IEEE Control Systems Letters paper presented at
2024 IEEE 63rd Conference on Decision and Control (CDC)
December 16-19, 2024. MiCo, Milan, Italy

Actor-Critic Physics-Informed Neural Lyapunov Control

Jiarui Wang and Mahyar Fazlyab, *Member, IEEE*

***Abstract*— Designing control policies for stabilization tasks with provable guarantees is a long-standing problem in nonlinear control. A crucial performance metric is the size of the resulting region of attraction, which essentially serves as a robustness "margin" of the closed-loop system against uncertainties. In this letter, we propose a new method to train a stabilizing neural network controller along with its corresponding Lyapunov certificate, aiming to maximize the resulting region of attraction while respecting the actuation constraints. Crucial to our approach is the use of Zubov's Partial Differential Equation (PDE), which precisely characterizes the true region of attraction of a given control policy. Our framework follows an actor-critic pattern where we alternate between improving the control policy (actor) and learning a Zubov function (critic). Finally, we compute the largest certifiable region of attraction by invoking an SMT solver after the training procedure. Our numerical experiments on several design problems show consistent and significant improvements in the size of the resulting region of attraction.**

I. Introduction

Neural network control policies have shown great promise in model-based control and outperformed existing design methods, owing to their capability to capture intricate nonlinear policies [1] [2] [3] [4]. For example, neural network control policies trained with reinforcement learning algorithms have been able to outperform human champions in drone racing [4]. Despite the outstanding empirical performance, the application of neural network policies in physical systems is of concern due to a lack of stability and safety guarantees.

Certificate functions such as Lyapunov functions provide a general framework for designing nonlinear control policies with provable guarantees. For instance, a quadratic Lyapunov function can be derived using a Linear Quadratic Regulator (LQR) [5] for linear systems, while a polynomial Lyapunov function can be obtained through Sum-of-Squares (SoS) optimization [6] for polynomial systems. However, as systems become more complex and nonlinear, the automated construction of Lyapunov functions becomes an increasingly crucial research focus within control theory [7], [8], [9], [10].

Learning-based methods combined with neural networks have been a promising alternative to optimization-based methods in constructing Lyapunov certificates for autonomous systems [8][11][12] or co-learning a neural network control policy together with a neural Lyapunov function [13], [14], [15]. In these learning-based methods, a typical approach is to minimize the violation of the Lyapunov conditions at a finite number of sampled states, and then invoke a verifier such as SMT solvers [16] to extend the validity of the learned certificate beyond the sampled states. Although applicable to more general nonlinear systems, the choice of the training loss function is critical to the learner's success. For example, for the case of stability analysis, we are interested in certifying the largest region of attraction, which is missing in a training objective that merely penalizes the Lyapunov condition violations.

Our Contribution: In this letter, we propose an algorithmic framework for co-learning a neural network control policy and a neural network Lyapunov function for actuation-constrained nonlinear systems. Our starting point is to use a physics-informed loss function based on Zubov's Partial Differential Equation (PDE), the solution of which characterizes the ground truth domain of attraction (DoA) for a given control policy. Our framework then follows an actor-critic pattern where we alternate between learning a Zubov function, by minimizing the PDE residual, and improving the control policy, by minimizing its Lie derivative akin to Sontag's formula [17]. To improve the efficiency of the method, we provide an implementation that essentially runs the actor and critic updates in parallel. Finally, we propose a novel method based on the softmax function to enforce polytopic actuation constraints without any projection layer. Our numerical experiments on several stabilization tasks show that our approach can enlarge the DoA significantly compared to state-of-the-art techniques. The source code is available at `https://github.com/bstars/ZubovControl`.

A. Related Work

Learning Lyapunov Functions: There is a large body of work on learning Lyapunov functions for autonomous systems [8] [11] [12] [18]. In the work of Abate et al [8], they proposed a counter-example guided framework to learn a neural Lyapunov function. A common feature of these methods is that the level set of the learned Lyapunov function does not match the true DoA, leading to a conservative estimation of the DoA despite using neural networks. Raissi et al [19] used Zubov's PDE as a training loss to capture the maximum region of attraction for autonomous systems. Our method also utilizes Zubov's equation to *co-learn* the control policy and the stability certificate together.

Co-learning Lyapunov Functions and Policies: Chang et al. [13] proposed a counter-example guided method similar to [8] to co-learn a Lyapunov function and a control policy that jointly satisfy the Lyapunov conditions for nonlinear systems. In their work, they add a regularizer to the loss function to enlarge the DoA. The method in [13] has also been extended

Jiarui Wang is with the Computer Science Department, Johns Hopkins University, Baltimore, MD 21218 USA(e-mail:jwang486@jhu.edu)

Mahyar Fazlyab is with the Electrical and Computer Engineering Department, Johns Hopkins University, Baltimore, MD 21218 USA(e-mail:mahyarfazlyab@jhu.edu)

to safety certificates [14], unknown nonlinear systems [15] and discrete-time systems [20]. In our method, regularization is unnecessary as the Zubov equation captures the true DoA.

