2023 62nd IEEE Conference on Decision and Control (CDC)
December 13-15, 2023. Marina Bay Sands, Singapore

Data-Driven Control with Inherent Lyapunov Stability

Youngjae Min[1], Spencer M. Richards[2], Navid Azizan[1]

Abstract— **Recent advances in learning-based control leverage deep function approximators, such as neural networks, to model the evolution of controlled dynamical systems over time. However, the problem of learning a dynamics model and a stabilizing controller persists, since the synthesis of a stabilizing feedback law for known nonlinear systems is a difficult task, let alone for complex parametric representations that must be fit to data. To this end, we propose *Control with Inherent Lyapunov Stability* (CoILS), a method for jointly learning parametric representations of a nonlinear dynamics model and a stabilizing controller from data. To do this, our approach simultaneously learns a parametric Lyapunov function which intrinsically constrains the dynamics model to be stabilizable by the learned controller. In addition to the stabilizability of the learned dynamics guaranteed by our novel construction, we show that the learned controller stabilizes the true dynamics under certain assumptions on the fidelity of the learned dynamics. Finally, we demonstrate the efficacy of CoILS on a variety of simulated nonlinear dynamical systems.**

I. Introduction

Data-driven approaches have shown notable successes in solving complex nonlinear control problems in robotics and autonomy, such as autonomous navigation, multi-agent control, and object grasping and manipulation [8], [11], [12]. In learning-based control problems, the task of controller synthesis is often compounded with a lack of knowledge, or uncertainty, about the system dynamics. To this end, neural networks have been widely used for system identification from data prior to controller synthesis [4], [7], [14], [15], [24]. However, when using such complex parametric representations to model dynamics, it can be challenging to provide any guarantees about the behavior of the learned system, particularly regarding the stability of the system under closed-loop feedback.

A traditional method for stabilizing nonlinear dynamical systems is linearizing the system dynamics around an equilibrium point and using the linear quadratic regulator (LQR) techniques to minimize deviation from that equilibrium. LQR methods can achieve closed-loop stability within a small region where the linear dynamics approximation is accurate, yet away from this region, they can fail spectacularly, particularly for highly nonlinear systems performing agile maneuvers [22]. Nonlinear controllers that are *certified* to be globally stabilizing can be synthesized if a control Lyapunov function (CLF) for the system is known [20]. However, constructing a CLF even for known dynamics can be difficult to do exactly, and thus approximate approaches are popular. For example, polynomial approximations of the dynamics enable a search for sum-of-squares (SOS) polynomials as Lyapunov functions via semidefinite programming (SDP) [23]. However, polynomial approximation can be a significant restriction on the class of function approximators used.

A. Related Work

To this end, there has been substantial growth in literature on *data-driven* learning of stability certificates for dynamical systems, particularly those modeled with complex parametric representations. One of the earliest of such works fits a Lyapunov function for a known uncontrolled dynamical system by penalizing violations of the corresponding Lyapunov decrease condition [16]. Learning certificate functions such as Lyapunov, barrier, and contraction metric functions for dynamical systems with *sampled* point penalties or constraints on stability violations is a ubiquitous theme in the literature [2], [5], [6], [10], [18]. However, when coupled with regression on unknown dynamics, such approaches do not even guarantee the learned model is stable. To resolve this issue, [13] recently proposed a method to jointly learn a stable uncontrolled dynamics model with a Lyapunov function. It guarantees stability of the learned dynamics model by restricting it to the stable halfspace described by the Lyapunov decrease condition. However, its naive extension to *controlled* dynamics would restrict the learned model to be stabilizable by any control input, thereby hindering its ability to model general controlled nonlinear systems.

For controlled dynamics, the paradigm of sampled point-wise constraints largely persists in the literature. [19] jointly learn a dynamics model and certificate to regularize the dynamics model to perform well over long time horizons with sampled linear matrix inequality (LMI) constraints, yet they do not learn any specific controller. [21] assume the dynamics are known, and jointly learn a controller and a contraction metric certifying the stability of the closed-loop system with loss terms corresponding to sampled point violations of a stability inequality. For Lyapunov-based approaches, [3] jointly learn a controller and a Lyapunov function for known dynamics, while they actively add training samples that violate the Lyapunov decrease condition using a falsifier. [25] extend this method to unknown control systems with some stability guarantees, but their method requires certain knowledge on the system such as its Lipschitz constant and linearized model around the origin. Moreover, their learned dynamics do not guarantee the existence of the Lyapunov function. [9] directly apply the approach from [13] to the

[1]Youngjae Min and Navid Azizan are with the Laboratory for Information and Decision Systems, Massachusetts Institute of Technology, Cambridge, MA 02139, USA. {yjm, azizan}@mit.edu

[2]Spencer M. Richards is with the Autonomous Systems Laboratory (ASL), Stanford University, Stanford, CA 94305, USA spenrich@stanford.edu

controlled case, albeit only for control-affine systems when the actuator matrix is known. All in all, efficient simultaneous learning of Lyapunov functions and nonlinear controllers for systems with completely unknown dynamics has remained an open problem in control and robotics.

