Proceedings of Machine Learning Research vol vvv:1–25, 2026

Scalable Verification of Neural Control Barrier Functions Using Linear Bound Propagation

Nikolaus Vertovec[1] NIKOLAUS.VERTOVEC@ST-HUGHS.OX.AC.UK
Frederik Baymler Mathiesen[2] F.B.MATHIESEN@TUDELFT.NL
Thom Badings[1] THOM.BADINGS@CS.OX.AC.UK
Luca Laurenti[2] L.LAURENTI@TUDELFT.NL
Alessandro Abate[1] ALESSANDRO.ABATE@CS.OX.AC.UK
[1]*Department of Computer Science, University of Oxford, Oxford, United Kingdom*
[2]*Delft Center for Systems and Control, TU Delft, Delft, The Netherlands*

Abstract

Control barrier functions (CBFs) are a popular tool for safety certification of nonlinear dynamical control systems. Recently, CBFs represented as neural networks have shown great promise due to their expressiveness and applicability to a broad class of dynamics and safety constraints. However, verifying that a trained neural network is indeed a valid CBF is a computational bottleneck that limits the size of the networks that can be used. To overcome this limitation, we present a novel framework for verifying neural CBFs based on piecewise linear upper and lower bounds on the conditions required for a neural network to be a CBF. Our approach is rooted in linear bound propagation (LBP) for neural networks, which we extend to compute bounds on the gradients of the network. Combined with McCormick relaxation, we derive linear upper and lower bounds on the CBF conditions, thereby eliminating the need for computationally expensive verification procedures. Our approach applies to arbitrary control-affine systems and a broad range of nonlinear activation functions. To reduce conservatism, we develop a parallelizable refinement strategy that adaptively refines the regions over which these bounds are computed. Our approach scales to larger neural networks than state-of-the-art verification procedures for CBFs, as demonstrated by our numerical experiments.

Keywords: safety verification, control barrier functions, neural networks, linear bound propagation

1. Introduction

Safety verification of autonomous control systems—such as unmanned aerial and ground vehicles or robotic manipulators—is critical for their deployment in real-world environments (Hsu et al., 2024). As a result, certifying the safety of these systems, commonly modelled as nonlinear dynamical control systems, has become increasingly important. To certify safety, *control barrier functions* (CBFs) have emerged as an effective tool for enforcing *forward control invariance*. CBFs have been successfully applied to robotic manipulators (Shaw-Cortez et al., 2021), vehicle cruise control (Ames et al., 2014), bipedal robots (Agrawal and Sreenath, 2017), and satellite navigation (Breeden and Panagou, 2023).

Finding a CBF—particularly one that maximizes the *control invariant set*—remains an ongoing challenge in the control literature (Ames et al., 2017; Xiao and Belta, 2022; Clark, 2021). A common approach relies on *sum-of-squares* (SOS) programming (Wang et al., 2023), which formulates safety and invariance conditions as semidefinite programs. However, SOS methods generally only apply when the dynamics, control inputs, and candidate barrier functions are polynomial. In addition,

finding a valid CBF typically requires careful selection of an appropriate monomial parameterization basis for the barrier function.

To overcome these challenges, CBFs represented as neural networks—*neural CBFs*—have become a popular alternative, enabling the *inductive synthesis* of barrier functions for a broad class of dynamics and safety constraints (Zhao et al., 2020; Zhang et al., 2023; Hu et al., 2024; Abate et al., 2021b). Neural CBFs allow for arbitrary nonlinear dynamics, and the expressiveness of the barrier function can be increased with the neural network's size, thereby expanding the obtainable control invariant set. However, the main drawback of learning a *candidate barrier function* arguably lies in the high cost of post-hoc verification required to prove that the candidate is a valid CBF. This verification stage constitutes the primary computational bottleneck, constrained by two key factors: (1) the size of the neural network and (2) the dimensionality of the dynamical system. Existing verification approaches predominantly rely on satisfiability modulo theories (SMT) solvers (Zhao et al., 2020; Abate et al., 2021a; Sha et al., 2021; Edwards et al., 2024) and mixed-integer programming (MIP) (Zhao et al., 2022), both of which exhibit poor scalability with respect to network size. Architecture-specific approaches (Zhang et al., 2023; Hu et al., 2024), meanwhile, lack generalization to networks with arbitrary nonlinear activation functions needed for sufficient expressivity.

Our approach In this paper, we propose a scalable verification technique for neural CBFs that overcomes the limitations of SMT-based verifiers while still allowing for arbitrary activation functions. Instead of using SMT solvers to reason over the exact nonlinear conditions required for a function to be a CBF, we derive sufficient linear conditions for more efficient verification of a neural candidate barrier function. To compute these sufficient linear conditions, we extend *linear bound propagation* (LBP) (Zhang et al., 2018) to derive linear upper and lower bounds on the gradients of the network. Rooted in recent advances in neural network verification (Zhang et al., 2018; Shi et al., 2025), LBP has been applied to the verification of *discrete-time* stochastic barrier functions (Mathiesen et al., 2023). Our setting with *continuous-time* dynamics requires a derivative condition to establish set invariance, necessitating linear bounds to the network gradients and linear relaxations of Lie derivatives.

We combine our techniques for computing bounds on the network gradients with McCormick relaxation (McCormick, 1976) and certified bounds on the dynamics to derive linear upper and lower bounds on the CBF conditions. Importantly, our approach supports the inclusion of control variables in the invariance condition, making our approach attractive for use with downstream applications such as safety filters (Wabersich et al., 2023). Finally, our verification approach is agnostic to the neural network training and can (while beyond our scope) be integrated into common learner-verifier frameworks for synthesizing neural CBFs (Peruffo et al., 2021; Dawson et al., 2023).

Overall, our key contributions are summarized as follows:

1. We introduce a novel method to verify candidate neural barrier functions via LBP and McCormick relaxation, thereby eliminating the need for computationally expensive verification procedures and enabling the verification of larger networks.
2. We develop a tailored and parallelizable refinement strategy that adaptively refines a simplicial mesh over the state space, thus reducing conservatism in the lower and upper bounds.
3. We demonstrate that our approach enables the verification of neural CBFs with larger networks than state-of-the-art SMT-based verification techniques.

We begin by providing background on neural CBFs for safety certification in Sect. 2. We introduce our contributions on extending LBP in Sect. 3 and the verification procedure for the CBF in Sect. 4. Finally, we experimentally demonstrate the scalability of our approach to larger networks in Sect. 5.

2. Problem Formulation

We consider nonlinear control-affine dynamical systems with state space $\mathcal{X} \subset \mathbb{R}^n$, where the state $x \in \mathcal{X}$ evolves according to

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

with continuous functions $f\colon \mathcal{X} \to \mathbb{R}^n$ and $g\colon \mathcal{X} \to \mathbb{R}^{n\times m}$, and a bounded control input $u(t) \in \mathcal{U} = [\underline{u}, \overline{u}] \subset \mathbb{R}^m$. The control objective is to ensure that the state $x(t)$ remains within a desired measurable set $\mathcal{S} \subset \mathcal{X}$ for all $t \geq 0$, which we refer to as the *safe set*. To guarantee the system remains safe (i.e., within the safe set), we relate safety to the notion of control invariance.

Definition 1 *A set $\mathcal{C} \subset \mathbb{R}^n$ is said to be* control invariant *with respect to Eq. (1) if, for any $x(0) \in \mathcal{C}$, there exists a measurable control signal $u\colon [0,\infty) \to [\underline{u}, \overline{u}]$ such that $x(t) \in \mathcal{C}$ for all $t \geq 0$.*

To find a control invariant subset of the safe set, we synthesize a *control barrier function* (CBF) (Ames et al., 2019). A continuously differentiable function $\mathcal{B}\colon \mathcal{X} \to \mathbb{R}$ is a CBF for Eq. (1) and the safe set $\mathcal{S}$ if the superlevel set $\mathcal{C} = \{x \in \mathcal{X} : \mathcal{B}(x) \geq 0\}$ satisfies $\mathcal{C} \subseteq \mathcal{S}$, and there exists an extended class-$\mathcal{K}$ function $\alpha\colon \mathbb{R} \to \mathbb{R}$ (i.e., α is strictly increasing and $\alpha(0) = 0$) such that, for all $x \in \mathcal{C}$,

$$\mathcal{L}_f\mathcal{B}(x) + \sup_{u\in\mathcal{U}} [\mathcal{L}_g\mathcal{B}(x)u] + \alpha(\mathcal{B}(x)) \geq 0, \tag{2}$$

where $\mathcal{L}_f\mathcal{B}(x) := \nabla_x\mathcal{B}(x)f(x)$ and $\mathcal{L}_g\mathcal{B}(x) := \nabla_x\mathcal{B}(x)g(x)$ denote the Lie derivatives of $\mathcal{B}$ along f and g, respectively.

Theorem 2 *[Ames et al. (2017)] If $\mathcal{B}$ is a CBF for the dynamical system in Eq. (1) and the safe set $\mathcal{S}$, then the superlevel set $\mathcal{C}$ is control invariant with respect to Eq. (1).*

Theorem 2 enables different formulations of the safety verification problem. For instance, given a set of initial states $\mathcal{X}_0 \subset \mathcal{X}$, one may seek to prove that the system in Eq. (1) is safe with respect to $\mathcal{S}$ and all $x(0) \in \mathcal{X}_0$ by finding a CBF $\mathcal{B}$ such that $\mathcal{X}_0 \subseteq \mathcal{C}$. Alternatively, one might aim to find a CBF whose superlevel set $\mathcal{C}$ maximizes the volume contained in $\mathcal{S}$.

2.1. Neural Control Barrier Functions

Finding a CBF can be challenging, especially for nonlinear dynamics and nonconvex safe sets. As such, it has become common practice to *learn* a CBF, often by training a feedforward neural network $\mathcal{B}_\theta\colon \mathbb{R}^n \to \mathbb{R}$ with parameters θ (Dawson et al., 2023).

Definition 3 *An $(L+1)$-layer neural network $\mathcal{B}_\theta$ with dimensions $n_0, \ldots, n_L$ is a function $\mathcal{B}_\theta\colon \mathbb{R}^{n_0} \to \mathbb{R}^{n_L}$ defined as $\mathcal{B}_\theta(x) = (h^{(L)} \circ h^{(L-1)} \circ \cdots \circ h^{(2)} \circ h^{(1)})(x)$, where $h^{(i)}(z_{i-1}) = \sigma^{(i)}(W^{(i)}z_{i-1} + b^{(i)}) = z_i$ is the i^{th} layer with parameters $W^{(i)} \in \mathbb{R}^{n_i\times n_{i-1}}$ and $b^{(i)} \in \mathbb{R}^{n_i}$ for $z_{i-1} \in \mathbb{R}^{n_{i-1}}$ and $z_i \in \mathbb{R}^{n_i}$, and nonlinear activation function $\sigma^{(i)}(\cdot) : \mathbb{R} \to \mathbb{R}$.*

Applying $\sigma^{(i)}(\cdot)$ to a vector $z_i \in \mathbb{R}^{n_i}$ is understood as an elementwise application of the scalar activation function $\sigma^{(i)}$. For each layer i, define the *pre-activation output* as $y_i = W^{(i)}z_{i-1} + b^{(i)}$ and the *post-activation output* as $z_i = \sigma^{(i)}(y_i)$. We assume the final layer does not have an activation function, i.e., $\sigma^{(L)}(y_L) = y_L$. Thus, the final output is $\mathcal{B}_\theta(x) = z_L = y_L$. To allow deriving linear bounds on the activation function and its derivative in Sect. 3, we make the following assumption.

Assumption 4 *We assume that the activation function is semidifferentiable.*

The candidate barrier function can be trained on by minimizing a loss function that resembles a differentiable version of the CBF conditions. Various approaches have been developed to guide the training, many of which are rooted in *counterexample-guided inductive synthesis* (CEGIS) techniques (Abate et al., 2018; Dawson et al., 2023).

2.2. Problem Statement: Certification of a Valid Control Barrier Function

When working with neural CBFs, a key challenge is to verify that a trained neural network is indeed a valid CBF. In this paper, we focus on this verification problem, which is formalized as follows.

