Safe Model-based Reinforcement Learning with Stability Guarantees

Felix Berkenkamp  
Department of Computer Science  
ETH Zurich  
befelix@inf.ethz.ch

Matteo Turchetta  
Department of Computer Science,  
ETH Zurich  
matteotu@inf.ethz.ch

Angela P. Schoellig  
Institute for Aerospace Studies  
University of Toronto  
schoellig@utias.utoronto.ca

Andreas Krause  
Department of Computer Science  
ETH Zurich  
krausea@ethz.ch

Abstract

Reinforcement learning is a powerful paradigm for learning optimal policies from experimental data. However, to find optimal policies, most reinforcement learning algorithms explore all possible actions, which may be harmful for real-world systems. As a consequence, learning algorithms are rarely applied on safety-critical systems in the real world. In this paper, we present a learning algorithm that explicitly considers safety, defined in terms of stability guarantees. Specifically, we extend control-theoretic results on Lyapunov stability verification and show how to use statistical models of the dynamics to obtain high-performance control policies with provable stability certificates. Moreover, under additional regularity assumptions in terms of a Gaussian process prior, we prove that one can effectively and safely collect data in order to learn about the dynamics and thus both improve control performance and expand the safe region of the state space. In our experiments, we show how the resulting algorithm can safely optimize a neural network policy on a simulated inverted pendulum, without the pendulum ever falling down.

1 Introduction

While reinforcement learning (RL, [1]) algorithms have achieved impressive results in games, for example on the Atari platform [2], they are rarely applied to real-world physical systems (e.g., robots) outside of academia. The main reason is that RL algorithms provide optimal policies only in the long-term, so that intermediate policies may be unsafe, break the system, or harm their environment. This is especially true in safety-critical systems that can affect human lives. Despite this, safety in RL has remained largely an open problem [3].

Consider, for example, a self-driving car. While it is desirable for the algorithm that drives the car to improve over time (e.g., by adapting to driver preferences and changing environments), any policy applied to the system has to guarantee safe driving. Thus, it is not possible to learn about the system through random exploratory actions, which almost certainly lead to a crash. In order to avoid this problem, the learning algorithm needs to consider its ability to safely recover from exploratory actions. In particular, we want the car to be able to recover to a safe state, for example, driving at a reasonable speed in the middle of the lane. This ability to recover is known as asymptotic stability in control theory [4]. Specifically, we care about the region of attraction of the closed-loop system under a policy. This is a subset of the state space that is forward invariant so that any state trajectory that starts within this set stays within it for all times and converges to a goal state eventually.
In this paper, we present a RL algorithm for continuous state-action spaces that provides these kind of high-probability safety guarantees for policies. In particular, we show how, starting from an initial, safe policy we can expand our estimate of the region of attraction by collecting data inside the safe region and adapt the policy to both increase the region of attraction and improve control performance.

**Related work.** Safety is an active research topic in RL and different definitions of safety exist [5, 6]. Discrete Markov decision processes (MDPs) are one class of tractable models that have been analyzed. In risk-sensitive RL, one specifies risk-aversion in the reward [7]. For example, [8] define risk as the probability of driving the agent to a set of known, undesirable states. Similarly, robust MDPs maximize rewards when transition probabilities are uncertain [9, 10]. Both [11] and [12] introduce algorithms to safely explore MDPs so that the agent never gets stuck without safe actions. All these methods require an accurate probabilistic model of the system.

In continuous state-action spaces, model-free policy search algorithms have been successful. These update policies without a system model by repeatedly executing the same task [13]. In this setting, [14] introduces safety guarantees in terms of constraint satisfaction that hold in expectation. High-probability worst-case safety guarantees are available for methods based on Bayesian optimization [15] together with Gaussian process models (GP, [16]) of the cost function. The algorithms in [17] and [18] provide high-probability safety guarantees for any parameter that is evaluated on the real system. These methods are used in [19] to safely optimize a parametric control policy on a quadrotor. However, resulting policies are task-specific and require the system to be reset.

In the model-based RL setting, research has focused on safety in terms of state constraints. In [20, 21], a priori known, safe global backup policies are used, while [22] learns to switch between several safe policies. However, it is not clear how one may find these policies in the first place. Other approaches use model predictive control with constraints, a model-based technique where the control actions are optimized online. For example, [23] models uncertain environmental constraints, while [24] uses approximate uncertainty propagation of GP dynamics along trajectories. In this setting, robust feasibility and constraint satisfaction can be guaranteed for a learned model with bounded errors using robust model predictive control [25]. The method in [26] uses reachability analysis to construct safe regions in the state space. The theoretical guarantees depend on the solution to a partial differential equation, which is approximated.