Actor-critic and Policy Iteration: Our framework is also inspired by the actor-critic method [21] [22] in reinforcement learning. Actor-critic methods consist of a control policy and a critic that evaluates the value function of the policy. Then the algorithm switches between evaluating the current policy and improving the policy by maximizing the value function.

Actor-critic methods can be seen as an approximation of policy iteration [22]. There have also been works that combine neural Lyapunov functions with policy iteration to learn formally verifiable control policies. In the work of [23], they use policy iteration to learn control policies for control-affine systems. The policy evaluation part is done by representing the value function by a physics-informed neural network and minimizing the violation of the Generalized Hamilton-Jacobi-Bellman equation [24], and the policy improvement is done by minimizing the value function. Our method also follows a policy-evaluation-policy-improvement paradigm as in the actor-critic method and policy iteration.

The rest of the letter is organized as follows. In section II, we provide background and problem statement. In section III, we describe how to co-learn a control policy and a Lyapunov function with a physics-informed neural network and the verification procedure after training. We provide numerical experiments in section IV.

B. Notation

We denote n-dimensional real vectors as $\mathbb{R}^n$. For $x \in \mathbb{R}^n$, $\|x\|$ denotes the Euclidean norm. $[x_1, ..., x_n]$ represents a matrix with columns $x_1, ..., x_n$. For a set $\mathcal{A}$, $\partial\mathcal{A}$ denotes the boundary of $\mathcal{A}$ and $\mathcal{A} \setminus \{0\}$ denotes the set $\mathcal{A}$ excluding the single point 0. For a vector $x \in \mathbb{R}^n$, we use the notation $(x)_+ = \max(x, 0)$. $\mathbf{sg}(\cdot)$ stands for "stop gradient", meaning that the argument in $\mathbf{sg}(\cdot)$ operator is treated as constant. (e.g., $\frac{d}{dx}(x \cdot \mathbf{sg}(ax)) = ax$.

II. Background and Problem Statement

A. Lyapunov Certificates

Consider the autonomous nonlinear continuous-time dynamical system

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

where $x \in \mathcal{D} \subseteq \mathbb{R}^n$ is the state and $f\colon \mathcal{D} \to \mathbb{R}^n$ is a Lipschitz continuous function. For an initial condition $x_0 \in \mathcal{D}$, we denote the unique solution by $x(t; x_0)$, which we assume exists for all $t \geq 0$. We assume that the origin is an equilibrium of the system, $f(0) = 0$.

Definition 1 (Stability[25]): The zero solution $x(t) = 0$ to (1) is stable if for any $\epsilon > 0$, there exists $\delta > 0$ such that if $\|x_0\| < \delta$, then $\|x(t; x_0)\| < \epsilon$ for all $t > 0$.

Definition 2 (Asymptotic Stability[25]): The zero solution $x(t) = 0$ to (1) is locally asymptotically stable if it is stable and there exists a $\delta > 0$ such that if $\|x_0\| < \delta$, then $\lim_{t\to\infty} \|x(t; x_0)\| = 0$.

Definition 3 (Domain of Attraction [25]): If the zero solution $x(t) = 0$ to (1) is asymptotically stable, then the domain of attraction (DoA) is given by $\mathcal{A} := \big\{x_0 \in \mathcal{D} : \lim_{t\to\infty} \|x(t; x_0)\| = 0\big\}$.

A common way to certify the stability of an equilibrium and estimate the DoA is via Lyapunov functions.

Theorem 1 (Lyapunov Stability[25]): Consider the dynamical system in (1). If there exists a continuously differentiable function $V\colon \mathcal{D} \to \mathbb{R}$ satisfying the following conditions

$$V(0) = 0 \tag{2a}$$

$$V(x) > 0 \ \forall \ x \in \mathcal{D} \setminus \{0\} \tag{2b}$$

$$\nabla V(x)^\top f(x) < 0 \ \forall x \in \mathcal{D} \setminus \{0\} \tag{2c}$$

then the zero solution $x(t) = 0$ is asymptotically stable.

Any sub-level set $\mathcal{D}_\beta = \{x : V(x) \leq \beta\} \subseteq \mathcal{D}$ containing the origin is a subset of DoA [25], and the largest $\mathcal{D}_\beta$ can be calculated by maximizing β subject to $\mathcal{D}_\beta \subseteq \mathcal{D}$.