B. Contributions

In this work, we tackle the difficult task of learning stabilizing feedback controllers with proven guarantees for unknown nonlinear dynamical systems. To this end, we propose *Control with Inherent Lyapunov Stability* (CoILS), a new method for jointly learning a controlled dynamical systems model and a feedback controller from data, such that the model is *guaranteed by construction* to be stabilized in closed-loop with the learned controller. CoILS does this by simultaneously learning a parametric Lyapunov function, which is used to constrain the open-loop dynamics onto the subspace of dynamics stabilizable in closed-loop by the learned controller. We further show that, under certain assumptions on the fidelity of our learned dynamics model, the learned controller is also guaranteed to stabilize the true dynamics. Finally, we demonstrate the performance of our joint learning method in a number of controlled nonlinear dynamical systems.

II. Problem Statement

In this paper, we are interested in controlling the unknown nonlinear dynamical system

$$\dot{x}(t) = f(x(t), u(t)) \tag{1}$$

with state $x(t) \in \mathcal{X} \subset \mathbb{R}^n$ and control input $u(t) \in \mathcal{U} \subset \mathbb{R}^m$ at time $t \in \mathbb{R}$. While we do not know the dynamics $f : \mathcal{X} \times \mathcal{U} \to \mathbb{R}^n$, we assume we have access to a finitely-sized dataset $\mathcal{D} = \{(x_i, u_i, \dot{x}_i)\}_{i=1}^N$ of input-output measurements of the system (1).

We want to *jointly* learn the dynamics f and how to control them. Specifically, we want to *stabilize* the system around an equilibrium point. The point $x_e \in \mathcal{X}$ is an *equilibrium* of the closed-loop system $f_{u^*}(x) \coloneqq f(x, u^*(x))$ with feedback controller $u^* : \mathcal{X} \to \mathcal{U}$ if

$$f_{u^*}(x_e) = f(x_e, u^*(x_e)) = 0. \tag{2}$$

There are many types of stability; we summarize the pertinent ones in the definition below for uncontrolled and closed-loop systems, i.e., where the dynamics are a function of the state x only.

Definition II.1. The system $\dot{x} = f(x)$ for $x \in \mathcal{X}$ is *stable* at its equilibrium point $x_e \in \mathcal{X}$ if for any $\epsilon > 0$, there exists $\delta_\epsilon > 0$ such that $\|x(0) - x_e\|_2 < \delta_\epsilon$ implies $\|x(t) - x_e\|_2 < \epsilon$ for all $t \geq 0$. The system is *asymptotically stable* at x_e w.r.t. $B \subset \mathcal{X}$ if it is stable at x_e and $\lim_{t\to\infty} \|x(t) - x_e\|_2 = 0$ for all $x(0) \in B$. The system is *exponentially stable* at x_e w.r.t. $B \subset \mathcal{X}$ if there exist $m, \alpha > 0$ such that $\|x(t) - x_e\|_2 \leq m\|x(0) - x_e\|_2 e^{-\alpha t}$ for all $x(0) \in B$.

For the controlled system (1), we assume a feedback controller $u^* : \mathcal{X} \to \mathcal{U}$ exists such that the resulting closed-loop system $\dot{x} = f_{u^*}(x)$ is exponentially stable at x_e w.r.t. $\mathcal{X}$. Without loss of generality, we assume $x_e = 0$. Thus, our overall goal is to *jointly* learn the dynamics f and an exponentially stabilizing feedback controller u^*. As we discuss in Section III, our approach relies on encoding this stabilizability by u^* in f by construction. However, the stability definitions in Definition II.1 are cumbersome to work with directly, so, we appeal to Lyapunov stability theory to introduce scalar-value functions that summarize stability properties of dynamical systems [1], [17].

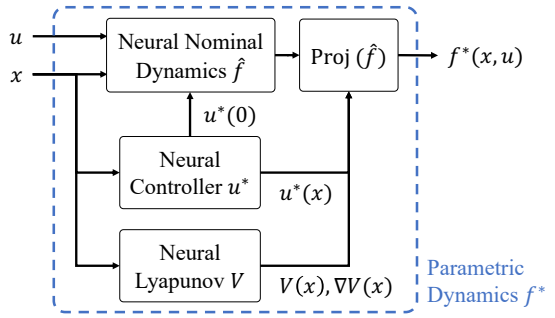

Fig. 1. Schematic diagram of CoILS. It projects the nominal dynamics model $\hat{f}$ to satisfy the stability by construction through incorporating the feedback controller u^* and the Lyapunov function V in the architecture.

