Learning Control Policies for Stochastic Systems with Reach-Avoid Guarantees

Đorđe Žikelić[1*], Mathias Lechner[2*], Thomas A. Henzinger[1], Krishnendu Chatterjee[1]

[1]Institute of Science and Technology Austria (ISTA)
[2]Massachusetts Institute of Technology (MIT)
djordje.zikelic@ist.ac.at, mlechner@mit.edu, tah@ist.ac.at, krishnendu.chatterjee@ist.ac.at

Abstract

We study the problem of learning controllers for discrete-time non-linear stochastic dynamical systems with formal reach-avoid guarantees. This work presents the first method for providing formal reach-avoid guarantees, which combine and generalize stability and safety guarantees, with a tolerable probability threshold $p \in [0, 1]$ over the infinite time horizon in general Lipschitz continuous systems. Our method leverages advances in machine learning literature and it represents formal certificates as neural networks. In particular, we learn a certificate in the form of a reach-avoid supermartingale (RASM), a novel notion that we introduce in this work. Our RASMs provide reachability and avoidance guarantees by imposing constraints on what can be viewed as a stochastic extension of level sets of Lyapunov functions for deterministic systems. Our approach solves several important problems – it can be used to learn a control policy from scratch, to verify a reach-avoid specification for a fixed control policy, or to fine-tune a pre-trained policy if it does not satisfy the reach-avoid specification. We validate our approach on 3 stochastic non-linear reinforcement learning tasks.

Introduction

Reinforcement learning (RL) has achieved impressive results in solving non-linear control problems, resulting in an interest to deploy RL algorithms in safety-critical applications. However, most RL algorithms focus solely on optimizing expected performance and do not take safety constraints into account (Sutton and Barto 2018). This raises concerns about their applicability to safety-critical domains in which unsafe behavior can lead to catastrophic consequences (Amodei et al. 2016; García and Fernández 2015). Complicating matters, models are usually imperfect approximations of real systems that are obtained from observed data, thus models often need to account for uncertainty which is modelled via stochastic disturbances. Formal safety verification of policies learned via RL algorithms and design of learning algorithms that take safety constraints into account have thus become very active research topics.

Reach-avoid constraints are one of the most common and practically relevant constraints appearing in safety-critical applications that generalize both reachability and safety constraints (Summers and Lygeros 2010). Given a target region and an unsafe region, the reach-avoid constraint requires that a system controlled by a policy converges to the target region while avoiding the unsafe region. For instance, a lane-keeping constraint requires a self-driving car to reach its destination without leaving the allowed car lanes (Vahidi and Eskandarian 2003). In the case of stochastic control problems, reach-avoid constraints are also specified by a minimal probability with which the system controlled by a policy needs to satisfy the reach-avoid constraint.

In this work, we consider discrete-time stochastic control problems under reach-avoid constraints. Following the recent trend that aims to leverage advances in deep RL to safe control, we propose a learning method that learns a control policy together with a formal reach-avoid certificate in the form of a *reach-avoid supermartingale (RASM)*, a novel notion that we introduce in this work. Informally, an RASM is a function assigning nonnegative real values to each state that is required to strictly decrease in expected value until the target region is reached, but needs to strictly increase for the system to reach the unsafe region. By carefully choosing the ratio of the initial level set of the RASM and the least level set that the RASM needs to attain for the system to reach the unsafe region (here we use the standard level set terminology of Lyapunov functions (Haddad and Chellaboina 2011)), we obtain a formal reach-avoid certificate. The name of RASMs is chosen to emphasize the connection to supermartingale processes in probability theory (Williams 1991). Our RASMs significantly generalize and unify the stochastic control barrier functions which are a standard certificate for safe control of stochastic systems (Prajna, Jadbabaie, and Pappas 2007) and ranking supermartingales that certify probability 1 reachability and stability in (Lechner et al. 2022).

Contributions. This work presents the first control method that provides formal *reach-avoid guarantees* for control of stochastic systems with a *specified probability threshold* over the *infinite time horizon* in *Lipschitz continuous* systems. In contrast, the existing approaches to control under reach-avoid constraints are only applicable to finite horizon settings, polynomial stochastic systems or to deterministic systems (see the following section for an overview of related work). Moreover, our method simultaneously *learns* the control policy and the RASM certificate in the form of neural networks

*Equal Contribution.

and is applicable to general non-linear systems. This contrasts the existing methods from the literature that are based on stochastic control barrier functions, which utilize convex optimization tools to compute control policies and are restricted to polynomial system dynamics and policies (Prajna, Jadbabaie, and Pappas 2007; Steinhardt and Tedrake 2012; Santoyo, Dutreix, and Coogan 2021; Xue et al. 2021). Our algorithm draws insight from established methods for learning Lyapunov functions for stability in deterministic control problems (Richards, Berkenkamp, and Krause 2018; Chang, Roohi, and Gao 2019; Abate et al. 2021), which were demonstrated to be more efficient than the existing convex optimization methods and were adapted in (Lechner et al. 2022) for probability 1 reachability and stability verification. Finally, our method learns a suitable policy on demand, or alternatively, *verifies* reach-avoid properties of a fixed Lipschitz continuous control policy. We experimentally validate our method on 3 stochastic RL tasks and show that it efficiently learns control policies with probabilistic reach-avoid guarantees in practice.

Related Work

Deterministic control problems There is extensive literature on safe control, with most works certifying stability via Lyapunov functions (Haddad and Chellaboina 2011) or safety via control barrier functions (Ames et al. 2019). Most early works rely either on hand-designed certificates, or automate their computation through convex optimization methods such as sum-of-squares (SOS) programming (Henrion and Garulli 2005; Parrilo 2000; Jarvis-Wloszek et al. 2003). Automation via SOS programming is restricted to problems with polynomial system dynamics and does not scale well with dimension. A promising approach to overcome these limitations is to learn a control policy together with a safety certificate in the form of neural networks, for instance see (Richards, Berkenkamp, and Krause 2018; Sun, Jha, and Fan 2020; Jin et al. 2020; Chang and Gao 2021; Qin et al. 2021). In particular, (Chang, Roohi, and Gao 2019; Abate et al. 2021) learn a control policy and a certificate as neural networks by using a learner-verifier framework which repeatedly learns a candidate policy and a certificate and then tries to either verify or refine them. Our method extends some of these ideas to stochastic systems.

Stochastic control problems Safe control of stochastic systems has received comparatively less attention. Most existing approaches are abstraction based – they consider finite-time horizon systems and approximate them via a finite-state Markov decision process (MDP). The constrained control problem is then solved for the MDP. Due to accumulation of the approximation error in each time step, the size of the MDP state space needs to grow with the length of the considered time horizon, making these methods applicable to systems that evolve over fixed finite time horizons. Notable examples include (Soudjani, Gevaerts, and Abate 2015; Lavaei et al. 2020; Cauchi and Abate 2019; Vinod, Gleason, and Oishi 2019; Vaidya 2015; Crespo and Sun 2003). An abstraction based method for obtaining infinite time horizon PAC-style guarantees on reach-avoidance in linear stochastic systems was proposed in (Badings et al. 2022). This method is applicable to systems with both aleatoric and epistemic uncertainty. Another line of work considers polynomial systems and utilizes stochastic control barrier functions and convex optimization tools to compute polynomial control policies (Prajna, Jadbabaie, and Pappas 2007; Steinhardt and Tedrake 2012; Santoyo, Dutreix, and Coogan 2021; Xue et al. 2021; Mazouz et al. 2022). Concurrently to our work, (Mathiesen, Calvert, and Laurenti 2023) proposed a learning-based method for formal safety verification in continuous stochastic control systems over a fixed finite time horizon, by learning a neural network stochastic control barrier function.

Constrained MDPs Safe RL has also been studied in the context of constrained MDPs (CMDPs) (Altman 1999; Geibel 2006). An agent in a CMDP must satisfy hard constraints on expected cost for one or more auxiliary notions of cost aggregated over an episode. Several works study RL algorithms for CMDPs (Uchibe and Doya 2007), notably the Constrained Policy Optimization (CPO) (Achiam et al. 2017) or the method (Chow et al. 2018) which proposed a Lyapunov method for solving CMDPs. While these algorithms perform well, their constraints are satisfied in expectation which makes them less suitable for safety-critical systems.