B. Problem Statement

Consider the nonlinear continuous-time dynamical system

$$\dot{x} = f(x, u) \tag{3}$$

where $x \in \mathcal{D} \subseteq \mathbb{R}^n$ is the state, $u \in \mathcal{U}$ is the control input, $\mathcal{U} \subseteq \mathbb{R}^m$ is the actuation constraint set, and $f\colon \mathcal{D} \times \mathcal{U} \to \mathbb{R}^n$ is a Lipschitz continuous function. This letter assumes that $\mathcal{U}$ is a convex polyhedron.

Given a locally-Lipschitz control policy $\pi(\cdot)$ and an initial condition $x_0 \in \mathbb{R}^n$, we denote the solution of the closed-loop system $\dot{x} = f(x, \pi(x)) = f_{cl}(x)$ at time $t \geq 0$ by $x(t, x_0; \pi)$. In this letter, we are interested in learning a nonlinear control policy $\pi_\gamma(\cdot)$, parameterized by a neural network with trainable parameters γ, with the following desiderata:

1. The control policy respects the actuation constraints, $\pi_\gamma(x) \in \mathcal{U} \ \forall \ x \in \mathcal{D}$;
2. The zero solution of the closed-loop system $\dot{x} = f(x, \pi_\gamma(x))$ is asymptotically stable; and
3. The DoA of the zero solution is maximized.

C. Neural Lyapunov Control

A typical learning-enabled approach to designing stabilizing controllers is to parameterize the Lyapunov function candidate V_θ and the controller π_γ with neural networks and aim to minimize the expected violation of Lyapunov conditions (2)

$$\min_{\theta,\gamma} E_{x\sim\rho}\big[V_\theta(0)^2 + (-V_\theta(x))_+ + \big(\nabla_x V_\theta(x)^\top f(x, \pi_\gamma(x))\big)_+\big], \tag{4}$$

where ρ is a predefined sampling distribution with support on $\mathcal{D}$ [13]. There are potentially infinitely many Lyapunov function candidates that minimize the expected loss, including the shortcut solution $V_\theta(x) = 0 \ \forall x \in \mathcal{D}$. To avoid this shortcut and promote large DoA, we must use regularization, e.g., $(\|x\|_2 - \alpha V_\theta(x))^2$ [13]. However, poorly chosen regularizers can lead to a mismatch between the shape of level sets of the

learned Lyapunov function and the true DoA. Motivated by this drawback, we will incorporate a physics-informed loss function to directly capture the true DoA.

III. Physics-Informed Actor-Critic Learning

A. Maximal Lyapunov Functions and Zubov's Method

To characterize the DoA exactly, we can use a variation of the Lyapunov function, the *maximal Lyapunov function* [26].

Theorem 2 (Maximal Lyapunov Function[26]): Suppose there exists a set $\mathcal{A} \subseteq \mathcal{D}$ containing the origin in its interior, a continuously differentiable function $V\colon \mathcal{A} \to \mathbb{R}$, and a positive definite function Φ satisfying the following conditions

$$V(0) = 0, V(x) > 0 \ \forall x \in \mathcal{A} \setminus \{0\} \tag{5a}$$
$$\nabla V(x)^\top f(x) = -\Phi(x) \ \forall x \in \mathcal{A} \tag{5b}$$
$$V(x) \to \infty \text{ as } x \to \partial\mathcal{A} \text{ or } \|x\| \to \infty \tag{5c}$$

Then $\mathcal{A}$ is the DoA for the system in (1).

One can verify that the function

$$V(x_0) = \begin{cases} \int_0^\infty \|x(t;x_0)\| dt & \text{if the integral converges} \\ \infty & \text{otherwise} \end{cases} \tag{6}$$

satisfies the conditions of Theorem 2 with $\Phi(x) = \|x\|$ [26].

Due to the presence of infinity in the integral, the above maximal Lyapunov function is hard to represent by function approximators such as neural networks. Another way to build a Lyapunov function and characterize the DoA is using Zubov's Theorem [25] [27].

Theorem 3 (Zubov's Theorem [27]): Consider the nonlinear dynamical system in (1). Let $\mathcal{A} \subseteq \mathcal{D}$ and assume there exists a continuously differentiable function $W\colon \mathcal{A} \to \mathbb{R}$ and a positive definite function $\Psi : \mathbb{R}^n \to \mathbb{R}$ satisfying the following conditions,

$$W(0) = 0 \tag{7a}$$
$$0 < W(x) < 1 \ \forall x \in \mathcal{A} \setminus \{0\} \tag{7b}$$
$$W(x) \to 1 \text{ as } x \to \partial\mathcal{A} \text{ or } \|x\| \to \infty \tag{7c}$$
$$\nabla W(x)^\top f(x) = -\Psi(x)(1 - W(x)). \tag{7d}$$

Then $x(t) = 0$ is asymptotically stable with DoA $\mathcal{A}$.

Intuitively, the Zubov PDE (7d) ensures that the Lie derivative of W on the boundary of $\mathcal{A}$ is zero, a property which precisely renders $\mathcal{A}$ the domain of attraction–see Figure 1 for an illustration.