Theoretical guarantees for the stability exist for the more tractable stability analysis and verification under a fixed control policy. In control, stability of a known system can be verified using a Lyapunov function [27]. A similar approach is used by [28] for deterministic, but unknown dynamics that are modeled as a GP, which allows for provably safe learning of regions of attraction for fixed policies. Similar results are shown in [29] for stochastic systems that are modeled as a GP. They use Bayesian quadrature to compute provably accurate estimates of the region of attraction. These approaches do not update the policy.

**Our contributions.** We introduce a novel algorithm that can safely optimize policies in continuous state-action spaces while providing high-probability safety guarantees in terms of stability. Moreover, we show that it is possible to exploit the regularity properties of the system in order to safely learn about the dynamics and thus improve the policy and increase the estimated safe region of attraction without ever leaving it. Specifically, starting from a policy that is known to stabilize the system locally, we gather data at informative, safe points and improve the policy safely based on the improved model of the system and prove that any exploration algorithm that gathers data at these points reaches a natural notion of full exploration. We show how the theoretical results transfer to a practical algorithm with safety guarantees and apply it to a simulated inverted pendulum stabilization task.

## 2 Background and Assumptions

We consider a deterministic, discrete-time dynamic system

\[ x_{t+1} = f(x_t, u_t) = h(x_t, u_t) + g(x_t, u_t), \]

with states \( x \in \mathcal{X} \subseteq \mathbb{R}^q \) and control actions \( u \in \mathcal{U} \subseteq \mathbb{R}^p \) and a discrete time index \( t \in \mathbb{N} \). The true dynamics \( f: \mathcal{X} \times \mathcal{U} \rightarrow \mathcal{X} \) consist of two parts: \( h(x_t, u_t) \) is a known, prior model that can be obtained from first principles, while \( g(x_t, u_t) \) represents *a priori* unknown model errors. While the model errors are unknown, we can obtain noisy measurements of \( f(x, u) \) by driving the system to the state \( x \) and taking action \( u \). We want this system to behave in a certain way, e.g., the car driving
on the road. To this end, we need to specify a control policy $\pi : \mathcal{X} \to \mathcal{U}$ that, given the current state, determines the appropriate control action that drives the system to some goal state, which we set as the origin without loss of generality [4]. We encode the performance requirements of how to drive the system to the origin through a positive cost $r(x, u)$ that is associated with states and actions and has $r(0, 0) = 0$. The policy aims to minimize the cumulative, discounted costs for each starting state.

The goal is to safely learn about the dynamics from measurements and adapt the policy for performance, without encountering system failures. Specifically, we define the safety constraint on the state divergence that occurs when leaving the region of attraction. This means that adapting the policy is not allowed to decrease the region of attraction and exploratory actions to learn about the dynamics $f(\cdot)$ are not allowed to drive the system outside the region of attraction. The region of attraction is not known a priori, but is implicitly defined through the system dynamics and the choice of policy. Thus, the policy not only defines performance as in typical RL, but also determines safety and where we can obtain measurements.

**Model assumptions** In general, this kind of safe learning is impossible without further assumptions. For example, in a discontinuous system even a slight change in the control policy can lead to drastically different behavior. Moreover, to expand the safe set we need to generalize learned knowledge about the dynamics to (potentially unsafe) states that we have not visited. To this end, we restrict ourselves to the general and practically relevant class of models that are Lipschitz continuous. This is a typical assumption in the control community [4]. Additionally, to ensure that the closed-loop system remains Lipschitz continuous when the control policy is applied, we restrict policies to the rich class of $L_\pi$-Lipschitz continuous functions $\Pi_L$, which also contains certain types of neural networks [30].

**Assumption 1** (continuity). The dynamics $h(\cdot)$ and $g(\cdot)$ in (1) are $L_h$- and $L_g$ Lipschitz continuous with respect to the 1-norm. The considered control policies $\pi$ lie in a set $\Pi_L$ of functions that are $L_\pi$-Lipschitz continuous with respect to the 1-norm.

To enable safe learning, we require a reliable statistical model. While we commit to GPs for the exploration analysis, for safety any suitable, well-calibrated model is applicable.

**Assumption 2** (well-calibrated model). Let $\mu_n(\cdot)$ and $\Sigma_n(\cdot)$ denote the posterior mean and covariance matrix functions of the statistical model of the dynamics (1) conditioned on $n$ noisy measurements. With $\sigma_n(\cdot) = \text{trace}(\Sigma_n^{1/2}(\cdot))$, there exists a $\beta_n > 0$ such that with probability at least $(1 - \delta)$ it holds for all $n \geq 0$, $x \in \mathcal{X}$, and $u \in \mathcal{U}$ that $\|f(x, u) - \mu_n(x, u)\|_1 \leq \beta_n \sigma_n(x, u)$.