Safe RL via shielding Some approaches ensure safety by computing two control policies – the main policy that optimizes the expected reward, and the backup policy that the system falls back to whenever a safety constraint may be violated (Michalska and Mayne 1993; Perkins and Barto 2002; Alshiekh et al. 2018; Elsayed-Aly et al. 2021; Giacobbe et al. 2021). The backup policy can thus be of simpler form. Shielding for stochastic linear systems with additive disturbances has been considered in (Wabersich and Zeilinger 2018). (Li and Bastani 2020; Bastani and Li 2021) are applicable to stochastic non-linear systems, however their safety guarantees are *statistical* – their algorithms are randomized with parameters $\delta, \epsilon \in (0, 1)$ and they with probability $1 - \delta$ compute an action that is safe in the current state with probability at least $1 - \epsilon$. The statistical error is accumulated at each state, hence these approaches are not suitable for infinite or long time horizons. In contrast, our approach targets *formal* guarantees for *infinite* time horizon problems.

Safe exploration Model-free RL algorithms need to explore the state space in order to learn high performing actions. Safe exploration RL restricts exploration in a way which ensures that given safety constraints are satisfied. The most common approach to ensuring safe exploration is learning the system dynamics' uncertainty bounds and limiting the exploratory actions within a high probability safety region, with the existing methods based on Gaussian Processes (Koller et al. 2018; Turchetta, Berkenkamp, and Krause 2019; Berkenkamp 2019), linearized models (Dalal et al. 2018), deep robust regression (Liu et al. 2020), safe padding (Hasanbeig, Abate, and Kroening 2020) and Bayesian neural networks (Lechner et al. 2021). Recent work has also considered learning stable stochastic dynamics from data (Umlauft and Hirche 2017; Lawrence et al. 2020).

Probabilistic program analysis Supermartingales have also been used for the analysis of probabilistic programs (PPs). In particular, RSMs were originally used to prove almost-sure termination in PPs (Chakarov and Sankaranarayanan 2013) and (Abate, Giacobbe, and Roy 2021) learns RSMs in PPs. Supermartingales were also used for probabilistic termination and safety analysis in PPs (Chatterjee, Novotný, and Zikelic 2017; Chatterjee et al. 2022).

Preliminaries

We consider discrete-time stochastic dynamical systems defined by the equation

$$\mathbf{x}_{t+1} = f(\mathbf{x}_t, \mathbf{u}_t, \omega_t),\ \mathbf{x}_0 \in \mathcal{X}_0.$$

The function $f : \mathcal{X} \times \mathcal{U} \times \mathcal{N} \to \mathcal{X}$ defines system dynamics, where $\mathcal{X} \subseteq \mathbb{R}^m$ is the system state space, $\mathcal{U} \subseteq \mathbb{R}^n$ is the control action space and $\mathcal{N} \subseteq \mathbb{R}^p$ is the stochastic disturbance space. We use $t \in \mathbb{N}_0$ to denote the time index, $\mathbf{x}_t \in \mathcal{X}$ the state of the system, $\mathbf{u}_t \in \mathcal{U}$ the action and $\omega_t \in \mathcal{N}$ the stochastic disturbance vector at time t. The set $\mathcal{X}_0 \subseteq \mathcal{X}$ is the set of initial states. The action $\mathbf{u}_t$ is chosen according to a control policy $\pi : \mathcal{X} \to \mathcal{U}$, i.e. $\mathbf{u}_t = \pi(\mathbf{x}_t)$. The stochastic disturbance vector ω_t is sampled according to a specified probability distribution d over $\mathbb{R}^p$. The dynamics function f, control policy π and probability distribution d together define a stochastic feedback loop system.

A sequence $(\mathbf{x}_t, \mathbf{u}_t, \omega_t)_{t\in\mathbb{N}_0}$ of state-action-disturbance triples is a trajectory of the system, if for each $t \in \mathbb{N}_0$ we have $\mathbf{u}_t = \pi(\mathbf{x}_t)$, $\omega_t \in \mathsf{support}(d)$ and $\mathbf{x}_{t+1} = f(\mathbf{x}_t, \mathbf{u}_t, \omega_t)$. For each initial state $\mathbf{x}_0 \in \mathcal{X}$, the system induces a Markov process which gives rise to the probability space over the set of all trajectories that start in $\mathbf{x}_0$ (Puterman 1994). We denote the probability measure and the expectation in this probability space by $\mathbb{P}_{\mathbf{x}_0}$ and $\mathbb{E}_{\mathbf{x}_0}$.

Assumptions We assume that $\mathcal{X} \subseteq \mathbb{R}^m$, $\mathcal{X}_0 \subseteq \mathbb{R}^m$, $\mathcal{U} \subseteq \mathbb{R}^n$ and $\mathcal{N} \subseteq \mathbb{R}^p$ are all Borel-measurable, which is a technical assumption necessary for the system semantics to be mathematically well-defined. We also assume that $\mathcal{X} \subseteq \mathbb{R}^m$ is compact and that the dynamics function f is Lipschitz continuous, which are common assumptions in control theory.

Probabilistic reach-avoid problem Let $\mathcal{X}_t \subseteq \mathcal{X}$ and $\mathcal{X}_u \subseteq \mathcal{X}$ be disjoint Borel-measurable subsets of $\mathbb{R}^m$, which we refer to as the *target set* and the *unsafe set*, respectively. Let $p \in [0, 1]$ be a probability threshold. Our goal is to learn a control policy which guarantees that, with probability at least p, the system reaches the target set $\mathcal{X}_t$ without reaching the unsafe set $\mathcal{X}_u$. Formally, we want to learn a control policy π such that, for any initial state $\mathbf{x}_0 \in \mathcal{X}_0$, we have

$$\mathbb{P}_{\mathbf{x}_0}\Big[\mathrm{ReachAvoid}(\mathcal{X}_t, \mathcal{X}_u)\Big] \geq p$$

with $\mathrm{ReachAvoid}(\mathcal{X}_t, \mathcal{X}_u) = \{(\mathbf{x}_t, \mathbf{u}_t, \omega_t)_{t\in\mathbb{N}_0} \mid \exists t \in \mathbb{N}_0.\, \mathbf{x}_t \in \mathcal{X}_t \wedge (\forall t' \leq t.\, \mathbf{x}_{t'} \notin \mathcal{X}_u)\}$ the set of trajectories that reach $\mathcal{X}_t$ without reaching $\mathcal{X}_u$.

We restrict to the cases when either $p < 1$, or $p = 1$ and $\mathcal{X}_u = \emptyset$. Our approach is not applicable to the case $p = 1$ and $\mathcal{X}_u \neq \emptyset$ due to technical issues that arise in defining our formal certificate, which we discuss in the following section. We remark that probabilistic reachability is a special instance of our problem obtained by setting $\mathcal{X}_u = \emptyset$. On the other hand, we cannot directly obtain the probabilistic safety problem by assuming any specific form of the target set $\mathcal{X}_t$, however we will show in the following section that our method implies probabilistic safety with respect to $\mathcal{X}_u$ if we provide it with $\mathcal{X}_t = \emptyset$.

Theoretical Results

We now present our framework for formally certifying a reach-avoid constraint with a given probability threshold. Our framework is based on the novel notion of *reach-avoid supermartingales (RASMs)* that we introduce in this work. Note that, in this section only, we assume that the policy is fixed. In the next section, we will present our algorithm for learning policies that provide formal reach-avoid guarantees in which RASMs will be an integral ingredient. In what follows, we consider a discrete-time stochastic dynamical system defined as in the previous section. For now, we assume that the probability threshold is strictly smaller than 1, i.e. $p < 1$. We will later show that our approach straightforwardly extends to the case $p = 1$ and $\mathcal{X}_u = \emptyset$.

Reach-avoid supermartingales We define a *reach-avoid supermartingale (RASM)* to be a continuous function $V : \mathcal{X} \to \mathbb{R}$ that assigns real values to system states. The name is chosen to emphasize the connection to supermartingale processes from probability theory (Williams 1991), which we will explore later in order to prove the effectiveness of RASMs for verifying reach-avoid properties. The value of V is required to be nonnegative over the state space $\mathcal{X}$ (Nonnegativity condition), to be bounded from above by 1 over the set of initial states $\mathcal{X}_0$ (Initial condition) and to be bounded from below by $\frac{1}{1-p}$ over the set of unsafe states $\mathcal{X}_u$ (Safety condition). Hence, in order for a system trajectory to reach an unsafe state and violate the safety specification, the value of the RASM V needs to increase at least $\frac{1}{1-p}$ times along the trajectory. Finally, we require the existence of $\epsilon > 0$ such that the value of V decreases in expected value by at least ϵ after every one-step evolution of the system from every system state $\mathbf{x} \in \mathcal{X}\backslash\mathcal{X}_t$ for which $V(\mathbf{x}) \leq \frac{1}{1-p}$ (Expected decrease condition). Intuitively, this last condition imposes that the system has a tendency to strictly *decrease* the value of V until either the target set $\mathcal{X}_t$ is reached or a state with $V(\mathbf{x}) \geq \frac{1}{1-p}$ is reached. However, as the value of V needs to *increase* at least $\frac{1}{1-p}$ times in order for the system to reach an unsafe state, these four conditions will allow us to use RASMs to certify that the reach-avoid constraint is satisfied with probability at least p.