The maximal Lyapunov function $V(\cdot)$ of Theorem 2 and the Zubov function $W(\cdot)$ of Theorem 3 can be related by [28]

$$W(x) = \tanh(\alpha V(x)) \tag{8a}$$
$$V(x) = \frac{1}{2\alpha} \log\left(\frac{1 + W(x)}{1 - W(x)}\right) \tag{8b}$$
$$\Psi(x) = \alpha(1 + W(x))\Phi(x) \tag{8c}$$

where $\alpha > 0$. In the rest of this letter, we choose $\Phi(x) = \|x\|_2$.

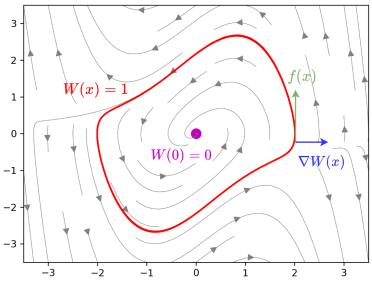


Fig. 1: The level set $\{x : W(x) = 1\}$ obtained by the Zubov PDE characterizes the boundary of the true DoA.

B. Learning the Zubov Function (Critic)

Suppose we have a *fixed* policy $\pi_\gamma(\cdot)$ and the corresponding closed-loop system $\dot{x} = f(x, \pi_\gamma(x))$. Let $V(\cdot)$ be chosen as in (6). The relationship (8a), combined with (7d), enables us to learn the Zubov function $W_\theta(x) \approx \tanh(\alpha V(x))$, which can be interpreted as an evaluation of the policy π_γ. Here $W_\theta(\cdot)$ is a neural network with sigmoid activation functions at the output layer [29] such that $W_\theta(x) \in (0, 1) \ \forall x \in \mathcal{D}$. To train $W_\theta(\cdot)$, we use the following loss function,

$$L(\theta) = L_z(\theta) + L_r(\theta) + L_p(\theta) \tag{9}$$

where

$$L_z(\theta) = W_\theta(0)^2 \tag{10a}$$
$$L_r(\theta) = \frac{1}{M} \sum_{i=1}^{M} \left(W_\theta(x_i) - \tanh(\alpha V(x_i))\right)^2 \tag{10b}$$
$$L_p(\theta) = \frac{1}{K} \sum_{i=1}^{K} \Big[\nabla_x W_\theta(x_i)^\top f(x_i, \pi_\gamma(x_i)) + \alpha(1 - W_\theta(x_i))(1 + W_\theta(x_i))\Phi(x_i)\Big]^2 \tag{10c}$$

The first and second terms, $L_z(\theta)$ and $L_r(\theta)$, penalize the violation of (7a) and (8a), respectively. We note that to evaluate $V(\cdot)$ that appears in this loss, we simulate the closed-loop system from M initial conditions $\{x_i\}_{i=1}^M$. Finally, the third term $L_p(\theta)$ is the physic-informed part to minimize the residual of the Zubov PDE in (7d) at K sampled states.

C. Policy Improvement (Actor)

Given a policy $\pi_\gamma(\cdot)$, the corresponding gound-truth Zubov function is $W(x) = \tanh(\alpha V^{\pi_\gamma}(x))$ where $V^{\pi_\gamma}(x)$ is defined as in (6) for the autonomous system $\dot{x} = f(x, \pi_\gamma(x))$. We can improve the policy to drive the state to $x = 0$ by minimizing the Lie derivative of W along f akin to Sontag's formula [17]. However, since we cannot differentiate $W(x)$ with respect to x, we use $W_\theta(\cdot)$ as a proxy of the true W, which results in the following loss

function

$$L_c(\gamma) = \frac{1}{K}\sum_{i=1}^{K}\left[\frac{\nabla_x W_\theta(x_i)^\top}{\|\nabla_x W_\theta(x_i)\|} f(x_i, \pi_\gamma(x_i))\right]$$

Here we normalize the gradient $\nabla_x W_\theta(x)$ so that the training samples are equally weighted in both the steep regions and flat regions of $W_\theta(\cdot)$.

D. Actor-Critic Learning

We now combine the critic of section III-B and the actor of section III-C to co-learn the controller and the Zubov function. We first define $R_1 \subseteq \mathcal{D}$ to be the region from which we sample $\{x_i\}$ and define the region $R_2 = \{ax : x \in R_1\}$ $(a > 1)$ as the region of interest. In this letter we choose $a = 2$. Since the controller is randomly initialized, the sampled trajectories might diverge. To prevent this, we add a loss function as follows,

$$L_b(\theta) = \frac{1}{K}\sum_{x_i' \in \partial R_2, i=1,\dots,K} \left|W_\theta(x_i') - 1\right|$$

Since $W(\cdot)$ is constrained to be in $[0, 1]$ and the actor is trained to minimize the Lie derivative of W, this loss prevents the states from going to ∂R_2, where $W(\cdot)$ takes the maximum value 1. This loss function stabilizes the training at the beginning.

