Proceedings of Machine Learning Research vol xxx:1–19, 2021

Safely Learning Dynamical Systems from Short Trajectories

Amir Ali Ahmadi AAA@PRINCETON.EDU
Abraar Chaudhry AZC@PRINCETON.EDU
ORFE, Princeton University

Vikas Sindhwani SINDHWANI@GOOGLE.COM
Stephen Tu STEPHENTU@GOOGLE.COM
Robotics at Google, New York

Abstract

A fundamental challenge in learning to control an unknown dynamical system is to reduce model uncertainty by making measurements while maintaining safety. In this work, we formulate a mathematical definition of what it means to safely learn a dynamical system by sequentially deciding where to initialize the next trajectory. In our framework, the state of the system is required to stay within a given safety region under the (possibly repeated) action of all dynamical systems that are consistent with the information gathered so far. For our first two results, we consider the setting of safely learning linear dynamics. We present a linear programming-based algorithm that either safely recovers the true dynamics from trajectories of length one, or certifies that safe learning is impossible. We also give an efficient semidefinite representation of the set of initial conditions whose resulting trajectories of length two are guaranteed to stay in the safety region. For our final result, we study the problem of safely learning a nonlinear dynamical system. We give a second-order cone programming based representation of the set of initial conditions that are guaranteed to remain in the safety region after one application of the system dynamics.

Keywords: learning dynamical systems, safe learning, uncertainty quantification, robust optimization, conic programming

1. Introduction and Problem Formulation

The core task in model-based reinforcement learning (Yang et al., 2020; Nagabandi et al., 2018; Singh et al., 2019; Lowrey et al., 2018; Venkatraman et al., 2016; Kaiser et al., 2019) is to estimate—from a small set of sampled trajectories—an unknown dynamical system prescribing the evolution of an agent's state given the current state and control input. During the initial stages of learning, deploying even a conservative feedback policy on a real robot is fraught with risk, even if the policy achieves high task performance and safe behavior in simulation. How should the robot be "set loose" in the real world so that the dynamics may be precisely estimated by observing state transitions, but with strong guarantees that the robot will remain safe? This interplay between *safety and uncertainty while learning dynamical systems* is the central theme of this paper.

We view the agent armed with a fixed feedback policy in closed loop over a short duration as an unknown discrete-time dynamical system

$$x_{t+1} = f_\star(x_t). \tag{1}$$

We consider the problem of safe data acquisition for estimating the unknown map $f_\star : \mathbb{R}^n \to \mathbb{R}^n$ from a collection of length-T trajectories $\{\phi_{f_\star,T}(x_k)\}_{k=1}^m$, where $\phi_{f,T}(x) := (x, f(x), \ldots, f^{(T)}(x))$. In our setting, we are given as input a set $S \subset \mathbb{R}^n$, called the *safety region*, in which the state should remain throughout the learning process. We say that a state $x \in \mathbb{R}^n$ is T*-step safe* under a map $f : \mathbb{R}^n \to \mathbb{R}^n$ if x belongs to the set

$$S^T(f) := \{x \in S \mid f^{(i)}(x) \in S, i = 1, \ldots, T\}.$$

In order to safely learn $f_\star$, we require that measurements are made only at points $x \in \mathbb{R}^n$ for which $x \in S^T(f_\star)$. Obviously, if we make no assumptions about $f_\star$, this task is impossible. We assume, therefore, that the map $f_\star$ belongs to a set of dynamics U_0, which we call the *initial uncertainty set*. As experience is gathered, the uncertainty over $f_\star$ decreases. Let us denote the uncertainty set after the agent has observed k trajectories $\{\phi_{f_\star,T}(x_j)\}_{j=1}^k$ by,

$$U_k := \{f \in U_0 \mid \phi_{f,T}(x_j) = \phi_{f_\star,T}(x_j)\,, j = 1, \ldots, k\}.$$

For a nonnegative integer k, define

$$S_k^T := \bigcap_{f \in U_k} S^T(f)\,,$$

the set of points that are T-step safe under all dynamics consistent with the data after observing k trajectories. Fix a distance metric $d(\cdot, \cdot)$ over U_0 and a scalar $\varepsilon > 0$. Given a safety region $S \subset \mathbb{R}^n$ and an initial uncertainty set U_0, we say that T*-step safe learning* is possible (with respect to the metric $d(\cdot, \cdot)$ and up to accuracy ε) if for some nonnegative integer m, we can sequentially choose vectors $x_1, \ldots, x_m$ such that,

1. **(Safety)** for each $k = 1, \ldots, m$, $x_k \in S_{k-1}^T$,
2. **(Learning)** $\sup_{f \in U_m} d(f, f_\star) \leq \varepsilon$.

Note that for any $T' > T$, we have $S_k^{T'} \subseteq S_k^T$ for all k. Hence, if T-step safe learning is impossible, then T'-step safe learning is also impossible. Therefore, the highest rate of safe information assimilation during the learning process is achieved when $T = 1$. One of the main contributions of this paper is to present an efficient algorithm for the exact one-step safe learning problem (i.e., when $\varepsilon = 0$ and $T = 1$) in the case where the dynamics in (1) are linear, U_0 is a polyhedron in the space of $n \times n$ matrices that define the dynamics, and S is a polyhedron (Algorithm 1 and Theorem 6).

Suppose furthermore that initializing the unknown system at a state $x \in S$ comes at a cost of $c(x)$. In such a setting, we are also interested in safely learning at minimum measurement cost. To do this, one naturally wants to solve an optimization problem of the type

$$\min_{x \in S_{k-1}^T} \; c(x)\,, \tag{2}$$

whose optimal solution gives us the next cheapest T-step safe query point x_k. Another contribution of this paper is to derive exact reformulations of problem (2), when $T \in \{1, 2\}$, in terms of tractable conic optimization problems. More specifically, under natural assumptions on S and U_0, when the unknown dynamics are linear, we show that problem (2) can be formulated as a linear program when $T = 1$ (Theorem 1) and as a semidefinite program when $T = 2$ (Theorem 8). Furthermore,

when $T = 1$ and the unknown dynamics are nonlinear (but bounded in a certain sense), we show that (2) can be formulated as a second-order cone program (Theorem 10). Finally, we note that we are currently preparing a draft to handle the case when $T = \infty$ using the set invariance tools of Ahmadi and Günlük (2018).

2. Related Work

Most related to our work is Dean et al. (2019), which uses the system-level synthesis framework (Anderson et al., 2019) to derive inner approximations to the infinite-step safety region of a linear system subject to polytopic uncertainty in the dynamics and bounded disturbances. Lu et al. (2017) considers a probabilistic version of one-step safety for linear systems and also presents an algorithm to conservatively compute the T-step safety regions. Unlike these papers that focus on inner approximations of safety regions, we are able to exactly characterize one-step and two-step safety regions under our proposed framework. We also note that we do not require any stability assumptions on the dynamical systems we want to learn.

We also review other works focused on the general problem of safely learning dynamics in both the control theory and reinforcement learning literature. Berkenkamp et al. (2017) combines Lyapunov functions and Gaussian process models to show how to safely explore an uncertain system and expand an inner estimate of the region of attraction of one of its equilibrium points. Akametalu et al. (2014) uses reachability analysis to compute maximal safe regions for uncertain dynamics, and proposes Gaussian processes to iteratively refine the uncertainty. Koller et al. (2019) shows how to propagate ellipsoidal uncertainty multiple steps into the future, and utilizes this uncertainty propagation in a model predictive control framework for safely learning to control. Wabersich and Zeilinger (2018) shows how to minimally perturb a controller designed to learn a linear system in order for the system to stay within a set of constraints that guarantee reachability to a safe target set.

We also note that our work has some conceptual connections to the literature on experiment design (see e.g., Pukelsheim, 2006; De Castro et al., 2019). However, this literature typically does not consider dynamical systems or notions of safety.

