overline{g}]$ for h from the Taylor expansion such that we can bound $\mathcal{R}_f$ as $\mathcal{R}_f = \mathcal{R}_h \oplus \overline{\mathbf{1}}_{\{x \geq 0\}}(g(c)) \mathcal{R}_g$ where

$$\mathcal{R}_h = \left\{ R \in \mathbb{R}^m : 0 \leq R \leq \overline{\mathcal{R}}_h \right\}, \quad \overline{\mathcal{R}}_h = \begin{cases} \overline{g} & \text{if } g(c) < 0 \\ -\underline{g} & \text{otherwise} \end{cases}. \tag{2.15}$$

See Figure 2.3 for a visual representation of how to calculate the residual $\mathcal{R}_h$.

A generalization of this method is CROWN [88], [89], a framework for propagating general linear bounds, which was developed for neural network verification and is considered state-of-the-art in this context [90]. Compared to propagating certified first-order Taylor expansions, CROWN (in forward-mode) directly propagates the linear relaxations $\underline{A}x + \underline{b}$ and $\overline{A}x + \overline{b}$ through the computation graph representing f and relaxes non-linear nodes with non-homogenous linear relaxations rather than a residual. However, CROWN can also be performed in backward-mode where linear bounds are propagated from the output to the input, which interestingly yields tighter bounds even under the same local relaxations [89]; although with significantly higher computational costs. Figure 2.4 shows an example of backward-mode CROWN. CROWN in forward-mode similar to Interval Bound Propagation (IBP) with the restriction $\underline{A} = \overline{A} = 0$ [91], [92].

While linear relaxations are powerful, as witnessed by being state-of-the-art for neural network verification, they can be conservative, especially when the input

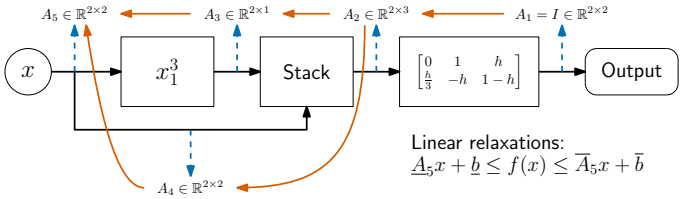


Figure 2.4: CROWN in backward-mode starts with $\underline{A}_1 = \overline{A}_1 = I$ and $\underline{b} = \overline{b} = 0$ at the output and propagates the linear relaxation backwards through the computation graph. At each node, new linear relaxations $\underline{A}_i$ and $\overline{A}_1$ are computed and relaxation errors are accumulated into $\underline{b}$ and $\overline{b}$ The computation graph in this figure is the function $f(x) = x + h\begin{pmatrix} x_2 \\ \frac{1}{3}x_1^3 - x_1 - x_2 \end{pmatrix}$ where $h \in \mathbb{R}_{>0}$ is given.

region spans highly non-linear ranges. Therefore, we introduce partitioning for tighter relaxations.

Definition 2.9 (Uncertain PWA relaxation)**.** *Assume a partitioning $Q = \{q_1, \dots, q_\ell\}$ of $X \subset \mathbb{R}^n$ and local linear relaxations $\underline{A}_i x + \underline{b}_i \leq f(x) \leq \overline{A}_i x + \overline{b}_i$ for all $x \in q_i$. Define*

$$\hat{f}(x, \alpha) = \max\{\hat{f}_1(x, \alpha), \dots, \hat{f}_\ell(x, \alpha)\} \tag{2.16}$$

where $\alpha \in [0,1]$ and $\hat{f}_i(x, \alpha) = A_i(\alpha)x + b_i(\alpha)$ if $x \in q_i$ with $A_i(\alpha) = \alpha\underline{A}_i + (1-\alpha)\overline{A}_i$ and $b_i(\alpha) = \alpha\underline{b}_i + (1-\alpha)\overline{b}_i$, and zero otherwise. Then $F(x) = \{\hat{f}(x, \alpha) : \alpha \in [0,1]\}$ is an uncertain PWA relaxation of f.

Proposition 2.4. *Given an uncertain PWA relaxation F of f, then it holds that $f(x) \in F(x)$ for all $x \in X$.*

Proof. For any input $x \in X$, since Q is a partitioning of X, there exists a region q_i such that $x \in q_i$. Then $F(x) = \{\hat{f}_i(x, \alpha) : \alpha \in [0,1]\}$ and by the definition of linear relaxations, there exists an $\alpha \in [0,1]$ such that $\hat{f}_i(x, \alpha) = f(x)$. □

Figure 2.5 shows an example of an uncertain PWA relaxation of a continuous function f over a bounded domain X.

2.2.3. LINEAR PROGRAMMING

Linear programming is the family of optimization problems where the objective (minimization or maximization) and the constraints are linear. The restriction to linear constraints enables efficient algorithms, in particular interior point and simplex methods, to solve the problem [93].

Definition 2.10 (Linear Programming (LP))**.** *An LP problem is a convex optimization problem of the form*

$$\begin{aligned} \min_{z \in \mathbb{R}^d} \quad & c^\top z \\ \text{s.t.} \quad & Az \leq b, \end{aligned} \tag{2.17}$$

for some given matrix $A \in \mathbb{R}^{m\times d}$ and vectors $b \in \mathbb{R}^m$, $c \in \mathbb{R}^d$.

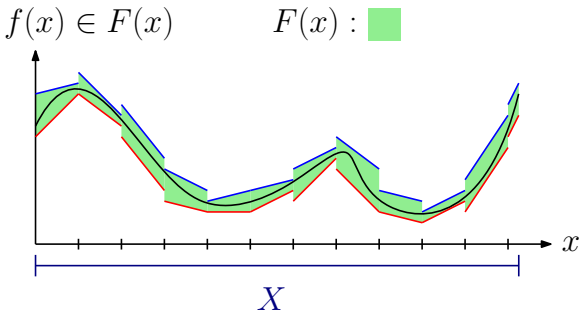


Figure 2.5: An uncertain PWA relaxation F of a function f over a bounded domain X is a set-valued uncertain PWA function such that $f(x) \in F(x)$ for all $x \in X$.

LP is special because it is the only class of convex optimization with a zero duality gap [93]. In duality theory, decision variables are replaced by dual variables and the minimization objective with maximizations, with the appropriate change of constraints and objective function. Only for LP does the original problem, called the primal, and the dual problem resolve to the same objective value.

Definition 2.11 (Asymmetric dual LP problem)**.** *The (asymmetric) dual LP problem to* (2.17) *is the following:*

$$\begin{aligned} \max_{\lambda \in \mathbb{R}^m} \quad & b^\top \lambda \\ \text{s.t.} \quad & A^\top \lambda = c, \\ & \lambda \leq 0, \end{aligned} \tag{2.18}$$

where λ are called the dual variables.

Proposition 2.5 (Strong duality [93, Section 5.2.1])**.** *Let $z^\star$ and $\lambda^\star$ denote the optimal solution to* (2.17) *and* (2.18) *respectively. Then $c^\top z^\star = b^\top \lambda^\star$.*

An extension of LP is robust, or semi-infinite, LPs where the coefficients are uncertain with some structure. We are interested in the case of polyhedral uncertainty as we can reformulate this class as a regular LP. This class is crucial to the developments in Chapter 6.

$$\begin{aligned} \min_{z \in \mathbb{R}^d} \quad & c^\top z \\ \text{s.t.} \quad & (Az + a)^\top x \leq B^\top z + b, \quad \text{for all } x \in X, \end{aligned} \tag{2.19}$$

where $c \in \mathbb{R}^d$, $A \in \mathbb{R}^{n \times d}$, $a \in \mathbb{R}^n$, $B \in \mathbb{R}^d$, and $b \in \mathbb{R}$ are given and $X \subset \mathbb{R}^n$ is a given polyhedron. As mentioned, using Proposition 2.5, we can recast (2.19) as an LP, enabling the use of standard solvers.

Proposition 2.6. *Denote the feasible set of* (2.19) *by*

$$\mathcal{Z} = \{z \in \mathbb{R}^d : (Az + a)^\top x \leq B^\top z + b, \text{ for all } x \in X\}. \tag{2.20}$$

Define the optimization problem

$$\begin{aligned} \min_{z\in\mathbb{R}^d,\lambda\in\mathbb{R}^m} \quad & c^\top z \\ \text{s.t.} \quad & h^\top\lambda \le B^\top z + b \\ & H^\top\lambda = Az + a, \quad \lambda \ge 0, \end{aligned} \tag{2.21}$$

where $X = \{x \in \mathbb{R}^n : Hx \le h\}$ *and the projected feasible set*

$$\mathcal{Z}' = \{z \in \mathbb{R}^d : \exists\lambda \in \mathbb{R}^n_{\ge 0},\ h^\top\lambda \le B^\top z + b,\ H^\top\lambda = (Az + a)\}. \tag{2.22}$$

Then, it holds that $\mathcal{Z} = \mathcal{Z}'$.

Proof. At the core of this result is the strong duality (Proposition 2.5 with *non-negative* dual variables) between the linear programs

$$\begin{aligned} \max_x \quad & (Az + a)^\top x \\ \text{s.t.} \quad & Hx \le h \end{aligned} \qquad \text{and} \qquad \begin{aligned} \min_\lambda \quad & \lambda^\top h \\ \text{s.t.} \quad & H^\top\lambda = Az + a, \quad \lambda \ge 0. \end{aligned}$$

We need to show that $\mathcal{Z} = \mathcal{Z}'$ In one direction, pick any element $z \in \mathcal{Z}$. Let $\bar{x} = \text{argmin}_{x\in X}(Az + a)^\top x$. Hence, by strong duality, there exists a $\lambda \in \mathbb{R}^n_{\ge 0}$ with $H^\top\lambda = Az + a$ such that $(Az + a)^\top\bar{x} = \lambda^\top h =\le B^\top z + b$, which implies $z \in \mathcal{Z}'$. For the other direction, fix any $z \in \mathcal{Z}'$ and pick $\bar{\lambda} = \text{argmin}_{H^\top\lambda = Az+a,\ \lambda\ge 0}\,\lambda^\top h$. We notice that by strong duality we have $\bar{\lambda}^\top h = \sup_{x\in X}(Az + a)^\top x \le B^\top z + b$. Then we conclude that $z \in \mathcal{Z}$, thus concluding the proof. □

2.2.4. SCENARIO THEORY

Chance-constrained programming problems are optimization problems where certain constraints are satisfied with high probability under some stochasticity. They are useful for ensuring solutions to stochastic problems are valid with high probability. To define chance-constrained programming problems, let $\mathbf{v}$ be a random variable taking values in $\mathbb{R}^n$ according to the continuous distribution $p_{\mathbf{v}}$. Then, given a violation threshold ϵ, a chance-constrained problem is the following:

$$\begin{aligned} \min_{z\in\mathbb{R}^d} \quad & c^\top z \\ \text{s.t.} \quad & \mathbb{P}_{\mathbf{v}\sim p_{\mathbf{v}}}\{g(z,\mathbf{v}) \le 0\} \ge 1 - \epsilon, \end{aligned} \tag{2.23}$$

where $g(z, v)$ is a function that is convex in z for each value of v and measurable in v for each value of z.

Although solving (2.23) directly is hard, the scenario approach theory allows one to solve chance-constrained programming problems with high confidence using samples [94]. Let $D = \{v_1, \dots, v_N\}$ be independent and identically distributed (i.i.d.) samples from $p_{\mathbf{v}}$. The sample set can be interpreted as a joint random variable $D : \Omega^N \to (\mathbb{R}^n)^N$ on the probability space $(\Omega^N, \bigotimes_N\mathcal{F}, \mathbb{P}^N)$, where Ω^N is the N-fold Cartesian product of Ω, and $\bigotimes_N\mathcal{F}$ is the product σ-algebra generated by the σ-algebra $\mathcal{F}$, and $\mathbb{P}^N$ is the induced product measure on Ω^N. Then, the scenario

program corresponding to (2.23) is the following

$$\begin{aligned} \min_{z\in\mathbb{R}^d} \quad & c^\top z \\ \text{s.t.} \quad & g(z,v) \leq 0, \quad \text{for all } v \in D. \end{aligned} \tag{2.24}$$

To apply scenario approach theory, the following conditions are necessary [94].

Assumption 2.1. *We assume that, almost surely wrt. the measure $\mathbb{P}^N$, the following two conditions hold:*

- *The feasible set $\mathcal{Z} = \{z \in \mathbb{R}^d : g(z,v) \leq 0, \forall v \in D\}$ has non-empty interior.*
- *The optimal solution of (2.24) exists and is unique.*

Uniqueness can always be enforced with a tie-break rule. We denote the unique solution of (2.24) by $z^\star(D)$, which is a well-defined random variable on the space Ω^N. A key result within the scenario approach theory establishes an upper bound on the tail distribution of the constraint violation probability associated with $z^\star(D)$.

Proposition 2.7 (Scenario confidence bounds [94])**.** *Consider (2.24) and suppose that Assumption 2.1 holds. Then, for any $\epsilon \in (0,1)$, we have that*

$$\mathbb{P}^N\{D : V(z^\star(D)) > \epsilon\} \leq \sum_{i=0}^{d-1} \binom{N}{i} \epsilon^i (1-\epsilon)^{N-i} = \beta, \tag{2.25}$$

where $V(z) = \mathbb{P}_{\mathbf{v}\sim p_{\mathbf{v}}}\{g(z,\mathbf{v}) > 0\}$ is the violation probability of $z \in \mathbb{R}^d$.

In other words, given a set D of i.i.d. samples from $p_{\mathbf{v}}$, the *optimal* solution to (2.24) is a solution to (2.23) with at least probability $1-\beta$. We call $1-\beta$ the confidence and β the confidence parameter.

2.3. Discrete-time Stochastic Systems

For the purpose of this dissertation, we consider discrete-time stochastic systems of the form

$$\mathbf{x}[k+1] = f(\mathbf{x}[k], \mathbf{v}[k]), \quad \mathbf{x}[0] = x_0 \tag{2.26}$$

where x_0 is the initial state of the system and $\mathbf{v}[k]$ is the process noise that affects the system, taking values in $\mathbb{R}^m$. We assume that the noise $\mathbf{v}[k]$ is i.i.d. for all time steps $k \in \mathbb{N}_0$ with the density $p_{\mathbf{v}}$. We write $\mathbb{P}_{\mathbf{v}}$ for the probability measure induced by $p_{\mathbf{v}}$. The vector-valued function $f : X \times \mathbb{R}^m \to X$, where $X \subseteq \mathbb{R}^n$ is the state space, represents the dynamics of the systems. We assume that X is equipped with its standard Borel σ-algebra and that for every $x \in X$, the function $f(x,\cdot)$ is Borel measurable. The stochasticity of the process $\mathbf{x}[k]$ is purely induced by $\mathbf{v}[k]$ via f.

Remark 2.1 (Relaxation of i.i.d. noise)**.** *The i.i.d. assumption on the noise is for some formal verification use cases restrictive. However, the choice including this assumption is only for ease of exposition. To relax the assumption, one could instead assume that $p_{\mathbf{v}}$ is time-varying or that $p_{\mathbf{v}}$ belongs to a known set of distributions, what is called an ambiguity set, for example, a Wasserstein ball.*

Remark 2.2 (Stopped process)**.** *If the range of f is larger than X, then the process $\mathbf{x}[k]$ may at some time $k \in \mathbb{N}_0$ leave X beyond which f may be undefined. In this case, we consider a stopped process $\tilde{\mathbf{x}}[k]$ where for a first exit time $\tilde{k} = \inf\{k \in \mathbb{N}_0 : \mathbf{x}[k] \notin X\}$, the stopped process is defined as*

$$\tilde{\mathbf{x}}[k] = \begin{cases} \mathbf{x}[k] & \text{if } k < \tilde{k} \\ \mathbf{x}[\tilde{k}] & \text{if } k \geq \tilde{k} \end{cases}. \tag{2.27}$$

To avoid obtuse notation, we henceforth, with slight abuse of notation and wlog., omit $\tilde{\cdot}$ and write $\mathbf{x}[k]$ for the stopped process.

Given an initial condition $\mathbf{x}[0] = x_0$, $\mathbf{x}[k]$ is a Markov process with a well-defined probability measure $\mathbb{P}^{x_0}$ [95, Proposition 7.45] generated by the noise distribution $p_{\mathbf{v}}$ such that for sets $X_0, X_{k+1} \subseteq X$ it holds that

$$\begin{aligned} &\mathbb{P}^{x_0}[\mathbf{x}[0] \in X_0] = \mathbf{1}_{X_0}(x_0) \\ &\mathbb{P}^{x_0}[\mathbf{x}[k+1] \in X_{k+1} \mid \mathbf{x}[k] = x_k] = T(X_{k+1} \mid x_k), \end{aligned} \tag{2.28}$$

where T is the stochastic kernel for the dynamical system, i.e.

$$T(X_{k+1} \mid x_k) = \int_{\mathbb{R}^m} \mathbf{1}_{X_{k+1}}(f(x_k, v))\, d\mathbb{P}_{\mathbf{v}}(v) \tag{2.29}$$

The objective in this dissertation is to certify a lower bound on the probability of safety; under different assumptions on the structure and availability of (2.26).

Definition 2.12 (Safety probability)**.** *Given a stochastic process $\mathbf{x}[k]$, $k \in \mathbb{N}_0$ governed by the dynamics in (2.26), a set of safe states X_s, a set of initial states $X_0 \subset X_s$, and a time horizon $K \in \mathbb{N}$, the probability of safety P_{safe} is*

$$P_{\text{safe}} = \inf_{x_0 \in X_0} \mathbb{P}^{x_0}[\forall k \in \{0, \dots, K\}, \mathbf{x}[k] \in X_s]. \tag{2.30}$$

We denote the complement of the safe set X_s called the unsafe set by X_u.

Remark 2.3 (Finite time safety)**.** *The probabilistic safety in Definition 2.12 is over a finite horizon, as for many systems, the safety probability P_{safe} over an infinite horizon is trivially zero; in particular systems with additive noise of unbounded support. For infinite horizon specifications, it is important to distinguish whether the property can be satisfied on a finite trace or requires an infinite trace [13]. Safety over an infinite horizon requires an infinite trace, as a finite trace does not provide sufficient evidence that the system will* never *exit the safe set.*

Generally, computing P_{safe} is intractable due to the non-enumerable nature of the state space. To further clarify this point, observe that the safety probability can be characterized by the following value function[1], which represents the probability of failure.

$$\begin{aligned} V_0(x) &= \mathbf{1}_{X_u}(x), \\ V_{k+1}(x) &= \mathbf{1}_{X_u}(x) + \mathbf{1}_{X_s}(x)\mathbb{E}[V_k(f(x,\mathbf{v}))]. \end{aligned} \tag{2.31}$$

More specifically, we have the probability of failure equal to the value function for the initial states at time step K, that is, $\mathbb{P}^{x_0}\left[\exists k \in \{0,\dots,K\}, \mathbf{x}[k] \in X_u\right] = V_K(x_0)$, and thus the safety probability is $P_{\text{safe}} = 1 - \left(\sup_{x_0 \in X_0} V_K(x_0)\right)$. Even for the simplest systems, e.g., linear 1D systems with additive Gaussian noise, and simple set geometries, exact computation of V_k is intractable for any $k \in \{1,\dots,K\}$. As a consequence, (sound) approximations to P_{safe} are necessary; these are the subject of this dissertation.

Remark 2.4. *As mentioned in Chapter 1, safety and reachability are dual of each other. Moreover, these objectives can be combined to a reach-avoid objective*

$$P_{\text{reach-avoid}} = \inf_{x_0 \in X_0} \mathbb{P}^{x_0}\left[\exists k \in \{0,\dots,K\}, \mathbf{x}[k] \in X_g, \forall k' \in \{0,\dots,k\}, \mathbf{x}[k'] \notin X_u\right] \tag{2.32}$$

for a goal set $X_g \subset X$ and an unsafe set $X_u = X \setminus X_u$, which can be characterized by

$$\begin{aligned} V_0(x) &= \mathbf{1}_{X_g}(x), \\ V_{k+1}(x) &= \mathbf{1}_{X_g}(x) + \mathbf{1}_{X_s}(x)\mathbb{E}[V_k(f(x,\mathbf{v}))]. \end{aligned} \tag{2.33}$$

For the remainder of this section, we consider specializations and generalizations of the form (2.26) that are particularly useful for the developments in this dissertation. We start with additive noise systems, which often allow explicit expressions of the underlying transition kernel, and moreover, when the additive noise is Gaussian, affords an analytical expression for tight bounds on the transition probability to any hyperrectangular region (see Chapter 5 for more details on this applicability).

Definition 2.13 (Additive noise systems)**.** *If some system dynamics has the form*

$$\mathbf{x}[k+1] = f(\mathbf{x}[k]) + \mathbf{v}[k], \quad \mathbf{x}[0] = x_0, \tag{2.34}$$

where $f : X \to X$ then the system is called an additive noise system.

For additive noise systems, the noise $\mathbf{v}[k]$ naturally takes values in $\mathbb{R}^n$.

If, for an additive noise, the function f is an affine, i.e. $f(x) = Ax + b$ for some given matrices $A \in \mathbb{R}^{n \times n}$ and $b \in \mathbb{R}^n$, then we say that the system is an affine system. Similarly, if f is PWA then we say that the system is a PWA system.

A generalization of (2.26) is to consider a control input, which can be used to control the system toward higher levels of safety according to the methods developed through this dissertation. This form will be explored in Chapter 9 where control is inherent to the method developed, although it also enables verification.

[1] The convention for value functions is to denote the last time step by 0 and increment going *backwards* in time.

Definition 2.14 (Control systems)**.** *A system of the form*

$$\mathbf{x}[k+1] = f(\mathbf{x}[k], u[k], \mathbf{v}[k]), \quad \mathbf{x}[0] = x_0 \tag{2.35}$$

where $f : X \times U \times \mathbb{R}^m \to X$ is the controlled vector field over a bounded control space $U \subset \mathbb{R}^{n_u}$ is a discrete-time stochastic control system. A given state-feedback controller $\pi : X \to U$ induces the control process $\mathbf{u}[k] = \pi(\mathbf{x}[k])$ and the closed-loop system $\mathbf{x}[k+1] = f(\mathbf{x}[k], \pi(\mathbf{x}[k]), \mathbf{v}[k])$, which is of the form (2.26)*.*

I

Stochastic Barrier Functions

3

Background on Stochastic Barrier Functions

As discussed in Section 2.3, computing the safety probability P_{safe} is generally intractable due to the uncountable nature of the state space [96]. Consequently, (under)-approximations are necessary. Stochastic Barrier Functions (SBFs) have emerged as a tool for this purpose [15], [18]. The general idea is to construct a Lyapunov-like function over the state domain whose value is bounded in a given unsafe set and also does not increase too rapidly with system evolution. By reasoning over growth rate of the value function, one can rely on this two properties to construct a formal certificate for (a lower bound of) the safety probability. The notion of SBF is related to stochastic Lyapunov functions, where a value function is designed to *decrease* in expectation along the system evolution [18], [97].

Definition 3.1 (Stochastic Barrier Function)**.** *Let $X_s \subset \mathbb{R}^n$, $X_0 \subseteq X_s$ and $X_u = \mathbb{R}^n \setminus X_s$ be the safe set, the set of initial states, and the unsafe set, respectively. Furthermore, let the system $\mathbf{x}[k]$ be governed by (2.26). Then, a non-negative measurable function $B : \mathbb{R}^n \to \mathbb{R}_{\geq 0}$ is a Stochastic Barrier Function for $\mathbf{x}[k]$ if there exists two constant $c \geq 0$ and $\eta \geq 0$ such that*

$$B(x) \geq 1 \qquad \forall x \in X_u, \tag{3.1a}$$

$$B(x) \leq \eta \qquad \forall x \in X_0, \tag{3.1b}$$

$$\mathbb{E}_{\mathbf{v}}[B(f(x, \mathbf{v}))] \leq B(x) + c \qquad \forall x \in X_s. \tag{3.1c}$$

We omit for brevity the subscript of $\mathbb{E}$ when the random variable over which the expected value is taken is unambiguous; e.g. in (3.1c) where $\mathbf{v}$ is a random variable while x is *not* a random variable (indicated by the lack of bold font) as it enters the equation via the for-all quantification.

A pictorial example of an SBF is shown in Figure 3.1. Intuitively, $B(\mathbf{x}[K])$ is a random variable and the barrier conditions guarantee that the expectation of B

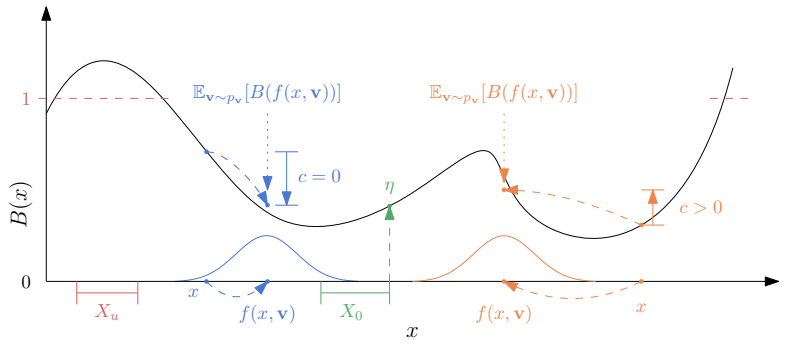


Figure 3.1: A barrier function B is a non-negative function that is greater than 1 in the unsafe region X_u. η is an upper bound of $B(x)$ for $x \in X_0$. $c \in \mathbb{R}_{\geq 0}$ is an upper bound on the one-step expected increase in value in the safe set $X_s = \mathbb{R}^n \setminus X_u$ under the dynamics in (2.26). The barrier function, through c and η, certifies a lower bound for the probabilistic safety over a finite horizon K, namely $P_{\text{safe}} \geq 1-(\eta + cK)$.

does not grow by more than c at each time step. In other words, $B(\mathbf{x}[K])$ is a c-martingale and therefore, Conditions (3.1a)-(3.1c) allow us to bound P_{safe}.

Proposition 3.1. *([17], [98], [15]) Let B be a barrier function for $\mathbf{x}[k]$ and $K \in \mathbb{N}$ be a given time horizon. Then, it holds that $P_{\text{safe}} \geq 1-(\eta + cK)$.*

Proof. We prove the certificate of SBFs by constructing a proper supermartingale from $B(\mathbf{x}[k])$ and then use Ville's inequality [83] to bound the probability that the value of the supermartingale exceeds a certain threshold within a time horizon.

To this end, define the stochastic process $W(\mathbf{x}[k]) = B(\mathbf{x}[k]) + c(K-k)$ and observe that the following holds for any $\mathbf{x}[0] = x_0$ and all $k \in \{0, \dots, K-1\}$,

$$\begin{aligned}\mathbb{E}[W(\mathbf{x}[k+1]) \mid \mathbf{x}[0] = x_0] &= \mathbb{E}[B(\mathbf{x}[k+1]) \mid \mathbf{x}[0] = x_0] + c(K-k-1) \\ &\leq B(\mathbf{x}[k]) + c(K-k) \\ &= W(\mathbf{x}[k]).\end{aligned} \tag{3.2}$$

Since we have $\mathbb{E}[W(\mathbf{x}[k+1]) \mid \mathbf{x}[0] = x_0] \leq W(\mathbf{x}[k])$, the process $W(\mathbf{x}[k])$ is a non-negative supermartingale for $k \in \{0, \dots, K\}$. As $W(\mathbf{x}[k])$ is a non-negative supermartingale (see Subsection 2.1.2), Ville's inequality [83] holds, which in this context can be formulated as the following: for any $x_0 \in X_0$,

$$\mathbb{P}^{x_0}\left[\sup_{k\in\{0,\dots,K\}} W(\mathbf{x}[k]) \geq 1\right] \leq \mathbb{E}[W(\mathbf{x}[0]) \mid \mathbf{x}[0] = x_0]. \tag{3.3}$$

Combining Ville's inequality with Condition (3.1a), the following chain of reasoning

provides analytical bounds on the probability of reaching X_u in finite time.

$$\begin{aligned}\sup_{x_0 \in X_0} \mathbb{P}^{x_0}\left[\exists k \in \{0, \dots, K\}, \mathbf{x}[k] \in X_u\right] &\leq \sup_{x_0 \in X_0} \mathbb{P}^{x_0}\left[\sup_{k \in \{0,\dots,K\}} W(\mathbf{x}[k]) \geq 1\right] \\ &\leq \sup_{x_0 \in X_0} \mathbb{E}[W(\mathbf{x}[0]) \mid \mathbf{x}[0] = x_0] \\ &= \sup_{x_0 \in X_0} B(x_0) + cK \leq \eta + cK.\end{aligned} \tag{3.4}$$

To relate the above inequality to the probability of safety, we use the safety-reachability duality as follows

$$P_{\text{safe}} = 1 - \sup_{x_0 \in X_0} \mathbb{P}^{x_0}\left[\exists k \in \{0, \dots, K\}, \mathbf{x}[k] \in X_u\right] \geq 1 - (\eta + cK), \tag{3.5}$$

which concludes the proof. □

The above proof relies on a supermartingale inequality whose theory originates in the analysis of betting strategies. An alternative proof, which provides more insight into the relation between the true level of safety and the lower bound guaranteed by an SBF, relies on dynamic programming. Namely, a proof-by-induction over the safety characterization in (2.31) shows that $B(x) + ck$ upper bounds the value function $V_k(x)$ at each time step $k \in \{0, \dots, K\}$. We encapsulate this reasoning in the following lemma.

Lemma 3.1. *Let V_k be defined as in (2.31) and let B be a SBF, i.e. satisfying the conditions in Definition 3.1. Then $V_k(x) \leq B(x) + ck$ for all $x \in X$ and all $k \in \{0, \dots, K\}$.*

Proof. Base case $k = 0$: Clearly, since $B(x) \geq 1$ for all $x \in X_u$ and $B : \mathbb{R}^n \to \mathbb{R}_{\geq 0}$, we have $B(x) \geq \mathbf{1}_{X_u}(x) = V_0(x)$. Therefore, we conclude that the statement holds for $k = 0$.

Inductive step: Assume that $V_k(x) \leq B(x) + ck$ for all $x \in X$. Then we will prove that $V_{k+1}(x) \leq B(x) + c(k+1)$ for all $x \in X$. First, for any $x \in X_u$, we apply the same reasoning as the base case; since $B(x) \geq 1$ for all $x \in X_u$ we have

$$B(x) + c(k+1) \geq \mathbf{1}_{X_u}(x) = V_{k+1}(x), \quad \text{for all } x \in X_u. \tag{3.6}$$

Next, for any $x \in X_s$, we have

$$\begin{aligned}V_{k+1}(x) &= \mathbb{E}[V_k(f(x, \mathbf{v}))] \\ &\leq \mathbb{E}[B(f(x, \mathbf{v})) + ck)] \\ &\leq B(x) + c(k+1).\end{aligned} \tag{3.7}$$

The last inequality holds by Condition (3.1c). Finally, since $X = X_s \cup X_u$, we have $V_{k+1}(x) \leq B(x) + c(k+1)$ for all $x \in X$, which concludes the proof. □

Clearly, since $V_K(x) \leq B(x) + cK$ by Lemma 3.1, it holds that

$$P_{\text{safe}} \geq 1 - \left(\sup_{x_0 \in X_0} B(x_0) + cK \right) = 1 - (\eta + cK). \tag{3.8}$$

Remark 3.1. *The proof of Lemma 3.1 is similar but alternative and simpler proof of Theorem 2 in [45]. The unifying perspective in [45] between SBFs and abstraction-based verification techniques through dynamic programming was initially discovered as part of the work towards piecewise constant SBFs (PWC-SBFs) [76] where the underlying system is effectively abstracted to an IMDP. Then, [76] constructs a SBF for this IMDP. See Chapter 5 for more details on PWC-SBFs.*

By reordering (3.7), we observe the following relation for all $k \in \{0, \dots, K-1\}$:

$$c \geq \mathbb{E}[B(f(x, \mathbf{v}))] - B(x) \geq \mathbb{E}[V_k(f(x, \mathbf{v}))] - V_{k+1}(x), \quad \text{for all } x \in X_s. \tag{3.9}$$

Therefore, c upper bounds the expected one-step increase not only in value of the SBF B, but also the failure probability $\{V_k\}_{k \in \mathbb{N}_0}$, for all states in the safe set X_s. Consequently, the term cK in Proposition 3.1 is *at least* as conservative as resetting the stochastic process $\mathbf{x}[k]$ to the state in X_s with the highest transition probability to X_u and performing that transition *at every time step*. Despite conservatism, SBFs may find use in scenarios that remain intractable for other methods. Furthermore, the function surface of B may provide valuable information for designing safety controllers, obtaining a desired level of safety [99].

To construct an SBF, state-of-the-art methods employ convex optimization for the following problem where θ represents the parameterization of the barrier template [15], [98], which minimizes the certified failure probability, or equivalently maximizes the certified safety probability.

$$\begin{aligned} \min_{\eta, c, \theta} \quad & \eta + cK \\ \text{s.t.} \quad & \eta \geq 0, c \geq 0, \\ & (3.1a), (3.1b), (3.1c). \end{aligned} \tag{3.10}$$

The choice of barrier template affects which subclass of convex optimization can be employed (if any). Examples include an exponential template $B(x) = e^{x^\top S_\theta x} - 1$ with $S \in \mathbb{S}^n_+$, which can be reformulated via relaxation to SDP, and polynomial templates $B(x) = \sum_i \theta_i p_i(x)$ for a set of basis polynomials $p_1(x), p_2(x), \dots$ where (3.10) can be solved using SoSP. Exponential templates are restricted to rotated ellipsoid safe sets, and polynomial templates lack scalability due to the multi-combinatorial number of monomials. These issues are the motivation for the first contribution presented in this dissertation in Chapter 4.

4

Stochastic Neural Barrier Functions

We can only see a short distance ahead,
but we can see plenty there that needs to be done.

Alan Turing

This chapter is a copy of *F. B. Mathiesen, S. C. Calvert, and L. Laurenti, "Safety certification for stochastic systems via neural barrier functions",* IEEE Control Systems Letters*, 2023.* **DOI**: 10.1109/LCSYS.2022.3229865 [75] with minor corrections to streamline presentation.

4.1. Introduction

Traditionally, the synthesis of an SBF has been framed as a Sum-of-Squares Programming (SoSP) problem [15], that is, (3.10) where the barrier function and dynamics are assumed to be polynomial. Then the conditions of an SBF in the Definition 3.1 are encoded as SoS constraints. A fundamental challenge central to this chapter and an intrinsic limitation of SoSP is the scalability of this synthesis process to higher dimensions and complex dynamics. Specifically, the number of monomial basis functions $p_1, p_2, \ldots$ in the barrier polynomial $B(x) = \sum_i \theta_i p_i(x)$ is equal to the binomial coefficient $\binom{n+d}{d}$ where n is the dimensionality of x and d is the polynomial degree [100]. Therefore, SoSP suffers poor scaling in both the number of variables and the degree of the polynomial, and as we will see in Section 4.4, empirical evidence shows a polynomial degree $d \geq 12$ is typically needed for a sufficient level of certified safety. Furthermore, in the supermartingale condition (3.1c), the barrier and dynamics polynomials are multiplied increasing the maximum degree in the SoS constraint. As a consequence, synthesizing SBFs for systems of higher dimensions or requiring high-degree polynomial templates due to complex dynamics remains intractable.

To overcome this limitation, we turn our attention to barriers parameterized by NNs. Neural networks have demonstrated remarkable success in scaling to real-world problems in various domains [101], [102]. The universal approximation theorems for NNs – that NNs can, in the asymptotic limit of size, approximate any function [103] – often serve as the theoretical justification for this success. Moreover, the exponential approximation accuracy [104], combined with computationally efficient and well-understood gradient-based training algorithms [105], [106], makes NNs an attractive alternative. In conclusion, NNs are undeniably powerful and flexible function approximators, which is useful in SBF synthesis.

The shift to neural SBFs, or Neural Barrier Functions (NBFs), introduces a new challenge: *how can we verify neural SBFs?* With previous synthesis methods based on convex optimization, satisfaction is inherently guaranteed through constrained optimization, e.g. interior point methods. However, with NNs, optimization is unconstrained, thus requiring post hoc verification. Furthermore, *how can we train these neural networks to satisfy the necessary conditions in a dense domain?* Training neural barrier candidates is problematic, since gradient-based training methods for NNs rely on sampling, while the barrier conditions must be satisfied everywhere. The goal of this chapter is to answer these two research questions.

Problem 4.1

Given a system $\mathbf{x}[k+1] = f(\mathbf{x}[k], \mathbf{v}[k])$ (i.e., of the general form (2.26)), an initial set X_0, a safe set X_s, and a time horizon K, compute a lower bound ρ on P_{safe}.

Approach Our framework for verifying NBFs draws on recent advances in the verification of NNs, namely CROWN [88], [89], to build linear relaxations. We ex-

tend CROWN to expectation operators to accommodate the supermartingale condition in (3.1c) and augment the verification method with a branch-and-bound scheme to reduce conservatism. The latter is necessary because experimental results show that CROWN in isolation is insufficient for dense verification over large sets. The verification procedure is described in Section 4.2. For training an NBF, the loss function intuitively encodes each of the SBF conditions to train the network towards satisfying the SBF conditions (see Section 4.3). However, as detailed before, this only improves the barrier on *individual points*, while it is necessary to cover *compact sets*. To address this challenge, we present two components that are critical in efficiently training an NBF: adversarial training and Counter-Example Guided Inductive Synthesis (CEGIS). Adversarial training [107] plays an essential role in ensuring coverage over a non-zero Lebesgue measure of the state space. CEGIS [108] is the process of training the NBF on counterexamples generated by the verification procedure to *efficiently* improve constraint satisfaction. To showcase the advances, we compare in Section 4.4 our proposed approach with state-of-the-art methods.

4.2. VERIFICATION OF NEURAL BARRIER FUNCTIONS

In this section, we will show our approach to verifying that a given NN B_θ with continuous activation functions, where θ represents the parameters (weights and biases), is a valid SBF for (2.26), that is, if it satisfies the conditions in Definition 3.1. For simplicity, we assume that B_θ is a fully-connected NN with no activation function on the output. Later, in Section 4.3, we will describe how to obtain a candidate function, i.e. training a NBF, based on the verification presented in this section. Before we continue to describe the verification method, we need some basic assumptions.

Assumption 4.1. *The safe set X_s is a compact set.*

As we shall see later, with the above Assumption 4.1, we can verify the validity of a candidate NBF over an unbounded space.

Assumption 4.2. *The neural network B_θ and the vector field f are locally Lipschitz.*

Assumption 4.2 ensures the existence of (local) linear relaxations of B_θ and f and is fairly non-restrictive. In fact, any fully-connected NN with continuous (potentially non-smooth) activation functions is locally Lipschitz. To construct linear relaxations of a NN we employ CROWN [88], which is considered state-of-the-art within neural network verification and has won VNN-COMP several years [90]. CROWN works by propagating linear lower and upper bounds through the computation graph that represents the function to bound; in this case B_θ and f.

Our verification approach is based on employing local relaxation techniques to build uncertain PWA functions of B_θ and f. That is, with a partitioning $Q = \{q_1, \dots, q_\ell\}$ of the state space $X \subset \mathbb{R}^n$, for each region q two affine functions such that

$$\underline{A}_q x + \underline{b}_q \leq B_\theta(x) \leq \overline{A}_q x + \overline{b}_q \quad \text{for all } x \in q \tag{4.1}$$

See Section 2.2.2 for more details on how to compute linear and uncertain PWA relaxations. The following lemma follows trivially from the above definition of linear relaxations to verify $B_\theta(x) \geq 0$ for all $x \in X$ and $B_\theta(x) \geq 1$ for all $x \in X_u$, that is, the non-negativity and unsafe set conditions of Definition 3.1.

Lemma 4.1. *Assume X is a compact set and let $Q_{X_u} \subset Q$ be the smallest set of regions such that $X_u \subseteq \cup_{q \in Q_{X_u}} q$. Then, if*

$$\min_{q \in Q} \min_{x \in q} \underline{A}_q x + \underline{b}_q \geq 0 \qquad \min_{q \in q_{X_u}} \min_{x \in q} \underline{A}_q x + \underline{b}_q \geq 1, \tag{4.2}$$

then the conditions $B_\theta(x) \geq 0$ for all $x \in X$ and $B_\theta(x) \geq 1$ for all $x \in X_u$ are satisfied.

4

Proof. We prove the statement for $B_\theta(x) \geq 0$ for all $x \in X$. The proof for $B_\theta(x) \geq 1$ for all $x \in X_u$ follows a similar structure.

First, note that the assumption that X is compact ensures that each region $q \in Q$ is compact and thus the existence of linear relaxations within this region. Therefore, $\min_{x \in q} B_\theta(x) \geq \min_{x \in q} \underline{A}_q x + \underline{b}_q \geq 0$ is well-defined. Since $X = \cup_{q \in Q} q$, the statement $\min_{q \in Q} \min_{x \in q} \underline{A}_q x + \underline{b}_q \geq 0$ implies $B_\theta(x) \geq 0$ for all $x \in X$, concluding the proof. □

Note that under the assumption that each $q \in Q$ is a convex polytope, which can always be enforced by the partition strategy, then Lemma 4.1 reduces to the solution of a set of linear programs, and if furthermore each region q is hyperrectangle, then there exists an analytical formula for the solution [92].

Remark 4.1 (Unbounded state space)**.** *Lemma 4.1 holds under the assumption that X is compact, i.e., closed and bounded. We can relax this assumption by cleverly modifying the NBF B_θ using Assumption 4.1 that the safe set is compact. In particular, we can define the* almost-everywhere continuous *function $B_\theta(x) = \max(1, \tilde{B}_\theta(x))$ where $\tilde{B}_\theta(x)) = NN_\theta(x)$ for $x \in X_s$ and zero otherwise, and where NN_θ is a neural network. In other words, the neural network only models the behaviour of the barrier in the safe set and we assume wlog. B_θ to be 1 everywhere else. Then, with this modification, (i) the condition $B_\theta(x) \geq 1$ for all $x \in X_u$ holds by construction, and (ii) it is only necessary to verify non-negativity over the compact domain X_s.*

Remark 4.2 (Correct-by-construction non-negativity)**.** *It is possible to encode non-negativity in the architecture of the neural network that is B_θ, by appending the computation graph with a* ReLU*. This, however, introduces issues for both verification and training. For training, if any gradient step updates the parameters θ such that $NN_\theta(x) \leq 0$ for all $x \in X$, then for all future steps, the gradient is blocked from backpropagating at the last* ReLU *such that the parameters will never update and the training stalls. For verification, appending* ReLU *reduces the available information for smart partitioning in the branch-and-bound algorithm that we will discuss in Subsection 4.2.1.*

Using a strategy similar to Lemma 4.1, we can compute the smallest *verified* η given the partition Q. To this end, let $Q_{X_0} \subset Q$ be the set of regions such that $X_0 \subseteq \cup_{q \in Q_{X_0}} q$. Then, we can choose η to be $\eta = \max_{q \in Q_{X_0}} \max_{x \in q} \overline{A}_q x + \overline{b}_q$ such that $\eta \geq B_\theta(x)$ for all $x \in X_0$.

We now turn our attention to c, and consequently to the computation of the supermartingale condition (3.1c). Unfortunately, due to the non-linearity of the functions involved, computing $\mathbb{E}[B_\theta(f(x, \mathbf{v}))]$ is analytically intractable. As a consequence, we again rely on computing local under- and over-approximations. In particular, consider finite partitions Q_{X_s} and q_v respectively of the (compact) safe set X_s and of the uncertainty space $V \subset \mathbb{R}^m$, and let $\tilde{Q} = Q_{X_s} \times Q_v$. For now, assume that V is compact. We will relax this assumption after Theorem 4.1. Then, as discussed above, for each region $q \in Q_{X_s}$ and each product region $\tilde{q} = q \times q_v \in \tilde{Q}$ we can find row vectors $\underline{A}_q, \overline{A}_{q_x}, \overline{A}_{q_v} \in \mathbb{R}^{1 \times n}$ and scalars $\underline{b}_q, \overline{b}_{\tilde{q}} \in \mathbb{R}$ such that

$$\forall x \in q, \quad \underline{A}_q x + \underline{b}_q \leq B_\theta(x) \tag{4.3}$$

$$\forall (x, v) \in \tilde{q}, \quad B_\theta(f(x, v)) \leq \overline{A}_{q_x} x + \overline{A}_{q_v} v + \overline{b}_{\tilde{q}}. \tag{4.4}$$

Note that if a NN is composed with a continuous function, i.e. $B_\theta(f(x, v))$, then CROWN-like techniques can still be applied on the composite computation graph to derive linear relaxations of the composed function. In particular, the continuous function can be treated as the first layer of the neural network and perform Linear Bound Propagation [89].

The following theorem uses the above relaxations to bound $\mathbb{E}[B(f(x, \mathbf{v}))]$ and consequently find a lower bound on c.

Theorem 4.1. *Let Q_{X_s} and Q_v respectively be partitions of X_s and V. For $\tilde{q} = q \times q_v \in Q_{X_s} \times Q_v$, define*

$$A_q = -\underline{A}_q + \sum_{q_v \in Q_v} \overline{A}_q \mathbb{P}_{\mathbf{v}}[\mathbf{v} \in q_v], \tag{4.5a}$$

$$b_q = -\underline{b}_q + \sum_{q_v \in Q_v} \left(\overline{b}_{\tilde{q}} + \overline{A}_{q_v} \mathbb{E}[\mathbf{v} : \mathbf{v} \in q_v] \right) \cdot \mathbb{P}_{\mathbf{v}}[\mathbf{v} \in q_v]. \tag{4.5b}$$

Let $c = \max_{q \in Q_{X_s}} \max_{x \in q} \left(A_q x + b_q \right)$. Then, for any $x \in X_s$ it holds that

$$\mathbb{E}[B_\theta(f(x, \mathbf{v}))] - B_\theta(x) \leq \max_{q \in Q_{X_s}} \max_{x \in q} \left(A_q x + b_q \right) = c. \tag{4.6}$$

Proof. Fix any $q \in Q_{X_s}$. For any $x \in q$ it holds that

$$\begin{aligned} \mathbb{E}[B_\theta(f(x, \mathbf{v}))] &= \sum_{q_v \in Q_v} \int_{q_v} B_\theta(f(x, v)) \, d\mathbb{P}_{\mathbf{v}}(v) \\ &\leq \sum_{q_v \in Q_v} \int_{q_v} \left(\overline{A}_{q_x} x + \overline{A}_{q_v} v + \overline{b}_{\tilde{q}} \right) d\mathbb{P}_{\mathbf{v}}(v). \end{aligned} \tag{4.7}$$

By leveraging linearity of the integral operator, we can lift all terms and factors but v outside the integration. Note that the remaining integrals are $\int_{q_v} d\mathbb{P}_{\mathbf{v}}(v) = \mathbb{P}_{\mathbf{v}}[\mathbf{v} \in q_v]$ and $\int_{q_v} v\, d\mathbb{P}_{\mathbf{v}}(v) = \mathbb{E}[\mathbf{v} : \mathbf{v} \in q_v]\mathbb{P}_{\mathbf{v}}[\mathbf{v} \in q_v]$. The latter is called a partial expectation.

Combining the above bound for the expectation with $\underline{A}_q x + \underline{b}_q \leq B_\theta(x)$ for all $x \in q$, we get an upper bound $A_q x + b_q$ for $\mathbb{E}[B_\theta(f(x, \mathbf{v})] - B_\theta(x)$ for all $x \in q$. It then follows that

$$\max_{x \in X_s} (\mathbb{E}[B_\theta(f(x, \mathbf{v})] - B_\theta(x)) \leq \max_{q \in Q_{X_s}} \max_{x \in q} \left(A_q x + b_q\right) = c. \tag{4.8}$$

□

4

The computation of A_q and b_q in Theorem 4.1 requires the evaluation $\mathbb{P}_{\mathbf{v}}[\mathbf{v} \in q_v]$ and $\mathbb{E}[\mathbf{v} : \mathbf{v} \in q_v]$. For various classes of distributions, such as Gaussian with diagonal covariance matrix, uniform, or finite support distributions, these expressions can be computed in closed forms. Otherwise, numerical approximations may be required.

Although we assumed that the support of the noise V has bounded support, Theorem 4.1 also applies in the case of unbounded support; it just requires a rigorous construction of linear relaxations in unbounded noise regions q_v of the partitioned uncertainty space V and the barrier construction in Remark 4.1. To this end, observe that since X_s is compact, there exists an $\overline{B} \geq 1$ such that $\overline{B} \geq B_\theta(x)$ for all $x \in X_s$, and thus also for all $x \in X$. Then, for an unbounded noise region $q_v \in Q_v$ and any (compact) region $q \in Q$, we can bound $B_\theta(f(x, v))$ in $\tilde{q} = q \times q_v$ by $\overline{B}$, that is, $B_\theta(f(x, v)) \leq \overline{b}_{\tilde{q}}$ for all $(x, v) \in \tilde{q}$ with $\overline{A}_{q_x} = \overline{A}_{q_v} = 0$ and $\overline{b}_{\tilde{q}} = \overline{B}$. If we capture enough probability mass of V by compact regions in the partitioning, then, since the linear bounds are summed together weighted by their probability in (4.5a)-(4.5b), the tails covered by the uniform upper bound should not dominate the linear bound for the expectation in Theorem 4.1.

4.2.1. A Branch and Bound Scheme for Verification

To improve the scalability of our verification framework, we adapt the branch-and-bound partitioning scheme of [109] to our setting. In particular, starting from a coarse partitioning of X, the method gradually refines it by splitting regions and pruning those that already satisfy the barrier conditions. For convenience, we assume that all regions q are hyperrectangles such that the linear programs in Lemma 4.1 and Theorem 4.1 have an analytical solution [92]. We perform branch-and-bound independently for each of the conditions in Definition 3.1. In what follows, we explain the partitioning scheme for the unsafe set condition in (3.1a); the others follow analogously.

We start with a coarse initial partition Q_{X_u} of X_u. Then, as shown in Lemma 4.1, the condition in (3.1a) reduces to checking if $\min_{q \in Q_{X_u}} \min_{x \in q} \underline{A}_q x + \underline{b}_q \geq 1$. As we start with a coarse partition, initially the bounds may be very conservative. Consequently, we gradually refine Q_{X_u}. At each iteration, we split all regions in Q_{X_u} and, based on

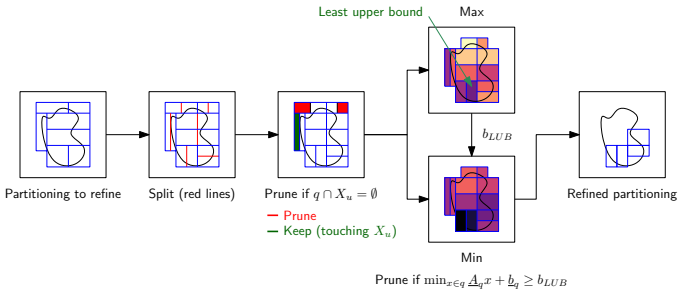


Figure 4.1: One iteration of the automatic partitioning scheme with splitting and pruning for (3.1a). The set X_u is shown as a black blob, and hyperrectangles $q \in Q_{X_u}$ are split and pruned. $b_{LUB} = \min_{q \in Q_{X_u}} \min_{x \in q} \overline{A}_q x + \overline{b}_q$ denotes the least upper bound.