Proposition II.2. *Consider the system $\dot{x} = f(x)$ where $x \in \mathcal{X}$. Suppose there exists a continuously differentiable function $V : \mathcal{X} \to \mathbb{R}$ that is positive definite (i.e., $V(x) > 0$ for $x \neq 0$ and $V(0) = 0$), and satisfies*

$$\nabla_f V(x) \coloneqq \nabla V(x)^\top f(x) < 0, \tag{3}$$

for all $x \in \mathcal{X} \setminus \{0\}$. Then the system is asymptotically stable at $x = 0$ w.r.t. $\mathcal{X}$.

Such a function V is termed a *Lyapunov function*. The key idea is that by condition (3), V is decreasing along any trajectories generated by f and eventually converges to 0, which implies $x = 0$. The existence of a Lyapunov function with additional properties is also a necessary and sufficient condition for exponential stability as follows.

Proposition II.3. *Consider the system $\dot{x} = f(x)$ where $x \in \mathcal{X}$. This system is exponentially stable at $x = 0$ w.r.t. $\mathcal{X}$ if and only if there exists a continuously differentiable function $V : \mathcal{X} \to \mathbb{R}$ such that*

$$c_1\|x\|_2^2 \leq V(x) \leq c_2\|x\|_2^2, \quad \nabla_f V(x) \leq -\alpha V(x), \tag{4}$$

for all $x \in \mathcal{X} \setminus \{0\}$ and some constants $\alpha, c_1, c_2 > 0$

When Propositions II.2 and II.3 are restated for a closed-loop system $\dot{x} = f_u(x)$ with controller $u : \mathcal{X} \to \mathcal{U}$, the accompanying Lyapunov function V is also known as a *control Lyapunov function (CLF)*. In the next section, we use Proposition II.3 to constrain a model of f to be exponentially stabilizable by a parametric controller u^* as guaranteed by an accompanying parametric Lyapunov function V.

III. Joint Learning of Dynamics, Controller and Lyapunov Function

In this paper, we propose a novel architecture shown in Figure 1 that satisfies the Lyapunov stability conditions

by construction. Given the difficulty, if not infeasibility, of synthesizing a Lyapunov function for a separately learned dynamics model, we instead consider jointly learning them. Specifically, we propose a parameterization of the dynamics model that incorporates a given parametric Lyapunov function and controller. This connection allows us to jointly optimize them by applying automatic differentiation from a single loss function to fit the dataset.

More specifically, we construct the dynamics model f^* by projecting a parametric *nominal* model $\hat{f} : \mathcal{X} \times \mathcal{U} \to \mathbb{R}^n$ according to a parametric Lyapunov function $V : \mathcal{X} \to \mathbb{R}$ and a parametric feedback controller $u^* : \mathcal{X} \to \mathcal{U}$. This projection constrains the dynamics model f^* to satisfy the Lyapunov stability condition

$$\nabla_{f^*_{u^*}} V(x) = \nabla V(x)^\top f^*(x, u^*(x)) \leq -\alpha V(x) \tag{5}$$

for all $x \in \mathcal{X} \setminus \{0\}$ with a given $\alpha > 0$. The construction of the dynamics model is as follows.

$$f^*(x,u) = \begin{cases} \hat{f}(x,u) & \text{if } x = 0 \\ \mathsf{Proj}_{u^*(x),V(x),\nabla V(x)}\left(\hat{f}(x,\cdot)\right)(u) & \text{o.w.} \end{cases} \tag{6}$$

where

$$\begin{aligned} &\mathsf{Proj}_{u^*(x),V(x),\nabla V(x)}\left(\hat{f}(x,\cdot)\right) \\ &:= \hat{f}(x,\cdot) + \underset{\Delta f \in \mathbb{R}^n}{\arg\min} \quad \|\Delta f\|_2 \\ &\qquad\qquad \text{s.t.} \quad f^*(x,\cdot) = \hat{f}(x,\cdot) + \Delta f \\ &\qquad\qquad\quad\ \ \nabla V(x)^\top f^*(x, u^*(x)) \leq -\alpha V(x) \\ &= \hat{f}(x,\cdot) - \nabla V(x) \frac{\mathsf{ReLU}\left(\nabla V(x)^\top \hat{f}(x, u^*(x)) + \alpha V(x)\right)}{\|\nabla V(x)\|_2^2} \end{aligned} \tag{7}$$

provided $\nabla V(x) \neq 0$ for $x \neq 0$.

The interpretation of the projection is as follows. For all $x \in \mathcal{X} \setminus \{0\}$, if the nominal model $\hat{f}$ satisfies the condition (5), it needs no modification so that $f^*(x,u) = f(x,u)$ for all $u \in \mathcal{U}$; on the other hand, if it violates the condition, we minimally adjust it (in the sense of ℓ^2 norm) so that the projected model f^* barely satisfies the condition (5) with the equality. This projected model is then guaranteed to be globally exponentially stable at the origin with the feedback controller u^* by construction, as described in Section IV.

We employ neural networks to represent the nominal dynamics model $\hat{f}$, the feedback controller u^*, and the Lyapunov function V. To ensure that the conditions required for each of these components are met, we carefully choose the architectures of their corresponding neural networks, as described next.