3. One-Step Safe Learning of Linear Systems

In this section, we focus on characterizing one-step safe learning for linear systems. Here, the state evolves according to

$$x_{t+1} = A_\star x_t, \tag{3}$$

where $A_\star$ is an unknown $n \times n$ matrix. We assume we know that $A_\star$ belongs to a set $U_0 \subset \mathbb{R}^{n\times n}$ that represents our prior knowledge of $A_\star$. In this section, we take U_0 to be a polyhedron; i.e.,

$$U_0 = \left\{A \in \mathbb{R}^{n\times n} \mid \text{Tr}(V_j^T A) \leq v_j \quad j = 1, \ldots, s\right\} \tag{4}$$

for some matrices $V_1, \ldots, V_s \in \mathbb{R}^{n\times n}$ and scalars $v_1, \ldots, v_s \in \mathbb{R}$. We also work with a polyhedral representation of the safety region S; i.e.,

$$S = \left\{x \in \mathbb{R}^n \mid h_i^T x \leq b_i \quad i = 1, \ldots, r\right\} \tag{5}$$

for some vectors $h_1, \ldots, h_r \in \mathbb{R}^n$ and some scalars $b_1, \ldots, b_r \in \mathbb{R}$. We assume that making a query at a point $x \in \mathbb{R}^n$ comes at a cost $c^T x$, where the vector $c \in \mathbb{R}^n$ is given[1]. An extension to more general semidefinite representable cost functions is possible using tools of conic optimization.

We start by finding the minimum cost point that is one-step safe under all valid dynamics, i.e., a point $x \in S$ such that $Ax \in S$ for all $A \in U_0$. Once this is done, we gain further information by observing the action $y = A_\star x$ of system (3) on our point x, which further constrains the uncertainty set U_0. We then repeat this procedure with the updated uncertainty set to find the next minimum cost one-step safe point. More generally, after collecting k measurements, our uncertainty in the dynamics reduces to the set

$$U_k = \{A \in U_0 \mid Ax_j = y_j \quad j = 1, \ldots, k\}. \tag{6}$$

Hence, the problem of finding the next cheapest one-step safe query point becomes:

$$\begin{aligned} \min_{x \in \mathbb{R}^n} \quad & c^T x \\ \text{s.t.} \quad & x \in S \\ & Ax \in S \quad \forall A \in U_k. \end{aligned} \tag{7}$$

In Section 3.1, we show that problem (7) can be efficiently solved. We then use (7) as a subroutine in a one-step safe learning algorithm which we present in Section 3.2.

3.1. Reformulation via Duality

In this subsection, we reformulate (7) as a linear program. To do this we introduce auxiliary variables $\mu_j^{(i)} \in \mathbb{R}$ and $\eta_k^{(i)} \in \mathbb{R}^n$ for $i = 1, \ldots, r$, $j = 1, \ldots, s$, and $k = 1, \ldots, m$.

Theorem 1 *The feasible set of problem* (7) *is the projection onto x-space of the feasible set of the following linear program:*

$$\begin{aligned} \min_{x,\mu,\eta} \quad & c^T x \\ \textit{s.t.} \quad & h_i^T x \le b_i \quad i = 1, \ldots, r \\ & \sum_{k=1}^{m} y_k^T \eta_k^{(i)} + \sum_{j=1}^{s} \mu_j^{(i)} v_j \le b_i \quad i = 1, \ldots, r \\ & x h_i^T = \sum_{k=1}^{m} x_k \eta_k^{(i)T} + \sum_{j=1}^{s} \mu_j^{(i)} V_j^T \quad i = 1, \ldots, r \\ & \mu^{(i)} \ge 0 \quad i = 1, \ldots, r. \end{aligned} \tag{8}$$

In particular, the optimal values of (7) *and* (8) *are the same and the optimal solutions of* (7) *are the optimal solutions of* (8) *projected onto x-space.*

1. In practice, measurement costs are typically nonnegative. If S is compact for example, one can always add a constant term to $c^T x$ to ensure this requirement without changing any of our optimization problems.

Proof Using the definitions of S and U_0, let us first rewrite (7) as bilevel program:

$$\begin{aligned} \min_{x} \quad & c^T x \\ \text{s.t.} \quad & h_i^T x \le b_i \quad i = 1, \dots, r \\ & \begin{bmatrix} \max_A & h_i^T A x \\ \text{s.t.} & \text{Tr}(V_j^T A) \le v_j \quad j = 1, \dots, s \\ & A x_k = y_k \quad k = 1, \dots, m \end{bmatrix} \le b_i \quad i = 1, \dots, r. \end{aligned} \tag{9}$$

We proceed by taking the dual of the r inner programs, treating the x variable as fixed. By introducing dual variables $\mu_j^{(i)}$ and $\eta_k^{(i)}$ for $i = 1, \dots, r$, $j = 1, \dots, s$, and $k = 1, \dots, m$, and by invoking strong duality of linear programming, we have

$$\begin{bmatrix} \max_A & h_i^T A x \\ \text{s.t.} & \text{Tr}(V_j^T A) \le v_j \; j = 1, \dots, s \\ & A x_k = y_k \; k = 1, \dots, m \end{bmatrix} = \begin{bmatrix} \min_{\mu^{(i)}, \eta^{(i)}} & \sum_{k=1}^m y_k^T \eta_k^{(i)} + \sum_{j=1}^s \mu_j^{(i)} v_j \\ \text{s.t.} & x h_i^T = \sum_{k=1}^m x_k \eta_k^{(i)T} + \sum_{j=1}^s \mu_j^{(i)} V_j^T \\ & \mu^{(i)} \ge 0 \end{bmatrix} \tag{10}$$

for $i = 1, \dots, r$. Thus by replacing the inner problem of (9) with the right-hand side of (10), the min-max problem (9) becomes a min-min problem. This min-min problem can be combined into a single minimization problem and be written as problem (8). Indeed, if x is feasible to (9), for that fixed x and for each i, there exist values of $\mu^{(i)}$ and $\eta^{(i)}$ that attain the optimal value for (10) and therefore the triple (x, μ, η) will be feasible to (8). Conversely, if some (x, μ, η) is feasible to (8), it follows that x is feasible to (9). This is because for any fixed x and for each i, the optimal value of the left-hand side of (10) is bounded from above by the objective value of the right-hand side evaluated at any feasible $\mu^{(i)}$ and $\eta^{(i)}$. ■

We remark that (8) can be modified so that one-step safety is achieved in the presence of disturbances. We can ensure, e.g., using linear programming, that $Ax + w \in S$ for all $A \in U_m$ and all w such that $\|w\| \le W$, where $\|\cdot\|$ is any norm whose unit ball is a polytope and W is a given scalar.

3.2. An Algorithm for One-Step Safe Learning

We start by giving a mathematical definition of (exact) safe learning specialized to the case of one-step safety and linear dynamics. Recall the definition of the set U_k in (6).

Definition 2 (One-Step Safe Learning) *We say that one-step safe learning is possible if for some nonnegative integer m, we can sequentially choose vectors $x_k \in S$, for $k = 1, \dots, m$, and observe measurements $y_k = A_\star x_k$ such that:*

1. ***(Safety)** for $k = 1, \dots, m$, we have $Ax_k \in S \quad \forall A \in U_{k-1}$,*
2. ***(Learning)** the set of matrices U_m is a singleton.*

We now present our algorithm for checking the possibility of one-step safe learning (Algorithm 1). The proof of correctness of Algorithm 1 is given in Theorem 6.

Remark 3 *As Theorem 6 will demonstrate, the particular choice of the parameter $\varepsilon \in (0, 1]$ in the input to Algorithm 1 does not affect the detection of one-step safe learning by this algorithm. However, a smaller ε leads to a lower cost of learning. Therefore, in practice, ε should be chosen positive and as small as possible without causing the matrix X in line 25 to be ill conditioned.*

Algorithm 1: One-Step Safe Learning Algorithm

Input : polyhedra $S \subset \mathbb{R}^n$ and $U_A \subset \mathbb{R}^{n\times n}$, cost vector $c \in \mathbb{R}^n$, and a constant $\varepsilon \in (0, 1]$.
Output: A matrix $A_\star \in \mathbb{R}^{n\times n}$ or a declaration that one-step safe learning is impossible.

for $k = 0, \dots, n-1$ **do**
  $D_k \leftarrow \{(x_j, y_j) \mid j = 1, \dots, k\}$
  $U_k \leftarrow \{A \in U_0 \mid Ax_j = y_j, \quad j = 1, \dots, k\}$
  if *U_k is a singleton (cf. Lemma 4)* **then**
    return the single element in U_k as $A_\star$
  end
  Let $x_k^\star$ be the projection onto x-space of an optimal solution to problem (8) with data D_k
  if *$x_k^\star$ is linearly independent from* $\{x_1, \dots, x_k\}$ **then**
    $x_{k+1} \leftarrow x_k^\star$
  else
    Let S_k^1 be the projection onto x-space of the feasible region of problem (8) with data D_k
    Compute a basis $B_k \subset S_k^1$ of span(S_k^1) (cf. Theorem 5)
    for $z_j \in B_k$ **do**
      if *z_j is linearly independent from* $\{x_1, \dots, x_k\}$ **then**
        $x_{k+1} \leftarrow (1-\varepsilon)x_k^\star + \varepsilon z_j$
        break
      end
    end
    if *no $z_j \in B_k$ is linearly independent from* $\{x_1, \dots, x_k\}$ **then**
      return one-step safe learning is impossible
    end
  end
  Observe $y_{k+1} \leftarrow A_\star x_{k+1}$
end
Define matrix $X = [x_1, \dots, x_n]$
Define matrix $Y = [y_1, \dots, y_n]$
return $A_\star = YX^{-1}$

Algorithm 1 invokes two subroutines which we present next in Lemma 4 and Theorem 5.