This assumption ensures that we can build confidence intervals on the dynamics that, when scaled by an appropriate constant $\beta_n$, cover the true function with high probability. We introduce a specific statistical model that fulfills both assumptions under certain regularity assumptions in Sec. 3.

**Lyapunov function** To satisfy the specified safety constraints for safe learning, we require a tool to determine whether individual states and actions are safe. In control theory, this safety is defined through the region of attraction, which can be computed for a fixed policy using Lyapunov functions [4]. Lyapunov functions are continuously differentiable functions $v : \mathcal{X} \to \mathbb{R}_{\geq 0}$ with $v(0) = 0$ and $v(x) > 0$ for all $x \in \mathcal{X} \setminus \{0\}$. The key idea behind using Lyapunov functions to show stability of the system (1) is similar to that of gradient descent on strictly quasiconvex functions: if one can show that, given a policy $\pi$, applying the dynamics $f$ on the state maps it to strictly smaller values on the Lyapunov function (‘going downhill’), then the state eventually converges to the equilibrium point at the origin (minimum). In particular, the assumptions in Theorem 1 below imply that $v$ is strictly quasiconvex within the region of attraction if the dynamics are Lipschitz continuous. As a result, the one step decrease property for all states within a level set guarantees eventual convergence to the origin.

**Theorem 1** ([4]). Let $v$ be a Lyapunov function, $f$ Lipschitz continuous dynamics, and $\pi$ a policy. If $v(f(x, \pi(x))) < v(x)$ for all $x$ within the level set $\mathcal{V}(c) = \{x \in \mathcal{X} \setminus \{0\} \mid v(x) \leq c\}$, $c > 0$, then $\mathcal{V}(c)$ is a region of attraction, so that $x_0 \in \mathcal{V}(c)$ implies $x_t \in \mathcal{V}(c)$ for all $t > 0$ and $\lim_{t \to \infty} x_t = 0$.

It is convenient to characterize the region of attraction through a level set of the Lyapunov function, since it replaces the challenging test for convergence with a one-step decrease condition on the Lyapunov function. For the theoretical analysis in this paper, we assume that a Lyapunov function is given to determine the region of attraction. For ease of notation, we also assume $\partial v(x)/\partial x \neq 0$ for all $x \in \mathcal{X} \setminus \{0\}$, which ensures that level sets $\mathcal{V}(c)$ are connected if $c > 0$. Since Lyapunov functions are continuously differentiable, they are $L_\pi$-Lipschitz continuous over the compact set $\mathcal{X}$.
In general, it is not easy to find suitable Lyapunov functions. However, for physical models, like the prior model $h$ in (1), the energy of the system (e.g., kinetic and potential for mechanical systems) is a good candidate Lyapunov function. Moreover, it has recently been shown that it is possible to compute suitable Lyapunov functions [31, 32]. In our experiments, we exploit the fact that value functions in RL are Lyapunov functions if the costs are strictly positive away from the origin. This follows directly from the definition of the value function, where $v(x) = r(x, \pi(x)) + v(f(x, \pi(x))) \leq v(f(x, \pi(x)))$. Thus, we can obtain Lyapunov candidates as a by-product of approximate dynamic programming.

**Initial safe policy** Lastly, we need to ensure that there exists a safe starting point for the learning process. Thus, we assume that we have an initial policy $\pi_0$ that renders the origin of the system in (1) asymptotically stable within some small set of states $S_0^\varepsilon$. For example, this policy may be designed using the prior model $h$ in (1), since most models are locally accurate but deteriorate in quality as state magnitude increases. This policy is explicitly not safe to use throughout the state space $X \setminus S_0^\varepsilon$.

**3 Theory**

In this section, we use these assumptions for safe reinforcement learning. We start by computing the region of attraction for a fixed policy under the statistical model. Next, we optimize the policy in order to expand the region of attraction. Lastly, we show that it is possible to safely learn about the dynamics and, under additional assumptions about the model and the system’s reachability properties, that this approach expands the estimated region of attraction safely. We consider an idealized algorithm that is amenable to analysis, which we convert to a practical variant in Sec. 4. See Fig. 1 for an illustrative run of the algorithm and examples of the sets defined below.

