Mixed Monotonicity Reachability Analysis of Neural ODE: A Trade-Off Between Tightness and Efficiency

Abdelrahman Sayed Sayed   ABDELRAHMAN.IBRAHIM@UNIV-EIFFEL.FR
Pierre-Jean Meyer   PIERRE-JEAN.MEYER@UNIV-EIFFEL.FR
Mohamed Ghazel   MOHAMED.GHAZEL@UNIV-EIFFEL.FR
Univ Gustave Eiffel, COSYS-ESTAS, F-59657 Villeneuve d'Ascq, France

Editors: List of editors' names

Abstract

Neural ordinary differential equations (neural ODE) are powerful continuous-time machine learning models for depicting the behavior of complex dynamical systems, but their verification remains challenging due to limited reachability analysis tools adapted to them. We propose a novel interval-based reachability method that leverages continuous-time mixed monotonicity techniques for dynamical systems to compute an over-approximation for the neural ODE reachable sets. By exploiting the geometric structure of full initial sets and their boundaries via the homeomorphism property, our approach ensures efficient bound propagation. By embedding neural ODE dynamics into a mixed monotone system, our interval-based reachability approach, implemented in TIRA with single-step, incremental, and boundary-based approaches, provides sound and computationally efficient over-approximations compared with CORA's zonotopes and NNV2.0 star set representations, while trading tightness for efficiency. This trade-off makes our method particularly suited for high-dimensional, real-time, and safety-critical applications. Applying mixed monotonicity to neural ODE reachability analysis paves the way for lightweight formal analysis by leveraging the symmetric structure of monotone embeddings and the geometric simplicity of interval boxes, opening new avenues for scalable verification. This novel approach is illustrated on two numerical examples of a spiral system and a fixed-point attractor system modeled as a neural ODE.

Keywords: Mixed-monotonicity, Neural ODE, Reachability analysis, Boundary analysis

1. Introduction

Neural ordinary differential equations (neural ODE) are a recent machine learning model that has been introduced into the machine learning community (Rico-Martinez et al., 1992; Chen et al., 2018), and since then, they have gained prominence in time-series modeling over traditional neural networks (Kidger, 2021; Haber and Ruthotto, 2017; Oh et al., 2025). As neural ODE constitute a relatively recent technique, fewer verification tools are available for them compared with those for traditional neural networks. In particular, reachability analysis is a useful and important approach for neural ODE, and most of the current available verification tools for neural ODE are based on reachability analysis.

The first work on neural ODE reachability analysis was introduced in Grunbacher et al. (2021) using Stochastic Lagrangian Reachability, an abstraction-based technique that computes confidence intervals for reachable sets with probabilistic guarantees. This work was later extended to the *GoTube* tool (Gruenbacher et al., 2022), enabling computation of neural ODE reachable sets over longer time horizons. However, both Grunbacher et al. (2021)

and Gruenbacher et al. (2022) provide only stochastic bounds on reachable sets, lacking formal guarantees. For deterministic neural ODE reachability analysis, Manzanas Lopez et al. (2022) have extended the neural network verification tool (*NNV*) (Tran et al., 2020) by introducing *NNVODE*, a general neural ODE class combining continuous and discrete time layers. *NNV* was later upgraded to *NNV 2.0* (Lopez et al., 2023), which offers support for neural ODE. A different form of reachability based formal verification for neural ODE was introduced in Sayed et al. (2025), where a formal relationship between neural ODE and ResNet was introduced to verify safety properties of one model based on the other without running verification tools twice on both models.

Mixed Monotonicity was first introduced in Gouzé and Hadeler (1994) and then introduced into reachability analysis field in Coogan and Arcak (2015). The first characterization of continuous-time systems that satisfy mixed monotonicity was investigated in Coogan and Arcak (2016); Coogan et al. (2016), showing that systems with sign-stable Jacobian matrices satisfy the mixed monotonicity property. This was then generalized for nonlinear systems in Yang et al. (2019), provided the Jacobian matrices are bounded. Such a result was subsequently applied to the neural network reachability analysis in Meyer (2022), and Coogan (2020) further discussed how mixed monotonicity enables efficient reachable set approximation for safety critical dynamical systems.