Lemma 4 *Let $A \in \mathbb{R}^{m\times n}$, $B \in \mathbb{R}^{m\times p}$, $c \in \mathbb{R}^m$, and define the polyhedron*

$$P := \{x \in \mathbb{R}^n \mid \exists y \in \mathbb{R}^p \quad \text{s.t.} \quad Ax + By \le c\}.$$

The problem of checking if P is a singleton can be reduced to solving $2n$ linear programs.

Proof For each $i = 1, \dots, n$, maximize and minimize the i-th coordinate of x over P. It is straightforward to check that P is a singleton if and only if the optimal values of these two linear programs coincide for every $i = 1, \dots, n$. ■

For stating our next theorem, we use the following notation: given a set $P \subseteq \mathbb{R}^n$, let span(P) denote the set of all linear combinations of points in P.

Theorem 5 *Let $A \in \mathbb{R}^{m\times n}$, $B \in \mathbb{R}^{m\times p}$, $c \in \mathbb{R}^m$, and define the polyhedron*

$$P := \{x \in \mathbb{R}^n \mid \exists y \in \mathbb{R}^p \quad \text{s.t.} \quad Ax + By \leq c\}.$$

One can find a basis of $\text{span}(P)$ *contained within P by solving at most $2n^2$ linear programs.*

Proof We form the desired basis $\{e_i\}$ iteratively and with an inductive argument. Let e_1 be any nonzero vector in P (existence of such a vector can be checked by the argument in the proof of Lemma 4); if there is no such vector, we return the empty set. Let $\{e_1, \ldots, e_k\}$ be a linearly independent set in P. We will either find an additional linearly independent vector $e_{k+1} \in P$, or show that the dimension of the span of P is k. Let x, x^+, and x^- be variables in $\mathbb{R}^n$, y^+ and y^- be variables in $\mathbb{R}^p$, and λ^+ and λ^- be variables in $\mathbb{R}$. Consider the following linear programming feasibility problem:

$$\begin{aligned} & e_i^T x = 0 \quad i = 1, \ldots, k \\ & x = x^+ - x^- \\ & Ax^+ + By^+ \leq \lambda^+ c \\ & Ax^- + By^- \leq \lambda^- c \\ & \lambda^+ \geq 0 \\ & \lambda^- \geq 0. \end{aligned} \tag{11}$$

Let $F \subseteq \mathbb{R}^n$ be the projection onto x-space of the feasible region of this problem. We claim that $F = \{0\}$ if and only if the dimension of $\text{span}(P)$ is k. Moreover, if there is solution to (11) with $x \neq 0$, then there is also a solution $(x, x^\pm, y^\pm, \lambda^\pm)$ where $\lambda^+, \lambda^- \neq 0$. In this case, either $\frac{x^+}{\lambda^+}$ or $\frac{x^+}{\lambda^+}$ can be taken as e_{k+1}.

Suppose first that the dimension $\text{span}(P)$ is at least $k+1$; then there is a vector $\tilde{x} \in \text{span}(P)$ that is linearly independent from $\{e_1, \ldots, e_k\}$. By subtracting the projection of $\tilde{x}$ onto $\text{span}(\{e_1, \ldots, e_k\})$, we will find a nonzero vector $x \in \text{span}(P)$ that is orthogonal to the vectors $e_1, \ldots, e_k$. We claim this vector x is feasible to (11) for some choice of $(x^\pm, y^\pm, \lambda^\pm)$. Indeed, since $x \in \text{span}(P)$, then

$$x = \sum_{j=1}^{r} \lambda_j x_j,$$

for some vectors $x_1, \ldots, x_r \in P$ and some nonzero scalars $\lambda_1, \ldots, \lambda_r$. For each j, as $x_j \in P$, there exists a vector y_j such that $Ax_j + By_j \leq c$. Let J denote the set of indices j such that $\lambda_j > 0$. It is easy to check that the assignment

$$\begin{aligned} (x^+, y^+, \lambda^+) &= \left(\sum_{j\in J} \lambda_j x_j, \sum_{j\in J} \lambda_j y_j, \sum_{j\in J} \lambda_j\right), \\ (x^-, y^-, \lambda^-) &= \left(-\sum_{j\notin J} \lambda_j x_j, -\sum_{j\notin J} \lambda_j y_j, -\sum_{j\notin J} \lambda_j\right) \end{aligned} \tag{12}$$

satisfies system (11). Hence, we have shown that if $F = \{0\}$ then the dimension of $\text{span}(P)$ is k.

To see the converse implication, suppose $x \neq 0$, and that the tuple $(x, x^\pm, y^\pm, \lambda^\pm)$ is feasible to system (11). Without loss of generality we assume $\lambda^\pm \geq 1$; if not, we replace the tuple with

$$(x, x^\pm + \hat{x}, y^\pm + \hat{y}, \lambda^\pm + 1), \tag{13}$$

where $\hat{x}$ and $\hat{y}$ are any vectors satisfying $A\hat{x} + B\hat{y} \leq c$. Then, since $A\frac{x^+}{\lambda^+} + B\frac{y^+}{\lambda^+} \leq c$, the vector $\frac{x^+}{\lambda^+} \in P$. By the same argument, $\frac{x^-}{\lambda^-} \in P$. It follows from the orthogonality constraint of (11) that at least one of the vectors $\frac{x^+}{\lambda^+}$ and $\frac{x^-}{\lambda^-}$ is linearly independent from $\{e_1, \ldots, e_k\}$ and can be taken as e_{k+1}, also proving that the dimension of span(P) is at least $k+1$.

Note that the condition $F = \{0\}$ can be checked by solving $2n$ linear programs (cf. the proof of Lemma 4); if $F \neq \{0\}$, then at least one of these $2n$ linear programs will return a tuple $(x, x^\pm, y^\pm, \lambda^\pm)$ where $x \neq 0$. We then transform this tuple via (13) to ensure that both $\lambda^+, \lambda^- \neq 0$ (we can take $\hat{x} = e_1$ and $\hat{y}$ to be any vector such that $Ae_1 + B\hat{y} \leq c$). Since we cannot have more than n linearly independent vectors in span(P), this procedure is repeated at most n times. ■

Our next theorem is the main result of the section.

Theorem 6 *Given a safety region $S \subset \mathbb{R}^n$ and an uncertainty set $U_0 \subset \mathbb{R}^{n\times n}$, one-step safe learning is possible if and only if Algorithm 1 (with an arbitrary choice of $c \in \mathbb{R}^n$ and $\varepsilon \in (0, 1]$) returns a matrix.*

Proof ["If"] By construction, the sequence of measurements chosen by Algorithm 1 satisfies the first condition of Definition 2, since the vectors $x_k^\star$ and z_j are both contained in S_k^1 and any vector in S_k^1 will remain in the safety region under the action of all matrices in U_k; i.e. all matrices in U_A that are consistent with the measurements made so far. If Algorithm 1 terminates early at line 5 for some iteration k, then clearly the uncertainty set U_k is a singleton. On the other hand, if we reach line 27, then we must have made n linearly independent measurements $\{x_1, \ldots, x_n\}$. From this, it is clear that the set $\{A \in U_0 \mid Ax_j = y_j, j = 1, \ldots, n\} = \{A_\star\}$.