4

the linear relaxations, check if the condition is provably satisfied or violated. That is, the following, respectively:

$$\min_{q \in Q_{X_u}} \min_{x \in q} \underline{A}_q x + \underline{b}_q \geq 1, \tag{4.9}$$

$$\min_{q \in Q_{X_u}} \min_{x \in q} \overline{A}_q x + \overline{b}_q \leq 1. \tag{4.10}$$

To select the split axis, we pick the one with the largest linear coefficients $|\underline{A}_q| + |\overline{A}_q|$, because that maximizes tightening of both upper and lower bounds, weighted by the width of the region $(\overline{q} - \underline{q})$ along the given axis to avoid elongated regions because they are shown empirically to yield loose bounds.

If not terminated, we proceed with the branch phase, where the regions $q \in Q_{X_u}$ are pruned if q does not influence the satisfaction of (3.1a) and can be discarded. Specifically, we may prune q if either $q \cap X_u = \emptyset$ or the minimum value of B_θ in q is greater than the least upper bound in another region $q' \in Q_{X_u}$. One iteration of splitting and pruning is shown in Figure 4.1. Finally, we stop the partitioning when the largest gap between the upper and lower bound for $B_\theta(x)$ is less than a threshold $t_{gap} > 0$, that is, if $\max_{q \in Q_{X_u}} \max_{x \in q} \left(\overline{A}_q x + \overline{b}_q\right) - \left(\underline{A}_q x + \underline{b}_q\right) < t_{gap}$, where the verification is considered inconclusive, or *unknown* in the terminology of three-valued logic, which suggests the additional training is beneficial. We show the complete branch-and-bound algorithm in Algorithm 4.1.

4.3. ADVERSARIAL TRAINING

We now describe the neural network training procedure, which is key to obtain a valid stochastic barrier function B_θ. As the conditions in Definition 3.1 need to hold over regions in the state space, the rationale behind our approach is to adapt certified adversarial training of NNs [91], [92], [110] to our setting.

Our training procedure starts by independently sampling M training points from each set X, X_u, X_0, and X_s. We denote the resulting data set, respectively, as D_X,

Algorithm 4.1 Branch-and-bound algorithm to verify $\min_{x \in X_u} B_\theta(x) \geq 1$, given an initial partition Q_{X_u} of X_u.

Require: Barrier candidate B_θ, initial partition Q_{X_u}, and threshold t_{gap}
Ensure: Sound SAT/UNSAT *or* UNKNOWN if

$$1 \in \left[\min_{q \in Q_{X_u}} \min_{x \in q} \underline{A}_q x + \underline{b}_q - t_{gap}, \min_{q \in Q_{X_u}} \min_{x \in q} \overline{A}_q x + \overline{b}_q + t_{gap} \right].$$

for Region q in Q_{X_u} **do**
  $\underline{A}_q, \underline{b}_q, \overline{A}_q, \overline{b}_q := \textbf{CROWN}(B_\theta, q)$
while $\max_{q \in Q_{X_u}} \max_{x \in q} \left(\overline{A}_q x + \overline{b}_q \right) - \left(\underline{A}_q x + \underline{b}_q \right) \leq t_{gap}$ **do**
  ▷ *Branch*
  $Q_{new} := \emptyset$
  for q in Q_{X_u} **do**
    $A_q := \left(|\underline{A}_q| + |\overline{A}_q| \right)^\top \odot (\overline{q} - \underline{q})$
    $d := \operatorname{argmax}_{1 \leq i \leq n} A_{q,i}$ ▷ *Select optimal axis to split*
    $q_1, q_2 := \textbf{SPLIT-MID}(q, d)$ ▷ *Split on axis d*
    $Q_{new} := Q_{new} \cup \{q_1, q_2\}$
  $Q_{X_u} := Q_{new}$
  ▷ *Bound*
  for Region q in Q **do**
    $\underline{A}_q, \underline{b}_q, \overline{A}_q, \overline{b}_q := \textbf{CROWN}(B_\theta, q)$
  if $\min_{q \in Q_{X_u}} \min_{x \in q} \underline{A}_q x + \underline{b}_q \geq 1$ **then**
    return SAT
  if $\min_{q \in Q_{X_u}} \min_{x \in q} \overline{A}_q x + \overline{b}_q \leq 1$ **then**
    return UNSAT
  $b_{lub} := \min_{q \in Q_{X_u}} \min_{x \in q} \overline{A}_q x + \overline{b}_q$ ▷ *Least upper bound*
  $Q := \{q \in Q_{X_u} : \min_{x \in q} \underline{A}_q x + \underline{b}_q \leq b_{lub}\}$
  $Q := \{q \in Q_{X_u} : q \cap X_u \neq \emptyset\}$
return UNKNOWN

D_{X_u}, D_{X_0}, D_{X_s}. Furthermore, N noise vectors $v_1, \cdots, v_N$ are independently sampled according to $p_{\mathbf{v}}$. Then, the loss function $\mathcal{L}$ is defined as follows:

$$\mathcal{L} = (1-\kappa)\mathcal{L}_v + \kappa(\eta^{(M)} + c^{(M)}K) \tag{4.11}$$

$$\mathcal{L}_v = \frac{1}{M}\sum_{x\in D_X} \mathcal{R}_\epsilon(-B_\theta(\cdot), x) + \frac{1}{M}\sum_{x\in D_{X_u}} \mathcal{R}_\epsilon(1 - B_\theta(\cdot), x) \tag{4.12}$$

$$\eta^{(M)} = \max_{x\in D_{X_0}} \mathcal{R}_\epsilon(B_\theta(\cdot), x) \tag{4.13}$$

$$c^{(M)} = \max_{x\in D_{X_s}} \mathcal{R}_\epsilon\left(\frac{1}{N}\sum_{j=1}^{N} B_\theta(f(\cdot, v_j)) - B_\theta(\cdot), x\right). \tag{4.14}$$

where $\kappa \in [0,1]$ weighs between a valid SBF and tight probability bounds. For a function $g : \mathbb{R}^n \to \mathbb{R}$, the operator $\mathcal{R}_\epsilon(g,x) = \max_{x':\|x-x'\|_\infty \le \epsilon} \text{ReLU}(g(x'), 0)$ returns an upper bound of $g(x)$ over a hyperrectangle centered at x with radius $\epsilon \ge 0$, which we employ to enforce the satisfaction of Conditions (3.1a)-(3.1c) on regions of the input space around the training points. This operator is what makes the training adversarial and it performs a function analogous to that of the consolidator in FOSSIL [111], which is a tool for synthesis of neural barrier and Lyapunov functions for deterministic systems. Intuitively, choosing a large value of ϵ for $\mathcal{R}_\epsilon(g,x)$ will facilitate the task of finding a valid SBF, while a small ϵ leads to tighter bounds on the safety probability. To find a certified upper bound for $\mathcal{R}(g,x)$ we employ IBP, as this counterintuitively results in higher certified robustness [91], [110]. Note that the analytical nature of IBP allows one to employ standard auto-diff tools to take the gradient of $\mathcal{L}$ wrt. θ.

With training progressing, we gradually reduce κ from 1 to 0 to shift from minimizing $\eta^{(M)} + c^{(M)}K$ to minimizing $\mathcal{L}_v$ with the goal of first finding regions of the parameter space with tight probability bounds and then anneal the neural network to a valid barrier. The following proposition establishes a theoretical non-asymptotic relationship between the conditions of the barrier function and the loss function by exploiting the robust loss, provided enough and sufficiently space samples. As the loss is a trade-off between $\mathcal{L}_v$ and $\eta^{(M)}+c^{(M)}K$ through κ, we analyse both extremes of κ.

Proposition 4.1 (Loss/satisfaction equivalence)**.** *First, assume wlog. the following:*

$$X = \bigcup_{x \in D_X} \{x' : \|x - x'\|_\infty \leq \epsilon\} \tag{4.15a}$$

$$X_u = \bigcup_{x \in D_{X_u}} \{x' : \|x - x'\|_\infty \leq \epsilon\}, \tag{4.15b}$$

$$X_0 = \bigcup_{x \in D_{X_0}} \{x' : \|x - x'\|_\infty \leq \epsilon\}, \tag{4.15c}$$

$$X_s = \bigcup_{x \in D_{X_s}} \{x' : \|x - x'\|_\infty \leq \epsilon\}. \tag{4.15d}$$

Then, the following two statements hold:

- *($\kappa = 0$): $B_\theta(x) \geq 0$ for all $x \in X$ and $B_\theta(x) \geq 1$ for all $x \in X_u$ are satisfied if and only if $\mathcal{L} = 0$.*
- *($\kappa = 1$): $\mathcal{L} = \eta + cK$.*

Proof. We start with the case $\kappa = 0$. In one direction, assume $B_\theta(x) \geq 0$ for all $x \in X$ and $B_\theta(x) \geq 1$ for all $x \in X_u$. Then, for any $x \in D_X$ it holds $\mathcal{R}_\epsilon(-B_\theta(\cdot), x) = 0$ and for any $x \in D_{X_u}$ it holds $\mathcal{R}_\epsilon(1 - B_\theta(\cdot), x) = 0$. Thus, $\mathcal{L} = \mathcal{L}_v = 0$.

In the other direction, assume that for any $x \in D_X$ it holds $\mathcal{R}_\epsilon(-B_\theta(\cdot), x) = 0$ and for any $x \in D_{X_u}$ it holds $\mathcal{R}_\epsilon(1 - B_\theta(\cdot), x) = 0$. Fix any $x \in D_X$, then we have $B_\theta(x') \geq 0$ for all x' such that $\|x - x'\|_\infty \leq \epsilon$. By (4.15a), we have $B_\theta(x) \geq 0$ for all $x \in X$. Using a similar argument, we have $B_\theta(x) \geq 1$ for all $x \in X_u$, concluding the first case.

For the second case $\kappa = 1$, notice that we have

$$\eta = \max_{x \in X_0} \max(B_\theta(x), 0) = \max_{x \in D_{X_0}} \max_{x' : \|x - x'\|_\infty \leq \epsilon} \max(B_\theta(x'), 0) = \eta^{(M)}. \tag{4.16}$$

By a similar argument, we have $c = c^{(M)}$ and thus $\mathcal{L} = \eta^{(M)} + c^{(M)}K = \eta + cK$, concluding the proof. □

4.4. Empirical Evaluation

The framework is evaluated on three benchmarks: a 2-D linear system from [15], the 2-D polynomial system from [18], and a 3-D non-polynomial Dubin's car model [111]. We will describe the benchmarks in greater detail in Subsection 4.4.1. To show the flexibility of our framework, for all systems we consider the same NBF architecture: a feed-forward neural network with 3 hidden layers with 128 neurons per layer and ReLU activation. Experiments are conducted on a computer with a Intel i7-6700k CPU, 16GB DDR4 RAM, Nvidia GTX 1060 GPU[1]. The optimization is

[1]Code for both NBF, Lipschitz certification and SoS is available under GNU GPLv3 license at https://github.com/DAI-Lab-HERALD/neural-barrier-functions.

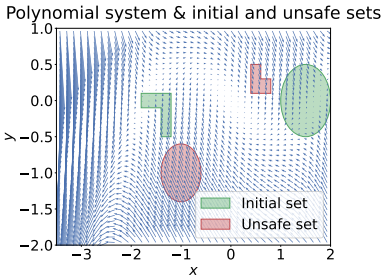


Figure 4.2: The nominal dynamics, i.e. of the vector field without noise, of the polynomial system adapted from [114] and the initial set and unsafe regions of the benchmarks.

done using ADAM with a learning rate of $1e{-}4$, with training batches of size $M = 50$. We train the NBF for 60000 iterations. As a baseline, we compare with a Sum-of-Squares (SoS)-based synthesis approach [15], [99] and a Lipschitz argument-based verification approach of NBFs [112].

4.4.1. BENCHMARKS

Linear dynamics We adopt the linear, discrete-time system from [15], which represents population dynamics with a juvenile and an adult stage [113]. The system is governed following stochastic difference equation

$$\mathbf{x}[k+1] = \underbrace{\begin{bmatrix} 0 & \psi_3 \\ \psi_1 & \psi_2 \end{bmatrix}}_{A} \mathbf{x}[k] + \begin{bmatrix} 0 \\ \mathbf{v}[k] \end{bmatrix} \tag{4.17}$$

We choose parameters $\psi_1 = 0.3$, $\psi_2 = 0.8$, and $\psi_3 = 0.4$, and $p_{\mathbf{v}} = \mathcal{N}(\cdot \mid 0, 0.1)$. Note that these parameter values differ from [15], since the parameters in [15] are unstable. Namely, with the values $\psi_1 = 0.5$, $\psi_2 = 0.95$, and $\psi_3 = 0.5$ from [15], the basic reproduction number is $R = \frac{\psi_1 \psi_3}{1-\psi_2} = 5 > 1$, meaning that the population always increases and the origin is an unstable equilibrium [113]. For our choice, the basic reproduction number is $R = 0.6$. The seemingly safe results reported [15] are the consequence of a discrepancy between the stochastic difference equation in the paper and the associated code.

We define the state space $X \subset \mathbb{R}^2$, initial set X_0, and safe set X_s as follows with $X_u = X \setminus X_s$:

$$X = [-3, 3]^2, \tag{4.18}$$
$$X_0 = \{x \in X : x_1^2 + x_2^2 \leq 1.5^2\}, \text{ and} \tag{4.19}$$
$$X_s = \{x \in X : x_1^2 + x_2^2 \leq 2^2\}. \tag{4.20}$$

Polynomial model For a polynomial system, we adapt a 2-D system from [114, Example 8.6] by discretizing time using an Euler integrator and adding noise. Due

to the discretization, letting h denote the step size, the time horizon is $K \cdot h$ with a step horizon K. We choose a step size $h = 0.1$.

$$\begin{aligned}\mathbf{x}[k+1]_1 &= \mathbf{x}[k]_1 + h \cdot \mathbf{x}[k]_2 + \mathbf{v}[k] \\ \mathbf{x}[k+1]_2 &= \mathbf{x}[k]_2 + h \cdot \left(\frac{1}{3}\mathbf{x}[k]_1^3 - \mathbf{x}[k]_1 - \mathbf{x}[k]_2\right)\end{aligned} \tag{4.21}$$

where $p_{\mathbf{v}} = \mathcal{N}(\cdot \mid 0, 0.01)$. The state space $X \subset \mathbb{R}^2$, initial set X_0, and unsafe set X_u are defined as follows with $X_s = X \setminus X_u$

$$X = \{x \in \mathbb{R}^2 : -3.5 \leq x_1 \leq 2 \text{ and } -2 \leq x_2 \leq 1\}, \tag{4.22}$$

$$\begin{aligned}R^1_{X_0} &= \{x \in X : (x_1 + 1.5)^2 + x_2^2 \leq 0.5^2\}, \\ R^2_{X_0} &= \{x \in X : -1.8 \leq x_1 \leq -1.2 \text{ and } -0.1 \leq x_2 \leq 0.1\}, \\ R^3_{X_0} &= \{x \in X : -1.4 \leq x_1 \leq -1.2 \text{ and } -0.5 \leq x_2 \leq 0.1\}, \\ X_0 &= R^1_{X_0} \cup R^2_{X_0} \cup R^3_{X_0},\end{aligned} \tag{4.23}$$

$$\begin{aligned}R^1_{X_u} &= \{x \in X : (x_1 + 1)^2 + (x_2 + 1)^2 \leq 0.4^2\}, \\ R^2_{X_u} &= \{x \in X : 0.4 \leq x_1 \leq 0.6 \text{ and } 0.1 \leq x_2 \leq 0.5\}, \\ R^3_{X_u} &= \{x \in X : 0.4 \leq x_1 \leq 0.8 \text{ and } 0.1 \leq x_2 \leq 0.3\}, \\ X_u &= R^1_{X_u} \cup R^2_{X_u} \cup R^3_{X_u}.\end{aligned} \tag{4.24}$$

Notably, for this benchmark, both the initial and unsafe sets consist of two disjoint regions, which is shown in Figure 4.2 with the nominal dynamics.

Dubin's car System **barr**$_4$ from [111], also known as Dubin's car, is a non-polynomial system. The state of the system is the position in a plane and the heading of the vehicle. We adapt the system from continuous-time deterministic to discrete-time stochastic by discretizing time with an Euler integrator and adding noise to the heading. Let h denote the step size, that is, $K \cdot h$ is the time horizon. Then, dynamics are governed by

$$\begin{aligned}\mathbf{x}[k+1]_1 &= \mathbf{x}[k]_1 + h \cdot v \sin(\mathbf{x}[k]_3) \\ \mathbf{x}[k+1]_2 &= \mathbf{x}[k]_2 + h \cdot v \cos(\mathbf{x}[k]_3) \\ \mathbf{x}[k+1]_3 &= \mathbf{x}[k]_3 + h \cdot u + \mathbf{v}[k]\end{aligned} \tag{4.25}$$

where $p_{\mathbf{v}} = \mathcal{N}(\cdot \mid 0, 0.01)$, v is the velocity, and u denotes the steering angle. We choose a steering angle $u = 1 / 0.95$ such that the vehicle drives in a clockwise circle with a radius equal to the distance between the origin and the starting position (see below). We also choose velocity $v = 1$ and step size $h = 0.1$. The state space $X \subset \mathbb{R}^3$, initial set X_0, and safe set X_s are defined as follows with $X_u = X \setminus X_s$

$$X = \{x \in \mathbb{R}^3 : x_1^2 \leq 4, x_2^2 \leq 4, \text{ and } x_3^2 \leq \pi^2/4\} \tag{4.26}$$

$$X_0 = \{(-0.95, 0, 0)\} \tag{4.27}$$

$$X_s = \{x \in X : x_1^2 \leq 1.9^2 \text{ and } x_2^2 \leq 1.9^2\} \tag{4.28}$$

Table 4.1: Certified lower bound for P_{safe}. Higher is better, and the best result for each system is highlighted in **bold**. Cells with "-" denotes that SoS failed to compute a barrier. BaB abbreviates branch-and-bound.

Method	Linear	2-D polynomial	Dubin's car
SoS (4) [15]	0.690906	0.000000	-
SoS (8)	0.975079	0.232710	-
SoS (13)	0.998405	0.681383	-
SoS (15)	0.999761	-	-
NBF (grid + Lipschitz) [112]	0.824654	0.792392	0
NBF (grid + CROWN)	0	0	0
NBF (BaB + CROWN)	**0.999969**	**0.991664**	**0.870272**

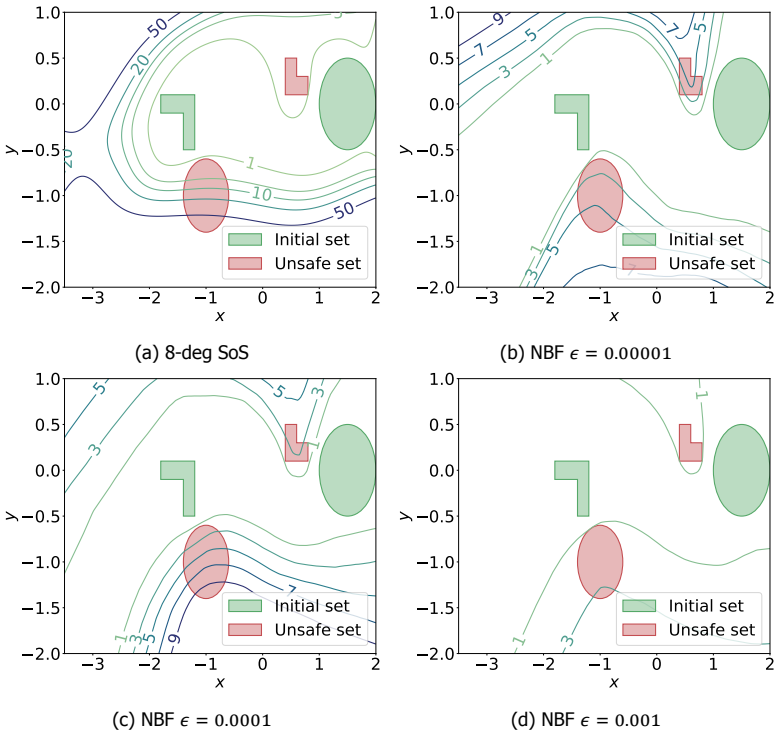
To encode the non-polynomial dynamics into a polynomial suitable for SoS, we use the method of [99] where the state space is partitioned into a grid and linear relaxations of the nominal dynamics are encoded with SoS Lagrange multipliers using Putinar's Positivstellensatz [115].

4.4.2. RESULTS

In Table 4.1, we report a lower bound of P_{safe} obtained with our approach, a SoSP-based approach [15], and a Lipschitz method adapted from [112]. We also report the verification of NBFs with CROWN and a grid-based partitioning as an ablation study for the impact of the branch-and-bound algorithm. The grid considered for both the linear and polynomial system is 320x320, and for Dubin's car it is 40x40x60. For the SoSPs-based approach, we consider polynomials of order up to 15 on all benchmarks.

For all benchmarks it is possible to observe that our approach based on NBFs outperforms both SoS optimization and Lipschitz certification in terms of the tightness of the bounds. For instance, for the Dubin's car model, arguably the hardest example we consider due to its non-polynomial nature, SoS fails due to excessive memory requirements and Lipschitz only finds a trivial certificate of 0, while our framework obtains a lower bound of 0.87. In contrast, for the linear system, both SoS and our approach obtain a similar certified level of safety, but SoS is substantially faster (orders of minutes for the linear system), as our method requires to first train a neural network and then certify it (orders of few hours for all benchmarks as we use the same neural network architecture).

To understand the difference in certified safety, in Figure 4.3 we study contour plots of the barrier functions for the 2-D polynomial example obtained with SoS and with NBF for different values of ϵ, where ϵ is the training parameter introduced in Section 4.3. Note that the certified lower bound for P_{safe} via Proposition 3.1 can be non-zero only in regions where $B(x) < 1$. Interestingly, we observe this region is significantly smaller for SoS compared to NBF for all ϵ, which is attributed to the reduced expressivity of the small-degree SoS polynomial. The differences in region sizes and the distances to the initial sets explain the result in Table 4.1. Furthermore,



(a) 8-deg SoS

(b) NBF $\epsilon = 0.00001$

(c) NBF $\epsilon = 0.0001$

(d) NBF $\epsilon = 0.001$

Figure 4.3: Levelset for **(a)** a SoS SBF and **(b, c, d)** NBFs with varying values of ϵ, all for the 2-D polynomial system. The NBFs are more flexible to capture complex shapes of the unsafe set and less sharply increasing in value compared to the SoS SBF. **(b, c, d)** show that larger ϵ results in a flatter surface, yielding a smaller c at the expense of a larger η.

we should stress that $\epsilon = 0.00001$ is the smaller value of ϵ for which our approach could find a valid barrier, illustrating the importance of this parameters in balancing between finding a valid barrier and obtaining high lower bounds of safety.

4.5. Conclusion

This chapter addressed the challenge of synthesizing Stochastic Barrier Function (SBF) certificates to certify probabilistic safety for stochastic, non-linear, discrete-time systems. Traditional methods based on Sum-of-Squares (SoS) programming have a key limitation: Scalability – SoS programs rapidly become intractable as the dimensionality of the system or the degree of the polynomial increases. To overcome this limitation, we introduced Neural Barrier Functions (NBFs), parameterized as feedforward neural networks. Neural networks provide a flexible functional class capable of approximating a wide range of barrier shapes beyond polynomials. Although NBFs address the scalability issues associated with SoS-based SBFs, their introduction necessitate *post hoc* verification, which carries its own set of challenges.

To overcome these issues, this chapter introduces two methods: neural network verification via CROWN, extended to random variables, and a branch-and-bound scheme. CROWN is a state-of-the-art neural network verification method based on linear relaxations of non-linear functions that has repeatedly won the Verification of Neural Network Competition [90]. However, until this work, it has been restricted to deterministic functions; we extend it to also cover random variables, to allow bounding the expectation in the martingale condition (3.1c). Since CROWN relaxes a non-linear problem into a linear one, it necessarily introduces conservatism. Although grid-based partitioning reduces the conservatism, it wastes computational resources, as for some regions a coarse partitioning suffices and while others require finer partitioning. Thus, to carefully and intelligently control the conservatism, we introduce a branch-and-bound scheme, which selects regions to split further based on the relaxation gap.

Experimental evidence shows that NBFs provide tighter safety probabilities than SoS-based SBFs, up to the computational limits. We also performed an ablation study, replacing the branch-and-bound scheme with grid-based partitioning, to highlight the importance of the branch-and-bound scheme.

A key insight from applying NBFs for safety verification is that the neural parameterization significantly expands the space of candidate barrier functions, allowing the certification of safety for dynamics that are too complex for traditional polynomial templates. However, while NBFs scale better than SoS methods, high-dimensional systems still pose challenges. The branch-and-bound approach mitigates this, but the combinatorial growth of partitions imposes practical limits; this is the curse of partitioning, a corollary to the curse of dimensionality [116], which is a pervasive challenge in safety-critical control and formal verification.

Another limitation of this approach is that it requires complete knowledge of the dynamics, including the noise distribution. In practice, this rarely holds, and thus, later in Chapter 6, we address this limitation using data-driven methods.

5

Piecewise Constant Stochastic Barrier Functions

Mathematics is a game played according to certain simple rules with meaningless marks on paper.

David Hilbert

This chapter is based on *R. Mazouz, F. B. Mathiesen, L. Laurenti,* et al., Piecewise stochastic barrier functions*, Accepted for Automatica, 2024. arXiv: 2404.16986 [cs.RO]* [76] (Sections 5.2-5.5) and *R. Mazouz, J. Skovbekk, F. B. Mathiesen,* et al., *"Data-driven permissible safe control with barrier certificates", in* IEEE 63rd Conference on Decision and Control (CDC)*, IEEE, 2024, pp. 6844–6849.* ***DOI****: 10.1109/CDC56724.2024.10886850* [77] (Section 5.6) with minor modifications to clarify assumptions and streamline the presentation. Rayan Mazouz and Frederik Baymler Mathiesen are joint first-authors in [76]. In [77], Frederik contributed with *Software* and *Writing - original draft* (CRediT taxonomy).

5.1. Introduction

In Chapter 4, we demonstrated that neural SBFs offer significant scalability advantages over the previous state-of-the-art, SoS-based SBF synthesis. However, verifying NBFs is very challenging and requires discretization and convex relaxations. The need for discretization inevitably leads to the question: *can the synthesis be improved by bypassing the neural network and directly employing a discretized structure?* To study this question, in this chapter, we focus on piecewise SBFs (PW-SBFs). By allowing the barrier function to change behaviour across different regions of the state space, piecewise formulations provide a natural[1] and computationally efficient way to approximate complex, possibly discontinuous safety boundaries. Furthermore, the continuous nature of NNs makes it difficult to capture sharp transitions or discontinuities that may naturally arise in the safety landscape of many systems [118]; a limitation that does not exist for for piecewise templates. We highlight the impact of this limitation in Figure 5.1 where we empirically show that both SoS-SBFs and NBFs suffer due to the continuity restriction. In the presented example, the unsafe set is a disjoint union of two *small* sets, of which one is situated close to the initial set. For a good safety guarantee, the barrier requires a rapid change in value near the unsafe sets, which SoS-SBFs and NBFs fail to capture. We further study the simplest class of PW-SBFs: piecewise constant SBFs (PWC-SBFs). In this chapter, we show that restricting to this subclass reduces the synthesis problem to an LP problem, thus enabling greater scalability and flexibility in algorithmic design.

Formally, we consider the following problem:

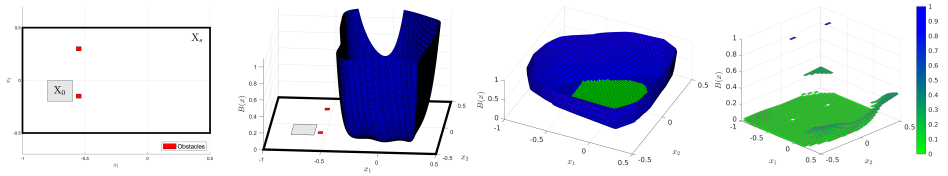
Problem 5.1

Given a system $\mathbf{x}[k+1] = f(\mathbf{x}[k]) + \mathbf{v}[k]$ where $\mathbf{v}[k]$ is a random variable with Gaussian distribution $p_{\mathbf{v}}$ (i.e., of the form (2.34)), an initial set X_0, a safe set X_s, and a time horizon K, compute a lower bound ρ on P_{safe}.

As will be shown in Section 5.3, specializing the notion of PW-SBFs to PWC-SBFs allows studying convergence in the asymptotic limit. In the following Chapter 6, we study piecewise affine SBFs (PWA-SBFs), which is another specialization of PW-SBFs, in the context of data-driven synthesis. A deeper theoretical analysis of PWC-SBFs uncovers strong connections to IMDPs, which will be the focus of Chapter 9. In Section 5.4 we show that, due to their structural simplicity, PWC-SBFs deliver considerable computational improvements compared to synthesis approaches based on SoS and NBF. The section presents three distinct methods for synthesizing PWC-SBF: Dual LP, LP-based CEGIS, and Gradient Descent (GD). These methods are listed here in decreasing order of computation time and memory requirements. The computational improvements are, however, at the cost of increasingly many hyperparameters that must be tuned for a given application to achieve optimal performance.

In Section 5.5, we systematically compare all three piecewise constant (PWC)

[1] At least to an engineer. Piecewise (constant) functions are pervasive in the field of engineering [117].



(a) Initial and safe sets (b) Degree-30 SoS-SBF (c) NBF (d) PWC-SBF

Figure 5.1: Example of a 2D stochastic system with a non-convex safe set and linear dynamics $\mathbf{x}[k+1] = 0.5\mathbf{x}[k] + \mathbf{v}[k]$, where noise $\mathbf{v}[k]$ follows a Gaussian distribution with diagonal covariance. The obtained safety probability, for $K = 10$, and computation time for SoS are $P_{\text{safe,SoS}} = 0.075$ and $\tau_{\text{SoS}} = 197$s, for NBF are $P_{\text{safe,NBF}} = 0.93$ and $\tau_{\text{NBF}} = 3600$s, and for PWC-SBF are $P_{\text{safe,PWC}} = 0.93$ and $\tau_{\text{PWC}} = 69$s.

synthesis methods against both neural and SoS-based SBFs across a diverse set of systems. Through these comparisons, we highlight the trade-offs in computation time, memory requirements, conservatism, and ease of hyperparameter tuning. Finally, in Section 5.6, we extend the study of PWC-SBFs to address the problem of control synthesis. Specifically, we demonstrate how PWC-SBFs can be used to construct (probabilistic) forward invariant sets, providing a foundation for safe control policy design under stochastic system dynamics.

5.2. PIECEWISE STOCHASTIC BARRIER FUNCTIONS

In this section, we introduce the general notion of PW-SBFs. With the change from continuous to piecewise continuous templates, it is necessary to consider the well-definedness of the SBF, in particular wrt. the expectation in Definition 3.1. We start by defining piecewise barrier templates. To this end, consider a partition $Q = \{q_1, \dots, q_\ell\}$ of the compact safe set X_s in ℓ compact sets, i.e., $\bigcup_{i=1}^{\ell} q_i = X_s$ and $q_i \cap q_j$ has zero Lebesgue measure for any two $i \neq j \in \{1, \dots, \ell\}$, such that vector field f is continuous in each region q_i. Assume that the boundary of each q_i has measure zero wrt. the transition kernel $T(\cdot \mid x)$ for any $x \in X_s$. Furthermore, for every $i \in \{1, \dots, \ell\}$, let $B_i : \mathbb{R}^n \to \mathbb{R}_{\geq 0}$ be an piecewise continuous function such that B_i is continuous on q_i and $B_i(x) = 0$ if $x \notin q_i$. Finally, define the piecewise constant function $B_u(x) = 1$ if $x \in X_u$ and zero otherwise. Then, a piecewise barrier function (template) is defined as

$$B(x) = \max\{B_u(x), B_1(x), \dots, B_\ell(x)\}. \tag{5.1}$$

Note that by construction B is upper semi-continuous. That is, at every point $x_0 \in X$ and for every real $\overline{B} > B(x_0)$ there exists a neighbourhood U of x_0 such that $\overline{B} > B(x)$ for all $x \in U$. This is important, as upper semi-continuity implies that B is Borel measurable. In conjunction with a general system $\mathbf{x}[k+1] = f(\mathbf{x}[k], \mathbf{v}[k])$ (i.e., of the form (2.26)), the random variable $B(f(x, \mathbf{v}))$ is well-defined, since B is Borel measurable and $f(x, \cdot)$ is assumed to be measurable for any $x \in X_s$. Thus, the expectation in (3.1c) is well-defined; as is the theory of SBFs.

To be able to decompose the expectation in (3.1c) into a summation over *closed* sets (barrier pieces), we require the following assumption regarding boundaries.

We write assumption in terms of a general (additive) noise distribution for it to be applicable in the subsequent Chapter 6.

Assumption 5.1. *For an additive noise system* $\mathbf{x}[k+1] = f(\mathbf{x}[k]) + \mathbf{v}[k]$ *with noise distribution* $p_{\mathbf{v}}$*, we assume that almost surely* $\mathbb{P}_{\mathbf{v}}[\mathbf{x}' \in q_i \cap q_j] = 0$ *where* $\mathbf{x}' = f(x) + \mathbf{v}$*, for any two* $i \neq j \in \{1, \dots, \ell\}$ *and any* $x \in X_s$*.*

The assumption holds for Gaussian noise distributions because (i) the distribution $p_{\mathbf{v}}$ is continuous and (ii) the intersections have zero Lebesgue measure.

Definition 5.1 (Piecewise SBF)**.** *Let* $X_s \subset \mathbb{R}^n$*,* $X_0 \subseteq X_s$ *and* $X_u = \mathbb{R}^n \setminus X_s$ *be, respectively, the safe set, the set of initial states, and the unsafe set. Furthermore, let the system* $\mathbf{x}[k]$ *be governed by* (2.34) *with a Gaussian noise distribution* $p_{\mathbf{v}}$*. Then the function* B *in* (5.1) *is a PW-SBF for* $\mathbf{x}[k]$ *if for all* $i \in \{1, \dots, \ell\}$ *there exist scalars* $c_i, \eta \geq 0$ *such that the following conditions hold*

$$B_i(x) \leq \eta \qquad \forall x \in q_i \cap X_0, \tag{5.2a}$$

$$\sum_{j=1}^{\ell} \mathbb{E}[B_j(\mathbf{x}') \mid \mathbf{x}' \in q_j] \cdot T(q_j \mid x) + T(X_u \mid x) \leq B_i(x) + c_i \qquad \forall x \in q_i, \tag{5.2b}$$

where $\mathbf{x}' = f(x) + \mathbf{v}$ *and* T *is the transition kernel for* (2.34)*.*

The following corollary specializes Proposition 3.1 for lower bounding the probability of safety P_{safe} to PW-SBFs.

Corollary 5.1. *Let* B *be an PW-SBF for* $\mathbf{x}[k]$*. Then, it holds that*

$$P_{\text{safe}} \geq 1 - (\eta + K \cdot \max\{c_i\}_{i=1}^{\ell}). \tag{5.3}$$

Proof. The proof is based on Proposition 3.1. First, by construction the barrier is non-negative. Next, it is clear that a piecewise candidate function B satisfying (5.2a) also satisfies (3.1b). Finally, it is enough to show that $\max_{i \in \{1,\dots,\ell\}} c_i$ from (5.2b) upper bounds the expression $\mathbb{E}[B(f(x, \mathbf{v}))] - B(x)$ in (3.1c). To this end fix any q_i and $x \in q_i$. Then, we have

$$\mathbb{E}[B(\mathbf{x}')] = \sum_{j=1}^{\ell} \mathbb{E}[B_j(\mathbf{x}') \mid \mathbf{x}' \in q_j] \cdot T(q_j \mid x) + T(X_u \mid x) \leq B_i(x) + c_i. \tag{5.4}$$

The equality holds by the law of total expectation and the inequality holds by (5.2b). Since (5.4) holds for every $x \in X_s = \cup_{i=1}^{\ell} q_i$, the expression $\mathbb{E}[B(f(x, \mathbf{v}))] - B(x)$ is upper bounded by $c = \max\{c_i\}_{i=1}^{\ell}$. □

The benefit of the formulation in Corollary 5.1 is that both the size of the partition and class of functions B_is (e.g., linear, polynomial, exponential, etc.) are design parameters. This provides flexibility for the SBF to fit different shapes of X_s. This flexibility, however, may introduce challenges as it can lead to a non-convex

optimization problem, even for simple choices for B_i such as polynomial or linear functions. We introduce a set of simple but effective choices that lend themselves to efficient computational tools that outperform the state-of-the-art SBF synthesis techniques.

A major difficulty in the optimization problem is a consequence of the product of the expectation term and the transition kernel function in (5.2b), namely, $\mathbb{E}[B_j(\mathbf{x}') \mid \mathbf{x}' \in q_j] \cdot T(q_j \mid x)$. The first term not only requires an expectation operation, but also a composition of B_j with the non-linear function f. The second term T is a non-linear function of x that involves an integral of probability density function $p_{\mathbf{v}}$. To reduce complexity, one can choose to use constant values for B_is, which reduces the expectation to a summation and avoids the need for composition of non-linear functions. This leads to a PWC-SBF. Furthermore, while the analytical form of $\mathbb{P}_{\mathbf{v}}$ may be hard to obtain, bounds can be efficiently computed using, e.g., the procedure in [63], for general f and non-standard $p_{\mathbf{v}}$ (e.g., non-symmetric, non-unimodal, etc.). In the next section, we detail the optimization problem for these choices.

5.3. PIECEWISE *CONSTANT* STOCHASTIC BARRIER FUNCTIONS

In this section, we formally define an optimization problem for synthesis of PWC-SBFs and study its convergence in the asymptotic limit to the optimal SBF.

For $i, j \in \{1, \dots, \ell\}$, let $\underline{\gamma}_{ij}, \overline{\gamma}_{ij} \in [0,1]$ denote the lower and upper bounds of the transition kernel $T(q_j \mid x)$ for every $x \in q_i$, respectively, i.e.,

$$\underline{\gamma}_{ij} \leq T(q_j \mid x) \leq \overline{\gamma}_{ij} \qquad \forall x \in q_i. \tag{5.5}$$

Similarly, we use $\underline{\gamma}_{iu}, \overline{\gamma}_{iu} \in [0,1]$ for the bounds of T to the unsafe set X_u, i.e.,

$$\underline{\gamma}_{iu} \leq T(X_u \mid x) \leq \overline{\gamma}_{iu} \qquad \forall x \in q_i. \tag{5.6}$$

Note that these bounds can be computed efficiently for general f and $p_{\mathbf{v}}$ using, e.g., the techniques in [16], [40], [63], [119]. We define the set of all feasible values for the transition kernel for all $x \in q_i$ as

$$\Gamma_i = \left\{ \gamma_i \in [0,1]^{\ell+1} : \underline{\gamma}_{ij} \leq \gamma_{ij} \leq \overline{\gamma}_{ij}, \forall j \in \{1, \dots, \ell, u\}, \sum_{j=1}^{\ell} \gamma_{ij} + \gamma_{iu} = 1 \right\}. \tag{5.7}$$

Note that Γ_i is a convex polytopic set, specifically the intersection between a hyper-rectangle and the probability simplex.

Remark 5.1 (Interpretation as an IMDP)**.** *The partitioning of the state space and the computation interval bounds for the transition kernel resembles the abstraction process to Interval Markov Decision Process in Section 7. In fact, they are equivalent with each region q_i corresponding to an abstract state and the transition*

ambiguity set Γ_i for each abstract state. As a consequence, PWC-SBFs can also be synthesized for IMDPs. Furthermore, it hints at proving the theory of SBFs using dynamic programming as we show in Lemma 3.1 with insights into the sources of conservatism.

The following theorem sets up an optimization problem for synthesis of a PWC-SBF.

Theorem 5.1 (PWC-SBF Synthesis)**.** *Given a ℓ-partition of X_s, let $\theta \in \Theta$ be the parameterization of piecewise constant functions in the form of (5.1). In particular, let $B_i(x) = w_i$ with $w \in \mathbb{R}_{\geq 0}$ for every $i \in \{1, \dots, \ell\}$ such that $\theta = (w_1, \dots, w_\ell)$ and $\Theta = \mathbb{R}^n_{\geq 0}$. Next, define $\Gamma = \Gamma_1 \times \cdots \times \Gamma_\ell$, where each Γ_i is the set of probability distributions defined in (5.7). Then, $B_{\theta^\star}$ is a PWC-SBF that maximizes P_{safe} if $\theta^\star$ is a solution to the following optimization problem:*

$$\theta^\star = \operatorname*{argmin}_{\theta \in \Theta} \max_{\{\gamma_i\}_{i=1}^{\ell} \in \Gamma} \quad \eta + cK \tag{5.8a}$$

$$\text{s.t.} \quad w_i \geq 0 \qquad \forall i \in \{1, \dots, \ell\}, \tag{5.8b}$$

$$w_i \leq \eta \qquad \forall i : q_i \cap X_0 \neq \emptyset, \tag{5.8c}$$

$$\sum_{j=1}^{K} w_j \cdot \gamma_{ij} + \gamma_{iu} \leq w_i + c_i \qquad \forall i \in \{1, \dots, \ell\}, \tag{5.8d}$$

$$0 \leq c_i \leq c \qquad \forall i \in \{1, \dots, \ell\}. \tag{5.8e}$$

Proof. It suffices to show that if for all $i \in \{1, \dots, \ell\}$, the function $B_i(x)$ satisfy (5.8b)-(5.8d), then $B(x)$ as defined in (5.1) satisfies the conditions in Corollary 5.1. The optimization problem in (5.8a) aims to maximize the safety probability over feasible values for the transition kernel Γ, which is expressed as a min-max problem. Each c_i is bounded according to (5.8d), and by Corollary 5.1, $\{c_i\}_{i=1}^{\ell}$ upper bounds $\mathbb{E}[B(\mathbf{x}')] - B(x)$. Hence, optimizing over objective $B_{\theta^\star}$ also maximizes the lower bound on safety probability P_{safe}. □

Given a partition of X_s, Theorem 5.1 provides a concrete optimization problem to synthesize a PWC-SBF that optimizes a lower bound on P_{safe}. This theorem immediately gives rise to questions on optimality (wrt. general SBFs) and computability. We address the computability question in Section 5.4. For optimality, the following proposition establishes that, in the limit of partition size, the problem in (5.8a) converges to the optimal safety probability bound ρ obtainable with measurable SBFs, and the corresponding optimal PWC-SBF to certify the bound.

Proposition 5.1. *Let ρ denote the optimal safety probability bound obtained from Proposition 3.1 from the class of measurable SBFs, i.e., the set of measurable functions satisfying the conditions in Definition 3.1. Consider a uniform partition of X_s into ℓ compact sets and let $B^\ell : X \to \mathbb{R}_{\geq 0}$ be the optimal PWC-SBF obtained from solving (5.8a). Denote by $(\eta + cK)^\star_\ell$ the resulting optimal safety probability bound. Then, it holds that*

$$\rho = \lim_{\ell \to \infty} (1 - (\eta + cK)^\star_\ell).$$

Proof. First, note that B^ℓ is Borel measurable for any $\ell \in \mathbb{N}$ (see Section 5.2), thus, belongs to the class of measurable SBFs. Next, assume that a measurable function $\hat{B} : X \to \mathbb{R}_{\geq 0}$ is the optimal barrier according to (3.10). We need to show that we necessarily have $\hat{B}(x) = 1$ for all $x \in X_u$. In order to do that note that

$$\mathbb{E}[\hat{B}(f(x,\mathbf{v}))] = \int_{v:f(x,v)\in X_s} \hat{B}(f(x,v))\, d\mathbb{P}_{\mathbf{v}}(v) + \int_{v:f(x,v)\in X_u} \hat{B}(f(x,v))\, d\mathbb{P}_{\mathbf{v}}(v). \quad (5.9)$$

Using this fact, we can rewrite (3.1c), with $\hat{B}$, as the fact that for all $x \in X_s$ it must hold that

$$\int_{v:f(x,v)\in X_s} \hat{B}(f(x,v))\, d\mathbb{P}_{\mathbf{v}}(v) + \int_{v:f(x,v)\in X_u} \hat{B}(f(x,v))\, d\mathbb{P}_{\mathbf{v}}(v) \leq \hat{B}(x) + c. \quad (5.10)$$

As for $x \in X_u$, $\hat{B}(x)$ only appear on the left hand side of the inequality; hence, the value of c required to satisfy the inequality is minimized by taking the smallest possible value of $\hat{B}(x)$ for $x \in X_u$. Because of Condition (3.1a), the smallest possible value is 1. Therefore, we also have $\hat{B}(x) = B^\ell(x)$ for all $x \in X_u$.

For the safe set X_s, since $\hat{B}$ is non-negative, by [120, Theorem 3.2.14], there exists a sequence of non-negative PWC functions B^ℓ such that $\lim_{\ell\to\infty} B^\ell(x) = \hat{B}(x)$ for all $x \in X_s$. As a consequence, we have $\lim_{\ell\to\infty} 1-(\eta+cK)^\star_\ell = 1-(\eta^\star+c^\star K) = \rho$ where $\eta^\star$ and $c^\star$ are the optimal values for (3.10) with $\hat{B}$. □

This proposition shows that, despite their simplicity, PWC-SBFs are as expressive as more complex forms of SBFs such as polynomials and NBFs. That is, PWC-SBFs can bound the safety probability *at least* as well as any continuous SBFs. In fact, our evaluations in Section 5.5 clearly demonstrate this point. Further, we also note that, from the proof of Proposition 5.1, it becomes evident that the choice of $B(x) = 1$ for all $x \in X_u$ in (5.1) is the optimal choice for PW-SBFs. Lastly, it is worth mentioning that a rigorous sub-optimality analysis of the PWC-SBF for a given finite partition is non-trivial and remains an open question.

5.4. SYNTHESIS METHODS FOR PIECEWISE CONSTANT STOCHASTIC BARRIER FUNCTIONS

A roadblock in solving the optimization problem in Theorem 5.1 is in Condition (5.8d), which includes products of decision variables w_j and γ_{ij}. As a consequence, the minimax optimization problem in Theorem 5.1 is bilinear, meaning that it is linear if either w_j or γ_{ij} are fixed. This class of problems is generally non-convex; hence, convex solvers, which provide efficiency and guaranteed convergence, cannot be utilized. In this section, we propose three approaches to (losslessly) convexify the problem, namely an LP duality-based approach, a LP-CEGIS procedure, and a GD-based method such that the optimal solution can be efficiently computed. They each solve the problem in Theorem 5.1, but with different solver properties including computational effort, scalability, and required parameter tuning. The dual LP approach allows the use of standard LP solvers to exactly compute the optimal solution, but the introduction of dual variables limits scalability. LP-CEGIS mitigates

the scalability problem by decoupling the inner maximization and outer minimization via iterative barrier candidate and counter-example generation, but may still require prohibitive amounts of memory. To address this problem, GD is presented, which utilizes sparsity and provides efficient gradient computation. However, GD requires tweaking of hyperparameters to achieve good convergence in practice, due to the non-smoothness of the objective function. We first present the dual LP approach.

5.4.1. Dual Linear Programming

Using strong duality for LP problems (Proposition 2.5), we first introduce an exact approach that encodes the optimization problem in Theorem 5.1 as a linear program, with the inner maximization is replaced by its dual minimization. We begin by observing that for each $i \in \{1, \dots, \ell\}$, the set of feasible transition kernels Γ_i in (5.7) is a polytope. Hence, it can be represented by an intersection of halfspaces as

$$\Gamma_i = \{\gamma_i : H_i \gamma_i \leq h_i\}, \tag{5.11}$$

5

where the matrix $H_i \in \mathbb{R}^{2(\ell+1)\times(\ell+1)}$ and the vector $h_i \in \mathbb{R}^{2(\ell+1)}$ are defined by the constraints in (5.7), and $\leq$ is interpreted elementwise. Next, we define the vector $\bar{w} = (w, 1)$, where $w \in \mathbb{R}^{\ell}_{\geq 0}$ is the variable value in each region, and the dual variable $\lambda_i \in \mathbb{R}^{2(\ell+1)}_{\geq 0}$. Then, Condition (5.8d) can be rewritten as two constraints:

$$h_i^\top \lambda_i \leq w_i + c_i, \tag{5.12a}$$
$$H_i^\top \lambda_i = \bar{w}. \tag{5.12b}$$

Since both constraints are linear in the decision variables λ_i, w, and c_i, the optimization problem in Theorem 5.1 can be written as an LP, which we formalize in Theorem 5.2.

Theorem 5.2 (PWC-SBF Dual LP)**.** *Let* $(w^{\star 1}, c^{\star 1}, \eta^{\star 1}, \lambda_1^\star, \dots, \lambda_\ell^\star)$ *be the optimal solution to the following LP*

$$\begin{aligned}
\min \quad & \eta + cK, && && (5.13a)\\
\text{s.t.} \quad & 0 \leq w_i \leq 1 && \forall i \in \{1, \dots, \ell\}, && (5.13b)\\
& b_i \leq \eta && \forall i : q_i \cap X_0 \neq \emptyset, && (5.13c)\\
& h_i^\top \lambda_i \leq w_i + c_i && \forall i \in \{1, \dots, \ell\}, && (5.13d)\\
& H_i^\top \lambda_i = \bar{b} && \forall i \in \{1, \dots, \ell\}, && (5.13e)\\
& \lambda_i \geq 0 && \forall i \in \{1, \dots, \ell\}, && (5.13f)\\
& 0 \leq c_i \leq c && \forall i \in \{1, \dots, \ell\}. && (5.13g)
\end{aligned}$$

Furthermore, let $(w^{\star 2}, c^{\star 2}, \eta^{\star 2}, \gamma_1^\star, \dots, \gamma_\ell^\star)$ *be the optimal solution to the optimization problem in Theorem 5.1. Then, the two solutions concide on the SBF decision variables* w, c, η*. That is,* $(w^{\star 1}, c^{\star 1}, \eta^{\star 1}) = (w^{\star 2}, c^{\star 2}, \eta^{\star 2})$.

Theorem 5.2 follows as an application of Proposition 2.5 to the inner maximization problem of the optimization problem in Theorem 5.1. Hence, $B(x)$ constitutes a proper SBF, which guarantees probability of safety $P_{\text{safe}} \geq 1 - (\eta + cK)$.

Computational Complexity The time complexity of a standard primal linear program is $O(n^2 m)$ where n is the number of decision variables and m is the number of constraints [93]. The program in Theorem 5.2 has $n = 2\ell^2 + 2\ell$ decision variables and $m = \ell^2 + 6\ell + L + 1$ constraints where $L = |\{i : q_i \cap X_0 \neq \emptyset\}|$ is the number of regions intersecting with the initial set. Combining the complexity of a linear program with the number of decision variables and constraints for our dual linear program, we get $O(n^2 m) = O(\ell^6)$. For many applications, this is prohibitive. Hence, in the following subsections, we introduce two iterative approaches that lead to an improved runtime complexity.

5.4.2. LP-BASED COUNTER-EXAMPLE GUIDED INDUCTIVE SYNTHESIS

We introduce another PWC-SBF synthesis method that is exact and computationally more efficient than the dual LP approach. The method is based on splitting the minimax problem in Theorem 5.1 into two separate LP problems. One LP problem generates a PWC-SBF that maximizes the safety probability given a set of finite feasible distributions $\tilde{\Gamma}_i \subset \Gamma_i$. The other LP calculates local c-martingale variable c_i and generates distribution witnesses (counter-examples) that show suboptimality of the safety probability guarantee by the PWC-SBF. Then, the witnesses are added to $\tilde{\Gamma}_i$, and the process is repeated until no more counter-examples can be generated. We dub this method as LP-CEGIS for PWC-SBFs.

Algorithm 5.1 LP-CEGIS for PWC-SBFs

Require: Initial set X_0, partition $Q = \{q_i\}_{i=1}^{\ell}$, time horizon K, and feasible transition kernel sets $\{\Gamma_i\}_{i=1}^{\ell}$.
Ensure: Optimal PWC-SBF $B_{\theta^\star}$ and safety probability bound ρ.