Problem 5 *Given the dynamics in Eq. (1), a safe set $\mathcal{S}$, and a candidate neural CBF $\mathcal{B}_\theta$, verify that $\mathcal{B}_\theta$ is a valid CBF, i.e., the superlevel set $\mathcal{C}$ of $\mathcal{B}_\theta$ is a control invariant subset of the safe set $\mathcal{S}$.*

To solve Problem 5, we need to establish that $\mathcal{B}_\theta$ satisfies $\mathcal{C} = \{x \in \mathcal{X} : \mathcal{B}_\theta(x) \geq 0\} \subseteq \mathcal{S}$ and the condition in Eq. (2) is satisfied, which can be encoded by the following formula:

$$\begin{aligned} \phi \coloneqq \forall x \in \mathcal{S} : \left[\mathcal{B}_\theta(x) < 0 \vee \frac{\partial \mathcal{B}_\theta(x)}{\partial x} f(x) + \sup_{u \in \mathcal{U}} \frac{\partial \mathcal{B}_\theta(x)}{\partial x} g(x) u + \alpha \mathcal{B}_\theta(x) \geq 0 \right] \\ \wedge \ \forall x \in \mathcal{S}^C : \left[\mathcal{B}_\theta(x) < 0 \right], \end{aligned} \tag{3}$$

where $\mathcal{S}^C = \mathcal{X} \setminus \mathcal{S}$ is the complement of $\mathcal{S}$. The satisfiability of the formula ϕ implies that the candidate barrier function $\mathcal{B}_\theta$ is a valid CBF. Previous works (Peruffo et al., 2021; Abate et al., 2021a; Sha et al., 2021; Zhao et al., 2020; Abate et al., 2021b) have used SMT solvers for quantifier-free nonlinear real arithmetic to search for satisfying assignments of the negation of ϕ. However, using SMT solvers introduces a significant computational bottleneck, thus limiting both the size of the neural network and its input dimension (corresponding to the state space dimension).

Overview of our approach To enable the use of larger neural networks, we circumvent reasoning over quantifier-free nonlinear real arithmetic and instead consider linear over- and under-approximations of the nonlinearities in the formula ϕ. To obtain these over- and under-approximations over a domain of interest $\Delta \subset \mathcal{X}$, we will bound the value $\mathcal{B}_\theta(x)$ and gradients $\frac{\partial \mathcal{B}_\theta(x)}{\partial x}$ of the network (discussed in Sect. 3), and the dynamics f and g (discussed in Sect. 4.1). In Sect. 4.2, we combine the resulting linear bounds into a (conservative) linear surrogate ϕ_{linear} for ϕ. In particular, satisfiability of the surrogate ϕ_{linear} implies satisfiability of ϕ and thus proves that $\mathcal{B}_\theta$ is a valid CBF. Finally, in Sect. 4.3 we present our refinement strategy crucial to reducing conservatism of ϕ_{linear}.

3. Linear Bound Propagation for Neural CBF Verification

In this section, we describe how to compute linear upper and lower bounds on $\mathcal{B}_\theta(x)$ and its gradients $\frac{\partial \mathcal{B}_\theta(x)}{\partial x}$ over a fixed domain of interest, denoted as $\Delta \subset \mathcal{X}$. We compute the bounds on $\mathcal{B}_\theta(x)$ using existing linear bound propagation (LBP) techniques from Zhang et al. (2018). Construction of the bounds on $\mathcal{B}_\theta(x)$ is discussed in Appendix B and yields

$$\underline{\mathcal{B}_\theta}(x) \coloneqq \underline{A}_{\mathcal{B}_\theta} x + \underline{a}_{\mathcal{B}_\theta} \leq \mathcal{B}_\theta(x) \leq \overline{A}_{\mathcal{B}_\theta} x + \overline{a}_{\mathcal{B}_\theta} =: \overline{\mathcal{B}_\theta}(x), \quad \forall x \in \Delta. \tag{4}$$

By contrast, the bounds on the gradient $\frac{\partial \mathcal{B}_\theta(x)}{\partial x}$ require further relaxation of bilinear terms and the derivatives of the activation function, which cannot be computed using standard LBP. In this section,

we present our first key contribution, which is an extension of LBP that incorporates these further relaxations to bound the gradient. We draw inspiration from Wicker et al. (2023), which obtains gradient bounds via a backward pass with interval bounds. In contrast, we use LBP to obtain tighter bounds and, as in Eiras et al. (2024), apply McCormick relaxations to the bilinear terms. As a novel feature, we formulate bounds with respect to pre-activation outputs and relax the bilinearity induced by the composition of successive layers. Notably, this approach eliminates the need for an additional forward pass of the neural network, as required in, e.g., Eiras et al. (2024). Our approach yields the following bounds on the gradient, with their construction being given in the proof of Theorem 6.

Theorem 6 (Linear bounds on Jacobian) *Given an $(L+1)$-layer neural network $\mathcal{B}_\theta(x)$ as in Def. 3 and a domain $\Delta \subset \mathcal{X}$, we can construct linear bounds on $\frac{\partial \mathcal{B}_\theta(x)}{\partial x}$ for all $x \in \Delta$ of the form*

$$\underline{\partial \mathcal{B}_\theta}(x) \coloneqq \underline{\Pi} x + \underline{\pi} \leq \frac{\partial \mathcal{B}_\theta(x)}{\partial x} \leq \overline{\Pi} x + \overline{\pi} =: \overline{\partial \mathcal{B}_\theta}(x), \tag{5}$$

where the inequalities are understood elementwise and the coefficients $(\underline{\Pi}, \underline{\pi}, \overline{\Pi}, \overline{\pi})$ are computed recursively through affine relaxations of the layer Jacobians.

Proof We can write the gradient via the chain rule as the product of the Jacobian of each layer:

$$\frac{\partial \mathcal{B}_\theta(x)}{\partial x} = \nabla_{z_{L-1}} h^{(L)}(z_{L-1}) \, \nabla_{z_{L-2}} h^{(L-1)}(z_{L-2}) \cdots \nabla_{z_1} h^{(2)}(z_1) \, \nabla_{z_0} h^{(1)}(z_0). \tag{6}$$

The Jacobian $\mathcal{J}^{(i)} \in \mathbb{R}^{n_i \times n_{i-1}}$ of layer i is defined by the derivative of the activation function $\sigma^{(i)\prime}$ and the weight matrix $W^{(i)}$:

$$\mathcal{J}^{(i)} \coloneqq \nabla_{z_{i-1}} h^{(i)}(z_{i-1}) = \frac{\partial z_i}{\partial y_i}(y_i) \frac{\partial y_i}{\partial z_{i-1}} = \operatorname{diag}\left(\sigma^{(i)\prime}(y_i)\right) W^{(i)},$$

where $\operatorname{diag}(x)$ is the diagonal matrix for the vector x. We recursively bound the product of the Jacobians in Eq. (6). First, we construct linear bounds on the Jacobian of the final layer. Then, the Jacobian of each previous layer $i-1$ is combined with the current bounds on the product of Jacobians from layers i to L. Thus, we seek to derive the following sequence of bounds:

$$\begin{aligned} \underline{\Pi}^{(L)} y_L + \underline{\pi}^{(L)} &\leq \mathcal{J}^{(L)}(y_L) \leq \overline{\Pi}^{(L)} y_L + \overline{\pi}^{(L)}, \\ \underline{\Pi}^{(i)} y_i + \underline{\pi}^{(i)} &\leq \mathcal{J}^{(i+1)}(y_{i+1}) \mathcal{J}^{(i)}(y_i) \leq \overline{\Pi}^{(i)} y_i + \overline{\pi}^{(i)}, \quad i = L-1, \ldots, 1. \end{aligned}$$

We derive the linear bounds on $\mathcal{J}^{(i)}(y_i)$ for $i = 1, \ldots, L$ in Appendix B and consider bounding the product of the Jacobians for two layers i and $i+1$ in the remainder of this proof. This requires writing $\mathcal{J}^{(i+1)}$ in terms of y_i. To do so, we substitute y_{i+1} with $W^{(i+1)}\sigma^{(i)}(y_i) + b^{(i+1)}$ and use the linear bounds on $\sigma^{(i)}(y_i)$ computed during the forward pass of standard LBP. The resulting linear lower and upper bounds of $\mathcal{J}^{(i+1)}(y_i)$ are given in Eq. (27) of Appendix B. Using Einstein notation, an element (j,k) of the resulting product is a sum over the index p: $(\mathcal{J}^{(i+1)}\mathcal{J}^{(i)})_{jk} = (\mathcal{J}^{(i+1)})_{jp}(\mathcal{J}^{(i)})_{pk}$. We use McCormick relaxations (McCormick, 1976) for bilinear terms to obtain

$$\begin{aligned} (\mathcal{J}^{(i+1)})_{jp}(\mathcal{J}^{(i)})_{pk} &\geq (\underline{\mathcal{J}}^{(i+1)})_{jp}(\mathcal{J}^{(i)})_{pk} + (\mathcal{J}^{(i+1)})_{jp}(\underline{\mathcal{J}}^{(i)})_{pk} - (\underline{\mathcal{J}}^{(i+1)})_{jp}(\underline{\mathcal{J}}^{(i)})_{pk}, \\ (\mathcal{J}^{(i+1)})_{jp}(\mathcal{J}^{(i)})_{pk} &\geq (\overline{\mathcal{J}}^{(i+1)})_{jp}(\mathcal{J}^{(i)})_{pk} + (\mathcal{J}^{(i+1)})_{jp}(\overline{\mathcal{J}}^{(i)})_{pk} - (\overline{\mathcal{J}}^{(i+1)})_{jp}(\overline{\mathcal{J}}^{(i)})_{pk}, \end{aligned}$$

where $(\underline{\mathcal{J}}^{(i)})_{pk}$ and $(\overline{\mathcal{J}}^{(i)})_{pk}$ denote interval bounds for $(\mathcal{J}^{(i)})_{pk}$ over Δ. Since either of the two lower bounds is valid individually, we consider the convex combination with parameter $\eta_{jpk} \in [0,1]$:

$$(\mathcal{J}^{(i+1)})_{jp}(\mathcal{J}^{(i)})_{pk} \geq \eta_{jpk}\Big((\underline{\mathcal{J}}^{(i+1)})_{jp}(\mathcal{J}^{(i)})_{pk} + (\mathcal{J}^{(i+1)})_{jp}(\underline{\mathcal{J}}^{(i)})_{pk} - (\underline{\mathcal{J}}^{(i+1)})_{jp}(\underline{\mathcal{J}}^{(i)})_{pk}\Big) + (1-\eta_{jpk})\Big((\overline{\mathcal{J}}^{(i+1)})_{jp}(\mathcal{J}^{(i)})_{pk} + (\mathcal{J}^{(i+1)})_{jp}(\overline{\mathcal{J}}^{(i)})_{pk} - (\overline{\mathcal{J}}^{(i+1)})_{jp}(\overline{\mathcal{J}}^{(i)})_{pk}\Big).$$

Let $G^+ = \max(G,0)$ and $G^- = \min(G,0)$ be the positive and negative parts of a matrix G. Substituting the affine bounds for $(\mathcal{J}^{(i)})_{pk}$ and $(\mathcal{J}^{(i+1)})_{jp}$ yields

$$\begin{aligned}(\mathcal{J}^{(i+1)})_{jp}(\mathcal{J}^{(i)})_{pk} \geq\ & \Big(\eta_{jpk}(\underline{\mathcal{J}}^{(i+1)})_{jp} + (1-\eta_{jpk})(\overline{\mathcal{J}}^{(i+1)})_{jp}\Big)^+ \Big((\underline{\Lambda}^{(i)})_{pkm}y_{i,m} + (\underline{\lambda}^{(i)})_{pk}\Big) \\ &+ \Big(\eta_{jpk}(\underline{\mathcal{J}}^{(i+1)})_{jp} + (1-\eta_{jpk})(\overline{\mathcal{J}}^{(i+1)})_{jp}\Big)^- \Big((\overline{\Lambda}^{(i)})_{pkm}y_{i,m} + (\overline{\lambda}^{(i)})_{pk}\Big) \\ &+ \Big((\underline{\Pi}^{(i+1)})_{jpm}y_{i,m} + (\underline{\pi}^{(i+1)})_{jp}\Big)\Big(\eta_{jpk}(\underline{\mathcal{J}}^{(i)})_{pk} + (1-\eta_{jpk})(\overline{\mathcal{J}}^{(i)})_{pk}\Big)^+ \\ &+ \Big((\overline{\Pi}^{(i+1)})_{jpm}y_{i,m} + (\overline{\pi}^{(i+1)})_{jp}\Big)\Big(\eta_{jpk}(\underline{\mathcal{J}}^{(i)})_{pk} + (1-\eta_{jpk})(\overline{\mathcal{J}}^{(i)})_{pk}\Big)^- \\ &- \eta_{jpk}(\underline{\mathcal{J}}^{(i+1)})_{jp}(\underline{\mathcal{J}}^{(i)})_{pk} - (1-\eta_{jpk})(\overline{\mathcal{J}}^{(i+1)})_{jp}(\overline{\mathcal{J}}^{(i)})_{pk}. \end{aligned} \tag{7}$$

To obtain the overall bounds on $\frac{\partial \mathcal{B}_\theta(x)}{\partial x}$, we first bound the product of $\mathcal{J}^{(L)}\mathcal{J}^{(L-1)}$ and then recursively obtain bounds on the product $\prod_{j=i}^{L}\mathcal{J}^{(j)}$ for $i = L-2, L-3, \ldots, 1$ by applying Eq. (7) to bound the product of $\mathcal{J}^{(i)}$ and $\prod_{j=i+1}^{L}\mathcal{J}^{(j)}$. Repeating this process yields the bounds in Eq. (5). ■

The computation of the linear bounds can be efficiently batched, which enables efficient and parallelized use of the GPU for computing bounds on $\frac{\partial \mathcal{B}_\theta(x)}{\partial x}$ over multiple regions Δ simultaneously.