["Only if"] Suppose Algorithm 1 chooses points $\{x_1, \ldots, x_m\}$ where $m < n$ and terminates at line 20. Then it is clear from the algorithm that $\{x_1, \ldots, x_m\}$ must form a basis of span(S_m^1) and that U_m is not a singleton. Take $\tilde{m}$ to be any nonnegative integer and $\{\tilde{x}_1, \ldots, \tilde{x}_{\tilde{m}}\}$ to be any sequence that satisfies the first condition of Definition 2. For $k = 1, \ldots, \tilde{m}$, let

$$\tilde{U}_k = \{A \in U_0 \mid A\tilde{x}_j = A_\star \tilde{x}_j, j = 1, \ldots, k\},$$
$$\tilde{S}_k^1 = \{x \in S \mid Ax \in S, \forall A \in \tilde{U}_k\}.$$

First we claim that $\tilde{x}_k \in S_m^1$ for $k = 1, \ldots, \tilde{m}$. We show this by induction. It is clear that $\tilde{x}_1 \in S_m^1$ since $\tilde{x}_1 \in S_0^1$ and $S_0^1 \subseteq S_m^1$. Now we assume $\tilde{x}_1, \ldots, \tilde{x}_k \in S_m^1$ and show that $\tilde{x}_{k+1} \in S_m^1$. Since $\{x_1, \ldots, x_m\}$ forms a basis of span(S_m^1), it follows that for any matrix A, $Ax_j = A_\star x_j$ for $j = 1, \ldots, m$ implies $Ax = A_\star x$ for all $x \in S_m^1$. In particular, for any matrix A, $Ax_j = A_\star x_j$ for $j = 1, \ldots, m$ implies $A\tilde{x}_j = A_\star \tilde{x}_j$ for all $j = 1, \ldots, k$. It follows that $U_m \subseteq \tilde{U}_k$ and therefore, $\tilde{S}_k^1 \subseteq S_m^1$. By the first condition of Definition 2, we must have $\tilde{x}_{k+1} \in \tilde{S}_k^1$, and thus, $\tilde{x}_{k+1} \in S_m^1$. This completes the inductive argument and shows that $\tilde{x}_k \in S_m^1$ for $k = 1, \ldots, \tilde{m}$. From this, it follows that $U_m \subseteq \tilde{U}_{\tilde{m}}$. Recall that U_m is not a singleton, thus $\tilde{U}_{\tilde{m}}$ is not a singleton either. Therefore, the sequence $\{\tilde{x}_1, \ldots, \tilde{x}_{\tilde{m}}\}$ does not satisfy the second condition of Definition 2. ■

Corollary 7 *Given a safety region $S \subset \mathbb{R}^n$ and an uncertainty set $U_0 \subset \mathbb{R}^{n\times n}$, if one-step safe learning is possible, then it is possible with at most n measurements.*

3.3. The Value of Exploiting Information on the Fly

In addition to detecting the possibility of safe learning, Algorithm 1 attempts to minimize the overall cost of learning (i.e., $\sum_{k=1}^{m} c^T x_k$) by exploiting information gathered at every step. In order to demonstrate the value of using information online, we construct an offline algorithm which chooses n measurement vectors $x_1, \ldots, x_n$ ahead of time based solely on U_0 and S, and succeeds under the assumption that S_0^1 contains a basis of $\mathbb{R}^n$.

Algorithm 2: Offline Safe Learning Algorithm

Input : polyhedra $S \subset \mathbb{R}^n$ and $U_0 \subset \mathbb{R}^{n \times n}$, cost vector $c \in \mathbb{R}^n$, and a constant $\varepsilon \in (0, 1]$.
Output: A matrix $A_\star \in \mathbb{R}^{n \times n}$ or failure.

if S_0^1 *does not contain a basis of* $\mathbb{R}^n$ *(cf. Theorem 5)* **then**
  return failure
end
Compute a basis $\{z_1, \ldots, z_n\} \subset S_0^1$ of $\mathbb{R}^n$
Let $x_0^\star$ be the projection onto x-space of an optimal solution to problem (8) with data D_0
Set $x_k = (1 - \varepsilon)x_0^\star + \varepsilon z_k$ for $k = 1, \ldots, n$
Observe $y_k \leftarrow A_\star x_k$ for $k = 1, \ldots, n$
Define matrix $X = [x_1, \ldots, x_n]$
Define matrix $Y = [y_1, \ldots, y_n]$
return $A_\star = YX^{-1}$

As ε tends to zero, the cost of Algorithm 2 approaches $nc^T x_0^\star$, where $x_0^\star$ is a minimum cost measurement vector in S_0^1. Therefore, $nc^T x_0^\star$ serves as an *upper bound* on the cost incurred by Algorithm 1. We note that $nc^T x_0^\star$ is also the minimum cost achievable by any one-step safe offline algorithm that takes n measurements, since all measurement vectors $\{x_k\}$ of such an algorithm must come from S_0^1.

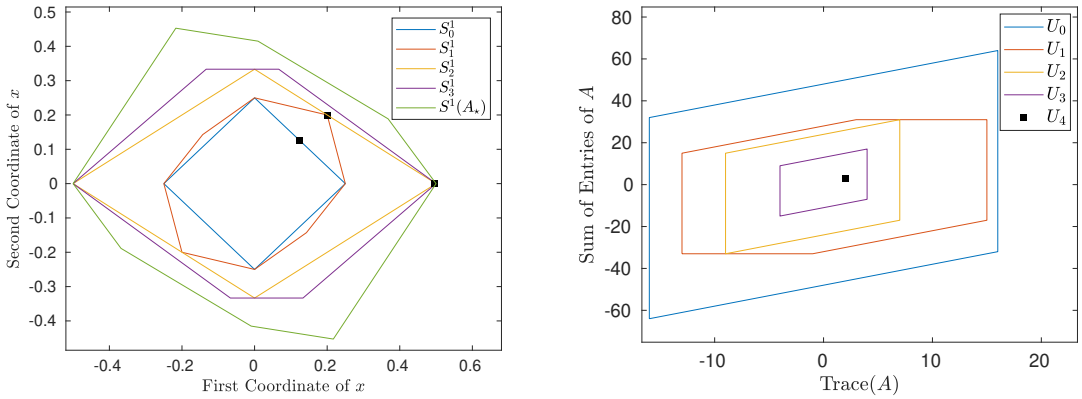
By assuming that we know $A_\star$, we can also compute a *lower bound* on the cost of one-step safe learning of any algorithm that takes n measurements. Let $S^1(A_\star) = \{x \in S \mid A_\star x \in S\}$ be the true one-step safety region of $A_\star$. Let $x^\star$ be an optimal solution to the linear program that minimizes $c^T x$ over $S^1(A_\star)$. Then, clearly, if we must pick n points that are all one-step safe, we cannot achieve cost lower than $nc^T x^\star$.

3.4. Numerical Example

We present a numerical example with $n = 4$. Here, we take $U_0 = \{A \in \mathbb{R}^{4\times 4} \mid |A_{ij}| \leq 4 \quad \forall i, j\}$, $S = \{x \in \mathbb{R}^4 \mid \|x\|_\infty \leq 1\}$, and $c = (-1, -1, 0, 0)^T$. We choose the matrix $A_\star$ uniformly at random among integer matrices in U_0:

$$A_\star = \begin{bmatrix} 2 & 1 & 4 & 2 \\ 2 & -3 & -1 & -2 \\ -2 & -3 & 1 & 0 \\ 2 & 0 & -2 & 2 \end{bmatrix}.$$

In this example, Algorithm 1 takes four steps to safely recover $A_\star$. The projection onto the first two dimensions of the four vectors that Algorithm 1 selects are plotted in Figure 1(*a*) (note that



(a) S_k^1 grows with k.

(b) U_k shrinks with k.

Figure 1: One-step safe learning associated with the numerical example in Section 3.4.

two of the points are very close to each other). Because of the cost vector c, points higher and further to the right in the plot have lower measurement cost. Also plotted in Figure 1(a) are the projections onto the first two dimensions of the sets S_k^1 for $k \in \{0, 1, 2, 3\}$ and of the set $S^1(A_\star)$, the true one-step safety region of $A_\star$. In Figure 1(b), we plot U_k (the remaining uncertainty after making k measurements) for $k \in \{0, 1, 2, 3, 4\}$; we draw a two-dimensional projection of these sets of matrices by looking at the trace and the sum of the entries of each matrix in the set. Note that U_4 is a single point since we have recovered the true dynamics after the fourth measurement.

The cost of learning (i.e., $\sum_{i=1}^4 c_i^T x_i$) is -1.0000 for the offline algorithm (Algorithm 2), and -1.6385 for Algorithm 1. The lower bound on the cost of learning is -2.2264 (cf. Section 3.3). We can see that the value of exploiting information on the fly is significant.

4. Two-Step Safe Learning of Linear Systems

In this section, we again focus on learning the linear dynamics in (3). However, unlike the previous section, we are interested in making queries to the system that are two-step safe. The advantage of this formulation is that we may have fewer system resets and can potentially learn the dynamics with lower measurement cost.