Combining policy evaluation and policy improvement, we can co-learn a controller and a Zubov function in an actor-critic fashion by minimizing the following loss function

$$\begin{aligned} L(\theta,\gamma) = {} & \lambda_0 W_\theta(0)^2 \\ & + \frac{1}{M}\sum_{x_i \in R_1, i=1,..,M} \left(W_\theta(x_i) - \tanh(\alpha V(x_i)\right)^2 \\ & + \frac{1}{K}\sum_{x_i \in R_1, i=1,..,K} \Big[\nabla_x W_\theta(x_i)^\top f(x_i, \mathbf{sg}(\pi_\gamma(x_i)) \\ & \qquad + \alpha(1 - W_\theta(x_i))(1 + W_\theta(x_i))\Phi(x)\Big]^2 \\ & + \frac{\lambda_c}{K}\sum_{x_i \in R_1, i=1,..,K} \left[\mathbf{sg}\left(\frac{\nabla_x W_\theta(x_i)^\top}{\|\nabla_x W_\theta(x_i)\|}\right) f(x_i, \pi_\gamma(x_i))\right] \\ & + \frac{\lambda_b}{K}\sum_{x_j' \in \partial R_2, j=1,\dots,K} \left|W_\theta(x_j') - 1\right| \end{aligned} \tag{11}$$

Note that rather than alternating between updating $W_\theta(\cdot)$ and $\pi_\gamma(\cdot)$ with the other one fixed, we use the $\mathbf{sg}(\cdot)$ operator to enable a simultaneous update of $W_\theta(\cdot)$ and $\pi_\gamma(\cdot)$. We outline the overall method in Algorithm 1.

Algorithm 1 Physics-Informed Neural Lyapunov Control

Randomly initialize W_θ and π_γ
for $n = 1, ..., T$ **do**
  Randomly sample $x_1, ..., x_K$ from R_1
  Randomly sample $x_1', ...x_K'$ from ∂R_2
  Simulate trajectories $x(t, x_i; \pi_\gamma)$ (with RK-4 integrator)
  Estimate $V(x_i) \approx \int_t \|x(t, x_i; \pi)\| dt$ for $i = 1, .., M$
  Take a gradient step to minimize $L(\theta, \gamma)$ in (11)
end for

E. Actuation Constraints

To respect actuation constraints, one approach is to append a Euclidean projection layer to a generic neural network $u_\gamma(\cdot)$, resulting in the control policy

$$\pi_\gamma(x) = \mathrm{Proj}_{\mathcal{U}}[u_\gamma(x)] = \mathrm{argmin}_{u' \in \mathcal{U}} \|u' - u_\gamma(x)\|_2^2 \tag{12}$$

For convex $\mathcal{U}$, we can compute the derivative $D_\gamma u_\gamma(x)$, which is needed to learn γ, using differentiable convex optimization layers [30]. This approach is particularly efficient for box constraint sets, $\mathcal{U} = \{u : \underline{u} \le u \le \bar{u}\}$, for which the projection can be computed in closed form. However, for general convex polyhedrons, the projection has to be solved numerically. To avoid this, our innovative solution is to span the actuation constraint set $\mathcal{U}$ using a convex combination of its vertices. Formally, suppose $V = [v_1 \ \cdots v_M]$ is the matrix of vertices of $\mathcal{U}$. We then parameterize the control policy as

$$\pi_\gamma(x) = V \cdot \mathrm{softmax}(u_\gamma(x)) = \sum_{i=1}^{M} V_i \mathrm{softmax}(u_\gamma(x))_i \tag{13}$$

where the vector-valued softmax function simply generates the coefficients of the convex combination of columns of V. In contrast to (12), this parameterization eliminates the need to compute any numerical optimization sub-routine.

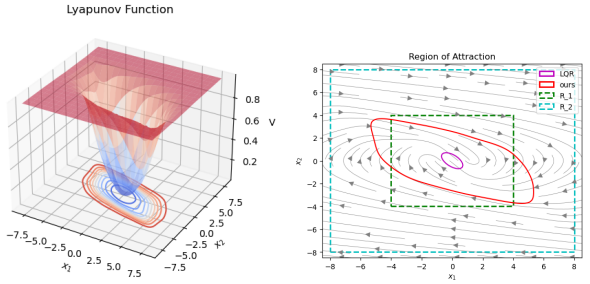
F. Verification

To formally verify the learned certificate, let $c \in (0, 1)$. Our goal is to verify the following three conditions

$$\begin{cases} W_\theta(x) > W_\theta(0) \ \forall x \in \{x \in R_2 \mid W_\theta(x) < c, \ \|x\| \ge \epsilon\} \\ \dot{W}_\theta(x) < 0 \ \forall x \in \{x \in R_2 \mid W_\theta(x) < c, \ \|x\| \ge \epsilon\} \\ W_\theta(x) > c \ \forall x \in \partial R_2 \end{cases} \tag{14}$$