Some of the tools that consider mixed monotonicity for neural network reachability analysis include *immrax*, a *JAX*-based tool that employs mixed monotonicity to compute interval-based over-approximations of reachable sets for neural networks, utilizing GPU acceleration for computational efficiency (Harapanahalli et al., 2024). Additionally, *npinterval* implemented in *numpy*, uses inclusion functions to provide interval bounds on a function's output (Harapanahalli et al., 2023). To the best of our knowledge, no other work has considered mixed monotonicity for the reachability analysis of neural ODE.

Our Contributions: We present a novel method for the reachability analysis of neural ODE by adapting existing mixed monotonicity reachability methods for continuous-time dynamical systems (Meyer et al., 2021) to obtain an interval over-approximation of the reachable output set starting from a given input set. We also compare our novel neural ODE mixed monotonicity reachability approach implemented in TIRA (Meyer et al., 2019) against two well-known reachability analysis toolboxes that can handle neural ODE which are CORA (Althoff, 2015) and NNV2.0 (Lopez et al., 2023), in addition to a boundary analysis-based reachability approach inspired by the work of Liang et al. (2023).

2. Preliminaries

We consider the following neural ODE:

$$\dot{x}(t) = \frac{dx(t)}{dt} = f(x(t)), \tag{1}$$

with state $x \in \mathbb{R}^n$, initial state $x(0) = u$, and vector field $f : \mathbb{R}^n \to \mathbb{R}^n$ defined as a finite sequence of classical neural network layers (such as fully connected layers, convolutional layers, activation functions, batch normalization). Although the proposed approach in this paper is applicable to any neural ODE with Lipschitz continuous vector field f, for simplicity

of presentation in the notations relying on the derivatives of f, we restrict ourselves to the cases where the vector field is continuously differentiable.

Definition 1 (neural ODE Reachability) *Given an initial set $\mathcal{X}_{in} \subseteq \mathbb{R}^n$ and final time t_f for the neural ODE, we define the set of reachable outputs as:*

$$\mathcal{R}_{neural\ ODE}(\mathcal{X}_{in}) = \{y \in \mathbb{R}^n \mid y = \Phi(t_f, u),\ u \in \mathcal{X}_{in}\},$$

where $\Phi : \mathbb{R} \times \mathbb{R}^n \to \mathbb{R}^n$ is the solution to the state trajectories of (1) based on the corresponding initial value problem:

$$x(t_f) = \Phi(t_f, x(0)) = \Phi(t_f, u).$$

Since we usually cannot compute these output reachable sets exactly, we rely on computing an over-approximation denoted as $\Omega(\mathcal{X}_{in})$ such that $\mathcal{R}_{\text{neural ODE}}(\mathcal{X}_{in}) \subseteq \Omega(\mathcal{X}_{in})$.

2.1. Homeomorphism

A homeomorphism function is a continuous function that preserves topological characteristics, mapping only boundaries to boundaries and interiors to interiors (Massey, 1991). In Liang et al. (2023), a homeomorphism is defined as a relational map between two sets that preserves all topological properties between them, as illustrated in Figure 1. The homeomorphism property was introduced in some reachability analysis works involving a set-boundary reachability method for ordinary differential equations (Xue et al., 2016), and delay differential equations (Xue et al., 2020).

Definition 2 (Homeomorphism) *For two given sets $\mathcal{X}, \mathcal{Y} \subseteq \mathbb{R}^n$, there exists a map $m(.) : \mathcal{X} \to \mathcal{Y}$ which is a homeomorphism w.r.t. $\mathcal{X}$ if it is a continuous bijection and the map inverse $m^{-1}(.) : \mathcal{Y} \to \mathcal{X}$ is also continuous.*

Figure 1: Homeomorphism (Right) vs. non-homeomorphism (Left) sets

Lemma 3 (Massey (1991)) *Assuming that the two sets $\mathcal{X}, \mathcal{Y} \subseteq \mathbb{R}^n$ are closed and bounded, i.e., compact. For a homeomorphism map $m(.) : \mathcal{X} \to \mathcal{Y}$, m maps the boundaries of the set $\mathcal{X}$ to the boundaries of the set $\mathcal{Y}$, and the interior of the set $\mathcal{X}$ to the interior of the set $\mathcal{Y}$.*

Neural ODE are naturally invertible (Liang et al., 2023), which means that they are able to reconstruct their inputs from their outputs, i.e., they correspond to continuous reversible maps. Based on Lemma 3, and the fact that the two sets $\mathcal{X}, \mathcal{Y}$ are compact, the neural

ODE exhibit the homeomorphism property, which allows us to over-approximate the neural ODE reachable set from the boundaries of its initial set instead of over-approximating the neural ODE reachable set from the full initial set. Based on that, we propose a mixed monotonicity set boundary reachability method for neural ODE, and their corresponding computation procedure is presented in Section 3.4 and Algorithm 3.