More formally, in the two-step safe learning problem, we have as input a polyhedral safety region $S \subset \mathbb{R}^n$ given in the form of (5), an objective function representing measurement cost which for simplicity we again take to be a linear function $c^T x$, and an uncertainty set $U_0 \subset \mathbb{R}^{n \times n}$ to which we assume $A_\star$ belongs. In this section, we take U_0 to be an ellipsoid; this means that there is a strictly convex quadratic function $q : \mathbb{R}^{n \times n} \to \mathbb{R}$ such that:

$$U_0 = \left\{ A \in \mathbb{R}^{n \times n} \mid q(A) \leq 0 \right\}.$$

An example of such an uncertainty set is $U_0 = \{A \in \mathbb{R}^{n \times n} \mid \|A - A_0\|_F \leq \gamma\}$, where A_0 is a nominal matrix, γ is a positive scalar, and $\|\cdot\|_F$ refers to the Frobenius norm. Having collected k safe length-two trajectories $\{(x_j, A_\star x_j, A_\star^2 x_j)\}_{j=1}^k$, our uncertainty around $A_\star$ reduces to:

$$U_k = \{A \in U_0 \mid Ax_j = A_\star x_j, A^2 x_j = A_\star^2 x_j, j = 1, \ldots, k\}. \tag{14}$$

The optimization problem we would like to solve to find the next best two-step safe query point is the following:

$$\begin{aligned} \min_{x} \quad & c^T x \\ \text{s.t.} \quad & x \in S \\ & Ax \in S \quad \forall A \in U_k \\ & A^2 x \in S \quad \forall A \in U_k. \end{aligned} \tag{15}$$

4.1. Reformulation via the S-Lemma

In this subsection, we derive a tractable reformulation of problem (15).

Theorem 8 *Problem (15) can be reformulated as a semidefinite program.*

Our proof makes use to the S-lemma (see e.g., Pólik and Terlaky, 2007) which we recall next.

Lemma 9 (S-lemma) *For two quadratics functions q_a and q_b, if there exists a point $\bar{x}$ such that $q_a(\bar{x}) < 0$, then the implication*

$$\forall x, [q_a(x) \leq 0 \Rightarrow q_b(x) \leq 0]$$

holds if and only if there exists a scalar $\lambda \geq 0$ such that

$$\lambda q_a(x) - q_b(x) \geq 0 \quad \forall x.$$

Proof [of Theorem 8] Note that the set of equations

$$Ax_j = A_\star x_j, \quad A^2 x_j = A_\star^2 x_j \quad j = 1, \ldots, k$$

in the definition of U_k in (14) is equivalent to the set of linear equations

$$Ax_j = A_\star x_j, \quad A(A_\star x_j) = A_\star^2 x_j \quad j = 1, \ldots, k. \tag{16}$$

If there is only one matrix in U_0 that satisfies all of the equality constraints in (16) (a condition that can be checked via a simple modification of Lemma 4), then we have found $A_\star$ and (15) becomes a linear program. Therefore, let us assume that more than one matrix in U_0 satisfies the constraints in (16). In order to apply the S-lemma, we need to remove these equality constraints, a task that we accomplish via variable elimination. Let $\hat{n}$ be the dimension of the affine subspace of matrices that satisfy the constraints in (16) and let $\hat{A} \in \mathbb{R}^{n \times n}$ be an arbitrary member of this affine subspace. Let $A_1, \ldots, A_{\hat{n}} \in \mathbb{R}^{n \times n}$ be a basis of the subspace

$$\{A \in \mathbb{R}^{n \times n} \mid Ax_j = 0, \quad A(A_\star x_j) = 0 \quad j = 1, \ldots, k\}.$$

Consider an affine function $g : \mathbb{R}^{\hat{n}} \to \mathbb{R}^{n \times n}$ defined as follows:

$$g(\hat{a}) := \hat{A} + \sum_{i=1}^{\hat{n}} \hat{a}_i A_i.$$

The function g has the properties that it is injective and that for each A that satisfies the equality constraints, there must be a vector $\hat{a}$ such that $A = g(\hat{a})$. In other words, the function g is simply

parametrizing the affine subspace of matrices that satisfy the equality constraints. Now we can reformulate (15) as:

$$\begin{aligned} \min_{x} \quad & c^T x \\ \text{s.t.} \quad & x \in S \\ & g(\hat{a})x \in S \quad \forall \hat{a} \quad \text{s.t.} \quad q(g(\hat{a})) \leq 0 \\ & g(\hat{a})^2 x \in S \quad \forall \hat{a} \quad \text{s.t.} \quad q(g(\hat{a})) \leq 0. \end{aligned} \tag{17}$$

Let $\hat{q} := q \circ g$. Since q is a strictly convex quadratic function and g is an injective affine map, $\hat{q}$ is also a strictly convex quadratic function. Since we are under the assumption that there are multiple matrices in U_0 that satisfy the equality constraints, there must be a vector $\bar{a} \in \mathbb{R}^{\hat{n}}$ such that $\hat{q}(\bar{a}) < 0$. To see this, take $\bar{a}_1 \neq \bar{a}_2$ such that $\hat{q}(\bar{a}_1), \hat{q}(\bar{a}_2) \leq 0$. It follows from strict convexity of $\hat{q}$ that $\hat{q}(\frac{1}{2}(\bar{a}_1 + \bar{a}_2)) < 0$. Using the definition of S, problem (17) can be rewritten as:

$$\begin{aligned} \min_{x} \quad & c^T x \\ \text{s.t.} \quad & h_i^T x \leq b_i \quad i = 1, \ldots, r \\ & \begin{bmatrix} \max_{\hat{a}} & h_i^T g(\hat{a}) x \\ \text{s.t.} & \hat{q}(\hat{a}) \leq 0 \end{bmatrix} \leq b_i \quad i = 1, \ldots, r \\ & \begin{bmatrix} \max_{\hat{a}} & h_i^T g(\hat{a})^2 x \\ \text{s.t.} & \hat{q}(\hat{a}) \leq 0 \end{bmatrix} \leq b_i \quad i = 1, \ldots, r. \end{aligned} \tag{18}$$

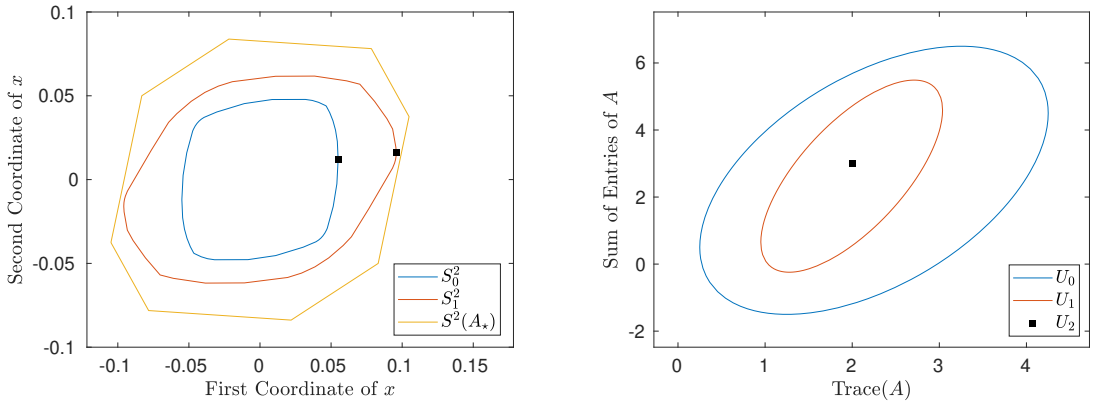
Let $q_{1,i}(\hat{a}; x) = h_i^T g(\hat{a}) x - b_i$ and $q_{2,i}(\hat{a}; x) = h_i^T g(\hat{a})^2 x - b_i$. We consider these functions as quadratic functions of $\hat{a}$ parametrized by x. Note that the coefficients of $q_{1,i}$ and $q_{2,i}$ depend affinely on x. Using logical implications, problem (18) can be rewritten as:

$$\begin{aligned} \min_{x} \quad & c^T x \\ \text{s.t.} \quad & h_i^T x \leq b_i \quad i = 1, \ldots, r \\ & \forall \hat{a}, [\hat{q}(\hat{a}) \leq 0 \Rightarrow q_{1,i}(\hat{a}; x) \leq 0] \quad i = 1, \ldots, r \\ & \forall \hat{a}, [\hat{q}(\hat{a}) \leq 0 \Rightarrow q_{2,i}(\hat{a}; x) \leq 0] \quad i = 1, \ldots, r. \end{aligned} \tag{19}$$