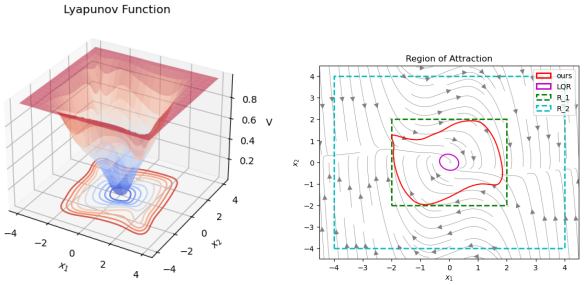
Since W_θ can have disjoint sub-level sets, we only consider sub-level sets contained in R_2.

The first and second conditions verify the Lyapunov conditions on the set $\mathcal{D} = R_2 \cap \{x : W_\theta(x) < c\}$. We ignore a smaller neighbor around the origin to avoid numerical errors. However, the first two conditions do not imply that $\mathcal{D}$ is the domain of attraction. We also need to verify the third condition, which implies that the set $\mathcal{D}$ is strictly contained in R_2 so that a trajectory does not leave R_2 with negative $\dot{W}_\theta$ along the trajectory. If these conditions are satisfied, then the function $W_\theta(x) - W_\theta(0)$ is a Lyapunov function in the region $R_2 \cap \{x : W_\theta(x) < c\}$.



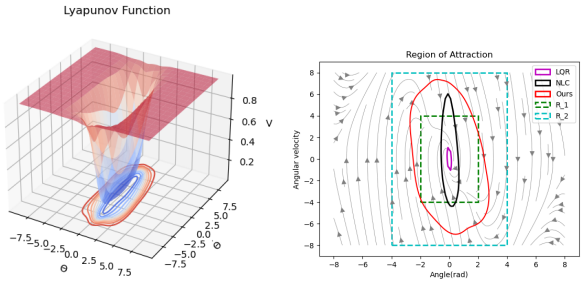
(a) $W_\theta(x)$ learned for double integrator system (b) Verified DoA and vector field $\dot{x} = f(x, \pi_\gamma(x))$

Fig. 2: Double integrator system



(a) $W_\theta(x)$ learned for Van der Pol system (b) Verified DoA and vector field $\dot{x} = f(x, \pi_\gamma(x))$

Fig. 3: Van der Pol system



(a) $W_\theta(x)$ learned for inverted pendulum system (b) Verified DoA and vector field $\dot{x} = f(x, \pi_\gamma(x))$

Fig. 4: Inverted Pendulum system

IV. Numerical Experiments

We demonstrate the effectiveness of our method on various nonlinear control problems. We first run our method on Double Integrator and Van der Pol and compare the verified DoA obtained by our method with that of the LQR controller. We then test our method on Inverted Pendulum and Bicycle Tracking and compare it to both LQR and Neural Lyapunov Control [13][1]. Across all examples, we use the same hyperparameter $M = 8, K = 64, \lambda_0 = 5, \lambda_c = 0.5$, and $\lambda_b = 5$ for the loss function. Other hyperparameters and verified sublevel set can be found in Table I. The effect of different choices of hyperparameters can be found in the preprint version [31]. In all experiments, we assumed that the actuation constraint is $u \in \mathcal{U} = \{u | \ \|u\|_\infty \leq 1\}$ and the learned controller satisfies this constraint by using $\tanh$ as the activation function in the output layer.

To find the DoA of LQR under actuation constraints, we first solve the Riccati equation of the linearized system with $Q = I, R = I$ to obtain the solution P and control matrix K, and then we verify the following conditions,

$$\begin{cases} x^\top P \dot{x} < 0 \ \forall x \text{ s.t } x^\top P x < c_{lqr} \text{ and } \|x\| \geq \epsilon \\ \|Kx\|_\infty \leq 1 \ \forall x \text{ s.t } x^\top P x < c_{lqr} \text{ and } \|x\| \geq \epsilon, \end{cases} \tag{15}$$

where $\dot{x}$ is given by the nonlinear closed-loop dynamics.

We used the SMT solver dReal [32] to verify the conditions (14) for the neural network controller and (15) for the LQR controller. To find the largest DoA for both LQR and our method, we perform bisection to find the largest c_{lqr} (c) such that conditions (14), (15) are satisfied.

A. Double Integrator

We first consider the double integrator dynamics $\dot{x}_1 = x_2$, $\dot{x}_2 = u$. The learned Lyapunov function is shown in Figure 2a. The verified DoA and the vector field induced by the learned controller are shown in Figure 2b.