Definition 1 (Reach-avoid supermartingales)**.** *Let $\mathcal{X}_t \subseteq \mathcal{X}$ and $\mathcal{X}_u \subseteq \mathcal{X}$ be the target set and the unsafe set, and let $p \in [0, 1)$ be the probability threshold. A continuous function $V : \mathcal{X} \to \mathbb{R}$ is said to be a* reach-avoid supermartingale (RASM) *with respect to $\mathcal{X}_t$, $\mathcal{X}_u$ and p if it satisfies:*

1. Nonnegativity condition. $V(\mathbf{x}) \geq 0$ *for each* $\mathbf{x} \in \mathcal{X}$.
2. Initial condition. $V(\mathbf{x}) \leq 1$ *for each* $\mathbf{x} \in \mathcal{X}_0$.
3. Safety condition. $V(\mathbf{x}) \geq \frac{1}{1-p}$ *for each* $\mathbf{x} \in \mathcal{X}_u$.
4. Expected decrease condition. *There exists $\epsilon > 0$ such that, for each $\mathbf{x} \in \mathcal{X}\backslash\mathcal{X}_t$ at which $V(\mathbf{x}) \leq \frac{1}{1-p}$, we have* $V(\mathbf{x}) \geq \mathbb{E}_{\omega\sim d}[V(f(\mathbf{x}, \pi(\mathbf{x}), \omega))] + \epsilon$.

Comparison to Lyapunov functions The defining properties of RASMs hint a connection to Lyapunov functions for deterministic control systems. However, the key difference between Lyapunov functions and our RASMs is that Lyapunov functions deterministically decrease in value whereas RASMs decrease in expectation. Deterministic decrease ensures that each level set of a Lyapunov function, i.e. a set of states at which the value of Lyapunov functions is at most l for some $l \geq 0$, is an invariant of the system. However, it is in general not possible to impose such a condition on stochastic systems. In contrast, our RASMs only require expected decrease in the level, and the Initial and the Unsafe conditions can be viewed as conditions on the *maximal initial level set* and the *minimal unsafe level set*. The choice of a ratio of these two level values allows us to use existing results from martingale theory in order to obtain probabilistic avoidance guarantees, while the Expected decrease condition by $\epsilon > 0$ furthermore provides us with probabilistic reachability guarantees.

Certifying reach-avoid constraints via RASMs We now show that the existence of an ϵ-RASM for some $\epsilon > 0$ implies that the reach-avoid constraint is satisfied with probability at least p.

Theorem 1. *Let $\mathcal{X}_t \subseteq \mathcal{X}$ and $\mathcal{X}_u \subseteq \mathcal{X}$ be the target set and the unsafe set, respectively, and let $p \in [0, 1)$ be the probability threshold. Suppose that there exists an RASM V with respect to $\mathcal{X}_t$, $\mathcal{X}_u$ and p. Then, for every $\mathbf{x}_0 \in \mathcal{X}_0$, $\mathbb{P}_{\mathbf{x}_0}[\mathit{ReachAvoid}(\mathcal{X}_t, \mathcal{X}_u)] \geq p$.*

The complete proof of Theorem 1 is provided in the extended version of the paper (Žikelić et al. 2022). However, in what follows we sketch the key ideas behind our proof, in order to illustrate the applicability of martingale theory to reasoning about stochastic systems which we believe to have significant potential for applications beyond the scope of this work. To prove the theorem, we first show that an ϵ-RASM V induces a *supermartingale* (Williams 1991) in the probability space over the set of all trajectories that start in an initial state $\mathbf{x}_0 \in \mathcal{X}_0$. Intuitively, a supermartingale in a probability space $(\Omega, \mathcal{F}, \mathbb{P})$ is a stochastic process $(X_t)_{t=0}^{\infty}$ such that, for each $t \in \mathbb{N}_0$, the expected value of X_{t+1} conditioned on the value of X_t is less than or equal to X_t. We formalize this definition together with the notion of conditional expectation and provide an overview of definitions and results form martingale theory that we use in our proof in the extended version of the paper (Žikelić et al. 2022).

Now, let $(\Omega_{\mathbf{x}_0}, \mathcal{F}_{\mathbf{x}_0}, \mathbb{P}_{\mathbf{x}_0})$ be the probability space of trajectories that start in $\mathbf{x}_0$. Then, for each time step $t \in \mathbb{N}_0$, we define a random variable

$$X_t(\rho) = \begin{cases} V(\mathbf{x}_t), & \text{if } \mathbf{x}_i \notin \mathcal{X}_t \text{ and } V(\mathbf{x}_i) < \frac{1}{1-p} \\ & \text{ for each } 0 \leq i \leq t \\ 0, & \text{if } \mathbf{x}_i \in \mathcal{X}_t \text{ for some } 0 \leq i \leq t \\ & \text{ and } V(\mathbf{x}_j) < \frac{1}{1-p} \text{ for each } 0 \leq j \leq i \\ \frac{1}{1-p}, & \text{otherwise} \end{cases}$$

for each trajectory $\rho = (\mathbf{x}_t, \mathbf{u}_t, \omega_t)_{t \in \mathbb{N}_0} \in \Omega_{\mathbf{x}_0}$. In other words, the value of X_t is equal to the value of V at $\mathbf{x}_t$, unless either the target set $\mathcal{X}_t$ has been reached first in which case we set all future values of $\mathcal{X}_t$ to 0, or a state in which V exceeds $\frac{1}{1-p}$ has been reached first in which case we set all future values of $\mathcal{X}_t$ to $\frac{1}{1-p}$. Then, since V satisfies the Nonnegativity and the Expected decrease condition of RASMs, we may show that $(X_t)_{t=0}^{\infty}$ is a supermartingale. in the probability space $(\Omega_{\mathbf{x}_0}, \mathcal{F}_{\mathbf{x}_0}, \mathbb{P}_{\mathbf{x}_0})$.

Next, we show that the nonnegative supermartingale $(X_t)_{t=0}^{\infty}$ with probability 1 converges to and reaches 0 or a value that is greater than or equal to $\frac{1}{1-p}$. To do this, we first employ the Supermartingale Convergence Theorem (see the extended version of the paper (Žikelić et al. 2022)) which states that every nonnegative supermartingale converges to some value with probability 1. We then use the fact that, in the Expected decrease condition of RASMs, the decrease in expected value is strict and by at least $\epsilon > 0$, in order to conclude that this value is reached and has to be either 0 or greater than or equal to $\frac{1}{1-p}$.

Finally, we use another classical result from martingale theory (see the extended version of the paper (Žikelić et al. 2022)) which states that, given a nonnegative supermartingale $(X_t)_{t=0}^{\infty}$ and $\lambda > 0$,

$$\mathbb{P}\Big[\sup_{i \geq 0} X_i \geq \lambda\Big] \leq \frac{\mathbb{E}[X_0]}{\lambda}.$$

Plugging $\lambda = \frac{1}{1-p}$ into the above inequality, it follows that $\mathbb{P}_{\mathbf{x}_0}[\sup_{i \geq 0} X_i \geq \frac{1}{1-p}] \leq (1-p) \cdot \mathbb{E}_{\mathbf{x}_0}[X_0] \leq 1 - p$. The second inequality follows since $X_0(\rho) = V(\mathbf{x}_0) \leq 1$ for every $\rho \in \Omega_{\mathbf{x}_0}$ by the Initial condition of RASMs. Hence, as $(X_t)_{t=0}^{\infty}$ with probability 1 either reaches 0 or a value that is greater than or equal to $\frac{1}{1-p}$, we conclude that $(X_t)_{t=0}^{\infty}$ reaches 0 without reaching a value that is greater than or equal to $\frac{1}{1-p}$ with probability at least p. By the definition of each X_t and by the Safety condition of RASMs, this implies that with probability at least p the system will reach the target set $\mathcal{X}_t$ without reaching the unsafe set $\mathcal{X}_u$, i.e. that $\mathbb{P}_{\mathbf{x}_0}[\mathrm{ReachAvoid}(\mathcal{X}_t, \mathcal{X}_u)] \geq p$.