Now we use the S-lemma to reformulate an implication between quadratic inequalities as a constraint on the global nonnegativity of a quadratic function. Note that as we have already argued for the existence of a vector $\bar{a}$ such that $\hat{q}(\bar{a}) < 0$, the condition of the S-lemma is satisfied. After introducing variables $\lambda_{1,i}$ and $\lambda_{2,i}$ for $i = 1, \ldots, r$, we apply the S-lemma $2r$ times to reformulate (19) as the following program:

$$\begin{aligned} \min_{x,\lambda} \quad & c^T x \\ \text{s.t.} \quad & h_i^T x \leq b_i \quad i = 1, \ldots, r \\ & \lambda_{1,i} \hat{q}(\hat{a}) - q_{1,i}(\hat{a}; x) \geq 0 \quad \forall \hat{a} \quad i = 1, \ldots, r \\ & \lambda_{2,i} \hat{q}(\hat{a}) - q_{2,i}(\hat{a}; x) \geq 0 \quad \forall \hat{a} \quad i = 1, \ldots, r \\ & \lambda_{1,i} \geq 0, \quad \lambda_{2,i} \geq 0 \quad i = 1, \ldots, r. \end{aligned} \tag{20}$$

It is a standard procedure to convert the constraint that a quadratic function is globally nonnegative into a semidefinite constraint. Note that the coefficients of $q_{1,i}$ and $q_{2,i}$ depend affinely on x; this results in linear matrix inequalities when (20) is converted into a semidefinite program. ■



(*a*) S_k^2 grows with k.

(*b*) U_k shrinks with k.

Figure 2: Two-step safe learning associated with the numerical example in Section 4.2.

4.2. Numerical Example

We present a numerical example, again with $n = 4$. Here we choose a nominal matrix

$$A_0 = \begin{bmatrix} 2.25 & 0.75 & 4.25 & 1.75 \\ 2.25 & -3.25 & -1.25 & -2.25 \\ -2.00 & -2.75 & 1.25 & 0.00 \\ 1.75 & -0.25 & -2.00 & 2.00 \end{bmatrix}$$

and let $U_0 = \{A \in \mathbb{R}^{4\times 4} \mid \|A - A_0\|_F \leq 1\}$. We let $S = \{x \in \mathbb{R}^4 \mid |x_i| \leq 1, i = 1, \ldots, 4\}$ and $c = (-1, 0, 0, 0)^T$. We choose the true matrix $A_\star$ to be the same matrix used in Section 3.4.

In this example, by solving two semidefinite programs, we learn the true matrix $A_\star$ by making two measurements that are each two-step safe. In other words, we choose $x_1 \in \mathbb{R}^4$, observe $A_\star x_1$, $A_\star^2 x_1$, and then choose $x_2 \in \mathbb{R}^4$, and observe $A_\star x_2$ and $A_\star^2 x_2$. We can verify that we have recovered $A_\star$ if $\{x_1, A_\star x_1, x_2, A_\star x_2\}$ are all linearly independent, which is the case. The projection onto the first two dimensions of the two measurements x_1 and x_2 that our semidefinite programs choose are plotted in Figure 2(*a*). Because of the cost vector c, points further to the right in the plot have lower measurement cost. Also plotted are the projections onto the first two dimensions of the sets:

$$\begin{aligned} S_0^2 &= \{x \in S \mid Ax \in S, A^2x \in S \quad \forall A \in U_0\}, \\ S_1^2 &= \{x \in S \mid Ax \in S, A^2x \in S \quad \forall A \in U_1\}, \\ S^2(A_\star) &= \{x \in S \mid A_\star x \in S, A_\star^2 x \in S\}. \end{aligned}$$

The first two sets are the projections onto x-space of the feasible regions of our two semidefinite programs. The third set is the true two-step safety region of $A_\star$. In Figure 2(*b*), we plot U_k (the remaining uncertainty after observing k trajectories of length two) for $k \in \{0, 1, 2\}$; we draw a two-dimensional projection of these sets of matrices by looking at the trace and the sum of the entries of each matrix in the set. Note that U_2 is a single point since we have recovered the true dynamics after observing the second trajectory. The cost of learning (i.e., $c^T x_1 + c^T x_2$) is -0.1508.

We can construct an analogue of the offline Algorithm 2 by only making measurements from S_0^2. This approach would first pick the optimal point in S_0^2 (i.e., x_1), and then another vector in S_0^2 close to x_1, but linearly independent from it. The cost of learning for this offline approach would be $2c^T x_1 = -0.1099$. Finally, we can again find a lower bound on the cost of learning of any algorithm that makes two measurements (that are each two-step safe) by assuming we know $A_\star$ ahead of time and optimizing $c^T x$ over $S^2(A_\star)$; in this example, the lower bound is -0.2097. Here, again, we see that by using information on the fly, we can succeed at safe learning at a lower cost than the offline approach.

5. One-Step Safe Learning of Nonlinear Systems

We consider the problem of safely learning a dynamical system of the form $x_{t+1} = f_\star(x_t)$, where

$$f_\star(x) = A_\star x + g_\star(x), \tag{21}$$

for some matrix $A_\star \in \mathbb{R}^{n\times n}$ and some possibly nonlinear map $g_\star : \mathbb{R}^n \to \mathbb{R}^n$. We take our safety region $S \subset \mathbb{R}^n$ to be the same as (5). Our initial knowledge about $A_\star, g_\star$ is membership in the sets

$$\begin{aligned} U_{0,A} &:= \left\{A \in \mathbb{R}^{n\times n} \mid \mathrm{Tr}(V_j^T A) \le v_j \quad j = 1,\dots,s\right\}, \\ U_{0,g} &:= \{g : \mathbb{R}^n \to \mathbb{R}^n \mid \|g(x)\|_\infty \le \gamma \|x\|_p^d \quad \forall x \in S\}. \end{aligned}$$

Here, $p \ge 1$ is either $+\infty$ or a rational number, γ is a given positive constant, and d is a given nonnegative integer. The use of the $\|\cdot\|_\infty$ on g in the definition of $U_{0,g}$ simplifies some of the following analysis, though an extension to other semidefinite representable norms is possible. Note that by taking $d = 0$ e.g., our model of uncertainty captures any map f which is bounded on S.

Again for simplicity, we assume a linear measurement cost $c^T x$ for some vector $c \in \mathbb{R}^n$. Having collected k safe measurements $\{(x_j, y_j)\}_{j=1}^k$ with $y_j = f_\star(x_j)$, the optimization problem we are interested in solving to find the next cheapest one-step safe measurement is:

$$\begin{aligned} \min_x \quad & c^T x \\ \text{s.t.} \quad & x \in S \\ & Ax + g(x) \in S \quad \forall\, (A,g) \in \{A \in U_{0,A}, g \in U_{0,g} \mid Ax_j + g(x_j) = y_j \quad j = 1,\dots,k\}. \end{aligned} \tag{22}$$

5.1. Reformulation as a Second-Order Cone Program

Our main result of this section is to derive a tractable reformulation of problem (22).

Theorem 10 *Problem (22) can be reformulated as a second-order cone program.*

Proof We start by rewriting problem (22) using the definition of S:

$$\begin{aligned} \min_x \quad & c^T x \\ \text{s.t.} \quad & h_i^T x \le b_i \quad i = 1,\dots,r \\ & \begin{bmatrix} \max_{A,g} & h_i^T(Ax + g(x)) \\ \text{s.t.} & \mathrm{Tr}(V_j^T A) \le v_j \quad j = 1,\dots,s \\ & \|g(x)\|_\infty \le \gamma \|x\|_p^d \quad \forall x \in S \\ & Ax_k + g(x_k) = y_k \quad k = 1,\dots,m \end{bmatrix} \le b_i \quad i = 1,\dots,r. \end{aligned} \tag{23}$$