4. Verification Procedure

Having bounded the values and gradients of $\mathcal{B}_\theta$, we now turn to bounding the dynamics in Eq. (1) and combining all bounds into a conservative but linear surrogate for the formula ϕ in Eq. (3).

4.1. First-Order Model of the System Dynamics

We derive upper and lower bounds on the nonlinear system dynamics $f(x)$ and $g(x)$ by certified first-order Taylor expansions over the domain $\Delta \subset \mathcal{X}$, which are defined as follows.

Proposition 7 (Certified first-order Taylor expansion) *Let $f : \mathcal{X} \to \mathbb{R}^n$ and $g : \mathcal{X} \to \mathbb{R}^{n\times m}$ be continuously differentiable functions, and let $\Delta \subset \mathcal{X}$ be a convex domain. Then, there exist hyperrectangles $\mathcal{R}_f \subseteq \mathbb{R}^n$ and $\mathcal{R}_g \subseteq \mathbb{R}^{n\times m}$ such that for all $x \in \Delta$,*

$$f(x) \in (f(c) + \nabla_x f(c)(x-c)) \oplus \mathcal{R}_f, \qquad g(x) \in (g(c) + \nabla_x g(c)(x-c)) \oplus \mathcal{R}_g,$$

where $\oplus$ denotes the Minkowski sum and $c \in \Delta$ is the expansion point.

Using the Lagrange error bound, the remainder terms $\mathcal{R}_f$ and $\mathcal{R}_g$ can be efficiently computed when f and g are twice continuously differentiable (see Appendix F). When f or g are not given in

elementary form, linear relaxations can be computed from the (local) Lipschitz constant. In the case that f or g are represented by neural networks, linear bounds can be obtained efficiently using LBP, analogous to the computation of bounds on $\mathcal{B}_\theta(x)$. We denote the linear bounds on $f(x)$, $x \in \Delta$, as

$$\underline{f}(x) \coloneqq A_f x + b_f - \underline{r}_f, \qquad \overline{f}(x) \coloneqq A_f x + b_f + \overline{r}_f,$$

where $\underline{r}_f, \overline{r}_f$ are the lower and upper bounds on $\mathcal{R}_f$. The bounds on $g(x)$ are denoted analogously.

4.2. Linear Bounds on the CBF Constraint

Having derived certified linear bounds for all terms in the CBF conditions, we now construct the linear surrogate of Eq. (3) to solve Problem 5. We denote bounds over the domain of interest Δ as

$$\begin{aligned} \mathcal{B}_{\min} \le \min_{x\in\Delta} \mathcal{B}_\theta(x) \le \mathcal{B}_{\max}, \qquad & \partial\mathcal{B}_{\min} \le \min_{x\in\Delta} \frac{\partial \mathcal{B}_\theta(x)}{\partial x} \le \partial\mathcal{B}_{\max}, \\ \mathcal{F}_{\min} \le \min_{x\in\Delta} f(x) \le \mathcal{F}_{\max}, \qquad & \mathcal{G}_{\min} \le \min_{x\in\Delta} g(x) \le \mathcal{G}_{\max}. \end{aligned} \tag{8}$$

which can be easily computed from the linear bounds already derived in Sects. 3 and 4.1. As in Sect. 3, we define $G^+ = \max(G, 0)$ and $G^- = \min(G, 0)$ for the matrix G. We utilize McCormick relaxations to bound the drift term $\frac{\partial \mathcal{B}_\theta(x)}{\partial x} f(x)$ based on the bounds in Eq. (8):

$$\begin{aligned} \frac{\partial \mathcal{B}_\theta(x)}{\partial x} f(x) \ge (C_{\partial\mathcal{B}}(\eta))^+ \underline{f}(x) + (C_{\partial\mathcal{B}}(\eta))^- \overline{f}(x) + (C_{\mathcal{F}}(\eta))^+ \underline{\partial\mathcal{B}_\theta}(x) \\ + (C_{\mathcal{F}}(\eta))^- \overline{\partial\mathcal{B}_\theta}(x) - \big(\eta \partial\mathcal{B}_{\min}\mathcal{F}_{\min} + (1-\eta)\partial\mathcal{B}_{\max}\mathcal{F}_{\max}\big), \end{aligned} \tag{9}$$

where $\eta \in [0,1]^n$ defines the convex combination of the McCormick inequalities, via $C_{\partial\mathcal{B}}(\eta) \coloneqq \eta\partial\mathcal{B}_{\min} + (1-\eta)\partial\mathcal{B}_{\max}$, and $C_{\mathcal{F}}(\eta) \coloneqq \eta\mathcal{F}_{\min} + (1-\eta)\mathcal{F}_{\max}$. We rewrite Eq. (9) in affine form as

$$\frac{\partial \mathcal{B}_\theta(x)}{\partial x} f(x) \ge \Gamma_{\text{drift,L}}(\eta)\, x + \beta_{\text{drift,L}}(\eta), \tag{10}$$

with the coefficients and constant terms defined as

$$\Gamma_{\text{drift,L}}(\eta) \coloneqq C_{\partial\mathcal{B}}(\eta)\, A_f + (C_{\mathcal{F}}(\eta))^+ \underline{\Pi} + (C_{\mathcal{F}}(\eta))^- \overline{\Pi}, \tag{11}$$

$$\begin{aligned} \beta_{\text{drift,L}}(\eta) \coloneqq C_{\partial\mathcal{B}}(\eta)\, b_f - (C_{\partial\mathcal{B}}(\eta))^+ \underline{r}_f + (C_{\partial\mathcal{B}}(\eta))^- \overline{r}_f + (C_{\mathcal{F}}(\eta))^+ \underline{\pi} \\ + (C_{\mathcal{F}}(\eta))^- \overline{\pi} - \big(\eta\, \partial\mathcal{B}_{\min}\, \mathcal{F}_{\min} + (1-\eta)\, \partial\mathcal{B}_{\max}\, \mathcal{F}_{\max}\big). \end{aligned} \tag{12}$$

Next, to bound the control term, recall from Sect. 2 that $\mathcal{U} = [\underline{u}_j, \overline{u}_j]$, such that $\sup_{u\in\mathcal{U}} J(x)g(x)u = \sum_{j=1}^m \max_{u_j \in [\underline{u}_j, \overline{u}_j]} \Big(\sum_{p=1}^n J_p(x) g_{pj}(x)\Big) u_j$. As with the drift term, we can derive linear bounds for the term $\sum_{p=1}^n J_p(x) g_{pj}(x)$, denoted as $\underline{v}_j(x)$ and $\overline{v}_j(x)$, so that we write the control term as:

$$\sup_{u\in\mathcal{U}} \frac{\partial \mathcal{B}_\theta(x)}{\partial x} g(x) u \ge \sum_{j=1}^m \max\left(\underline{v}_j(x)\underline{u}_j,\, \underline{v}_j(x)\overline{u}_j\right) =: \Gamma_{\text{ctrl,L}}(\eta)\, x + \beta_{\text{ctrl,L}}(\eta). \tag{13}$$

Finally, we combine the bounds on the control and drift term to derive a linear surrogate ϕ_{linear} of the formula ϕ defined in Eq. (3):

$$\phi_{\text{linear}} \coloneqq \forall x \in \mathcal{S} : \left[\overline{\mathcal{B}_\theta}(x) < 0 \vee \psi_{\text{invar}}(x)\right] \wedge\ \forall x \in \mathcal{S}^C : \left[\overline{\mathcal{B}_\theta}(x) < 0\right], \tag{14}$$

where $\psi_{\text{invar}}(x)$ is a SAT formula that represents the CBF invariance condition in Eq. (2):

$$\psi_{\text{invar}}(x) \coloneqq \left[\Gamma_{\text{drift,L}}(\eta) + \Gamma_{\text{ctrl,L}}(\eta) + \alpha \underline{A}_{\mathcal{B}_\theta}\right] x + \beta_{\text{drift,L}}(\eta) + \beta_{\text{ctrl,L}}(\eta) + \alpha \underline{a}_{\mathcal{B}_\theta} \geq 0. \quad (15)$$

The formula ϕ_{linear} can be used as a conservative surrogate of the original formula ϕ, i.e., satisfaction of ϕ_{linear} implies satisfaction of ϕ. This yields the next theorem, for which the proof is in Appendix D:

Theorem 8 *If ϕ_{linear} is satisfied, then ϕ in Eq. (3) is satisfied and $\mathcal{B}_\theta(x)$ is a valid CBF.*

Counterexamples We extend our framework to search for counterexamples of ϕ, i.e., satisfying assignments to the negation of ϕ in Eq. (3). We define the linear surrogate for the negation of ϕ as

$$\phi^{\text{NEG}}_{\text{linear}} \coloneqq \exists x \in \mathcal{S} : \left[\underline{\mathcal{B}_\theta}(x) \geq 0 \wedge \psi^{\text{NEG}}_{\text{invar}}(x)\right] \vee \exists x \in \mathcal{S}^C : \left[\underline{\mathcal{B}_\theta}(x) \geq 0\right], \quad (16)$$

where the SAT formula $\psi^{\text{NEG}}_{\text{invar}}(x)$ is defined as

$$\psi^{\text{NEG}}_{\text{invar}}(x) \coloneqq \left[\Gamma_{\text{drift,U}}(\eta) + \Gamma_{\text{ctrl,U}}(\eta) + \alpha \overline{A}_{\mathcal{B}_\theta}\right] x + \beta_{\text{drift,U}}(\eta) + \beta_{\text{ctrl,U}}(\eta) + \alpha \overline{a}_{\mathcal{B}_\theta} < 0. \quad (17)$$

Note that Eqs. (16) and (17) use the opposite bounds on each term compared to Eqs. (14) and (15). Hence, any satisfying assignment of x to $\phi^{\text{NEG}}_{\text{linear}}$ is also a satisfying assignment to the negation of ϕ and thus proves that $\mathcal{B}_\theta$ is *not* a valid CBF. This ability to provide counterexamples enables the use of our approach in common counterexample-guided learner-verifier frameworks (Peruffo et al., 2021).

4.3. Refinement of the Domains for Linear Bounds

Linear upper and lower bounds are generally overly conservative over large input domains. As a result, for a given domain Δ, the linear surrogates ϕ_{linear} and $\phi^{\text{NEG}}_{\text{linear}}$ might both be unsatisfiable, i.e., the bounds are too loose to obtain a conclusive verification outcome. A standard way to mitigate this conservatism is to employ an adaptive refinement strategy that splits regions in which both ϕ_{linear} and $\phi^{\text{NEG}}_{\text{linear}}$ are not satisfied (Clarke et al., 2003; Dierks et al., 2007; Tiwari and Khanna, 2002). Refinements with hyperrectangular regions are especially common due to their simplicity (Mathiesen et al., 2025; Zikelic et al., 2023; Junges et al., 2024; Badings et al., 2025). Closest to our setting is Mathiesen et al. (2025), which prioritizes refinements in the dimension that contributes most to the Lagrange error bounds of the Taylor expansion used for linearizing the dynamics.

When certifying CBFs, however, identifying suitable refinement directions is substantially more difficult. The CBF invariance condition in Eq. (2) generally couples multiple input dimensions in a nonlinear manner, reducing the effectiveness of refinements based on hyperrectangular regions. To overcome this problem, we instead adopt a *simplicial mesh* over the input domain, i.e., a partition into simplices, and refine regions by splitting them along their longest edge.

Definition 9 (Simplex (Crane, 2018)) *A simplex, Δ_v, is the convex hull in $\mathbb{R}^n$ of $n+1$ affinely independent points $v = \{v_0, \ldots, v_n\}$, called its vertices: $\Delta_v \coloneqq \text{conv}\{v_0, \ldots, v_n\} \subset \mathbb{R}^n$.*

Initially, we partition the input domain $\mathcal{X}$ into a set of simplices $\{\Delta^1_v, \ldots, \Delta^q_v\}$ such that $\mathcal{X} \subseteq \bigcup_{i=1}^q \Delta^i_v$. For each simplex Δ^i_v, we perform the next steps. First, we compute linear upper and lower bounds on $\mathcal{B}_\theta(x)$ and evaluate these affine bounds at the vertices to obtain the maximal and minimal attainable values of $\mathcal{B}_\theta(x)$ within that region (denoted as $\mathcal{B}_{\max}$ and $\mathcal{B}_{\min}$, respectively). Then, we sequentially check for the satisfiability of the linear surrogate formulae ϕ_{linear} and $\phi^{\text{NEG}}_{\text{linear}}$ defined in Eqs. (14) and (16), as shown in Fig. 1. If the verification is inconclusive (i.e., both ϕ_{linear} and $\phi^{\text{NEG}}_{\text{linear}}$ are unsatisfiable), we *split* Δ^i along its longest edge and repeat the verification for both sub-simplices. The satisfiability of ϕ_{linear} for *all* simplices proves that $\mathcal{B}_\theta$ is a valid CBF as per Theorem 8. By contrast, the satisfiability of $\phi^{\text{NEG}}_{\text{linear}}$ for *any* simplex proves that $\mathcal{B}_\theta$ is not a valid CBF.