A. Nominal Dynamics Model and Controller

First, we should make sure that the origin is an equilibrium point for the closed-loop system $f^*_{u^*}$, i.e., $f^*(0, u^*(0)) = \hat{f}(0, u^*(0)) = 0$. This could be ensured by setting the nominal dynamics model as

$$\hat{f}(x,u) = g_f(x,u) - g_f(0, u^*(0)) \tag{8}$$

where $g_f : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ is an unconstrained neural network with an arbitrary architecture.

For the controller, we require a candidate controller to have its range confined to a pre-specified set $\mathcal{U}$. The control inputs are often generated by motors in most robotics applications and thus are saturated by physical limitations. To handle such prevalent scenarios, we consider $\mathcal{U} = \{u \in \mathbb{R}^m : -u_{\lim} \leq u \leq u_{\lim}\}$ for some $u_{\lim} \geq 0$. Then, we limit the range of the feedback controller by setting

$$u^*(x) = \mathsf{diag}(u_{\lim}) \tanh(g_u(x)), \tag{9}$$

where $\mathsf{diag}(u_{\lim})$ is a diagonal matrix with its diagonal being $u_{\lim}$, and $g_u : \mathbb{R}^n \to \mathbb{R}^m$ is an unconstrained neural network.

B. Lyapunov Function

The eligibility conditions for the Lyapunov function are threefold. First, V should be positive definite, i.e., $V(x) > 0$ for $x \neq 0$ and $V(0) = 0$. We ensure this condition by defining V in the form of

$$V(x) = \sigma(g_V(x) - g_V(0)) + \epsilon_{\text{pd}} \|x\|^2 \tag{10}$$

for some $\epsilon_{\text{pd}} > 0$, where $g_V : \mathbb{R}^n \to \mathbb{R}$ is a neural network and $\sigma : \mathbb{R} \to \mathbb{R}$ is a function with $\sigma(w) = 0$ for all $w \leq 0$.

Further, V should be continuously differentiable. We secure this property by using the smoothed ReLU function

$$\sigma(z) = \begin{cases} 0 & \text{if } x \leq 0 \\ x^2/2d & \text{if } 0 < x < d \\ x - d/2 & \text{otherwise} \end{cases} \tag{11}$$

in (10) and as the activations for the neural network g_V.

Lastly, V should not have non-zero critical points since its time derivative is strictly negative for any non-zero state. One way to ensure this property is using an input-convex neural network (ICNN) for g_V as introduced by [13]. Together with the strictly increasing σ, V is strictly convex and free of non-zero critical points. However, Lyapunov functions are not necessarily convex, and using an ICNN restricts the model capacity of V. Instead, we consider using an unconstrained neural network for g_V and approximate the projection in (7) as

$$\hat{f}(x,u) - \nabla V(x) \frac{\mathsf{ReLU}\left(\nabla_{\hat{f}} V(x, u^*(x)) + \alpha V(x)\right)}{\max(\|\nabla V(x)\|_2^2, \epsilon_{\text{proj}})} \tag{12}$$

for some small $\epsilon_{\text{proj}} > 0$. Once the network has been trained, (5) with positive definite $V(x)$ would imply $\nabla V(x)$ to be non-zero due to the strictly negative $\nabla_{f^*_{u^*}} V(x)$. Then, the approximation (12) would also recover the original projection.

C. Loss Function

We jointly learn the three components by minimizing the following loss function:

$$\mathcal{L}(\theta) = \frac{1}{N} \sum_{(x,u,\dot{x}) \in \mathcal{D}} k(u, u^*(x)) \|\dot{x} - f^*(x,u)\|^2 + \lambda \|\theta\|_2^2 \tag{13}$$

where θ, $k(\cdot,\cdot)$, and λ denote the parameters for the neural networks, a kernel function, and the regularization coefficient,

respectively. The kernel function $k : \mathcal{U} \times \mathcal{U} \to \mathbb{R}$ attains a larger value as the two arguments are getting closer. By weighting the ℓ^2 difference, we encourage u^* to learn a policy that generates trajectories with smaller model error. Conversely, we also encourage f^* to learn better around feasible trajectories, i.e., $(x, u^*(x))$. In this work, we use the following kernel function:

$$k(u, u') = 1 + \exp(-\beta \|u - u'\|_2^2). \tag{14}$$

IV. Stability Guarantees

In this section, we rigorously analyze the stability properties of the learned models. We first establish the inherent stability of the constructed model f^* with the feedback controller u^* (Theorem IV.1). Then, we connect the inherent stability to the behavior of the true system f depending on the learning performance (Theorem IV.2).