**Region of attraction** We start by computing the region of attraction for a fixed policy $\pi$ by an extension of the method in [28] to discrete-time systems. We want to use the Lyapunov decrease condition in Theorem 1 to guarantee safety for the statistical model of the dynamics. However, the posterior uncertainty in the statistical model of the dynamics means that one step predictions about $v(f(\cdot))$ are uncertain too. We account for this by constructing high-probability confidence intervals on $v(f(x, u))$: $Q_n(x, u) := [v_{\mu_n}(x, u) \pm \delta_n(x, u)]$. From Assumption 2 together with the Lipschitz property of $v$, we know that $v(f(x, u))$ is contained in $Q_n(x, u)$ with probability at least $(1 - \delta)$. For our exploration analysis, we need to ensure that safe state-actions cannot become unsafe; that is, an initial set of safe set $S_0$ remains safe (defined later). To this end, we intersect the confidence intervals: $C_n(x, u) := C_{n-1} \cap Q_n(x, u)$, where the set $C$ is initialized to $C_0(x, u) = (-\infty, v(x) - L_{\Delta_n} \tau)$ when $(x, u) \in S_0$ and $C_0(x, u) = \mathbb{R}$ otherwise. Note that $v(f(x, u))$ is contained in $C_n(x, u)$ with the same $(1 - \delta)$ probability as in Assumption 2. The upper and lower bounds on $v(f(\cdot))$ are defined as $\hat{u}_n(x, u) := \max C_n(x, u)$ and $l_n(x, u) := \min C_n(x, u)$.

Given these high-probability confidence intervals, the system is stable according to Theorem 1 if $v(f(x, u)) \leq \hat{u}_n(x) < v(x)$ for all $x \in \mathcal{V}(c)$. However, it is intractable to verify this condition directly on the continuous domain without additional, restrictive assumptions about the model. Instead, we consider a discretization of the state space $\mathcal{X}_T \subset \mathcal{X}$ into cells, so that $|x - \lfloor x \rfloor_\tau| \leq \tau$ holds for all $x \in \mathcal{X}$. Here, $\lfloor x \rfloor_\tau$ denotes the point in $\mathcal{X}_T$ with the smallest $l_1$ distance to $x$. Given this discretization, we bound the decrease variation on the Lyapunov function for states in $\mathcal{X}_T$, and use the Lipschitz continuity to generalize to the continuous state space $\mathcal{X}$.

**Theorem 2. Under Assumptions 1 and 2 with $L_{\Delta_n} := L_{\Delta}L_f(L_{\pi} + 1) + L_v$, let $\mathcal{X}_T$ be a discretization of $\mathcal{X}$ such that $|x - \lfloor x \rfloor_\tau| \leq \tau$ for all $x \in \mathcal{X}$. If, for all $x \in \mathcal{V}(c) \cap \mathcal{X}_T$ with $c > 0$, $u = \pi(x)$, and for some $n \geq 0$ it holds that $u_n(x, u) < v(x) - L_{\Delta_n} \tau$, then $v(f(x, \pi(x))) < v(x)$ holds for all $x \in \mathcal{V}(c)$ with probability at least $(1 - \delta)$ and $\mathcal{V}(c)$ is a region of attraction for (1) under policy $\pi$.

The proof is given in Appendix A.1. Theorem 2 states that, given confidence intervals on the statistical model of the dynamics, it is sufficient to check the stricter decrease condition in Theorem 2 on the discretized domain $\mathcal{X}_T$ to guarantee the requirements for the region of attraction in the continuous domain in Theorem 1. The bound in Theorem 2 becomes tight as the discretization constant $\tau$ and $|v(f(\cdot)) - u_n(\cdot)|$ go to zero. Thus, the discretization constant trades off computation costs for accuracy, while $u_n$ approaches $v(f(\cdot))$ as we obtain more measurement data and the posterior model uncertainty about the dynamics, $\sqrt{\beta_n} \delta_n$ decreases. The confidence intervals on $v(f(x, \pi(x)))$ and $v(x)$ and the corresponding estimated region of attraction (red line) can be seen in the bottom half of Fig. 1.