Probabilistic safety In order to solve the probabilistic safety problem and verify that a control policy guarantees that the unsafe set $\mathcal{X}_u$ is not reached with probability at least p, we may modify the Expected decrease condition of RASMs by setting $\mathcal{X}_t = \emptyset$. Thus, RASMs are also effective for the probabilistic safety problem. This claim follows immediately from our proof of Theorem 1. In this case and if we set $\epsilon = 0$, then our RASMs coincide with stochastic barrier functions of (Prajna, Jadbabaie, and Pappas 2007). However, if $\mathcal{X}_t$ is not empty, then we must have $\epsilon > 0$ in order to enforce convergence and reachability of $\mathcal{X}_t$.

Extension to $p = 1$ and $\mathcal{X}_u = \emptyset$ and comparison to RSMs So far, we have only considered $p \in [0, 1)$. The difficulty in the case $p = 1$ arises since the value $\frac{1}{1-p}$ in the Safety and the Expected decrease conditions in Definition 1 would not be well-defined. However, if $\mathcal{X}_u = \emptyset$, then the Safety condition need not be imposed at any state. Moreover, it follows directly from our proof that imposing the expected decrease condition at all states in $\mathcal{X} \backslash \mathcal{X}_t$ makes RASMs sound for certifying probability 1 reachability. In fact, in this special case our

RASMs reduce to the RSMs of (Lechner et al. 2022). The key novelty of our RASMs over RSMs is that we also employ *level set reasoning* in order to obtain probabilistic reach-avoid guarantees, thus presenting a true *stochastic extension of Lyapunov functions* that allow reasoning both about reach-avoid specifications as well as quantitative reasoning about the probability with which they are satisfied. In contrast, RSMs do not reason about level sets and can only certify probability 1 reachability.

Learning Reach-avoid Policies

We now present our algorithm for learning policies with reach-avoid guarantees, which learns a policy together with an RASM certificate. The algorithm consists of two modules called *learner* and *verifier*, which are composed into a loop. In each loop iteration, the learner learns a policy together with an RASM candidate as two neural networks π_θ and V_ν, with θ and ν being vectors of neural network parameters. The verifier then formally verifies whether the learned RASM candidate is indeed an RASM for the system and the learned policy. If the answer is positive, then the algorithm concludes that the learned policy provides formal reach-avoid guarantees. Otherwise, the verifier computes a counterexample which shows that the learned RASM candidate is not an RASM. The counterexample is passed to the learner and used to modify the loss function towards learning a new policy and an RASM candidate. The loop is repeated until either a candidate is successfully verified or the algorithm reaches a specified timeout. The algorithm is presented in Algorithm 1. We note that our algorithm can also *verify* whether a given Lipschitz continuous policy provides reach-avoid guarantees, by fixing the policy only learning the RASM neural network.

Policy Initialization Learning two networks concurrently with multiple objectives can be unstable due to dependencies between the two networks and differences in the scale of the objective loss terms. To mitigate these instabilities, we propose pre-training of the policy network so that our algorithm starts from a proper initialization. In particular, from the given dynamical system and the safety specification, we induce a Markov decision process (MDP) intending to reach the target set while avoiding the unsafe set. The reward term r_t is given by $r_t := \mathbb{1}[\mathcal{X}_t](\mathbf{x}_t) - \mathbb{1}[\mathcal{X}_u](\mathbf{x}_t)$ and we use proximal policy optimization (PPO) (Schulman et al. 2017) to train the policy.

State Space Discretization When it comes to verifying learned candidates, the key difficulty lies in checking the Expected decrease condition. This is because, in general, it is not possible to compute a closed form expression for the expected value of an RASM over successor system states, as both the policy and the RASM are neural networks. In order to overcome this difficulty, our algorithm discretizes the state space of the system. Given a *mesh* parameter $\tau > 0$, a *discretization* $\tilde{\mathcal{X}}$ of $\mathcal{X}$ with mesh τ is a set of states such that, for every $\mathbf{x} \in \mathcal{X}$, there exists a state $\tilde{\mathbf{x}} \in \tilde{\mathcal{X}}$ such that $||\mathbf{x} - \tilde{\mathbf{x}}||_1 < \tau$. Due to $\mathcal{X}$ being compact and therefore bounded, for any $\tau > 0$ it is possible to compute its finite discretization with mesh τ by simply considering vertices of a grid with sufficiently small cells. Note that f, π_θ and

Algorithm 1: Algorithm for learning reach-avoid policies

Input f, d, $\mathcal{X}$, $\mathcal{X}_0$, $\mathcal{X}_t$, $\mathcal{X}_u$, L_f, $p \in [0, 1]$
Parameters mesh $\tau > 0$, number of samples $N \in \mathbb{N}$, regularization constant $\lambda > 0$
$\pi_\theta \leftarrow$ trained by PPO
$\tilde{\mathcal{X}} \leftarrow$ discretization of $\mathcal{X}$ with mesh τ
$C_{\text{init}}, C_{\text{unsafe}}, C_{\text{decrease}} \leftarrow \tilde{\mathcal{X}} \cap \mathcal{X}_0,\ \tilde{\mathcal{X}} \cap \mathcal{X}_u,\ \tilde{\mathcal{X}} \cap (\mathcal{X} \backslash \mathcal{X}_t)$
$V_\nu \leftarrow$ trained by minimizing the loss function
while timeout not reached **do**
  $L_\pi, L_V \leftarrow$ Lipschitz constants of π_θ, V_ν
  $K \leftarrow L_V \cdot (L_f \cdot (L_\pi + 1) + 1)$
  $\tilde{\mathcal{X}}_e \leftarrow$ vertices of discr. $\tilde{\mathcal{X}}$ whose adjacent cells intersect $\mathcal{X} \backslash \mathcal{X}_t$ and contain $\mathbf{x}$ s.t. $V_\nu(\mathbf{x}) < \frac{1}{1-p}$
  $\text{Cells}_{\mathcal{X}_0}$, $\text{Cells}_{\mathcal{X}_u} \leftarrow$ discr. cells that intersect $\mathcal{X}_0$, $\mathcal{X}_u$
  if $\exists \tilde{\mathbf{x}} \in \tilde{\mathcal{X}}_e \cap (\mathcal{X} \backslash \mathcal{X}_t)$ s.t. $\mathbb{E}_{\omega \sim d}[V_\nu(f(\tilde{\mathbf{x}}, \pi(\tilde{\mathbf{x}}), \omega))] \geq V_\nu(\tilde{\mathbf{x}}) - \tau \cdot K$ and $V_\nu(\tilde{\mathbf{x}}) < \frac{1}{1-p}$ **then**
    $C_{\text{decrease}} \leftarrow C_{\text{decrease}} \cup \{\mathbf{x}\}$
  else if $\exists$cell $\in \text{Cells}_{\mathcal{X}_0}$ s.t. $\sup_{\mathbf{x} \in \text{cell}} V_\nu(\mathbf{x}) > 1$ **then**
    $C_{\text{init}} \leftarrow C_{\text{init}} \cup (\{\text{vertices of cell}\} \cap \mathcal{X}_0)$
  else if $\exists$cell $\in \text{Cells}_{\mathcal{X}_u}$ s.t. $\inf_{\mathbf{x} \in \text{cell}} V_\nu(\mathbf{x}) < \frac{1}{1-p}$ **then**
    $C_{\text{unsafe}} \leftarrow C_{\text{unsafe}} \cup (\{\text{vertices of cell}\} \cap \mathcal{X}_u)$
  else
    Return Reach-avoid guarantee with probability p
  end if
  $V_\nu, \pi_\theta, \leftarrow$ trained by minimizing the loss function
  $\tilde{\mathcal{X}} \leftarrow$ refined discretization
end while
Return Unknown

V_ν are all continuous, hence due to $\mathcal{X}$ being compact f, π_θ and V_ν are also Lipschitz continuous. This will allow us to verify that the Expected decrease condition is satisfied by checking a slightly stricter condition only at the vertices of the discretization grid. The initial discretization $\tilde{\mathcal{X}}$ is also used to initialize counterexample sets used by the learner. In particular, the learner initializes three sets $C_{\text{init}} = \tilde{\mathcal{X}} \cap \mathcal{X}_0$, $C_{\text{unsafe}} = \tilde{\mathcal{X}} \cap \mathcal{X}_u$ and $C_{\text{decrease}} = \tilde{\mathcal{X}} \cap (\mathcal{X} \backslash \mathcal{X}_t)$. These sets will later be extended by counterexamples computed by the verifier. Conversely, the discretization used by the verifier for checking the defining properties of RASMs will at each iteration of the loop be refined by a discretization with a smaller mesh, in order to relax the conditions that are checked by the verifier.