Theorem IV.1 (Inherent Stability)**.** *Consider a closed-loop system $f^*_{u^*}$ for a projected dynamics f^* defined by (6) with $\hat{f}$ in (8), u^* in (9), and V in (10). For any $r_2 \geq r_1 > 0$, let $B_{r_1,r_2} := \{x \in \mathcal{X} : r_1 \leq \|x\|_2 \leq r_2\}$. Then,*
1) $f^*_{u^*}$ is exponentially stable at the origin w.r.t. B_{r_1,r_2}.
2) $f^*_{u^*}$ is exponentially stable at the origin w.r.t. $\mathcal{X}$ if there exists $c > 0$ such that $V(x) \leq c\|x\|_2^2$ for all $x \in \mathcal{X}$.

CoILS inherently guarantees the exponential stability of the learned models, as proven in Theorem IV.1. Note that this property holds even for any initialization of the models. Next, we further argue about its connection to the true dynamics. We could expect this stability property to be transferred to the true dynamics when satisfactory learning performance is achieved. We elaborate on this relationship in the next theorem.

Theorem IV.2. *Consider a dynamical system f in (1) and its learned model f^* in (6) with $\hat{f}$ in (8), u^* in (9), and V in (10). Assume f and f^* are Lipschitz continuous with Lipschitz constants L_f and L_{f^*}, respectively. Let δ and e denote how densely and accurately, respectively, f^* is learned by defining*

$$\begin{aligned} \delta &:= \sup_{x \in \mathcal{X}} \min_{(y,v) \in \mathcal{D}} \|(x, u^*(x)) - (y, v)\|_2, \\ e &:= \max_{(y,v) \in \mathcal{D}} \|f(y, v) - f^*(y, v)\|_2. \end{aligned} \tag{15}$$

For any $r > 0$, let the neighborhood of the origin $N_r := \{x \in \mathcal{X} : \|x\|_2 < r\}$ and $M_r := \sup_{x \in \mathcal{X} \setminus N_r} \|\nabla V(x)\|_2$. Then, the closed-loop system f_{u^*} always arrives at N_r, i.e., for any trajectory $x(t)$ generated by f_{u^*} from $x(0) \in \mathcal{X}$, there exists $T \geq 0$ such that $x(T) \in N_r$, if δ and e are small enough such that

$$(L_f + L_{f^*})\delta + e < \frac{\alpha \epsilon_{\text{pd}} r^2}{M_r}. \tag{16}$$

V. Experiments

In this section, we demonstrate the effectiveness of CoILS in stabilizing unknown nonlinear control systems. We first verify the stability properties of the projected models achieved by our proposed architecture (Section V-A). Then, we demonstrate the performance and accuracy of the learned controller and dynamics, respectively, in three different control problems (Section V-B-V-D).

TABLE I
Hyperparameters for the experiments

parameter	α	β	λ	ϵ_{pd}	ϵ_{proj}	d
value	1.0 or 5.0	$5/u_{\text{lim}}$	0.0	0.5	0.001	0.005

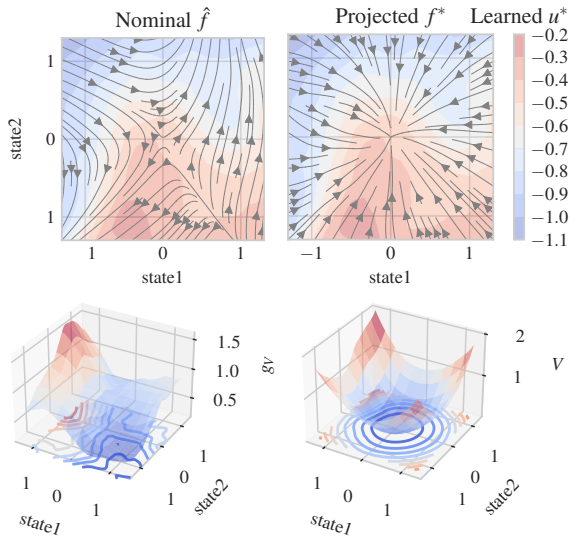

Fig. 2. Closed-loop dynamics and Lyapunov function for randomly initialized models. Top-left: Nominal model $\hat{f}$ with controller u^*. Top-right: Projected model f^* with controller u^*. Bottom-left: Neural network g_V. Bottom-right: Candidate Lyapunov function V.

For each scenario, we train the models with $N = 10^5$ data tuples which are uniformly sampled over $\mathcal{X} = \{x \in \mathbb{R}^n : x_{lb} \leq x \leq x_{ub}\}$ and $\mathcal{U} = \{u \in \mathbb{R}^m : -u_{\text{lim}} \leq u \leq u_{\text{lim}}\}$. The models are constructed with neural networks as described in Section III. We use 3-layer fully connected neural networks (FCN) for g_f, g_u, and g_V. They have 100, 50, and 50 hidden neurons, respectively, in each hidden layer. For g_V, we add tanh activation multiplied by 10 at the output layer to limit the scale of the Lyapunov function. The models are trained using a mini-batch gradient descent optimizer with gradient clipping and a learning rate 0.0001. The other hyperparameters used in the experiments are presented in Table I.