**Policy optimization** So far, we have focused on estimating the region of attraction for a fixed policy. Safety is a property of states under a fixed policy. This means that the policy directly determines
which states are safe. Specifically, to form a region of attraction all states in the discretization $\mathcal{X}$
within a level set of the Lyapunov function need to fulfill the decrease condition in Theorem 2 that
states are safe. Specifically, we want to pick the policy that leads to the largest possible
region of attraction according to Theorem 2. This requires that for each discrete state in $\mathcal{X}$
the corresponding state-action pair under the policy must be in the set $\mathcal{D}_n$. Thus, we optimize the policy according to
\[
\pi_n, c_n = \arg\max_{\pi \in \Pi_L, c \in \mathbb{R}_{>0}} c, \text{ such that for all } x \in \mathcal{V}(c) \cap \mathcal{X}, (x, \pi(x)) \in \mathcal{D}_n.
\]

The region of attraction that corresponds to the optimised policy $\pi_n$, according to (3) is given by
$\mathcal{V}(c_n)$, see Fig. 1(b). It is the largest level set of the Lyapunov function for which all state-action
pairs $(x, \pi_n(x))$ that correspond to discrete states within $\mathcal{V}(c_n) \cap \mathcal{X}$ are contained in $\mathcal{D}_n$. This
means that these state-action pairs fulfill the requirements of Theorem 2 and $\mathcal{V}(c_n)$ is a region of
attraction of the true system under policy $\pi_n$. The following theorem is thus a direct consequence
of Theorem 2 and (3).

**Theorem 3.** Let $\mathcal{R}_{\pi_n}$ be the true region of attraction of (1) under the policy $\pi_n$. For any $\delta \in (0, 1)$,
we have with probability at least $(1 - \delta)$ that $\mathcal{V}(c_n) \subseteq \mathcal{R}_{\pi_n}$ for all $n > 0$.

Thus, when we optimize the policy subject to the constraint in (3) the estimated region of attraction is
always an inner approximation of the true region of attraction. However, solving the optimization
problem in (3) is intractable in general. We approximate the policy update step in Sec. 4.

**Collecting measurements** Given these stability guarantees, it is natural to ask how one might obtain
data points in order to improve the model of $g(\cdot)$ and thus efficiently increase the region of attraction.
This question is difficult to answer in general, since it depends on the property of the statistical model.
In particular, for general statistical models it is often not clear whether the confidence intervals
contract sufficiently quickly. In the following, we make additional assumptions about the model and
reachability within $\mathcal{V}(c_n)$ in order to provide exploration guarantees. These assumptions allow us to
highlight fundamental requirements for safe data acquisition and that safe exploration is possible.
We assume that the unknown model errors \( g(\cdot) \) have bounded norm in a reproducing kernel Hilbert space (RKHS, [33]) corresponding to a differentiable kernel \( k \), \( \|g(\cdot)\| \leq B_g \). These are a class of well-behaved functions of the form \( g(z) = \sum_{i=0}^\infty \alpha_i k(z, z) \) defined through representer points \( z_i \) and weights \( \alpha_i \) that decay sufficiently fast with \( i \). This assumption ensures that \( g \) satisfies the Lipschitz property in Assumption 1, see [28]. Moreover, with \( \beta_n = B_g + 4\sigma \sqrt{\gamma_n + 1 + \ln(1/\delta)} \) we can use GP models for the dynamics that fulfill Assumption 2 if the state if fully observable and the measurement noise is \( \sigma \)-sub-Gaussian (e.g., bounded in \([-\sigma, \sigma]\)), see [34]. Here \( \gamma_n \) is the information capacity. It corresponds to the amount of mutual information that can be obtained about \( g \) from \( nq \) measurements, a measure of the size of the function class encoded by the model. The information capacity has a sublinear dependence on \( n \) for common kernels and upper bounds can be computed efficiently [35]. More details about this model are given in Appendix A.2.

In order to quantify the exploration properties of our algorithm, we consider a discrete action space \( \mathcal{U}_c \subset \mathcal{U} \). We define exploration as the number of state-action pairs in \( \mathcal{X}_c \times \mathcal{U}_c \) that we can safely learn about without leaving the true region of attraction. Note that despite this discretization, nearby states are sufficiently correlated under the model to enable generalization using (2). In practice, any (model-based) exploration strategy that aims to commit to a specific method. In practice, any (model-based) exploration strategy that aims to

Suppose we are given a set \( S_0 \) of state-action pairs about which we can learn safely. Specifically, this means that we have a policy such that, for any state-action pair \((x, u)\) in \( S_0 \), if we apply action \( u \) in state \( x \) and then apply actions according to the policy, the state converges to the origin. Such a set can be constructed using the initial policy \( \pi_0 \) from Sec. 2 as \( S_0 = \{(x, \pi_0(x)) \mid x \in S_0^s\} \). Starting from this set, we want to update the policy to expand the region of attraction according to Theorem 2.