Verifier We now describe the verifier module of our algorithm. Suppose that the learner has learned a policy π_θ and an RASM candidate V_ν. Since V_ν is a neural network, we know that it is a continuous function. Furthermore, we design the learner to apply a softplus activation function to the output layer of V_ν, which ensures that the Nonnegativity condition of RASMs is satisfied by default. Thus, the verifier only needs to check the Initial, Safety and Expected decrease conditions in Definition 1.

Let L_f, L_π and L_V be the Lipschitz constants of f, π_θ and V_ν, respectively. We assume that a Lipschitz constant

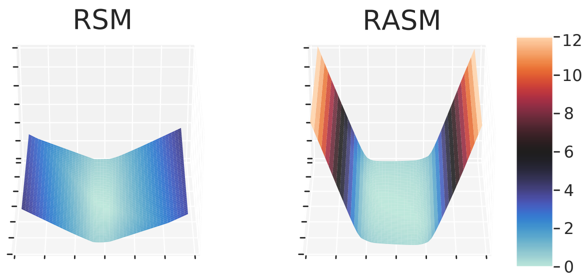

Figure 1: Visualization of a neural network RSM (Lechner et al. 2022) and our RASM on the inverted pendulum task. The RASM provides better probability bounds of reaching the unsafe states.

for the dynamics function f is provided, and use the method of (Szegedy et al. 2014) to compute Lipschitz constants of neural networks π_θ and V_ν. To verify the Expected decrease condition, the verifier collects the superset $\tilde{\mathcal{X}}_e$ of discretization points whose adjacent grid cells contain a non-target state and over which V_ν attains a value that is smaller than $\frac{1}{1-p}$. This set is computed by first collecting all cells that intersect $\mathcal{X}\backslash\mathcal{X}_t$, then using interval arithmetic abstract interpretation (IA-AI) (Cousot and Cousot 1977; Gowal et al. 2018) which propagates interval bounds across neural network layers in order to bound from below the minimal value that V_ν attains over each collected cell, and finally collecting vertices of all cells at which this lower bound is less than $\frac{1}{1-p}$. The verifier then checks a stricter condition for each state $\tilde{\mathbf{x}} \in \tilde{\mathcal{X}}_e$:

$$\mathbb{E}_{\omega\sim d}\Big[V_\nu\Big(f(\tilde{\mathbf{x}},\pi_\theta(\tilde{\mathbf{x}}),\omega)\Big)\Big] < V_\nu(\tilde{\mathbf{x}}) - \tau\cdot K, \qquad (1)$$

where $K = L_V \cdot (L_f \cdot (L_\pi + 1) + 1)$. The expected value in eq. (1) is also bounded from above via IA-AI, where one partitions the support of d into intervals, propagates intervals and multiplies each interval bound by its probability weight in order to bound the expected value of a neural network function over a probability distribution. Due to space restrictions, we provide more details on expected value computation in the extended version of the paper (Žikelić et al. 2022) and note that this method requires that the probability distribution d either has bounded support or is a product of independent univariate distributions.

In order to verify the Initial condition, the verifier collects the set $\text{Cells}_{\mathcal{X}_0}$ of all cells of the discretization grid that intersect the initial set $\mathcal{X}_0$. Then, for each cell $\in \text{Cells}_{\mathcal{X}_0}$, it checks whether

$$\sup_{\mathbf{x}\,\in\,\text{cell}} V_\nu(\mathbf{x}) > 1, \qquad (2)$$

where the supremum of V_ν over the cell is bounded from above by using IA-AI. Similarly, to verify the Unsafe condition, the verifier collects the set $\text{Cells}_{\mathcal{X}_u}$ of all cells of the discretization grid that intersect the unsafe set $\mathcal{X}_u$. Then, for each cell $\in \text{Cells}_{\mathcal{X}_u}$, it uses IA-AI to check whether

$$\inf_{\mathbf{x}\,\in\,\text{cell}} V_\nu(\mathbf{x}) < \frac{1}{1-p}. \qquad (3)$$

If the verifier shows that V_ν satisfies eq. (1) for each $\tilde{\mathbf{x}} \in \tilde{\mathcal{X}}_e$, eq. (2) for each cell $\in \text{Cells}_{\mathcal{X}_0}$ and eq. (3) for each cell $\in \text{Cells}_{\mathcal{X}_u}$, it concludes that V_ν is an RASM. Otherwise, if a counterexample $\tilde{\mathbf{x}}$ to eq. (1) is found and we have $\tilde{\mathbf{x}} \in \mathcal{X}\backslash\mathcal{X}_t$ and $V_\nu(\mathbf{x}) < \frac{1}{1-p}$, it is added to C_{decrease}. Similarly, if counterexample cells to eq. (2) and eq. (3) are found, all their vertices that are contained in $\mathcal{X}_0$ and $\mathcal{X}_u$ are added to C_{init} and C_{unsafe}, respectively.

The following theorem shows that checking the above conditions is sufficient to formally verify whether an RASM candidate is indeed an RASM. The proof follows by exploiting the fact that f, π_θ and V_ν are all Lipschitz continuous and that $\mathcal{X}$ is compact, and we include it in the extended version of the paper (Žikelić et al. 2022).

Theorem 2. *Suppose that the verifier verifies that V_ν satisfies eq. (1) for each $\tilde{\mathbf{x}} \in \tilde{\mathcal{X}}_e$, eq. (2) for each cell $\in$ Cells$_{\mathcal{X}_0}$ and eq. (3) for each cell $\in$ Cells$_{\mathcal{X}_u}$. Then the function V_ν is an RASM for the system with respect to $\mathcal{X}_t$, $\mathcal{X}_u$ and p.*

Learner A policy and an RASM candidate are learned by minimizing the loss function

$$\begin{aligned}\mathcal{L}(\theta,\nu) =& \mathcal{L}_{\text{Init}}(\nu) + \mathcal{L}_{\text{Unsafe}}(\nu) + \mathcal{L}_{\text{Decrease}}(\theta,\nu)\\ &+ \lambda\cdot(\mathcal{L}_{\text{Lipschitz}}(\theta) + \mathcal{L}_{\text{Lipschitz}}(\nu)).\end{aligned}$$

The first three loss terms are used to guide the learner towards learning a true RASM by forcing the learned candidate towards satisfying the Initial, Safety and Expected decrease conditions in Definition 1. They are defined as follows:

$$\mathcal{L}_{\text{Init}}(\nu) = \max_{\mathbf{x}\in C_{\text{init}}}\{V_\nu(\mathbf{x}) - 1, 0\}$$

$$\mathcal{L}_{\text{Unsafe}}(\nu) = \max_{\mathbf{x}\in C_{\text{unsafe}}}\{\frac{1}{1-p} - V_\nu(\mathbf{x}), 0\}$$

$$\begin{aligned}\mathcal{L}_{\text{Decrease}}(\theta,\nu) &= \frac{1}{|C_{\text{decrease}}|}\cdot\\ \sum_{\mathbf{x}\in C_{\text{decrease}}} &\Big(\max\Big\{\sum_{\omega_1,\dots,\omega_N\sim\mathcal{N}}\frac{V_\nu(f(\mathbf{x},\pi_\theta(\mathbf{x}),\omega_i))}{N}\\ &\quad - V_\theta(\mathbf{x}) + \tau\cdot K, 0\Big\}\Big)\end{aligned}$$

Each loss term is designed to incur a loss at a state whenever that state violates the corresponding condition in Definition 1 that needs to be checked by the verifier. In the expression for $\mathcal{L}_{\text{Decrease}}(\theta,\nu)$, we approximate the expected value of V_ν by taking the mean value of V_ν at N sampled successor states, where $N \in \mathbb{N}$ is an algorithm parameter. This is necessary as it is not possible to compute a closed form expression for the expected value of a neural network V_ν.