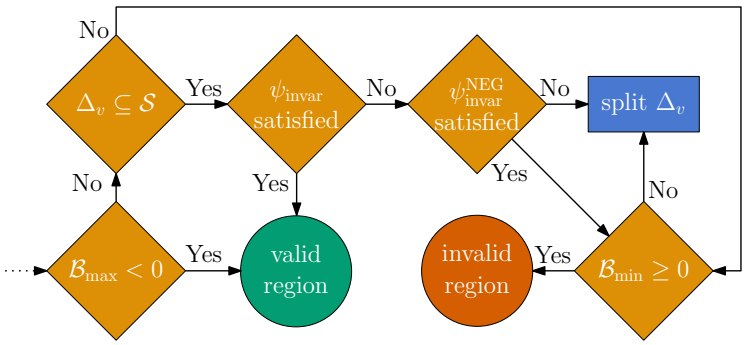

Figure 1: Verification flowchart illustrating the refinement and validation procedure for each simplex.

Batching simplices Our approach enables efficient utilization of computational resources by performing LBP on batches of simplices using GPUs. In particular, we collect simplices that require evaluation of ψ_{invar} into batches and compute the necessary linear bounds on the invariance condition jointly over each batch. By further parallelizing verification over simplices, potentially over multiple GPUs, our approach allows for efficient utilization of computational resources to handle more challenging verification tasks involving larger networks.

5. Experimental Results

We evaluate the effectiveness of our proposed approach in scaling to larger neural network architectures. We adopt standard benchmarks from the literature (Zeng et al., 2016; Zhang et al., 2023; Abate et al., 2021a; Liu et al., 2015; Prajna, 2006; Zhao et al., 2020; Barry et al., 2012), along with two novel benchmarks (*2D Control* and *Cart Pole*) chosen to demonstrate the incorporation of the $\sup_u g(x)u$ term in Eq. (2). Additional details on all benchmarks are provided in Appendix E. Across all experiments, we employ the `tanh` activation function, as it consistently yields the largest control invariant sets. Implementations of the corresponding relaxations for the `sigmoid`, `ReLU`, and `leaky-ReLU` activation functions are included in the accompanying codebase[1]. We fix the McCormick relaxation parameter to $\eta = \frac{1}{2}$, and class-$\mathcal{K}$ function $\alpha = 1$ for all experiments. All computations ran on a machine with an Intel i7-6700K CPU, 16 GB RAM, and an NVIDIA GTX 1060 GPU (6 GB VRAM). Details regarding network training procedures are provided in Appendix A.

Results Fig. 2 presents the verification results for the *Darboux* benchmark. As shown in this figure, regions where the invariance condition (Eq. (15)) or the value of $\mathcal{B}_\theta$ are close to zero require the most refinement, while the mesh can remain coarse in regions where even conservative linearization is adequate to establish validity. The figures for the other benchmarks are presented in Appendix E.

Table 1 summarizes the verification performance across all benchmarks, reporting the network size, the proportion of the state space satisfying the formula ϕ in Eq. (3), and the corresponding verification time. The column "number of regions" denotes the total number of regions examined during verification, including those that required subdivision. Because our networks employ `tanh` activations, a direct comparison with prior ReLU-based verification methods (e.g., Zhang et al.

1. The code can be found at https://github.com/Zinoex/verification-of-neural-cbf-via-lbp.

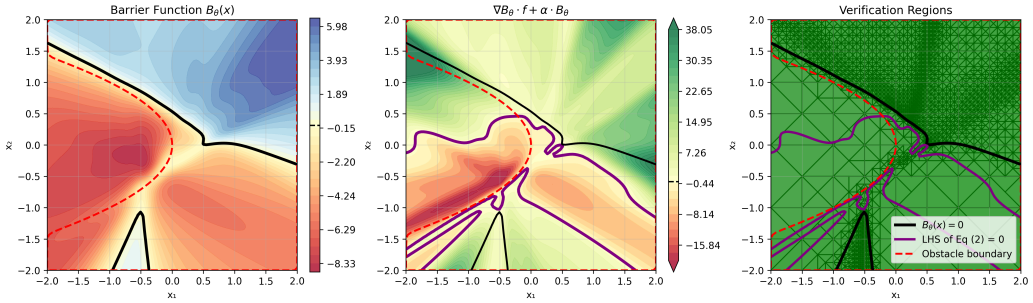

Figure 2: Illustration of the neural control barrier function $\mathcal{B}_\theta(x)$, the CBF invariance condition from Eq. (2), and the resulting verification regions after adaptive mesh refinement.

Table 1: Verification results for the numerical experiments.

		Our approach			`dreal`	
Model	**Network**	**Time (s)**	**Certified (%)**	**# regions**	**Time (s)**	**Result**
Barrier 2	[64, 64]	1.66s	100.0	1432	635.36s	✓
Barrier 3	[64, 64]	5.83s	100.0	3554	1820.33s	✓
Barrier 4	[64, 64]	25.37s	100.0	25446	`Timeout`	-
Darboux	[128, 256, 128]	144.68s	100.0	9142	`Timeout`	-
2D-Control	[64, 64, 8]	5.86s	100.0	4608	`N/A`	
Cart-Pole	[64, 64]	2146.84s	100.0	3026098	`N/A`	

(2023)) is not meaningful. Instead, we compare against the SMT-based verification approaches of Abate et al. (2021a); Edwards et al. (2024), implemented via the SMT solver `dreal` (Gao et al., 2013). For a fair comparison, we use identically trained networks in both verification pipelines. As shown in Table 1, our method achieves substantial performance gains over SMT-based verification, efficiently accommodates control inputs, and scales to significantly larger neural networks.

6. Conclusion

We have presented a scalable verification framework for neural control barrier functions that effectively handles larger neural network architectures and incorporates control inputs into the verification process. By leveraging linear bound propagation and first-order Taylor expansions, we constructed upper and lower linear bounds on the nonlinearity inherent in the verification task, thereby circumventing the need for SMT solvers capable of reasoning over quantifier-free nonlinear real arithmetic formulae. Our numerical experiments on a variety of benchmarks have demonstrated that our approach applies to a wide class of dynamics and to nonlinear activation functions, outperforms existing SMT-based verification tools, and scales to significantly larger networks. Future work will consider extending our verification framework to certificates of specifications beyond invariance, such as for reachability and avoidance.

Acknowledgments

This paper was supported by the EPSRC grant EP/Y028872/1, Mathematical Foundations of Intelligence: An "Erlangen Programme" for AI. We would like to thank Matthew Wicker for the helpful discussion and insights on linear bound propagation.

References

Alessandro Abate, Cristina David, Pascal Kesseli, Daniel Kroening, and Elizabeth Polgreen. Counterexample guided inductive synthesis modulo theories. In *CAV (1)*, volume 10981 of *Lecture Notes in Computer Science*, pages 270–288. Springer, 2018. doi: 10.1007/978-3-319-96145-3_15.

Alessandro Abate, Daniele Ahmed, Alec Edwards, Mirco Giacobbe, and Andrea Peruffo. FOSSIL: a software tool for the formal synthesis of Lyapunov functions and barrier certificates using neural networks. In *HSCC*, pages 24:1–24:11. ACM, 2021a. doi: 10.1145/3447928.3456646.

Alessandro Abate, Daniele Ahmed, Mirco Giacobbe, and Andrea Peruffo. Formal synthesis of lyapunov neural networks. *IEEE Control. Syst. Lett.*, 5(3):773–778, 2021b. doi: 10.1109/LCSYS.2020.3005328.

Ayush Agrawal and Koushil Sreenath. Discrete control barrier functions for safety-critical control of discrete systems with application to bipedal robot navigation. In *Robotics: Science and Systems*, 2017. doi: 10.15607/RSS.2017.XIII.073.

Aaron D. Ames, Jessy W. Grizzle, and Paulo Tabuada. Control barrier function based quadratic programs with application to adaptive cruise control. In *CDC*, pages 6271–6278. IEEE, 2014. doi: 10.1109/CDC.2014.7040372.

Aaron D. Ames, Xiangru Xu, Jessy W. Grizzle, and Paulo Tabuada. Control barrier function based quadratic programs for safety critical systems. *IEEE Trans. Autom. Control.*, 62(8):3861–3876, 2017. doi: 10.1109/TAC.2016.2638961.

Aaron D. Ames, Samuel Coogan, Magnus Egerstedt, Gennaro Notomista, Koushil Sreenath, and Paulo Tabuada. Control barrier functions: Theory and applications. In *ECC*, pages 3420–3431. IEEE, 2019. doi: 10.23919/ECC.2019.8796030.

Thom Badings, Wietze Koops, Sebastian Junges, and Nils Jansen. Policy verification in stochastic dynamical systems using logarithmic neural certificates. In *CAV (2)*, volume 15932 of *Lecture Notes in Computer Science*, pages 349–375. Springer, 2025. doi: 10.1007/978-3-031-98679-6_16.

Andrew J. Barry, Anirudha Majumdar, and Russ Tedrake. Safety verification of reactive controllers for UAV flight in cluttered environments using barrier certificates. In *ICRA*, pages 484–490. IEEE, 2012. doi: 10.1109/ICRA.2012.6225351.

Joseph Breeden and Dimitra Panagou. Robust control barrier functions under high relative degree and input constraints for satellite trajectories. *Autom.*, 155:111109, 2023. doi: 10.1016/J.AUTOMATICA.2023.111109.

Andrew Clark. Verification and synthesis of control barrier functions. In *CDC*, pages 6105–6112. IEEE, 2021. doi: 10.1109/CDC45484.2021.9683520.

Edmund M. Clarke, Orna Grumberg, Somesh Jha, Yuan Lu, and Helmut Veith. Counterexample-guided abstraction refinement for symbolic model checking. *J. ACM*, 50(5):752–794, 2003. doi: 10.1145/876638.876643.

Keenan Crane. Discrete differential geometry: An applied introduction. *Notices of the AMS, Communication*, 1153, 2018.

Charles Dawson, Sicun Gao, and Chuchu Fan. Safe control with learned certificates: A survey of neural lyapunov, barrier, and contraction methods for robotics and control. *IEEE Trans. Robotics*, 39(3):1749–1767, 2023. doi: 10.1109/TRO.2022.3232542.

Henning Dierks, Sebastian Kupferschmid, and Kim Guldstrand Larsen. Automatic abstraction refinement for timed automata. In *FORMATS*, volume 4763 of *Lecture Notes in Computer Science*, pages 114–129. Springer, 2007. doi: 10.1007/978-3-540-75454-1_10.

Alec Edwards, Andrea Peruffo, and Alessandro Abate. Fossil 2.0: Formal certificate synthesis for the verification and control of dynamical models. In *HSCC*, pages 26:1–26:10. ACM, 2024. doi: 10.1145/3641513.3651398.

Francisco Eiras, Adel Bibi, Rudy Bunel, Krishnamurthy Dj Dvijotham, Philip Torr, and M. Pawan Kumar. Efficient error certification for physics-informed neural networks. In *ICML*. OpenReview.net, 2024.

Sicun Gao, Soonho Kong, and Edmund M. Clarke. dreal: An SMT solver for nonlinear theories over the reals. In *CADE*, volume 7898 of *Lecture Notes in Computer Science*, pages 208–214. Springer, 2013. doi: 10.1007/978-3-642-38574-2_14.

Kai-Chieh Hsu, Haimin Hu, and Jaime F. Fisac. The safety filter: A unified view of safety-critical control in autonomous systems. *Annu. Rev. Control. Robotics Auton. Syst.*, 7(1), 2024. doi: 10.1146/ANNUREV-CONTROL-071723-102940.

Hanjiang Hu, Yujie Yang, Tianhao Wei, and Changliu Liu. Verification of neural control barrier functions with symbolic derivative bounds propagation. In *CoRL*, volume 270 of *Proceedings of Machine Learning Research*, pages 1797–1814. PMLR, 2024.

Sebastian Junges, Erika Ábrahám, Christian Hensel, Nils Jansen, Joost-Pieter Katoen, Tim Quatmann, and Matthias Volk. Parameter synthesis for markov models: covering the parameter space. *Formal Methods Syst. Des.*, 62(1):181–259, 2024. doi: 10.1007/S10703-023-00442-X.

H. Kimura and S. Kobayashi. Stochastic real-valued reinforcement learning to solve a nonlinear control problem. In *IEEE SMC'99 Conference Proceedings. 1999 IEEE International Conference on Systems, Man, and Cybernetics (Cat. No.99CH37028)*, volume 5, pages 510–515 vol.5, 1999. doi: 10.1109/ICSMC.1999.815604.

Jiang Liu, Naijun Zhan, Hengjun Zhao, and Liang Zou. Abstraction of elementary hybrid systems by variable transformation. In *FM*, volume 9109 of *Lecture Notes in Computer Science*, pages 360–377. Springer, 2015. doi: 10.1007/978-3-319-19249-9_23.