2.2. Continuous time Mixed Monotonicity

Mixed monotonicity is a property of dynamical systems that allows for efficient computation of reachable sets by decomposing the system's vector field into components that are monotonically increasing and decreasing in their arguments (Coogan, 2020). Such decomposition allows the embedding of the original system into a higher-dimensional monotone system, where the bounds can be propagated forward to over-approximate the reachable sets (Meyer et al., 2021).

Definition 4 (Neural ODE Mixed Monotonicity) *The neural ODE* (1) *is mixed monotone if there exists a decomposition function* $g : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^n$ *such that* g *is increasing in its first argument,* g *is decreasing in its second argument based on Definition II.1 from* Angeli and Sontag (2003), *and* f *is embedded in the diagonal of* g: $g(x, x) = f(x)$.[1]

The mixed monotonicity property is satisfied on any system that is Lipschitz continuous. Lipschitz continuity ensures that the system's vector field has a bounded Jacobian matrix, which is the only condition required to construct such a decomposition function using the continuous-time mixed monotonicity methods from (Meyer et al., 2021).

In this paper, we leverage an existing tool based on mixed monotonicity and named TIRA (Meyer et al., 2019) for neural ODE. Notably, mixed monotonicity is an interval-based approach, relying on hyperrectangular (box) bounds for computational simplicity while ensuring soundness (Meyer et al., 2021), compared with other approaches focusing on other set representations, such as zonotopes used in CORA (Althoff, 2015), and star sets in NNV 2.0 (Lopez et al., 2023).

3. Boundary analysis Mixed Monotonicity Reachability for neural ODE

In this section, we present our approach for computing reachable set over-approximations for neural ODE using mixed monotonicity, enhanced with boundary analysis to leverage the homeomorphism property introduced in Section 2.1 (Lemma 3). We use Jacobian bounds based on the neural ODE reachable tube $\mathcal{R}^{\text{tube}}_{\text{neural ODE}}$. These bounds enable two continuous-time mixed monotonicity methods from (Meyer et al., 2021) which are continuous-time mixed monotonicity and sampled-data mixed monotonicity adapted to neural ODE using three different approaches for reachable set over-approximation: single-step, incremental, and boundary-based. These approaches, illustrated in Figure 2, are evaluated on the spiral and FPA systems.

---

1. The enthusiastic reader is encouraged to refer to Appendix C for more details.
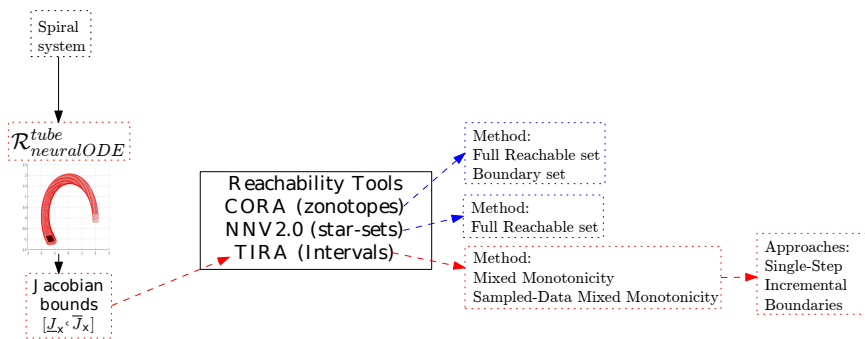
Figure 2: Illustration of the steps for reachability analysis of neural ODE using different tools, methods and mixed monotonicity approaches