A. Random Networks

In this experiment, we investigate the efficacy of the projection with randomly initialized models. As proven in Theorem IV.1, we verify that the projected model f^* in (6) equips the inherent stability in closed-loop with the feedback controller u^* even for their random initialization. We initialize the neural networks for $\mathcal{X} \in \mathbb{R}^2$ and $\mathcal{U} \in \mathbb{R}$. The results are shown in Figure 2. The projected model f^* is stabilized at the origin by the controller u^*, while the nominal model $\hat{f}$ is not. As induced by (8), the nominal model $\hat{f}$ as well as the projected model f^* achieves zero value at the origin in closed-loop with u^*. This ensures that the origin is an equilibrium point of the closed-loop systems. Interestingly, we observe that the candidate Lyapunov function V does not

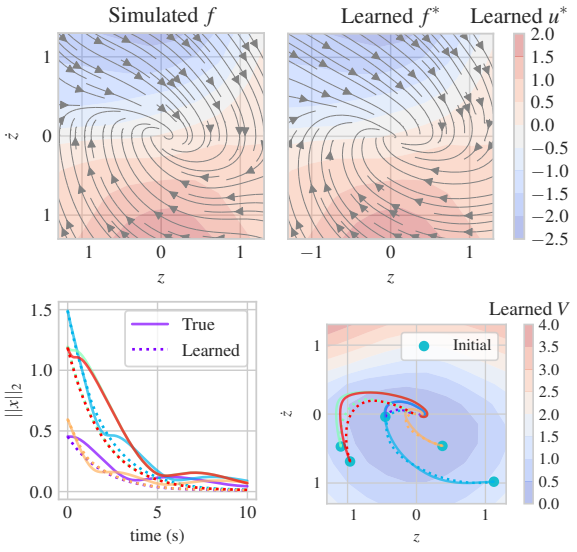

Fig. 3. Comparison of simulation and the learned model for the Van der Pol oscillator. Top-left: Closed-loop dynamics for the true model f with the learned controller u^*. Top-right: Closed-loop dynamics for the learned model f^* with the learned controller u^*. Bottom-left: Comparison of 5 randomly initialized trajectories for the true and learned system. Bottom-right: Plot of the trajectories in the state space with the contour map of the learned Lyapunov function.

have any critical points although we use a generic FCN for g_V in (10), as shown in the figure. This favored construction is observed in most cases.

B. Van der Pol Oscillator

We test our proposed method in three different control problems. We start with stabilizing a system, the Van der Pol oscillator, which is already stable at the origin without any control input. The autonomous Van der Pol oscillator is a well-known nonlinear system that exhibits stable oscillations in the form of a limit cycle. By adding a control input $u \in \mathbb{R}$, we consider the true dynamics as $\ddot{z} = u - z + \mu(1 - z^2)\dot{z}$ for state $x = [z, \dot{z}] \in \mathbb{R}^2$ with the parameter μ=1. The dataset is sampled with x_{lb}=$[-1.3, -1.3]$, x_{ub}=$[1.3, 1.3]$, $u_{\lim}$=5. The results are shown in Figure 3. The closed-loop dynamics for the true system f_{u^*} and the learned system $f^*_{u^*}$ show similar behaviors. Due to some model errors, the trajectories are not perfectly aligned, but the learned controller successfully stabilizes both true and learned systems to the origin.

C. Inverted Pendulum

Next, we demonstrate CoILS in a more challenging control problem, stabilizing an inverted pendulum. The inverted pendulum, shown in Figure 4, easily falls off the origin without proper control due to gravity. For angular position θ from the inverted position and its angular velocity $\dot{\theta}$, we consider states $x = [\theta, \dot{\theta}] \in \mathbb{R}^2$. By providing a torque $u \in \mathbb{R}$ at the pivot as a control input, we consider the true dynamics as $ml^2\ddot{\theta} = mgl \sin\theta + u - b\dot{\theta}$. We set the parameters $m = 0.15, g = 9.81, l = 0.5, b = 0.1$ with $x_{lb} = [-4, -4], x_{ub} = [4, 4], u_{\lim} = 5$. The results are shown in Figure 5. The overall behaviors of the closed-loop dynamics are similar for the true system f_{u^*} and the learned system

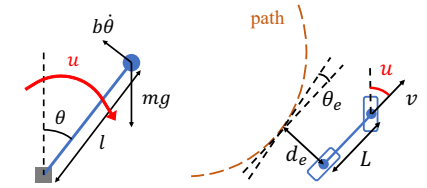

Fig. 4. Schematic diagrams for (left) the inverted pendulum and (right) the bicycle path following system.