Note that in the inner maximization problem in (23), the variable x is fixed. We claim that if $x \notin \{x_1, \ldots, x_k\}$, then

$$\begin{bmatrix} \max_{A,g} & h_i^T(Ax + g(x)) \\ \text{s.t.} & \text{Tr}(V_j^T A) \le v_j \quad j = 1, \ldots, s \\ & \|g(x)\|_\infty \le \gamma \|x\|_p^d \quad \forall x \in S \\ & Ax_k + g(x_k) = y_k \quad k = 1, \ldots, m \end{bmatrix} \tag{24}$$

$$= \begin{bmatrix} \max_{A,g} & h_i^T Ax \\ \text{s.t.} & \text{Tr}(V_j^T A) \le v_j \quad \forall j \\ & Ax_k + g(x_k) = y_k \quad \forall k \\ & \|g(x)\|_\infty \le \gamma \|x\|_p^d \quad \forall x \in S \end{bmatrix} + \begin{bmatrix} \max_{A,g} & h_i^T g(x) \\ \text{s.t.} & \text{Tr}(V_j^T A) \le v_j \quad \forall j \\ & Ax_k + g(x_k) = y_k \quad \forall k \\ & \|g(x)\|_\infty \le \gamma \|x\|_p^d \quad \forall x \in S \end{bmatrix}.$$

It is clear that the left-hand side is upper bounded by the right-hand side. To show the reverse inequality, let (A_1, g_1) (resp. (A_2, g_2)) be feasible to the first (resp. second) problem on the right-hand side (if any of these of these problems is infeasible, then the inequality we are after is immediate). Now let

$$\hat{g}_2(x) = \begin{cases} g_2(x) & \text{if } x \notin \{x_1, \ldots, x_k\} \\ y_k - A_1 x_k & \text{if } x = x_k. \end{cases}$$

It is straightforward to check that the pair $(A_1, \hat{g}_2)$ is feasible to the left-hand side of (24), therefore proving (24).

We now focus on reformulating each term on the right-hand side of (24), again under the assumption that $x \notin \{x_1, \ldots, x_m\}$. Using the constraint on g, the first term can be rewritten as follows:

$$\begin{aligned} \max_{A} \quad & h_i^T Ax \\ \text{s.t.} \quad & \text{Tr}(V_j^T A) \le v_j \quad j = 1, \ldots, s \\ & \|Ax_k - y_k\|_\infty \le \gamma \|x_k\|_p^d \quad k = 1, \ldots, m. \end{aligned} \tag{25}$$

Note that (25) is a linear program as it is equivalent to:

$$\begin{aligned} \max_{A} \quad & h_i^T Ax \\ \text{s.t.} \quad & \text{Tr}(V_j^T A) \le v_j \quad j = 1, \ldots, s \\ & (Ax_k - y_k)_l \le \gamma \|x_k\|_p^d \quad k = 1, \ldots, m \quad l = 1, \ldots, n \\ & -(Ax_k - y_k)_l \le \gamma \|x_k\|_p^d \quad k = 1, \ldots, m \quad l = 1, \ldots, n. \end{aligned} \tag{26}$$

Here, the notation $(Ax_k - y_k)_l$ represents the l-th coordinate of the vector $(Ax_k - y_k)$. Following the same approach as in Section 3, we proceed by taking the dual of this linear program. For $j = 1, \ldots, s$, $k = 1, \ldots, m$, and $l = 1, \ldots, n$, let $\mu_j, \eta_{kl}^+, \eta_{kl}^- \in \mathbb{R}$ be dual variables. The dual of problem (26) reads:

$$\begin{aligned} \min_{\mu, \eta^+, \eta^-} \quad & \sum_j \mu_j v_j + \sum_{kl} \eta_{kl}^+ (\gamma \|x_k\|_p^d + (y_k)_l) + \sum_{kl} \eta_{kl}^- (\gamma \|x_k\|_p^d - (y_k)_l) \\ \text{s.t.} \quad & x h_i^T = \sum_j \mu_j V_j^T + \sum_{kl} \eta_{kl}^+ x_k e_l^T - \sum_{kl} \eta_{kl}^+ x_k e_l^T \\ & \mu \ge 0, \quad \eta^+ \ge 0, \quad \eta^- \ge 0, \end{aligned} \tag{27}$$

where e_l is the l-th coordinate vector. Now we turn our attention to the second term on the right-hand side of (24). After eliminating the irrelevant constraints, the problem can be rewritten as:

$$\begin{aligned} \max_{g} \quad & h_i^T g(x) \\ \text{s.t.} \quad & \|g(x)\|_\infty \leq \gamma \|x\|_p^d. \end{aligned} \tag{28}$$

Recall that the dual norm of $\|\cdot\|_\infty$ is $\|\cdot\|_1$. Therefore, the optimal value of this optimization problem is simply $\gamma \|h_i\|_1 \cdot \|x\|_p^d$.

Now consider the optimization problem:

$$\begin{aligned} \min_{x,\mu,\eta^+,\eta^-} \quad & c^T x \\ \text{s.t.} \quad & h_i^T x \leq b_i \quad i = 1, \ldots, r \\ & \sum_j \mu_j v_j + \sum_{kl} \eta_{kl}^+ (\gamma \|x_k\|_p^d + (y_k)_l) \\ & \qquad + \sum_{kl} \eta_{kl}^- (\gamma \|x_k\|_p^d - (y_k)_l) + \gamma \|h_i\|_1 \cdot \|x\|_p^d \leq b_i \quad i = 1, \ldots, r \\ & \mu \geq 0, \quad \eta^+ \geq 0, \quad \eta^- \geq 0. \end{aligned} \tag{29}$$

If $d = 0$, or if $d = 1$ and $p \in \{1, +\infty\}$, then (29) is a linear program. Otherwise, the rationality of p ensures that $\|x\|_p^d$ is *second-order cone representable* (see Ben-Tal and Nemirovski, 2001, Sect. 2.3; Lobo et al., 1998, Sect. 2.5). This means that (29) is indeed a second-order cone program.

Let $F \subset \mathbb{R}^n$ denote the projection of the feasible set of (29) onto x-space. We claim that the feasible set of (22) equals $F \cup \{x_1, \ldots, x_k\}$. Indeed, since the vectors x_k are one-step safe measurements, we have that $x_k \in S$ and $y_k \in S$. This implies that x_k is feasible to (22). Furthermore, for $x \in F \setminus \{x_1, \ldots, x_k\}$, we have shown that x satisfies the constraints of (23) if and only if x satisfies the constraints of (29).

Therefore, optimizing an objective function over the feasible set of (22) is equivalent to optimizing the same objective function over $F \cup \{x_1, \ldots, x_k\}$. ■

5.2. Numerical Example

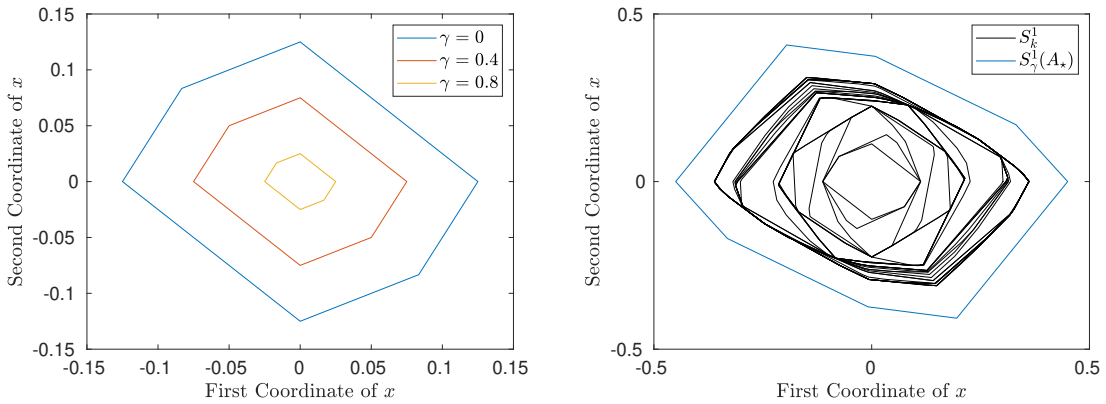
We present a numerical example, again with $n = 4$. Here we take:

$$\begin{aligned} S &= \{x \in \mathbb{R}^4 \mid |x_i| \leq 1, i = 1, \ldots, 4\}, \\ U_{0,A} &= \{A \in \mathbb{R}^{4\times 4} \mid -4 \leq A_{ij} \leq 8, i = 1, \ldots, 4, j = 1, \ldots, 4\}, \\ U_{0,g} &= \{g : \mathbb{R}^4 \to \mathbb{R}^4 \mid \|g(x)\|_\infty \leq \gamma \quad \forall x \in S\}. \end{aligned}$$

In Figure 3(*a*), we plot S_0^1 (the one-step safety region without any measurements) projected onto the first two dimensions of x for $\gamma \in \{0, 0.4, 0.8\}$. As expected, larger values of γ result in smaller one-step safety regions.