The last loss term $\lambda\cdot(\mathcal{L}_{\text{Lipschitz}}(\theta) + \mathcal{L}_{\text{Lipschitz}}(\nu))$ is the regularization term used to guide the learner towards a policy and an RASM candidate with Lipschitz constants below a tolerable threshold ρ, with $\lambda > 0$ being a regularization constant. By preferring networks with small Lipschitz constants, we allow the verifier to use a wider mesh, which significantly speeds up the verification process. The regularization term for π_θ (and analogously for V_ν) is defined via

$$\mathcal{L}_{\text{Lipschitz}}(\theta) = \max\Big\{\prod_{W,b\in\theta}\max_j\sum_i|W_{i,j}|-\rho, 0\Big\},$$

Environment	RSM (reach-avoid extension)	RASM (ours)
2D system	83.4%	93.3%
Inverted pendulum	47.9%	92.1%
Collision avoidance	Fail	90.4%

Table 1: Reach-avoid probability obtained by our method and by the naive extension of RSMs. In each case, we report the largest probability successfully verified by the method.

where W and b weight matrices and bias vectors for each layer in π_θ. Finally, in our implementation we also add an auxiliary loss term that does not enforce any of the defining conditions of RASMs, however it is used to guide the learner towards a candidate that attains the global minimum in a state that is contained within the target set $\mathcal{X}_t$. We empirically observed that this term sometimes helps the updated policy from diverging from its objective to stabilize the system. Due to space restrictions, details are provided in the extended version of the paper (Žikelić et al. 2022).

We remark that the loss function is always nonnegative but is not necessarily equal to 0 even if V_ν satisfies all conditions checked by the verifier and if Lipschitz constants are below the specified thresholds. This is because the expected values in $\mathcal{L}_{\text{Decrease}}(\theta,\nu)$ are approximated via sample means. However, in the following theorem we show that in this case $\mathcal{L}(\theta,\nu) \to 0$ with probability 1 as we add independent samples. The claim follows from the Strong Law of Large Numbers and the proof can be found in the extended version of the paper (Žikelić et al. 2022).

Theorem 3. *Let N be the number of samples used to approximate expected values in $\mathcal{L}_{Decrease}(\theta,\nu)$. Suppose that V_ν satisfies eq.* (1) *for each $\tilde{\mathbf{x}} \in \tilde{\mathcal{X}}_e$, eq.* (2) *for each cell $\in$ Cells$_{\mathcal{X}_0}$ and eq.* (3) *for each cell $\in$ Cells$_{\mathcal{X}_u}$. Suppose that Lipschitz constants of π_θ and V_ν are below the thresholds specified by $\mathcal{L}_{Lipschitz}(\theta)$ and $\mathcal{L}_{Lipschitz}(\nu)$ and that the samples in $\mathcal{L}_{Decrease}(\theta,\nu)$ are independent. Then $\lim_{N\to\infty}\mathcal{L}(\theta,\nu) = 0$ with probability* 1.

Experiments

We experimentally validate our method on 3 non-linear RL environments. Since no available baseline provides reach-avoid guarantees of stochastic systems over the infinite time horizon, as well as sampling and discretization approaches can only reason over finite time horizons, we aim our experiment as a validation of algorithm 1 in practice [1].

Our first two environments are a linear 2D system with non-linear control bounds and the stochastic inverted pendulum control problem. The linear 2D system is of the form $\mathbf{x}_{t+1} = A\mathbf{x}_t + Bg(\mathbf{u}_t) + \omega_t$, where $g : u \mapsto \min(\max(u, -1, 1))$ limits the admissible action of the policy and ω_t is sampled from a triangular noise distribution. The inverted pendulum environment is taken from the OpenAI Gym (Brockman et al. 2016) and made more difficult by adding noise perturbations to its state. Our third environment concerns a collision avoidance task. The objective of this environment is to navigate an agent to the target region while avoiding crashing into one of two obstacles. Further details on all environments an be found in the extended version (Žikelić et al. 2022).

The policy and RASM networks consist of two hidden layers (128 units each, ReLU). The RASM network has a single output unit with a softplus activation. We run our algorithm with a timeout of 3 hours.

The goal of our first experiment is to empirically evaluate the ability of our approach to learn probabilistic reach-avoid policies and to understand the importance of combining reachability with level set reasoning towards safety in stochastic systems. For all tasks, we pre-train the policy networks using 100 iterations of PPO. To evaluate our approach, we run our algorithm with several probability thresholds and report the highest threshold for which a policy together with an RASM is successfully learned. In order to understand the importance of simultaneous reasoning about reachability and level sets, we then compare our approach with a much simpler extension of the method of (Lechner et al. 2022) which learns RSMs to certify probability 1 reachability but does not consider any form of safety specifications. In particular, we run the method of (Lechner et al. 2022) without the safety constraint and, in case a valid RSM is found, we normalize the function such that the Nonnegativity and the Initial conditions of RASMs are satisfied. We then bound from below the smallest value that the RSM attains over the unsafe region, and extract the corresponding reach-avoid probability bound according to the Safety condition of RASMs. Note that, even though this extension also exploits the ideas behind the level set reasoning in our RASMs, it *first* performs reachability analysis and only *afterwards* considers safety. We remark that there is no existing method that provides reach-avoid guarantees of stochastic systems over the infinite time horizon, i.e. there is no existing baseline to compare against, thus we compare our level set reasoning with the extension of (Lechner et al. 2022) which is the closest related work.

Table 1 shows results of our first experiment. In particular, in the third column we see that our method successfully learns policies that provide high probability reach-avoid guarantees for all benchmarks. On the other hand, comparison to the second column shows that *simultaneous* reasoning about reachability and safety that is allowed by our RASMs provides significantly better probabilistic reach-avoid guarantees than when such reasoning is decoupled. Figure 1 visualizes the RSM computed by the baseline and our RASM.

In our second experiment, we study how well our algorithm can *repair* (or *fine-tune*) an unsafe policy. In particular, we pre-train the policy network using only 20 PPO iterations. We then run our algorithm with fixed policy parameters θ, i.e. we only learn an RASM in order to verify a probabilistic reach-avoid guarantee provided by a pre-trained policy. Next, we run our Algorithm 1 with both ν and θ as trainable parameters. Table 2 shows that, compared to a standalone verification method, our algorithm is able to repair unsafe policies in practice. However, the inability to repair the inverted pendulum policy illustrates that a decent starting policy

[1] Code is available at https://github.com/mlech26l/neural_martingales

	V_ν	V_ν and π_θ
2D system	Fail (10 iters.)	96.7% (4 iters.)
Collision avoidance	Fail (9 iters.)	80.9% (3 iters.)
Inverted pendulum	Fail (7 iters.)	Fail (7 iters.)

Table 2: Reach-avoid probabilities obtained by repairing unsafe policies. Verifying a policy by only learning the RASM V_ν times out, while jointly optimizing V_ν and π_θ yields a valid RASM. In each case, we report the largest reach-avoid probability successfully verified by the respective method.

is necessary for our algorithm, emphasizing the importance of policy initialization. Since the Policy Initialization step in Algorithm 1 initialises the policy by using PPO with a reward function that encodes the reach-avoid specification, our second experiment also demonstrates that a policy initialised by using RL on a tailored reward function is not sufficient to learn a reach-avoid policy with guarantees and that the learned policy requires "correction" in order to provide reach-avoid guarantees. The "correction" is achieved precisely by keeping the policy parameters trainable in the learner-verifier framework and fine-tuning them.

Conclusion

In this work, we present a method for learning controllers for discrete-time stochastic non-linear dynamical systems with formal reach-avoid guarantees. Our method learns a policy together with a reach-avoid supermartingale (RASM), a novel notion that we introduce in this work. It solves several important problems, including control with reach-avoid guarantees, verification of reach-avoid properties for a fixed policy, or fine-tuning of a given policy that does not satisfy a reach-avoid property. We demonstrated the effectiveness of our approach on three RL benchmarks. An interesting future direction would be to study certified control and verification of more general properties in stochastic systems. Since the aim of AI safety and formal verification is to ensure that systems do not behave in undesirable ways and that safety violating events are avoided, we are not aware of any potential negative societal impacts of our work.

Acknowledgments

This work was supported in part by the ERC-2020-AdG 101020093, ERC CoG 863818 (FoRM-SMArt) and the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie Grant Agreement No. 665385.