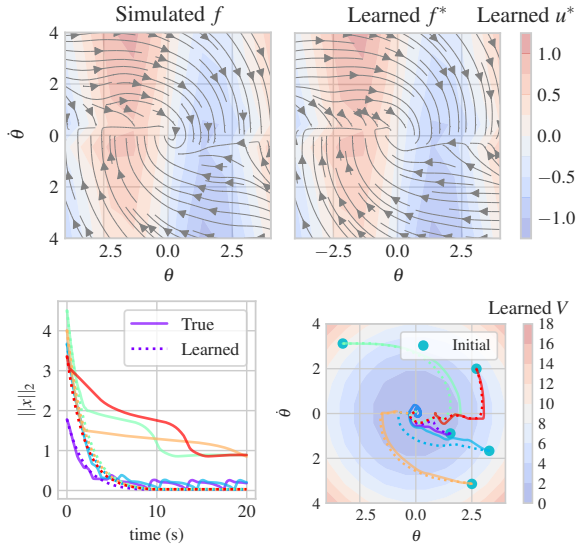

Fig. 5. Comparison of simulation and the learned model for the inverted pendulum system. Top-left: Closed-loop dynamics for the true model f. Top-right: Closed-loop dynamics for the learned model f^*. Bottom-left: Comparison of 5 randomly initialized trajectories. Bottom-right: Plot of the trajectories in the state space with the contour map of the learned Lyapunov function.

$f^*_{u^*}$. However, the trajectories exhibit quite dissimilar ℓ^2 norm evolution. This difference occurs from the model errors in the small area around $\dot{\theta} = 0$. The small magnitude of f_{u^*} in those areas slows down the movement of the state even though the direction of the movement is similar for both systems. These model errors could be improved if we utilize the physical relationship between the state elements, i.e., the second element is the derivative of the first element.

D. Bicycle Path Following

The previous examples are both control-affine systems, while CoILS is applicable to general control systems without such structure. In this experiment, we evaluate CoILS for a control system in which the control input has a nonlinear effect. We consider the problem of controlling a constant-speed bicycle to follow a unit circle path as shown in Figure 4. We aim to drive the distance error d_e and angular error θ_e to be zero by controlling the steering angle $u \in \mathbb{R}$. For the state $x = [d_e, \theta_e] \in \mathbb{R}^2$, the dynamics are given as $\dot{d}_e = v \sin\theta_e$, and $\dot{\theta}_e = (v \tan u)/L - (v \cos\theta_e)/(1 - d_e)$. We set the parameters $v = 6, L = 1$ with $x_{lb} = [-0.8, -0.8]$, $x_{ub} = [0.8, 0.8]$, $u_{\lim} = 0.4\pi$. The results are shown in Figure 6. The closed-loop dynamics for the true system f_{u^*} and the learned system $f^*_{u^*}$ are similar to each other. Also,

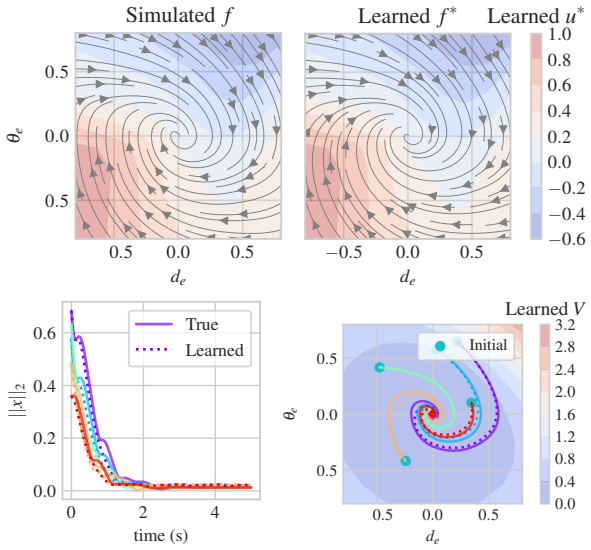

Fig. 6. Comparison of simulation and the learned model for the bicycle path following system. Top-left: Closed-loop dynamics for the true model f with the learned controller u^*. Top-right: Closed-loop dynamics for the learned model f^* with the learned controller u^*. Bottom-left: Comparison of 5 randomly initialized trajectories for the true and learned system. Bottom-right: Plot of the trajectories in the state space with the contour map of the learned Lyapunov function.

the learned controller generates trajectories close for both systems and successfully stabilizes both systems to the origin.

VI. Conclusion

We presented a new data-driven method CoILS to stabilize unknown controlled dynamical systems. We jointly learn the dynamics model and a feedback controller with a Lyapunov function, such that the projected model is *guaranteed by construction* to be stabilized in closed-loop by the learned controller. We further showed that, under certain assumptions on the fidelity of our learned dynamics model, the learned controller is also guaranteed to stabilize the true dynamics. We demonstrated the performance of our method in the simulation of a number of controlled nonlinear dynamical systems.

There are several interesting avenues for future work. First, it is feasible to explore different loss functions, for instance, to include control-oriented metrics. This work focused on learning the dynamics model without explicitly optimizing the performance of the controller. Since there exist various controllers that can stabilize the system in practice, guiding the learning to find one with better performance would be an interesting direction. Also, exploring different hyperparameters for our method would be beneficial to further optimize our approach. Furthermore, one can think about extending our approach to partially observable systems. It would be challenging but worthwhile to find an output-feedback controller that stabilizes unknown dynamical systems.