Frederik Baymler Mathiesen, Simeon C. Calvert, and Luca Laurenti. Safety certification for stochastic systems via neural barrier functions. *IEEE Control. Syst. Lett.*, 7:973–978, 2023. doi: 10.1109/LCSYS.2022.3229865.

Frederik Baymler Mathiesen, Nikolaus Vertovec, Francesco Fabiano, Luca Laurenti, and Alessandro Abate. Certified neural approximations of nonlinear dynamics. *CoRR*, abs/2505.15497, 2025. doi: 10.48550/ARXIV.2505.15497.

Garth P. McCormick. Computability of global solutions to factorable nonconvex programs: Part I - convex underestimating problems. *Math. Program.*, 10(1):147–175, 1976. doi: 10.1007/BF01580665.

Andrea Peruffo, Daniele Ahmed, and Alessandro Abate. Automated and formal synthesis of neural barrier certificates for dynamical models. In *TACAS (1)*, volume 12651 of *Lecture Notes in Computer Science*, pages 370–388. Springer, 2021. doi: 10.1007/978-3-030-72016-2_20.

Stephen Prajna. Barrier certificates for nonlinear model validation. *Autom.*, 42(1):117–126, 2006. doi: 10.1016/J.AUTOMATICA.2005.08.007.

Meng Sha, Xin Chen, Yuzhe Ji, Qingye Zhao, Zhengfeng Yang, Wang Lin, Enyi Tang, Qiguang Chen, and Xuandong Li. Synthesizing barrier certificates of neural network controlled continuous systems via approximations. In *DAC*, pages 631–636. IEEE, 2021. doi: 10.1109/DAC18074.2021.9586327.

Wenceslao Shaw-Cortez, Denny Oetomo, Chris Manzie, and Peter Choong. Control barrier functions for mechanical systems: Theory and application to robotic grasping. *IEEE Trans. Control. Syst. Technol.*, 29(2):530–545, 2021. doi: 10.1109/TCST.2019.2952317.

Zhouxing Shi, Qirui Jin, Zico Kolter, Suman Jana, Cho-Jui Hsieh, and Huan Zhang. Neural network verification with branch-and-bound for general nonlinearities. In *TACAS (1)*, volume 15696 of *Lecture Notes in Computer Science*, pages 315–335. Springer, 2025. doi: 10.1007/978-3-031-90643-5_17.

Ashish Tiwari and Gaurav Khanna. Series of abstractions for hybrid automata. In *HSCC*, volume 2289 of *Lecture Notes in Computer Science*, pages 465–478. Springer, 2002. doi: 10.1007/3-540-45873-5_36.

Kim P. Wabersich, Andrew J. Taylor, Jason J. Choi, Koushil Sreenath, Claire J. Tomlin, Aaron D. Ames, and Melanie N. Zeilinger. Data-driven safety filters: Hamilton-jacobi reachability, control barrier functions, and predictive methods for uncertain systems. *IEEE Control Systems Magazine*, 43(5):137–177, 2023. doi: 10.1109/MCS.2023.3291885.

Han Wang, Kostas Margellos, and Antonis Papachristodoulou. Assessing Safety for Control Systems Using Sum-of-Squares Programming. In *Polynomial Optimization, Moments, and Applications*, volume 206, pages 207–234. Springer Nature Switzerland, Cham, 2023. ISBN 978-3-031-38658-9 978-3-031-38659-6. doi: 10.1007/978-3-031-38659-6_7.

Matthew Wicker, Juyeon Heo, Luca Costabello, and Adrian Weller. Robust explanation constraints for neural networks. In *ICLR*. OpenReview.net, 2023.

Wei Xiao and Calin Belta. High-order control barrier functions. *IEEE Trans. Autom. Control.*, 67(7): 3655–3662, 2022. doi: 10.1109/TAC.2021.3105491.

Xia Zeng, Wang Lin, Zhengfeng Yang, Xin Chen, and Lilei Wang. Darboux-type barrier certificates for safety verification of nonlinear hybrid systems. In *EMSOFT*, pages 11:1–11:10. ACM, 2016. doi: 10.1145/2968478.2968484.

Hongchao Zhang, Junlin Wu, Yevgeniy Vorobeychik, and Andrew Clark. Exact verification of relu neural control barrier functions. In *NeurIPS*, 2023.

Huan Zhang, Tsui-Wei Weng, Pin-Yu Chen, Cho-Jui Hsieh, and Luca Daniel. Efficient neural network robustness certification with general activation functions. In *NeurIPS*, pages 4944–4953, 2018.

Hengjun Zhao, Xia Zeng, Taolue Chen, and Zhiming Liu. Synthesizing barrier certificates using neural networks. In *HSCC*, pages 25:1–25:11. ACM, 2020. doi: 10.1145/3365365.3382222.

Qingye Zhao, Xin Chen, Zhuoyu Zhao, Yifan Zhang, Enyi Tang, and Xuandong Li. Verifying neural network controlled systems using neural networks. In *HSCC*, pages 3:1–3:11. ACM, 2022. doi: 10.1145/3501710.3519511.

Dorde Zikelic, Mathias Lechner, Thomas A. Henzinger, and Krishnendu Chatterjee. Learning control policies for stochastic systems with reach-avoid guarantees. In *AAAI*, pages 11926–11935. AAAI Press, 2023. doi: 10.1609/AAAI.V37I10.26407.

Appendix A. Neural Network Training

Learning a candidate barrier function is inherently challenging and somewhat heuristic; it is not the primary focus of this work. We therefore briefly summarize the training procedure used to obtain $\mathcal{B}_\theta$, which is subsequently employed in our verification benchmarks.

The neural network $\mathcal{B}_\theta$ is trained using a two-phase curriculum learning scheme. In *Phase 1*, the model learns to distinguish between safe and unsafe regions, while in *Phase 2*, the CBF invariance condition is introduced to enforce forward invariance of the safe set. During training, batches of samples $\mathcal{D}$ are periodically generated and divided into safe and unsafe subsets, $\mathcal{D}_S$ and $\mathcal{D}_U$, respectively. We also define $\mathcal{D}_U^p$ as the top p-th percentile of unsafe samples with the largest $\mathcal{B}_\theta(x)$ values, which are penalized more strongly to refine the decision boundary.

Margins δ_{safe} and δ_{unsafe} encourage separation between regions, improving numerical stability. The total loss minimized during training is:

$$\mathcal{L}_{\text{total}} = \lambda_{\text{safe}}\mathcal{L}_{\text{safe}} + \lambda_{\text{unsafe}}\mathcal{L}_{\text{unsafe}} + \lambda_{\text{unsafe-max}}\mathcal{L}_{\text{unsafe-max}} + \lambda_{\text{CBF}}\mathcal{L}_{\text{CBF}}, \tag{18}$$

$$\mathcal{L}_{\text{safe}} = \mathbb{E}_{x\in\mathcal{D}_S}\left[\text{softplus}_{\beta_{\text{safe}}}(\delta_{\text{safe}} - \mathcal{B}_\theta(x))\right], \tag{19}$$

$$\mathcal{L}_{\text{unsafe}} = \mathbb{E}_{x\in\mathcal{D}_U}\left[\text{softplus}_{\beta_{\text{unsafe}}}(\mathcal{B}_\theta(x) + \delta_{\text{unsafe}})\right], \tag{20}$$

$$\mathcal{L}_{\text{unsafe-max}} = \mathbb{E}_{x\in\mathcal{D}_U^p}\left[\text{softplus}_{\beta_{\text{unsafe}}}(\mathcal{B}_\theta(x) + \delta_{\text{unsafe}})\right], \tag{21}$$

$$\mathcal{L}_{\text{CBF}} = \mathbb{E}_{x\in\mathcal{D}_S}\left[\text{softplus}_{\beta_{\text{CBF}}}\left(-\left(\nabla_x\mathcal{B}_\theta(x)f(x) + \max_{u\in\mathcal{U}}\nabla_x\mathcal{B}_\theta(x)g(x)\,u + \alpha\,\mathcal{B}_\theta(x)\right)\right)\right], \tag{22}$$

where $\text{softplus}_\beta(z) = \frac{1}{\beta}\log(1 + e^{\beta z})$ denotes a smooth approximation to $\max(0, z)$ with sharpness parameter β. We use distinct β values for stability and gradient quality: $\beta_{\text{safe}} = 100, \beta_{\text{unsafe}} = \beta_{\text{CBF}} = 5$. Optimization employs the AdamW optimizer with weight decay regularization. Phase 1 uses cosine annealing for smooth learning rate decay, while Phase 2 employs adaptive reduction of the learning rate upon plateau detection. The weight λ_{CBF} is gradually increased during the transition between phases to ensure stable curriculum progression. Hyperparameters are tuned using Bayesian optimization, and the final configuration is provided in the accompanying repository.

Appendix B. Linear Bound Propagation: Additional Remarks

To construct linear bounds on $\mathcal{B}_\theta$ over a convex domain $\Delta \subset \mathcal{X}$, we apply the LBP procedure introduced by Zhang et al. (2018). For each layer i of the network, we consider linear relaxations of the activation function $\sigma^{(i)}$ of the form

$$\underline{G}_m^{(i)} y_{i,m} + \underline{g}_m^{(i)} \le \sigma^{(i)}(y_{i,m}) \le \overline{G}_m^{(i)} y_{i,m} + \overline{g}_m^{(i)}, \tag{23}$$

where the coefficients $\underline{G}_m^{(i)}, \overline{G}_m^{(i)}, \underline{g}_m^{(i)}$, and $\overline{g}_m^{(i)}$ are computed over the projection of Δ to the i^{th} layer. We provide further details on linear relaxation of the activation function in Appendix C. By substituting these bounds into each layer, we bound the pre-activation output y_{i+1} as a function of y_i as

$$y_{i+1} \ge (W^{(i+1)})^+\left(\underline{G}^{(i)} y_i + \underline{g}^{(i)}\right) + (W^{(i+1)})^-\left(\overline{G}^{(i)} y_i + \overline{g}^{(i)}\right) + b^{(i+1)}, \tag{24}$$

$$y_{i+1} \le (W^{(i+1)})^+\left(\overline{G}^{(i)} y_i + \overline{g}^{(i)}\right) + (W^{(i+1)})^-\left(\underline{G}^{(i)} y_i + \underline{g}^{(i)}\right) + b^{(i+1)}, \tag{25}$$

By iterating this procedure from the input layer onward, we obtain linear bounds (with appropriate coefficients $\underline{A}_{\mathcal{B}_\theta}, \overline{a}_{\mathcal{B}_\theta}, \underline{a}_{\mathcal{B}_\theta}$, and $\overline{a}_{\mathcal{B}_\theta}$) on the value $\mathcal{B}_\theta(x)$ of the network for all $x \in \Delta$ of the form

$$\underline{\mathcal{B}_\theta}(x) := \underline{A}_{\mathcal{B}_\theta} x + \underline{a}_{\mathcal{B}_\theta} \leq \mathcal{B}_\theta(x) \leq \overline{A}_{\mathcal{B}_\theta} x + \overline{a}_{\mathcal{B}_\theta} =: \overline{\mathcal{B}_\theta}(x), \quad \forall x \in \Delta.$$

For deriving the linear bounds on the gradient, we also require linear bounds on the activation derivative $\sigma^{(i)'}(y_{i,m})$ over the projection of the domain Δ onto the i^{th} layer:

$$\underline{S}_m^{(i)} y_{i,m} + \underline{s}_m^{(i)} \leq \sigma^{(i)'}(y_{i,m}) \leq \overline{S}_m^{(i)} y_{i,m} + \overline{s}_m^{(i)}. \tag{26}$$

These can be used to bound $\mathcal{J}^{(i)}(y_i)$. Each element (p,k) of the Jacobian $\mathcal{J}^{(i)}(y_i)$ can be expressed as $(\mathcal{J}^{(i)})_{pk}(y_i) = (\sigma^{(i)'}(y_i))_p W_{pk}^{(i)}$. Since the weight $W_{pk}^{(i)}$ is constant, we can construct affine bounds for $(\mathcal{J}^{(i)}(y_i))_{pk}$ by multiplying the weight by the activation derivative bounds. For the lower bound, we obtain the following.

$$(\underline{S}_p^{(i)} y_{i,p} + \underline{s}_p^{(i)})(W_{pk}^{(i)})^+ + (\overline{S}_p^{(i)} y_{i,p} + \overline{s}_p^{(i)})(W_{pk}^{(i)})^- \leq (\mathcal{J}^{(i)}(y_i))_{pk}, \tag{27}$$

The derivation of the upper bound is analogous to that of the lower bound and is thus omitted for brevity. Using the bounds on $\sigma^{(i)}(y_i)$ from Eq. (23), we obtain the following lower bound for y_{i+1} as a function of y_i:

$$\begin{aligned}
\underline{y}_{i+1,j}(y_i) &= (W_{jm}^{(i+1)})^+ (\underline{G}_m^{(i)} y_{i,m} + \underline{g}_m^{(i)}) + (W_{jm}^{(i+1)})^- (\overline{G}_m^{(i)} y_{i,m} + \overline{g}_m^{(i)}) + b_j^{(i+1)} \\
&= \underbrace{\left((W_{jm}^{(i+1)})^+ \underline{G}_m^{(i)} + (W_{jm}^{(i+1)})^- \overline{G}_m^{(i)} \right)}_{(\underline{K})_{jm}} y_{i,m} + \underbrace{\left(b_j^{(i+1)} + \sum_m (W_{jm}^{(i+1)})^+ \underline{g}_m^{(i)} + (W_{jm}^{(i+1)})^- \overline{g}_m^{(i)} \right)}_{(\underline{k})_j}.
\end{aligned}$$

Substituting the bounds on y_{i+1} into the expression for the lower bound of $(\mathcal{J}^{(i+1)}(y_i))_{jp}$ yields:

$$\begin{aligned}
&(W_{jp}^{(i+1)})^+ \left(\underline{S}_j^{(i+1)} \left(\sum_m (\underline{K})_{jm} y_{i,m} + (\underline{k})_j \right) + \underline{s}_j^{(i+1)} \right) \\
&\quad + (W_{jp}^{(i+1)})^- \left(\overline{S}_j^{(i+1)} \left(\sum_m (\overline{K})_{jm} y_{i,m} + (\overline{k})_j \right) + \overline{s}_j^{(i+1)} \right) \leq (\mathcal{J}^{(i)}(y_i))_{pk}.
\end{aligned}$$

To obtain the interval bounds on $(\mathcal{J}^{(i)}(y_i))_{jp}$ over the domain Δ, as used in the McCormick constraints (Eq. (7)), we project bounds on Δ through each layer according to Eqs. (24) and (25). For a simplex, this constitutes projecting each of the vertices and taking the minimum and maximum over the projected vertex.