$c^\star := 0$
for each i in $\{0, \dots, \ell\}$ **do**
    $\tilde{\Gamma}_i := \{\textbf{SAMPLEDIST}(\Gamma_i)\}$
    $c_i := 1$
while $c^\star < \max\{c_i\}_{i=1}^{\ell}$ **do**
    $\theta^\star, \eta^\star, c^\star := \textbf{PWCSYNTHLP}(X_0, Q, K, \{\tilde{\Gamma}_i\}_{i=1}^{\ell})$
    for each i in $\{0, \dots, \ell\}$ **do**
        $\gamma_i, c_i := \textbf{COUNTEREXAMPLELP}(B_{\theta^\star}, \Gamma_i, i)$
        $\tilde{\Gamma}_i := \tilde{\Gamma}_i \cup \{\gamma_i\}$ ▷ *Add counterexample*
return $B_{\theta^\star}$ and $\rho := 1 - (\eta^\star + c^\star K)$

The CEGIS algorithm is shown in Algorithm 5.1, which relies on subroutines in Algorithms 5.2 and 5.3. The main algorithm initializes the counter-example distribution sets $\tilde{\Gamma}_i$ by sampling any feasible distribution from each Γ_i. Then, based on $\tilde{\Gamma}_i$, it synthesizes an optimal PWC-SBF candidate $B_{\theta^\star}$ that maximizes the safety probability with its corresponding scalar $c^\star$ using the subroutine **PWCSYNTHLP**. In a traditional CEGIS setting, this corresponds to the learner. As shown in Algorithm 5.2, the **PWCSYNTHLP** subroutine is an LP problem that captures the min

component (outer minimization) of the minimax problem in Theorem 5.1.

Next, given $B_{\theta^\star}$, for each partition region q_i, using the subroutine **COUNTEREXAMPLELP**, the distribution γ_i that maximizes its corresponding martingale gap c_i is computed and then added to the set of witnesses $\tilde{\Gamma}_i$. As shown in Algorithm 5.3, **COUNTEREXAMPLELP** is also an LP problem with the added requirement that the solution is a vertex. This requirement can be enforced by solving the LP problem via O-maximization (see Section 8.4), which in addition solves the problem more efficiently than standard LP solvers. The **COUNTEREXAMPLELP** subroutine captures the max component (inner maximization) of the minimax problem, which corresponds to the verifier in a traditional CEGIS setting. If the obtained martingale gap c_i is greater than $c^\star$ for some region q_i, it means that the candidate $B_{\theta^\star}$ is not optimal and there exists at least a distribution that violates the probabilistic guarantee of the candidate. Hence, the process repeats with the updated witnesses until $c_i \leq c^\star$ for all $i \in \{0, \dots, \ell\}$.

Algorithm 5.2 Barrier candidate generation ("learner")

function PWCSYNTHLP$(X_0, Q, K, \{\tilde{\Gamma}_i\}_{i=1}^{\ell})$

$$w^\star, c^\star, \eta^\star := \underset{w,c,\eta}{\operatorname{argmin}} \quad \eta + cK,$$

$$\text{s.t.} \quad 0 \leq w_i \leq 1 \qquad \forall i \in \{1, \dots, \ell\},$$

$$w_i \leq \eta \qquad \forall i : q_i \cap X_0 \neq \emptyset,$$

$$\sum_{j=1}^{\ell} w_j \cdot \gamma_{ij} + \gamma_{iu} \leq w_i + c \qquad \forall i \in \{1, \dots, \ell\}, \forall \gamma_i \in \tilde{\Gamma}_i.$$

return $\theta^\star := (w^\star, 1)$ and $c^\star, \eta^\star$

Algorithm 5.3 Optimality counter-example generation ("verifier")

function COUNTEREXAMPLELP(B, Γ_i, i)

$$\gamma_i, c_i := \underset{\gamma_i, c_i}{\operatorname{argmax}} \quad c_i,$$

$$\text{s.t.} \quad \sum_{j=1}^{\ell} w_j \cdot \gamma_{ij} + \gamma_{iu} = w_i + c_i,$$

$$\gamma_i \in \operatorname{vert}(\Gamma_i).$$

return γ_i, c_i

The following theorem guarantees that the CEGIS algorithm terminates in finite time with an optimal solution.

Theorem 5.3 (Convergence of CEGIS for PWC-SBFs)**.** *Algorithm 5.1 terminates in finite time to the optimal PWC-SBF $B_{\theta^\star}$ wrt. the problem in Theorem 5.1.*

Proof. The strategy of the proof is as follows: (i) we first prove convergence in

finite number of iterations of Algorithm 5.1 using standard results from linear programming and then, (ii) we prove that the algorithm converges to the optimum using a proof by contradiction.

To prove convergence of Algorithm 5.1 in finite number of iteration, we start by observing that the constraint $\sum_{j=1}^{\ell} w_j \cdot \gamma_{ij} + \gamma_{iu} \leq w_i + c_i$ for all $\gamma_i \in \Gamma_i$ is true if and only if $\sum_{j=1}^{\ell} w_j \cdot \gamma_{ij} + \gamma_{iu} \leq w_i + c_i$ for all $\gamma_i \in \text{vert}(\Gamma_i)$ [121]. Hence for the c_i computed using Algorithm 5.3, it holds that $\sum_{j=1}^{\ell} w_j \cdot \gamma_{ij} + \gamma_{iu} \leq w_i + c_i$ for all $\gamma_i \in \Gamma_i$. Next, let $(w^\star, \eta^\star, c^\star)$ denote the optimal solution of Algorithm 5.2 with respect to a given set of distributions $\{\tilde{\Gamma}_i\}_{i=1}^{\ell}$. Notice that $c^\star$ is a (non-strict) lower bound for $\max\{c_i\}_{i=1}^{\ell}$, provided that c_i is computed from $w^\star$, since $\tilde{\Gamma}_i \subset \Gamma_i$. Then, there are two cases:

- If for some $i \in \{1, \ldots, \ell\}$ it holds that $c^\star < c_i$, then there exists a distribution $\gamma_i \in \Gamma_i \setminus \tilde{\Gamma}_i$ such that $\sum_{j=1}^{\ell} w_j^\star \cdot \gamma_{ij} + \gamma_{iu} > w_i^\star + c^\star$. As a consequence, if Algorithm 5.3 returns a c_i such that $c^\star < c_i$ then the accompanying distribution $\gamma_i \in \text{vert}(\Gamma_i)$ is a *previously unseen* counter-example for the condition $\sum_{j=1}^{\ell} w_j^\star \cdot \gamma_{ij} + \gamma_{iu} \leq w_i^\star + c^\star$ for all $\gamma_i \in \Gamma_i$.
- If instead $c^\star = \max\{c_i\}_{i=1}^{\ell}$, then it holds that $\sum_{j=1}^{\ell} w_j^\star \cdot \gamma_{ij} + \gamma_{iu} \leq w_i^\star + c^\star$ for all $\gamma_i \in \Gamma_i$.

To conclude, it suffices to note that since Γ_i is a polytope, it has a finite number of vertices.

To prove that Algorithm 5.1 converges to the optimal solution, we can employ a proof by contradiction. Assume that $c^\star = \max\{c_i\}_{i=1}^{\ell}$ and the resulting $c^\star$ is not optimal. Then, there must exists a distribution $\gamma_j \in \Gamma_j$ for some $j \in \{1, \ldots, \ell\}$ such that $c^{\star 1} < c^{\star 2}$ with[2]

$$w^{\star 1}, c^{\star 1}, \eta^{\star 1} := \textbf{PwcSynthLP}(X_0, Q, K, \{\tilde{\Gamma}_i\}_{i=1}^{\ell} \cup \gamma_j), \tag{5.14a}$$

$$w^{\star 2}, c^{\star 2}, \eta^{\star 2} := \textbf{PwcSynthLP}(X_0, Q, K, \{\tilde{\Gamma}_i\}_{i=1}^{\ell}). \tag{5.14b}$$

This is a contradiction as $\{\tilde{\Gamma}_i\}_{i=1}^{\ell} \subseteq \{\tilde{\Gamma}_i\}_{i=1}^{\ell} \cup \gamma_j$. □

The convergence to optimality can also be interpreted through the lens of monotone functions and fixed-point theorems. Specifically, **PwcSynthLP** is a monotone function wrt. its last parameter and the $c^\star$ output, and thus induces a (partial) ordering on $\{\tilde{\Gamma}_i\}_{i=1}^{\ell}$. Since **CounterExampleLP** always generates new examples (from a finite set), through the Knaster–Tarski fixed-point theorem, one can prove that the LP-CEGIS algorithm always will converge to the optimal value.

Computational Complexity As noted previously, the LP problem in subroutine **CounterExampleLP** (Algorithm 5.3) can be solved more efficiently using the O-maximization algorithm [122] by recognizing $\bar{w}$ as a discrete value function and

[2] We abuse the notation slightly by letting $\{\tilde{\Gamma}_i\}_{i=1}^{\ell} \cup \gamma_j$ mean $\{\hat{\Gamma}_i\}_{i=1}^{\ell}$ with $\hat{\Gamma}_i = \tilde{\Gamma}_i \cup \{\gamma_j\}$ if $i = j$ and just $\tilde{\Gamma}_i$ otherwise.

that the optimal solution assigns the most probability mass to the highest valued segments. The time complexity of this procedure is $O(n \log n)$, whereas a standard LP solver runs in $O(n^2 m)$ time for n decision variables and m constraints. As a consequence, the LP-CEGIS algorithm is dominated by **PwcSynthLP**, which for a kth iteration of the LP-CEGIS algorithm has the complexity $O(\ell^3 k)$. For a worst-case analysis, assume that all the vertices of the polytope Γ_i must be explored for all $i \in \{1, \dots, \ell\}$, i.e. $\ell \cdot \ell!$ iterations. The result is a complexity of $O(\ell^6 \cdot \ell!^3)$, which asymptotically is worse than the dual LP approach. However, through strategic counter-example generation, it is never necessary to traverse every vertex. It is further noted that Algorithm 5.1 keeps track of the history of counter-examples γ_i added at each iteration, making the LP-CEGIS approach potentially memory intensive, although often less than the dual LP approach.

5.4.3. Gradient Descent

While LP-CEGIS is faster than the dual LP approach, it can be memory intensive. To alleviate this and allow better scalability, we present a third method for computing a PWC-SBF, namely, a Gradient Descent (GD)-based approach.

With abuse of notation, let $c_i(w)$ be martingale gap of region q_i for a given PWC-SBF defined by vector $w = (w_1, \dots, w_K)$. Specifically,

$$c_i(w) = \sup_{\gamma_i \in \Gamma_i} \max \left\{ 0, \sum_{j=1}^{K} w_j \cdot \gamma_{ij} + \gamma_{iu} - w_i \right\}, \tag{5.15}$$

and $c(w) = \max\{c_i(w)\}_{i=1}^{\ell}$. Similarly, we denote

$$\eta(w) = \max_{i \,:\, q_i \cap X_0 \neq \emptyset} w_i. \tag{5.16}$$

Then, we define the loss (objective) function

$$\mathcal{L}(w) = \eta(w) + c(w)K. \tag{5.17}$$

This loss function indeed describes the objective function of the minimax problem in Theorem 5.1. Hence, by minimizing $\mathcal{L}(w)$, we solve the minimax saddlepoint problem. Below, we show that $\mathcal{L}(w)$ is convex, and we can use a GD-based method for its optimization to the global minimum. More precisely, our approach is based on projected subgradient descent, since the elements of w are constrained to $[0, 1]$ and the maximization and supremum in (5.15) are non-smooth, admitting only subgradients.

Proposition 5.2 (Convexity of $\mathcal{L}(w)$)**.** *The function $\mathcal{L}(w)$ is convex in w.*

Proof. The proof follows a standard structure from disciplined convex programming, where functions are composed under convexity-preserving operations. We start by proving that $c_i(w)$ is convex in w. To this end, observe that $\sum_{j=1}^{\ell} w_j \cdot \gamma_{ij} + \gamma_{iu} - w_j$ is convex in w, invariant to the value γ_i. Next, $\max(0, \cdot)$ is a convexity-preserving

function, thus $c_i(w)$ is convex in w. The finite maximization for both η and c are once again convexity-preserving operations and finally, addition is convexity-preserving. Therefore, $\mathcal{L}(w)$ is convex in w. □

Notice that the computation of $c_i(w)$ in (5.15) is equivalent to the inner optimization problem of LP-CEGIS. Hence, the **COUNTEREXAMPLELP** routine in Algorithm 5.3 (based on the O-maximization algorithm) can be used for efficient computation of subgradients of c_i. One subgradient for $\mathcal{L}(w)$ is the following

$$\nabla_w \mathcal{L}(w) = \vec{1}_\eta(w) + \vec{1}_c(w)K, \tag{5.18}$$

where $\vec{1}_\eta(w) = \vec{1}_i$ is a one-hot vector with the 1 being in element $i = \underset{i:q_i \cap X_0 \neq \emptyset}{\text{argmax}}\, w_i$, and

$$\vec{1}_c(w) = \nabla_w c_l(w) = \gamma_l^\star - \vec{1}_l, \tag{5.19}$$

where $l = \text{argmax}_l\, c_l(w)$, and $\gamma_l^\star = \text{argmax}_{\gamma_l \in \Gamma_l} \sum_{j=1}^{\ell} w_j \cdot \gamma_{lj} + \gamma_{lu} - w_l$.

A challenge with applying subgradient methods in practice is that small step sizes are required due to the non-smoothness, slowing down the convergence. We propose to ameliorate this by substituting the maximization in $\mathcal{L}(w)$ with an L_p-norm where $1 < p < \infty$. Specifically, let $\tilde{\eta}(w) = (w_1, \ldots, w_m)$ be a vector of PWC-SBF values corresponding to the regions that overlap with X_0, and $\tilde{c}(w) = (c_1(w), \ldots, c_\ell(w))$ be a vector of $c(w)_i$ values. Then, the proposed loss function is

$$\tilde{\mathcal{L}}_p(w) = \|\tilde{\eta}(w)\|_p + \|\tilde{c}(w)\|_p K, \tag{5.20}$$

and the corresponding gradient is

$$\nabla_w \tilde{\mathcal{L}}_p(w) = \sum_{k=1}^{m} \left(\frac{\tilde{\eta}(w)_k}{\|\tilde{\eta}(w)\|_p} \right)^{p-1} \vec{1}_{i_k} + \sum_{i=1}^{\ell} \left(\frac{\tilde{c}_i(w)}{\|\tilde{c}(w)\|_p} \right)^{p-1} \nabla_w c_i(w). \tag{5.21}$$

The benefit of this loss is that L_p-norms are smooth, and that $\tilde{\mathcal{L}}_p(w) \geq \mathcal{L}(w)$ for every $\gamma_i \in \Gamma_i$ and all w. Furthermore, the magnitude of an L_p-norm for any $p \in \mathbb{R}_{\geq 1}$ is bounded from both sides the L_∞-norm; namely, by $\|y\|_\infty \leq \|y\|_p \leq \sqrt[p]{r}\|y\|_\infty$ for any $y \in \mathbb{R}^r$. With the L_p-norm, we may tune the over-approximation to a trade-off between smoothness and tighter approximating the L_∞-norm. The modified loss, through smoothness, also addresses the issue that each Γ_i is often very sparse, i.e., $\overline{\gamma}_{ij} \approx 0$ for many j such that $\nabla_w \mathcal{L}(w)$ is virtually zero for most regions. We remark that the modified loss $\tilde{\mathcal{L}}_p(w)$ is not smooth due to the maximization and supremum in the computation of $c_i(w)$.

The projected subgradient descent procedure is described in Algorithm 5.4. For each iteration, the algorithm calculates the gradient according to (5.21) and an appropriate decreasing step size α, followed by the gradient step. Since the PWC-SBF must reside in $[0,1]^\ell$ and following the gradient step, w can violate this, it is projected back onto the admissible space. Finally, since subgradient methods are not steepest descent methods, the algorithm keeps track of the best observed loss.

Algorithm 5.4 GD for PWC-SBF

Require: Initial set X_0, partition $Q = \{q_i\}_{i=1}^{\ell}$, time horizon K, and feasible transition kernel sets $\{\Gamma_i\}_{i=1}^{\ell}$.
Ensure: Optimal PWC-SBF $B_{\theta^\star}$ and safety probability bound ρ

$w^{(1)} := (\gamma_{1u}, \dots, \gamma_{\ell u})$
$w_{\text{best}} := w$
$\mathcal{L}_{\text{best}} := \infty$
$k := 1$ ▷ *Iteration number*
while not converged **do**
  $g := \nabla_w \tilde{\mathcal{L}}_p(w)$ ▷ *According to (5.21)*
  $\alpha := \textsc{StepSize}(k, g)$
  $w := \text{proj}_{[0,1]^\ell}(w - \alpha g)$ ▷ *Gradient descent step*
  $\eta, c := \|\tilde{\eta}(w)\|_\infty, \|\tilde{c}(w)\|_\infty$
  if $\eta + cK < \mathcal{L}_{\text{best}}$ **then**
    $w_{\text{best}} := w$
    $\mathcal{L}_{\text{best}} := \eta + cK$
  $k := k + 1$
return $B_{\theta^\star}$ with $\theta^\star := (w_{\text{best}}, 1)$ and $\rho := 1 - \mathcal{L}_{\text{best}}$

To guarantee convergence of gradient descent (Algorithm 5.4) towards the optimum, it is paramount that the objective is convex. Proposition 5.2 shows that $\mathcal{L}(w)$ is convex. From this, it immediately follows that $\tilde{\mathcal{L}}_p(w)$ is also convex.

Corollary 5.2. *For any $p \in \mathbb{R}_{\geq 1}$, the modified objective $\tilde{\mathcal{L}}_p(w)$ is convex in w.*

The proof follows from the fact that $\tilde{\eta}(w)$ and $\tilde{c}(w)$ are trivially convex in w. Further, L_p-norms are convexity-preserving operations [93].

We may characterize the non-asymptotic convergence of Algorithm 5.4, i.e. that it finds an ϵ-optimal w within a finite number of steps.

Proposition 5.3. *Let ϵ denote the error bound and define the gradient bound as $G \geq \sup_{w \in [0,1]^\ell} \|\nabla_w \tilde{\mathcal{L}}_p(w)\|_2$. Furthermore, bound the distance between the initial $w^{(1)}$ and optimal barrier $w^\star$ by $R \geq \|w^{(1)} - w^\star\|_2$. Then there exist integers M_1 and M_2 such that $\alpha_k \leq \epsilon / G$ for all $k > M_1$ and*

$$\sum_{k=1}^{M_2} \alpha_k \geq \frac{1}{\epsilon} \left(R^2 + G^2 \sum_{k=1}^{M_1} \alpha_k^2 \right). \tag{5.22}$$

For any $k > M = \max(M_1, M_2)$ with w_{best} computed by Algorithm 5.4 it holds that

$$\tilde{\mathcal{L}}(w_{\text{best}}) - \tilde{\mathcal{L}}(w^\star) \leq \epsilon. \tag{5.23}$$

The ϵ-optimal convergence follows directly from a result for convergence with strictly decreasing and non-summable step sizes in [123]. A conservative bound on R is ℓ and on G is $m + M\ell\sqrt{2}$. To conclude, Algorithm 5.4 converges to a desired precision in a finite number of steps [123].

Hyperparameters The key hyperparameters for running Algorithm 5.4 are:

- the maximum number of iterations,
- the initial step size,
- the step size decay parameter, which fine-tunes convergence, and
- the momentum parameter, which is used to accelerate convergence.

The maximum number of iterations must be sufficiently high to allow convergence, while the initial learning rate generally performs well with its default value. Reducing the learning rate slows convergence, whereas increasing it too much can lead to divergence until the decay sufficiently reduces it. The decay parameter plays a crucial role, as it must be neither too large nor too small to ensure stability. Finally, momentum is essential for accelerating convergence; however, a too large momentum risks overstepping and introducing training instability. In our implementation, the default values are 10,000 for max iterations, 0.01 for the initial step size, 0.9999 for the decay parameter, and 0.9 for the momentum.

Computational Complexity The complexity per iteration of the projected subgradient descent (Algorithm 5.4), utilizing the O-maximization procedure to compute c_i, is $O(\ell \log \ell)$. We emphasize that the computation can be parallelized.

5.5. EMPIRICAL EVALUATION

To show the power of PWC-SBFs and efficacy of our proposed synthesis methods, we study their performance on a set of seven benchmark problems, consisting of stochastic systems with linear and non-linear dynamics and varying dimensionality, from 2D to 8D. We also compare the performance of PWC-SBFs against state-of-the-art (continuous) SBF techniques, namely SoS and NBF.

Our implementations of all methods, except NBF, including the three proposed PWC-SBF synthesis methods, are in Julia. The code is publicly available on GitHub[3]. The experiments are all executed on a computer with 3.9 GHz 8-Core CPU and 128 GB of memory.

The stochastic systems we considered are:

- 2D linear stochastic system with a (i) convex X_s, and (ii) non-convex X_s,
- 2D pendulum Neural Network Dynamic Model (NNDM) adapted from [99],
- 4D non-linear unicycle model under a hybrid feedback control law,
- 6D linearized lateral and vertical quadrotor guidance model with a hybrid controller with a (i) convex X_s, and (i) non-convex X_s,
- 8D linearized lateral and longitudinal quadrotor guidance model with a hybrid controller.

[3] https://github.com/aria-systems-group/StochasticBarrier.jl.

Table 5.1: Benchmark results for the PWC and continuous SBF methods. The three PWC-SBF synthesis methods are: dual LP, CEGIS and GD. These are compared against state-of-the-art methods: SoS and NBF. ℓ denotes the size of partition of X_s, t_p is the computation time for the bounds on the transition kernel, ρ is the obtained lower bound on the safety probability for $K = 10$ time steps, t_o is the computation time for the SBF synthesis, and Deg is the degree of the SoS polynomial for SBF. Synthesis methods for SoS and NBF, i.e. continuous SBFs, do not use bounds on the transition kernel.

			Piecewise Constant Stochastic Barriers						Continuous Stochastic Barriers				
			Dual LP		LP-CEGIS		GD		NBF		SoS		
Model	ℓ	$t_p(s)$	ρ	$t_o(s)$	ρ	$t_o(s)$	ρ	$t_o(s)$	ρ	$t_o(s)$	Deg	ρ	$t_o(s)$
Linear	64	0.02	0.992	0.52	0.992	0.04	0.952	0.04	0.585	3850.93	4	0.582	0.014
2D	225	0.31	0.998	164.60	0.998	0.44	0.973	0.20	0.940	3991.47	8	0.582	0.265
Convex	900	8.85	0.999	1087.78	0.999	17.93	0.990	7.22	0.961	4025.67	30	0.978	151.16
	2500	41.44	0.999	2897.77	0.999	88.45	0.998	52.78	0.976	4085.65	36	0.992	458.21
Linear	900	5.04	0.494	1197.99	0.494	3.79	0.494	3.52	0.792	3546.69	12	0.010	0.02
2D	1225	8.20	0.800	1389.78	0.800	7.22	0.800	6.64	0.844	3579.58	20	0.010	11.08
Non-Convex	1444	9.18	0.921	1545.45	0.921	11.65	0.921	10.12	0.855	3589.13	24	0.023	37.89
	2926	47.98	0.927	3161.56	0.927	98.36	0.927	20.86	0.928	3599.85	26	0.034	62.88
	5890	179.94	0.929	8191.65	0.929	458.44	0.929	133.84	0.929	3675.77	30	0.075	196.85
	11818	477.45	-	TO	0.936	1875.49	0.936	842.67	0.931	3744.23	34	-	OOM
	24336	987.65	-	TO	0.938	4441.55	0.938	3099.30	0.936	4234.56	36	-	OOM
Pendulum	120	6.37	1.00	0.51	0.99	5.84	0.989	3.75	0.999	4242.89	4	0.999	7.71
2D	240	18.33	1.00	6.08	1.00	14.88	0.990	9.99	0.999	4457.82	4	0.999	34.96
	480	37.84	1.00	29.39	1.00	43.42	0.999	17.88	0.999	4675.12	4	0.999	187.60
Unicycle	1250	1103.42	0.750	1000.19	0.750	26.37	0.750	5.68			2	0.00	3110.21
4D	1800	1756.25	0.975	1719.58	0.975	92.26	0.975	25.78			4	0.00	5451.19
	2400	2001.11	0.998	2548.56	0.998	145.45	0.998	55.59			6	-	OOM
Quadrotor	7865	80.80	0.770	9906.67	0.770	1174.49	0.770	2589.56			2	0.584	0.20
6D	15625	160.61	-	TO	0.901	3788.98	0.901	3258.87			8	0.900	8628.58
Convex	46656	458.59	-	TO	-	TO	0.912	9542.75			12	-	OOM
Quadrotor	15625	188.10	-	TO	0.670	3845.25	0.670	3478.31			4	0.00	4.91
6D	31250	395.59	-	TO	0.810	9548.78	0.810	5878.28			8	0.00	8715.54
Non-Convex	46656	506.99	-	TO	-	TO	0.900	9789.54			12	-	OOM
Quadrotor	65536	845.44	-	TO	-	TO	0.500	19377.90			2	0.00	14830.23
8D	128000	2530.74	-	TO	-	TO	0.560	39132.59			4	-	OOM

The results are shown in Table 5.1. For all case studies, $K = 10$ time steps. TO and OOM denote time-out ($t > 45 \times 10^3$s) and out-of-memory, respectively. Due to limitations pertaining to memory and hybrid control, NBFs are synthesized for the 2D linear system and the pendulum NNDM. For all NBF experiments, the NBF are trained by 60k iterations. Each NBF architecture consists of 2 hidden layers with 32 ReLU activated neurons per layer. For a relevant comparison, it is noted that we use the same initial partitioning for NBF as the partitioning for the PWC-SBF methods. We provide detailed discussions on the experimental setup and obtained results below. For case studies with non-convex X_s, we use the notion of *obstacle* to refer to the unsafe sets. Each obstacle is hyperrectangle $\mathcal{H}_r(c)$ defined by a center point $c \in \mathbb{R}^n$ and $r \in \mathbb{R}^n_{\geq 0}$.

5.5.1. 2D LINEAR SYSTEM

We considered a linear stochastic system on $\mathbb{R}^2$ with the dynamics

$$\mathbf{x}[k+1] = 0.5\mathbf{x}[k] + \mathbf{v}[k] \tag{5.24}$$

where $\mathbf{v} \sim \mathcal{N}(\cdot \mid 0, 10^{-2}I)$ and I is the identity matrix. The initial set is $X_0 = [-0.8, -0.6] \times [-0.2, 0.0]$. We consider two cases for the safe set X_s, both defined based on the set $X_D = [-1, 0.5] \times [-0.5, 0.5]$ as shown in Figure 5.1a:

1. Convex safe set: $X_s = X_D$, and
2. Non-convex safe set: $X_s = X_D \setminus \big(\mathcal{H}_{r_1}(c_1) \cup \mathcal{H}_{r_2}(c_2)\big)\big)$, where

$$c_1 = (-0.55, 0.30), \qquad r_1 = (0.02, 0.02),$$
$$c_2 = (-0.55, -0.15), \qquad r_2 = (0.02, 0.02).$$

Convex X_s case Observe in Table 5.1 that all three PWC-SBF methods outperform the state-of-the-art continuous SBF techniques SoS and NBF. It takes a degree 36 polynomial SBF to obtain a result similar to that of the PWC methods using just $\ell = 64$, despite the absence of obstacles (convexity of X_s). This is mainly attributed to the fact that the initial set is not symmetric around 0, making curve fitting with SoS difficult for even simple 2-dimensional problems.

The main limitation of NBF in this experiment is the training time. Note that it takes the continuous SBF methods 3-4 orders of magnitude longer computation time to arrive at a similar value of ρ as the PWC methods. In addition, the continuous SBF methods never reached $\rho = 0.999$.

Among the PWC-SBF methods, dual LP and LP-CEGIS always obtain the same ρ, as expected as both are rely on standard LP solvers to optimality. However, it takes LP-CEGIS at least an order of magnitude less time to compute this optimal value. This is due to the large number of (dual) decision variables in the dual LP, which increases quadratically as ℓ increases. The GD approach is the fastest method; however, its hyperparameters are tricky to design. For this case study, GD terminated before convergence to the optimal ρ.

Non-convex X_s case For this case, the power of the PWC-SBFs is even more evident (see Figure 5.1). Due to the fact that the obstacles are within the safe set, the SoS approach has difficulties fitting a function. This forces $B(x) \geq 1$ over the entire state space, as is evident from Figure 5.1b. This phenomenon does not improve much for a significant increase in the degree of the SoS polynomial, as it is an inherent limitation of the appraoch.

On the other hand, our methods effectively tackle this problem by assigning $w_i = 1$ to the regions that overlap with the obstacles, and optimize over the remainder of the state space. This results in the PWC as depicted in Figure 5.1d. The difference is likewise noticeable in terms of the safety probability ρ, where SoS can only guarantee safety probability of 0.075 for the polynomial degree 30, while two of PWC-SBF methods provide up to 0.938 safety probability.

Note that for large values of ℓ, the dual LP is unable to terminate, due to the scope of the required convex optimization. A similar fact is observed for high degree SoS barrier polynomials. The NBF method performs well in terms of safety probability; at lower number of regions, even better than PWC-SBF. Nonetheless, PWC-SBFs are superior in terms of computation time.

5

5.5.2. 2D Pendulum NNDMs

In this case study, we consider the NNDM from [99] with dynamics $\mathbf{x}[k+1] = f^{NN}(\mathbf{x}[k]) + \mathbf{v}[k]$, where f^{NN} is a NN trained from data of a pendulum with a fixed mass m and length l in a closed-loop with an NN controller. The actuator is limited by $u \in [-1, 1]$ and the dynamics of the pendulum are

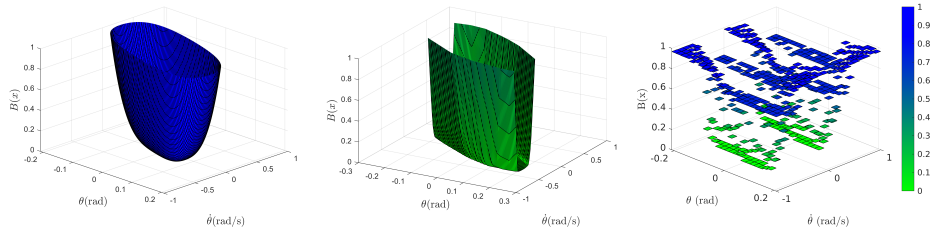
$$\dot{\theta}_{k+1} = \dot{\theta}_k + \frac{3g}{2l}\sin(\theta_k)h^2 + \frac{3}{ml^2}uh^2, \tag{5.25}$$

$$\theta_{k+1} = \theta_k + \dot{\theta}_{k+1}h \tag{5.26}$$

For the details on the trained model, we refer to [99]. For this system, noise $\mathbf{v} \sim \mathcal{N}(\cdot \mid 0, 10^{-4}I)$. The safe set is $\theta \in [-\frac{\pi}{15}, \frac{\pi}{15}]$ and $\dot{\theta} \in [-1, 1]$ and the initial set is $\theta, \dot{\theta} \in [-0.01, 0.01]$.

We note that since for this particular case study the dynamics are given by an NN, the SoS approach also uses the partitioning of size ℓ, where local linear relaxations of f^{NN} are computed using CROWN [88] (see Section 2.2.2 for more details on CROWN and linear relaxations) and the relaxations are embedded into a SoSP problem [99].

As can be observed from Table 5.1, the PWC, SoS and NBF methods perform similarly in terms of safety probability. It is noted that SoS performs well due to the fact that the initial set is centered in the state space. In Figure 5.2, we show the plots of the SoS, NBF, and PWC-SBF (dual LP approach) barriers. In Figure 5.2a, we observe that the obtained SoS polynomial greatly under-approximates the safe set and that the values near the boundary of the safe set by far exceed one. We observe a similar yet less excessive pattern for the NBF in Figure 5.2b. For the PWC-SBF the barrier in the domain is less then or equal to 1. Among the PWC-SBF methods, a general trend is observed: the dual LP is the slowest algorithm and GD the fastest. It is also worth noting that that the GD method terminates with a



(a) Degree-8 SoS-polynomial (b) NBF with $|\ell| = 480$. (c) PWC (dual LP) with $|\ell| = 480$.

Figure 5.2: SBF plots for the pendulum NNDM case study.

ρ that is slightly below the optimal value, indicating the difficulty in designing the termination criterion.

5.5.3. 4D UNICYCLE

In this case study, we consider a wheeled mobile robot with the dynamics of a unicycle

$$\dot{x} = v\cos\theta, \quad \dot{y} = v\sin\theta, \quad \dot{\theta} = \omega, \quad \dot{v} = a, \tag{5.27}$$

where $x \in [-1.0, 0.5]$ and $y \in [-0.5, 1.0]$ are the Cartesian position, $\theta \in [-1.75, 0.5]$ is the orientation with respect to the $x-$axis, and $v \in [-0.5, 1.0]$ is the speed. The inputs are steering rate ω and acceleration a. We design a feedback linearization controller according to [124] coupled with an LQR stabilizing controller, making this a hybrid model. We use $h = 0.01$ to obtain discrete time dynamics, using the Euler method. We add noise $\mathbf{v} \sim \mathcal{N}(\cdot \mid 0, 10^{-4}I)$ to capture the time-discretization error that is inherent to the Euler method.

Figure 5.3 shows 10^5 Monte Carlo simulations of the trajectories of the unicycle from the initial set, defined by a hyperrectangle $\mathcal{H}_r(c)$ with $c = (-0.5, -0.5, 0, 0)$ and $r = (0.01, 0.01, 0.01, 0.01)$. These simulated trajectories suggest that the system is stable under the hybrid control law. This is in line with the results in Table 5.1 for the PWC methods, especially for $\ell = 2400$, for which $\rho = 0.998$. Note that the SoS approach highly suffers for this non-linear case study, with $\rho = 0$, despite very long computation times. This further emphasizes the efficacy of our methods. Observe that the GD approach terminates with the optimal ρ value in this case study. We finally note that since the controller is hybrid, current NBF formulations cannot be employed for this problem.

5.5.4. 6D AND 8D QUADCOPTER SYSTEMS

To evaluate the scalability of the proposed PWC-SBF methods, we considered a set of high-dimensional systems. Specifically, we considered a quadrotor guidance model where the dynamics consists of lateral and vertical dynamics and the longitudinal and spin variables are constrained, and then a lateral and longitudinal motion model by fixing the vertical and spin variables.

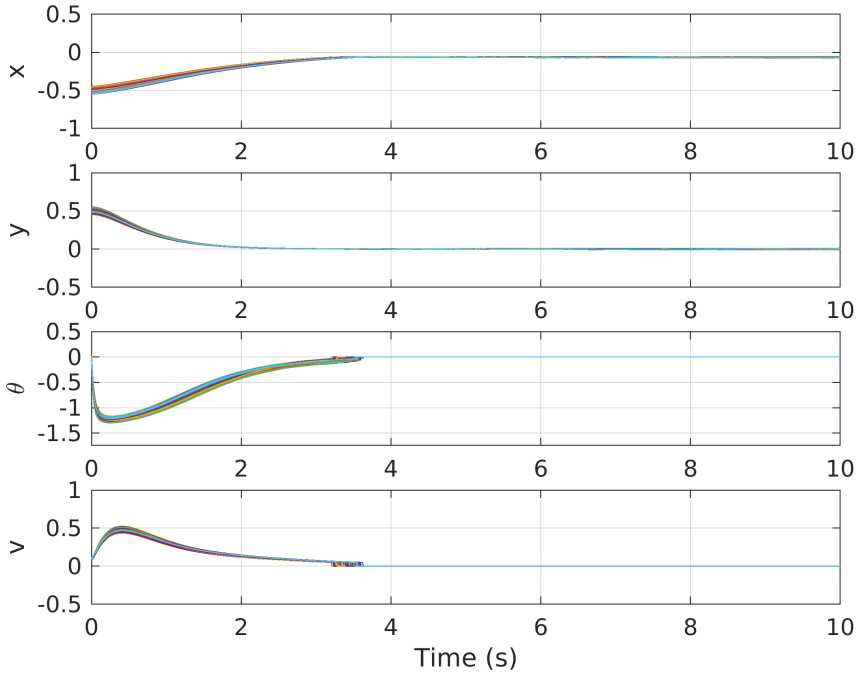
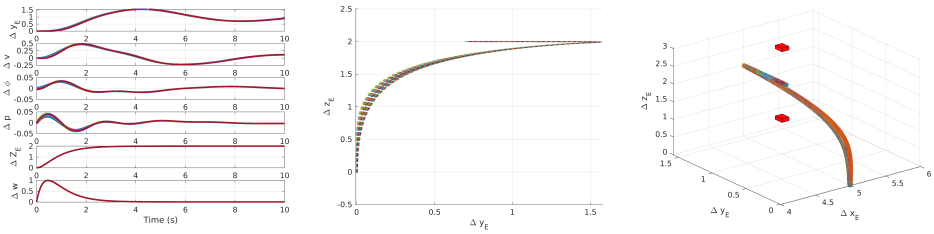


Figure 5.3: Monte Carlo simulation (10^5 instances) for the unicycle model over a timespan $t = (0, 10)$s.



(a) State Evolution (b) y-z Plane (c) 3D Workspace

Figure 5.4: Monte Carlo simulations (10^5 instances) for the 6D quadcopter model with lateral-vertical dynamics over a timespan $t = (0, 10)$s. Trajectories are visualized in (a) state vs. time plot, (b) y-z plane, and (c) 3D workspace. The red boxes in (c) are obstacles. The control laws stabilize the quadrotor around the equilibrium point $x_{\text{eq}} = (1, 0, 0, 0, 2, 0)$.

6D LATERAL-VERTICAL MODEL

The lateral closed-loop state space model is defined as

$$\begin{aligned} \Delta\dot{y}_E &= \Delta v, \quad \Delta\dot{v} = g\Delta\phi, \\ \Delta\dot{\phi} &= \Delta p, \quad \Delta\dot{p} = \frac{1}{I_x}\Delta L_c, \end{aligned} \tag{5.28}$$

where g is gravity, and I_x is the inertia about the $x-$axis. State $y_E \in [-0.5, 2.0]$ is the $y-$position, and $v \in [-1.0, 1.0]$ is the corresponding velocity. The roll angle $\phi \in [-0.1, 0.1]$, the roll rate $p \in [-0.1, 0.1]$, and L_c is the roll-control moment. The control law follows directly from

$$\Delta L_c = -k_1\Delta p - k_2\Delta\phi - k_3\Delta v - k_3k_4\Delta y_E + k_3k_4\Delta y_{E,r}, \tag{5.29}$$

where k_i denotes the gain for $i \in \{1, \dots, 4\}$, and $\Delta y_{E,r}$ is the reference position.

The lateral model is appended with a guidance model for vertical motion given by

$$\Delta\dot{z}_E = \Delta w, \qquad \Delta\dot{w} = \frac{1}{m}\Delta Z_c, \tag{5.30}$$

where m denotes the quadrotor mass, $z_E \in [-0.5, 3.0]$ denotes the z-position and $w \in [-0.5, 1.5]$ is the yaw rate. The control law ΔZ_c is defined as

$$\Delta Z_c = -k_1\Delta z_E - k_2\Delta w - Fr \tag{5.31}$$

where Fr denotes the reference signal for the vertical dynamics. The full dynamics constitutes a 6D linear model. Using $h = 0.01$, we obtained a discrete-time dynamics via the Euler method to which we added noise $\mathbf{v} \sim \mathcal{N}(\cdot \mid 0, 10^{-4}I)$.

The designed control laws stabilize the quadrotor around the equilibrium point $x_{\text{eq}} = (1, 0, 0, 0, 2, 0)$. Monte Carlo simulations of 10^5 trajectories of this model is shown in Figure 5.4. The 3D workspace (environment) is shown in Figure 5.4c.

We considered two safe sets:

1. Convex safe set: ${X_s}^{conv}$ is defined by the given ranges of the variables above,
2. Non-convex safe set: $X_s = {X_s}^{conv} \setminus \left(\mathcal{H}_{r_1}(c_1) \cup \mathcal{H}_{r_2}(c_2)\right)$, where

$$\begin{aligned} c_1 &= (1, 0, 0, 0, 1, 0), & r_1 &= (0.01, 0.01, 0.01, 0.01, 0.01, 0.01) \\ c_2 &= (1, 0, 0, 0, 2.75, 0), & r_2 &= (0.01, 0.01, 0.01, 0.01, 0.01, 0.01). \end{aligned}$$

Figure 5.4c depicts the obstacles.

Convex X_s case Table 5.1 shows that two PWC methods, namely, CEGIS and GD, outperform SoS in terms of both computation time and safety probability certification. It is important to realize that systems with linear dynamics and convex X_s are an ideal setup for SoS optimization. However, the dimensionality of the system poses a challenge for SoS as the results suggest. For this case study, a barrier polynomial of degree greater than 8, the memory requirements for SoS exceeds available memory (128GB). Finally, we were unable to run NBF for this problem, since it cannot handle hybrid controllers and suffers from scalability.

Non-convex X_s case In this case, it is even more evident that PWC-SBFs can be more powerful than continuous SBFs. The SoS method provides $\rho = 0.00$, whereas the GD method gets to $\rho = 0.900$. Observe that the dual LP method exceeds the time limit, which is also true for the CEGIS method when $\ell > 4.5 \times 10^4$. The GD approach is able to handle this, showing the capability of PWC-SBFs in scalability.

5.5.5. 8D Lateral-Longitudinal Dynamics

Here, we aim to test the performance boundary of PWC-SBFs in terms of scalability. We consider an 8D model by combining the lateral dynamics of the quadcopter with its longitudinal and fixing vertical and spin variables. In a similar fashion as lateral, the longitudinal guidance is constructed under the dynamics

$$\Delta\dot{x}_E = \Delta u, \qquad \Delta\dot{u} = -g\Delta\delta,$$
$$\Delta\dot{\theta} = \Delta q, \qquad \Delta\dot{q} = \frac{1}{I_y}\Delta M_c,$$

5

where g is gravity, and I_y is the inertia about the $y-$axis. State $x_E \in [-0.5, 4.0]$ is the $x-$position, and $u \in [-0.5, 1.5]$ is the corresponding velocity. The pitch angle $\theta \in [-0.1, 0.1]$, the pitch rate $q \in [-0.1, 0.1]$, and M_c is the pitch control moment. The control law follows directly from

$$\Delta M_c = -l_1\Delta q - l_2\Delta\theta - l_3\Delta u - l_3 l_4 \Delta x_E + l_3 l_4 \Delta x_{E,r}, \tag{5.32}$$

where l_i denotes the gain for $i \in \{1, \dots, 4\}$, and $\Delta x_{E,r}$ is the reference position.

The results are shown in the bottom two rows of Table 5.1. For this system, the SoS approach is unable to provide any safety guarantees, i.e., $\rho = 0.00$. The dimensionality naturally has a major effect on the performance of the SoS optimizer. The dual LP and LP-CEGIS also suffer due to the dimensionality and the partitioning of the state space. Nonetheless, the GD approach is still able to handle the large scale optimization, and can provide a lower safety probability of $\rho = 0.560$. While the safety certificate is not near the statistical safety threshold obtained through simulation ($P_{\text{safe,sim}} \approx 0.95$), it clearly shows that the proposed PWC-SBF method outperforms current state-of-the-art SBF methods.

5.6. Control Synthesis and Forward Invariance via Piecewise Constant Stochastic Barrier Functions

In this section, the methods for syntesizing PWC-SBFs presented in this chapter will be extended to control systems and the computation of probabilistic forward invariance [77]. That is, find the set of admissible controls, for each state, to ensure a guaranteed safety probability of at least P_{safe}, for a given safe set X_s, initial set X_0, and time horizon K. The admissible control set can be interpreted as a pre-shield [125] such that *any* downstream controller obeying the admissible set also inherits the guaranteed safety probability and can thus focus on optimizing for performance.

The following theorem formalizes an optimization problem for synthesis of a permissible strategy set $\Pi_x : X \rightrightarrows U$ using PWC-SBFs and the accompanying safety certificate.

Theorem 5.4 (PWC-SBFs for Permissible Strategies)**.** *Consider System* (2.35) *with the safe set* $X_s \subset \mathbb{R}^n$*, the initial set* $X_0 \subseteq X_s$*, time horizon* K*, and safety bound* ρ*. Assume given a partition* $Q = \{q_1, \dots, q_\ell\}$ *of the safe set* X_s *and a partition* $\mathcal{U} = \{U_1, \dots, U_L\}$ *of the control space* U*. Define the index set* $I = \{1, \dots, \ell\}$ *and the (controlled) ambiguity set* $\Gamma^a = \prod_{i \in I} \Gamma_i^a$*, where* Γ_i^a *is the set of feasible transition probabilities in* (5.7) *for* $a \in A = \{1, \dots, L\}$ *and* $i \in I$*. Let* $\Pi : I \rightrightarrows A$ *be a function to assign sets of admissible control subset indices to each region index* $i \in I$*, and let* $\Pi_x(x) = \cup_{a \in \Pi(i)} U_a$ *for* $x \in q_i$ *be the equivalent on the domain* X *and the codomain* U*. Finally, assume a PWC function template in the form* (5.1)*, with* $B_i(x) = w_i \in \mathbb{R}_{\geq 0}$*, for every* $i \in I$*, and let* $\theta^\star \in \Theta = \mathbb{R}_{\geq 0}^\ell$ *be the optimal solution to the following optimization problem*

$$\theta^\star = \underset{\theta \in \Theta}{\operatorname{argmin}} \max_{\{\Gamma^a\}_{a \in A},\, \gamma_i^a \in \Gamma_i^a} \quad \eta + cK \tag{5.33a}$$

$$w_i \leq \eta \qquad \forall i \in I,\ q_i \cap X_0 \neq \emptyset, \tag{5.33b}$$

$$\sum_{j=1}^{K} b_j \cdot \gamma_{ij}^a + \gamma_{iu}^a \leq w_i + c_i^l \quad \forall i \in I,\ \forall a \in \Pi(i), \tag{5.33c}$$

$$0 \leq c_i^a \leq c, \qquad \forall i \in I,\ \forall a \in A, \tag{5.33d}$$

where γ_{ij}^a *denotes the transition probability from region* q_i *to region* q_j *under the action (set)* U_a*, and* γ_{iu}^a *similarly with the destination* X_u*. Then,*

1. $B_{\theta^\star}$ *constitutes a SBF for System* (2.35) *that guarantees safety probability* $P_{\text{safe},\pi} \geq 1 - (\eta + cK)$ *for all* $\pi \in \Pi_x$*, and*
2. *if* $c_i^a \leq (1 - \eta - \rho)/K$*, then every choice of control input* $u \in U_a$ *is permissible for every* $x \in q_i$ *of System* (2.35)*.*

Proof. The optimization is an exact solution to the problem in Theorem 5.1. It is straightforward that $B_{\theta^\star}$ is a proper barrier certificate, and that for every strategy $\pi \in \Pi_s$, the safety guarantees is $P_{\text{safe}} \geq 1 - (\eta + cK)$, where $c = \max\{c_i^l\}_{\{i \in \{1, \dots, \ell\}, l \in \Pi(i)\}}$. Further, it follows that if $c_i^l \leq (1 - \eta - \rho)/K$, state-control partition $Z_{il} = q_i \times U_l$ is rendered safe for all $(x, u) \in Z_{il}$. Therefore, the probabilistic safety guarantees hold for System (2.35). □

The above theorem provides a method of identifying permissible strategies, which requires solving a minimax optimization problem. This problem, similar to Theorem 5.1, includes bilinear terms, that is, $w_j \cdot \gamma_{ij}^l$ in Condition (5.33c). Nonetheless, following the same approach developed in this chapter, the resulting minimax

Algorithm 5.5 PWC-SBF based Control Invariant Sets

Require: Initial set X_0, state-control space $Z = \{Z_{il}\}_{i\in\{1,\dots,\ell\},l\in\{1,\dots,L\}}$, time horizon K, feasible transition probabilities $\Gamma = \{\Gamma^l\}_{l=1}^{L}$, and probability threshold ρ.

Ensure: Permissible strategy set Π_s or `False` if no permissible strategy can be found with $P_{\text{safe}} \geq \rho$.

1: $\Pi := i \mapsto \{1, \dots, L\}$
2: $\eta, \{c_i^l\}_{i\in\{1,\dots,\ell\},l\in\Pi(i)}, \theta^\star :=$ **BARRIER**(X_0, Z, K, Γ) ▷ *Theorem 5.4*
3: **while** $1 - (\eta + \max\{c_i^l\}_{i\in\{1,\dots,\ell\},l\in\Pi(i)} K) < \rho$ **do**
4: $\quad il := \text{argmax}_{i\in\{1,\dots,\ell\},l\in\Pi(i)}\, c_i^l$
5: $\quad Z := Z \setminus Z_{il}$
6: $\quad$ **if** all control sets removed for some q_i **then**
7: $\quad\quad$ **return** `False`
8: $\quad \Pi := i \mapsto \{l : (q_i, U_l) \in Z\}$
9: $\quad \eta, \{c_i^l\}_{i\in\{1,\dots,\ell\},l\in\Pi(i)}, \theta^\star :=$ **BARRIER**(X_0, Z, K, Γ) ▷ *Theorem 5.4*
10: **return** $\Pi_s(x) = \{U_l : (q_i, U_l) \in Z\}$ for $x \in q_i \subset X_s$ and $B_{\theta^\star}$

problem can be efficiently solved by the LP-CEGIS or GD method. The evaluations in this section use LP-CEGIS.

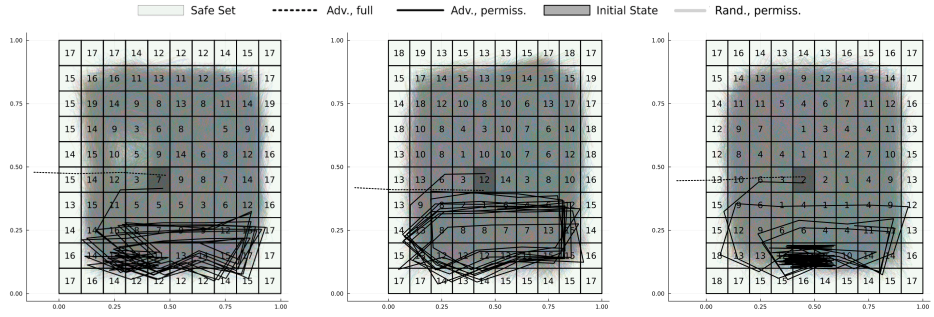
We propose an algorithm that computes a maximal set of permissible strategies Π_s wrt. to the partitions of the safe set and control space. An overview is shown in Algorithm 5.5, taking as input the initial set X_0, time horizon K, set $Z_=\{Z_{il}\}_{i\in\{1,\dots,\ell\},l\in\{1,\dots,L\}}$, the ambiguity sets Γ, and the desired safety threshold P_{safe}. On Line 2, the algorithm computes an initial PWC-SBF using Theorem 5.4.

Using the PWC-SBF, the state-control pair Z_{il} corresponding to $\max\{c_i^l\}_{i\in\{1,\dots,\ell\},l\in\Pi(i)}$, i.e. restricting higher safety guarantees, is identified and removed on Lines 4-5. Intuitively, this means that the algorithm sequentially prunes the control region U_l that is deemed not permissible for a given state partition q_i. Then, a new PWC-SBF is synthesized on Line 9 using Theorem 5.4. A check after pruning is incorporated to ensure that for all regions q_i, there exists at least one control region U_l in the updated space Z. If not, then the algorithm failed to find a permissible strategy with $P_{\text{safe}} \geq \rho$ and it terminates as `False`.