References

Abate, A.; Ahmed, D.; Giacobbe, M.; and Peruffo, A. 2021. Formal Synthesis of Lyapunov Neural Networks. *IEEE Control. Syst. Lett.*, 5(3): 773–778.

Abate, A.; Giacobbe, M.; and Roy, D. 2021. Learning Probabilistic Termination Proofs. In Silva, A.; and Leino, K. R. M., eds., *Computer Aided Verification - 33rd International Conference, CAV 2021, Virtual Event, July 20-23, 2021, Proceedings, Part II*, volume 12760 of *Lecture Notes in Computer Science*, 3–26. Springer.

Achiam, J.; Held, D.; Tamar, A.; and Abbeel, P. 2017. Constrained policy optimization. In *International Conference on Machine Learning*, 22–31. PMLR.

Alshiekh, M.; Bloem, R.; Ehlers, R.; Könighofer, B.; Niekum, S.; and Topcu, U. 2018. Safe Reinforcement Learning via Shielding. In McIlraith, S. A.; and Weinberger, K. Q., eds., *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, 2669–2678. AAAI Press.

Altman, E. 1999. *Constrained Markov decision processes*, volume 7. CRC Press.

Ames, A. D.; Coogan, S.; Egerstedt, M.; Notomista, G.; Sreenath, K.; and Tabuada, P. 2019. Control Barrier Functions: Theory and Applications. In *17th European Control Conference, ECC 2019, Naples, Italy, June 25-28, 2019*, 3420–3431. IEEE.

Amodei, D.; Olah, C.; Steinhardt, J.; Christiano, P. F.; Schulman, J.; and Mané, D. 2016. Concrete Problems in AI Safety. *CoRR*, abs/1606.06565.

Badings, T. S.; Romao, L.; Abate, A.; and Jansen, N. 2022. Probabilities Are Not Enough: Formal Controller Synthesis for Stochastic Dynamical Models with Epistemic Uncertainty. *CoRR*, abs/2210.05989.

Bastani, O.; and Li, S. 2021. Safe Reinforcement Learning via Statistical Model Predictive Shielding. In Shell, D. A.; Toussaint, M.; and Hsieh, M. A., eds., *Robotics: Science and Systems XVII, Virtual Event, July 12-16, 2021.*

Berkenkamp, F. 2019. Safe Exploration in Reinforcement Learning: Theory and Applications in Robotics. *Ph.D. thesis, ETH Zurich.*

Brockman, G.; Cheung, V.; Pettersson, L.; Schneider, J.; Schulman, J.; Tang, J.; and Zaremba, W. 2016. OpenAI Gym. *arXiv preprint arXiv:1606.01540*.

Cauchi, N.; and Abate, A. 2019. StocHy-automated verification and synthesis of stochastic processes. In *Proceedings of the 22nd ACM International Conference on Hybrid Systems: Computation and Control*, 258–259.

Chakarov, A.; and Sankaranarayanan, S. 2013. Probabilistic Program Analysis with Martingales. In Sharygina, N.; and Veith, H., eds., *Computer Aided Verification - 25th International Conference, CAV 2013, Saint Petersburg, Russia, July 13-19, 2013. Proceedings*, volume 8044 of *Lecture Notes in Computer Science*, 511–526. Springer.

Chang, Y.; and Gao, S. 2021. Stabilizing Neural Control Using Self-Learned Almost Lyapunov Critics. In *IEEE International Conference on Robotics and Automation, ICRA 2021, Xi'an, China, May 30 - June 5, 2021*, 1803–1809. IEEE.

Chang, Y.; Roohi, N.; and Gao, S. 2019. Neural Lyapunov Control. In Wallach, H. M.; Larochelle, H.; Beygelzimer, A.; d'Alché-Buc, F.; Fox, E. B.; and Garnett, R., eds., *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems*

2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, 3240–3249.

Chatterjee, K.; Goharshady, A. K.; Meggendorfer, T.; and Zikelic, D. 2022. Sound and Complete Certificates for Quantitative Termination Analysis of Probabilistic Programs. In Shoham, S.; and Vizel, Y., eds., *Computer Aided Verification - 34th International Conference, CAV 2022, Haifa, Israel, August 7-10, 2022, Proceedings, Part I*, volume 13371 of *Lecture Notes in Computer Science*, 55–78. Springer.

Chatterjee, K.; Novotný, P.; and Zikelic, D. 2017. Stochastic invariants for probabilistic termination. In Castagna, G.; and Gordon, A. D., eds., *Proceedings of the 44th ACM SIGPLAN Symposium on Principles of Programming Languages, POPL 2017, Paris, France, January 18-20, 2017*, 145–160. ACM.

Chow, Y.; Nachum, O.; Duéñez-Guzmán, E. A.; and Ghavamzadeh, M. 2018. A Lyapunov-based Approach to Safe Reinforcement Learning. In Bengio, S.; Wallach, H. M.; Larochelle, H.; Grauman, K.; Cesa-Bianchi, N.; and Garnett, R., eds., *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, 8103–8112.

Cousot, P.; and Cousot, R. 1977. Abstract Interpretation: A Unified Lattice Model for Static Analysis of Programs by Construction or Approximation of Fixpoints. In Graham, R. M.; Harrison, M. A.; and Sethi, R., eds., *Conference Record of the Fourth ACM Symposium on Principles of Programming Languages, Los Angeles, California, USA, January 1977*, 238–252. ACM.

Crespo, L. G.; and Sun, J. 2003. Stochastic optimal control via Bellman's principle. *Autom.*, 39(12): 2109–2114.

Dalal, G.; Dvijotham, K.; Vecerík, M.; Hester, T.; Paduraru, C.; and Tassa, Y. 2018. Safe Exploration in Continuous Action Spaces. *ArXiv*, abs/1801.08757.

Elsayed-Aly, I.; Bharadwaj, S.; Amato, C.; Ehlers, R.; Topcu, U.; and Feng, L. 2021. Safe Multi-Agent Reinforcement Learning via Shielding. In Dignum, F.; Lomuscio, A.; Endriss, U.; and Nowé, A., eds., *AAMAS '21: 20th International Conference on Autonomous Agents and Multiagent Systems, Virtual Event, United Kingdom, May 3-7, 2021*, 483–491. ACM.

García, J.; and Fernández, F. 2015. A comprehensive survey on safe reinforcement learning. *J. Mach. Learn. Res.*, 16: 1437–1480.

Geibel, P. 2006. Reinforcement Learning for MDPs with Constraints. In Fürnkranz, J.; Scheffer, T.; and Spiliopoulou, M., eds., *Machine Learning: ECML 2006, 17th European Conference on Machine Learning, Berlin, Germany, September 18-22, 2006, Proceedings*, volume 4212 of *Lecture Notes in Computer Science*, 646–653. Springer.

Giacobbe, M.; Hasanbeig, M.; Kroening, D.; and Wijk, H. 2021. Shielding Atari Games with Bounded Prescience. In Dignum, F.; Lomuscio, A.; Endriss, U.; and Nowé, A., eds., *AAMAS '21: 20th International Conference on Autonomous Agents and Multiagent Systems, Virtual Event, United Kingdom, May 3-7, 2021*, 1507–1509. ACM.

Gowal, S.; Dvijotham, K.; Stanforth, R.; Bunel, R.; Qin, C.; Uesato, J.; Arandjelovic, R.; Mann, T. A.; and Kohli, P. 2018. On the Effectiveness of Interval Bound Propagation for Training Verifiably Robust Models. *CoRR*, abs/1810.12715.

Haddad, W. M.; and Chellaboina, V. 2011. *Nonlinear dynamical systems and control*. Princeton university press.

Hasanbeig, M.; Abate, A.; and Kroening, D. 2020. Cautious Reinforcement Learning with Logical Constraints. In *Proceedings of the 19th International Conference on Autonomous Agents and Multiagent Systems, AAMAS '20, Auckland, New Zealand, May 9-13, 2020*, 483–491. International Foundation for Autonomous Agents and Multiagent Systems.

Henrion, D.; and Garulli, A. 2005. *Positive polynomials in control*, volume 312. Springer Science & Business Media.

Jarvis-Wloszek, Z.; Feeley, R.; Tan, W.; Sun, K.; and Packard, A. 2003. Some controls applications of sum of squares programming. In *42nd IEEE international conference on decision and control (IEEE Cat. No. 03CH37475)*, volume 5, 4676–4681. IEEE.