Appendix C. Linear Bounds on the Activation Functions

For a given activation function $\sigma(y)$ and input interval $[l,u]$, we construct affine relaxations for the activation function, defined as

$$\underline{G}_m y_m + \underline{g}_m \;\leq\; \sigma(y_m) \;\leq\; \overline{G}_m y_m + \overline{g}_m,$$

and corresponding linear bounds on the derivative:

$$\underline{S}_m y_m + \underline{s}_m \;\leq\; \sigma'(y_m) \;\leq\; \overline{S}_m y_m + \overline{s}_m.$$

A. ReLU Relaxation

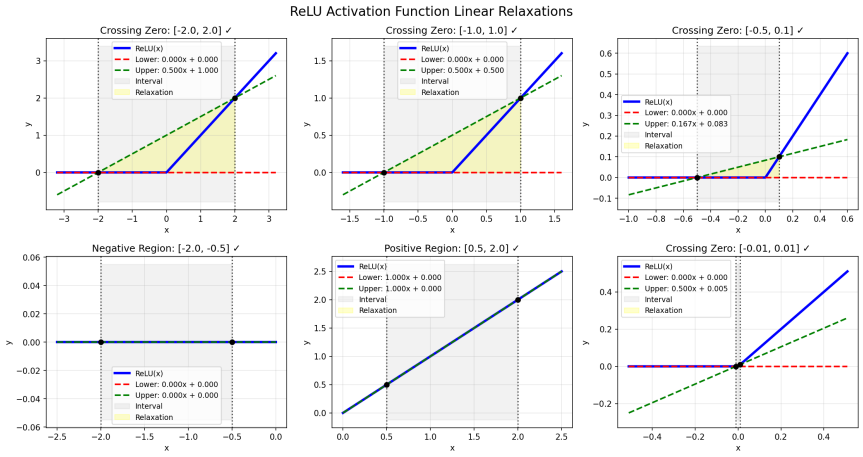

Figure 3: ReLU Relaxation.

For the Rectified Linear Unit $\sigma(y) = \max(0, y)$, the relaxation depends on whether the interval crosses zero.

- **Active region** ($l \geq 0$):

$$\underline{G}_m = \overline{G}_m = 1, \qquad \underline{g}_m = \overline{g}_m = 0.$$
$$\underline{S}_m = \overline{S}_m = 0, \qquad \underline{s}_m = \overline{s}_m = 1.$$

- **Inactive region** ($u \leq 0$):

$$\underline{G}_m = \overline{G}_m = 0, \qquad \underline{g}_m = \overline{g}_m = 0.$$
$$\underline{S}_m = \overline{S}_m = 0, \qquad \underline{s}_m = \overline{s}_m = 0.$$

- **Unstable region** ($l < 0 < u$):

$$\underline{G}_m = 0, \quad \overline{G}_m = \frac{u}{u - l}, \qquad \underline{g}_m = 0, \quad \overline{g}_m = -l\,\overline{G}_m.$$
$$\underline{S}_m = \overline{S}_m = 0, \qquad \underline{s}_m = 0, \quad \overline{s}_m = 1,$$

B. Leaky ReLU Relaxation

For the Leaky ReLU function $\sigma(y) = \max(y, \alpha y)$ with $\alpha \in (0, 1)$:

- **Active region** ($l \geq 0$):

$$\underline{G}_m = \overline{G}_m = 1, \qquad \underline{g}_m = \overline{g}_m = 0.$$
$$\underline{S}_m = \overline{S}_m = 0, \qquad \underline{s}_m = \overline{s}_m = 1.$$

- **Negative region** ($u \leq 0$):

$$\underline{G}_m = \overline{G}_m = \alpha, \qquad \underline{g}_m = \overline{g}_m = 0.$$
$$\underline{S}_m = \overline{S}_m = 0, \qquad \underline{s}_m = \overline{s}_m = \alpha.$$

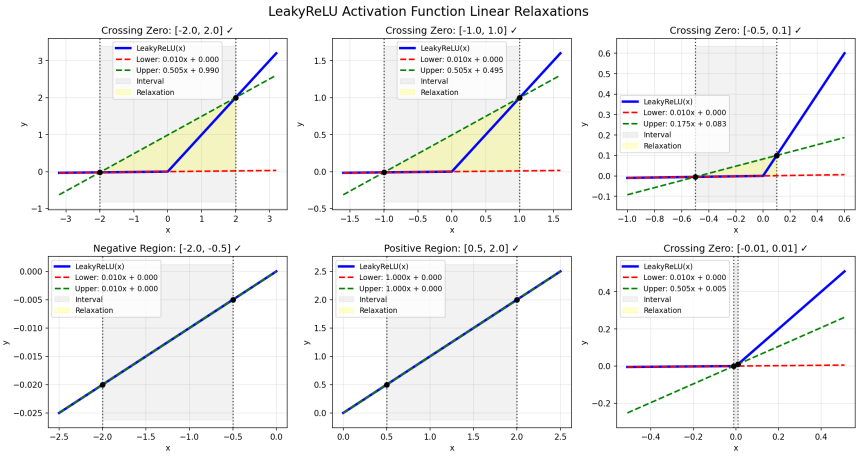

Figure 4: Leaky ReLU Relaxation.

- **Unstable region** ($l < 0 < u$):

$$\underline{G}_m = \alpha, \quad \overline{G}_m = \frac{u - \alpha l}{u - l}, \qquad \underline{g}_m = 0, \quad \overline{g}_m = u - \overline{G}_m\, u.$$
$$\underline{S}_m = \overline{S}_m = 0, \qquad \underline{s}_m = \alpha, \quad \overline{s}_m = 1,$$

C. Sigmoid Relaxation

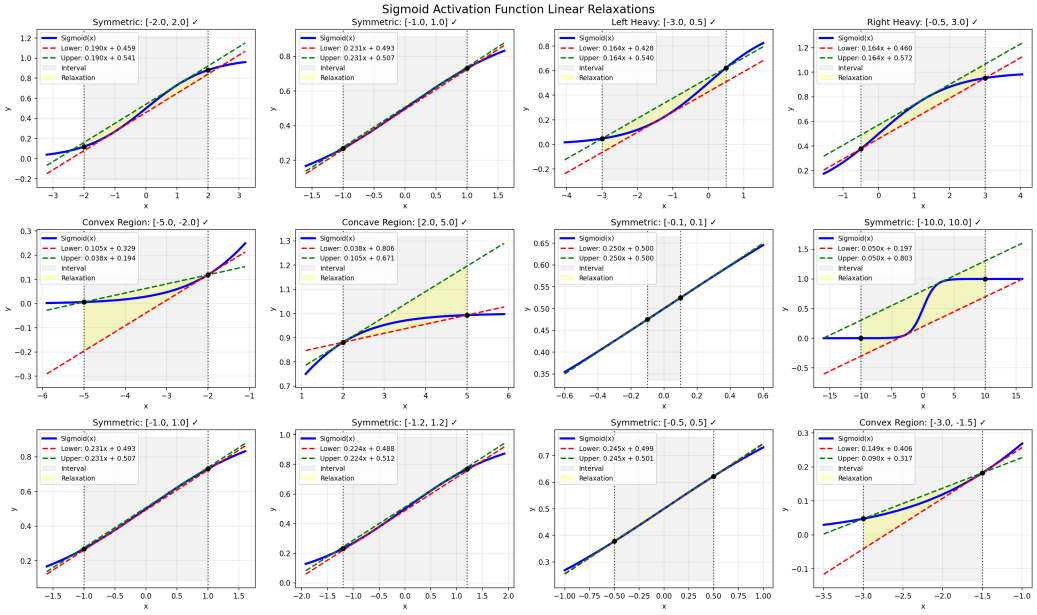

Figure 5: Sigmoid Relaxation.

The sigmoid activation $\sigma(y) = \frac{1}{1+e^{-y}}$ is convex for $y < 0$ and concave for $y > 0$. The derivative has inflection points at $\pm \log \frac{3+\sqrt{3}}{3-\sqrt{3}}$ and is concave in between the inflection points, and convex when

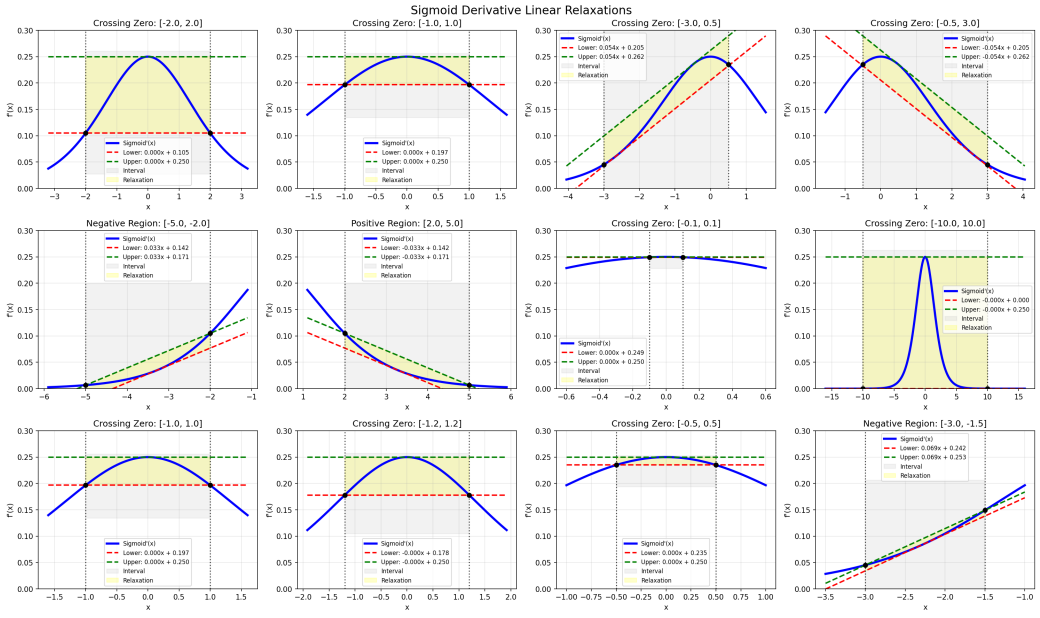

Figure 6: Derivative of Sigmoid Relaxation.

the domain lies outside the inflection points. Define

$$m_{\text{act}} = \frac{\sigma(u) - \sigma(l)}{u - l}, \qquad m_{\text{der}} = \frac{\sigma'(u) - \sigma'(l)}{u - l}.$$

Let $t_\star \in (0,1)$ solve $2t^3 - 3t^2 + t - m_{\text{der}} = 0$ and define $x_\star = \log\frac{t_\star}{1-t_\star} \in (l,u)$. Then

- **Convex region**:

$$\underline{G}_m = \sigma'(u), \quad \overline{G}_m = m_{\text{act}}, \quad \underline{g}_m = \sigma(u) - \sigma'(u)\,u, \quad \overline{g}_m = \sigma(l) - m_{\text{act}}\,l,$$
$$\underline{S}_m = m_{\text{der}}, \quad \overline{S}_m = m_{\text{der}}, \quad \underline{s}_m = \sigma'(x_\star) - m_{\text{der}}\,x_\star, \quad \overline{s}_m = \sigma'(u) - m_{\text{der}}\,u.$$