Proposition 5.4. *Algorithm 5.5 terminates in finite time. If it terminates with the return statement on Line 10, then it returns Π_s, which is guaranteed to include only permissible strategies.*

Proof. Synthesizing a barrier terminates in finite time and since Algorithm 5.5 removes a pair Z_{il} from a finite set Z, the algorithm terminates in finite time. The proof that Π_s is guaranteed to include only permissible strategies is a direct implication of Theorem 5.4. □

Algorithm 5.5 enables not only synthesizing SBFs for a broader class of systems, but also achieving



(a) GP 500 - $P_{\text{safe}} \geq 1 - 8.96 \times 10^{-3}$ (b) GP 1000 - $P_{\text{safe}} \geq 1 - 9.10 \times 10^{-3}$ (c) GP 1500 - $P_{\text{safe}} \geq 1 - 9.95 \times 10^{-3}$

Figure 5.5: Non-linear system case studies using various sizes of datasets. Regions with the number of removed control regions and simulated trajectories over 100 steps are shown in each plot. Adversarial trajectories show permissible control sets maintain safety.

5.6.1. EVALUATION

Consider the non-linear dynamical system $\mathbf{x}[k+1] = \mathbf{x}[k] + 0.2 \begin{pmatrix} \cos(\mathbf{u}[k]) \\ \sin(\mathbf{u}[k]) \end{pmatrix} + \mathbf{v}[k]$ with $U = [-\pi, \pi]$, which is partitioned in 20 regions, and the noise $\mathbf{v}[k]$ is diagonal Gaussian with zero-mean and standard deviation 0.01 in each dimension. This is a simplified Dubin's car model, where the heading is controlled directly, allowing movement in any direction. The initial set is $X_0 = [0.4, 0.5]^2$ and the safe set is $X_s = [0.0, 1.0]^2$. Moreover, the nominal dynamics, that is, without the noise $\mathbf{v}[k]$, are unknown and instead learnt with a Gaussian Process (GP) from a dataset D with the method from [66]. Then, formal bounds Γ on the underlying transition kernel are established. Permissible sets are constructed with Algorithm 5.5 using a convergence threshold of $P_{\text{safe}} = 1 - 10^{-4}$ over a horizon of $K = 100$. Computing the permissible sets takes approximately 1-1.5 hours.

Figure 5.5 illustrates the comparison of permissible strategy sets obtained for the system using datasets of 500, 1000, and 1500 datapoints. Despite the variations in dataset size, each permissible set constrains the probability of exiting the safe set within a similar bound. The resulting permissible sets for the system learned with 500 and 1000 datapoints maintained 39.5% and 40.1% of all available controls, respectively. Adding 500 datapoints increased the available controls to 49.5%. Out of the 1000 sampled trajectories none exited the safe set, and only the adversarial trajectory using the full control set violated safety. The strict bound on c during permissible set synthesis removed a majority of the controls to ensure safety. Relaxing the condition could add more strategies to the permissible set.

5.7. CONCLUSION

This chapter extends the theory of Stochastic Barrier Functions (SBFs) to piecewise barrier templates for discrete-time stochastic systems with additive Gaussian noise. The extension piecewise SBFs enables greater flexibility in set geometries, which

is less accessible with NBF (see the previous Chapter 4) and is important in some contexts. Furthermore, it is shown that *in the limit* the class of piecewise constant SBFs (PWC-SBFs) are optimal over the set of measurable SBFs.

Three complementary synthesis methods for PWC-SBFs are proposed: dual LP, LP-based CEGIS, and gradient descent. Dual LP is a dual formulation of the min-max problem that is the PWC barrier synthesis program. Consequently, the method is exact, but is also computationally expensive due to the additional dual variables and constraints. The LP-based CEGIS approach reduces this computational burden by alternating between a candidate generation and verification phase, corresponding to the min and max of the original problem respectively. A complexity analysis reveals that in the worst case, LP-CEGIS is asymptotically worse than the dual LP approach, however, in practice, LP-CEGIS as shown in Section 5.5. The final method, gradient descent, is derived by observing that the min-max problem is a convex-concave saddle point problem, thus allowing convergence proofs to the saddle point. Furthermore, the inner maximization problem admits an efficient algorithm for its solution: O-maximization, which also hints at the connections between PWC-SBFs and IMDP-based finite-state abstractions, which we study in Chapter 9. A disadvantage of the gradient descent approach is the need for application-dependent hyperparameter tuning with parameters such as step size (schedule), momentum, and convergence criteria.

A key insight of this work is that limiting barrier functions to piecewise-constant forms significantly reduces the complexity of verification, without sacrificing formal guarantees. This highlights a broader principle: carefully chosen structural assumptions can enable scalability without compromising expressivity.

The approach and setting of this chapter are not without limitations. In particular, the approaches are design under the assumption that the dynamics are known and the noise is additive and Gaussian. In practice, these assumptions are strong and require relaxation. In the following Chapter 6, we will indeed relax the assumption of a known noise distribution, by replacing it with a sampling access assumption and designing a data-driven synthesis framework. Furthermore, the premise of these methods is that a partition is given; it remains an open question how to design a good partition, balancing safety guarantee and memory/computational requirements. A potential approach is adaptive refinement such as the one proposed in [40] for finite-state abstractions.

6

Scenario Theory-based Synthesis of Stochastic Barrier Functions

To consult the statistician after an experiment is finished
is often merely to ask him to conduct a post mortem examination.
He can perhaps say what the experiment died of.

Ronald Fisher

This chapter is a copy of *F. B. Mathiesen, L. Romao, S. C. Calvert,* et al., A data-driven approach for safety quantification of non-linear stochastic systems with unknown additive noise distribution*, Accepted for Automatica, 2024. arXiv: 2410.06662 [eess.SY]* [79] with minor corrections to streamline the presentation. A preliminary version focusing on piecewise affine systems appeared in [78].

6.1. Introduction

The previous Chapters 4 and 5 introduced two new methods for synthesizing SBFs. Both approaches, however, rely on a critical assumption: the distribution of the stochastic noise affecting the system is known and Gaussian. In practice, this assumption is often unrealistic. Real-world systems rarely provide precise knowledge of the underlying noise distribution [126], making such distributional assumptions problematic for safety verification. This raises a fundamental question: how can we compute formal certificates of safety for stochastic systems when the noise distribution is unknown?

The goal is to provide formal safety guarantees that hold with high probability. These guarantees fall into the category of Probably Approximately Correct (PAC)-like results. Specifically, given a sufficiently large dataset D, the safety probability of the system is computed with confidence level $1-\beta$.

Problem 6.1

Given a system $\mathbf{x}[k+1] = f(\mathbf{x}[k]) + \mathbf{v}[k]$ (i.e., of additive form (2.34)) with *unknown* noise distribution $p_{\mathbf{v}}$, a dataset $D = \{v_1, \dots, v_N\}$ sampled i.i.d. according to $p_{\mathbf{v}}$, an initial set X_0, a safe set X_s, and a time horizon K, compute a lower bound ρ_D on P_{safe}, which holds with probability at least $1-\beta$.

The confidence level is defined with respect to the sample probability of the dataset. That is, if ρ_D is the safety probability bound computed given D, then

$$\mathbb{P}_{D\sim(p_{\mathbf{v}})^N}[\rho_D \leq P_{\text{safe}}] \geq 1-\beta \tag{6.1}$$

where $(p_{\mathbf{v}})^N$ is the N-fold probability distribution over $p_{\mathbf{v}}$. Intuitively, this means that ρ_D should be designed to be robust to the sampling of D such that it has a vanishingly small probability of being incorrect; typically with β in the range 10^{-6} to 10^{-9}. It is important to stress that we desire the largest ρ_D, while satisfying (6.1).

Remark 6.1. *The assumption that f is known exactly can be relaxed. As we will detail in Subsection 6.2.1, our approach only requires f to be represented as a piecewise uncertain affine function, thus, allowing for, e.g. parameteric uncertainty, in f. However, for ease of exposition, we assume a system is of the form (2.34).*

Recent research has started to address this question by adopting data-driven methods for SBF synthesis to work directly with observed samples of system behaviour. Notably, [33], [127] apply a Sample Average Approximation (SAA) method combined with Chebyshev's inequality to synthesize an SBF. While this represents a step forward, the approach suffers from poor sample complexity: the number of required samples grows linearly with the desired confidence level, i.e. the number of samples is proportional to $1/\beta$, making the method impractical for high-confidence guarantees.

In this chapter, we propose a data-driven solution based on the scenario approach, which achieves a much more favourable logarithmic relationship between the number of samples and the confidence level [94]. The method is built on a novel

inner approximation of the stochastic program that typically defines SBF synthesis. By reformulating the problem as a chance-constrained optimization, we enable the direct application of scenario approach theory, which provides probabilistic guarantees with sample-efficient logarithmic performance, namely the number of samples required is proportional to $\log(1/\beta)$ [94].

For the barrier function templates, we focus on PWA forms, which is a specialization of the piecewise barrier functions discussed in Chapter 5. Specifically, given a partition $Q = \{q_1, \dots, q_\ell\}$ of X, we consider templates of the form $B_\theta(x) = \max\{B_1(x), \dots, B_\ell(x)\}$ where $B_i(x) = u_i^\top x + w_i$ for $x \in q_i$, and zero otherwise. For this class of functions, the barrier parameters are $\theta = (u_i, w_i)_{i=1}^{\ell} \in \Theta = \mathbb{R}^{\ell(n+1)}$. The piecewise template requires that we assume zero-probability boundaries, i.e. Assumption 5.1 from the previous Chapter 5. To accommodate systems with nonlinear dynamics, we integrate the neural network verification techniques introduced in Subsection 2.2.2. This combination allows the proposed method to handle both complex dynamics and unknown stochastic disturbances while providing formal, data-driven safety guarantees.

6.2. CHANCE-CONSTRAINED APPROXIMATIONS OF BARRIER PROGRAM

Solving the (generic) barrier program (3.10), that is, $\min_{\theta,\eta,c} \eta + cK$ subject to the barrier conditions in Definition 3.1, is challenging, even in the case that the probability distribution underpinning (3.1c) is known. In this section, we show how to relax (3.10) using a reformulation in terms of a chance-constrained optimisation problem whose feasible set is contained in the set of feasible solutions of (3.10). Such ideas have never been used in this context. Hence, we depart completely from the approaches taken by [15], [18], [98], [127]–[129], which either rely on approximating the martingale condition (3.1c) with the empirical distribution, make strong assumptions about the noise distribution to analytically compute the expectation and recast (3.1c) as a convex constraint, or rely on convex overapproximations of the expectation to (conservatively) verify (3.1c). Furthermore, due to the inner approximation of (3.10) in terms of a chance-constrained problem, our approach opens the road to use the tools of scenario optimization discussed in Subsection 2.2.4 to obtain strong sample complexity guarantees on the safety probability of System (2.34).

The inner chance-constrained approximation relies on partitioning the sample space $\mathbb{R}^m$ for the expectation in (3.1c) into two subsets: the value in one subset is bounded by the deterministic equivalent of (3.1c) while the other subset is covered by a uniform upper bound. To formalize the above idea, fix any B_θ and $x \in \mathbb{R}^n$, and let $V \in \mathcal{F}$ be a Borel measurable set. Then, observe that

$$\mathbb{E}[B_\theta(f(x) + \mathbf{v})] = \int_V B_\theta(f(x) + v)\, d\mathbb{P}(v) + \int_{V^c} B_\theta(f(x) + v)\, d\mathbb{P}(v). \tag{6.2}$$

Under the assumption that $B_\theta(x) \leq \overline{B}$ for any $x \in \mathbb{R}^n$, which can always be enforced for PWA-SBFs, the second term on the right-hand side of (6.2) can be bounded by

$\overline{B}\mathbb{P}[V^c]\}$. Our main idea then is to choose a particular V^c that allows us to control the right-hand side of (6.2), and conquer enough of $\mathcal{F}$ to ensure the desired tightness. Such reasoning is made formal in the next lemma.

Lemma 6.1. *For B_θ with $\|B_\theta\|_\infty = \overline{B}$, let c and v be defined as in (3.10). Define the set*

$$V = \{v \in \mathbb{R}^m \,:\, B(f(x)+v) + \xi \leq B(x) + c,\ \forall x \in X_s\}, \tag{6.3}$$

for some constant $\xi \geq 0$. If there exists an $\epsilon \in (0,1)$ such that $\xi \geq \overline{B}\frac{\epsilon}{1-\epsilon}$ and $\mathbb{P}[\mathbf{v} \in V^c] \leq \epsilon$, then we have $\mathbb{E}[B(f(x)+\mathbf{v})] \leq B(x) + c$ for all $x \in X_s$.

Proof. The proof of our result is based on (6.2). We first remark that V is a Borel measurable set due to the structure of B (upper semi-continuous, thus Borel measurable) and the fact that measurability is closed under addition and composition. Next, fix any $x \in X_s$ and define the set

$$V_x = \{v \in \mathbb{R}^m \,:\, B(f(x)+v) + \xi \leq B(x) + c\}. \tag{6.4}$$

Notice that, by definition, we have $V \subset V_x$. Furthermore, the set V_x is also Borel measurable for any $x \in X$ by similar reasoning to V. Using (6.2), we then obtain

$$\mathbb{E}[B(f(x)+\mathbf{v})] \leq (B(x) + c - \xi)\mathbb{P}[\mathbf{v} \in V_x] + \overline{B}\mathbb{P}[\mathbf{v} \in V_x^c], \tag{6.5}$$

where the first term on the right-hand side of (6.5) follows from the definition of V_x in (6.4), and the second by the uniform boundedness condition on B. It remains to show that $(B(x) + c - \xi)\mathbb{P}[\mathbf{v} \in V_x] + \overline{B}\mathbb{P}[\mathbf{v} \in V_x^c] \leq B(x) + c$. To this end, it suffices to show that

$$-\xi\mathbb{P}[\mathbf{v} \in V_x] + \overline{B}\mathbb{P}[\mathbf{v} \in V_x^c] \leq 0 \tag{6.6}$$

holds, since $B(x) + c$ is non-negative and $\mathbb{P}[\mathbf{v} \in V_x] \leq 1$ by definition. Substituting the two inequalities $\mathbb{P}[\mathbf{v} \in V_x] \geq \mathbb{P}[\mathbf{v} \in V] \geq 1 - \epsilon$ and $\mathbb{P}[\mathbf{v} \in V_x^c] \leq \mathbb{P}[\mathbf{v} \in V^c] \leq \epsilon$ into the left-hand side of (6.6) we obtain $-\xi\mathbb{P}[\mathbf{v} \in V_x] + \overline{B}\mathbb{P}[\mathbf{v} \in V_x^c] \leq -\xi(1-\epsilon) + \overline{B}\epsilon$, whose right-hand side is less than or equal to zero due to the fact that $\xi \geq \overline{B}\frac{\epsilon}{1-\epsilon}$. This concludes the proof of the lemma. □

Lemma 6.1 is enabled by the chance-constraint tightening variable ξ through the condition that $\xi \geq \overline{B}\frac{\epsilon}{1-\epsilon}$. A reader may question how to choose ξ and ϵ. Choosing ϵ is a trade-off between assigning less probability mass to the uniform upper bound, which is desirable as the uniform upper bound represents a worst-case for barrier value at the next step, and the amount of data required when applying the scenario approach theory. In our experiments (see Section 6.5), we employ $\epsilon = 0.005$. Once ϵ is chosen, the optimal choice of ξ to minimize c is $\xi = \overline{B}\frac{\epsilon}{1-\epsilon}$, which follows from the fact that ξ is on the left-hand side of the inner inequality of V (and V_x) with c on the right-hand side and $\overline{B}\frac{\epsilon}{1-\epsilon}$ is the smallest allowed value of ξ.

An immediate consequence of Lemma 6.1 is the fact that we can obtain an inner approximation of the optimisation problem (3.10) in terms of a chance-constrained optimisation problem.

Theorem 6.1. *Consider the dynamical system given in System (2.34). Then, we have that for all $\epsilon \in (0,1)$ and positive integers K and $\overline{B} \geq 1$ such that $\xi \geq \overline{B}\frac{\epsilon}{1-\epsilon}$, the feasible set of*

$$\begin{aligned} \min_{\theta \in \Theta, \eta, c} \quad & \eta + cK \\ \text{s.t.} \quad & \text{(3.1a)}, \quad \text{(3.1b)}, \quad \eta \geq 0, \quad c \geq 0, \\ & B(x) \leq \overline{B}, \ \forall x \in \mathbb{R}^n \\ & \mathbb{P}[\mathbf{v} \in V] \geq 1 - \epsilon, \end{aligned} \tag{6.7}$$

where V is defined as in (6.3), is contained in the feasible set of (3.10).

Proof. Conditions (3.1a) and (3.1b) are shared between (3.10) and the chance-constrained barrier program (6.7). Thus, it is sufficient to show that the chance-constraint of (6.7) implies that Condition (3.1c) is satisfied. By construction B is uniformly bounded by $\overline{B}$ and by assumption we have $\xi \geq \overline{B}\frac{\epsilon}{1-\epsilon}$. Finally, due to the chance-constraint, it holds that $\mathbb{P}[\mathbf{v} \in V^c] \leq \epsilon$, and thus the conditions of Lemma 6.1 are satisfied. Therefore, if the chance-constraint of (6.7) is satisfied, then Condition (3.1c) is satisfied. □

Unfortunately, the computational burden of solving (6.7) is not negligible, mainly due to the quantification over all $x \in X_S$ and due to the non-linear function f. In Subsection 6.2.1, we will show how the convex relaxations presented in Subsection 2.2.2 can alleviate this burden.

6.2.1. OVER-APPROXIMATING NON-LINEAR DYNAMICS

To relax the non-linearities in (6.7) as a step towards enabling efficient (data-driven) synthesis of SBFs, we use the techniques introduced in Subsection 2.2.2 to construct an uncertain PWA system that over-approximates (2.34), also called an infinite-state abstraction. This relaxation reduces (6.7) to the more structured and convex class of chance-constrained robust LP programming, which in the next Section 6.3 is further relaxed to a regular LP problem using scenario approach and duality theory. Formally, an uncertain PWA over-approximation of System (2.34) is described as follows.

$$\mathbf{x}[k+1] \in F(\mathbf{x}[k]) + \mathbf{v}[k], \tag{6.8}$$

where $F : X \rightrightarrows \mathbb{R}^n$ is a set-valued mapping defined as $F(x) = \{\hat{f}(x,\alpha) : \alpha \in [0,1]\}$, with $\hat{f}$ being an uncertain PWA function with the uncertain parameter α and such that for any $x \in X$ it holds that $f(x) \in F(x)$. Such a definition of an uncertain, or non-deterministic, PWA over-approximation of System (2.34) guarantees that between System (2.34) and PWA System (6.8) there is a *behavioural inclusion* relation, that is, for any $x \in X$ there exists $\alpha \in [0,1]$ such that $\hat{f}(x,\alpha) = f(x)$. Intuitively, we would like to synthesise an SBF for System (6.8) and rely on the behavioural relation described above to ensure that the synthezised SBF also is an SBF for System (2.34). This is what we show in the following Theorem, extending Theorem 6.1.

Theorem 6.2. *Consider the dynamical system given in* (2.34) *and assume access to an uncertain PWA over-approximation* (6.8) *of the system. Then, we have that for all* $\epsilon \in (0,1)$ *and positive integer* K*, if there exist* $\overline{B} \geq 1$ *and* $\xi \geq \overline{B}\frac{\epsilon}{1-\epsilon}$*, then the feasible set of*

$$\begin{aligned} \min_{\theta\in\Theta,\eta,c} \quad & \eta + cK \\ \text{s.t.} \quad & (3.1a), \quad (3.1b), \quad \eta \geq 0, \quad c \geq 0, \\ & B_\theta(x) \leq \overline{B}, \ \forall x \in \mathbb{R}^n \\ & \mathbb{P}[\mathbf{v} \in V] \geq 1-\epsilon, \end{aligned} \tag{6.9}$$

where

$$V = \{v \in \mathbb{R}^m \ : \ B_\theta(y+v) + \xi \leq B_\theta(x) + c, \ \forall x \in X_s, \ \forall y \in F(x)\} \tag{6.10}$$

is contained in the feasible set of (3.10)*.*

Proof. Conditions (3.1a) and (3.1b) are imposed directly in the uncertain chance-constrained barrier program (6.9). Since $f(x) \in F(x)$ implies that there exists an $\alpha \in [0,1]$ such that $\hat{f}(x,\alpha) = f(x)$, it holds that $B(f(x)+\mathbf{v}) \leq \sup_{y\in F(x)} B(y+\mathbf{v})$. Therefore, the chance-constraint of (6.9) implies the chance-constraint of (6.7). By Theorem 6.1 and transitivity of the subset relation, the feasible set of (6.9) is contained in the feasible set of (3.10). □

6.3. Data-driven synthesis

While (6.9) is a chance-constrained robust LP problem and significantly simpler than the non-linear (6.7), it is still problematic for two reasons: (i) we do not know the distribution $p_{\mathbf{v}}$ or the associated measure $\mathbb{P}_{\mathbf{v}}$, and (ii) even if we knew $p_{\mathbf{v}}$, the form does not admit an analytical solution or even derivative wrt. θ. Instead, our approach is to employ scenario approach theory, given a dataset $D = \{v_1, \dots, v_N\}$ sampled i.i.d. according to $p_{\mathbf{v}}$, to relax the problem to a robust LP problem, at the expense of instead providing PAC-like safety guarantees. Later in this section, we will transform the robust LP problem to a regular LP problem using duality theory.

First, to reformulate (6.9) as a robust LP problem, we need to introduce some mathematical notation. Let $Q = \{q_1, \dots, q_\ell\}$ be the partition of the state space associated with a barrier function and denote four collections of indices by

$$\begin{aligned} & I = \{1, \dots, \ell\}, \\ & I_{X_u} = \{i \in I : q_i \cap (\mathbb{R}^n \setminus X_s) \neq \emptyset\}, \quad I_{X_0} = \{i \in I : q_i \cap X_0 \neq \emptyset\}, \\ & I_{X_s} = \{i \in I : q_i \cap X_s \neq \emptyset\}, \end{aligned} \tag{6.11}$$

which represent the collection of all indices, and the elements of Q with non-empty intersections with the initial state, the safe set, and unsafe set, respectively. Finally, for each pair $(i,j) \in I_{X_s} \times I$, we denote the set

$$q_{ij}(v) = \left\{x \in q_i : \exists y \in F(x), y + v \in q_j\right\}, \tag{6.12}$$

representing the subset of q_i with $i \in I_{X_s}$ that is mapped to q_j under a given realization of the noise. A pictorial example of $q_{ij}(v)$ can be found in Figure 6.1. Leveraging the results of Theorem 6.2 and Proposition 2.7 and using the notation we have introduced so far, we obtain the following intermediate result. The goal of Lemma 6.2 below is two-fold: (i) to impose piecewise constraints on the barrier synthesis and (ii) apply scenario approach theory to obtain a tractable solution in the face of an unknown noise distribution. The intuition behind Lemma 6.2 is to impose the chance-constraint in Problem (6.9) independently for each pair (i,j) and sample v so that the constraint becomes $B_j(\hat{f}_i(x,\alpha)+v)+\xi \leq B_i(x)+c$ for all $(v,i,j) \in D \times I_{X_s} \times I$, $\alpha \in [0,1]$, and $x \in q_{ij}(v)$. The resulting program is a robust LP due to linearity in θ and quantification over α and x.

Lemma 6.2. *Let $D = \{v_1, \dots, v_N\}$ be a collection of N independent samples from the noise distribution $p_{\mathbf{v}}$. Fix $\epsilon \in (0,1)$, $\overline{B} \geq 1$, and $\xi \geq \overline{B}\frac{\epsilon}{1-\epsilon}$, and consider the scenario optimization program*

$$
\begin{aligned}
\min_{z} \quad & \eta + cK \\
\text{s.t.} \quad & \eta \geq 0, \quad c \geq 0, \\
& B_i(x) \in [0, \overline{B}], && \forall i \in I, x \in q_i, \\
& B_i(x) \geq 1, && \forall i \in I_{X_u}, x \in q_i, \\
& B_i(x) \leq \eta, && \forall i \in I_{X_0}, x \in q_i \cap X_0, \\
& B_j(y+v)+\xi \leq B_i(x)+c, && \forall v \in D, i \in I_{X_s}, j \in I, x \in q_{ij}(v), y \in F(x)
\end{aligned}
\tag{6.13}
$$

where $z = (\eta, c, \theta)$ is the collection of decision variables in (6.9) and $d = |z| = 2+\ell(n+1)$ is the number of decision variables. Then, with probability at least $1-\beta$, where β is the right-hand side of the inequality in Proposition 2.7, the optimal solution of the scenario barrier program (6.13) satisfies the conditions of Definition 3.1.

The relevance of Lemma 6.2 towards the general framework proposed in the paper can be summarised by two main points: (1) Lemma 6.2 establishes a sufficient condition to enforce the constraints of (6.9) by restricting the attention to each element of the partition Q individually; (2) it allows us to use the duality results of Subsection 2.2.3 to obtain a computationally tractable reformulation of the optimization problem in Lemma 6.2 into a robust LP program.

For the remainder of this section, we will discuss how to obtain this finite, yet equivalent, representation of (6.13). The reformulation of the first three semi-infinite constraints of (6.13) follows standard dualization arguments, i.e. using Proposition 2.6, thus we defer their explicit reasoning to the proof of Corollary 6.1. The remaining constraint $B_j(y+v)+\xi \leq B_i(x)+c$ for all $v \in D$, $i \in I_{X_s}$, $j \in I$, $x \in q_{ij}(v)$, and $y \in F(x)$ requires more care, yet Lemma 6.2 also enables a computationally tractable reformulation of this. To this end, consider any $(i,j) \in I_{X_s} \times I$ and $v \in D$. A challenge in reformulating this constraint as a linear constraint is that the set $q_{ij}(v)$ is not a polyhedron or even convex (see Figure 6.2). For now, we assume access to a polyhedral over-approximation $q_{ij}(v) \subset \overline{q}_{ij}(v) = \{x :$

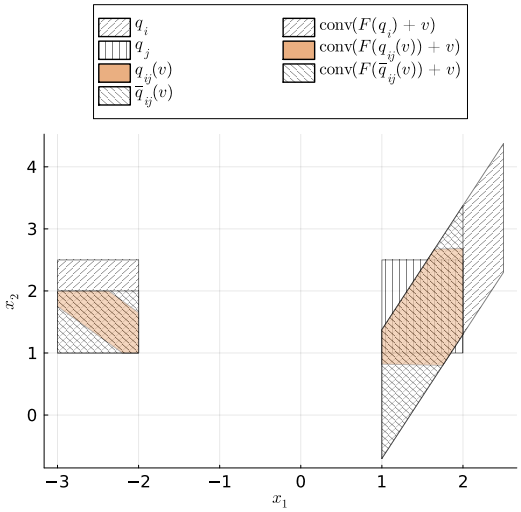


Figure 6.1: Given two regions q_i, q_j and a realisation of the noise v, the set $q_{ij}(v)$ represents the subset of $x \in q_i$ such that $\hat{f}(x,\alpha) + v \in q_j$ for some $\alpha \in [0,1]$. In other words, $q_{ij}(v)$ is the subset of q_i that can reach q_j given the realisation of the noise v. Unfortunately, $q_{ij}(v)$ is not easily computed (the example above is an exception, see Figure 6.2) and thus in Subsection 6.4.1, we will compute an over-approximation $\overline{q}_{ij}(v)$.

$H_{ijv}x \leq h_{ijv}\}$. Then if we impose the constraint for all $x \in \overline{q}_{ij}(v)$, then it trivially follows that it also holds for all $x \in q_{ij}(v)$. We will defer the discussion of how to compute $\overline{q}_{ij}(v)$ to Subsection 6.4.1.

Proposition 6.1. *Let $(i,j) \in I_{X_s} \times I$ and $v \in D$ be given. Assume access to two affine functions such that $\underline{A}_i x + \underline{b}_i \leq F(x) \leq \overline{A}_i x + \overline{b}_i$ for all $x \in q_i$. Then it holds that $B_j(y+v) + \xi \leq B_i(x) + c$ for all $x \in q_{ij}(v)$ and $y \in F(x)$ if and only if the following constraints hold*

$$h_{ijv}^\top \underline{\lambda}_{ijv} \leq w_i - w_j - u_j^\top(\underline{b}_i + v) + c - \xi, \tag{6.14a}$$

$$H_{ijv}^\top \underline{\lambda}_{ijv} = \underline{A}_i^\top u_j - u_i, \tag{6.14b}$$

$$h_{ijv}^\top \overline{\lambda}_{ijv} \leq w_i - w_j - u_j^\top(\overline{b}_i + v) + c - \xi, \tag{6.14c}$$

$$H_{ijv}^\top \overline{\lambda}_{ijv} = \overline{A}_i^\top u_j - u_i, \tag{6.14d}$$

where $\underline{\lambda}_{ijv}$, $\overline{\lambda}_{ijv}$ are a non-negative dual variables.

Proof. The proof strategy is similar to the proof of Proposition 2.2 where a (linearly) uncertain affine transformation is characterized by the vertices of the transformation. To this end, first, fix any $(i,j) \in I_{X_s} \times I$ and $v \in D$. By the linearity of the uncertainty parameter α in $\hat{f}_i(x,\alpha)$ for any $x \in q_i$, it holds that $B_j(y+v)+\xi \leq B_i(x)+c$ for

all $x \in q_{ij}(v)$ and $y \in F(x)$ if and only if for all $x \in q_{ij}(v)$ the following constraints hold

$$B_j(\underline{A}_i x + \underline{b}_i + v) + \xi \leq B_i(x) + c \quad \text{and} \tag{6.15a}$$

$$B_j(\overline{A}_i x + \overline{b}_i + v) + \xi \leq B_i(x) + c. \tag{6.15b}$$

Finally, we apply the dualization argument of Proposition 2.6 to replace the semi-infinite robust constraints by finite constraints. □

Collecting together all finite sets of constraints, a finite representation of the semi-infinite program (6.13) is given as

$$\begin{aligned}
\min_{z} \quad & \eta + cK \\
\text{s.t.} \quad & \eta \geq 0,\ c \geq 0, \\
\text{(Non-negativity)} \quad & h_i^\top \underline{\mu}_i \leq w_i,\ H_i^\top \underline{\mu}_i = -u_i, \text{ for all } i \in I, \\
\text{(Uniform upper bound)} \quad & h_i^\top \overline{\mu}_i \leq \overline{B} - w_i,\ H_i^\top \overline{\mu}_i = u_i, \text{ for all } i \in I, \\
\text{(Unsafe set)} \quad & h_i^\top \psi_i^u \leq w_i - 1,\ H_i^\top \psi_i^u = -u_i, \text{ for all } i \in I_{X_u}, \\
\text{(Initial set)} \quad & h_{i0}^\top \psi_i^0 \leq \eta - w_i,\ H_{i0}^\top \psi_i^0 = u_i, \text{ for all } i \in I_{X_0}, \\
\text{(One-step constraints)} \quad & h_{ijv}^\top \underline{\lambda}_{ijv} \leq w_i - w_j - u_j^\top (\underline{b}_i + v) + c - \xi, \\
& H_{ijv}^\top \underline{\lambda}_{ijv} = \underline{A}_i^\top u_j - u_i, \\
& h_{ijv}^\top \overline{\lambda}_{ijv} \leq w_i - w_j - u_j^\top (\overline{b}_i + v) + c - \xi, \\
& H_{ijv}^\top \overline{\lambda}_{ijv} = \overline{A}_i^\top u_j - u_i, \text{ for all } (i,j) \in I_{X_s} \times I, v \in D
\end{aligned} \tag{6.16}$$

where $\underline{\mu}_i, \overline{\mu}_i, \psi_i^u, \psi_i^0, \underline{\lambda}_{ijv}, \overline{\lambda}_{ijv}$ are all non-negative dual variables. (H_{i0}, h_{i0}) denotes the half-space representation of $q_i \cap X_0$.

In Corollary 6.1, we combine the introduced results and show how probabilistic safety can be computed using the scenario approach using samples of the random variable $\mathbf{v}$.

Corollary 6.1. *Assume that $D = \{v_1, \ldots, v_N\}$ is a collection of N independent samples from the noise distribution $p_{\mathbf{v}}$. Fix $\epsilon \in (0,1)$, $\overline{B} \geq 1$, and $\xi \geq \overline{B}\frac{\epsilon}{1-\epsilon}$, and let β be defined as Proposition 2.7 where $d = 2 + \ell(n+1)$ is the number of (non-dual) decision variables in (6.16). Consider the optimal (primal) solution $z^\star(D) = (c^\star, \eta^\star, \theta^\star)$ of (6.16). Then, with confidence $1-\beta$, it holds that*

$$P_{\text{safe}} \geq \max(1 - (\eta^\star + c^\star K), 0). \tag{6.17}$$

Proof. It is enough to show an equivalence between (6.13) and (6.16). To do that, in what follows, we show that each semi-infinite constraint of (6.13) is equivalent to a set of constraints of (6.16). We start by showing the equivalence for the initial set

constraint (condition $B_i(x) \leq \eta$ for all $i \in I_{X_0}$ and $x \in q_i \cap X_0$ in (6.13)) to the initial set constraints in (6.16). For this, let I_{X_0} be the index set for regions intersecting with X_0 and consider its corresponding partition $q_{i0} = q_i \cap X_0 = \{x : H_{i0}x \leq h_{i0}\}$, $i \in I_{X_0}$. Then, the condition $B(x) \leq \eta$ for all $x \in q_{i0}$ can be written equivalently as $h_{i0}^{\mathsf{T}}\psi_i^0 \leq \eta - w_i$ and $H_{i0}^{\mathsf{T}}\psi_i^0 = u_i$ where ψ_i^0 is a vector-valued non-negative dual variable. This follows from the fact that within the region q_{i0} the function $B(x) = B_i(x)$ is affine and consequently, by relying on Proposition 2.6, we can rewrite this robust constraint as two regular linear constraints, with the dual variable ψ_i^0. The equivalence for the non-negativity, uniform upper bound, and unsafe set robust constraints (see Definition 3.1) to regular linear constraints follow a similar argument. Hence, for brevity, we omit the reformulation of these. The dualization of the remaining constraint follows directly from Proposition 6.1. Thus, the proof is concluded by noting that the cost to minimize is the same in both (6.13) and (6.16). □

Remark 6.2. *Corollary 6.1 can easily be modified to the setting where the noise samples are not known exactly, but they are only known to lie in specified sets. This can be done by simply adding a quantifier for each of these sets in* (6.16).

Thus, by solving (6.16), we can certify probabilistic safety with high confidence. Note that naïvely trying to solve (6.16) can soon become intractable on contemporary hardware, due to both memory requirements and computational time. In particular, the cardinality of the Cartesian product of I_{X_s}, I, and D can already be prohibitively large for relatively small systems. In the next section, we will discuss algorithmic strategies that make (6.16) computationally tractable.

6.4. Algorithms for program construction

In this section, we discuss three aspects that allows one to solve (6.13) efficiently. Namely, in Subsection 6.4.1, we discuss how to compute a polyhedral over-approximation of $q_{ij}(v)$. In Subsection 6.4.2, we introduce an a-priori sample discarding procedure for Problem (6.16), which guarantees the same optimal solution, but allowing one to consider less samples in the optimization problem. Finally, in Subsection 6.4.3, we employ spatial indexing methods to efficiently find the set of triplets with a non-empty $q_{ij}(v)$. More specifically, we will rely on the fact that, often, for many triplets (i, j, v), the set $q_{ij}(v)$ is empty, thus the martingale condition is trivially satisfied. That is, $q_{ij}(v)$ is empty if $\mathrm{im}_i(q_i, v) \cap q_j = \emptyset$ where the image for region q_i is defined as

$$\mathrm{im}_i(q_i, v) = \{y + v : x \in q_i, y \in F(x)\}. \tag{6.18}$$

6.4.1. Over-approximation of $q_{ij}(v)$

As illustrated in Figure 6.1, computing a polyhedral over-approximation of $q_{ij}(v)$ is challenging, as $q_{ij}(v)$ is (possibly) non-convex due to the uncertain affine transformation F[1]. Furthermore, it is not sufficient to compute the convex hull for the

[1] If F is a deterministic affine transformation and q_j is a polyhedron, then $q_{ij}(v)$ is also a polyhedron and analytical methods for computation exist.

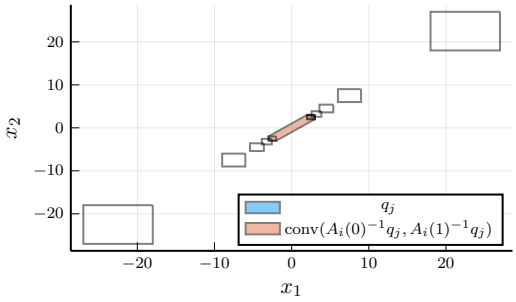


Figure 6.2: A pictorial example that $q_{ij}(v)$ is not necessarily convex. In this example, we have $\underline{A}_i = -I$, $\overline{A}_i = I$, and $\underline{b}_i = \overline{b}_i = 0$, which is a valid uncertain affine relaxation of the trivial function $f(x) = 0$ in the non-negative orthant. We consider the region $q_j = [2,3]^2$ and plot $A_i(\alpha)^{-1}q_j$ for 10 uniformly spread values of $\alpha \in [0,1]$. Note that in this case $\overline{A}_i = I$, hence $q_j = A_i(1)^{-1}q_j$ and q_j (blue) is contained in the convex hull (pink). $q_{ij}(v)$ can take on complex shapes and no method exists for exactly computing $q_{ij}(v)$. Therefore, we compute a sound over-approximation $\overline{q}_{ij}(v) \supset q_{ij}(v)$.

vertices of the uncertainty variable α, that is,

$$\mathrm{conv}\left(\left\{x \in q_i : \hat{f}_i(x,0) + v) \in q_j\right\} \cup \left\{x \in q_i : \hat{f}_i(x,1) + v) \in q_j\right\}\right), \tag{6.19}$$

as shown in Figure 6.1. However, we note that by definition $q_{ij}(v) \subseteq q_i$, that is, q_i is a (generally conservative) polyhedral over-approximation of $q_{ij}(v)$. Hence, our approach to find a polyhedral over-approximation of $q_{ij}(v)$, denoted $\overline{q}_{ij}(v)$, is to start from q_i and then iteratively removing subsets of $q_i \setminus q_{ij}(v)$. To accomplish this, we rely on repeated bisection. To simplify the presentation, in what follows, we assume that q_i is a hyperrectangle. Note, however, that the procedure generalises to compact polyhedra in half-space representation.

Figure 6.3 shows an example of the bisection algorithm for two regions q_i, q_j and a given sample v. The bisection is repeated twice along each axis, once to increase the lower bound, and once to decrease the upper bound. Note that $q_{ij}(v)$, although depicted in Figure 6.3, is unknown and possibly non-convex, and further we cannot readily check if $q_{ij}(v) \subset \overline{q}_{ij}(v)$. However, recall that $q_{ij}(v)$ is the subset of region q_i that under a realisation of the noise v reaches region q_j in one time step. As a result, we can instead check if $\mathrm{conv}(\overline{q}_{ij}(v) + v)$ intersects with q_j. By applying this algorithm, we compute an over-approximation $\overline{q}_{ij}(v)$ of $q_{ij}(v)$. Indeed, as $\overline{q}_{ij}(v)$ is an over-approximation, using $\overline{q}_{ij}(v)$ for the constraints in Proposition 6.1 yields a sound, although slightly conservative, solution. The details of our bisection-based are shown in Algorithm 6.1. The computational complexity of the algorithm, for a fixed number of bisection steps t per halfplane and assuming that q_i is hyperrectangular, is $O(2n\ell t)$.

6.4.2. CONVEX HULL OVER THE SAMPLE SET

To reduce the number of constraints of (6.16), observe that the constraints of the problem are affine in v. This implies that the active constraints, also known

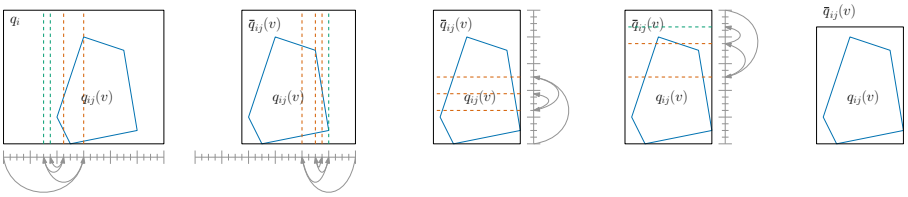


Figure 6.3: An example of the bisection algorithm to compute the over-approximation $\overline{q}_{ij}(v)$ of $q_{ij}(v)$. The set $q_{ij}(v)$ is unknown and possibly non-polyhedral, but q_i is a sound over-approximation. By bisection from either side (first the lower then the upper bound) along each axis we obtain a smaller over-approximation. We start by bisecting for x_1 (**(a)** and **(b)**) followed by x_2 (**(c)** and **(d)**). This results in the over-approximation $\overline{q}_{ij}(v)$ in **(e)**.

Algorithm 6.1 Bisection-based algorithm for computing a subset $\overline{q}_{ij}(v)$ of region q_i as an over-approximation of $q_{ij}(v)$.

Require: Source q_i, destination q_j, dynamics $\hat{f}_i$, noise realization v, and bisection steps t

Ensure: A region $\overline{q}_{ij}(v)$ such that $q_{ij}(v) \subseteq \overline{q}_{ij}(v) \subseteq q_i$

$\overline{q}_{ij}(v) := q_i$
for k from 1 to n **do** CommentFor each axis
  $\underline{l}_k, \overline{l}_k := l_k, u_k$ ▷ *Increase lower bound*
  for s from 1 to t **do**
    $c_k := \frac{\underline{l}_k + \overline{l}_k}{2}$
    $\overline{q}_{ij}(v)' := \overline{q}_{ij}(v) \cap \{x : x_k \leq c_k\}$
    if $\mathrm{im}_i(\overline{q}_{ij}(v)', v) \cap q_j = \emptyset$ **then**
      $\underline{l}_k := c_k$
    else
      $\overline{l}_k := c_k$
  $\underline{u}_k, \overline{u}_k := l_k, u_k$ ▷ *Decrease upper bound*
  for s from 1 to t **do**
    $c_k := \frac{\underline{u}_k + \overline{u}_k}{2}$
    $\overline{q}_{ij}(v)' := \overline{q}_{ij}(v) \cap \{x : x_k \geq c_k\}$
    if $\mathrm{im}_i(\overline{q}_{ij}(v)', \omega) \cap q_j = \emptyset$ **then**
      $\overline{u}_k := c_k$
    else
      $\underline{u}_k := c_k$

  $\overline{q}_{ij}(v) := \overline{q}_{ij}(v) \cap \{x : \underline{l}_k \leq x_k \leq \overline{u}_k\}$
return $\overline{q}_{ij}(v)$

as support constraints, will always belong to the vertices of the convex hull over $D = \{v_1, \dots, v_N\}$, enabling a great reduction in the number of constraints in the program. That is, we can impose the constraint only on the vertices of the convex hull of the dataset $\overline{D} = \text{vert}(\text{conv}(D))$.

Proposition 6.2. *Let $\mathcal{Z}(D)$ denote the feasible set of Problem (6.16) with respect to a sample set D. Then we have $\mathcal{Z}(D) = \mathcal{Z}(\overline{D})$.*

Proof. First, observe that $\mathcal{Z}(D)$ and $\mathcal{Z}(\overline{D})$ share all constraints except $h_{ijv}^\top \underline{\lambda}_{ijv} \leq w_i - w_j - u_j^\top(\underline{b}_i + v) + c - \xi$ for all $(v, i, j) \in D \times I_{X_s} \times I$ (and similarly for the upper bound) and that v enters the constraint linearly. Therefore, we can rewrite the constraint with a convex combination of D and of $\overline{D}$. Now, since $\overline{D}$ is the vertices of the convex hull of D, we have $\text{conv}(D) = \text{conv}(\overline{D})$ and thus, the feasible sets $\mathcal{Z}(D)$ and $\mathcal{Z}(\overline{D})$ coincide. □

Proposition 6.2 guarantees that to solve Problem (6.16) we can reduce the number of constraints by simply considering the samples in $\overline{D}$. In the experiments conducted (see Section 6.5), we find that generally, in practice, the cardinality of $\overline{D}$ is orders of magnitude lower than the cardinality of D. Thus, this can greatly improve the efficiency of our approach.

Remark 6.3. *The method presented in this subsection was discovered independently, but is similar to the method presented in [130] with the exception of that our method requires an exact convex hull rather than an approximate convex hull. The proposed method for sample reduction works for any scenario program that is affine in the random variable* **v**.

6.4.3. SPATIAL INDEXING FOR INTERSECTION SEARCH

Constructing (6.16) efficiently is also a non-trivial problem due to memory limits. In fact, a naïve approach to construct the problem is to iterate over all triplets (i, j, v) in $I_{X_s} \times I \times D$, check if $q_{ij}(v) \neq \emptyset$, and add a set of constraints if the test is positive. This approach is only tractable for smaller problems as it has a time complexity of $O(\ell^3)$. To reduce the complexity, we can exploit methods from database theory; namely spatial indexing, which is the structuring and querying of spatially distributed data, such as maps, with improved computational complexity [131].

To apply spatial indexing to our setting, we must first establish the data and query. It holds that $q_{ij}(v) \neq \emptyset$ only if $\text{im}_i(q_i, v) \cap q_j \neq \emptyset$. Hence, if we search for regions q_j that intersect with the image $\text{im}_i(q_i, v)$, we find all pairs (i, j) such that $q_{ij}(v) \neq \emptyset$. For spatial indexing, we focus on R-trees as it is a well-studied and widely available method [131]. The idea is to structure the data in a tree structure and at each node store a Minimum Bounding Rectangle for the nodes below. Then, querying the tree for the intersection with another region proceeds recursively down the tree, where it is only necessary to search down a branch if the Minimum Bounding Rectangle of the branch and the query intersect, which is an inexpensive operation by the separating hyperplane theorem [93]. Figure 6.4 shows an example of an R-tree for a partitioned state space. This method improves complexity by efficiently searching for relevant triplets (i, j, v).

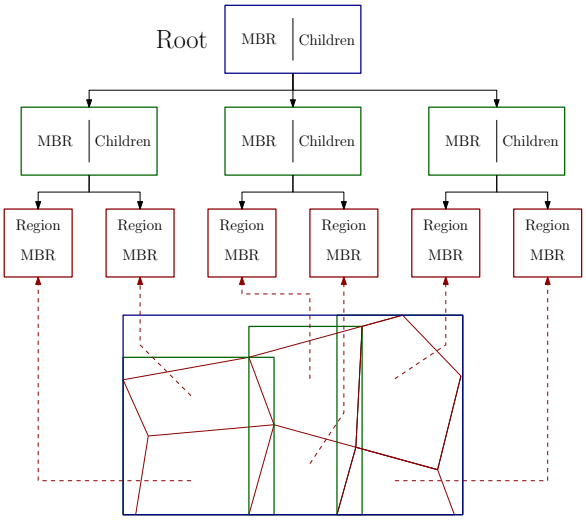


Figure 6.4: An example of an R-tree applied to a partitioned state space to allow efficient search for regions intersecting with $\text{im}_i(q_i, v)$. The rectangles are the Minimum Bounding Rectangle for each node in the tree.

In summary, to use the framework to compute data-driven safety certificates for non-linear systems: let the nominal dynamics f, initial and safe set X_0, X_s, a horizon K, and a dataset of samples D be given. Then start by abstracting the non-linear dynamics f to uncertain PWA dynamics $\hat{f}$ using Linear Bound Propagation (LBP) techniques. Compute the vertices $\overline{D}$ of the convex hull of D and discard all interior points. For each region q_i, find, using an R-tree, regions that intersect with the image of the dynamics $\text{im}_i(q_i, v)$ and add constraints accordingly. Solve the LP problem (6.16), then the solution $z^\star(\overline{D}) = (c^\star, \eta^\star, c^\star)$ is a safety certificate $P_{\text{safe}} \geq 1-(\eta^\star + c^\star K)$ with confidence $1-\beta$ where β is defined as in Proposition 2.7.

6.5. Empirical Evaluation

To investigate the efficacy of our approach, we implemented our framework in Julia[2] and performed an empirical analysis on various benchmarks. The experiments have been conducted on a computer with an Intel i7-1365U CPU and 16GB RAM, running Linux 6.8.0-52-generic. We start by describing the benchmarks followed by the results. For comparison with state-of-the-art, we consider the Sample Average Approximation (SAA) method proposed in [127], [33][3]. As this SBF synthesis method has been developed specifically for linear or polynomial systems, namely through SoS optimisation, in order to provide a general baseline, in the case of non-

[2]Code is available at https://github.com/DAI-Lab-HERALD/scenario-barrier under GNU GPLv3 license.

[3]We compare only with the SAA and not the scenario approach for robust constraints such that the confidence level for both methods pertain to the probability that the c-martingale constraint in Condition (3.1c) is satisfied.

polynomial and/or uncertain systems, we combine it with the method proposed in [99], to find a valid SBF in the general case. We also compare against [42], which is a framework for data-driven abstraction.

6.5.1. BENCHMARKS

The simplest system considered is an uncertain 1D linear system $\mathbf{x}[k+1] = \mathbf{x}[k] + b(\alpha) + \mathbf{v}$ where $b(\alpha) = -0.05 + 0.1\alpha$, i.e. the uncertainty is in $b(\alpha)$ with uncertainty variable $\alpha \in [0, 1]$. Starting from a set $X_0 = \{|x| \leq 0.5\}$, the goal is to stay within a larger set $X_s = \{|x| \leq 2.5\}$ for a horizon $K = 10$. The distribution of the noise is a zero-mean normal distribution with standard deviation 0.01.

We also consider a 2D system from [132] representing the longitudinal dynamics of a drone. The coordinates x_1, x_2 are the position and velocity, respectively, and the system has the following dynamics

$$\mathbf{x}[k+1] = \begin{pmatrix} 1 & h \\ 0 & 1 - 0.1h \end{pmatrix} \mathbf{x}[k] + \mathbf{v}[k], \tag{6.20}$$

where $\mathbf{v}[k]$ has a zero-mean normal distribution with diagonal covariance of $[0.01\ 0.01]$. The variable h represents the discretisation step, which is set to $h = 1$. As with the 1D linear system, we certify safety for a horizon $K = 10$.

The third benchmark represents a model of a vehicle travelling down a straight road when it experiences an (uncertain) gust of wind. The coordinates x_1, x_2 represent, respectively, the longitudinal and lateral position of the vehicle. Similar to the drone system, h represents the discretisation step. The goal is certify the probability of staying on the road $X_s = \{|x_1| \leq 2.5\}$ for a horizon $K = 10$, when the system evolves according to the following dynamics

$$\mathbf{x}[k+1] = \begin{pmatrix} 1 & 0 \\ 0 & 0.95 \end{pmatrix} \mathbf{x}[k] + \begin{pmatrix} \frac{50}{3.6} \cdot h \\ \frac{1}{2} a_{lat} \cdot h^2 \end{pmatrix} + \mathbf{v}[k], \tag{6.21}$$

where $h = 1$, and $a_{lat} = 0$ for regions where $x_1 \leq 80$ or $x_1 \geq 120$, and $a_{lat} \in [0.0913, 0.364]$ for regions where $80 \leq x_1 \leq 120$. The noise $\mathbf{v}[k]$ has a zero-mean normal distribution with diagonal covariance of $[0.01\ 0.01]$.