To this end, we use the confidence intervals on \( v(f(\cdot)) \) for states inside \( S_0 \) to determine state-action pairs that fulfill the decrease condition. We thus redefine \( D_n \) for the exploration analysis to

\[
D_n = \bigcup_{(x,u) \in S_{n-1}} \{z' \in \mathcal{X}_c \times \mathcal{U}_c \mid u_n(x, u) - v(x) + L\Delta_n \|z' - (x, u)\|_1 < -L\Delta_n \tau\}. \tag{4}
\]

This formulation is equivalent to (2), except that it uses the Lipschitz constant to generalize safety. Given \( D_n \), we can again find a region of attraction \( V(c_n) \) by committing to a policy according to (3). In order to expand this region of attraction effectively we need to decrease the posterior model uncertainty about the dynamics of the GP by collecting measurements. However, to ensure safety as outlined in Sec. 2, we are not only restricted to states within \( V(c_n) \), but also need to ensure that the state after taking an action is safe; that is, the dynamics map the state back into the region of attraction \( V(c_n) \). We again use the Lipschitz constant in order to determine this set,

\[
S_n = \bigcup_{z \in S_{n-1}} \{z' \in V(c_n) \cap \mathcal{X}_c \times \mathcal{U}_c \mid u_n(z) + L_n L_f \|z - z'\|_1 \leq c_n\}. \tag{5}
\]

The set \( S_n \) contains state-action pairs that we can safely evaluate under the current policy \( \pi_n \) without leaving the region of attraction, see Fig. 1 (top, white shaded).

What remains is to define a strategy for collecting data points within \( S_n \) to effectively decrease model uncertainty. We specifically focus on the high-level requirements for any exploration scheme without committing to a specific method. In practice, any (model-based) exploration strategy that aims to decrease model uncertainty by driving the system to specific states may be used. Safety can be ensured by picking actions according to \( \pi_n \) whenever the exploration strategy reaches the boundary of the safe region \( V(c_n) \); that is, when \( u_n(x, u) > c_n \). This way, we can use \( \pi_n \) as a backup policy for exploration.

The high-level goal of the exploration strategy is to shrink the confidence intervals at state-action pairs \( S_n \) in order to expand the safe region. Specifically, the exploration strategy should aim to visit state-action pairs in \( S_n \) at which we are the most uncertain about the dynamics; that is, where the confidence interval is the largest:

\[
(x_n, u_n) = \arg\max_{(x,u) \in S_n} u_n(x, u) - l_n(x, u). \tag{6}
\]

As we keep collecting data points according to (6), we decrease the uncertainty about the dynamics for different actions throughout the region of attraction and adapt the policy, until eventually we
We compare the active learning scheme in (6) to an oracle baseline that starts from the same initial
which is guaranteed to be safe.

Algorithm 1 SAFE Lyapunov Learning

1: Input: Initial safe policy \( \pi_0 \), dynamics model \( \mathcal{P}(\mu(z), k(z, z')) \)
2: for all \( n = 1, \ldots \) do 
3: Compute policy \( \pi_n \) via SGD on (7)
4: \( c_n = \text{argmax}_c, \) such that \( \forall x \in \mathcal{V}(c_n) \cap \mathcal{X} \; \colon \; u_n(x, \pi_n(x)) - v(x) < -L_\Delta v \)
5: \( S_n = \{ (x, u) \in \mathcal{V}(c_n) \times \mathcal{U} \mid u_n(x, u) \leq c_n \} \)
6: Select \( (x_n, u_n) \) within \( S_n \) using (6) and drive system there with backup policy \( \pi_n \)
7: Update GP with measurements \( f(x_n, u_n) + \epsilon_n \)

have gathered enough information in order to expand it. While (6) implicitly assumes that any state within \( \mathcal{V}(c_n) \) can be reached by the exploration policy, it achieves the high-level goal of any exploration algorithm that aims to reduce model uncertainty. In practice, any safe exploration scheme is limited by unreachable parts of the state space.

We compare the active learning scheme in (6) to an oracle baseline that starts from the same initial safe set \( S_0 \) and knows \( v(f(x, u)) \) up to \( \epsilon \) accuracy within the safe set. The oracle also uses knowledge about the Lipschitz constants and the optimal policy in \( \Pi_L \) at each iteration. We denote the set that this baseline manages to determine as safe with \( \overline{R}_\epsilon(S_0) \) and provide a detailed definition in Appendix A.3.

**Theorem 4.** Assume \( \sigma \)-sub-Gaussian measurement noise and that the model error \( g(\cdot) \) in (1) has RKHS norm smaller than \( B_\beta \). Under the assumptions of Theorem 2, with \( \beta_n = B_\beta + 4\sigma \sqrt{\gamma_n} + 1 + \ln(1/\delta) \), and with measurements collected according to (6), let \( n^* \) be the smallest positive integer so that \( \frac{n^*}{\beta_n^2 + \gamma_n^*} \geq C0(\overline{R}(S_0)+1) \) where \( C = 8/\log(1+\sigma^{-2}) \).