- **Concave region**:

$$\underline{G}_m = m_{\text{act}}, \quad \overline{G}_m = \sigma'(l), \quad \underline{g}_m = \sigma(l) - m_{\text{act}}\,l, \quad \overline{g}_m = \sigma(l) - \sigma'(l)\,l,$$
$$\underline{S}_m = m_{\text{der}}, \quad \overline{S}_m = m_{\text{der}}, \quad \underline{s}_m = \sigma'(l) - m_{\text{der}}\,l, \quad \overline{s}_m = \sigma'(x_\star) - m_{\text{der}}\,x_\star.$$

- **Mixed region**: Define

$$y_\lambda = \frac{1 - \sqrt{1 - 4\,m_{\text{act}}}}{2}, \quad y_\mu = \frac{1 + \sqrt{1 - 4\,m_{\text{act}}}}{2}, \quad x_\lambda = \log\frac{y_\lambda}{1 - y_\lambda}, \quad x_\mu = -x_\lambda,$$

and let $\{t_i\} \subset (0,1)$ be the real roots of $2t^3 - 3t^2 + t - m_{\text{der}} = 0$ mapped to $x_i = \log\frac{t_i}{1-t_i}$

$$x^- = \arg\min_{x_i} \{\sigma'(x_i) - m_{\text{der}}\,x_i\}, \qquad x^+ = \arg\max_{x_i} \{\sigma'(x_i) - m_{\text{der}}\,x_i\}.$$

Then

$$\underline{G}_m = m_{\text{act}}, \quad \overline{G}_m = m_{\text{act}}, \quad \underline{g}_m = y_\lambda - m_{\text{act}}\,x_\lambda, \quad \overline{g}_m = y_\mu - m_{\text{act}}\,x_\mu, \tag{28}$$
$$\underline{S}_m = m_{\text{der}}, \quad \overline{S}_m = m_{\text{der}}, \quad \underline{s}_m = \sigma'(x^-) - m_{\text{der}}\,x^-, \quad \overline{s}_m = \sigma'(x^+) - m_{\text{der}}\,x^+. \tag{29}$$

D. Tanh Relaxation

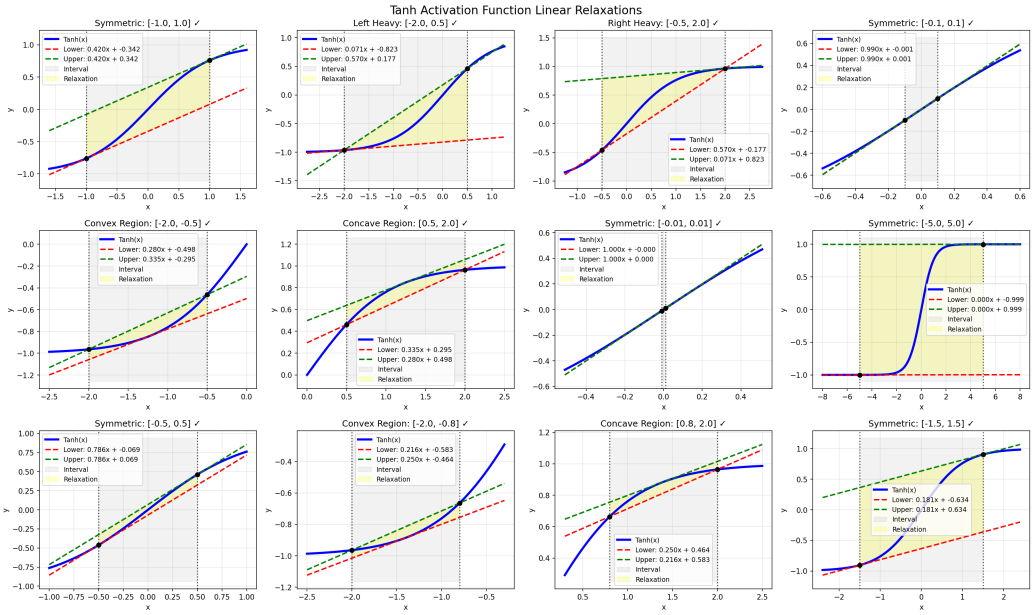

Figure 7: Tanh Relaxation.

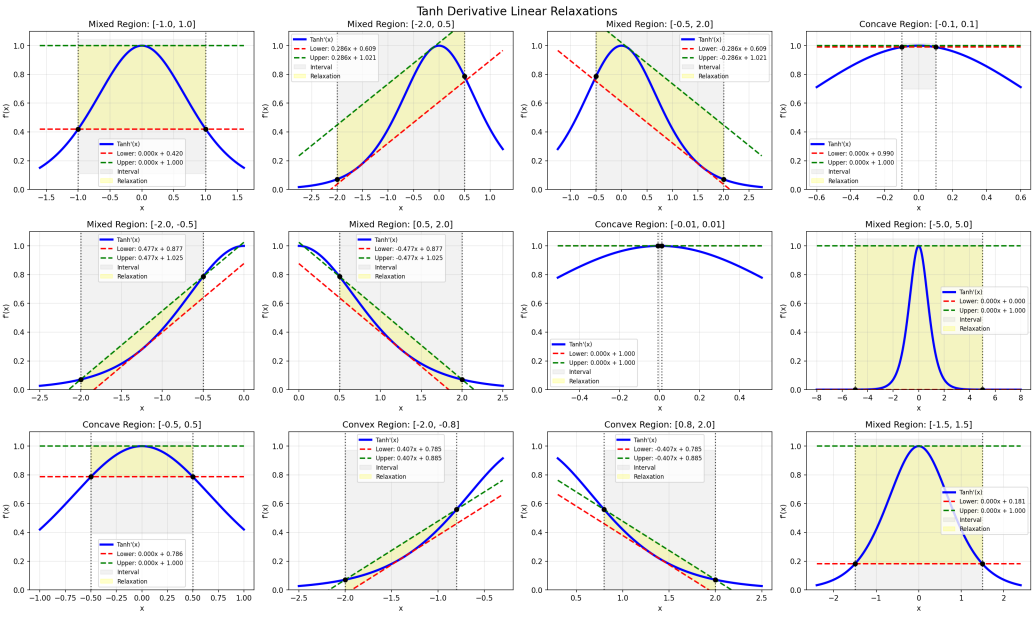

Figure 8: Derivative Tanh Relaxation.

The hyperbolic tangent activation $\sigma(y) = \tanh(y)$ is convex for $y < 0$ and concave for $y > 0$. The derivative has inflection points at $\pm \operatorname{arctanh} \frac{1}{\sqrt{3}}$ and is concave in between the inflection points, and convex when the domain lies outside the inflection points.

Define $m_{\text{act}} = \dfrac{\tanh(u) - \tanh(l)}{u - l}$, $m_{\text{der}} = \dfrac{\tanh^2(l) - \tanh^2(u)}{u - l}$, and the midpoint $m = \frac{l+u}{2}$.

- **Convex region**:

$$\begin{aligned} \underline{G}_m &= 1 - \tanh^2(m), & \underline{g}_m &= \tanh(m) - \underline{G}_m\, m, \\ \overline{G}_m &= m_{\text{act}}, & \overline{g}_m &= \tanh(u) - m_{\text{act}}\, u, \\ \underline{S}_m &= m_{\text{der}}, & \underline{s}_m &= 1 - \tanh^2(x_{\text{tan}}) - m_{\text{der}}\, x_{\text{tan}}, \\ \overline{S}_m &= m_{\text{der}}, & \overline{s}_m &= 1 - \tanh^2(l) - m_{\text{der}}\, l. \end{aligned}$$

- **Concave region**:

$$\begin{aligned} \underline{G}_m &= m_{\text{act}}, & \underline{g}_m &= \tanh(l) - m_{\text{act}}\, l, \\ \overline{G}_m &= 1 - \tanh^2(m), & \overline{g}_m &= \tanh(m) - \overline{G}_m\, m, \\ \underline{S}_m &= m_{\text{der}}, & \underline{s}_m &= 1 - \tanh^2(l) - m_{\text{der}}\, l, \\ \overline{S}_m &= m_{\text{der}}, & \overline{s}_m &= 1 - \tanh^2(x_{\text{tan}}) - m_{\text{der}}\, x_{\text{tan}}. \end{aligned}$$

- **Mixed region**:

$$\begin{aligned} \underline{G}_m &= 1 - \tanh^2(l), & \underline{g}_m &= \tanh(l) - \underline{G}_m\, l, \\ \overline{G}_m &= 1 - \tanh^2(u), & \overline{g}_m &= \tanh(u) - \overline{G}_m\, u, \\ \underline{S}_m &= m_{\text{der}}, & \underline{s}_m &= \min_{x\in[l,u]} \left\{1 - \tanh^2(x) - m_{\text{der}}\, x\right\}, \\ \overline{S}_m &= m_{\text{der}}, & \overline{s}_m &= \max_{x\in[l,u]} \left\{1 - \tanh^2(x) - m_{\text{der}}\, x\right\}. \end{aligned}$$

The tangent points for the derivative envelope are obtained by solving

$$2t^3 - 2t - m_{\text{der}} = 0 \quad \text{for } t = \tanh(x), \qquad x_{\text{tan}} = \operatorname{arctanh}(t) \in (l, u), \tag{30}$$

which specify where the parallel lines of slope m_{der} become tight.

Appendix D. Proof of Theorem 8

Proof By the bounds $\underline{\mathcal{B}_\theta}(x) \le \mathcal{B}_\theta(x) \le \overline{\mathcal{B}_\theta}(x)$, we can certify the sign of $\mathcal{B}_\theta$ from its envelopes. In particular, on $\mathcal{S}^C$, the condition $\overline{\mathcal{B}_\theta}(x) < 0$ implies $\mathcal{B}_\theta(x) < 0$. Likewise, on $\mathcal{S}$, the condition $\underline{\mathcal{B}_\theta}(x) > 0$ implies $\mathcal{B}_\theta(x) > 0$. For invariance, for every $x \in \mathcal{S}$ we have

$$\frac{\partial \mathcal{B}_\theta(x)}{\partial x} f(x) \ge \Gamma_{\text{drift,L}}(\eta)\, x + \beta_{\text{drift,L}}(\eta), \tag{31}$$

$$\sup_{u\in\mathcal{U}} \frac{\partial \mathcal{B}_\theta(x)}{\partial x} g(x)u \ge \Gamma_{\text{ctrl,L}}(\eta)\, x + \beta_{\text{ctrl,L}}(\eta), \tag{32}$$

$$\alpha\, \mathcal{B}_\theta(x) \ge \alpha\, \underline{\Pi}\, x + \underline{\pi}. \tag{33}$$

Adding Eqs. (31) to (33) yields the left-hand side of ψ_{invar}. Since the right-hand side is nonnegative on $\mathcal{S}$ for all $x \in \mathcal{S}$, it follows that

$$\frac{\partial \mathcal{B}_\theta(x)}{\partial x} f(x) + \sup_{u \in \mathcal{U}} \frac{\partial \mathcal{B}_\theta(x)}{\partial x} g(x) u + \alpha \, \mathcal{B}_\theta(x) \geq 0. \tag{34}$$

■

Appendix E. Benchmarks

E.1. 2D-Control

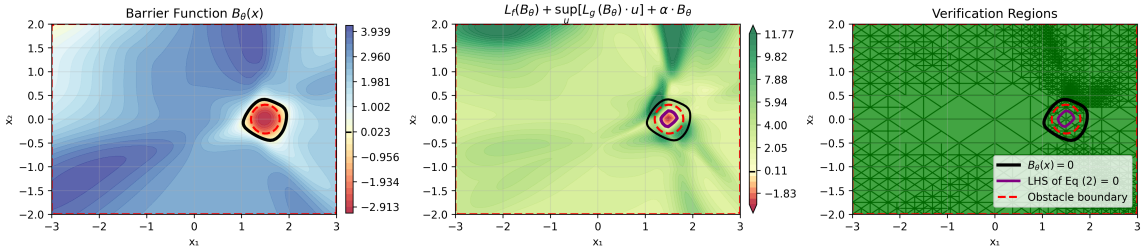

Figure 9: 2D-Control: Illustration of the barrier function $\mathcal{B}_\theta(x)$, the CBF invariance condition, and the resulting verification regions after adaptive mesh refinement.

States $x = (x_1, x_2) \in \mathbb{R}^2$ with affine control $u = (u_1, u_2)$ and dynamics:

$$\dot{x} = \begin{bmatrix} -x_1 x_2 \\ -x_2^2 \end{bmatrix} + \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} u, \tag{35}$$

Control bounds are $u \in [-\frac{1}{2}, \frac{1}{2}]^2$. The state space is $\mathcal{X} = [-3, 3] \times [-2, 2]$, with the safe set defined as $\mathcal{S} = \mathcal{X} \backslash \{(x, y) \in \mathbb{R}^2 \mid \sqrt{(x - 1.5)^2 + y^2} \leq 0.3\}$.