A 3D benchmark with linear dynamics is the room temperature verification from [33]. The dynamics are $\mathbf{x}[k+1] = A\mathbf{x}[k] + b + \mathbf{v}[k]$ where

$$A = \begin{pmatrix} 1 - h(\zeta + \zeta_e) & h\zeta & 0 \\ h\zeta & 1 - h(2\zeta + \zeta_e) & h\zeta \\ 0 & h\zeta & 1 - h(\zeta + \zeta_e), \end{pmatrix},$$

$b = \begin{pmatrix} h\zeta_e T_e & h\zeta_e T_e & h\zeta_e T_e \end{pmatrix}^\mathsf{T}$ with $h = 5$, $\zeta = 0.0062$, $\zeta_e = 0.008$, $T_e = 10$. $\mathbf{v}$ has a zero-mean normal distribution with diagonal covariance of $[0.0001\ 0.0001\ 0.0001]$. The initial set is $X_0 = [17, 18]^3$, the safe set is $X_s = [17, 29]^3$, and the time horizon is $K = 3$.

While the previous models were linear, we also considered non-linear models. In particular, we consider Neural Network Dynamic Models (NNDMs) with 1 and 2

hidden layers of 64 neurons each modelling a pendulum taken from [99]. Second is the 3D Dubin's car benchmark from [75] for a time horizon $K = 10$. Dubin's car is a unicycle model where the state is (x, y, ϕ) with ϕ being the heading of the vehicle. We consider a grid-based partitioning of 10 segments along each axis, i.e. 1000 regions. The noise is only applied to the last dimension and has a normal distribution with mean of $60 \cdot \frac{\pi}{180} \approx 1.053$ and standard deviation 0.1. Finally, we consider the lane keeping benchmarking from [33] where the dynamics are

$$\mathbf{x}[k+1] = \mathbf{x}[k] + hv \begin{pmatrix} \cos(\mathbf{x}[k]_3 + b) \\ \sin(\mathbf{x}[k]_3 + b) \\ \sin(b)/l_r \end{pmatrix} + \mathbf{v}[k], \tag{6.22}$$

with $b = \frac{l_r \tan^{-1}(\delta_f)}{l_r + l_f}$, $\delta_f = 5$, $l_r = l_f = 1.384$, $v = 5$, and $h = 0.1$. The noise $\mathbf{v}[k]$ has a zero-mean normal distribution with diagonal covariance of $[0.0001\ 0.0001\ 0.000001]$. The initial set is $X_0 = [1, 2] \times [-0.5, 0.5] \times [-0.005, 0.005]$, the safe set is $X_s = [1, 10] \times [-6, 6] \times [-0.05, 0.05]$, and the time horizon is $K = 3$.

6.5.2. Results

Table 6.1 reports the results across all benchmarks. Both the safety probability and the computation time are reported averaged over 100 trials, and for all cases, the number of samples is selected to ensure a confidence $1 - 10^{-9}$. We observe that, depending on the system, the method can certify safety to $> 99\%$ with high confidence, e.g. 99.5% certified safety for the NNDM model of a pendulum with 2 layers and 64 neurons. This certification requires relatively few regions of 10-30 segments per axis. Comparing the NNDM pendulum model with 2 and 3 layers (1 and 2 hidden layers, respectively), the complexity of the nominal dynamics impacts both computation time and certifiable safety, e.g. 99.5% safety probability in 45.0s vs 97.6% safety probability in 78.5s for 480 regions, 2 and 3 layers respectively. This behaviour can be explained by LBP computing wider uncertain affine transformations lead to more non-empty $Q_{ij}(\omega)$. Remarkably, the linearity of the underlying system has little impact on the certifiable safety. This is observed in that both the 1D linear and drone systems exhibit uncertain linear behaviour, yet the 1D linear system is certifiable to 50.8% safety while the drone is certifiable to 99.5%. Furthermore, for the largest systems considered, Dubin's car and lane keeping, which both include trigonometric functions, safety is certified to 99.9%.

To compare against state-of-the-art, we report in Table 6.2 the certified safety by using our method, SAA [127], and data-driven abstractions [42], averaged over 100 trials. We observe that for our method, certifying for higher confidence (10^{-2} to 10^{-4}) has a small impact in the computation time (1.2% faster for the 1D linear system and 0.5% slower for the pendulum model) and achieves similar levels of certified safety. In contrast, for SAA, going from 10^{-2} to 10^{-3} increases the computation time between 1.3x and 1.9x and become infeasible already for $\beta = 10^{-4}$. The achieved level of certified safety is also marginally better with our proposed method (e.g. 0.995 vs 0.903 for the pendulum model). When comparing to data-driven abstractions, the certified probability of safety is comparable or marginally

Table 6.1: Certified safety and computation time using our approach. Results are reported as the mean over 100 iterations for each case study. n is the dimensionality of the system and ℓ is the number of regions. P_{safe} is the certified level of safety for $\beta = 10^{-9}$ where $1 - \beta$ is the level of confidence.

System	n	ℓ	P_{safe}	Time (s)
Linear	1	27	0.511	0.359
Drone	2	37	0.995	44.7
Vehicle	2	18	0.606	1.89
		42	0.710	4.28
		54	0.828	5.24
		150	0.995	13.6
Room temperature	3	2197	0.998	356
Pendulum (NNDM)	2	120	0.346	11.4
- 2 layers		240	0.752	28.3
- 64 neurons		480	0.995	71.4
Pendulum (NNDM)	2	120	0.265	28.33
- 3 layers		240	0.376	68.9
- 64 neurons		480	0.980	130
Dubin's car	3	1000	0.999	321
Lane keeping	3	2520	1.000	33.6

better. However, the computation is considerably faster; 17x for the drone benchmark and 2x for the room temperature benchmark.

6.6. CONCLUSION

This chapter studies SBF-based safety verification in the context where the distribution of the noise is unknown. This is important as the distribution is rarely Gaussian or known exactly as assume in the previous Chapters 4 and 5. The lack of knowledge about the random variable that is the noise requires a shift to data-driven verification methods, which relies on sampling this variable. To this end, we propose a Scenario Approach theory-based method. In particular, we reframe the supermartingale condition (3.1c) as a chance-constraint by splitting the expectation integral and managing the majority of the probability with a chance-constraint and a constraint tightening variable, and the remainder by a uniform upper bound on the barrier. This reformulation is what allows for the use of the Scenario Approach, and thus a data-driven verification method. To manage arbitrary continuous dynamics, the system under study is abstracted to an uncertain PWA system with the use of CROWN. In particular, given a partitioning, sound linear relaxations are computed within each region using the methodology in Subsection 2.2.2.

A consequence of data-driven verification is that the bound on the probability of safety is associated with formal (probabilistic) confidence $1 - \beta$. This should be interpreted as *"if drawing another sample from the same noise distribution, the probability of synthesizing another SBF with a lower probability of safety is less than β"*. With the proposed method, β is in the range of 10^{-6} to 10^{-9}; in other words,

Table 6.2: Comparison of our method against Sample Average Approximation (SAA) combined with the method presented in [99], to synthesise SBFs in a data-driven fashion, and against [42] for data-driven abstractions. We do not compare with [42] on the pendulum model, since the Matlab implementation does not support uncertain PWA dynamics. Results are reported as the mean over 100 iterations for each case study. $1-\beta$ is the confidence in the certificate and P_{safe} is the certified level of safety. OOM means the certification procedure exceeded available memory. The number of samples required for our method and [42] increases from $\approx 3.1\cdot 10^6$ to $\approx 3.4\cdot 10^6$ with β for the drone and pendulum benchmarks and from $\approx 4.3\cdot 10^6$ to $\approx 4.6\cdot 10^6$ for the room temperature benchmark, while for SAA, the number grows from $8\cdot 10^6$ to $8\cdot 10^{13}$ as $\beta \to 10^{-9}$ for all benchmarks.

System	Method	β	P_{safe}	Time (s)
Drone	Ours	10^{-2}	0.995	0.403
		10^{-3}	0.995	0.361
		10^{-4}	0.995	0.392
		10^{-9}	0.995	0.398
	SAA	10^{-2}	0.995	0.595
		10^{-3}	0.995	0.798
		10^{-4}	-	OOM
		10^{-9}	-	OOM
	Data-driven abstraction	10^{-2}	0.989	6.84
		10^{-3}	0.988	6.39
		10^{-4}	0.986	6.63
		10^{-9}	0.981	6.70
Room temp.	Ours	10^{-2}	0.995	445
		10^{-3}	0.995	446
		10^{-4}	0.995	437
		10^{-9}	0.995	432
	SAA	10^{-2}	-	OOM
		10^{-3}	-	OOM
		10^{-4}	-	OOM
		10^{-9}	-	OOM
	Data-driven abstraction	10^{-2}	0.997	884
		10^{-3}	0.997	833
		10^{-4}	0.996	886
		10^{-9}	0.995	901
Pendulum (NNDM) - 2 layers - 64 neurons - ReLU act. - 480 regions	Ours	10^{-2}	0.995	38.6
		10^{-3}	0.995	38.7
		10^{-4}	0.995	38.7
		10^{-9}	0.995	38.8
	SAA	10^{-2}	0.903	14.4
		10^{-3}	0.903	27.8
		10^{-4}	-	OOM
		10^{-9}	-	OOM

the probability of an invalid certificate is between 1 in 1 million and 1 in 1 billion and thus extremely unlikely. Previously, the data-driven synthesis methods were only able to achieve a confidence parameter β around 0.005 [127], so a key contribution of this paper is a framework for producing more reliable certificates.

In addition to theoretical advances, this chapter also introduces algorithmic advances to enable tractability. Namely, a bisection-based over-approximation of an uncertain affine pre-image, an a priori sample discarding method for Scenario Approach theory when the random variable enters affinely, and applying spatial indexing from database theory for efficient querying of piecewise affine functions.

The framework is evaluated on various non-linear stochastic systems, demonstrating tractability on a variety of applications. Additionally, a comparison against Sample Average Approximation (SAA)-based SBF synthesis and against data-driven abstractions reveals that the approach is both memory efficient, which is tightly linked to sample efficiency, and generally faster.

Overall, this chapter demonstrates that data-driven synthesis can achieve formal probabilistic guarantees in realistic stochastic settings where exact models are unavailable, bridging a critical gap between theoretical verification methods and practical deployment.

Similar to the previous Chapter 5, we assume that a partitioning is given. However, an intelligent or adaptive partitioning scheme has the potential to improve the scalability including to higher dimensional systems. Another restrictive assumption, specific to this chapter, is i.i.d. sample access for the noise $\mathbf{v}$. In practice, it is more common to observe (a subset of) the state variables, from which one has to derive the noise realization. If the nominal dynamics are known exactly, then it is easy to compute as the difference between subsequent states subject to the dynamics. If not, however, it is necessary to introduce an over-approximation of the realization, under the uncertain dynamics.

II

Interval Markov Decision Processes

7

Background on Finite-state Models and Abstractions

An alternative to SBFs for verifying the safety of stochastic systems is abstraction-based approaches where an *abstract* finite-state model is constructed over the *concrete* system such that there exists a simulation relation between the abstract model and the concrete system. Then properties can be verified over the abstract model using standard tools for probabilistic model checking and, using the simulation relation, lifted to the concrete system. This family of methods is commonly called *abstraction-based*. Conversely, methods based on SBFs are sometimes called abstraction-free.

The structure of this section is as follows: first, we define some classes of finite-state models and describe their relation. Then, we describe how to abstract a concrete system to an IMDP. In the following Chapter 8, we recall some standard results on probabilistic model checking over IMDPs, and then elaborate algorithmic design, which are important for scalability. These methods are implemented in our tool **IntervalMDP.jl**.

We start by defining an MDP.

Definition 7.1 (Markov Decision Process (MDP))**.** *An MDP is a tuple* $M = (S, A, \gamma)$ *where*

- S *is a finite set of states,*
- A *is a finite set of actions assumed to be available at each state, and*
- $\gamma = \{\gamma_{s,a}\}_{s \in S, a \in A}$ *is a set of transition probability distributions* $\gamma_{s,a} \in \mathcal{D}(S)$ *for each source-action pair* $(s, a) \in S \times A$*.*

We use the triplet (s, a, t) to denote a transition from state s to state t under action a. Outside the context of a transition, we use the letter s to denote a state. A path of an MDP is a sequence of states and actions $\omega = (s[0], a[0]), (s[1], a[1]), \ldots$, where $(s[k], a[k]) \in S \times A$. We denote by $\omega[k] = s[k]$ the state of the path at time $k \in \mathbb{N}_0$. A Markov strategy or policy for an MDP is a sequence $\pi = (\pi[0], \pi[1], \ldots)$ such that $\pi[k] : S \to A$ assigns an action to each state of an MDP at time step k. We study time-varying, deterministic Markov policies as they are sufficient for optimality for finite-time safety properties [45], [122], [133]–[135].

Although MDPs are powerful and widespread in computer science, control theory, and engineering in general as a method for modelling, analysing, and controlling systems, their weakness is the assumption of perfect knowledge of transition probabilities. A generalization that addresses this weakness is Robust Markov Decision Processes (RMDPs) where uncertainty is inherent to the model and can, in fact, be interpreted as a family of MDPs [135]–[137].

Definition 7.2 (Robust Markov Decision Process (RMDP))**.** *An (s, a)-rectangular RMDP is a tuple $M = (S, A, \Gamma)$ where*

- *S is a finite set of states,*
- *A is a finite set of actions assumed to be available at each state, and*
- *$\Gamma = \{\Gamma_{s,a}\}_{s \in S, a \in A}$ are sets of feasible transition probability distributions $\Gamma_{s,a} \subseteq \mathcal{D}(S)$ for each source-action pair $(s, a) \in S \times A$.*

$\mathcal{D}(S)$ denotes the set of all (discrete) distributions over S (see Chapter 2). The RMDP is called (s, a)-rectangular because the uncertainty in the transition probability, i.e., the transition ambiguity set, is independent for each source-action pair (s, a) That is, an adversary, also sometimes called nature, can pick the target distribution $\gamma_{s,a} \in \Gamma_{s,a}$ for source-action pair (s, a) without impacting the uncertainty for other source-action pairs. Although RMDPs are more powerful with s-regularity or without rectangularity [135], we assume, and later construct models to enforce, (s, a)-rectangularity, as it reduces the computational burden during verification. Furthermore, (s, a)-rectangularity imposes the assumption that the adversary has access to the control action selected by the controller π, or in other words, in the alternating game between the controller and the adversary, the controller always picks first. Before formally defining an adversary, we must also discuss static and dynamic uncertainty semantics. The distinction is whether the adversary may or may not pick different distributions at each time step k for each source-action pair (s, a), referred to as dynamic and static uncertainty semantics respectively. For the purpose of abstraction-based verification, we are only interested in dynamic uncertainty to robustify the verification against the worst-case distribution in the ambiguity set under changing value functions. Thus, we arrive at the following definition.

Definition 7.3 (RMDP adversary)**.** *An adversary for an RMDP is a sequence of functions $\tau = (\tau[0], \tau[1], \ldots)$ where at each time step k the function $\tau[k] : S \times A \to \mathcal{D}(S)$ satisfy the condition $\tau[k](s, a) \in \Gamma_{s,a}$ for every source-action pair (s, a).*

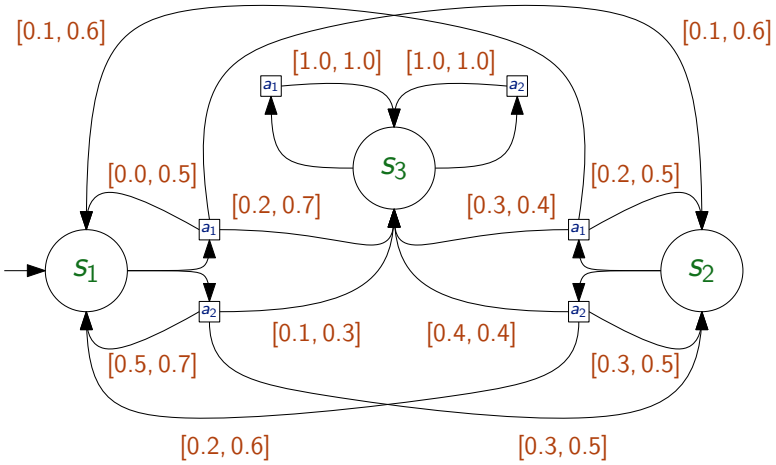


Figure 7.1: An example of an IMDP with three states and two actions where the state s_3 is a sink state, meaning that it transitions to itself with probability one. When drawing IMDPs, it is typical to only denote the interval bounds and leave the constraint that the transition probabilities must sum to one implicit.

Given a policy π and an adversary τ, an RMDP collapses to a (time-varying) finite Markov chain.

To enable algorithms with better computational complexity, it is useful to impose a further structure on the ambiguity sets. We are in particular interested in IMDPs where the ambiguity set is an interval ambiguity set, defined as follows.

Definition 7.4 (Interval Markov Decision Process (IMDP))**.** *An IMDP is a tuple* $M = (S, A, \Gamma)$ *where*

- S *is a finite set of states,*
- A *is a finite set of actions assumed to be available at each state, and*
- $\Gamma = \{\Gamma_{s,a}\}_{s \in S, a \in A}$ *are sets of feasible transition probability distributions with* $\Gamma_{s,a} \in \operatorname{int amb}(S)$.

$\operatorname{int amb}(S)$ denotes the set of all interval ambiguity sets over S (see Chapter 2). The interval ambiguity set $\Gamma_{s,a}$ for each source-action pair (s, a) is defined via upper and lower bounds for each destination t. A pictorial example of the interval ambiguity sets is shown in Figure 7.1. Geometrically, an interval ambiguity set can be interpreted as the intersection between a hyperrectangle in $\mathbb{R}_{\geq 0}^{|S|}$, from the interval bounds, with the probability simplex again in $|S|$ dimensions. That the ambiguity set is defined by interval bounds for each transition (s, a, t) enables the use of the O-maxmimization algorithm [122], [138] as the adversary – to pick worst-case distributions given a value function (see the Section 8.4 for further details on O-maximization). In Chapter 9, we will impose additional structure on the abstract model, primarily to improve space complexity compared to IMDPs.

7.1. Abstractions of Stochastic Systems

As described in the introduction to this chapter, abstraction-based verification approximate a complex, concrete system with a simpler (often finite) abstract model, which is easier to analyse. To ensure that the safety probability of the finite-state abstraction applies directly or can be lifted to the concrete system, there must necessarily exist some relation between these systems [13], [139]. More specifically, the relation must be either an approximate or alternating probabilistic simulation relation to guarantee that every action in the finite-state model can be matched to a control input in the concrete system such that the stochastic transitions of the two systems can be (approximately) coupled. Moreover, it is important that the outputs, be it discrete or continuous, are equivalent or sufficiently close in some sense of distance [139]–[142]. With this kind of relation, it is possible to prove, typically via dynamic programming, that satisfaction probabilities can be lifted from the abstract to the concrete system. We defer a formal treatment of this subject to Section 9.5 where we define a specific type of simulation relation that is useful in abstracting stochastic dynamical systems to any subclass of (s,a)-rectangular RMDPs for the purpose of verification or control synthesis wrt. a safety specification.

For the remainder of this section, we describe a standard procedure for abstracting stochastic systems to IMDPs, as it serves as the foundation and justification for the developments in Chapter 9. The construction of such an abstraction depends on the structure of the concrete system and the availability of information. For the purpose of describing a typical procedure, in this section we assume that the system is additive (2.34) with zero-mean and diagonal-covariance Gaussian noise. That is,

$$\mathbf{x}[k+1] = f(\mathbf{x}[k]) + \mathbf{v}[k], \quad p_{\mathbf{v}} = \mathcal{N}(0,\Sigma) \tag{7.1}$$

where $\Sigma \in \mathbb{S}_+^n$ is a diagonal matrix.

Remark 7.1. *If Σ is not diagonal, then it can be diagonalized with a (linear) eigentransformation of the entire system. Formally, let $\Sigma = E\Lambda E^{-1}$ be the eigendecomposition of Σ where Λ is a matrix with the eigenvalues of Σ on the diagonal and E has the corresponding eigenvectors as columns. Then (7.1) can be transformed as follows:*

$$E\mathbf{x}[k+1] = Ef(\mathbf{x}[k]) + E\mathbf{v}[k], \qquad p_{\mathbf{v}} = \mathcal{N}(0,\Sigma), \text{ or} \tag{7.2}$$

$$\tilde{\mathbf{x}}[k+1] = \tilde{f}(\tilde{\mathbf{x}}[k]) + \tilde{\mathbf{v}}[k], \qquad p_{\tilde{\mathbf{v}}} = \mathcal{N}(0,\Lambda) \tag{7.3}$$

where $\tilde{f}(\tilde{x}) = Ef(E^{-1}\tilde{x})$. Therefore, it is wlog. to consider diagonal covariance for additive Gaussian noise. Note also that some works rely on the Mahalanobis transformation of Σ [16], [40], but unlike the Mahalanobis transformation, the eigentransformation works in the presence of a degenerate distribution, i.e. the standard deviation is zero for some dimensions.

The standard approach to abstract systems of the form (7.1) to an IMDP is to partition the region of interest X into hyperrectangular regions $\{q_1, \dots, q_\ell\}$ and associate each region with a state s in S. Then, for a state $s \in S$, Γ_s (ignoring

actions as (7.1) is autonomous) is defined as

$$\Gamma_s = \{\gamma_s \in \mathcal{D}(S) : \min_{x \in q_s} T(q_t \mid x) \leq \gamma_s(t) \leq \max_{x \in q_s} T(q_t \mid x),\ \forall t \in S\}. \tag{7.4}$$

To solve $\max_{x \in q_s} T(q_t \mid x) = \min_{x \in q_s} \mathcal{N}(q_t \mid f(x), \Sigma)$, one can exploit the fact that for any *hyperrectangle* q_t, the function $\mathcal{N}(q_t \mid \cdot, \Sigma)$ is log-concave [16]. Thus, if f is affine, we can use projected gradient descent, a Frank-Wolfe algorithm, KKT conditions, or interior point methods to find the optimal point x. For the minimization, we can rely on vertex enumeration [16], [40]. For efficiency, we can first compute the set $Y = f(q_s)$ such that $\min_{x \in q_s} \mathcal{N}(q_t \mid f(x), \Sigma) = \min_{y \in Y} \mathcal{N}(q_t \mid y, \Sigma)$ [16]. If f is non-linear, we can rely on reachability analysis tools (see Subsection 2.2.2) to compute an over-approximation of Y for a sound abstraction. Furthermore, if we compute a hyperrectangular over-approximation $\hat{Y} = [\underline{y}_1, \overline{y}_1] \times \cdots \times [\underline{y}_n, \overline{y}_n]$ of Y, then, at the expense of conservatism in the abstraction, an analytical formula can be applied [40]:

$$\max_{y \in \hat{Y}} \mathcal{N}(q_t \mid y, \Sigma) = \prod_{i=1}^{n} \max_{y_i \in [\underline{y}_i, \overline{y}_i]} \mathcal{N}(q_t^i \mid y_i, \Sigma_{ii}), \tag{7.5}$$

where q_t^i is the interval of q_t along dimension i. Then, a similar approach can be applied to minimization. The algorithm is described for a (control) system with additive noise in Algorithm 7.1.

Remark 7.2. *The computation of $\underline{\gamma}_s(t)$ and $\overline{\gamma}_s(t)$ when* **v** *is not normally distributed has been addressed in [63] via noise partitioning and [41] via data-driven distributionally robust methods. Both methods rely on the computation of the nominally reachable set Y.*

An unaddressed nuance is the need to include an extra sink state, representing the transition to X^c. The interval bounds for transitioning to this state can be computed as above, with the modification that it is computed as the dual probability of transitioning to X.

With an abstraction constructed, the next step is to verify the abstract model before lifting the guarantees to the concrete system. In the next Chapter 8, we will describe the safety verification objective for IMDPs and hardware-aware algorithmic innovations to reduce the computation time of the verification algorithm.

Algorithm 7.1 IMDP Abstraction

Require: Nominal dynamics f, region of interest X, control set U
Ensure: IMDP M that simulates f, initial states S_0, unsafe states S_u

Partition X into regions $Q := \{q_1, \ldots, q_\ell\}$
$S := \{s_1, \ldots, s_\ell\}$
Select a finite set $\{u_1, \ldots, u_{|A|}\}$ of U
$A := \{a_1, \ldots, a_{|A|}\}$
for each $(s, a) \in S \times A$ **do**
    Compute $Y := f(q_s, u_a)$
    for each destination $t \in S$ **do**
        Let $\underline{\gamma}_{s,a}(t) := \min_{x \in q_s} T(q_t \mid x) = \min_{y \in Y} \mathcal{N}(q_t \mid y, \Sigma)$
        Let $\overline{\gamma}_{s,a}(t) := \max_{x \in q_s} T(q_t \mid x) = \max_{y \in Y} \mathcal{N}(q_t \mid y, \Sigma)$
$S_0 := \{s \in S : q_s \cap X_0 \neq \emptyset\}$
$S_u := \{s \in S : q_s \cap X_u \neq \emptyset\}$
return $M = (S, A, \Gamma)$, S_0, S_u

8

Accelerated Value Iteration for Interval Markov Decision Processes

Sequential programming is really hard,
and parallel programming is a step beyond that.

Andrew S. Tanenbaum

This chapter is a copy of *F. B. Mathiesen, M. Lahijanian, and L. Laurenti, "IntervalMDP.jl: Accelerated value iteration for interval markov decision processes",* IFAC-PapersOnLine*, vol. 58, no. 11, pp. 1–6, 2024, 8th IFAC Conference on Analysis and Design of Hybrid Systems.* **DOI**: 10.1016/j.ifacol.2024.07.416 [80] with minor corrections to streamline the presentation.

8.1. Introduction

Interval Markov Decision Processes (IMDPs) are a popular target model for abstraction-based verification and control of stochastic hybrid systems [16], [143]. Consequently, it is important to both study algorithms and develop tools for this class of models. Attention has been given to the convergence of value iteration [144], extensions to the IMDP model [58], [62], [134], [145], new properties [146], techniques for abstraction to IMDPs [16], [40], [61], [64], [66], [147]–[150], and convergence of abstractions to IMDPs [45]. Prior to this work, two tools supported probabilistic model checking for IMDPs: PRISM [19] and bmdp-tool [151][1]. However, these tools lack parallelization via multi-threading and GPU-acceleration, under-utilizing available hardware and leaving a gap that this chapter aims to fill.

To this end, we introduce **IntervalMDP.jl**, which is a Julia package for model checking of IMDPs. The tool supports safety, reachability, reach-avoid, discounted reward, and expected hitting time properties, and allows the user to query for both optimal strategies and quantitative values of satisfaction. The package contains both a CPU and a GPU implementation, allowing one to use CUDA-capable hardware [153] to accelerate the computation. The package is developed in Julia, which is a modern programming language that targets the scientific community [154]. It enables both fast prototyping and accelerated code to be written in the same language, including the ability to write CUDA kernels for custom accelerated computations. Furthermore, through Julia's parametric typing, **IntervalMDP.jl** supports single- and double-precision floating point numbers, as well as rational numbers for exact arithmetic with transition probabilities.

We evaluate **IntervalMDP.jl** on various benchmarks and compare it with PRISM [19] and bdmp-tool [151]. The benchmarks include 35 IMDPs taken from the literature, with the total number of transitions between states ranging from a few tens for the smaller models to tens of millions for the larger models. The empirical analysis shows that **IntervalMDP.jl** CPU implementation is on average 2-4× faster compared to the state of the art, while the GPU implementation can achieve speed-ups of various orders of magnitude on the larger systems. Furthermore, because of the use of sparse matrices and the Julia type system, in all cases, **IntervalMDP.jl** requires less memory compared to PRISM and bmdp-tool.

The chapter is organized as follows. First, in Section 8.2, we formally introduce robust value iteration for IMDPs. Then, in Section 8.3, we give an overview of **IntervalMDP.jl** and describe how create an IMDP model in the tool and how to perform strategy synthesis and verification. In Section 8.4, we detail our algorithmic approach to robust value iteration on GPUs. Finally, in Section 8.5, we illustrate the effectiveness of **IntervalMDP.jl** on various benchmarks.

[1] Concurrent with this work, the probabilistic model checker **Storm** [20] added support for IMDPs and **IMPaCT** [152], a tool for abstraction of stochastic systems to IMDPs and model checking thereof, was released.

8.2. THE GOAL OF IntervalMDP.jl: ROBUST VALUE ITERATION FOR IMDPS

Recall that an IMDP is a tuple $M = (S, A, \Gamma)$ where $\Gamma = \{\Gamma_{s,a}\}_{s \in S, a \in A}$ is the set of transition ambiguity sets where $\Gamma_{s,a} \in \text{int amb}(S)$ for each $s \in S$ and $a \in A$. Then, the goal of **IntervalMDP.jl** is to compute quantitative properties over and synthesize controllers for a given IMDP including probabilistic safety, reachability, reach-avoid, discounted reward, and expected hitting time. For ease of exposition and with the fact that the goal of this dissertation is safety verification, we restrict ourselves to those properties in this chapter. For safety on IMDPs, we define a set of unsafe states $S_u \subseteq S$ and a time horizon $K \in \mathbb{N}_0$. Then, the objective is to solve the following optimization problems:

$$P_{\text{safe}}(s_0) = \max_{\pi} \min_{\tau} \mathbb{P}^{s_0}_{\pi,\tau} \left[\omega \in \Omega \mid \forall k \in [0, K],\, \omega[k] \notin S_u\right], \tag{8.1}$$

where $\mathbb{P}^{s_0}_{\pi,\tau}$ is the probability of the Markov chain induced by strategy π and adversary τ, starting from $s_0 \in S$. Equation (8.1) can be computed by solving the following value iteration:

$$\begin{aligned} V_0(s) &= \mathbf{1}_{S_u}(s) \\ V_k(s) &= \mathbf{1}_{S_u}(s) + \mathbf{1}_{S \setminus S_u}(s) \min_{a \in A} \max_{\gamma_{s,a} \in \Gamma_{s,a}} \sum_{t \in S} V_{k-1}(t) \gamma_{s,a}(t), \end{aligned} \tag{8.2}$$

such that $P_{\text{safe}}(s_0) = 1 - V_K(s_0)$. With trivial modifications, (8.2) can be used to verify the aforementioned properties, and via a product construction with a task automaton [13], [16], temporal properties can also be verified.

Note that the inner maximization in (8.2) is an LP problem. However, it exhibits a certain structure that enables an algorithm, O-maximization [122], [138], that is faster than applying an LP solver. This algorithm is the source of study in this chapter, and in Section 8.4, we show how it can be efficiently parallelized to take advantage of GPU hardware architectures.

8.3. OVERVIEW OF IntervalMDP.jl

IntervalMDP.jl is a Julia package that introduces parallelization and GPU-powered processing to perform value iteration in (8.2), offering efficiency for verification and strategy synthesis for IMDPs. **IntervalMDP.jl** has the following main features:

- modelling of IMDPs and fIMDPs (see the following Chapter 9),
- lazy product constructions with the system model and task automaton,
- value iteration and strategy synthesis for a wide variety of properties (finite and infinite horizon safety, reachability, and reach-avoid, as well as discounted reward and expected hitting time) and for all combinations of satisfaction and strategy modes,

- configurable Bellman operator implementations including O-maximization, vertex enumeration, and using LP solvers,
- dense and sparse matrix support, including mixed modes for fIMDPs,
- customizable numerical precision including exact (rational) arithmetic on CPUs,
- multi-threaded CPU and CUDA-accelerated value iteration, and
- data loading and writing in various formats (e.g., PRISM, bmdp-tool, and **IntervalMDP.jl**).

In this section, we show how to create an IMDP model and perform verification in **IntervalMDP.jl**. The source code can be found at `https://github.com/Zinoex/IntervalMDP.jl`. To install and import the package, run the following commands in Julia:

```
using Pkg
Pkg.add("IntervalMDP")

using IntervalMDP, IntervalMDP.Data
```

The module `IntervalMDP` contains structures and functions for modeling, value iteration, and strategy synthesis. The submodule `IntervalMDP.Data` contains functions relevant to reading and writing IMDPs in various data formats.

Remark 8.1. **IntervalMDP.jl** *is developed as a software package, and not a standalone software tool, with the intention of entering into an ecosystem of packages that co-evolve and enables scalable and flexible verification and control of stochastic systems. Previous tools [19], [151] were developed as standalone tools, which complicates interfacing (requires custom model file generation, solution parsing) when, e.g., building abstraction-based verification tools.*

8.3.1. System modeling

We programmatically construct an `IntervalMarkovDecisionProcess` object and pass it to **IntervalMDP.jl**. Here, we include an example of how to construct a 3-state IMDP with the third state being a sink state.

```
prob1 = IntervalAmbiguitySets(;
    lower = [0.0 0.5; 0.1 0.3; 0.2 0.1],
    upper = [0.5 0.7; 0.6 0.5; 0.7 0.3],
)
prob2 = IntervalAmbiguitySets(;
    lower = [0.1 0.2; 0.2 0.3; 0.3 0.4],
    upper = [0.6 0.6; 0.5 0.5; 0.4 0.4],
)
prob3 = IntervalAmbiguitySets(;
    lower = [0.0 0.0; 0.0 0.0; 1.0 1.0],
    upper = [0.0 0.0; 0.0 0.0; 1.0 1.0]
```

```
)
transition_probs = [prob1, prob2, prob3]
imdp = IntervalMarkovDecisionProcess(transition_probs)
```

Other constructors exist for `IntervalMarkovDecisionProcess`; see package documentation for details.

Figure 7.1 shows a pictorial representation of the IMDP constructed with the code above. The interval transition probabilities are specified as a list of interval ambiguity sets where each element corresponds to one state and the upper and lower bounds are specified using dense matrices with columns representing the source-action pairs and rows the destination. This is contrary to the typical transition matrix format with the source in rows and destination in columns; however, it is necessary for cache efficiency, as Julia is a column-major language. Note that **IntervalMDP.jl** also supports sparse matrices encoded in the Compressed Sparse Column (CSC) format [155] for better memory utilization and computational efficiency, which is used in the empirical evaluation in Section 8.5.

8.3.2. SPECIFICATION AND VERIFICATION

The tool minimizes or maximizes the optimistic or pessimistic probability of reaching a given set of states, or optimizes a reward. Continuing with the example from Section 8.3.1, we show how to compute the maximum pessimistic probability of avoiding state 3 within 100 time steps.

```
prop = FiniteTimeSafety([3], 100)
spec = Specification(prop, Pessimistic, Maximize)
problem = VerificationProblem(imdp, spec)

# Compute the value function
V, k, residual = solve(problem)
```

For a `VerificationProblem`, the function `solve` returns a 3-tuple containing the following: the robust optimal probability (in the above example, maximum pessimistic probability) of satisfying the safety property, the number of iterations performed (100 in the above example), and finally, the Bellman residual for the last iteration. That is, the state-wise difference between the value function for the last and the second to last iteration. For a finite horizon property, the number of iterations is fixed, while for an infinite horizon property, a user may specify a tolerance for convergence, measured by the maximum Bellman residual.

Using the library, we may also synthesize the optimal policy.

```
# Compute optimal finite-time policy
prop = FiniteTimeSafety([3], 100)
spec = Specification(prop, Pessimistic, Maximize)
problem = ControlSynthesisProblem(imdp, spec)
time_dependent_policy, V, k, residual = solve(problem)
```

With a finite time horizon property, the optimal strategy is deterministic and time dependent, and thus, **IntervalMDP.jl** returns a list of length K with actions for each time step. The action list for each time step is of size $|S|$.

A core feature of **IntervalMDP.jl** is GPU-accelerated algorithms. To enable the use of CUDA, we only need to transfer the problem to the GPU. Note that GPUs generally have less memory than what is available to CPUs, and as a consequence, it is highly recommended the use of the sparse format for IMDPs on GPUs.

For the example above, the following code enables the use of a GPU, if available:

```
using CUDA

prop = FiniteTimeSafety([3], 100)
spec = Specification(prop, Pessimistic, Maximize)
problem = VerificationProblem(imdp, spec)

if CUDA.functional()
    problem = IntervalMDP.cu(problem)
end

V, k, residual = solve(problem)
```

This trivial modification in usage allows improvements in terms of computational time of 50-200× as is illustrated in Section 8.5.

8.4. Value Iteration on CUDA-capable GPUs

In this section, we discuss our algorithmic approach to solve (8.1) on CUDA-capable hardware. For completeness, we include a short description of the CUDA programming model in Appendix 8.A. Recall from (8.1) that computing the optimal pessimistic value reduces to iteratively solving the following problem for each $s \in S$ and for each $k \in [1, K]$

$$\max_{a \in A} \min_{\gamma_{s,a} \in \Gamma_{s,a}} \sum_{t \in S} V_{k-1}(t)\gamma_{s,a}(t). \tag{8.3}$$

As (8.3) is solved independently for each state s, its solution for different states can be *trivially* parallelized. In what follows, we show how robust value iteration can be parallelized further within each state, achieving significant acceleration.

The inner problem in (8.3) is a linear problem that must be solved for every $a \in A$. Consequently, it is possible to solve it using LP solvers, but the problem also exhibits structure that enables a significantly more efficient algorithm: O-maximization [122], [138]. The general idea is to sort the states based on the value function V_{k-1} and then assign probability mass to states with a high value until the budget $(1 - \sum_t \gamma_{s,a}(t))$ is spent. Both phases can benefit from parallelization on a GPU. For ordering the states based on the value of the value function at the current time step V_{k-1}, we apply existing parallel sorting algorithms that are particularly suited to run efficiently on a GPU. In particular, in **IntervalMDP.jl**, we use

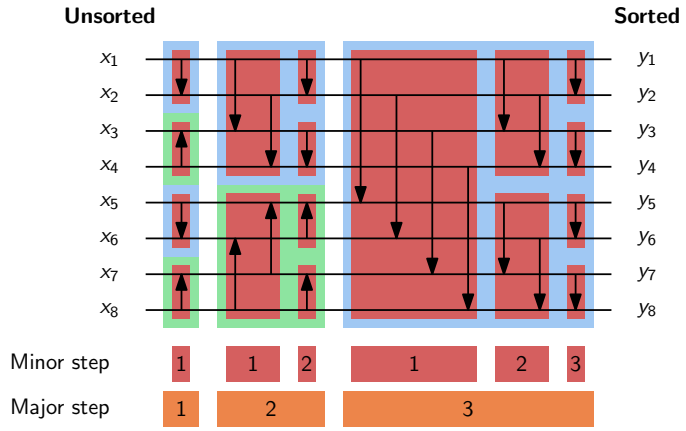


Figure 8.1: Bitonic sorting network structure in $\log_2(n)$ major rounds, each of which is up to $\log_2(n)$ minor rounds. Each arrow represents a comparison and points towards the larger element. After each major round, subsets of increasing size are bitonic (increasing order, then decreasing of the same size).

bitonic sort [156], which is a parallel sorting algorithm with $O(\log_2(n)^2)$ latency[2]. Figure 8.1 shows an example of how the sorting is performed when n is a power of 2. A bitonic sorting treats the list as bitonic subsets. A bitonic set is a sequence of first increasing then decreasing values of equal size. These subsets are then merged over $\log_2(n)$ major rounds, each of which consists of up to $\log_2(n)$ minor rounds to preserve the bitonic property. After $\log_2(n)$ major rounds, the set is the first half of a bitonic set, which implies that it is sorted in increasing.

Remark 8.2. *To implement bitonic sorting* efficiently *on GPUs, it is important to avoid division and modulo in index calculations, as they are expensive operations on GPUs. Instead, exploiting that bitonic sorting operates on powers of two, the indices are computed using bitwise arithmetic, as bitwise operations are cheap. To facilitate sets that are not powers of two, one may virtually enlarge the set to the next power of two, and if any index in a pair exceeds the bounds, skip any potential swapping.*

The second phase consists of assigning probabilities within the interval ambiguity set to give the most probability mass possible to states early in the ordering. To perform this assignment step in parallel, we have to parallelize the sequential algorithm below.

```
function omaximization(ordering, lower, upper)
    p = copy(lower)
    rem = 1 - sum(lower)
    gap = upper - lower
```

[2]Note that when assessing parallel algorithms, the asymptotic performance is measured by the latency, which is the delay in the number of parallel operations before the result is available. This is in contrast to traditional algorithms, which are assessed by the total number of operations.

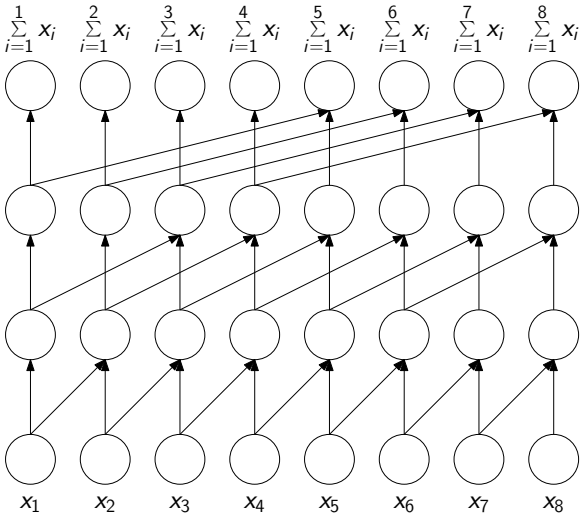


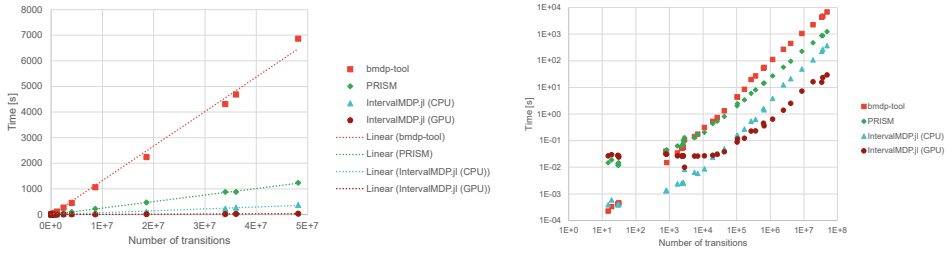
Figure 8.2: Cumulative sum as a tree-reduction. This tree reduction has a latency of $O(\log_2(n))$ as opposed to $O(n)$ of a traditional iterative algorithm.

```
    for o in ordering
        if gap[o] < rem
            p[o] += gap[o]
            rem -= gap[o]
        else
            p[o] += rem
            break;
        end
    end

    return p
end
```

While seemingly sequential in nature, the method implicitly implements a cumulative summation procedure according to the ordering previously computed. Consequently, we can employ a parallel algorithm for cumulative summation based on a binary tree reduction developed in [157] (see Figure 8.2), which has latency $O(\log_2(n))$. In particular, for O-maximization, the cumulative sum is computed over the gap between the upper and lower bounds according to the ordering. Thus, each element will have the sum of the gaps of the states up to and including itself in the ordering. Then, each iteration in O-maximization is independent and thus can be performed in parallel, that is, with a latency of $O(1)$.

```
function omaximization(ordering, lower, upper)
    p = copy(lower)
    rem = 1 - sum(lower)
    gap = upper - lower
```



(a) Linear scale with trendline

(b) Log-log scale

Figure 8.3: Computation time of **IntervalMDP.jl** compared against bmdp-tool and PRISM on 35 IMDPs of varying sizes.

```
    # Ordered cumulative sum of gaps
    cumgap = cumulative_sum(gap[ordering])

    for (i, o) in enumerate(ordering)
        rem_state = max(rem - cumgap[i] + gap[o], 0)
        if gap[o] < rem_state
            p[o] += gap[o]
        else
            p[o] += rem_state
        end
    end

    return p
end
```

Again, implementing this algorithm for execution on CUDA-capable hardware requires consideration beyond algorithmic design. Namely, memory and register allocation, local register shuffles, memory caching, and scheduling.

8.5. EXPERIMENTS

In order to show the effectiveness of **IntervalMDP.jl**, we compare it against bmdp-tool [151] and PRISM [19]. We benchmark the tools on 35 models, whose details can be found in Table 8.1 in Appendix 8.B. The models are taken from the literature and include abstractions of linear and non-linear systems, including neural networks, with number of transitions ranging from few tens to tens of millions.

For a fair comparison, for all models we run a (maximum pessimistic) finite time safety query (200 time steps). All experiments were conducted on a computer with 16GB RAM, an Intel I7-6700K CPU (4 cores), and an NVIDIA GTX1060 6GB VRAM GPU. For each tool, we measure the computation time only, and not the time it takes to load each file.

The results obtained are shown in Figure 8.3 as plots of the computation time

as a function of the number of transitions in the IMDP. In Figure 8.3a, we see that **IntervalMDP.jl** substantially outperforms the other tools in terms of computation time. For instance, computing the query for the largest model takes bmdp-tool 6865s and PRISM 1235s, while **IntervalMDP.jl** takes 372s and 30s for the CPU and GPU implementation respectively. This is a speed-up of 228× and 41× of the GPU implementation of **IntervalMDP.jl** relative to bmdp-tool and PRISM respectively.

The log-log plot in Figure 8.3b highlights the performance differences on the smaller models. Specifically, the CPU implementation of **IntervalMDP.jl** and the bmdp-tool are faster than PRISM on the smallest models, with a speed-up of 30-60×. We conjecture that the better scaling of PRISM compared to bmdp-tool is due to using a dictionary of non-zero probability destinations for each source-action pair stored in a list, rather than a dictionary keyed by source/action/destination triplets. **IntervalMDP.jl**, on the other hand, does not store any dictionary. Instead, we track the indices for source, action, and destination in the CSC-format sequentially, allowing better caching when accessing probabilities. In Figure 8.3b, we also see that the GPU implementation of **IntervalMDP.jl** has overhead that is dominant for smaller models, while for larger models it is consistently orders of magnitude faster than the other implementations. The cut-off is at ≈25000 transitions in the IMDP.

A common bottleneck for IMDP tools is memory consumption, which is only exacerbated on a GPU, as they generally have less memory available. However, due to the CSC-format with `Float64` values and `Int32` indices, **IntervalMDP.jl** generally requires less memory compared to PRISM and bmdp-tool. For example, to run value iteration on the largest model (i.e., `pimdp_2` in Table 8.1 in Appendix 8.B), **IntervalMDP.jl** requires 4.88GB of memory. In contrast, PRISM uses 6.32GB of memory and bmdp-tool uses 5.38GB to run value iteration on the same problem. This is a 23% reduction relative to PRISM and 9% relative to bmdp-tool, which is including the Julia runtime.

8.6. Conclusion

In this chapter, we presented an accelerated value iteration algorithm for IMDPs. Using an efficient algorithm, O-maximization, as foundation, we introduce algorithmic enhancements that enable better use of CUDA-enabled GPUs. In particular, we decompose O-maximization into two phases, sorting and probability assignment, that are parallelized individually. The sorting phase is parallelized using bitonic sorting, which is well-suited for GPUs for its synchronous comparator network. The probability assignment phase is parallelized by observing that the phase implicitly computes a cumulative sum, which can be efficiently computed in parallel using a parallel prefix sum algorithm. These algorithms are integrated into a new tool for IMDPs, **IntervalMDP.jl**, which we evaluate on a set of benchmark models. Our evaluation shows that **IntervalMDP.jl** achieves significant speedups over existing tools, up to a factor of 41×.

While previous works have relied on the trivially parallel nature of value iteration to parallelize the algorithm, i.e. independently updating each state, we demonstrate that further algorithmic improvements can be made by carefully analyzing the individual components of the algorithm. Our work opens up new avenues for future

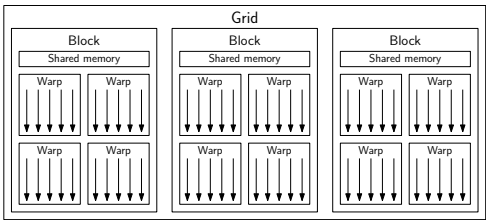


Figure 8.4: Each arrow represents a thread, a warp is executed (mostly) synchronously, a block shares memory and allow synchronization, and the grid encapsulates the entire execution.

research in accelerating algorithms for IMDPs and other related models using GPUs. It is important to stress that we believe further improvements are possible by optimizing the implementation further to better utilize the GPU architecture, e.g., by optimizing memory access patterns and reducing register pressure.

Despite performance improvements in model checking for IMDPs, challenges remain in scaling to very large models due to the limited memory capacity of GPUs. The following Chapter 9 addressed this challenge by introducing a structured model representation that reduces memory consumption significantly.

8.A. CUDA Programming Model

The CUDA framework is a Single Instruction Multiple Thread (SIMT) parallel execution platform and API [153]. This is in contrast to Single Instruction Multiple Data (SIMD) where all data must be processed homogeneously without control flow. SIMT makes CUDA more flexible for heterogeneous processing and control flow. The smallest execution unit in CUDA is a thread, which is a sequential processing of instructions. A thread is uniquely identified by its thread index, which allows indexing into the global data for parallel processing. A group of 32 threads is called a warp, which will be executed *mostly* synchronously on a streaming multiprocessor. If control flow makes threads in a wrap diverge, instructions may need to be decoded twice and executed in two separate cycles. Due to this synchronous behaviour, data can be shared in registers between threads in a warp for maximum performance. A collection of (up to) 1024 threads is called a block, and this is the largest aggregation that can be synchronized. Furthermore, threads in a block share the appropriately named shared memory. This is memory that is stored locally on the streaming multiprocessor for fast access. Note that shared memory is unintuitively faster than local memory (not to be confused with registers) due to local memory being allocated in off-chip memory. Finally, a collection of (up to) 65536 blocks is called the grid of a kernel, which is the set of instructions to be executed. The grid is singular as only a single ever exists per launched kernel. Hence, if more blocks are necessary to process the amount of data, then a grid-strided loop or multiple kernels are necessary. Figure 8.4 shows a pictorial representation of the CUDA Programming Model.

8.B. Benchmark details

Table 8.1: List of the 35 models that was used in the empirical study (see Section 8.5) and their properties.

Model	Number of states	Number of actions	Number of transitions
pimdp_0	21174	3	33908939
pimdp_1	32336	3	36019324
pimdp_2	42634	3	48106480
multiObj_robotIMDP	207	4	2784
linear_5_states_0.9_f_0.01_sigma	6	1	15
linear_5_states_0.9_f_0.05_sigma	6	1	29
linear_5_states_0.9_f_0.1_sigma	6	1	31
linear_5_states_1.05_f_0.01_sigma	6	1	19
linear_5_states_1.05_f_0.05_sigma	6	1	29
linear_5_states_1.05_f_0.1_sigma	6	1	31
linear_50_states_0.9_f_0.01_sigma	51	1	819
linear_50_states_0.9_f_0.05_sigma	51	1	2379
linear_50_states_0.9_f_0.1_sigma	51	1	2551
linear_50_states_1.05_f_0.01_sigma	51	1	773
linear_50_states_1.05_f_0.05_sigma	51	1	2299
linear_50_states_1.05_f_0.1_sigma	51	1	2551
pendulum_1_layer_120_states_0.01_sigma	121	1	1709
pendulum_1_layer_120_states_0.05_sigma	121	1	6877
pendulum_1_layer_120_states_0.1_sigma	121	1	10839
pendulum_1_layer_240_states_0.01_sigma	241	1	5489
pendulum_1_layer_240_states_0.05_sigma	241	1	26321
pendulum_1_layer_240_states_0.1_sigma	241	1	42480
pendulum_1_layer_480_states_0.01_sigma	481	1	19323
pendulum_1_layer_480_states_0.05_sigma	481	1	102213
pendulum_1_layer_480_states_0.1_sigma	481	1	167398
cartpole_1_layer_960_states_0.01_sigma	961	1	105047
cartpole_1_layer_960_states_0.05_sigma	961	1	361621
cartpole_1_layer_960_states_0.1_sigma	961	1	647640
cartpole_1_layer_1920_states_0.01_sigma	1921	1	266855
cartpole_1_layer_1920_states_0.05_sigma	1921	1	1159587
cartpole_1_layer_1920_states_0.1_sigma	1921	1	2432256
cartpole_1_layer_3840_states_0.01_sigma	3841	1	620296
cartpole_1_layer_3840_states_0.05_sigma	3841	1	4014802
cartpole_1_layer_3840_states_0.1_sigma	3841	1	8614913
harrier_25920_states_[0.05,0.05,0.02,0.01,0.01,0.01]_sigma	25921	1	18574307

9

Abstraction to Factored Interval Markov Decision Processes

Out of nothing I have created a strange new universe.