For our next experiment, we choose the matrix $A_\star$ in (21) to be the same matrix used in the example in Section 3.4. We let $\gamma = 0.1$, and

$$g_\star(x) = \frac{\gamma}{2} \left(x_2^2 - x_3 x_4, \quad \sqrt{x_1^4 + x_3^4}, \quad x_3 \sin^2(x_1), \quad \sin^2(x_2) \right)^T \in U_{0,g}.$$



(a) Dependence of S_0^1 on γ.

(b) S_k^1 grows with k.

Figure 3: One-step safe learning of a nonlinear system associated with the example in Section 5.2.

Since the true system is not linear, we cannot hope to learn the exactly dynamics in n steps as we did in the linear case. We instead pick 30 one-step safe points $x_1, \ldots, x_{30}$ (by sequentially solving the second-order cone program from Theorem 10) and observe $y_k = f_\star(x_k)$ for each $k = 1, \ldots, 30$. In order to encourage exploration of the state space, we optimize in random directions in every iteration (instead of optimizing the same cost function throughout the process). In Figure 3(b), we plot S_k^1 (the one-step safety region after k measurements) projected onto the first two dimensions of x for $k = 0, \ldots, 30$. We also plot the projection of $S_\gamma^1(A_\star)$, which we define as the set of one-step safe points if we knew $A_\star$, but not $g_\star$:

$$S_\gamma^1(A_\star) := \{x \in S \mid A_\star x + g(x) \in S \quad \forall g \in U_{0,g}\}.$$

Finally, we undertake the task of learning the unknown nonlinear dynamics. We only use information from our first 8 data points in order to make the fitting task more challenging. We fit a function of the form

$$\hat{f}(x) = \hat{A}x + \hat{g}(x),$$

where $\hat{A} \in \mathbb{R}^{4\times 4}$ and each entry of $\hat{g} : \mathbb{R}^4 \to \mathbb{R}^4$ is a homogeneous quadratic function of x. Our regression is done by minimizing the least-squares loss function

$$L(\hat{f}) = \sum_{k=1}^{8} \|\hat{f}(x_k) - y_k\|^2.$$

We train two models. The first model, $\hat{f}_{\text{ls}}$, minimizes the least-squares loss with no constraints. The second model, $\hat{f}_{\text{SOS}}$, minimizes the least-squares loss subject to the constraints that $\hat{A} \in U_{0,A}$, $\|\hat{A}x_k - y_k\|_\infty \leq \gamma$ for $k = 1, \ldots, 8$, and $\hat{g} \in U_{0,g}$. The constraint that $\hat{g} \in U_{0,g}$ is imposed via sum of squares constraints (see, e.g., Parrilo, 2000; Ahmadi and Khadir, 2020 for details). More specifically, we require that for $j = 1, \ldots, 4$,

$$\gamma \pm \hat{g}_j(x) = \sigma_0^{j,\pm}(x) + \sum_{i=1}^{r} \sigma_i^{j,\pm}(x)(b_i - h_i^T x) \quad \forall x \in \mathbb{R}^4.$$

Here, $\hat{g}_j(x)$ is the j-th entry of the vector $\hat{g}(x)$, and the functions $\sigma_i^{j,\pm}$, for $i = 0, \ldots, r$ and $j = 1, \ldots, 4$, are sum of squares quadratic functions of x. These constraints can be imposed by semidefinite programming.

We sample test points $z_1, \ldots, z_{1000}$ uniformly at random in S in order to estimate the generalization error. The root-mean-square error (RMSE) is computed as:

$$\text{RMSE}(\hat{f}) = \sqrt{\frac{1}{1000} \sum_{j=1}^{1000} \|\hat{f}(z_i) - f_\star(z_i)\|^2}.$$

The $\text{RMSE}(\hat{f}_{\text{SOS}})$ of the constrained model is 0.0851 and the $\text{RMSE}(\hat{f}_{\text{ls}})$ of the unconstrained model is 0.2567. We see that imposing prior knowledge with sum of squares constraints results in a significantly better fit.

Acknowledgments

AAA and AC were partially supported by the MURI award of the AFOSR, the DARPA Young Faculty Award, the CAREER Award of the NSF, the Google Faculty Award, the Innovation Award of the School of Engineering and Applied Sciences at Princeton University, and the Sloan Fellowship.

References

Amir Ali Ahmadi and Oktay Günlük. Robust-to-dynamics optimization. *arXiv:1805.03682*, 2018.

Amir Ali Ahmadi and Bachir El Khadir. Learning dynamical systems with side information. In *Proceedings of Machine Learning Research*, volume 120, pages 718–727. PMLR, 10–11 Jun 2020.

Anayo K. Akametalu, Jaime F. Fisac, Jeremy H. Gillula, Shahab Kaynama, Melanie N. Zeilinger, and Claire J. Tomlin. Reachability-based safe learning with gaussian processes. In *53rd IEEE Conference on Decision and Control*, 2014.

James Anderson, John C. Doyle, Steven H. Low, and Nikolai Matni. System level synthesis. *Annual Reviews in Control*, 47:364–393, 2019.

Aharon Ben-Tal and Arkadi Nemirovski. *Lectures on Modern Convex Optimization*. Society for Industrial and Applied Mathematics, 2001.

Felix Berkenkamp, Matteo Turchetta, Angela P. Schoellig, and Andreas Krause. Safe model-based reinforcement learning with stability guarantees. In *Neural Information Processing Systems*, 2017.

Yohann De Castro, Fabrice Gamboa, Didier Henrion, Roxana Hess, and Jean-Bernard Lasserre. Approximate optimal designs for multivariate polynomial regression. *Ann. Statist.*, 47(1):127–155, 02 2019.

Sarah Dean, Stephen Tu, Nikolai Matni, and Benjamin Recht. Safely learning to control the constrained linear quadratic regulator. In *2019 American Control Conference (ACC)*, 2019.

Lukasz Kaiser, Mohammad Babaeizadeh, Piotr Milos, Blazej Osinski, Roy H Campbell, Konrad Czechowski, Dumitru Erhan, Chelsea Finn, Piotr Kozakowski, Sergey Levine, et al. Model-based reinforcement learning for Atari. *arXiv:1903.00374*, 2019.

Torsten Koller, Felix Berkenkamp, Matteo Turchetta, Joschka Boedecker, and Andreas Krause. Learning-based model predictive control for safe exploration and reinforcement learning. *arXiv:1906.12189*, 2019.

Miguel Sousa Lobo, Lieven Vandenberghe, Stephen Boyd, and Hervé Lebret. Applications of second-order cone programming. *Linear Algebra and its Applications*, 284(1):193–228, 1998.

Kendall Lowrey, Aravind Rajeswaran, Sham Kakade, Emanuel Todorov, and Igor Mordatch. Plan online, learn offline: Efficient learning and exploration via model-based control. *arXiv:1811.01848*, 2018.

Tyler Lu, Martin Zinkevich, Craig Boutilier, Binz Roy, and Dale Schuurmans. Safe exploration for identifying linear systems via robust optimization. *arXiv:1711.11165*, 2017.

Anusha Nagabandi, Gregory Kahn, Ronald S Fearing, and Sergey Levine. Neural network dynamics for model-based deep reinforcement learning with model-free fine-tuning. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018.

Pablo Parrilo. *Structured semidefinite programs and semialgebraic geometry methods in robustness and optimization*. PhD thesis, California Institute of Technology, 2000.

Friedrich Pukelsheim. *Optimal Design of Experiments*. Society for Industrial and Applied Mathematics, 2006.

Imre Pólik and Tamás Terlaky. A survey of the S-lemma. *SIAM Review*, 49(3):371–418, 2007.

Sumeet Singh, Spencer M. Richards, Vikas Sindhwani, Jean-Jacques E. Slotine, and Marco Pavone. Learning stabilizable nonlinear dynamics with contraction-based regularization. *arXiv:1907.13122*, 2019.

Arun Venkatraman, Roberto Capobianco, Lerrel Pinto, Martial Hebert, Daniele Nardi, and J Andrew Bagnell. Improved learning of dynamics models for control. In *International Symposium on Experimental Robotics*, pages 703–713. Springer, 2016.

Kim P. Wabersich and Melanie N. Zeilinger. Linear model predictive safety certification for learning-based control. In *2018 IEEE Conference on Decision and Control (CDC)*, 2018.

Yuxiang Yang, Ken Caluwaerts, Atil Iscen, Tingnan Zhang, Jie Tan, and Vikas Sindhwani. Data efficient reinforcement learning for legged robots. In *Conference on Robot Learning*, pages 1–10. PMLR, 2020.