E.2. Cart-Pole

The dynamics for the cart pole are taken from (Kimura and Kobayashi, 1999) with states $x = (y, \dot{y}, \theta, \dot{\theta}) \in \mathbb{R}^4$ and single-input affine control $u = F \in \mathbb{R}$. The parameters are the cart mass m_c, pole mass m_p and length L, gravitational acceleration g and pole-friction coefficient μ_p. The cart-track friction μ_c is neglected.

Define

$$\ddot{\theta}(x) = \frac{g \sin\theta \; + \; \cos\theta \Big(- m_p L \, \dot{\theta}^2 \sin\theta \Big) / (m_c + m_p) \; - \; \mu_p \, \dot{\theta} / (m_p L)}{L \left(\frac{4}{3} \; - \; \dfrac{m_p \cos^2\theta}{m_c + m_p} \right)}, \tag{36}$$

$$\ddot{y}(x) = \frac{m_p L \Big(\dot{\theta}^2 \sin\theta \; - \; \ddot{\theta}(x) \cos\theta \Big)}{m_c + m_p}. \tag{37}$$

Then

$$\dot{x} \;=\; \begin{bmatrix} \dot{y} \\ \ddot{y}(x) \\ \dot{\theta} \\ \ddot{\theta}(x) \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1 - m_p L \cos\theta \, g_{\ddot{\theta}}(x)}{m_c + m_p} \\ 0 \\ g_{\ddot{\theta}}(x) \end{bmatrix} u, \tag{38}$$

where

$$g_{\ddot{\theta}}(x) \;=\; -\frac{\cos\theta}{(m_c + m_p)\, L \left(\frac{4}{3} - \frac{m_p \cos^2\theta}{m_c + m_p} \right)}. \tag{39}$$

The control bounds are $u \in [-u_{\max},\, u_{\max}]$, with $u_{\max} = 10$ N. The state space is

$$\mathcal{X} \;=\; [-2.4,\, 2.4] \times [-3,\, 3] \times \left[-\tfrac{\pi}{6},\, \tfrac{\pi}{6} \right] \times [-2,\, 2] \;\subset\; \mathbb{R}^4. \tag{40}$$

The safe set constrains only the cart position y (within the state space bounds for the other coordinates): $\mathcal{S} \;=\; \{(y, \dot{y}, \theta, \dot{\theta}) \in \mathcal{X} \mid y \in [-2,\, 2]\}$. Parameters used in the benchmark are $m_c = 1.0$ kg, $m_p = 0.1$ kg, $L = 0.5$ m, $g = 9.81$ m/s^2, $\mu_p = 0.01$.

E.3. Darboux (Barrier 1 in Abate et al. (2021b))

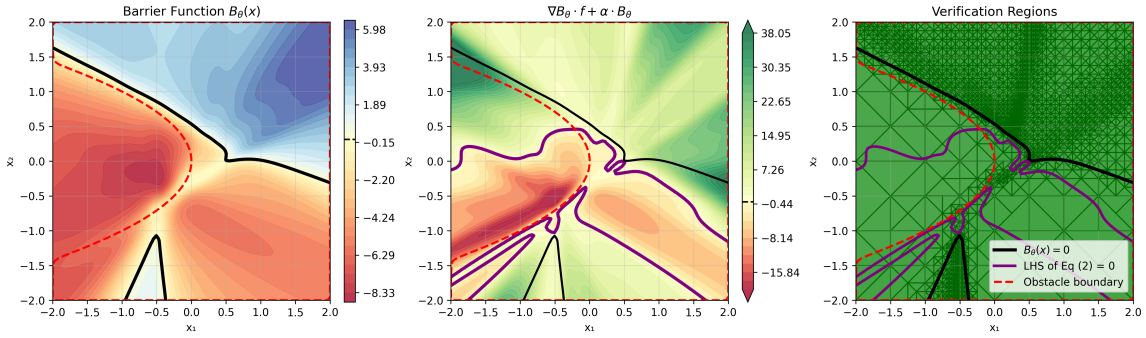

Figure 10: Darboux: Illustration of the barrier function $\mathcal{B}_\theta(x)$, the CBF invariance condition, and the resulting verification regions after adaptive mesh refinement.

Originally presented in Zeng et al. (2016), the system has been reported to not admit an LMI-based barrier function with a degree less than 6 (Abate et al., 2021a), making it a common benchmark to demonstrate the expressivity of neural CBFs. Using a large network and `tanh` activation functions, we are able to obtain a larger control invariant when compared to e.g. Zhang et al. (2023).

States $x = (x_1, x_2) \in \mathbb{R}^2$ and autonomous dynamics:

$$\dot{x} = \begin{bmatrix} x_2 + 2x_1x_2 \\ -x_1 - x_2^2 + 2x_1^2 \end{bmatrix}. \tag{41}$$

The state space is $\mathcal{X} = [-2, 2] \times [-2, 2]$, with the safe set defined as $\mathcal{S} = \mathcal{X} \backslash \{(x_1, x_2) \in \mathbb{R}^2 \mid x_1 + x_2^2 \leq 0\}$.

E.4. Barrier 2

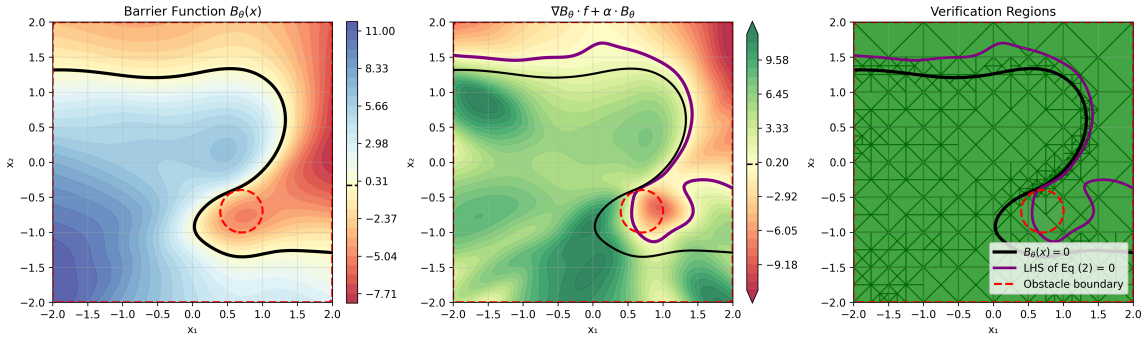

Figure 11: Barrier 2: Illustration of the barrier function $\mathcal{B}_\theta(x)$, the CBF invariance condition, and the resulting verification regions after adaptive mesh refinement.

Presented in Liu et al. (2015) and chosen for its high degree of nonlinearity and non-polynomial (exponential and trigonometric) terms. States $x = (x_1, x_2) \in \mathbb{R}^2$ and autonomous dynamics:

$$\dot{x_1} = \begin{bmatrix} e^{-x_1} + x_2 - 1 \\ -\sin^2(x_1) \end{bmatrix}, \tag{42}$$

The state space is $\mathcal{X} = [-2, 2] \times [-2, 2]$, with the safe set defined as $\mathcal{S} = \mathcal{X} \backslash \{(x, y) \in \mathbb{R}^2 \mid \sqrt{(x - 0.7)^2 + (y + 0.7)^2} \leq 0.3\}$.

E.5. Barrier 3

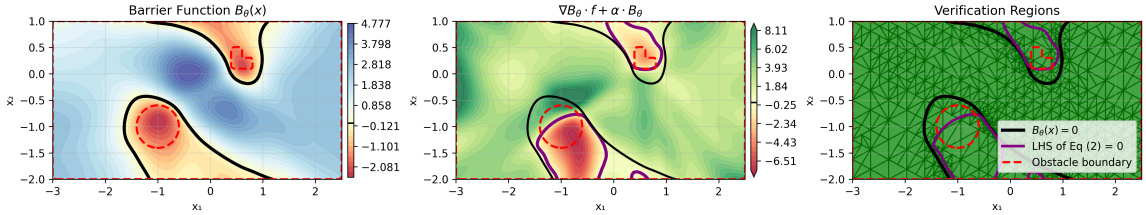

Figure 12: Barrier 3: Illustration of the barrier function $\mathcal{B}_\theta(x)$, the CBF invariance condition, and the resulting verification regions after adaptive mesh refinement.

A modification of the dynamical system presented in Prajna (2006) with non-convex domains, as in Abate et al. (2021a). States $x = (x_1, x_2) \in \mathbb{R}^2$ and autonomous dynamics:

$$\dot{x} = \begin{bmatrix} x_2 \\ -x_1 - x_2 + \frac{1}{3}x_1^3 \end{bmatrix}. \tag{43}$$

The state space is $\mathcal{X} = [-3, 2.5] \times [-2, 1]$, with the safe set defined as $\mathcal{S} = \mathcal{X} \backslash \mathcal{X}_{\text{unsafe}}$. The unsafe safe set $\mathcal{X}_{\text{unsafe}}$ is given by two obstacles, a disk and an L-shaped region. The circle is

$\{(x, y) \mid (x+1)^2 + (y+1)^2 \leq 0.4^2\}$ while the L-shaped region is given by the union of two axis-aligned rectangles, $[[0.4, 0.6] \times [0.1, 0.5]] \cup [[0.4, 0.8] \times [0.1, 0.3]]$.

E.6. Barrier 4

An Unmanned Aerial Vehicle (UAV) avoiding collision with an obstacle as presented in (Barry et al., 2012; Zhang et al., 2023). States $\mathbf{x} = (x, y, \phi) \in \mathbb{R}^3$. The dynamics are

$$\dot{\mathbf{x}} = \begin{bmatrix} v \sin\phi \\ v \cos\phi \\ -\sin\phi \ + \ 3\,\dfrac{x \sin\phi + y \cos\phi}{0.5 + x^2 + y^2} \end{bmatrix}, \tag{44}$$

with $v = 1$. The last component represents a heading-rate term that depends on the current pose (x, y, ϕ). The state space is $\mathcal{X} = [-2, 2] \times [-2, 2] \times [-\frac{\pi}{2}, \frac{\pi}{2}]$, with the safe set defined as $\mathcal{S} = \mathcal{X} \backslash \{(x, y) \in \mathbb{R}^2 \mid \sqrt{x^2 + y^2} \leq 0.2\}$.

Appendix F. Taylor Expansions of Elementary Functions over a Simplex

As certified Taylor expansions have been discussed in the literature, particularly in (Mathiesen et al., 2025), we only briefly mention the extensions required to bound the expansion over a simplex. Let $f : \mathcal{X} \to \mathbb{R}^n$ be continuously differentiable, and let the domain of interest be a simplex

$$\Delta_v = \mathrm{conv}\{v_0, v_1, \ldots, v_n\} = \Big\{ x = \sum_{i=0}^{n} \lambda_i v_i \;\Big|\; \lambda_i \geq 0,\ \sum_i \lambda_i = 1 \Big\}.$$

The barycentric coordinates λ_i parameterize any point $x \in \Delta_v$ as an affine combination of its vertices. We chose the expansion point $c \in \Delta_v$ to be the barycenter $c = \frac{1}{n+1} \sum_i v_i$. The Taylor expansion of f around c is $f(x) = f(c) + \nabla_x f(c)\,(x - c) + R(x)$, where the Lagrange remainder satisfies $R(x) = \frac{1}{2}(x - c)^\top \nabla_x^2 f(\xi_x)(x - c)$, for $\xi_x \in \Delta_v$. To certify the approximation, we construct bounds $[R_{\min}, R_{\max}]$ enclosing all possible values of $R(x)$ over Δ_v. To this end, we consider the quadratic form $q(x) = (J(x - c))^2$ arising in the remainder above, where $J = \nabla_x f(c)$. Over a simplex, the range of any polynomial can be bounded exactly by its *Bernstein coefficients*. Writing a degree-d polynomial $p(x)$ in barycentric form,

$$p(x) = \sum_{|\alpha|=d} b_\alpha\, B_\alpha^d(\lambda(x)), \qquad B_\alpha^d(\lambda) = \frac{d!}{\alpha_0! \ldots \alpha_n!} \lambda_0^{\alpha_0} \ldots \lambda_n^{\alpha_n},$$

we obtain $\min_\alpha b_\alpha \leq p(x) \leq \max_\alpha b_\alpha$, for $x \in \Delta_v$. For the quadratic term, the degree-2 Bernstein coefficients can be computed directly from the vertex evaluations of the linear form $L(x) = J(x - c)$:

$$b_\alpha(q) = \sum_{|\beta|=1, |\gamma|=1} b_\beta(L)\, b_\gamma(L),$$

where each $b_\beta(L)$ equals the value of L at the corresponding vertex v_i. This yields the certified remainder bounds

$$R_{\min} = \tfrac{1}{2} \min_\alpha b_\alpha(q)\, M_{\min}(f''), \qquad R_{\max} = \tfrac{1}{2} \max_\alpha b_\alpha(q)\, M_{\max}(f''),$$

with $M_{\min}(f'')$ and $M_{\max}(f'')$ denoting lower and upper bounds on the Hessian entries of f over Δ_v.