Janos Bolyai

This chapter is a copy of *F. B. Mathiesen, S. Haesaert, and L. Laurenti, "Scalable control synthesis for stochastic systems via structural IMDP abstractions", in* Proceedings of the 28th International Conference on Hybrid Systems: Computation and Control*, Association for Computing Machinery, 2025.* **DOI**: 10.1145/3716863.3718031 [81] with minor corrections to streamline the presentation.

9.1. Introduction

IMDPs provide an alternative framework to SBFs by enabling the explicit evolution of the stochastic system over time. Through formal abstraction techniques [16], [61], [63], [138], [149], [158] and probabilistic model checking [122], [138], IMDPs allow us to rigorously compute bounds on the probability of reaching unsafe states over finite, and potentially infinite, time horizons. Moreover, the IMDP framework offers the flexibility to verify a broader class of temporal logic properties beyond simple safety, making it an attractive option for more complex specification scenarios [13]. However, a fundamental challenge limits the applicability of IMDPs-based abstraction to high-dimensional systems: memory scalability. In particular, assuming grid partitioning, the number of states in an IMDP grows exponentially with the system dimension n and the number of transitions grows quadratically with the number of states, making it prohibitively large for even moderately complex systems [16], [40], [80], [132], [152]. Despite advances such as sparse matrix representations like the CSC format [80], the memory bottleneck remains a major obstacle to the wider adoption of this approach. Compositional approaches ameliorate this issue by decomposing the problem into subsystems [48]–[56], yet are restricted to subsystems of small dimensions. This leads to the research question: how can we improve the abstraction model for a stochastic system of the form (2.35) to reduce memory requirements, without minimal loss of accuracy?

Problem 9.1

Given a system $\mathbf{x}[k+1] = f(\mathbf{x}[k], u[k], \mathbf{v}[k])$ (i.e., a control system of the form (2.35)), a region of interest X, a safe set X_s, an initial set X_0, and a time horizon K, synthesize a policy π to maximize a lower bound ρ on P_{safe}.

We address the memory scalability issue by drawing inspiration from factored Markov Decision Processes (fMDPs) [159] and Dynamic Bayesian Networks (DBNs) [47], classes of models that exploit the problem structure to represent large state spaces compactly. By introducing a factored representation that represents transition probabilities through state variables to IMDPs, we aim to significantly reduce memory requirements while preserving formal safety guarantees. This chapter presents the theory of fIMDPs including efficient probabilistic model checking and theoretical analyses of complexity and expressivity, the methodology for constructing factored IMDPs, and experimental results demonstrating their potential for scalable stochastic safety verification. Interestingly, the expressivity analysis reveals representation inefficiencies in IMDP-based abstractions and how structured abstraction can ameliorate these deficiencies.

9.2. Factored Interval MDPs

Merging ideas from IMDPs [122], compositional analysis of MDPs [159], [160], DBNs [47], and Dynamic Credal Networks (DCNs) [161], we propose a subclass of RMDPs, which we call *factored Interval Markov Decision Processes (fIMDPs)*. The

general idea is to decompose the states into *state variables* and similarly decompose actions into *action variables*. Then, the ambiguity set is decomposed as a product of an ambiguity set for transitioning along each marginal (i.e. for each state variable), conditional on the (joint) source state and action. Moreover, the conditioned variables can be restricted to a subset of the state and action variables for additional memory efficiency. In what follows, in this section, we first formally introduce fIMDPs. Then, we show how the memory requirements for this class of models are generally orders of magnitude lower compared to those of IMDPs. Furthermore, we also prove that they can produce tighter ambiguity sets compared to IMDPs. These properties will be employed in Section 9.3 to efficiently abstract System (2.35) into an fIMDPs.

Definition 9.1 (factored Interval Markov Decision Process (fIMDP))**.** *Let $S_1, \dots, S_n$ be finite sets of values that each state variable $S_i \in \{s_1^i, \dots, s_{n_i}^i\}$ may take on. Similarly, define the finite sets of values $A_1, \dots, A_m$ for each action variable $A_j \in \{a_1^j, \dots, a_{n_j}^j\}$. Furthermore, assume a given bipartite graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with*

$$\mathcal{V} = \underbrace{\{s^1, \dots, s^n\} \cup \{a^1, \dots, a^m\}}_{\text{Condition variables}} \cup \underbrace{\{S_1, \dots, S_n\}}_{\text{Target variables}}. \tag{9.1}$$

Let $\text{Pa}_{\mathcal{G}}(S_i)$ denote the parent variables of S_i in the graph $\mathcal{G}$. In other words, $\mathcal{G}$ is a dependency graph and $\text{Pa}_{\mathcal{G}}(S_i)$ denotes the conditional variables for the state variable S_i. Then, an fIMDP is a tuple $M = (S, A, \mathcal{G}, \Gamma)$ where

- *$S = S_1 \times \cdots \times S_n$ is a finite set of joint states,*
- *$A = A_1, \dots, A_m$ is a finite set of joint actions, and*
- *$\Gamma = \{\Gamma_{s,a}\}_{s \in S, a \in A}$ are sets of feasible transition probability distributions where $\Gamma_{s,a} = \bigotimes_{i=1}^{n} \Gamma^i_{(s,a) \cap \text{Pa}_{\mathcal{G}}(S_i)}$ with $\Gamma^i_{(s,a) \cap \text{Pa}_{\mathcal{G}}(S_i)} \in \text{int amb}(S_i)$.*

For notational convenience, with slight abuse of notation, we write $\Gamma^i_{s,a}$ to mean $\Gamma^i_{(s,a) \cap \text{Pa}_{\mathcal{G}}(S_i)}$.

Remark 9.1. *If $\mathcal{G}$ is a* complete *bipartite graph, that is, each target variable depends on all variables of s, a, then we obtain an orthogonally decoupled Interval Markov Decision Process (odIMDP); a type of model introduced in [81]. OdIMDPs represent a worst-case, from a memory requirements perspective, of fIMDPs. When analyzing the space complexity later in Subsection 9.2.1, we state the results for odIMDPs to keep the results as general as possible, and yet show that the complexity is smaller than IMDPs.*

Similarly to IMDPs [138], A path of an MDP is a sequence of states and actions $\omega = (s[0], a[0]), (s[1], a[1]), \dots$, where $(s[k], a[k]) \in S \times A$. We denote by $\omega[k] = s[k]$ the state of the path at time $k \in \mathbb{N}_0$. A Markov strategy or policy for an MDP is a sequence $\pi = (\pi[0], \pi[1], \dots)$ such that $\pi[k] : S \to A$ assigns an action to each state of an MDP at time step k. An adversary is a function that assigns a feasible distribution $\gamma^i_{s,a} \in \Gamma^i_{s,a}$ to each marginal i, given a source-action pair $(s, a) \in S \times A$

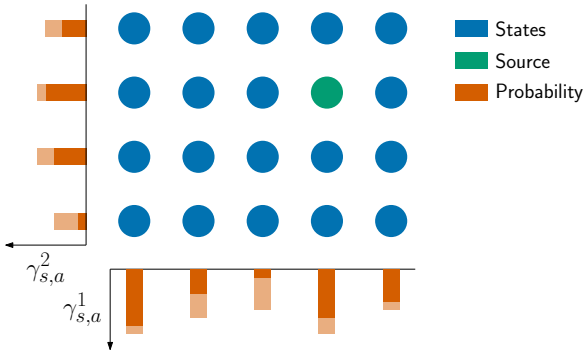


Figure 9.1: An example of an fIMDP where, given a source-action pair (s, a) (the green state), the ambiguity set for the transition probability can be decomposed into the product between two independent interval ambiguity sets, $\Gamma^1_{s,a}, \Gamma^2_{s,a}$. In this example, it is only necessary to store 9 transitions (per source-action pair) in contrast to 20 transitions for a traditional IMDP.

at time step k [122]. Given a strategy and an adversary, an fIMDP collapses to a finite Markov chain, with the transition probability matrix specified by marginal distributions.

9.2.1. ANALYSIS OF SPACE COMPLEXITY

Before proceeding further, we now discuss the memory requirements to store an fIMDP. To simplify the presentation, we assume that $\mathcal{G}$ is a complete bipartite graph and that the number of states in each marginal is equal, that is, $|S_i| = |S_1|$ for all $i = 1, \dots, n$. Hence, we have that the number of states along each marginal is $|S_i| = \sqrt[n]{|S|}$. To store an odIMDP, we need to store the bounds on the transition probability. Then, for each source-action pair, we must store bounds for the ambiguity set along each marginal. That is, the upper and lower bound for $|S_i|$ states for each marginal. In total, this requires storing $2|S||A|n\sqrt[n]{|S|}$ scalar values. This is a crucial difference compared to traditional IMDP, where, as discussed in Section 7.1, for each state-action pair, we need to store $2|S|^2|A|$ scalar values. An example to clarify the different memory requirements between fIMDP and IMDP is reported in Figure 9.1.

Example 9.1. *An, albeit pathological, example of extremely sparse dependencies and thus potential in formulating as an fIMDP is from [162] where* $\mathbf{x}[k+1] = 0.8\mathbf{x}[k] + 0.2\mathbf{v}[k]$, $\mathbf{v}[k]$ *has diagonal normal distribution, and* $X = X_s = [-1, 1]^n$ *for varying* n*, with the purpose of testing scalability. The concrete abstraction procedure is of lesser importance; however, note that each axis* $\mathbf{x}_i[k]$ *is independent. Thus, when abstracting, we may take the parent of the corresponding state variable to only be itself the time step before. Following [162] and [152], we assume two segments per axis in the abstraction such that the number of states is* 2^n*, e.g.* $16\,384$ *states for* $n = 14$*, which is the typical limit. To store the IMDP, it requires* 2^{2n+1} *scalar values. If we instead construct and store an odIMDP (see Section 9.3) with* n *state variables then it requires* $n2^{n+2}$ *scalar values to store; an exponential reduction compared to the IMDP. Finally, if we abstract to an fIMDP with the same*

state variable structure then, due to the decoupling, it requires $8n$ *to store; an exponential reduction compared to the odIMDP.*

The above example highlights the importance of factorization and sparsity for memory. How factorization and sparsity impact the time complexity is deferred to Subsection 9.4.1 where we describe probabilistic model checking, via value iteration, over fIMDPs.

9.2.2. COMPARISON OF AMBIGUITY SETS

At first glance, the ambiguity set of an fIMDPs may appear equivalent to that of an IMDP obtained by multiplying the interval bounds of the interval ambiguity sets of each marginal, for each source-action pair, as is commonly done (see (7.5)). That is, to construct the joint (interval) ambiguity set as

$$\bar{\Gamma}_{s,a} = \{\gamma \in \mathcal{D}(S) \,:\, \underline{\gamma}_{s,a}(t) = \prod_{i=1}^{n} \underline{\gamma}^{i}_{s,a}(t) \leq \gamma_{s,a}(t) \leq \prod_{i=1}^{n} \overline{\gamma}^{i}_{s,a}(t) = \overline{\gamma}_{s,a}(t),\ \forall t \in S\} \tag{9.2}$$

Remarkably, this is, however, a fallacy. In fact, multiplying the interval bounds of the marginal ambiguity sets introduces spurious distributions that are not part of the product ambiguity set, as they may not satisfy the additional constraints on the marginals that are present in fIMDPs. To better understand why this theoretical quirk occurs, consider the following example.

Example 9.2. *Figure 9.2 shows an fIMDP represented by an IMDP for each marginal. The states are numbered to index into the joint probability distribution as* $[0,1]^4$*. Furthermore, on the right, an interval ambiguity set is constructed from the fIMDP by multiplying the marginal bounds together, corresponding to the joint transition. If we consider a distribution* $[0.4, 0.3, 0.08, 0.22]^\top$*, we see that it is clearly contained in the joint interval ambiguity set on the right of Figure 9.2. However, since the value* 0.4 *in the first entry is equal to the upper bound for that entry in the ambiguity set, it forces* $[0.5, \cdot]$ *and* $[0.8, \cdot]$ *in the distribution for the vertical and horizontal marginals, respectively (because* $0.4 = 0.5 \cdot 0.8$*). For the vertical marginal to sum to one, we must have* $[0.5, 0.5]$*. The marginal decomposition of the last element* 0.22 *thus needs to be* $0.5p = 0.22$*, i.e.* $p = 0.22/0.5 = 0.44$*, which is not contained in the horizontal marginal ambiguity set (upper bound* 0.4*). Hence, the distribution contained in the joint interval ambiguity set cannot be factored into two distributions from the marginal ambiguity sets.*

The intuition provided in Example 9.2 is formalized in Theorem 9.1 below, where we show that the ambiguity set of an fIMDP is contained in that of an IMDP obtained by multiplying the interval bounds of the interval ambiguity sets of each marginal of the fIMDPs. This result will allow us to build abstractions for System (2.35) that not only require less memory to be stored compared to those described in Subsection 7.1, but are also guaranteed to provide tighter error bounds for the same partition size of the state space.

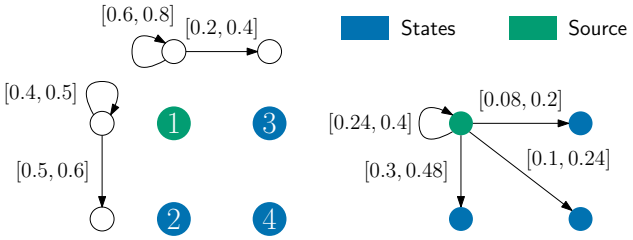


Figure 9.2: On the left, for the green state, we report two marginal interval ambiguity sets, i.e. an interval ambiguity set for each marginal of a product ambiguity set, with outgoing transitions to all other states. On the right, an IMDP is constructed by multiplying the interval bounds of the marginal ambiguity sets. By this multiplication of bounds, joint distributions are introduced that cannot be represented as a product of distributions from the marginal ambiguity sets.

Theorem 9.1. *For $S = S_1 \times \cdots \times S_n$, consider the interval ambiguity sets $\Gamma^1 \in \mathcal{D}(S_1), \dots, \Gamma^n \in \mathcal{D}(S_n)$, where $\Gamma^i = \{\gamma \in \mathcal{D}(S_i) : \underline{\gamma}^i(s^i) \leq \gamma(s^i) \leq \overline{\gamma}^i(s^i),\ \forall s^i \in S_i\}$. Call $\Gamma = \bigotimes_{i=1}^n \Gamma^i$. Then, it holds that $\Gamma \subseteq \bar{\Gamma}$ where*

$$\bar{\Gamma} = \left\{ \gamma \in \mathcal{D}(S) : \prod_{i=1}^n \underline{\gamma}^i(s^i) \leq \gamma(s) \leq \prod_{i=1}^n \overline{\gamma}^i(s^i),\ \forall s \in S \right\}. \tag{9.3}$$

Proof. It is sufficient to show that any distribution $\gamma \in \Gamma$ is also contained in $\bar{\Gamma}$, that is, it satisfies the interval bounds $\prod_{i=1}^n \underline{\gamma}^i(s^i) \leq \gamma(s) \leq \prod_{i=1}^n \overline{\gamma}^i(s^i)$ for each $s \in S$. By multiplying the (non-negative) inequalities $\underline{\gamma}^i(s^i) \leq \gamma^i(s^i) \leq \overline{\gamma}^i(s^i)$ for each component $i = 1, \dots, n$, it holds for each $s \in S$ that

$$\prod_{i=1}^n \underline{\gamma}^i(s^i) \leq \prod_{i=1}^n \gamma^i(s^i) \leq \prod_{i=1}^n \overline{\gamma}^i(s^i). \tag{9.4}$$

Therefore, we can conclude that $\gamma = \bigotimes_{i=1}^n \gamma^i$ is contained in $\bar{\Gamma}$. □

9

9.3. ABSTRACTION

In order to describe the abstraction process of System (2.35) into an fIMDP, we first start from the case where the transition kernel is Gaussian and then move to the more general case. In what follows, we assume that the region of interest $X \subset \mathbb{R}^n$ is hyperrectangular[1], that is, $X = [\underline{x}_1, \overline{x}_1] \times \cdots \times [\underline{x}_n, \overline{x}_n]$. To abstract a system into an fIMDP, we start by partitioning X into a grid $Q = Q_1 \times \cdots \times Q_n$ and associate with each (partitioned) axis $Q_i = \{q_1^i, \dots, q_{n_i}^i\}$ a state variable $S_i = \{s_1^i, \dots, s_{n_i}^i\}$. The collection of state variables defines the set of states $S = S_1 \times \cdots \times S_n$. We denote the (hyperrectangular) region associated with some state s by q_s. To avoid obtuse

[1]Note that if the region of interest is not hyperrectangular, but bounded, one can always over-approximate it with a hyperrectangular region.

notation, we omit the subscript of state variables s_j^i when it is clear to which we refer, for example, the composed state $s = (s^1, \dots, s^n)$ for some $s \in S$. For completeness, we assume for each state variable S_i that one value is a sink state $\check{s}^i$ to represent exiting the set $[\underline{x}_i, \overline{x}_i]$ such that Q effectively partitions $\mathbb{R}^n$ with some regions infinite in size. Finally, we select a finite subset $\tilde{U}$ of U and assign an abstract action $a \in A$ to each $u_a \in \tilde{U}$ such that the controller applied to the concrete system is a piecewise constant function of the state. We will describe the mapping from abstract to concrete control action in greater detail in Section 9.5 along with the correctness of the abstraction.

Remark 9.2. *It is possible to design more complicated controller mappings such as assigning each abstract action to compact subsets of the control space, e.g., as a safety shield, and PWA feedback controllers to allow tighter bounds abstractions and model order reductions [39].*

To make the above description more concrete, we provide for the remainder of this section a running example.

Example 9.3 (Running example.)**.** *Consider a stochastically-switched autonomous system with the following dynamics: the transition probability is a mixture of two Gaussians whose mean depends on x and the switching between the modes is governed by a Bernoulli random variable Z with $\mathbb{P}(Z = 0) = 0.7$ and*

$$\mathbf{x}[k+1] \sim \underbrace{\mathbb{P}(Z=0)}_{\alpha_1} p_1(\mathbf{x}[k]) + \underbrace{\mathbb{P}(Z=1)}_{\alpha_2} p_2(\mathbf{x}[k]) \tag{9.5}$$

We defer the definition of p_1 and p_2 to the following subsection, where the interval ambiguity sets of the abstract model are computed.

The region of interest X is the set $[-2, 2]^2$, which we uniformly partition along each axis into 40 segments. That is, $|Q_i| = |S_i| = 41$; the extra element is corresponding to X_i^c. This partitioning is depicted in Figure 9.3. As the system is autonomous, we choose $A = \{a\}$ whose value is irrelevant.

In the following subsections, we focus on computing bounds on the transition probability starting with the Gaussian case.

9.3.1. GAUSSIAN CASE

We start by considering the case of a Gaussian transition kernel:

$$T(X \mid x, u) = \int_X d\mathcal{N}(x' \mid \mu(x,u), \Sigma(x,u)). \tag{9.6}$$

where we assume $\Sigma(x,u)$ is diagonal for every $x, u \in X \times U$.

Remark 9.3. *Note that the transition kernel in (9.6) arises for many applications of practical interest. In fact, it arises any time f in System (2.35) is linear in $\mathbf{v}[k]$ (and possibly non-linear in $\mathbf{x}[k]$). This is the case for linear and non-linear systems with additive Gaussian noise [16], including NNDM systems [40] and a large class of systems learned through GP regression [64].*

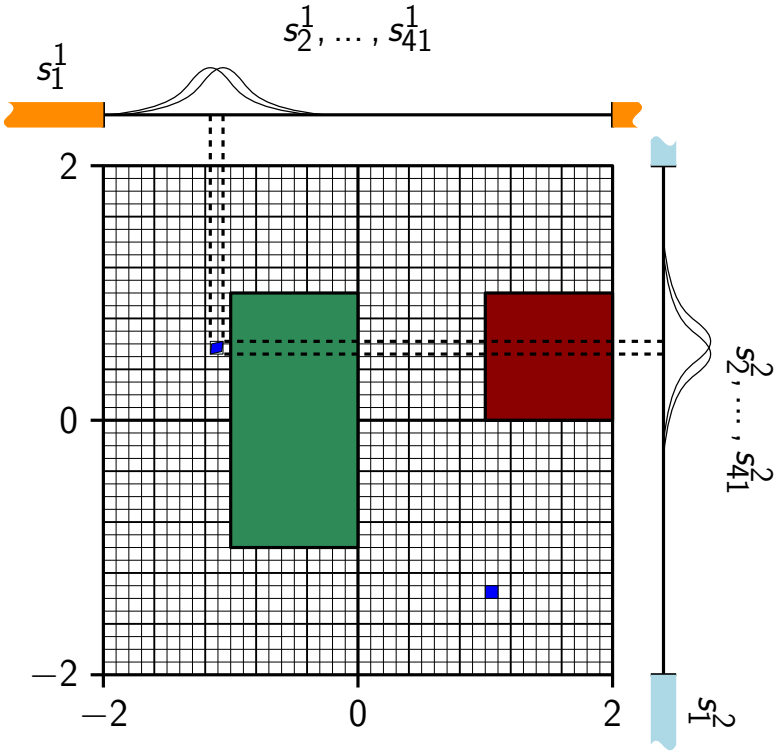


Figure 9.3: Pictorial representation of the running Example 9.3 of a linear stochastically-switched system. It is a 2D system where X is partitioned into 40 segments along each axis, i.e., the abstract state variables $s_2^i, \ldots, s_{41}^i$ for $i \in \{1, 2\}$; the state variables s_1^i, $i \in \{1, 2\}$ are reserved for X_i^c, which includes regions on both sides of the partitioned space. The objective is reach-avoid with the avoid region $[1, 2] \times [0, 1]$ (red) and the goal region $[-1, 0] \times [-1, 1]$ (green). The blue boxes depict how transition probability along each axis, starting the state $s = (s_{32}^1, s_8^2)$, is computed: (i) the starting region is transformed under the nominal dynamics to calculate the set of means (this is the parallelogram), (ii) the set of means is projected onto each axis, and (iii) the interval bounds $\underline{\gamma}_{s,a}^i$ and $\overline{\gamma}_{s,a}^i$ are computed analytically according to [16]. See Example 9.4 for more details on computing the interval bounds of this example.

First, we consider sparsity patterns in the transition kernel. In particular, we build dependency graphs $\mathcal{G}_\mu$ and $\mathcal{G}_\Sigma$ for $\mu(x,u)$ and $\Sigma(x,u)$, respectively. Then, we take the intersection $\mathcal{G}_\mu \cap \mathcal{G}_\Sigma$ and map it to the abstract system such that in the abstract dependency graph $\mathcal{G}$ there exists an edge from s^j to S_i if either $\mu(x,u)_i$ or $\Sigma(x,u)_{ii}$ depends on x_j, and similarly for actions.

Next, to define the marginal interval ambiguity sets $\Gamma^i_{(s,a)\cap \mathrm{Pa}_\mathcal{G}(S_i)}$, we need to introduce some notation. In particular, for each $(s,a) \cap \mathrm{Pa}_\mathcal{G}(S_i)$ we define intervals $[\underline{\mu}^i_{s,a}, \overline{\mu}^i_{s,a}]$ and $[\underline{\Sigma}^i_{s,a}, \overline{\Sigma}^i_{s,a}]$ such that for all $x \in q_s$:

$$\underline{\mu}^i_{s,a} \leq \mu(x,u_a)_i \leq \overline{\mu}^i_{s,a} \quad \text{and} \quad \underline{\Sigma}^i_{s,a} \leq \Sigma(x,u_a)_{ii} \leq \overline{\Sigma}^i_{s,a}. \tag{9.7}$$

That is, $[\underline{\mu}^i_{s,a}, \overline{\mu}^i_{s,a}]$ and $[\underline{\Sigma}^i_{s,a}, \overline{\Sigma}^i_{s,a}]$ represent interval bounds for the mean and variance of the i-th marginal of System (2.35) starting from region q_s. The computation of the bounds on mean and variance in (9.7) is a well-studied problem for which there exist tools based on convex optimization [16], [40], [46], [163] that can also be applied in the context of Gaussian process regression [164].

Then, computation of $\Gamma^i_{(s,a)\cap \mathrm{Pa}_\mathcal{G}(S_i)}$ reduces to computing for each (non-sink state) $t^i \in S_i$ the following transition probabilities

$$\underline{\gamma}^i_{s,a}(t^i) = \min_{\mu^i \in [\underline{\mu}^i_{s,a}, \overline{\mu}^i_{s,a}],\, \Sigma^i \in \{\underline{\Sigma}^i_{s,a}, \overline{\Sigma}^i_{s,a}\}} \int_{\underline{q}^i_t}^{\overline{q}^i_t} d\mathcal{N}(y_i \mid \mu^i, \Sigma^i), \tag{9.8}$$

$$\overline{\gamma}^i_{s,a}(t^i) = \max_{\mu^i \in [\underline{\mu}^i_{s,a}, \overline{\mu}^i_{s,a}],\, \Sigma^i \in \{\underline{\Sigma}^i_{s,a}, \overline{\Sigma}^i_{s,a}\}} \int_{\underline{q}^i_t}^{\overline{q}^i_t} d\mathcal{N}(y_i \mid \mu^i, \Sigma^i). \tag{9.9}$$

Analytical solutions for (9.8) and (9.9) exist [16], [40], [64] (see Section 7.1 for more details).

For the transition to the sink state $\check{t}^i$, we observe that this is equivalent to the dual probability of transitioning to X. Hence, we may compute the interval bounds by the following:

$$\underline{\gamma}^i_{s,a}(\check{t}^i) = 1 - \max_{\mu^i \in [\underline{\mu}^i_{s,a}, \overline{\mu}^i_{s,a}],\, \Sigma^i \in \{\underline{\Sigma}^i_{s,a}, \overline{\Sigma}^i_{s,a}\}} \int_{\underline{x}_i}^{\overline{x}_i} d\mathcal{N}(y_i \mid \mu^i, \Sigma^i), \tag{9.10}$$

$$\overline{\gamma}^i_{s,a}(\check{t}^i) = 1 - \min_{\mu^i \in [\underline{\mu}^i_{s,a}, \overline{\mu}^i_{s,a}],\, \Sigma^i \in \{\underline{\Sigma}^i_{s,a}, \overline{\Sigma}^i_{s,a}\}} \int_{\underline{x}_i}^{\overline{x}_i} d\mathcal{N}(y_i \mid \mu^i, \Sigma^i). \tag{9.11}$$

Finally, for completeness, the sink states are absorbing, that is, for all states $s = (s^1, \ldots, s^{i-1}, \check{s}^i, s^{i+1}, \ldots, s^n)$ for some $i \in \{1, \ldots, n\}$, we define $\underline{\gamma}^i_{s,a}(\check{s}^i) = \overline{\gamma}^i_{s,a}(\check{s}^i) = 1$ and zero otherwise. From the computation of all interval bounds, we can define Γ

as the product of marginal interval ambiguity sets $\Gamma^i_{s,a\cap \mathrm{Pa}_{\mathcal{G}}(S_i)}$ for each source-action pair (s,a). Then, the fIMDP is $M = (S, a, \mathcal{G}, \Gamma)$.

Example 9.4 (Running example cont.)**.** *Continuing with Example 9.3, the transition kernel is defined by*

$$\begin{aligned} p_1(\mathbf{x}[k]) &= \mathcal{N}\Bigg(\underbrace{\begin{pmatrix} 0.1 & 0.9 \\ 0.8 & 0.2 \end{pmatrix}\mathbf{x}[k]}_{\mu^1(\mathbf{x}[k])}, \underbrace{\begin{pmatrix} 0.3^2 & 0 \\ 0 & 0.2^2 \end{pmatrix}}_{\Sigma^1} \Bigg) \\ p_2(\mathbf{x}[k]) &= \mathcal{N}\Bigg(\underbrace{\begin{pmatrix} 0.8 & 0.2 \\ 0.1 & 0.9 \end{pmatrix}\mathbf{x}[k]}_{\mu^2(\mathbf{x}[k])}, \underbrace{\begin{pmatrix} 0.2^2 & 0 \\ 0 & 0.1^2 \end{pmatrix}}_{\Sigma^2} \Bigg). \end{aligned} \tag{9.12}$$

Because in this subsection we restrict ourselves to the Gaussian case for simplicity, we simplify these dynamics slightly. In particular, we assume that the dynamics are described by $\mathbf{x}[k+1] \sim p_1(\mathbf{x}[k])$*. We show in Figure 9.3 how the transition probability bounds along each axis are computed, starting from the state* $s = (s^1_{32}, s^2_8)$ *corresponding to the region* $[1, 1.1] \times [-1.4, -1.3]$*. To compute the interval bounds* $\underline{\gamma}^i_{s,a}$ *and* $\overline{\gamma}^i_{s,a}$*, we first compute the transformation of this region under the linear dynamics* $\mu^1(\cdot)$*; this represents the set of possible means starting from* s*. Since the transformation is linear, the set of means is a parallelogram, in this case, with the vertices* $[-1.06, 0.62]$*,* $[-1.15, 0.6]$*,* $[-1.16, 0.52]$*, and* $[-1.07, 0.54]$*; if the dynamics were non-linear, it would be necessary to* over-approximate *the set of means. We then project this set onto each axis, which does introduce some conservatism as the projection is equivalent to over-approximating the set of means by a rectangle. However, the more expressive representation of fIMDPs often improves the accuracy beyond this conservatism. Moreover, the projection is necessary to build the fIMDP abstraction. Now, because the covariance of* p_1 *is diagonal and we have projected the set of means onto each axis, we can compute* $\underline{\gamma}^i_{s,a}(t^i)$ *and* $\overline{\gamma}^i_{s,a}(t^i)$ *for each* t^i *analytically using equation (17) from [16]. For the transition to* s^1_1 *and* s^2_1*, that is, the orange and light blue segments in Figure 9.3, we compute this, using the same projected sets of means, as the complement of staying within the region of interest, i.e., the probability of* not *entering the orange and light blue segments. This process is then repeated for every starting state.*

9.3.2. MIXTURE MODELS

To extend the applicability, we consider Gaussian mixture models with the following transition kernel structure:

$$T(S \mid x, u) = \sum_{r \in R} \alpha_r(x,u) \int_X d\mathcal{N}(x' \mid \mu^r(x,u), \Sigma^r(x,u)), \tag{9.13}$$

where $\mu^r(x,u) \in \mathbb{R}^n$ and $\Sigma^r(x,u) \in \mathbb{S}^n_+$ are control- and state-dependent mean and covariance functions of the r-th Gaussian in the mixture. $\alpha_r(x,u)$ is a weighting

function such that $\alpha_r(x,u) \geq 0$ and $\sum_{r\in R} \alpha_r(x,u) = 1$ for each source-action pair (x,u). We assume μ^r, Σ^r, and α_r to be continuous in x for each u.

The transition kernel (9.13) is a generalization of the Gaussian case with $|R| > 1$. However, a key additional technical difficulty is that even if the covariance matrix of each Gaussian in the mixture is diagonal, then the covariance matrix for the mixture distribution is not necessarily diagonal. Specifically, the joint covariance of the mixture distribution in (9.13), omitting the dependence on (x,u), is

$$\Sigma = \sum_{r\in R}^{K} \alpha_r \Sigma^r + \sum_{r\in R} \alpha_r (\mu^r - \bar{\mu})(\mu^r - \bar{\mu})^\top \tag{9.14}$$

with the joint mean $\bar{\mu} = \sum_{r\in R} \alpha_r \mu^r$. Due to the second term of (9.14), the joint covariance Σ is generally not diagonal. Consequently, we cannot directly abstract the system to fIMDPs as it does not exhibit the required product form. To solve this issue, in what follows, we build an fIMDPs for each Gaussian in the mixture following the approach in Subsection 9.3.1 and then abstract System (2.35) into a mixture of fIMDPs.

Definition 9.2. *A mixture of R fIMDPs with n marginals is a tuple $M = (S, A, \mathcal{G}, \Gamma, \Gamma^\alpha)$ where*

- *$S = S_1 \times \cdots \times S_n$ is a finite set of joint states with S_i being the set of states in marginal i,*
- *$A = A_1, \ldots, A_m$ is a finite set of joint actions,*
- *$\mathcal{G} = \{\mathcal{G}_r\}_{r\in R}$ is the set of dependency graphs for each element in the mixture,*
- *$\Gamma = \{\Gamma_{r,s,a}\}_{r\in R, s\in S, a\in A}$ are a set of R product ambiguity sets, where $\Gamma_{r,s,a} = \bigotimes_{i=1}^{n} \Gamma^i_{r,(s,a)\cap \mathrm{Pa}_{\mathcal{G}_r(S_i)}}$ with $\Gamma^i_{r,(s,a)\cap \mathrm{Pa}_{\mathcal{G}_r(S_i)}} \in \mathrm{int\,amb}(S_i)$, and*
- *$\Gamma^\alpha = \{\Gamma^\alpha_{s,a}\}_{s\in S, a\in A}$ are sets of feasible weightings distributions, where $\Gamma^\alpha_{s,a} \in \mathrm{int\,amb}(R)$. A feasible weighting distribution for a source-action pair (s,a) is denoted by $\alpha_{s,a} \in \Gamma^\alpha_{s,a}$.*

The model $M = (S, A, \mathcal{G}, \Gamma, \Gamma^\alpha)$ can be interpreted as a set of R fIMDPs sharing the same (decomposed) states S and the same set of actions A, but that can have different individual ambiguity sets $\Gamma_{r,s,a}$ for each source-action pair (s,a). The weighting $\alpha_{s,a} \in \Gamma^\alpha_{s,a}$ defines the probability of selecting a component from the mixture. Thus, a feasible distribution for M is any distribution $\gamma_{s,a} = \sum_{r\in R} \alpha_{s,a}(r)\gamma^r_{s,a}$, where $\alpha_{s,a} \in \Gamma^\alpha_{s,a}$ and $\gamma^r_{s,a} \in \Gamma_{r,s,a}$ for each $r \in R$.

The process of abstracting System (2.35) to a mixture of fIMDPs, is as follows:

1. S and A are computed as in the introduction of this Section 9.3 and are shared among all Gaussians in the mixture,
2. for the r-th Gaussian in the mixture follow the approach in Subsection 9.3.1 to compute $\mathcal{G}_r$ and $\Gamma_{r,s,a}$ for any source-action pair $(s,a) \in S \times A$, and

3. for each source-action pair (s,a), the interval ambiguity set $\Gamma^{\alpha}_{s,a} \in \text{int amb}(R)$ is constructed such that for all $x \in q_s$ there exists $\alpha_{s,a} \in \Gamma^{\alpha}_{s,u_a}$ with $\alpha^r(x,a) = \alpha_{s,a}(r)$ for all $r \in R$. Similar to the marginal interval ambiguity sets case, we do that by constructing $\Gamma^{\alpha}_{s,a}$ such that for each $r \in R$ for all $\alpha_{s,a} \in \Gamma^{\alpha}_{s,a}$ it holds that $\min_{x \in s} \alpha^r(x,a) \leq \alpha_{s,a}(r) \leq \max_{x \in s} \alpha^r(x,a)$.

Example 9.5 (Running example cont.)**.** *Following the process described above, the complete dynamics in Example 9.4 are abstracted as follows:* $\Gamma_{1,s,a}$ *and* $\Gamma_{2,s,a}$*, corresponding to* p_1 *and* p_2*, respectively, are computed using the procedure described in Example 9.4. Finally,* $\Gamma^{\alpha}_{s,a}$ *is trivially constructed, as* α *is independently of* s *and* a*. That is,* $\alpha_{s,a}(1) = \alpha_1 = 0.7$ *an* $\alpha_{s,a}(2) = \alpha_2 = 0.3$*.*

9.4. PROBABILISTIC MODEL CHECKING

With an abstraction of System (2.35) into an fIMDPs or a mixture of fIMDPs, it remains to synthesise an optimal strategy that can then be mapped back to the original system. The construction of the abstract specification and the mapping of the controller and guarantees back to the original system is deferred to Section 9.5. In this section, we simply assume access to a set of safe sets $S_s \subset S$ and the goal is to compute for every state $s_0 \in S_0$

$$P^{s_0}_{\text{safe,abstract}} = \max_{\pi \in \Pi} \min_{(\gamma)^K \in \Gamma^K} \mathbb{P}^{s_0,\pi}_{(\gamma)^K}[\omega[k] \in S_s,\ \forall k \in \{0,\dots,K\}]. \tag{9.15}$$

where $\mathbb{P}^{s_0,\pi}_{(\gamma)^K}$ denotes the probability measure over the random variable of a path ω, which starts in s_0 and transitions according to the controller π and the distribution $(\gamma)^K$ at each time step. To dissect, the goal is to synthesize the controller that maximizes the lower bound on the probability of staying in the safe set S_s over a finite horizon K with a dynamic adversary (i.e. the adversary can pick a new distribution at every time step). In what follows, we consider the fIMDPs case; the mixture case follows similarly as we will illustrate in (9.22).

For an fIMDPs $M = (S, A, \mathcal{G}, \Gamma)$ synthesizing an optimal strategy and computing the value for $P_{\text{safe,abstract}}$ reduces to solving the following pessimistic robust value iteration [122], [138] (see also Section 8.4):

$$\begin{aligned} V_0(s) &= \mathbf{1}_{S_u}(s) \\ V_k(s) &= \min_{a \in A} g\left(s, \max_{\gamma_{s,a} \in \Gamma_{s,a}} \sum_{t \in S} V_{k-1}(t)\gamma_{s,a}(t)\right) \\ &= \min_{a \in A} g\left(s, \max_{\gamma^i_{s,a} \in \Gamma^i_{s,a},\, i=1,\dots,n} \sum_{t \in S} V_{k-1}(t) \prod_{i=1}^{n} \gamma^i_{s,a}(t^i)\right), \end{aligned} \tag{9.16}$$

where $S_u = S \setminus S_s$ and $g(s,v) = \mathbf{1}_{S_u}(s) + \mathbf{1}_{S_s}(s)v$. Then, $P^{s_0}_{\text{safe,abstract}} = 1 - V_K(s_0)$. Each step of (9.16) is called a Bellman update. The last equality of the Bellman update is due to the fact that for fIMDPs $\Gamma_{s,a} = \otimes_{i=1}^{n} \Gamma^i_{s,a}$. That is, the ambiguity

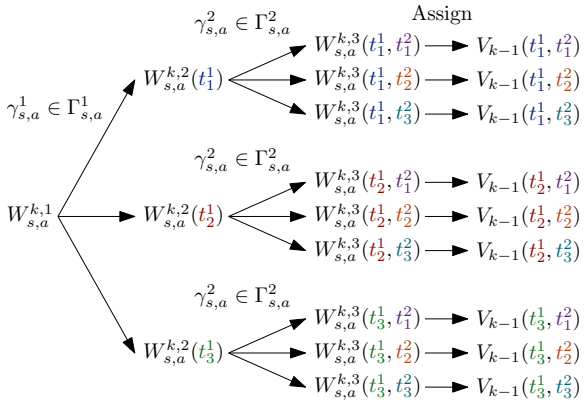


Figure 9.4: An example of the recursive structure for the proposed algorithm for value iteration over fIMDPs. The example is for an fIMDP with two marginals of three states each. Notice going right-to-left that we optimize over $\gamma^2_{s,a}$ three times for each subproblem, we assume t^1 is given.

set is the product of the marginal ambiguity sets. If a strategy is given, we denote the value function by $V^\pi_k(s)$ and if furthermore the inner maximization is replaced by a minimization, we use the notation $\check{V}^\pi_k(s)$. As a consequence of the product structure, solving (9.16) reduces to iteratively solving the following optimization problem:

$$\max_{\gamma^i_{s,a} \in \Gamma^i_{s,a},\, i=1,\dots,n} \sum_{t \in S} V_{k-1}(t) \prod_{i=1}^{n} \gamma^i_{s,a}(t^i) \tag{9.17}$$

Because of its multilinear structure, (9.17) cannot be solved directly using standard linear programming approaches for IMDPs, e.g. O-maximization [122]. To solve this issue, we develop a divide-and-conquer approach in which we recursively compute a lower bound on (9.17) by decomposing the problem into a set of linear programming. Specifically, we compute an under-approximation of (9.17), $W^k_{s,a}$ defined recursively as follows:

$$\begin{aligned} W^{k,n}_{s,a}(t^1,\dots,t^n) &= V_{k-1}(t) \\ W^{k,i-1}_{s,a}(t^1,\dots,t^{i-1}) &= \max_{\gamma^i_{s,a} \in \Gamma^i_{s,a}} \sum_{t^i \in S_i} W^{k,i}_{s,a}(t^1,\dots,t^i)\gamma^i_{s,a}(t^i), \\ &\qquad \text{for } i = 2,\dots,n \\ W^k_{s,a} := W^{k,0}_{s,a} &= \max_{\gamma^1_{s,a} \in \Gamma^1_{s,a}} \sum_{t^1 \in S_1} W^{k,1}_{s,a}(t^1)\gamma^1_{s,a}(t^1). \end{aligned} \tag{9.18}$$

Figure 9.4 gives an example of (9.18) for a 2-marginals fIMDP with three states in each marginal. The intuition is that we derive a bound for (9.17) by solving a set of simpler optimization problems, one marginal at a time. In particular, $W^{k,i-1}_{s,a}(t^1,\dots,t^{i-1})$ optimizes the expectation of $W^{k,i}_{s,a}$, which is itself a bound of

V_{k-1}, for the marginal i, while keeping the values of the other marginals fixed to $t^1, \dots, t^{i-1}$. Note that, as illustrated in Figure 9.4, the resulting algorithm is exponential in n. However, as each of the optimization problems is a particularly simple linear problem that can be solved efficiently using, e.g., the O-maximization algorithm [122], [138] and as we will show in Subsection 9.4.1, the resulting computational time complexity will still be lower compared to the time complexity for IMDPs.

In the rest of this section, in Theorem 9.2 and Proposition 9.1 we show, respectively that the approach in (9.18) is sound and guaranteed to improve the tightness of the results compared to the approach described in Section 7.

Lemma 9.1. *For any source-action pair $(s, a) \in S \times A$, it holds that*

$$W_{s,a}^{k,0} \geq \max_{\gamma_{s,a}^i \in \Gamma_{s,a}^i, i=1,\dots,n} \sum_{t \in S} V_{k-1}(t) \prod_{i=1}^{n} \gamma_{s,a}^i(t^i). \tag{9.19}$$

Proof.

$$\begin{aligned}
W_{s,a}^{k,0} &= \max_{\gamma_{s,a}^1 \in \Gamma_{s,a}^1} \sum_{t^1 \in S_1} W_{s,a}^{k,1}(t^1) \gamma_{s,a}^1(t^1) \\
&= \max_{\gamma_{s,a}^1 \in \Gamma_{s,a}^1} \sum_{t^1 \in S_1} \max_{\gamma_{s,a}^2 \in \Gamma_{s,a}^2} \sum_{t^2 \in S_2} W_{s,a}^{k,2}(t^1, t^2) \gamma_{s,a}^1(t^1) \gamma_{s,a}^2(t^2) \\
&\geq \max_{\gamma_{s,a}^1 \in \Gamma_{s,a}^1} \max_{\gamma_{s,a}^2 \in \Gamma_{s,a}^2} \sum_{t^1 \in S_1} \sum_{t^2 \in S_2} W_{s,a}^{k,2}(t^1, t^2) \gamma_{s,a}^1(t^1) \gamma_{s,a}^2(t^2) \\
&\geq \max_{\gamma_{s,a}^i \in \Gamma_{s,a}^i, i=1,\dots,n} \sum_{t^1 \in S_1} \cdots \sum_{t^n \in S_n} W_{s,a}^{k,n}(t^1, \dots, t^n) \prod_{i=1}^{n} \gamma_{s,a}^i(t^i) \\
&= \max_{\gamma_{s,a}^i \in \Gamma_{s,a}^i, i=1,\dots,n} \sum_{t \in S} V_{k-1}(t) \prod_{i=1}^{n} \gamma_{s,a}^i(t^i).
\end{aligned}$$

□

Theorem 9.2. *Let $\tilde{V}_0(s) = \mathbf{1}_{S_u}(s)$ and $\tilde{V}_k(s) = \min_{a \in A} g\left(s, W_{s,a}^k\right)$ where $W_{s,a}^k$ is computed as in (9.18). Further, let V_k be defined as in (9.16). Then, $\tilde{V}_K(s) \geq V_K(s)$ for every $s \in S$.*

Proof. The proof follows by induction of Lemma 9.1. In particular, we have $\tilde{V}_0(s) = V_0(s) = \mathbf{1}_{S_u}(s)$ as a base case. For the induction step, for any $k \in \mathbb{N}$, assume that $\tilde{V}_{k-1}(s) \geq V_{k-1}(s)$ for all $s \in S$, then by Lemma 9.1 it holds that $\tilde{V}_k(s) \geq V_k(s)$ for all $s \in S$. □

Note that an alternative option to the divide-and-conquer algorithm considered here would be to simply multiply the interval bounds of the marginal ambiguity

sets for each source-action pair, obtaining an ambiguity set identical to the one in Definition 7.4. As already mentioned, we avoid this approach because it would increase the conservativism. This is proved in the following proposition that guarantees that our approach in (9.18) is no more conservative than the one described in Section 7.1. In practice, in the experimental section, we show how our approach consistently returns substantially tighter bounds.

Proposition 9.1. *For an fIMDPs $M = (S, A, \mathcal{G}, \Gamma)$, consider the function $V_{k-1} : S \to \mathbb{R}$ and define the interval ambiguity set associated to M as*

$$\bar{\Gamma}_{s,a} = \left\{\gamma \in \mathcal{D}(S) : \underline{\gamma}_{s,a}(t) \leq \gamma_{s,a}(t) \leq \overline{\gamma}_{s,a}(t),\ \forall t \in S\right\}, \tag{9.20}$$

where $\underline{\gamma}_{s,a}(t) = \prod_{i=1}^{n} \underline{\gamma}^{i}_{s,a}(t^i)$ and $\overline{\gamma}_{s,a}(t) = \prod_{i=1}^{n} \overline{\gamma}^{i}_{s,a}(t^i)$ Then, for any $(s, a) \in S \times A$ it holds that $W^{k}_{s,a} \geq \min_{\gamma_{s,a} \in \bar{\Gamma}_{s,a}} \sum_{t \in S} V_{k-1}(t)\gamma_{s,a}(t^i)$.

Proof. To conclude the proof, it suffices to show that for any source-action pair $(s, a) \in S \times A$, the joint distribution $\gamma_{s,a}(t)$ obtained by optimizing (9.18) is contained $\bar{\Gamma}_{s,a}$. By definition of sets $\Gamma^{i}_{s,a}$ in (9.18) it holds that the i-th marginal of $\gamma_{s,a}(t)$ must be contained within $[\underline{\gamma}^{i}_{s,a}(t^i), \overline{\gamma}^{i}_{s,a}(t^i)]$. This implies that

$$\prod_{i=1}^{n} \underline{\gamma}^{i}_{s,a}(t^i) \leq \gamma_{s,a}(t) \leq \prod_{i=1}^{n} \overline{\gamma}^{i}_{s,a}(t^i). \tag{9.21}$$

□

We show in Figure 9.5 the impact of Proposition 9.1 on a concrete example where (9.18) on the fIMDP yields a value of 2.16 to a value of 2.06 for (9.16) on the IMDP.

Remark 9.4. *Subsequent to this work, [165] showed that any solution $\gamma^{i}_{s,a}$, $i \in \{1, \ldots, n\}$ to (9.17) (over polytopic marginal ambiguity sets) is a vertex of each marginal ambiguity set $\Gamma^{i}_{s,a}$ and proposed a solution method based on recursive McCormick relaxations, with the restriction that the auxiliary variables sum to one. The approach is only slightly slower and yields significantly tighter bounds compared to multiplying the interval bounds of the marginal ambiguity sets and applying O-maximization. In fact, for many cases, this McCormick relaxation-based approach is exact, as computed by a naïve vertex enumeration.*

The proposed divide-and-conquer algorithm extends easily to mixtures of fIMDPs by executing the algorithm in (9.18) for each element of the mixture and optimizing for the worst mixture distribution. That is, by solving:

$$W^{k}_{s,a} := \max_{\alpha \in \Gamma^{\alpha}_{s,a}} \sum_{r \in R} \alpha_r W^{k,0}_{r,s,a}, \tag{9.22}$$

where $W^{k,0}_{r,s,a}$ is computed as in (9.18). As $\Gamma^{\alpha}_{s,a}$ is an interval ambiguity set, the above is a linear program that can be efficiently solved using O-maximization [122].

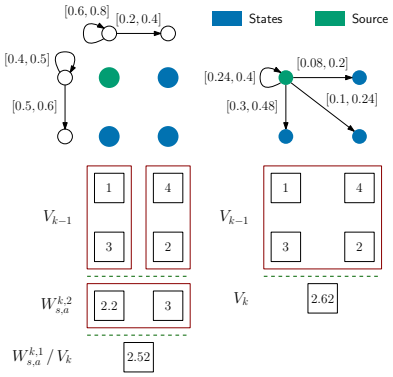


Figure 9.5: One (pessimistic) Bellman update on the example in Figure 9.2 with an fIMDP on the left and an IMDP on the right constructed by multiplying the marginal interval bounds of the fIMDP. The values immediately below the fIMDP and IMDP is the value function at the previous step V_{k-1}. For the fIMDPs, the Bellman update using (9.18) is computed by recursively using O-maximization [122], [138] with each red box outlining an O-maximization step. For the IMDP, the Bellman update using (9.16) is computed using a one-shot O-maximization step. The result is that the fIMDP and (9.18) together yields a tighter upper bound.

9.4.1. ANALYSIS OF TIME COMPLEXITY

Similarly to the analysis of the space complexity (see Subsection 9.2.1), we assume that the number of states in each marginal is equal, that is, $|S_i| = \sqrt[n]{|S|}$. Since value iteration is performed independently for each source-action pair, to compute the time complexity of the algorithm, we can analyze each pair separately. To do this, we start by observing the number of (branching) internal nodes in the recursion tree required to solve (9.18) (see Figure 9.4) is a geometric series with base factor $|S_i|$ up to exponent $n-1$, since the recursion tree splits with the base factor for a height of $n-1$. The complexity of the geometric series is $O((|S_i|)^{n-1})$ [166]. At each internal node of the tree, the O-maximization algorithm [138] is executed once on a set of size $|S_i|$. O-maximization is limited by the required sorting, which is of complexity $O(|S_i| \lg |S_i|)$. Now, since $|S_i| = \sqrt[n]{|S|} = |S|^{\frac{1}{n}}$, the complexity for each source-action pair is $O(|S| \lg |S_i|)$, and the full complexity is thus $O(|S|^2|A| \lg \sqrt[n]{|S|})$. In contrast, value iteration for a regular IMDP [122] has a time complexity of $O(|S|^2|A| \lg |S|)$. In practice, this difference in complexity is negligible and only confirms that fIMDPs are not more computationally expensive compared to traditional IMDPs.

9.5. CORRECTNESS

The probabilistic model checking guarantees a safety probability wrt. any fIMDP model. In this section, we are concerned about lifting this guarantee to a concrete system when abstracted by the method presented in Section 9.3. In other words, we are interested in the correctness of the abstraction and its relation to the verification of the abstract model. Before we establish the relation in Subsection 9.5.2, we first need to abstract the specification to ensure that the verification

of the abstract model is wrt. an abstract specification that can soundly be lifted to the concrete system.