B. Van der Pol

Consider the Van der Pol system $\dot{x}_1 = x_2$ and $\dot{x}_2 = x_1 - \mu(1 - x_1^2)x_2 + u$. The learned Lyapunov function is shown in Figure 3a. The verified DoA and the vector field induced by the learned controller are shown in Figure 3b.

[1]The code repository of [13] does not support the first two examples.

C. Inverted Pendulum

The inverted pendulum system has two states, the angular position θ, angular velocity ω, and control input u. The dynamical system of inverted pendulum is $\dot{\theta} = \omega$ and $\dot{\omega} = \frac{g}{l}\sin(\theta) - \frac{b\omega}{ml^2} + \frac{u}{ml^2}$. The learned Lyapunov function is shown in Figure 4a. The verified DoA and the vector field induced by the learned controller are shown in Figure 4b.

D. Bicycle Tracking

The bicycle tracking system has two states, the distance d_e, angle error θ_e, and control input u. The dynamical system for the tracking problem is $\dot{d}_e = v\sin(\theta_e)$ and $\dot{\theta}_e = v\frac{\tan(u)}{l} - \frac{\cos(\theta_e)}{1-d_e}$. The learned Lyapunov function is shown in Figure 5a. The verified DoA and the vector field induced by the learned controller are shown in Figure 5b.

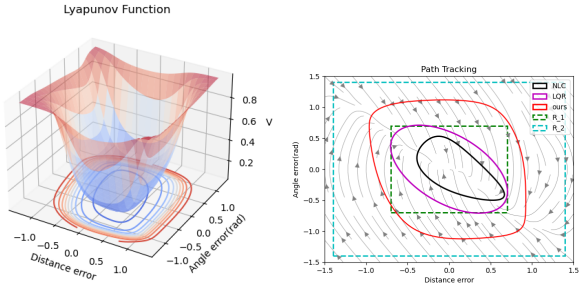
In summary, all the experiments show that our verified DoA is 2$\sim$4 times larger than other methods.

V. Conclusions

In this letter, we developed an actor-critic framework to jointly train a neural network control policy and the

Dynamical System	W_θ dimension	π_γ dimension	α	c
Double Integrator	[2,20,20,1]	[2, 10, 10, 1]	0.05	0.7
Van der Pol	[2,30,30,1]	[2, 30, 30, 1]	0.1	0.5
Inverted Pendulum	[2, 20, 20, 1]	[2, 5, 5, 1]	0.2	0.7
Bicycle Tracking	[2, 20, 20, 1]	[2, 10, 10, 1]	1.5	0.4

TABLE I: Experiment Details



(a) $W_\theta(x)$ learned for bicycle tracking system (b) Verified DoA and vector field $\dot{x} = f(x, \pi_\gamma(x))$

Fig. 5: Bicycle tracking system

corresponding stability certificate, with the explicit goal of maximizing the induced region of attraction by leveraging Zubov's partial differential equation to inform the loss function of the shape of the boundary of the true domain of attraction. Our numerical experiments on several nonlinear benchmark examples corroborate the superiority of the proposed method over competitive approaches in enlarging the domain of attraction. For future work, we will incorporate robustness to model uncertainty in our loss function. We will also investigate the extension of the proposed method to discrete-time systems [20].

REFERENCES

[1] S. Li, E. Öztürk, C. De Wagter, G. C. De Croon, and D. Izzo, "Aggressive online control of a quadrotor via deep network representations of optimality principles," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6282–6287, IEEE, 2020.

[2] X. Yang, D. Liu, and Y. Huang, "Neural-network-based online optimal control for uncertain non-linear continuous-time systems with control constraints," *IET Control Theory & Applications*, vol. 7, no. 17, pp. 2037–2047, 2013.

[3] J. Zhang, Q. Zhu, and W. Lin, "Neural stochastic control," *Advances in Neural Information Processing Systems*, vol. 35, pp. 9098–9110, 2022.

[4] E. Kaufmann, L. Bauersfeld, A. Loquercio, M. Müller, V. Koltun, and D. Scaramuzza, "Champion-level drone racing using deep reinforcement learning," *Nature*, vol. 620, no. 7976, pp. 982–987, 2023.

[5] H. Kwakernaak and R. Sivan, *Linear optimal control systems*, vol. 1. Wiley-interscience New York, 1972.

[6] A. A. Ahmadi and A. Majumdar, "Some applications of polynomial optimization in operations research and real-time decision making," *Optimization Letters*, vol. 10, pp. 709–729, 2016.

[7] C. Verhoek, P. J. Koelewijn, S. Haesaert, and R. Tóth, "Convex incremental dissipativity analysis of nonlinear systems," *Automatica*, vol. 150, p. 110859, Apr. 2023.