3.1. Computing the Jacobian bounds

To apply mixed monotonicity, we first need to satisfy the only required condition which is to bound the Jacobian matrix of the neural ODE vector field $f(x)$, ensuring the existence of a sign-stable decomposition (Yang et al., 2019). Using CORA or NNV2.0 (Althoff, 2015; Lopez et al., 2023), we compute the neural ODE reachable tube $\mathcal{R}_{\text{neural ODE}}^{\text{tube}}$ over the time interval $[0, 1]$ for the spiral system and $[0, 2]$ for the FPA system, as a sequence of zonotopes, where each zonotope corresponds to an intermediate time range[2]. We then compute the union of all the reachable sets in $\mathcal{R}_{\text{neural ODE}}^{\text{tube}}$ to approximate the region containing all neural ODE trajectories over the time interval. From this union, we extract the minimum and maximum values along each dimension across all the reachable sets corresponding to the lower and upper bounds of the reachable tube. Then based on these bounds, we use interval arithmetic (Jaulin et al., 2001) as it is a simple and computationally efficient way to derive bounds on the Jacobian matrices, which will be used as the foundation for the over-approximation methods of continuous-time mixed monotonicity[3] (Meyer et al., 2021).

3.2. Single-Step Reachability analysis

In the single-step approach, we compute the reachable set over-approximation directly from the initial time to the final time t_f without intermediate subdivisions. For the neural ODE (1) with initial set $\mathcal{X}_{in}$, we embed the system into a mixed monotone form using the decomposition function $g(x, \hat{x})$ derived based on Jacobian bounds. The embedded system is solved over the full time horizon $[0, t_f]$, with $t_f = 1$ for the spiral system and $t_f = 2$ for the FPA system, propagating the interval bounds $[\underline{x}(0), \overline{x}(0)]$ from $\mathcal{X}_{in}$ to obtain the interval over-approximation $\Omega(\mathcal{X}_{in}) = [\underline{x}(t_f), \overline{x}(t_f)]$.

---

2. We performed the test while using a longer time range than the ones used in our examples and the resulting Jacobian matrices made the over-approximations not as tight as using only the Jacobian matrices based on the reachable tube of our examples time interval.

3. For further details about bounding the Jacobian matrices and its usage in mixed monotonicity, please refer to Appendices B and C

This approach is computationally efficient, as it involves a single integration of the embedded monotone system, providing a direct interval over-approximation of the reachable set at the final time. It is particularly well-suited for systems where achieving a balance between simplicity and soundness is more important than obtaining the tightest bounds.

3.3. Incremental Reachability analysis

The incremental approach refines the single-step method by dividing the time horizon into smaller steps, computing intermediate reachable set over-approximations to potentially reduce conservatism. For each step, we apply the mixed monotonicity embedding and propagate the bounds sequentially, using the output of one step as the input for the next one.

For the FPA system, we use a step size of 0.05, resulting in 40 incremental reachable set over-approximations, i.e., $\frac{t_f}{\text{step size}}$. For the spiral system, a finer step size of 0.01 is considered, resulting in 100 over-approximations. This method can yield tighter over-approximations than the single-step method, especially for systems with varying dynamics, but at the cost of increased computational time due to repeated numerical integrations of the embedded monotone system.

3.4. Boundary Reachability analysis

To take advantage of the homeomorphism property of neural ODE, we extend the single-step approach to compute the over-approximation of the reachable set from the boundaries of the initial set only. This can reduce the computation by focusing on the boundary of $\mathcal{X}_{in}$ rather than the entire set, as the interior reachable states are enclosed by the boundary evolution (Liang et al., 2023; Xue et al., 2020, 2016).

For the spiral 2-dimensional system, we compute the reachable sets from $2 \times n = 4$ boundaries (the faces of the dashed-zonotope in Figure 5 of Appendix D). For the FPA 5-dimensional system, this involves $2 \times 5 = 10$ boundaries (the intervals with the blue and red boundary points in Figure 11 of Appendix D). For each boundary, we apply the mixed monotonicity embedding and integrate the embedded system over the full time horizon $[0, t_f]$. We then take the interval hull of the union of the resulting interval over-approximations to form the final over-approximation $\Omega(\mathcal{X}_{in})$. This approach remains sound by enclosing all reachable sets within the boundary evolution, while leveraging the homeomorphism property of the neural ODE, ensuring that the invertible flow map preserves the topological structure of $\mathcal{X}_{in}$. This approach offers computational efficiency for both low and high dimensional systems, as it scales linearly with the state dimension.