Jin, W.; Wang, Z.; Yang, Z.; and Mou, S. 2020. Neural Certificates for Safe Control Policies. *CoRR*, abs/2006.08465.

Koller, T.; Berkenkamp, F.; Turchetta, M.; and Krause, A. 2018. Learning-Based Model Predictive Control for Safe Exploration. *2018 IEEE Conference on Decision and Control (CDC)*, 6059–6066.

Lavaei, A.; Khaled, M.; Soudjani, S.; and Zamani, M. 2020. AMYTISS: Parallelized Automated Controller Synthesis for Large-Scale Stochastic Systems. In Lahiri, S. K.; and Wang, C., eds., *Computer Aided Verification - 32nd International Conference, CAV 2020, Los Angeles, CA, USA, July 21-24, 2020, Proceedings, Part II*, volume 12225 of *Lecture Notes in Computer Science*, 461–474. Springer.

Lawrence, N. P.; Loewen, P. D.; Forbes, M. G.; Backström, J. U.; and Gopaluni, R. B. 2020. Almost Surely Stable Deep Dynamics. In Larochelle, H.; Ranzato, M.; Hadsell, R.; Balcan, M.; and Lin, H., eds., *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Lechner, M.; Zikelic, D.; Chatterjee, K.; and Henzinger, T. A. 2021. Infinite Time Horizon Safety of Bayesian Neural Networks. In Ranzato, M.; Beygelzimer, A.; Dauphin, Y. N.; Liang, P.; and Vaughan, J. W., eds., *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, 10171–10185.

Lechner, M.; Zikelic, D.; Chatterjee, K.; and Henzinger, T. A. 2022. Stability Verification in Stochastic Control Systems via Neural Network Supermartingales. In *Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022, Thirty-Fourth Conference on Innovative Applications of Artificial Intelligence, IAAI 2022, The Twelveth Symposium on Educational Advances in Artificial Intelligence, EAAI 2022 Virtual Event, February 22 - March 1, 2022*, 7326–7336. AAAI Press.

Li, S.; and Bastani, O. 2020. Robust Model Predictive Shielding for Safe Reinforcement Learning with Stochastic Dynamics. In *2020 IEEE International Conference on Robotics and*

Automation, ICRA 2020, Paris, France, May 31 - August 31, 2020, 7166–7172. IEEE.

Liu, A.; Shi, G.; Chung, S.-J.; Anandkumar, A.; and Yue, Y. 2020. Robust Regression for Safe Exploration in Control. In *L4DC*.

Mathiesen, F. B.; Calvert, S. C.; and Laurenti, L. 2023. Safety Certification for Stochastic Systems via Neural Barrier Functions. *IEEE Control. Syst. Lett.*, 7: 973–978.

Mazouz, R.; Muvvala, K.; Babu, A. R.; Laurenti, L.; and Lahijanian, M. 2022. Safety Guarantees for Neural Network Dynamic Systems via Stochastic Barrier Functions. In Oh, A. H.; Agarwal, A.; Belgrave, D.; and Cho, K., eds., *Advances in Neural Information Processing Systems*.

Michalska, H.; and Mayne, D. Q. 1993. Robust receding horizon control of constrained nonlinear systems. *IEEE Trans. Autom. Control.*, 38(11): 1623–1633.

Parrilo, P. A. 2000. *Structured semidefinite programs and semialgebraic geometry methods in robustness and optimization*. California Institute of Technology.

Perkins, T. J.; and Barto, A. G. 2002. Lyapunov Design for Safe Reinforcement Learning. *J. Mach. Learn. Res.*, 3: 803–832.

Prajna, S.; Jadbabaie, A.; and Pappas, G. J. 2007. A Framework for Worst-Case and Stochastic Safety Verification Using Barrier Certificates. *IEEE Trans. Autom. Control.*, 52(8): 1415–1428.

Puterman, M. L. 1994. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. Wiley Series in Probability and Statistics. Wiley. ISBN 978-0-47161977-2.

Qin, Z.; Zhang, K.; Chen, Y.; Chen, J.; and Fan, C. 2021. Learning Safe Multi-agent Control with Decentralized Neural Barrier Certificates. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Richards, S. M.; Berkenkamp, F.; and Krause, A. 2018. The Lyapunov Neural Network: Adaptive Stability Certification for Safe Learning of Dynamical Systems. In *2nd Annual Conference on Robot Learning, CoRL 2018, Zürich, Switzerland, 29-31 October 2018, Proceedings*, volume 87 of *Proceedings of Machine Learning Research*, 466–476. PMLR.

Santoyo, C.; Dutreix, M.; and Coogan, S. 2021. A barrier function approach to finite-time stochastic system verification and control. *Autom.*, 125: 109439.

Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; and Klimov, O. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Soudjani, S. E. Z.; Gevaerts, C.; and Abate, A. 2015. FAUST ^{2} : Formal Abstractions of Uncountable-STate STochastic Processes. In *Tools and Algorithms for the Construction and Analysis of Systems - 21st International Conference, TACAS 2015, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2015, London, UK, April 11-18, 2015. Proceedings*, volume 9035 of *Lecture Notes in Computer Science*, 272–286. Springer.

Steinhardt, J.; and Tedrake, R. 2012. Finite-time regional verification of stochastic non-linear systems. *Int. J. Robotics Res.*, 31(7): 901–923.

Summers, S.; and Lygeros, J. 2010. Verification of discrete time stochastic hybrid systems: A stochastic reach-avoid decision problem. *Autom.*, 46(12): 1951–1961.

Sun, D.; Jha, S.; and Fan, C. 2020. Learning Certified Control Using Contraction Metric. In *4th Conference on Robot Learning, CoRL 2020, 16-18 November 2020, Virtual Event / Cambridge, MA, USA*, volume 155 of *Proceedings of Machine Learning Research*, 1519–1539. PMLR.

Sutton, R. S.; and Barto, A. G. 2018. *Reinforcement learning: An introduction*. MIT press.

Szegedy, C.; Zaremba, W.; Sutskever, I.; Bruna, J.; Erhan, D.; Goodfellow, I. J.; and Fergus, R. 2014. Intriguing properties of neural networks. In *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*.

Turchetta, M.; Berkenkamp, F.; and Krause, A. 2019. Safe Exploration for Interactive Machine Learning. In *NeurIPS*.

Uchibe, E.; and Doya, K. 2007. Constrained reinforcement learning from intrinsic and extrinsic rewards. In *2007 IEEE 6th International Conference on Development and Learning*, 163–168. IEEE.

Umlauft, J.; and Hirche, S. 2017. Learning Stable Stochastic Nonlinear Dynamical Systems. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, 3502–3510. PMLR.

Vahidi, A.; and Eskandarian, A. 2003. Research advances in intelligent collision avoidance and adaptive cruise control. *IEEE Trans. Intell. Transp. Syst.*, 4(3): 143–153.

Vaidya, U. 2015. Stochastic stability analysis of discrete-time system using Lyapunov measure. In *American Control Conference, ACC 2015, Chicago, IL, USA, July 1-3, 2015*, 4646–4651. IEEE.

Vinod, A. P.; Gleason, J. D.; and Oishi, M. M. K. 2019. SReachTools: a MATLAB stochastic reachability toolbox. In *Proceedings of the 22nd ACM International Conference on Hybrid Systems: Computation and Control, HSCC 2019, Montreal, QC, Canada, April 16-18, 2019*, 33–38. ACM.

Wabersich, K. P.; and Zeilinger, M. N. 2018. Linear Model Predictive Safety Certification for Learning-Based Control. In *57th IEEE Conference on Decision and Control, CDC 2018, Miami, FL, USA, December 17-19, 2018*, 7130–7135. IEEE.

Williams, D. 1991. *Probability with Martingales*. Cambridge mathematical textbooks. Cambridge University Press. ISBN 978-0-521-40605-5.

Xue, B.; Li, R.; Zhan, N.; and Fränzle, M. 2021. Reach-avoid Analysis for Stochastic Discrete-time Systems. In *2021 American Control Conference, ACC 2021, New Orleans, LA, USA, May 25-28, 2021*, 4879–4885. IEEE.

Žikelić, Đ.; Lechner, M.; Henzinger, T. A.; and Chatterjee, K. 2022. Learning Control Policies for Stochastic Systems with Reach-avoid Guarantees. *arXiv preprint arXiv:2210.05308*.