[8] A. Abate, D. Ahmed, M. Giacobbe, and A. Peruffo, "Formal synthesis of lyapunov neural networks," *IEEE Control Systems Letters*, vol. 5, no. 3, pp. 773–778, 2020.

[9] D. Ahmed, A. Peruffo, and A. Abate, "Automated and sound synthesis of lyapunov functions with smt solvers," in *Tools and Algorithms for the Construction and Analysis of Systems: 26th International Conference, TACAS 2020, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2020, Dublin, Ireland, April 25–30, 2020, Proceedings, Part I 26*, pp. 97–114, Springer, 2020.

[10] C. Dawson, S. Gao, and C. Fan, "Safe control with learned certificates: A survey of neural lyapunov, barrier, and contraction methods for robotics and control," *IEEE Transactions on Robotics*, 2023.

[11] N. Gaby, F. Zhang, and X. Ye, "Lyapunov-net: A deep neural network architecture for lyapunov function approximation," in *2022 IEEE 61st Conference on Decision and Control (CDC)*, pp. 2091–2096, IEEE, 2022.

[12] L. Grüne, "Computing lyapunov functions using deep neural networks," *arXiv preprint arXiv:2005.08965*, 2020.

[13] Y.-C. Chang, N. Roohi, and S. Gao, "Neural Lyapunov control," *Advances in neural information processing systems*, vol. 32, 2019.

[14] W. Jin, Z. Wang, Z. Yang, and S. Mou, "Neural certificates for safe control policies," *arXiv preprint arXiv:2006.08465*, 2020.

[15] R. Zhou, T. Quartz, H. D. Sterck, and J. Liu, "Neural Lyapunov control of unknown nonlinear systems with stability guarantees," 2022.

[16] C. Barrett and C. Tinelli, "Satisfiability modulo theories," *Handbook of model checking*, pp. 305–343, 2018.

[17] E. D. Sontag, *Mathematical control theory: deterministic finite dimensional systems*, vol. 6. Springer Science & Business Media, 2013.

[18] N. Boffi, S. Tu, N. Matni, J.-J. Slotine, and V. Sindhwani, "Learning stability certificates from data," in *Conference on Robot Learning*, pp. 1341–1350, PMLR, 2021.

[19] M. Raissi, P. Perdikaris, and G. E. Karniadakis, "Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations," *Journal of Computational physics*, vol. 378, pp. 686–707, 2019.

[20] J. Wu, A. Clark, Y. Kantaros, and Y. Vorobeychik, "Neural Lyapunov control for discrete-time systems," *Advances in Neural Information Processing Systems*, vol. 36, 2024.

[21] V. Konda and J. Tsitsiklis, "Actor-critic algorithms," *Advances in neural information processing systems*, vol. 12, 1999.

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

[23] Y. Meng, R. Zhou, A. Mukherjee, M. Fitzsimmons, C. Song, and J. Liu, "Physics-informed neural network policy iteration: Algorithms, convergence, and verification," *arXiv preprint arXiv:2402.10119*, 2024.

[24] R. Leake and R.-W. Liu, "Construction of suboptimal control sequences," *SIAM Journal on Control*, vol. 5, no. 1, pp. 54–63, 1967.

[25] W. M. Haddad and V. Chellaboina, *Nonlinear Dynamical Systems and control: A Lyapunov-based approach*. Princeton University, 2008.

[26] A. Vannelli and M. Vidyasagar, "Maximal Lyapunov functions and domains of attraction for autonomous nonlinear systems," *Automatica*, vol. 21, no. 1, pp. 69–80, 1985.

[27] V. I. Zubov, *Methods of AM Lyapunov and their application*, vol. 4439. US Atomic Energy Commission, 1961.

[28] W. Kang, K. Sun, and L. Xu, "Data-driven computational methods for the domain of attraction and Zubov's equation," *IEEE Transactions on Automatic Control*, 2023.

[29] J. Liu, Y. Meng, M. Fitzsimmons, and R. Zhou, "Physics-informed neural network Lyapunov functions: Pde characterization, learning, and verification," *arXiv preprint arXiv:2312.09131*, 2023.

[30] A. Agrawal, B. Amos, S. Barratt, S. Boyd, S. Diamond, and J. Z. Kolter, "Differentiable convex optimization layers," *Advances in neural information processing systems*, vol. 32, 2019.

[31] J. Wang and M. Fazlyab, "Actor-critic physics-informed neural lyapunov control," 2024.

[32] S. Gao, S. Kong, and E. M. Clarke, "dreal: An smt solver for nonlinear theories over the reals," in *Automated Deduction–CADE-24: 24th International Conference on Automated Deduction, Lake Placid, NY, USA, June 9-14, 2013. Proceedings 24*, pp. 208–214, Springer, 2013.