Acknowledgements

This work was supported in part by the MIT-IBM Watson AI Lab and by MathWorks.

References

[1] F. Blanchini and S. Miani. *Set-theoretic methods in control*, volume 78. Springer, 2008.
[2] N. M. Boffi, S. Tu, N. Matni, J.-J. E. Slotine, and V. Sindhwani. Learning stability certificates from data. In *Conf. on Robot Learning*, 2020.
[3] Y.-C. Chang, N. Roohi, and S. Gao. Neural lyapunov control. *Advances in neural information processing systems*, 32, 2019.
[4] S. Chen, S. A. Billings, and P. Grant. Non-linear system identification using neural networks. *International journal of control*, 51(6):1191–1214, 1990.
[5] C. Dawson, S. Gao, and C. Fan. Safe control with learned certificates: A survey of neural Lyapunov, barrier, and contraction methods. Available at https://arxiv.org/abs/2202.11762, 2022.
[6] B. El Khadir, J. Varley, and V. Sindhwani. Teleoperator imitation with continuous-time safety. In *Robotics: Science and Systems*, 2019.
[7] J. K. Gupta, K. Menda, Z. Manchester, and M. J. Kochenderfer. Structured mechanical models for robot learning and control. In *Learning for Dynamics & Control Conference*, 2020.
[8] L. Hewing, K. P. Wabersich, M. Menner, and M. N. Zeilinger. Learning-based model predictive control: Toward safe learning in control. *Annual Review of Control, Robotics, and Autonomous Systems*, 3:269–296, 2020.
[9] K. Kashima, R. Yoshiuchi, and Y. Kawano. Learning stabilizable deep dynamics models. Available at https://arxiv.org/abs/2203.09710, 2022.
[10] M. Khosravi and R. S. Smith. Nonlinear system identification with prior knowledge on the region of attraction. *IEEE Control Systems Letters*, 5(3):1091–1096, 2021.
[11] S. Levine, P. Pastor, A. Krizhevsky, J. Ibarz, and D. Quillen. Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection. *The International journal of robotics research*, 37(4-5):421–436, 2018.
[12] S. Liu, G. Lever, Z. Wang, J. Merel, S. A. Eslami, D. Hennes, W. M. Czarnecki, Y. Tassa, S. Omidshafiei, A. Abdolmaleki, et al. From motor control to team play in simulated humanoid football. *Science Robotics*, 7(69):eabo0235, 2022.
[13] G. Manek and J. Z. Kolter. Learning stable deep dynamics models. In *Conf. on Neural Information Processing Systems*, 2019.
[14] S. M. Richards, N. Azizan, J.-J. E. Slotine, and M. Pavone. Adaptive-control-oriented meta-learning for nonlinear systems. In *Robotics: Science and Systems*, 2021.
[15] S. M. Richards, N. Azizan, J.-J. E. Slotine, and M. Pavone. Control-oriented meta-learning. *Int. Journal of Robotics Research*, 2023. In press.
[16] S. M. Richards, F. Berkenkamp, and A. Krause. The Lyapunov neural network: Adaptive stability certification for safe learning of dynamical systems. In *Conf. on Robot Learning*, 2018.
[17] S. Sastry. *Nonlinear systems: analysis, stability, and control*, volume 10. Springer Science & Business Media, 2013.
[18] V. Sindhwani, S. Tu, and M. Khansari. Learning contracting vector fields for stable imitation learning. Available at https://arxiv.org/abs/1804.04878, 2018.
[19] S. Singh, S. M. Richards, V. Sindhwani, J.-J. E. Slotine, and M. Pavone. Learning stabilizable nonlinear dynamics with contraction-based regularization. *Int. Journal of Robotics Research*, 2020.
[20] J.-J. E. Slotine and W. Li. *Applied Nonlinear Control*. Prentice Hall, 1991.
[21] D. Sun, S. Jha, and C. Fan. Learning certified control using contraction metric. In *Conf. on Robot Learning*, 2020.
[22] S. Sun, A. Romero, P. Foehn, E. Kaufmann, and D. Scaramuzza. A comparative study of nonlinear MPC and differential-flatness-based control for quadrotor agile flight. *IEEE Transactions on Robotics*, 38(6):3357–3373, 2022.
[23] R. Tedrake, I. R. Manchester, M. Tobenkin, and J. W. Roberts. LQR-trees: Feedback motion planning via sums-of-squares verification. *Int. Journal of Robotics Research*, 29(8):1038–1052, 2010.
[24] Y. Wang. A new concept using lstm neural networks for dynamic system identification. In *2017 American control conference (ACC)*, pages 5324–5329. IEEE, 2017.
[25] R. Zhou, T. Quartz, H. De Sterck, and J. Liu. Neural lyapunov control of unknown nonlinear systems with stability guarantees. *Advances in Neural Information Processing Systems*, 2022.