9.5.1. ABSTRACTING SPECIFICATIONS

In addition to abstracting the state and action sets, with appropriate transition ambiguity sets between, it is necessary to abstract both the initial and safe sets X_0 and X_s such that verification on the abstract model can be lifted to the concrete system. Starting with X_0, since safety verification is computed as (the upper bound of) the dual probability of reaching $X_u = \mathbb{R}^n \setminus X_s$, it is necessary to over-approximate the initial set by the regions corresponding abstract states, i.e. to over-estimate the maximum probability of reaching X_u. Thus, we designate the following set of initial abstract states $S_0 = \{s \in S : q_s \cap X_0 \neq \emptyset\}$ where $q_s \in Q$ is the region in the partitioning Q of the state space $\mathbb{R}^n$ corresponding to the abstract state s (see Section 9.3). Similarly, we over-approximate X_u by selecting the (abstract) set $S_u = \{s \in S : q_s \cap X_u \neq \emptyset\}$ such that the probability of reaching X_u is over-estimated.

Remark 9.5. *It is common to consider for MDP-like models a singular initial state or a set of initial states S_0, and sometimes also a labelling function L, as part of the system tuple like $M = (S, S_0, A, L, \Gamma)$. However, in this, we consider S_0 and S_u to be part of the specification $\varphi_M = (S_0, S_u)$ for the following reason: the states, actions, and transition dynamics are invariant to the initial and unsafe sets φ_M. In fact, it is often a useful scenario to build one abstract model with the method in Section 9.3 and multiple different specifications as above. Furthermore, one can extend the methods of this chapter to other types of specifications, including complex temporal specifications, by substituting the function $g : S \times \mathbb{R} \to \mathbb{R}$ in (9.16) depending on the type of specification φ_M. This approach is taken in* **IntervalMDP.jl** *to support various specifications with a general implementation of the Bellman update [80].*

Computing an upper bound on the safety probability is often useful, e.g. to analyse convergence of the model checking in the infinite horizon case. This requires modifications in abstracting the specification, namely to *under-approximate* the initial and unsafe sets, that is, $S_0 = \{s \in S : q_s \subseteq X_0\}$ and $S_u = \{s \in S : q_s \subseteq X_u\}$. See [16, Section 7] for more details on the construction of upper-bound specifications and the cause of this theoretical quirk.

9.5.2. SIMULATION RELATION

To show the correctness of the abstraction and lift the safety guarantee to the concrete system $\{\mathbf{x}[k]\}_{k\in\mathbb{N}}$, proofs have traditionally relied on the value iteration and a mapping from concrete to abstract states, implicitly defining a relation between the abstract and concrete system. In this subsection, we make this (simulation) relation explicit and use the relation to prove that the safety guarantees for the abstract model applies to the concrete system. First, we define the following *asymmetric* simulation relation.

Definition 9.3 (Abstracting safety simulation relation)**.** *Let a (concrete) system $\mathbf{x}[k+1] = f(\mathbf{x}[k], u[k], \mathbf{v}[k])$ (i.e. (2.35)), an abstract fIMDP model $M = (S, A, \Gamma)$,*

initial sets $X_0 \subset X$ and $S_0 \subset S_X$, and unsafe sets $X_u \subset \mathbb{R}^n \setminus X_0$ and $S_u \subset S \setminus S_0$ be given. Then, a relation $\mathcal{R} \subset X \times S$ is an abstracting simulation relation if

- *for all $x_0 \in X_0$ there exists $s_0 \in S_0$ such that $(x_0, s_0) \in \mathcal{R}$,*
- *for all $(x, s) \in \mathcal{R}$, if $x \in X_u$ then $s \in S_u$, and*
- *for all $(x, s) \in \mathcal{R}$, if $s \in S_s$ then for every $a \in A$ there exists a control action $u_a \in U$ and a distribution $\gamma_{s,a} \in \Gamma_{s,a}$ with a coupling probability measure $\mathbb{W}$ on the measurable space $(X \times S, \mathcal{F})$ where $\mathcal{F}$ is the standard Borel σ-algebra for $X \times S$ such that*
  - *for any set $X \subset \mathbb{R}^n$, it holds $\mathbb{W}(X, S) = T(X \mid x, u_a)$,*
  - *for any $t \in S$, it holds $\mathbb{W}(\mathbb{R}^n, t) = \gamma_{s,a}(t)$, and*
  - $\mathbb{W}(\mathcal{R}) = 1$.

If there exists a simulation relation $\mathcal{R}$ between a concrete system $\{\mathbf{x}[k]\}_{k\in\mathbb{N}}$ and an abstract model M with the sets $\varphi_{\mathbf{x}} = (X_0, X_u)$ and $\varphi_M = (S_0, S_u)$ then we write the relation between the two systems as $(M, \varphi_M) \preceq (\{\mathbf{x}[k]\}_{k\in\mathbb{N}}, \varphi_{\mathbf{x}})$.

In the following Proposition 9.2, we prove that abstracting a stochastic control system according to the method presented in Section 9.3 implicitly builds a simulation relation between the concrete and abstract system.

Proposition 9.2. *Let a concrete system $\{\mathbf{x}[k]\}_{k\in\mathbb{N}}$, initial and unsafe sets $\varphi_{\mathbf{x}} = (X_0, X_u)$ be given. Then, construct an abstract model M according to the method presented in Section 9.3 and the sets $\varphi_M = (S_0, S_u)$ according to Subsection 9.5.1. Then, we have $(M, \varphi_M) \preceq (\{\mathbf{x}[k]\}_{k\in\mathbb{N}}, \varphi_{\mathbf{x}})$..*

Proof. To prove the statement, we construct a relation $\mathcal{R}$ and show that it satisfies the conditions in Definition 9.3. The relation is defined as follows:

$$\mathcal{R} = \{(x, s) \in \mathbb{R}^n \times S \,:\, x \in q_s\} \tag{9.23}$$

where $Q = \{q_1, \ldots, q_\ell\}$ defines a partitioning of the state space $\mathbb{R}^n$, with a bijective mapping between Q and S. Since Q is partition, by construction in Subsection 9.5.1, for every $x_0 \in X_0$ there exists a $s_0 \in S_0$ such that $x_0 \in q_{s_0}$ and $(x_0, s_0) \in \mathcal{R}$. Similarly, if $(x, s) \in \mathcal{R}$ then $x \in q_s$ and thus if $x \in X_u$ then it must hold $q_s \cap X_u \neq \emptyset$ and $s \in X_u$.

Therefore, it remains only to show that for each $(x, s) \in \mathcal{R}$ where $s \in S_s$ and for each $a \in A$ there exists a coupling probability measure $\mathbb{W}$ for some control action $u_a \in U$ and distribution $\gamma_{s,a} \in \Gamma_{s,a}$. Assume that the distribution $\gamma_{s,a}$ is given such that $T(q_t \mid x, u_a) = \gamma_{s,a}(t)$ for all $t \in S$. The existence of such a distribution follows from the fact that by construction for each $t \in S$, we have $T(q_t \mid x, u_a) = \prod_{i=1}^n T^i(q_t^i \mid x, u_a) \in \Gamma_{s,a}$ since $T^i(q_t^i \mid x, u_a) \in \Gamma_{s,a}^i$ and $\Gamma_{s,a} = \otimes \Gamma_{s,a}^i$. Thus, there must exist a distribution $\gamma_{s,a} \in \Gamma_{s,a}$ such that $T(q_t \mid x, u_a) = \gamma_{s,a}(t)$ for each $t \in S$. Then, the coupling measure is $\mathbb{W}(X, t) = T(X \cap q_t \mid x, u_a)$. Clearly, $\mathbb{W}(X, S) = T(X \mid x, u_a)$ for any set $X \subset \mathbb{R}^n$ and $\mathbb{W}(\mathbb{R}^n, t) = T(\mathbb{R}^n \cap q_t \mid x, u_a) = \gamma_{s,a}(t)$ for any $t \in S$. Finally, $\mathbb{W}(\mathcal{R}) = \sum_{t\in S} \mathbb{W}(q_t, t) = 1$ since Q is a partition, which concludes the proof. □

Proposition 9.2 relies on a bijective mapping through the partitioning Q of the state space $\mathbb{R}^n$. However, other abstracting safety simulation relations may exist and the conditions for such relations in Definition 9.3 are sufficient to guarantee that the safety probability for the abstract model lower bounds the safety probability of the concrete system. We will formalize the idea in the following proposition.

Proposition 9.3. *Let a (concrete) system* $\mathbf{x}[k+1] = f(\mathbf{x}[k], u[k], \mathbf{v}[k])$ *(i.e. (2.35)), an abstract fIMDP model* $M = (S, A, \Gamma)$*, initial sets* $X_0 \subset X$ *and* $S_0 \subset S_X$*, and unsafe sets* $X_u \subset \mathbb{R}^n \setminus X_0$ *and* $S_u \subset S \setminus S_0$ *be given with an abstracting simulation relation* $(M, \varphi_M) \preceq (\{\mathbf{x}[k]\}_{k\in\mathbb{N}}, \varphi_{\mathbf{x}})$ *between them with the relation* $\mathcal{R}$*. Furthermore, let* $\{V_k\}_{k\in\mathbb{N}}$ *be a sequence of value functions* $V_k : S \to [0,1]$ *computed by (9.16) for* M *and* S_u *and let* $\{\hat{V}_k\}_{k\in\mathbb{N}}$ *be a sequence of value functions* $\hat{V}_k : \mathbb{R}^n \to [0,1]$ *computed by (2.31) for* $\{\mathbf{x}[k]\}_{k\in\mathbb{N}}$ *and* X_u*. Then, for any* $(x,s) \in \mathcal{R}$ *and for any* $k \in \mathbb{N}$ *it holds that* $V_k(s) \geq \hat{V}_k(x)$*.*

Proof. The proof proceeds by induction over $k \in \mathbb{N}$. For the base case $k = 0$, for any $(x,s) \in \mathcal{R}$ we have $V_0(s) = \mathbf{1}_{S_u}(s)$ and $\hat{V}_0(x) = \mathbf{1}_{X_u}(x)$. Since $(x,s) \in \mathcal{R}$, we have that $x \in X_u$ implies $s \in S_u$, thus $V_0(s) \geq \hat{V}_0(x)$. For the inductive hypothesis, we assume that $V_k(s) \geq \hat{V}_k(x)$ for all $(x,s) \in \mathcal{R}$ and we want to show that $V_{k+1}(s) \geq \hat{V}_{k+1}(x)$ for all $(x,s) \in \mathcal{R}$. To this end, fix any $(x,s) \in \mathcal{R}$. Then, the respective Bellman updates are the following

$$V_{k+1}(s) = \min_{a\in A}\left(\mathbf{1}_{S_u}(s) + \mathbf{1}_{S_s}(s) \max_{\gamma_{s,a}\in\Gamma_{s,a}} \mathbb{E}_{t\sim\gamma_{s,a}}[V_k(t)]\right), \tag{9.24}$$

$$\hat{V}_{k+1}(x) = \min_{u\in U}\left(\mathbf{1}_{X_u}(x) + \mathbf{1}_{X_s}(x)\mathbb{E}_{\mathbf{v}}[\hat{V}_k(f(x,u,\mathbf{v}))]\right). \tag{9.25}$$

If $s \in S_u$ then $V_{k+1}(s) = 1 \geq \hat{V}_{k+1}(x)$. If instead $s \in S_s$ then $x \in X_s$ and thus it suffices to show that $\min_{a\in A} \max_{\gamma_{s,a}\in\Gamma_{s,a}} \mathbb{E}_{t\sim\gamma_{s,a}}[V_k(t)] \geq \min_{u\in U} \mathbb{E}_{\mathbf{v}}[\hat{V}_k(f(x,u,\mathbf{v}))]$. Select $a^\star = \operatorname{argmin}_{a\in A} \max_{\gamma_{s,a}\in\Gamma_{s,a}} \mathbb{E}_{t\sim\gamma_{s,a}}[V_k(t)]$. Then, since $(x,s) \in \mathcal{R}$, there exists a control input $u_{a^\star}$ and a distribution $\gamma_{s,a^\star} \in \Gamma_{s,a^\star}$ with a coupling $\mathbb{W}$. Therefore, we can rewrite the expectation of the concrete system as follows:

$$\mathbb{E}_{\mathbf{v}}[\hat{V}_k(f(x,u_{a^\star},\mathbf{v}))] = \int_{\mathcal{R}} \hat{V}_k(y)\, d\mathbb{W}(y,t) \leq \int_{\mathcal{R}} V_k(t)\, d\mathbb{W}(y,t) = \mathbb{E}_{t\sim\gamma_{s,a^\star}}[V_k(t)]. \tag{9.26}$$

The first equality holds because $\mathbb{W}(X,S) = T(X \mid x, u_a)$ for any set $X \subset \mathbb{R}^n$ and $\mathbb{W}(\mathcal{R}) = 1$. The inequality follows from $V_k(t) \geq \hat{V}_k(y)$ for all $(y,t) \in \mathcal{R}$ and the last equality from $\mathbb{W}(\mathbb{R}^n, t) = \gamma_{s,a}(t)$ for any $t \in S$. Since this statement holds for a specific distribution $\gamma_{s,a} \in \Gamma_{s,a}$, we conclude that for $a^\star$ it holds $\max_{\gamma_{s,a^\star}\in\Gamma_{s,a^\star}} \mathbb{E}_{t\sim\gamma_{s,a^\star}}[V_k(t)] \geq \mathbb{E}_{\mathbf{v}}[\hat{V}_k(f(x,u_{a^\star},\mathbf{v}))]$. Finally, since $u_{a^\star} \in U$, we conclude $\min_{a\in A} \max_{\gamma_{s,a}\in\Gamma_{s,a}} \mathbb{E}_{t\sim\gamma_{s,a}}[V_k(t)] \geq \min_{u\in U} \mathbb{E}_{\mathbf{v}}[\hat{V}_k(f(x,u,\mathbf{v}))]$ and $V_{k+1}(s) \geq \hat{V}_{k+1}(x)$ for all $(x,s) \in \mathcal{R}$. □

9.6. EMPIRICAL EVALUATION

To show the efficacy of our approach, we conducted a wide range of empirical studies with benchmarks ranging from linear 2D systems to non-linear systems,

Table 9.1: A summary of the benchmarks and their characteristics. Note that the number of states for the different experiments is the specified amount unless otherwise stated.

Name	Dimension	Dynamics type	Property	# inputs	# states
Car parking [39]	2	Additive linear	Reach-avoid	9	1600
Robot reachability [152]	2	Additive linear (non-linear control)	Reachability	121	400
Robot reach-avoid [152]	2	Additive linear (non-linear control)	Reach-avoid	441	1600
Building automation system [152]	4	Additive affine	Safety	4	1225
Van der Pol [39]	2	Additive polynomial	Reachability	11	2500
NNDM Cartpole	4	Additive NNDM	Safety	2	192000
6D linear model	6	Additive linear	Safety	0	262144
7D linear model	7	Additive linear	Safety	0	2097152
Dubin's car GP [64]	3	Gaussian process with deep kernel learning	Reach-avoid	7	25600
Stochastic switched linear	2	Gaussian mixture, each linear	Reach-avoid	0	1600

linear 7D systems, Neural Network Dynamic Models (NNDMs), and Gaussian mixture models (see the following Subsection 9.6.1). All benchmarks are verified for a time horizon $K = 10$ unless otherwise specified. The benchmarks include generalizations beyond safety to reachability and reach-avoid properties; it requires only minor modifications to the function $g(s, v)$ in (9.16) as stressed in Remark 2.4 of Chapter 2. We compare our results against state-of-the-art tools for abstractions to IMDPs and MDPs. In particular, for IMDPs, we compare with the method in [40, Theorem 1], as well as **IMPaCT** [152], which uses non-linear optimization and Monte Carlo integration to compute the transition probabilities needed to compute the abstraction as illustrated in Section 7.1. **IMPaCT** uses a dense representation and [40, Theorem 1] uses a sparse representation of the ambiguity set. Furthermore, we also compare with **SySCoRe** [39], which is based on model reduction and abstractions to MDPs. To run value iteration for both the proposed method and the method of [40], we use **IntervalMDP.jl** [80], which is our package for parallelized value iteration over IMDPs and odIMDPs written in Julia [154]. The experiments are all run on the TU Delft supercomputer, DelftBlue [167], on one memory partition node with an Intel Xeon E5-6248R CPU (12 cores allocated) and 200 GB memory. We stress that despite recent advances in performing value iteration on GPUs [80], [152], the experiments are all conducted on CPUs.

We have explored comparing against **SReachTools** [168], however, this tool supports a different class of problems. Namely that, *given an initial region*, find a reachability tube with high probability, or given a target tube, find a set of initial states with a minimum probability of satisfaction. We have also tested **StocHy** [162], but the tool crashed in 3 of its 4 case studies and we have not succeed in running **StocHy** on our benchmarks.

9.6.1. BENCHMARKS

A summary of the benchmarks and their core properties can be found in Table 9.1.

Car parking A 2D case study from [39] of parking a car with dynamics $\mathbf{x}[k+1] = 0.9\mathbf{x}[k]+0.7\mathbf{u}[k]+\mathbf{v}[k]$ with $p_{\mathbf{v}} = \mathcal{N}(\cdot \mid 0, I)$. The region of interest is $X = [-10, 10]^2$ and the input space $U = \{-1, 0, 1\}^2$. The reach region is $R = [4, 10] \times [-4, 0]$ and the avoid region is $O = [4, 10] \times [0, 4]$. The goal is to maximize the pessimistic

reach-avoid probability over 10 time steps. We consider a partition of the region of interest of $(40, 40)$.

2D robot A 2D case study from [152] of a planar robot with non-linear input

$$\mathbf{x}[k+1] = \mathbf{x}[k] + 10\mathbf{u}_1[k]\begin{pmatrix}\cos(\mathbf{u}_2[k])\\ \sin(\mathbf{u}_2[k])\end{pmatrix} + \mathbf{v}[k] \tag{9.27}$$

where $p_{\mathbf{v}} = \mathcal{N}(\cdot \mid 0, 0.75I)$. The region of interest is $X = [-10, 10]^2$ and the control space is $U = [-1, 1]^2$ uniformly discretized into L^u points. The benchmark is similar to the car parking benchmark, except with non-linear inputs. For running the benchmark using our methods, we transform the input to $\tilde{\mathbf{u}} = \begin{pmatrix}\mathbf{u}_1\cos(\mathbf{u}_2) & \mathbf{u}1\sin(\mathbf{u}_2)\end{pmatrix}^\mathsf{T}$ such that the dynamics are linear in the (non-linearly partitioned) control inputs.

We consider two different specifications: reachability and reach-avoid, each with a different partitioning. For both specifications, the reach region is $R = [5, 7]^2$ and for the reach-avoid specification, the avoid region is $O = [-2, 2]^2$. For reachability, we use a state partitioning of $(20, 20)$ and an action partitioning of $(11, 11)$, while for reach-avoid we use a state partitioning of $(40, 40)$ and an action partitioning of $(21, 21)$. For both properties, we maximize the pessimistic reach-avoid probability over 10 time steps.

4D building automation system A 4D case study from [152] of building automation system with the linear dynamics $\mathbf{x}[k+1] = A\mathbf{x}[k] + B\mathbf{u}[k] + \mathbf{v}[k]$ where $p_{\mathbf{v}} = \mathcal{N}(\cdot \mid 0, \Sigma)$ with

$$\begin{aligned} A &= \begin{pmatrix} 0.6682 & 0 & 0.02632 & 0 \\ 0 & 0.683 & 0 & 0.02096 \\ 1.0005 & 0 & -0.000499 & 0 \\ 0 & 0.8004 & 0 & 0.1996 \end{pmatrix}, \\ B &= \begin{pmatrix} 0.1320 \\ 0.1402 \\ 0 \\ 0.0 \end{pmatrix}, \text{ and} \\ \Sigma &= \text{diag}(1/12.9199, 1/12.9199, 1/2.5826, 1/3.2276). \end{aligned} \tag{9.28}$$

The region of interest is $X = [18.75, 21.25]^2 \times [29.5, 36.5]^2$ and the input space $U = \{17, 18, 19, 20\}$. The safe region is $X_s = X$. The goal is to maximize the pessimistic probability of safety in 10 time steps, or by duality, minimize the optimistic reachability probability to the complement of the safe set X^c. Then the complement probability of that result is maximized pessimistic safety probability. We consider a partition of the region of interest of $(5, 5, 7, 7)$.

Van der Pol Oscillator A 2D non-linear case study from [39] with the following dynamics:

$$\mathbf{x}[k+1] = \begin{pmatrix} \mathbf{x}_1[k] + \mathbf{x}_2[k]h \\ \mathbf{x}_2[k] + (-\mathbf{x}_1[k] + (1 - \mathbf{x}_1[k])^2\mathbf{x}_2[k])h + \mathbf{u}[k] \end{pmatrix} + \mathbf{v}[k] \tag{9.29}$$

where $h = 0.1$ and $p_{\mathbf{v}} = \mathcal{N}(\cdot \mid 0, 0.2I)$. The region of interest is $X = [-4, 4]^2$ and the action space is $[-1, 1]$ partitioned into 11 points. The goal is to maximize the pessimistic reachability probability to a reach region $R = [-1.4, -0.7] \times [-2.9, -2.0]$ over 10 time steps. We consider a partition of the region of interest of $(50, 50)$.

Neural Network Dynamic Model As a complex benchmark with non-smooth dynamics, we consider a NNDM with dynamics $\mathbf{x}[k+1] = f_\theta(\mathbf{x}[k], \mathbf{u}[k]) + \mathbf{v}[k]$ where $f_\theta : \mathbb{R}^n \times U \to \mathbb{R}^n$ is a neural network. The specific benchmark is a surrogate model for the classic cartpole model [169] where the states are $(x, \dot{x}, \theta, \dot{\theta})$, the region of interest is $X = [-a, a]$ with $a = [2, 34.028235, 0.41887903, 34.028235]$, and the input is $U = \{0, 1\}$ (push the cart left and right, respectively). The noise is $p_{\mathbf{v}} = \mathcal{N}(\cdot \mid 0, 0.1I)$.

The model is a neural network with two hidden layers of 256 neurons and ReLU activation function (no activation function on the output). The input to the network is 6-dimensional since the two discrete actions are one-hot encoded. The model is trained for 10,000 iterations on batches of 100 samples where the samples are obtained by uniformly sampling from X. The ADAM optimization algorithm is used with a learning rate of $1\text{e}{-3}$, exponential decay with a multiplicative factor of 0.999, and the standard mean-squared error is used as the loss function. For reachability analysis of the neural network, CROWN [88], [89] is used for each region and input. We use $(20, 20, 24, 20)$.

n-D linear model To show scalability, we include a linear model with a configurable dimension n. The dynamics are $\mathbf{x}[k+1] = A\mathbf{x}[k] + \mathbf{v}[k]$ where A is a Toeplitz matrix

$$A = \underbrace{\begin{pmatrix} 0.7 & -0.1 & \cdots & 0 \\ 0 & 0.7 & \ddots & \vdots \\ \vdots & \ddots & 0.7 & -0.1 \\ -0.1 & \cdots & 0 & 0.7 \end{pmatrix}}_{n} \tag{9.30}$$

and $p_{\mathbf{v}} = \mathcal{N}(\cdot \mid 0, 0.1I)$. The safe set is $X = [-1, 1]^n$ and the goal is to stay within X for 10 time steps. We use a partitioning $\underbrace{(8, \ldots, 8)}_{n}$ of the region of interest.

Experiments are conducted with $n = 6$ and $n = 7$.

Dubin's car Gaussian Process A 3D case study from [64] where the dynamics of each mode (input), of 7, are learned as a Gaussian Process (GP) with Deep Kernel Learning. That is, a Gaussian Process $GP(\mu, k_\theta)$ where $\mu : \mathbb{R}^n \to \mathbb{R}$ is a mean function and $k_\theta : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}_{\geq 0}$ is a deep kernel, with $k_\theta(x, x') = k(g_\theta(x), g_\theta(x'))$ for a given base kernel $k : \mathbb{R}^s \times \mathbb{R}^s \to \mathbb{R}_{\geq 0}$ and a neural network $g_\theta : \mathbb{R}^n \to \mathbb{R}^s$. We refer to [64] for details on the deep kernel including the training process and on the number of samples for the posterior. Bounds on the standard deviation and covariance are obtained using the method from [164]. We remark that while [64] introduces a refinement algorithm, we use the GP bounds without refinement, as

refining the abstraction results in a non-gridded abstraction, which is incompatible with the proposed method.

The property to synthesize an optimal strategy for is reach-avoid with $O = [4, 6] \times [0, 1] \times [-0.5, 0.5]$ and $R = [8, 10] \times [0, 1] \times [-0.5, 0.5]$ for 10 time steps. We use a region of interest $[0, 10] \times [0, 2] \times [-0.5, 0.5]$ and a partitioning $(80, 16, 20)$.

Stochastic switched linear system We have designed a new benchmark for the class of stochastic switched systems [170], [171] to showcase the functionality on Gaussian mixture models. The dynamics are a mixture of two Gaussians whose mean depends on x and the switching between the modes is governed by a Bernoulli random variable Z with $\mathbb{P}(Z = 0) = 0.7$ and

$$\mathbf{x}[k+1] \sim \underbrace{\mathbb{P}(Z=0)}_{\alpha_1} p_1(\mathbf{x}[k]) + \underbrace{\mathbb{P}(Z=1)}_{\alpha_2} p_2(\mathbf{x}[k]) \tag{9.31}$$

where

$$\begin{aligned} p_1(\mathbf{x}[k]) &= \mathcal{N}\left(\underbrace{\begin{pmatrix} 0.1 & 0.9 \\ 0.8 & 0.2 \end{pmatrix} \mathbf{x}[k]}_{\mu^1(\mathbf{x}[k])}, \underbrace{\begin{pmatrix} 0.3^2 & 0 \\ 0 & 0.2^2 \end{pmatrix}}_{\Sigma^1} \right) \\ p_2(\mathbf{x}[k]) &= \mathcal{N}\left(\underbrace{\begin{pmatrix} 0.8 & 0.2 \\ 0.1 & 0.9 \end{pmatrix} \mathbf{x}[k]}_{\mu^2(\mathbf{x}[k])}, \underbrace{\begin{pmatrix} 0.2^2 & 0 \\ 0 & 0.1^2 \end{pmatrix}}_{\Sigma^2} \right). \end{aligned} \tag{9.32}$$

The specification asserted is a reach-avoid property with the reach region $R = [1, 2] \times [0, 1]$ and avoid region $O = [-1, 0] \times [-1, 1]$ over 10 time steps. We use $(40, 40)$.

9.6.2. COMPARISON WITH IMDP-BASED APPROACHES

To compare abstraction approaches based on odIMDPs and IMDPs, for each method, we report abstraction and certification time and memory usage. Furthermore, to quantify the conservatism of the results, we consider the following metrics: (i) the mean lower bound $\text{mean}(\{P_K^\pi(s) : s \in S \setminus T\})$ and (ii) the mean error, $\varepsilon = \text{mean}(\{\hat{P}_K^\pi(s) - P_K^\pi(s) : s \in S \setminus T\})$ where T is the set of terminal states and π is the optimal strategy for (9.18). P_K^π and $\hat{P}_K^\pi$ are the lower and upper bounds on the satisfaction probability returned by our approach. That is, the solution of (9.15) using the approach in (9.18). Furthermore, we report the (aggregated) difference in V between the methods as $\text{agg}(\{P_K^{\text{odIMDP}}(s) - P_K^{\text{other}}(s) : s \in S \setminus T\})$ where the aggregation functions are $\text{agg} \in \{\min, \max, \text{mean}\}$ and the methods are $\text{other} \in \{\text{IMDP}, \text{IMPaCT}\}$ (IMDP refers to the method in [40, Theorem 1]). We consider the same discretization size in the abstraction process for all methods, reported in Table 9.1.

The results of the comparison with IMDP-based approaches can be found in Table 9.2 and Table 9.3 for computation performance and satisfaction probability, respectively. Comparing the computation time of [40, Theorem 1] and **IMPaCT** [152] with that of our method based on odIMDPs, the abstraction time of odIMDPs

Table 9.2: Computation time and memory requirements for IMDP-based approaches across different benchmarks. Time is measured in seconds, and memory is measured in MB. OOM denotes out of memory and TO denotes timeout (24h). To compare the memory requirements for the benchmarks with OOM, we have calculated the amount of memory required to store the transition interval bounds (as a dense matrix) using the formula for IMDPs in Subsection 9.2.1.

	Our method			Adams, Lahijanian, and Laurenti [40]			IMPaCT [152]		
Benchmark	Abs. time	Cert. time	Mem.	Abs. time	Cert. time	Mem.	Abs. time	Cert. time	Mem.
Car parking	0.150	0.146	19.2	13.497	0.255	138.1	19.570	0.846	304.9
Robot reachability	0.665	0.168	21.6	12.659	0.129	88.0	20.611	0.765	306.7
Robot reach-avoid	14.259	7.641	547.2	1136.720	6.739	4143.8	918.856	37.526	16388.0
Building automation system	0.023	0.237	3.1	7.273	0.285	105.1	17.564	0.318	96.0
Van der Pol	1.658	0.257	45.5	27.255	0.827	353.6	113.235	3.233	1093.4
NNDM Cartpole	236.154	550.747	610.8	42326.400	3.751	1590.9	Incompatible dynamics		
6D linear model	8.959	359.887	237.3		OOM		OOM ($\approx 1.1TB$)		
7D linear model	66.231	13903.540	2310.8		TO (24h)		OOM ($\approx 70.4TB$)		
Dubin's car GP	13.816	19.562	352.2	336.940	31.333	14265.8	Incompatible dynamics		
Stochastic switched linear	0.033	0.045	4.5	3.391	0.038	41.0	NLopt failure		

Table 9.3: Satisfaction probabilities for IMDP-based approaches. V denotes the lower bound satisfaction probability and ε the mean error. For Adams, Lahijanian, and Laurenti [40] and IMPaCT [152], we also report the (minimum, maximum, and mean over regions) difference δ in V to our method, where positive means that our method yields a higher satisfaction probability and vice versa.

	Our method		Adams, Lahijanian, and Laurenti [40]					IMPaCT [152]				
Benchmark	Mean P	ε	Mean P	ε	Min δ	Max δ	Mean δ	Mean P	ε	Min δ	Max δ	Mean δ
Car parking	0.269	0.3885	0.213	0.5315	0.0040	0.1428	0.0560	0.213	0.5183	0.0040	0.1422	0.0556
Robot reachability	0.889	0.1108	0.881	0.1186	0.0060	0.0119	0.0079	0.890	0.1098	-0.0058	0.0022	-0.0010
Robot reach-avoid	0.980	0.0199	0.979	0.0208	0.0005	0.0027	0.0008	0.980	0.0202	-0.0001	0.0018	0.0003
Building automation system	0.263	0.7336	0.090	0.9076	0.0510	0.2297	0.1733	0.174	0.8237	0.0304	0.1131	0.0897
Van der Pol	0.069	0.3367	0.051	0.4178	0.0000	0.0529	0.0177	0.061	0.2431	-0.0080	0.0305	0.0084
NNDM Cartpole	0.004	0.7634	0.000	0.7184	0.0000	0.4101	0.0037	Incompatible dynamics				
6D linear model	0.958	0.0419			OOM					OOM		
7D linear model	0.952	0.0483			TO					OOM		
Dubin's car GP	0.362	0.3461	0.216	0.5046	0.0000	0.8383	0.1458	Incompatible dynamics				
Stochastic switched linear	0.411	0.2828	0.366	0.3605	0.0000	0.0979	0.0456	NLopt failure				

is at least 5x faster, up to 80x faster, for 2D benchmarks, and at least two orders of magnitude faster for >3D benchmarks. The certification time is comparable to or faster than both [40, Theorem 1] and IMPaCT. As expected, the memory requirements are considerably lower for odIMDPs: Compared to IMPaCT, odIMDPs use at least an order of magnitude less memory. Most remarkably, for the 6D and 7D linear models, odIMDPs uses 4682x and 30476x less memory, respectively, compared to IMPaCT. If we compare the satisfaction probability results in Table 9.3, the proposed method is at least as good as the method [40, Theorem 1] for all states in all examples, as proven in Section 9.4. This ranges from almost the same satisfaction probability for the 2D robot with a reach-avoid specification to 17.33 percentage points better on average for the 4D building automation system, which corresponds to our expectation that the tightness of satisfaction probabilities relative to IMDPs improve with higher dimensions as IMDPs lack the structural information. A similar analysis holds for IMPaCT, with the caveat that IMPaCT uses non-linear optimization to construct an IMDP, thus its results are generally tighter compared to those of [40, Theorem 1].

Table 9.4: The memory usage and the mean error ε at 100 time steps for **SySCoRe** [39] and odIMDPs on the car parking benchmark (see Table 9.1).

# regions	**SySCoRe**		odIMDPs	
	Mem. (MB)	ε	Mem. (MB)	ε
2500	0.80	0.2650	35.55	0.0204
10000	3.20	0.1568	279.51	0.0107
22500	7.20	0.1254	937.86	0.0070
90000	28.81	0.0288	7459.02	0.0028
360000	115.21	0.0060	59499.08	0.0014

9.6.3. COMPARISON WITH SySCoRe (MDP-BASED APPROACH)

We now also compare with **SySCoRe**[2] [39], which represents a state-of-the-art tool for control and verification of stochastic systems based on abstractions to MDPs. Note that, as **SySCoRe** is only tractable for gridding of 2D systems (after model order reductions), we cannot compare on systems of higher dimension. In particular, in Figure 9.6, we consider the car parking benchmark introduced in Table 9.1, and for various sizes of the abstraction, we report the error ε, as introduced in the previous Subsection 9.6.2. From the figure, we have three key observations:

- **SySCoRe** requires a relatively large number of regions in the partition before its mean error decreases with larger time horizons.
- odIMDPs tends to obtain substantially tighter bounds for the same abstration size. For example, the mean error at time 100 of **SySCoRe** with an MDP of size $90k$ is larger than that of our approach with an odIMDPs of $2.5k$ states.
- The mean error for odIMDPs first increases with time horizons to a level slightly above **SySCoRe** with the same number of regions and then decreases for longer horizons.

We should, however, stress that **SySCoRe** requires significantly less memory compared to our approach for the same partition size. The memory usage compared to the mean error at convergence for both **SySCoRe** and odIMDPs is reported in Table 9.4. Nevertheless, especially for longer time horizon, because of the fewer regions per dimension required by our approach to achieve a similar level of error, we expect our approach based on odIMDPs to be substantially more scalable for higher dimensions. In fact, this has already been observed in [16] for IMDP-based approaches compared to MDP-based approaches and, as illustrated in Table 9.2, our approach substantially outperforms that of [16] in terms of scalability.

[2] We only compare with the finite-state abstraction of **SySCoRe**, i.e. without model order reductions, to assess the impact of the model choice. Model order reductions are also interesting in the context of IMDP-based abstractions and is subject to future work.

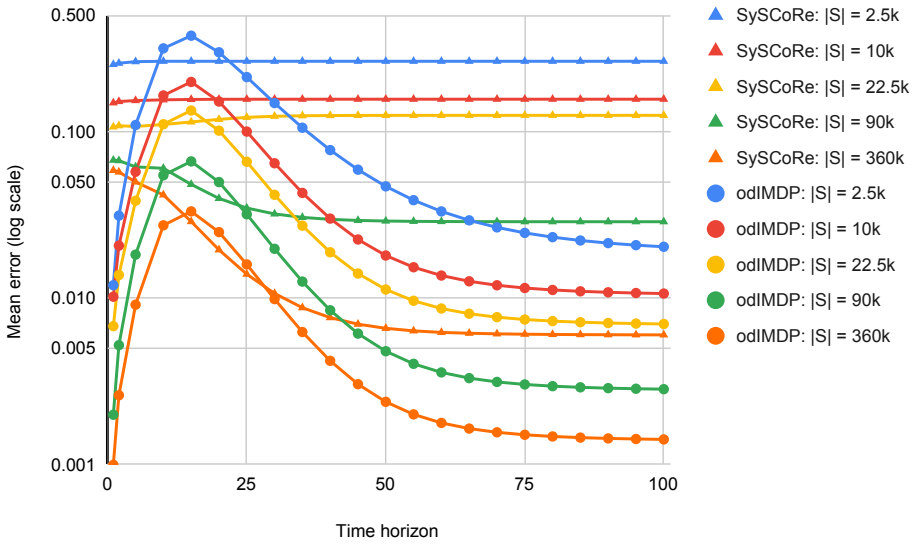


Figure 9.6: Comparing ε for a varying number of regions on the abstraction (assuming a uniform partition of the state space) and time horizons for both SySCoRe [39] and odIMDPs.

9.6.4. EMPIRICAL CONVERGENCE ANALYSIS

As indicated in Figure 9.6, the error of our approach is reduced with a finer partitioning of the region of interest X. To analyze this behaviour in more detail, in Figure 9.7 we report the mean error and the 95% confidence interval as a function of the number of regions per axis for the car parking and building automation system benchmarks. In both cases, the mean error decreases with more regions for both systems. The slight increase at various points for the car parking benchmark is attributed to a misalignment of the reach and avoid regions with the partitioning of X (see Section 9.5 for more details). The confidence interval for the car parking benchmark suggests that while a larger number of regions quickly reach low errors, some hardly converge. Future work will study the convergence from a theoretical perspective.

9.7. CONCLUSION

This chapter introduces a subclass of factored Interval Markov Decision Process (fIMDP) with the goal of reducing memory requirements for finite-state abstractions by including structural information of the concrete system. In particular, the system dynamics $\mathbf{x}[k+1] = f(\mathbf{x}[k], u[k], \mathbf{v}[k])$ often exhibits both independence between the elements of the vector-valued random variable $\mathbf{v}[k]$ and a sparse dependency structure, which is valuable structural information to include in the abstraction. Due to their simplistic structure, it is impossible to encode this information in an IMDP; instead, we propose fIMDPs, which decomposes states into state variables – this

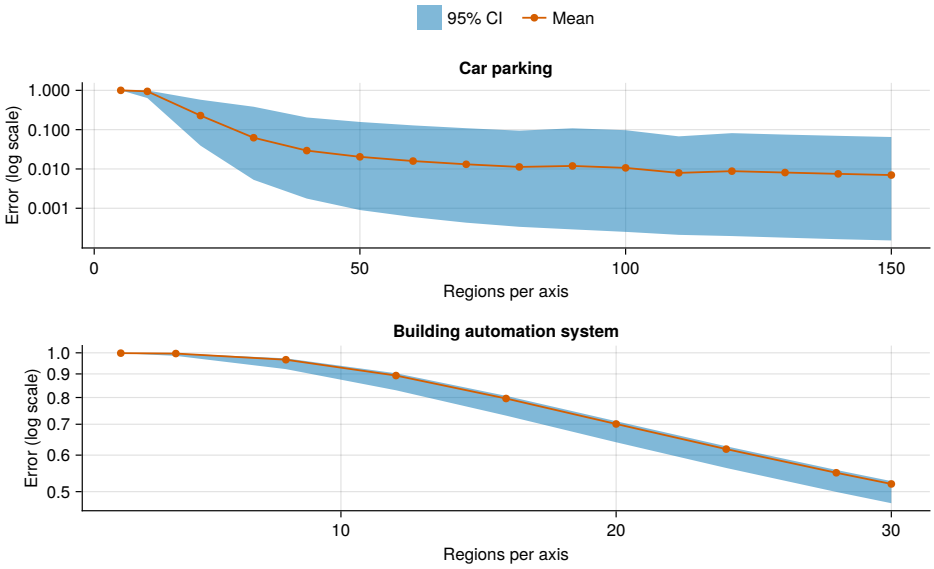


Figure 9.7: Mean error and 95% confidence interval, with respect to a uniform distribution of initial conditions, for various sizes of partitions for the car parking and building automation system benchmarks.

helps encoding the independence – and includes a dependency graph, which is useful to encode the concrete dependency structure. The consequence of including structural information is that the space complexity reduces from $O(|A||S|^2)$ to $O(|A||S|n\sqrt[n]{|S|})$; an exponential reduction. Thus, structural information enables the synthesis of controllers for higher-dimensional systems that would be intractable under traditional methods.

An issue with fIMDPs is that the robust expectation, within the Bellman operator, is no LP problem, thus, we cannot rely on LP solvers nor O-maximization. Instead, we propose a relaxed, but sound variation of the Bellman operator. Interestingly, we show that despite this relaxation, the safety guarantee obtained with fIMDPs is *at least* as tight as with IMDP-based abstractions. To ensure the soundness of the abstraction, we prove that there exists a simulation relation between the concrete and abstract system, and that because such a relation exists, the safety probability of any abstract state is a lower bound for related concrete states.

To highlight the importance and impact of using structural information, we compare against two IMDP-based abstraction approaches on a wide variety of benchmarks including linear, non-linear, neural network, and Gaussian process dynamics. In addition, we compare with an abstraction approach based on MDPs, which shows that the proposed approach can obtain a similar level of mean error with a significantly coarser partition.

A key insight of this work is that exploiting structural information is crucial to enable scalable verification and control synthesis for discrete-time stochastic sys-

tems.

Despite the improvements, several limitations remain:

- Independence assumption: The framework assumes stochastic independence across dimensions. Systems with correlated disturbances may require alternative abstraction techniques.
- Computational complexity in very high dimensions: While fIMDPs reduce memory compared to standard IMDPs, extremely large or high dimensional state and/or action spaces remain beyond reach due to computational rather than space constraints.
- Known dynamics: This chapter, relative to Chapter 6, reverted to the scenario of known dynamics, which we know is restrictive. We include one example that relaxes this assumption, the Gaussian process dynamics of Dubin's car, but the plethora of methods for data-driven abstractions will require adaptation, including discovering sparsity patterns, to be applicable to this new abstract model class.

III

Discussion and Conclusion

10

Discussion and Conclusion

10.1. Summary of Contributions

This dissertation addresses the fundamental challenge of safety verification for stochastic discrete-time systems in cyber-physical and autonomous applications where formal guarantees are critical. The work presents both theoretical advances and practical computational tools designed to provide rigorous safety verification while maintaining scalability for real-world systems. The core contribution is built on two complementary approaches: *Stochastic Barrier Functions* (Chapters 4 – 6) and *finite-state abstractions* (Chapters 8 – 9).

- **Stochastic barrier functions**
  We developed a suite of synthesis methods for stochastic barrier functions with varying computational trade-offs and applicability to different system classes. Chapter 4 presents Neural Barrier Functions (NBFs), which are neural networks trained to satisfy the barrier conditions and verified post-hoc using state-of-the-art neural network verification techniques. This approach improves expressiveness and scalability compared to existing Sum-of-Squares Programming (SoSP)-based methods. To further elucidate the theoretical foundations of SBFs and provide alternative, scalable synthesis methods, Chapter 5 introduces piecewise template-based barriers and the specialization of PWC barriers. The latter enables the use of LP and gradient descent for barrier synthesis, and the gradient descent method hints at key connections between PWC-SBFs and IMDP-based finite-state abstractions as both rely on O-maximization for efficiency. In a different context, namely of systems with unknown or partially known dynamics, Chapter 6 develops a scenario approach-based SBF synthesis that uses sampled system trajectories to construct probabilistic guarantees.

- **Finite-state models and abstractions**
  Building on Interval Markov Decision Processes (IMDPs), we introduced signif-

icant algorithmic improvements (Chapter 8) that substantially enhance scalability for safety verification, with experimental validation showing orders-of-magnitude speedups on benchmark problems. We further introduce *factored Interval Markov Decision Processes (fIMDPs)* in Chapter 9, a novel abstraction model that unconditionally reduces discretization error and memory requirements without compromising computational efficiency. We develop numeric-symbolic algorithms for both fIMDP construction and safety property verification, and demonstrate experimentally that fIMDPs can handle considerably larger systems than traditional IMDPs while maintaining comparable safety guarantees.

The connection between the two families was, following the work in Chapter 5, further explored in [45] with the proverbial torch of dynamic programming to unify them and reveal their respective benefits and disadvantages. A key insight is that both methods introduce approximations to the optimal value function characterizing safety probabilities, with the conservatism in SBFs arising from generalizing one-step increases in barrier value along system trajectory, and the conservatism in finite-state abstractions stemming from state-space discretization and over-approximation of transition probabilities. Crucially, the safety guarantees using SBFs can be reinterpreted as the safety of an auxiliary system that is reset to the worst-case state at every time step, and consequently, it is ill-suited for capturing long-term dependencies and safety *verification*. It is important to stress that SBFs remain a valuable tool for *synthesis of (continuous) safe controllers* where the SBF acts as safety field and short-term guarantees are often sufficient. In contrast, finite-state abstractions inherently account for multi-step dynamics through the transition structure of the abstract model. This perspective clarifies the strengths and weaknesses of each approach: SBFs offer computational efficiency and continuous-state analysis but may struggle with long-horizon properties, while finite-state abstractions provide more accurate long-term safety estimates at the cost of increased computational complexity. Furthermore, finite-state abstractions are guaranteed to converge in the limit to true optimal probability, while SBFs provably will not.

10.2. Limitations

While this dissertation advances the state-of-the-art in stochastic safety verification, acknowledging key limitations in both algorithmic design and testing methodology are essential for contextualization.

- **System assumptions and generality:** The developed methods primarily target discrete-time stochastic systems with specific structural properties, e.g., Markovian, continuous, and (at least partially) known dynamics. Extensions to broader classes of systems, such as hybrid systems or those with non-Markovian dynamics, remain open challenges. A particular limitation is that the discrete-time assumption, which underpins both the SBF and finite-state abstraction frameworks presented here. A consequence is that the dynamics

models only capture system evolution at discrete time steps, i.e., jumps, and can thus "jump" over/through obstacles between time steps such as the Car Parking benchmark in Chapter 9. Another key assumption is that the entire state is observable, which may not hold in many practical scenarios where partial observability is prevalent. Addressing these limitations would enhance the applicability of the methods to a wider range of real-world systems.

- **Verification vs. synthesis:** The focus of this dissertation is on safety verification rather than controller synthesis. Although finite-state abstraction methods are developed explicitly for control synthesis, the SBF-based methods presented have been mainly treated as verification tools. Recent works have explored SBFs for control synthesis [77], [99], [128], [172]; however, the developments remain preliminary from theoretical, computational, and experimental perspectives. Furthermore, the impact of a *safety controller* in a closed-loop system remains unexplored in this dissertation.

- **Computational complexity and conservatism trade-off:** Despite significant algorithmic improvements, the methods still face the curse of dimensionality; a pervasive challenge when applying formal methods to high-dimensional systems. Importantly, while adaptive partitioning schemes, such as the one presented in Chapter 4, mitigate the curse of dimensionality to some degree, they introduce an alternative computational bottleneck: the curse of partitioning, which places fundamental practical limits on scalability. In this context of sound but approximate quantitative verification, the consequence is a trade-off between conservatism and computational requirements; with additional computational power, it is possible to reduce conservatism. However, improved algorithms, such as presented in this dissertation, may push the Pareto curve.

- **Empirical validation:** The case studies conducted for this dissertation are limited to simulation-based experiments and idealized benchmarks. Broader validation on an extensive range of real-world systems would strengthen confidence in practical applicability and motivate further theoretical and computational developments.

10.3. FUTURE DIRECTIONS

The results presented in this dissertation open several promising avenues for future research.

- **Scalability enhancement:** Scalability is a key concern and challenge in most formal methods, in terms of not only state space or dimensionality, but also system complexity, required precision, and sample complexity. Developing more sophisticated algorithms that e.g., integrate symbolic-numeric methods including neuro-symbolic methods, which utilize all available information about the systems under study, could further improve scalability for high-dimensional problems. Newer methods from statistical learning theory

and machine learning may also provide avenues for reducing sample complexity in data-driven verification contexts.

- **Relaxing system assumptions:** Future work should explore methods that relax key assumptions, such as extending verification frameworks to handle non-Markovian dynamics and non-stationary disturbances. Such extensions would substantially broaden the applicability of the developed techniques to more diverse real-world scenarios including human-in-the-loop systems and environments with evolving uncertainties.

- **Discrete-time controllers of continuous-time systems:** A natural extension is to adapt the developed verification frameworks to continuous-time stochastic systems, and these have indeed been studied [30], [119], [173], [174]. However, as discussed in Chapter 1, most contemporary controllers are digital and thus operate in discrete time. To combine continuous-time stochastic systems and discrete-time controllers involves addressing challenges related to inter-sample behaviour and ensuring that safety guarantees hold continuously between discrete control updates.

- **Verification across operating conditions:** A fundamental open question remains: how to integrate verification results from individual scenarios into coherent assurances about system safety across a distribution of operating conditions. One promising direction is combining the quantitative verification techniques developed here with extreme value theory [175] to estimate probabilities of rare but catastrophic events from limited scenario analyses.

10.4. Broader Impact and Significance

Safety verification is indispensable for deploying autonomous and cyber-physical systems in real-world applications where failures carry severe consequences. Formal verification techniques provide rigorous safety guarantees that are essential for regulatory approval, certification processes and building public trust in autonomous systems. A key limitation of formal verification is, however, scalability, limiting their applicability in practice. The theoretical and computational contributions of this dissertation address this critical challenge through multiple complementary angles. By making these techniques more scalable, we expand the set of systems that can be formally analysed.

The significant improvements in computational efficiency, demonstrated through significant speed-ups and handling systems with larger dimensions and complexity, broaden the practical utility of formal methods for discrete-time stochastic systems. This has implications for applications in autonomous vehicles, robotic systems, medical devices, and industrial control, where both safety and performance are critical.

Beyond immediate applications, this work deepens our theoretical understanding of stochastic systems and the trade-offs inherent in verification approaches. The connections established between barrier functions and dynamic programming, the characterization of abstraction errors, and the introduction of factored abstractions

all contribute to a theoretical foundation for the field. These insights will serve as building blocks for future methodological advances.

Bibliography

[1] P. Derler, E. A. Lee, and A. S. Vincentelli, "Modeling cyber-physical systems", *Proceedings of the IEEE*, vol. 100, no. 1, pp. 13–28, 2011. **DOI**: 10.1109/JPROC.2011.2160929.

[2] M. Střelec, K. Macek, and A. Abate, "Modeling and simulation of a microgrid as a stochastic hybrid system", in *IEEE PES Innovative Smart Grid Technologies Europe*, 2012, pp. 1–9. **DOI**: 10.1109/ISGTEurope.2012.6465655.

[3] H. van Dam, "Physics of nuclear reactor safety", *Reports on Progress in Physics*, vol. 55, no. 11, p. 2025, 1992. **DOI**: 10.1088/0034-4885/55/11/003.

[4] N. G. Leveson and C. S. Turner, "An investigation of the Therac-25 accidents", *Computer*, vol. 26, no. 7, pp. 18–41, 1993. **DOI**: 10.1109/MC.1993.274940.

[5] J. Guiochet, M. Machin, and H. Waeselynck, "Safety-critical advanced robots: A survey", *Robotics and Autonomous Systems*, vol. 94, pp. 43–52, 2017. **DOI**: 10.1016/j.robot.2017.04.004.

[6] Y. C. Yeh, "Safety critical avionics for the 777 primary flight controls system", in *20th Digital Avionics Systems Conference*, vol. 1, IEEE, 2001, pp. 1C2/1–1C2/11. **DOI**: 10.1109/DASC.2001.963311.

[7] J. A. Wise, V. D. Hopkin, and M. L. Smith, Eds., *Automation and Systems Issues in Air Traffic Control*. Springer Berlin Heidelberg, 1991. **DOI**: 10.1007/978-3-642-76556-8.

[8] K. Kanso, F. Moller, and A. Setzer, "Automated verification of signalling principles in railway interlocking systems", *Electronic Notes in Theoretical Computer Science*, vol. 250, no. 2, pp. 19–31, 2009. **DOI**: 10.1016/j.entcs.2009.08.015.