Let \( R_\pi \) be the true region of attraction of (1) under a policy \( \pi \). For any \( \epsilon > 0 \) and \( \delta \in (0, 1) \), the following holds jointly with probability at least \( (1 - \delta) \) for all \( n > 0 \):

(i) \( \mathcal{V}(c_n) \subseteq R_{\pi_n} \)
(ii) \( f(x, u) \in \mathcal{V}_{\pi_n} \forall (x, u) \in S_n \)
(iii) \( \overline{R}_\epsilon(S_0) \subseteq S_n \subseteq \overline{R}_0(S_0) \).

Theorem 4 states that, when selecting data points according to (6), the estimated region of attraction \( \mathcal{V}(c_n) \) is (i) contained in the true region of attraction under the current policy and (ii) selected data points do not cause the system to leave the region of attraction. This means that any exploration method that considers the safety constraint (5) is able to safely learn about the system without leaving the region of attraction. The last part of Theorem 4, (iii), states that after a finite number of data points \( n^* \) we achieve at least the exploration performance of the oracle baseline, while we do not classify unsafe state-action pairs as safe. This means that the algorithm explores the largest region of attraction possible for a given Lyapunov function with residual uncertainty about \( v(f(\cdot)) \) smaller than \( \epsilon \). Details of the comparison baseline are given in the appendix. In practice, this means that any exploration method that manages to reduce the maximal uncertainty about the dynamics within \( S_n \) is able to expand the region of attraction.

An example run of repeatedly evaluating (6) for a one-dimensional state-space is shown in Fig. 1. It can be seen that, by only selecting data points within the current estimate of the region of attraction, the algorithm can efficiently optimize the policy and expand the safe region over time.

### 4 Practical Implementation and Experiments

In the previous section, we have given strong theoretical results on safety and exploration for an idealized algorithm that can solve (3). In this section, we provide a practical variant of the theoretical algorithm in the previous section. In particular, while we retain safety guarantees, we sacrifice exploration guarantees to obtain a more practical algorithm. This is summarized in Algorithm 1.

The policy optimization problem in (3) is intractable to solve and only considers safety, rather than a performance metric. We propose to use an approximate policy update that that maximizes approximate performance while providing stability guarantees. It proceeds by optimizing the policy first and then computes the region of attraction \( \mathcal{V}(c_n) \) for the new, fixed policy. This does not impact safety, since data is still only collected inside the region of attraction. Moreover, should the optimization fail and the region of attraction decrease, one can always revert to the previous policy, which is guaranteed to be safe.
In our experiments, we use approximate dynamic programming [36] to capture the performance of the policy. Given a policy \( \pi_\theta \) with parameters \( \theta \), we compute an estimate of the cost-to-go \( J_{\pi_\theta}(x) \) for the mean dynamics \( \mu_{\pi_\theta} \) based on the cost \( r(x, u) \) \( \geq 0 \). At each state, \( J_{\pi_\theta}(x) \) is the sum of \( \gamma \)-discounted rewards encountered when following the policy \( \pi_\theta \). The goal is to adapt the parameters of the policy for minimum cost as measured by \( J_{\pi_\theta} \), while ensuring that the safety constraint on the worst-case decrease on the Lyapunov function in Theorem 2 is not violated. A Lagrangian formulation to this constrained optimization problem is

\[
\pi_\theta = \arg\min_{\pi_\theta \in \Pi_{\mathcal{L}}} \int_{x \in \mathcal{X}} r(x, \pi_\theta(x)) + \gamma J_{\pi_\theta}(\mu_{\pi_\theta-1}(x, \pi_\theta(x))) + \lambda \left( u_n(x, \pi_\theta(x)) - v(x) + L_{\Delta v} \right),
\]

where the first term measures long-term cost to go and \( \lambda \geq 0 \) is a Lagrange multiplier for the safety constraint from Theorem 2. In our experiments, we use the value function as a Lyapunov function candidate, \( v = J \) with \( r(\cdot, \cdot) \geq 0 \), and set \( \lambda = 1 \). In this case, (7) corresponds to an high-probability upper bound on the cost-to-go given the uncertainty in the dynamics. This is similar to worst-case performance formulations found in robust MDPs [9, 10], which consider worst-case value functions given parametric uncertainty in MDP transition model. Moreover, since \( L_{\Delta v} \) depends on the Lipschitz constant of the policy, this simultaneously serves as a regularizer on the parameters \( \theta \).