[9] S. Shalev-Shwartz, S. Shammah, and A. Shashua, *On a formal model of safe and scalable self-driving cars*, 2018. arXiv: 1708.06374 [cs.RO].

[10] S. Lou, Z. Hu, Y. Zhang, Y. Feng, M. Zhou, and C. Lv, "Human-cyber-physical system for industry 5.0: A review from a human-centric perspective", *IEEE Transactions on Automation Science and Engineering*, vol. 22, pp. 494–511, 2024. **DOI**: 10.1109/TASE.2024.3360476.

[11] J. C. Knight, "Safety critical systems: Challenges and directions", in *Proceedings of the 24th International Conference on Software Engineering*, 2002, pp. 547–550. **DOI**: 10.1145/581339.581406.

[12] N. G. Leveson, *Engineering a Safer World: Systems Thinking Applied to Safety*. The MIT Press, 2012, **ISBN**: 9780262298247. **DOI**: 10.7551/mitpress/8179.001.0001.

[13] C. Baier and J.-P. Katoen, *Principles of Model Checking*. MIT press, 2008, **ISBN**: 9780262026499.

[14] C. Fan, "Formal methods for safe autonomy: Data-driven verification, synthesis, and applications", Ph.D. dissertation, University of Illinois at Urbana-Champaign, 2019.

[15] C. Santoyo, M. Dutreix, and S. Coogan, "A barrier function approach to finite-time stochastic system verification and control", *Automatica*, vol. 125, p. 109 439, 2021. **DOI**: 10.1016/j.automatica.2020.109439.

[16] N. Cauchi, L. Laurenti, M. Lahijanian, A. Abate, M. Kwiatkowska, and L. Cardelli, "Efficiency through uncertainty: Scalable formal synthesis for stochastic hybrid systems", in *Proceedings of the 22nd international conference on Hybrid Systems: Computation and Control*, 2019, pp. 240–251. **DOI**: 10.1145/3302504.3311805.

[17] H. J. Kushner, "Stochastic stability and control", Brown Univ Providence RI, Tech. Rep., 1967.

[18] S. Prajna, A. Jadbabaie, and G. J. Pappas, "A framework for worst-case and stochastic safety verification using barrier certificates", *IEEE Transactions on Automatic Control*, vol. 52, no. 8, pp. 1415–1428, 2007. **DOI**: 10.1109/TAC.2007.902736.

[19] M. Kwiatkowska, G. Norman, and D. Parker, "PRISM 4.0: Verification of probabilistic real-time systems", in *International conference on computer aided verification*, Springer, 2011, pp. 585–591. **DOI**: 10.1007/978-3-642-22110-1_47.

[20] C. Hensel, S. Junges, J.-P. Katoen, T. Quatmann, and M. Volk, "The probabilistic model checker storm", *International Journal on Software Tools for Technology Transfer*, vol. 24, no. 4, pp. 589–610, 2022. **DOI**: 10.1007/s10009-021-00633-z.

[21] R. Alur, *Principles of Cyber-physical Systems*. MIT press, 2015, **ISBN**: 9780262548922.

[22] G. De Giacomo and M. Y. Vardi, "Linear temporal logic and linear dynamic logic on finite traces", in *IJCAI*, vol. 13, 2013, pp. 854–860.

[23] B. Kohlen, M. Schäffeler, M. Abdulaziz, A. Hartmanns, and P. Lammich, "A formally verified IEEE 754 floating-point implementation of interval iteration for MDPs", in *International Conference on Computer Aided Verification*, Springer, 2025, pp. 122–146. **DOI**: 10.1007/978-3-031-98679-6_6.

[24] C. Watterson and D. Heffernan, "Runtime verification and monitoring of embedded systems", *IET software*, vol. 1, no. 5, pp. 172–179, 2007. **DOI**: 10.1049/iet-sen:20060076.

[25] A. D. Ames, S. Coogan, M. Egerstedt, G. Notomista, K. Sreenath, and P. Tabuada, "Control barrier functions: Theory and applications", in *18th European Control Conference (ECC)*, 2019, pp. 3420–3431. **DOI**: 10.23919/ECC.2019.8796030.

[26] C. Le Guernic and A. Girard, "Reachability analysis of hybrid systems using support functions", in *International Conference on Computer Aided Verification*, Springer, 2009, pp. 540–554. **DOI**: 10.1007/978-3-642-02658-4_40.

[27] A. Valmari, "The state explosion problem", in *Lectures on Petri Nets I: Basic Models: Advances in Petri Nets*. Springer Berlin Heidelberg, 1998, pp. 429–528. **DOI**: 10.1007/3-540-65306-6_21.

[28] P. Jagtap, S. Soudjani, and M. Zamani, "Temporal logic verification of stochastic systems using barrier certificates", in *International Symposium on Automated Technology for Verification and Analysis*, Springer, 2018, pp. 177–193. **DOI**: 10.1007/978-3-030-01090-4_11.

[29] P. A. Parrilo, "Semidefinite programming relaxations for semialgebraic problems", *Mathematical programming*, vol. 96, no. 2, pp. 293–320, 2003. **DOI**: 10.1007/s10107-003-0387-5.

[30] A. Nejati, S. Soudjani, and M. Zamani, "Compositional construction of control barrier certificates for large-scale stochastic switched systems", *IEEE Control Systems Letters*, vol. 4, no. 4, pp. 845–850, 2020. **DOI**: 10.1109/LCSYS.2020.2994039.

[31] M. Anand, A. Lavaei, and M. Zamani, "From small-gain theory to compositional construction of barrier certificates for large-scale stochastic systems", *IEEE Transactions on Automatic Control*, vol. 67, no. 10, pp. 5638–5645, 2022. **DOI**: 10.1109/TAC.2022.3183032.

[32] A. Salamati and M. Zamani, "Safety verification of stochastic systems: A repetitive scenario approach", *IEEE Control Systems Letters*, vol. 7, pp. 448–453, 2022. **DOI**: 10.1109/LCSYS.2022.3186932.

[33] A. Salamati, A. Lavaei, S. Soudjani, and M. Zamani, "Data-driven verification and synthesis of stochastic systems via barrier certificates", *Automatica*, vol. 159, 2024. **DOI**: 10.1016/j.automatica.2023.111323.

[34] M. Anand, V. Murali, A. Trivedi, and M. Zamani, "k-inductive barrier certificates for stochastic systems", in *Proceedings of the 25th International Conference on Hybrid Systems: Computation and Control*, 2022, pp. 1–11. **DOI**: 10.1145/3501710.3519532.

[35] M. A. Oumer, V. Murali, and M. Zamani, *k-inductive and interpolation-inspired barrier certificates for stochastic dynamical systems*, 2025. arXiv: 2504.15412 [math.OC].

[36] R. K. Cosner, P. Culbertson, A. J. Taylor, and A. D. Ames, "Robust safety under stochastic uncertainty with discrete-time control barrier functions", in *Robotics: Science and systems*, 2023.

[37] R. K. Cosner, P. Culbertson, and A. D. Ames, "Bounding stochastic safety: Leveraging freedman's inequality with discrete-time control barrier functions", *IEEE Control Systems Letters*, vol. 8, pp. 1937–1942, 2024. **DOI**: 10.1109/LCSYS.2024.3409105.

[38] B. van Huijgevoort and S. Haesaert, "Temporal logic control of nonlinear stochastic systems using a piecewise-affine abstraction", *IEEE Control Systems Letters*, vol. 7, pp. 1039–1044, 2022. **DOI**: 10.1109/LCSYS.2022.3230765.

[39] B. Van Huijgevoort, O. Schön, S. Soudjani, and S. Haesaert, "Syscore: Synthesis via stochastic coupling relations", in *Proceedings of the 26th International Conference on Hybrid Systems: Computation and Control*, 2023. **DOI**: 10.1145/3575870.3587123.

[40] S. Adams, M. Lahijanian, and L. Laurenti, "Formal control synthesis for stochastic neural network dynamic models", *IEEE Control Systems Letters*, vol. 6, pp. 2858–2863, 2022. **DOI**: 10.1109/LCSYS.2022.3178143.

[41] I. Gracia, D. Boskos, L. Laurenti, and M. Lahijanian, "Data-driven strategy synthesis for stochastic systems with unknown nonlinear disturbances", in *6th Annual Learning for Dynamics & Control Conference*, ser. Proceedings of Machine Learning Research, vol. 242, PMLR, 2024, pp. 1633–1645.

[42] I. Gracia, L. Laurenti, M. Mazo Jr, A. Abate, and M. Lahijanian, "Temporal logic control for nonlinear stochastic systems under unknown disturbances", in *7th Annual Learning for Dynamics & Control Conference*, PMLR, 2025, pp. 1537–1549.

[43] A. Abate, A. D'Innocenzo, and M. D. Di Benedetto, "Approximate abstractions of stochastic hybrid systems", *IEEE Transactions on Automatic Control*, vol. 56, no. 11, pp. 2688–2694, 2011. **DOI**: 10.1109/TAC.2011.2160595.

[44] S. E. Z. Soudjani and A. Abate, "Adaptive gridding for abstraction and verification of stochastic hybrid systems", in *8th International Conference on Quantitative Evaluation of SysTems*, 2011, pp. 59–68. **DOI**: 10.1109/QEST.2011.16.

[45] L. Laurenti and M. Lahijanian, "A unifying perspective for safety of stochastic systems: From barrier functions to finite abstractions", *IEEE Transactions on Automatic Control*, 2025. **DOI**: 10.1109/TAC.2025.3597569.

[46] B. C. van Huijgevoort, S. Weiland, and S. Haesaert, "Temporal logic control of nonlinear stochastic systems using a piecewise-affine abstraction", *IEEE Control Systems Letters*, 2023. **DOI**: 10.1109/LCSYS.2022.3230765.

[47] S. Esmaeil Zadeh Soudjani, A. Abate, and R. Majumdar, "Dynamic bayesian networks for formal verification of structured stochastic processes", *Acta Informatica*, vol. 54, no. 2, pp. 217–242, 2016. **DOI**: 10.1007/s00236-016-0287-9.

[48] A. Lavaei and M. Zamani, "From dissipativity theory to compositional synthesis of large-scale stochastic switched systems", *IEEE Transactions on Automatic Control*, vol. 67, no. 9, pp. 4422–4437, 2022. **DOI**: 10.1109/TAC.2022.3159190.

[49] A. Lavaei, S. Soudjani, and M. Zamani, "Compositional synthesis of finite abstractions for continuous-space stochastic control systems: A small-gain approach", *IFAC-PapersOnLine*, vol. 51, no. 16, pp. 265–270, 2018. **DOI**: 10.1016/j.ifacol.2018.08.045.

[50] A. Lavaei and M. Zamani, "Compositional verification of large-scale stochastic systems via relaxed small-gain conditions", in *58th IEEE Conference on Decision and Control (CDC)*, IEEE, 2019, pp. 2574–2579. **DOI**: 10.1109/CDC40024.2019.9029485.

[51] A. Lavaei, S. Soudjani, and M. Zamani, "Compositional synthesis of not necessarily stabilizable stochastic systems via finite abstractions", in *18th European Control Conference (ECC)*, 2019, pp. 2802–2807. **DOI**: 10.23919/ECC.2019.8795996.

[52] A. Lavaei, S. Soudjani, and M. Zamani, "Compositional abstraction-based synthesis for networks of stochastic switched systems", *Automatica*, vol. 114, p. 108 827, 2020. **DOI**: 10.1016/j.automatica.2020.108827.

[53] A. Lavaei, S. Soudjani, and M. Zamani, "Compositional (in)finite abstractions for large-scale interconnected stochastic systems", *IEEE Transactions on Automatic Control*, vol. 65, no. 12, pp. 5280–5295, 2020. **DOI**: 10.1109/TAC.2020.2975812.

[54] A. Lavaei, S. Soudjani, and M. Zamani, "From dissipativity theory to compositional construction of finite markov decision processes", in *Proceedings of the 21st International Conference on Hybrid Systems: Computation and Control*, 2018, pp. 21–30. **DOI**: 10.1145/3178126.3178135.

[55] A. Lavaei, S. Soudjani, and M. Zamani, "Compositional construction of infinite abstractions for networks of stochastic control systems", *Automatica*, vol. 107, pp. 125–137, 2019. **DOI**: 10.1016/j.automatica.2019.05.043.

[56] A. Lavaei, S. Soudjani, and M. Zamani, "Compositional abstraction of large-scale stochastic systems: A relaxed dissipativity approach", *Nonlinear Analysis: Hybrid Systems*, vol. 36, p. 100 880, 2020. **DOI**: 10.1016/j.nahs.2020.100880.

[57] I. Gracia, D. Boskos, L. Laurenti, and M. Mazo Jr, "Distributionally robust strategy synthesis for switched stochastic systems", in *Proceedings of the 26th International Conference on Hybrid Systems: Computation and Control*, 2023, pp. 1–10. **DOI**: 10.1145/3575870.358712.

[58] I. Gracia, D. Boskos, M. Lahijanian, L. Laurenti, and M. Mazo Jr, "Efficient strategy synthesis for switched stochastic systems with distributional uncertainty", *Nonlinear Analysis: Hybrid Systems*, vol. 55, 2025. **DOI**: 10.1016/j.nahs.2024.101554.

[59] M. Nazeri, T. Badings, S. Soudjani, and A. Abate, *Data-driven yet formal policy synthesis for stochastic nonlinear dynamical systems*, 2025. arXiv: 2501.01191 [eess.SY].

[60] T. S. Badings, A. Abate, N. Jansen, D. Parker, H. A. Poonawala, and M. Stoelinga, "Sampling-based robust control of autonomous systems with non-gaussian noise", in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, 2022, pp. 9669–9678.

[61] T. Badings, L. Romao, A. Abate, *et al.*, "Robust control for dynamical systems with non-gaussian noise via formal abstractions", *Journal of Artificial Intelligence Research*, vol. 76, pp. 341–391, 2023. **DOI**: 10.1613/jair.1.14253.

[62] R. Coppola, A. Peruffo, L. Romao, A. Abate, and M. Mazo, "Enhancing data-driven stochastic control via bundled interval MDP", *IEEE Control Systems Letters*, vol. 8, pp. 2069–2074, 2024. **DOI**: 10.1109/LCSYS.2024.3417852.

[63] J. Skovbekk, L. Laurenti, E. Frew, and M. Lahijanian, "Formal abstraction of general stochastic systems via noise partitioning", *IEEE Control Systems Letters*, vol. 7, pp. 3711–3716, 2023. **DOI**: 10.1109/LCSYS.2023.3340621.

[64] R. Reed, L. Laurenti, and M. Lahijanian, "Promises of deep kernel learning for control synthesis", *IEEE Control Systems Letters*, vol. 7, pp. 3986–3991, 2023. **DOI**: 10.1109/LCSYS.2023.3340995.

[65] O. Schön, S. Naseer, B. Wooding, and S. Soudjani, "Data-driven abstractions via binary-tree gaussian processes for formal verification", *IFAC-PapersOnLine*, vol. 58, no. 11, pp. 115–122, 2024. **DOI**: 10.1016/j.ifacol.2024.07.434.

[66] J. Skovbekk, L. Laurenti, E. Frew, and M. Lahijanian, "Formal verification of unknown dynamical systems via gaussian process regression", *IEEE Transactions on Automatic Control*, 2025. **DOI**: 10.1109/TAC.2025.3532812.

[67] O. Schön, B. van Huijgevoort, S. Haesaert, and S. Soudjani, "Bayesian formal synthesis of unknown systems via robust simulation relations", *IEEE Transactions on Automatic Control*, 2024. **DOI**: 10.1109/TAC.2024.3459308.

[68] A. Abate, M. Prandini, J. Lygeros, and S. Sastry, "Probabilistic reachability and safety for controlled discrete time stochastic hybrid systems", *Automatica*, vol. 44, no. 11, pp. 2724–2734, 2008. **DOI**: 10.1016/j.automatica.2008.03.027.

[69] A. Abate, S. Amin, M. Prandini, J. Lygeros, and S. Sastry, "Computational approaches to reachability analysis of stochastic hybrid systems", in *Proceedings of the 10th International Conference on Hybrid Systems: Computation and Control*, Springer Berlin Heidelberg, 2007, pp. 4–17. **DOI**: 10.1007/978-3-540-71493-4_4.

[70] S. Summers and J. Lygeros, "Verification of discrete time stochastic hybrid systems: A stochastic reach-avoid decision problem", *Automatica*, vol. 46, no. 12, pp. 1951–1961, 2010. **DOI**: 10.1016/j.automatica.2010.08.006.

[71] A. P. Vinod and M. M. K. Oishi, "Stochastic reachability of a target tube: Theory and computation", *Automatica*, vol. 125, p. 109 458, 2021. **DOI**: 10.1016/j.automatica.2020.109458.

[72] K. Lesser, M. Oishi, and R. S. Erwin, "Stochastic reachability for control of spacecraft relative motion", in *52nd IEEE Conference on Decision and Control (CDC)*, IEEE, 2013, pp. 4705–4712. **DOI**: 10.1109/CDC.2013.6760626.

[73] A. J. Thorpe, K. R. Ortiz, and M. M. K. Oishi, "Sreachtools kernel module: Data-driven stochastic reachability using hilbert space embeddings of distributions", in *60th IEEE Conference on Decision and Control (CDC)*, IEEE Press, 2021, pp. 5073–5079. **DOI**: 10.1109/CDC45484.2021.9683169.

[74] J. D. Gleason, A. P. Vinod, and M. M. K. Oishi, "Lagrangian approximations for stochastic reachability of a target tube", *Automatica*, vol. 128, p. 109 546, 2021. **DOI**: 10.1016/j.automatica.2021.109546.

[75] F. B. Mathiesen, S. C. Calvert, and L. Laurenti, "Safety certification for stochastic systems via neural barrier functions", *IEEE Control Systems Letters*, 2023. **DOI**: 10.1109/LCSYS.2022.3229865.

[76] R. Mazouz, F. B. Mathiesen, L. Laurenti, and M. Lahijanian, *Piecewise stochastic barrier functions*, Accepted for Automatica, 2024. arXiv: 2404.16986 [cs.RO].

[77] R. Mazouz, J. Skovbekk, F. B. Mathiesen, E. Frew, L. Laurenti, and M. Lahijanian, "Data-driven permissible safe control with barrier certificates", in *IEEE 63rd Conference on Decision and Control (CDC)*, IEEE, 2024, pp. 6844–6849. **DOI**: 10.1109/CDC56724.2024.10886850.

[78] F. B. Mathiesen, L. Romao, S. C. Calvert, A. Abate, and L. Laurenti, "Inner approximations of stochastic programs for data-driven stochastic barrier function design", in *62nd IEEE Conference on Decision and Control (CDC)*, IEEE, 2023, pp. 3073–3080. **DOI**: 10.1109/CDC49753.2023.10383306.

[79] F. B. Mathiesen, L. Romao, S. C. Calvert, L. Laurenti, and A. Abate, *A data-driven approach for safety quantification of non-linear stochastic systems with unknown additive noise distribution*, Accepted for Automatica, 2024. arXiv: 2410.06662 [eess.SY].

[80] F. B. Mathiesen, M. Lahijanian, and L. Laurenti, "IntervalMDP.jl: Accelerated value iteration for interval markov decision processes", *IFAC-PapersOnLine*, vol. 58, no. 11, pp. 1–6, 2024, 8th IFAC Conference on Analysis and Design of Hybrid Systems. **DOI**: 10.1016/j.ifacol.2024.07.416.

[81] F. B. Mathiesen, S. Haesaert, and L. Laurenti, "Scalable control synthesis for stochastic systems via structural IMDP abstractions", in *Proceedings of the 28th International Conference on Hybrid Systems: Computation and Control*, Association for Computing Machinery, 2025. **DOI**: 10.1145/3716863.3718031.

[82] J. L. Doob, *Stochastic Processes*. John Wiley & Sons Inc, 1953, p. 654, **ISBN**: 9780471218135.

[83] J. Ville, *Étude critique de la notion de collectif*. Gauthier-Villars Paris, 1939, vol. 3.

[84] K. Fukuda and A. Prodon, "Double description method revisited", in *Combinatorics and Computer Science*, Springer Berlin Heidelberg, 1996, pp. 91–111. **DOI**: 10.1007/3-540-61576-8_77.

[85] B. Legat, R. Deits, G. Goretkin, *et al.*, *Juliapolyhedra/polyhedra.jl: V0.6.16*, version v0.6.16, 2021. **DOI**: 10.5281/zenodo.4993670.

[86] C. Belta, B. Yordanov, and E. Aydin Gol, *Formal Methods for Discrete-Time Dynamical Systems*. Springer International Publishing, 2017. **DOI**: 10.1007/978-3-319-50763-7.

[87] M. M. Joldes, "Rigorous polynomial approximations and applications", Ph.D. dissertation, Ecole Normale Supérieure de Lyon, 2011.

[88] H. Zhang, T.-W. Weng, P.-Y. Chen, C.-J. Hsieh, and L. Daniel, "Efficient neural network robustness certification with general activation functions", *Advances in Neural Information Processing Systems*, vol. 31, 2018.

[89] K. Xu, Z. Shi, H. Zhang, *et al.*, "Automatic perturbation analysis for scalable certified robustness and beyond", *Advances in Neural Information Processing Systems*, vol. 33, pp. 1129–1141, 2020.

[90] C. Brix, M. N. Müller, S. Bak, T. T. Johnson, and C. Liu, "First three years of the international verification of neural networks competition (VNN-COMP)", *International Journal on Software Tools for Technology Transfer*, vol. 25, no. 3, pp. 329–339, 2023. **DOI**: 10.1007/s10009-023-00703-4.

[91] S. Gowal, K. Dvijotham, R. Stanforth, *et al.*, *On the effectiveness of interval bound propagation for training verifiably robust models*, 2019. arXiv: 1810.12715 [cs.LG].

[92] H. Zhang, H. Chen, C. Xiao, *et al.*, "Towards stable and efficient training of verifiably robust neural networks", in *The 7th International Conference on Learning Representations*, 2019.

[93] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004.

[94] M. Campi and S. Garatti, "The exact feasibility of randomized solutions of uncertain convex programs", *SIAM Journal on Optimization*, 2008. **DOI**: 10.1137/07069821X.

[95] D. Bertsekas and S. E. Shreve, *Stochastic optimal control: the discrete-time case*. Athena Scientific, 1996, vol. 5, **ISBN**: 9781886529038.

[96] A. Abate, J.-P. Katoen, J. Lygeros, and M. Prandini, "Approximate model checking of stochastic hybrid systems", *European Journal of Control*, vol. 16, no. 6, pp. 624–641, 2010. **DOI**: 10.3166/ejc.16.624-641.

[97] R. D. McAllister and J. B. Rawlings, "Stochastic lyapunov functions and asymptotic stability in probability", Texas-Wisconsin-California Control Consortium, Tech. Rep. 2, 2020, pp. 1–26.

[98] J. Steinhardt and R. Tedrake, "Finite-time regional verification of stochastic non-linear systems", *The International Journal of Robotics Research*, 2012.

[99] R. Mazouz, K. Muvvala, A. Ratheesh, L. Laurenti, and M. Lahijanian, "Safety guarantees for neural network dynamic systems via stochastic barrier functions", in *Proceedings of the 36th International Conference on Neural Information Processing Systems*, Curran Associates Inc., 2022.

[100] P. A. Parrilo and B. Sturmfels, "Minimizing polynomial functions", in *Algorithmic and Quantitative Aspects of Real Algebraic Geometry in Mathematics and Computer Science, Proceedings of a DIMACS Workshop*, ser. DIMACS Series in Discrete Mathematics and Theoretical Computer Science, vol. 60, DIMACS/AMS, 2001, pp. 83–99. **DOI**: 10.1090/DIMACS/060/08.

[101] Z. Li, F. Liu, W. Yang, S. Peng, and J. Zhou, "A survey of convolutional neural networks: Analysis, applications, and prospects", *IEEE Transactions on Neural Networks and Learning Systems*, vol. 33, no. 12, pp. 6999–7019, 2022. **DOI**: 10.1109/TNNLS.2021.3084827.

[102] K. S. Narendra, "Neural networks for control theory and practice", *Proceedings of the IEEE*, vol. 84, no. 10, pp. 1385–1406, 2002. **DOI**: 10.1109/5.537106.

[103] P. Kidger and T. Lyons, "Universal approximation with deep narrow networks", in *Conference on learning theory*, PMLR, 2020, pp. 2306–2327.

[104] D. Elbrächter, D. Perekrestenko, P. Grohs, and H. Bölcskei, "Deep neural network approximation theory", *IEEE Transactions on Information Theory*, vol. 67, no. 5, pp. 2581–2623, 2021. **DOI**: 10.1109/TIT.2021.3062161.

[105] M. Andrychowicz, M. Denil, S. Gomez, *et al.*, "Learning to learn by gradient descent by gradient descent", *Advances in Neural Information Processing Systems*, vol. 29, 2016.

[106] S. Du, J. Lee, H. Li, L. Wang, and X. Zhai, "Gradient descent finds global minima of deep neural networks", in *International Conference on Machine Learning*, PMLR, 2019, pp. 1675–1685.

[107] T. Bai, J. Luo, J. Zhao, B. Wen, and Q. Wang, "Recent advances in adversarial training for adversarial robustness", in *Proceedings of the 30th International Joint Conference on Artificial Intelligence*, 2021, pp. 4312–4321. **DOI**: 10.24963/ijcai.2021/591.

[108] A. Solar-Lezama, L. Tancau, R. Bodik, S. Seshia, and V. Saraswat, "Combinatorial sketching for finite programs", *SIGARCH Comput. Archit. News*, vol. 34, no. 5, pp. 404–415, 2006. **DOI**: 10.1145/1168919.1168907.

[109] R. Bunel, I. Turkaslan, P. H. S. Torr, P. Kohli, and M. P. Kumar, "A unified view of piecewise linear neural network verification", in *Advances in Neural Information Processing Systems*, 2018.

[110] N. Jovanović, M. Balunović, M. Baader, and M. Vechev, "On the paradox of certified training", *Transactions on Machine Learning Research*, 2022.

[111] A. Abate, D. Ahmed, A. Edwards, M. Giacobbe, and A. Peruffo, "FOSSIL: A software tool for the formal synthesis of lyapunov functions and barrier certificates using neural networks", in *Proceedings of the 24th International Conference on Hybrid Systems: Computation and Control*, Association for Computing Machinery, 2021. **DOI**: 10.1145/3447928.3456646.

[112] M. Lechner, Đ. Žikelić, K. Chatterjee, and T. A. Henzinger, "Stability verification in stochastic control systems via neural network supermartingales", in *AAAI*, 2022.

[113] M. Iannelli and A. Pugliese, *An Introduction to Mathematical Population Dynamics* (UNITEXT). Springer International Publishing, 2014, **ISBN**: 9783319030258.

[114] H. K. Khalil, *Nonlinear Systems* (Pearson Education). Prentice Hall, 2002, **ISBN**: 9780130673893.

[115] M. Putinar, "Positive polynomials on compact semi-algebraic sets", *Indiana University Mathematics Journal*, vol. 42, no. 3, pp. 969–984, 1993, **ISSN**: 00222518, 19435258.

[116] R. E. Bellman, *Dynamic programming*, en. Princeton University Press, 1957, **ISBN**: 9780691079516.

[117] L. Dai, *Nonlinear dynamics of piecewise constant systems and implementation of piecewise constant arguments*. World Scientific Publishing, 2008.

[118] H. Li, W. Liu, C. Yang, W. Wang, T. Qie, and C. Xiang, "An optimization-based path planning approach for autonomous vehicles using the DynEFWA-artificial potential field", *IEEE Transactions on Intelligent Vehicles*, vol. 7, no. 2, pp. 263–272, 2021. **DOI**: 10.1109/TIV.2021.3123341.

[119] L. Laurenti, M. Lahijanian, A. Abate, L. Cardelli, and M. Kwiatkowska, "Formal and efficient synthesis for continuous-time linear stochastic hybrid processes", *IEEE Transactions on Automatic Control*, vol. 66, no. 1, pp. 17–32, 2020. **DOI**: 10.1109/TAC.2020.2975028.

[120] C. Heil, *Introduction to Real Analysis*. Springer International Publishing, 2019, **ISBN**: 9783030269036.

[121] A. Ben-Tal and A. Nemirovski, "Robust convex optimization", *Mathematics of operations research*, vol. 23, no. 4, pp. 769–805, 1998. **DOI**: 10.1287/moor.23.4.769.

[122] R. Givan, S. Leach, and T. Dean, "Bounded-parameter markov decision processes", *Artificial Intelligence*, vol. 122, no. 1, pp. 71–109, 2000. **DOI**: 10.1016/S0004-3702(00)00047-3.

[123] S. Boyd, L. Xiao, and A. Mutapcic, "Subgradient methods", *lecture notes of EE392o, Stanford University, Autumn Quarter*, 2003.

[124] A. De Luca, G. Oriolo, and M. Vendittelli, "Stabilization of the unicycle via dynamic feedback linearization", *IFAC Proceedings Volumes*, vol. 33, no. 27, pp. 687–692, 2000. **DOI**: 10.1016/S1474-6670(17)38011-4.

[125] B. Könighofer, F. Lorber, N. Jansen, and R. Bloem, "Shield synthesis for reinforcement learning", in *Leveraging Applications of Formal Methods, Verification and Validation: Verification Principles*, Springer International Publishing, 2020, pp. 290–306. **DOI**: 10.1007/978-3-030-61362-4_16.

[126] B. P. G. Van Parys, D. Kuhn, P. J. Goulart, and M. Morari, "Distributionally robust control of constrained stochastic systems", *IEEE Transactions on Automatic Control*, vol. 61, no. 2, pp. 430–442, 2015. **DOI**: 10.1109/TAC.2015.2444134.

[127] A. Salamati, A. Lavaei, S. Soudjani, and M. Zamani, "Data-driven safety verification of stochastic systems via barrier certificates", *IFAC-PapersOnLine*, 2021. **DOI**: 10.1016/j.ifacol.2021.08.466.

[128] P. Jagtap, S. Soudjani, and M. Zamani, "Formal synthesis of stochastic systems via control barrier certificates", *IEEE Transactions on Automatic Control*, 2020. **DOI**: 10.1109/TAC.2020.3013916.

[129] D. Žikelić, M. Lechner, T. A. Henzinger, and K. Chatterjee, "Learning control policies for stochastic systems with reach-avoid guarantees", in *Proceedings of the Thirty-Seventh AAAI Conference on Artificial Intelligence*, AAAI Press, 2023. **DOI**: 10.1609/aaai.v37i10.26407.

[130] H. Sartipizadeh and B. Acikmese, "Approximate convex hull based sample truncation for scenario approach to chance constrained trajectory optimization", in *American Control Conference (ACC)*, 2018. **DOI**: 10.23919/acc.2018.8430936.

[131] A. Guttman, "R-trees: A dynamic index structure for spatial searching", *SIGMOD Record*, 1984. **DOI**: 10.1145/971697.602266.

[132] T. Badings, L. Romao, A. Abate, and N. Jansen, "Probabilities are not enough: Formal controller synthesis for stochastic dynamical models with epistemic uncertainty", in *Proceedings of the Thirty-Seventh AAAI Conference on Artificial Intelligence*, AAAI Press, 2023. **DOI**: 10.1609/aaai.v37i12.26718.

[133] M. L. Puterman, *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, Inc., 1994, **ISBN**: 9780471619772.

[134] G. Delimpaltadakis, M. Lahijanian, M. Mazo Jr, and L. Laurenti, "Interval markov decision processes with continuous action-spaces", in *Proceedings of the 26th International Conference on Hybrid Systems: Computation and Control*, 2023, pp. 1–10. **DOI**: 10.1145/3575870.3587117.

[135] M. Suilen, T. Badings, E. M. Bovy, D. Parker, and N. Jansen, "Robust markov decision processes: A place where ai and formal methods meet", in *Principles of Verification: Cycling the Probabilistic Landscape: Essays Dedicated to Joost-Pieter Katoen on the Occasion of His 60th Birthday, Part III*, Springer, 2024, pp. 126–154. **DOI**: 10.1007/978-3-031-75778-5_7.

[136] A. Nilim and L. El Ghaoui, "Robust control of markov decision processes with uncertain transition matrices", *Operations Research*, 2005. **DOI**: 10.1287/opre.1050.0216.

[137] W. Wiesemann, D. Kuhn, and B. Rustem, "Robust markov decision processes", *Mathematics of Operations Research*, vol. 38, no. 1, pp. 153–183, 2013. **DOI**: 10.1287/moor.1120.0566.

[138] M. Lahijanian, S. B. Andersson, and C. Belta, "Formal verification and synthesis for discrete-time stochastic systems", *IEEE Transactions on Automatic Control*, vol. 60, no. 8, pp. 2031–2045, 2015. **DOI**: 10.1109/TAC.2015.2398883.

[139] S. Haesaert and S. Soudjani, "Robust dynamic programming for temporal logic control of stochastic systems", *IEEE Transactions on Automatic Control*, vol. 66, no. 6, pp. 2496–2511, 2021. **DOI**: 10.1109/TAC.2020.3010490.

[140] R. Coppola, A. Peruffo, and M. Mazo, "Data-driven abstractions for verification of linear systems", *IEEE Control Systems Letters*, vol. 7, pp. 2737–2742, 2023. **DOI**: 10.1109/LCSYS.2023.3288731.

[141] A. Banse, L. Romao, A. Abate, and R. Jungers, "Data-driven memory-dependent abstractions of dynamical systems", in *Learning for Dynamics and Control Conference*, PMLR, 2023, pp. 891–902.

[142] A. Banse, L. Romao, A. Abate, and R. M. Jungers, "Data-driven abstractions via adaptive refinements and a kantorovich metric", in *62nd IEEE Conference on Decision and Control (CDC)*, 2023, pp. 6038–6043. **DOI**: 10.1109/CDC49753.2023.10383513.

[143] A. Lavaei, S. Soudjani, A. Abate, and M. Zamani, "Automated verification and synthesis of stochastic hybrid systems: A survey", *Automatica*, vol. 146, p. 110 617, 2022. **DOI**: 10.1016/j.automatica.2022.110617.

[144] S. Haddad and B. Monmege, "Interval iteration algorithm for MDPs and IMDPs", *Theoretical Computer Science*, vol. 735, pp. 111–131, 2018, Reachability Problems 2014: Special Issue. **DOI**: 10.1016/j.tcs.2016.12.003.

[145] J. Jiang, Y. Zhao, and S. Coogan, "Local-global interval MDPs for efficient motion planning with learnable uncertainty", in *American Control Conference (ACC)*, 2024, pp. 1343–1349. **DOI**: 10.23919/ACC60939.2024.10644660.

[146] M. J. T. C. van Zutphen, G. Delimpaltadakis, W. P. M. H. Heemels, and D. J. Antunes, "Predictable interval MDPs through entropy regularization", in *IEEE 63rd Conference on Decision and Control (CDC)*, 2024, pp. 6659–6664. **DOI**: 10.1109/CDC56724.2024.10886853.

[147] Y. Z. Lun, J. Wheatley, A. D'Innocenzo, and A. Abate, "Approximate abstractions of markov chains with interval decision processes", *IFAC-PapersOnLine*, vol. 51, no. 16, pp. 91–96, 2018. **DOI**: 10.1016/j.ifacol.2018.08.016.

[148] G. Delimpaltadakis, L. Laurenti, and M. Mazo, "Abstracting the sampling behaviour of stochastic linear periodic event-triggered control systems", in *60th IEEE Conference on Decision and Control (CDC)*, IEEE, 2021, pp. 1287–1294. **DOI**: 10.1109/CDC45484.2021.9683751.

[149] J. Jiang, Y. Zhao, and S. Coogan, "Safe learning for uncertainty-aware planning via interval MDP abstraction", *IEEE Control Systems Letters*, vol. 6, pp. 2641–2646, 2022. **DOI**: 10.1109/LCSYS.2022.3173993.

[150] M. Dutreix, J. Huh, and S. Coogan, "Abstraction-based synthesis for stochastic systems with omega-regular objectives", *Nonlinear Analysis: Hybrid Systems*, vol. 45, p. 101 204, 2022. **DOI**: 10.1016/j.nahs.2022.101204.

[151] M. Lahijanian, *bmdp-tool*, Standalone software, 2021. [Online]. Available: https://github.com/aria-systems-group/bmdp-tool.

[152] B. Wooding and A. Lavaei, "Impact: Interval MDP parallel construction for controller synthesis of large-scale stochastic systems", in *Quantitative Evaluation of Systems and Formal Modeling and Analysis of Timed Systems*, Springer Nature Switzerland, 2024, pp. 249–267. **DOI**: 10.1007/978-3-031-68416-6_15.

[153] N. Corporation, *NVIDIA CUDA C++ programming guide*, Version 12.3, 2023.

[154] J. Bezanson, S. Karpinski, V. B. Shah, and A. Edelman, *Julia: A fast dynamic language for technical computing*, 2012. arXiv: 1209.5145 [cs.PL].

[155] A. Buluç, J. T. Fineman, M. Frigo, J. R. Gilbert, and C. E. Leiserson, "Parallel sparse matrix-vector and matrix-transpose-vector multiplication using compressed sparse blocks", in *Proceedings of the twenty-first annual symposium on Parallelism in algorithms and architectures*, 2009, pp. 233–244. **DOI**: 10.1145/1583991.1584053.

[156] H. Peters, O. Schulz-Hildebrandt, and N. Luttenberger, "Fast in-place sorting with CUDA based on bitonic sort", in *Parallel Processing and Applied Mathematics: 8th International Conference*, Springer, 2010, pp. 403–410. **DOI**: 10.1007/978-3-642-14390-8_42.

[157] R. E. Ladner and M. J. Fischer, "Parallel prefix computation", *Journal of the ACM (JACM)*, vol. 27, no. 4, pp. 831–838, 1980. **DOI**: 10.1145/322217.322232.

[158] K. Chatterjee, K. Sen, and T. A. Henzinger, "Model-checking ω-regular properties of interval markov chains", in *Foundations of Software Science and Computational Structures: 11th International Conference*, Springer, 2008, pp. 302–317. **DOI**: 10.1007/978-3-540-78499-9_22.

[159] C. Boutilier, R. Dearden, and M. Goldszmidt, "Stochastic dynamic programming with factored representations", *Artificial Intelligence*, vol. 121, no. 1, pp. 49–107, 2000. **DOI**: 10.1016/S0004-3702(00)00033-3.

[160] P. Nilsson, S. Haesaert, R. Thakker, *et al.*, "Toward specification-guided active mars exploration for cooperative robot teams.", in *Robotics: Science and systems*, vol. 14, 2018, pp. 1–9.

[161] K. V. Delgado, S. Sanner, and L. N. de Barros, "Efficient solutions to factored MDPs with imprecise transition probabilities", *Artificial Intelligence*, vol. 175, no. 9, pp. 1498–1527, 2011. **DOI**: 10.1016/j.artint.2011.01.001.

[162] N. Cauchi and A. Abate, "StocHy: Automated verification and synthesis of stochastic processes", in *Tools and Algorithms for the Construction and Analysis of Systems*, Springer International Publishing, 2019, pp. 247–264. **DOI**: 10.1145/3302504.3313349.

[163] C. Liu, T. Arnon, C. Lazarus, C. Strong, C. Barrett, M. J. Kochenderfer, *et al.*, *Algorithms for verifying deep neural networks*. Now Publishers Inc., 2021, **ISBN**: 9781680837865.

[164] A. Patane, A. Blaas, L. Laurenti, L. Cardelli, S. Roberts, and M. Kwiatkowska, "Adversarial robustness guarantees for gaussian processes", *Journal of Machine Learning Research*, vol. 23, no. 146, pp. 1–55, 2022.

[165] Y. Schnitzer, A. Abate, and D. Parker, *Efficient solution and learning of robust factored MDPs*, 2025. arXiv: 2508.00707 [cs.LG].

[166] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*, 2nd ed. The MIT Press, 2001, **ISBN**: 9780262032933.

[167] Delft High Performance Computing Centre, *DelftBlue Supercomputer (Phase 2)*, https://www.tudelft.nl/dhpc/ark:/44463/DelftBluePhase2, 2024.

[168] A. P. Vinod, J. D. Gleason, and M. M. K. Oishi, *SReachTools: A MATLAB Stochastic Reachability Toolbox*, 2019-04. **DOI**: 10.1145/3302504.3313352.

[169] A. G. Barto, R. S. Sutton, and C. W. Anderson, "Neuronlike adaptive elements that can solve difficult learning control problems", *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SMC-13, no. 5, pp. 834–846, 1983. **DOI**: 10.1109/TSMC.1983.6313077.

[170] E.-K. Boukas, *Stochastic switching systems: analysis and design*. Springer Science & Business Media, 2007, **ISBN**: 9780817637828.

[171] M. Zamani, A. Abate, and A. Girard, "Symbolic models for stochastic switched systems: A discretization and a discretization-free approach", *Automatica*, vol. 55, pp. 183–196, 2015. **DOI**: 10.1016/j.automatica.2015.03.004.

[172] R. Mazouz, L. Laurenti, and M. Lahijanian, *Piecewise control barrier functions for stochastic systems*, 2025. arXiv: 2507.17703 [eess.SY].

[173] A. Nejati, A. Lavaei, P. Jagtap, S. Soudjani, and M. Zamani, "Formal verification of unknown discrete-and continuous-time systems: A data-driven approach", *IEEE Transactions on Automatic Control*, vol. 68, no. 5, pp. 3011–3024, 2023. **DOI**: 10.1109/TAC.2023.3255141.

[174] B. Xue, "Safe exit controllers synthesis for continuous-time stochastic systems", in *IEEE 63rd Conference on Decision and Control (CDC)*, IEEE, 2024, pp. 5150–5157. **DOI**: 10.1109/CDC56724.2024.10886819.

[175] M. I. Gomes and A. Guillou, "Extreme value theory and statistics of univariate extremes: A review", *International statistical review*, vol. 83, no. 2, pp. 263–292, 2015. **DOI**: 10.1111/insr.12058.

Acknowledgements

First and foremost, I would like to express my greatest gratitude to my supervisors, Prof. Luca Laurenti and Prof. Simeon C. Calvert, for giving me the opportunity to do a PhD and allowing me to undertake the intellectual and academic voyage this entails. I started my PhD during COVID and your unwavering support and invaluable guidance helped keep me sane, when the world appeared much less so. I also extend my gratitude to my promotor, Prof. Manuel Mazo, who asked tough but necessary questions regarding my research direction. I wish to acknowledge Prof. Tamás Keviczky under whom I TA'd the course "Control Theory". Before joining Delft Center for System and Control, my academic journey was strictly focused on computer science, and I was quite naïve about control theory. Nevertheless, Tamás welcomed me into the teaching team, allowing me to learn and grow – the experience taught me not only about control theory as a subject and how to teach Master's students, but also how to learn a new subject *quickly*. I also thank Professors Arkady Zgonnikov, Manon Kok, Dimitri Boskos, and Mohammad Khosravi for good discussions about research, academia, and beyond. The DCSC Head of Department, Prof. Bart de Schutter, also deserves a note of appreciation for listening to and taking issues of PhD candidates seriously; it is a tireless job with little appreciation, but extremely important, and for that you have my gratitude. My sincere thanks go to my doctoral committee, Prof. Matthijs Spaan, Prof. Nils Jansen, Prof. Raphael Jungers, Dr. Nicola Paoletti, and Prof. Jan-Willem van Wingerden, for their time, effort, and valuable feedback on this dissertation, as well as participating in the defense.

I would also like to express my appreciation and gratitude to all my collaborators, and to Luca for opening the door to many of these collaborations. In alphabetical order by surname, Alessandro Abate, Javier Alonso-Mora, Thom Badings, Simeon C. Calvert, Francesco Fabiano, Eduardo Figueiredo, Sofie Haesaert, Jeroen Hagenus, Luzia Knoedler, Jens Kober, Morteza Lahijanian, Luca Laurenti, Rayan Mazouz, Licio Romao, Julian F. Schumann, John Skovbekk, Nikolaus Vertovec, Xinyu Wang, and Arkady Zgonnikov. Their expertise, effort, contributions, and valuable input both significantly accelerated and enriched the research. I have come to the realization that I work best in collaborations, and working with these great people has in a very literal sense enabled me to complete my PhD. To Prof. Alessandro Abate and Dr. Nikolaus Vertovec, thank you for hosting me at Oxford University. It was a short and quite intense research visit, but simultaneously, in my opinion, a fun and successful visit.

I also would like to extend my thanks to my officemates. At first, our office was quite empty, in part due to COVID, and admittedly lonely. But after the first year, I got to share the office with two amazing people: Leila and Max, both of whom I have had many discussions about research and international politics. Later, David

and Maria joined the office, and I apologize to both of you for all my rants about papers. I greatly appreciate your friendship and I hope that we keep in touch.

Next, a word of appreciation goes to the PhD candidates and post.docs. from the labs of Prof. Laurenti and Prof. Zgonnikov: Steven, Ashwin, Julian, Eduardo, Mahshad, Darya, and Konstantinos. You made the everyday very enjoyable. A special thanks goes to Steven who started at the same time on the same project as me. You being on the same trajectory as me has been like having an academic twin; same but different. Most importantly, thank you for helping me settle in the Netherlands and for making me feel welcome. Many other PhD candidates, post.docs, and lecturers from DCSC also have my gratitude and deserve the greatest admiration. An extensive but non-exhaustive list: Afra, Albert, Alessandro, Alex, Amin, Anil, Athina, Atindriyo, Bart, Changrui, Claudia, Coen, Daniel, Emilio, Eva, Fililppo, Francesco, Frida, Fritz, Gabriel, Georgios, Giannis, Gianpetro, Ioannis, Ivo, Lars, Lotfi, Luca, Luyao, Maarten, Marcus, Maria, Mattia, Max, Meenakshi, Rayyan, Reza, Rogier, Rudi, Salim, Sam, Sander, Sasan, Shabnam, Sreeshma, Suad, Thomas, Tolga, Tushar, Wicak, Wolfram, Yichen, and Yun. For discussions about research, academia, software, culture, politics, history, and many other subjects over a cup of (machine) coffee at the department or a beer at Bebop. A special thanks goes to Léo – your kindness and persistence knows no bounds; both Laura and I are deeply grateful for your friendship.

Beyond academia, I enjoyed the company of many friends, old and new. Anyone who has completed a PhD can attest to the importance of having a good network outside of academia, and so I would like to express my gratitude to the following. To Anders, Simon, and Nikolaj with whom I studied in Aalborg – thank you for fantastic weekends with board games, barbecuing, fun jargon, and rants about software. To Marjolein, the first friend I made in the Netherlands – thank you for being you, being open-minded, and introducing me to Dutch culture. Finally, to Lisa-Marie, Andrea, Carmen, Roland, JC, Sebastian, Iiris, Marije, and Tom – thank you for welcoming me in your group, for fun and diverse board games, and for being fantastic company.

I owe everything to my family – my parents, Lisbeth and René, and my sisters, Sofie and Carina, and their families – for making me who I am, always welcoming me home, and keeping me grounded. I know that my research is so niche (as is most research) that you do not understand the specifics, but you listen regardless and for that I thank you. Finally, I express my most sincere gratitude to Laura – thank you for supporting me throughout this journey!

Curriculum Vitæ

Frederik Baymler Mathiesen

16-01-1996 Born in Hobro, Denmark.

Education

2016 – 2019 Bachelor of Science in Computer Science
Aalborg University, Denmark

2019 – 2021 Master of Science in Computer Science
Aalborg University, Denmark
Thesis: HyperVerlet: a Deep Learning Method for Numerically Solving Initial Value Problems of Hamiltonian Systems
Supervisors: Prof. dr. B. Yang
Dr. J. Hu

2021 – 2026 PhD. Systems and Control
Delft University of Technology
Thesis: Safety verification of stochastic systems with applications to autonomous vehicles
Promotors: Dr. Manuel Mazo Espinoza
Dr. Simeon Craig Calvert
Copromotor: Dr. Luca Laurenti

Awards and Certificates

2024 Honorable Mention for IFAC Young Author Award at IFAC ADHS 2024

2024 Outstanding Teaching Assistant Award
Delft Center for Systems and Control

2021 – 2025 DISC Certificate for Graduate Studies
Dutch Institute for Systems and Control

List of Publications

Journal papers

4. Schumann, J. F., Hagenus, J., **Mathiesen, F. B.**, & Zgonnikov, A. (2026). Realistic Adversarial Attacks for Robustness Evaluation of Trajectory Prediction Models via Future State Perturbation. ACM Journal on Autonomous Transportation Systems.

3. **Mathiesen, F. B.**, Romao, L., Calvert, S. C., Laurenti, L., & Abate, A. (2024). A data-driven approach for safety quantification of non-linear stochastic systems with unknown additive noise distribution. Accepted for Automatica, available on arXiv:2410.06662.

2. Mazouz, R., **Mathiesen, F. B.**, Laurenti, L., & Lahijanian, M. (2024). Piecewise Stochastic Barrier Functions. Accepted for Automatica, available on arXiv:2404.16986.

1. **Mathiesen, F. B.**, Calvert, S. C., & Laurenti, L. (2022). Safety certification for stochastic systems via neural barrier functions. IEEE Control Systems Letters, 7, 973-978.

Conference papers

9. Vertovec, N., **Mathiesen, F. B.**, Badings, T.., Laurenti, L., & Abate, A. (2025). Scalable Verification of Neural Control Barrier Functions Using Linear Bound Propagation. Accepted for Learning for Decision and Control, 2026, available on arXiv:2511.06341.

8. **Mathiesen, F. B.**, Vertovec, N., Fabiano, F., Laurenti, L., & Abate, A. (2025). Certified Neural Approximations of Nonlinear Dynamics. Under review, available on arXiv:2505.15497.

7. **Mathiesen, F. B.**, Haesaert, S., & Laurenti, L. (2025). Scalable control synthesis for stochastic systems via structural IMDP abstractions. In Proceedings of the 28th ACM International Conference on Hybrid Systems: Computation and Control (pp. 1-12).

6. Mazouz, R., Skovbekk, J., **Mathiesen, F. B.**, Frew, E., Laurenti, L., & Lahijanian, M. (2024). Data-driven permissible safe control with barrier certificates. In 2024 IEEE 63rd Conference on Decision and Control (CDC) (pp. 6844-6849). IEEE.

5. Wang, X., Knoedler, L., **Mathiesen, F. B.**, & Alonso-Mora, J. (2024). Simultaneous synthesis and verification of neural control barrier functions through branch-and-bound verification-in-the-loop training. In 2024 European Control Conference (ECC) (pp. 571-578). IEEE.

4. Hagenus, J., **Mathiesen, F. B.**, Schumann, J. F., & Zgonnikov, A. (2024). Robustness in trajectory prediction for autonomous vehicles: a survey. In 2024 IEEE Intelligent Vehicles Symposium (IV) (pp. 969-976). IEEE.

3. **Mathiesen, F. B.**, Lahijanian, M., & Laurenti, L. (2024). IntervalMDP. jl: Accelerated Value Iteration for Interval Markov Decision Processes. In 2024 Analysis and Design of Hybrid Systems (ADHS). IFAC.

2. **Mathiesen, F. B.**, Romao, L., Calvert, S. C., Abate, A., & Laurenti, L. (2023). Inner approximations of stochastic programs for data-driven stochastic barrier function design. In 2023 62nd IEEE Conference on Decision and Control (CDC) (pp. 3073-3080). IEEE.

1. **Mathiesen, F. B.**, Yang, B., & Hu, J. (2022). Hyperverlet: A symplectic hypersolver for Hamiltonian systems. In Proceedings of the AAAI Conference on Artificial Intelligence (Vol. 36, No. 4, pp. 4575-4582).

indicates that (part of) the manuscript is included in the dissertation.