To verify safety, we use the GP confidence intervals \( l_n \) and \( u_n \), directly, as in (2). We also use confidence to compute \( S_n \) for the active learning scheme, see Algorithm 1, Line 5. In practice, we do not need to compute the entire set \( S_n \) to solve (3), but can use a global optimization method or even a random sampling scheme within \( \mathcal{V}(c_n) \) to find suitable state-actions. Moreover, measurements for actions that are far away from the current policy are unlikely to expand \( \mathcal{V}(c_n) \), see Fig. 1(c). As we optimize (7) via gradient descent, the policy changes only locally. Thus, we can achieve better data-efficiency by restricting the exploratory actions \( u \) with \((x, u) \in S_n \) to be close to \( \pi_n \), \( u \in [\pi_n(x) - \bar{u}, \pi_n(x) + \bar{u}] \) for some constant \( \bar{u} \).

Computing the region of attraction by verifying the stability condition on a discretized domain suffers from the curse of dimensionality. However, it is not necessary to update policies in real time. In particular, since any policy that is returned by the algorithm is provably safe within some level set, any of these policies can be used safely for an arbitrary number of time steps. To scale this method to higher-dimensional system, one would have to consider an adaptive discretization for the verification as in [27].

**Experiments** A Python implementation of Algorithm 1 and the experiments based on TensorFlow [37] and GPflow [38] is available at https://github.com/befelix/safe_learning.

We verify our approach on an inverted pendulum benchmark problem. The true, continuous-time dynamics are given by \( ml^2 \ddot{\psi} = gml \sin(\psi) - \lambda \dot{\psi} + u \), where \( \psi \) is the angle, \( m \) the mass, \( g \) the gravitational constant, and \( u \) the torque applied to the pendulum. The control torque is limited, so that the pendulum necessarily falls down beyond a certain angle. We use a GP model for the discrete-time dynamics, where the mean dynamics are given by a linearized and discretized model of the true dynamics that considers a wrong, lower mass and neglects friction. As a result, the optimal policy for
the mean dynamics does not perform well and has a small region of attraction as it underactuates the system. We use a combination of linear and Matérn kernels in order to capture the model errors that result from parameter and integration errors.

For the policy, we use a neural network with two hidden layers and 32 neurons with ReLU activations each. We compute a conservative estimate of the Lipschitz constant as in [30]. We use standard approximate dynamic programming with a quadratic, normalized cost \( r(x, u) = x^T Q x + u^T R u \), where \( Q \) and \( R \) are positive-definite, to compute the cost-to-go \( J_{\pi^n} \). Specifically, we use a piecewise-linear triangulation of the state-space as to approximate \( J_{\pi^n} \), see [39]. This allows us to quickly verify the assumptions that we made about the Lyapunov function in Sec. 2 using a graph search. In practice, one may use other function approximators. We optimize the policy via stochastic gradient descent on (7), where we sample a finite subset of \( \mathcal{X} \) and replace the integral in (7) with a sum.

The theoretical confidence intervals for the GP model are conservative. To enable more data-efficient learning, we fix \( \beta_n = 2 \). This corresponds to a high-probability decrease condition per-state, rather than jointly over the state space. Moreover, we use local Lipschitz constants of the Lyapunov function rather than the global one. While this does not affect guarantees, it greatly speeds up exploration.

For the initial policy, we use approximate dynamic programming to compute the optimal policy for the prior mean dynamics. This policy is unstable for large deviations from the initial state and has poor performance, as shown in Fig. 2(b). Under this initial, suboptimal policy, the system is stable within a small region of the state-space Fig. 2(a). Starting from this initial safe set, the algorithm proceeds to collect safe data points and improve the policy. As the uncertainty about the dynamics decreases, the policy improves and the estimated region of attraction increases. The region of attraction after 50 data points is shown in Fig. 2(a). The resulting set \( V(c_n) \) is contained within the true safe region of the optimized policy \( \pi_n \). At the same time, the control performance improves drastically relative to the initial policy, as can be seen in Fig. 2(b). Overall, the approach enables safe learning about dynamic systems, as all data points collected during learning are safely collected under the current policy.

5 Conclusion

We have shown how classical reinforcement learning can be combined with safety constraints in terms of stability. Specifically, we showed how to safely optimize policies and give stability certificates based on statistical models of the dynamics. Moreover, we provided theoretical safety and exploration guarantees for an algorithm that can drive the system to desired state-action pairs during learning. We believe that our results present an important first step towards safe reinforcement learning algorithms that are applicable to real-world problems.

Acknowledgments

This research was supported by SNSF grant 200020_159557, the Max Planck ETH Center for Learning Systems, NSERC grant RGPIN-2014-04634, and the Ontario Early Researcher Award.

References