4. Numerical illustration

In this section, two commonly used neural ODE academic examples (Chen et al., 2018; Musau and Johnson, 2018), described in detail in Appendices A.1 and A.2, are used to demonstrate our neural ODE mixed monotonicity reachability approaches. The spiral 2-dimensional example results at $t = 1$ are illustrated in Table 2 and Figure 3, while the FPA 5-dimensional example results at $t = 2$ are presented in Table 3 and Figure 4, using three subfigures to project the 5-dimensional results into two dimensions for visual comparison with TIRA's interval boxes.

Using the Jacobian bounds computed in Appendix B, we satisfy the only condition required to compute the over-approximation of the neural ODE mixed monotonicity based reachability approaches, as discussed in Section 3.

In the spiral example, as shown in Figure 3, the single-step and incremental mixed monotonicity approaches gives identical interval over-approximations, but the incremental approach requires approximately `96 times` more computational time than the single-step approach. Similarly, the single-step and dashed boundary-based sampled-data mixed monotonicity approaches yield equivalent over-approximations, where the boundary-based method is approximately `5 times` slower than single-step due to computing reachable sets across the four boundaries of the 2D initial set. Comparing across tools, CORA's zonotopes and NNV2.0 star set achieve tighter over-approximations than TIRA's approaches, with CORA's dashed-boundaries zonotope being the tightest. However, CORA's computational time is approximately `25 times` greater, and NNV2.0 is approximately `6 times` greater than TIRA's single-step mixed monotonicity approach, highlighting TIRA's efficiency for simpler interval-based representations.
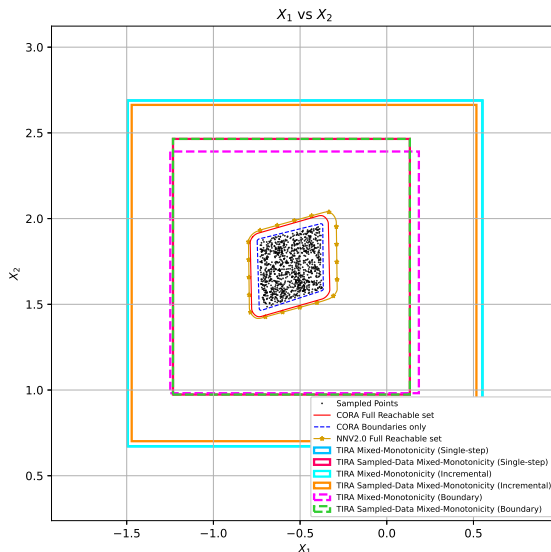

Figure 3: Spiral Comparison between TIRA vs. CORA vs. NNV2.0 at $t = 1$

In contrast, for the FPA example illustrated in Figure 4, the single-step, incremental and dashed-boundary-based mixed monotonicity yield identical over-approximation, but the computational time of the incremental approach is approximately `31 times` greater than that of the single-step approach, while the dashed-boundary approach is approximately `9 times` greater, making single-step the most efficient choice due to equivalent results. Additionally, the sampled-data mixed monotonicity approach produces less tight over-approximations than continuous-time mixed monotonicity at a significantly higher computational cost. Across tools, CORA's zonotopes consistently outperform NNV2.0 star sets in tightness with comparable computation times, except for CORA's dashed-boundaries zonotope which achieve the tightest over-approximations overall. However, TIRA's single-step

mixed monotonicity remains the fastest method, with CORA requiring approximately `131 times` more computation time than TIRA's single-step mixed monotonicity approach.

For both examples, we used three different set representations to illustrate their strengths and weaknesses. Zonotopes have a more flexible representation, allowing them to model diverse shapes and fit tightly to any set (Girard, 2005). Thus, for both examples, it is evident that CORA's zonotopes provide the tightest over-approximations compared with NNV2.0 star sets and TIRA's interval boxes, which are very simple rectangular box shapes, but very constrained as well[4]. In terms of complexity, the simple rectangular box shapes of intervals are easier to compute, resulting in shorter computational times for TIRA compared to CORA and NNV2.0 as illustrated in Table 1, and this is due to the fact that zonotopes require processing and storing more data, leading to longer computational times.
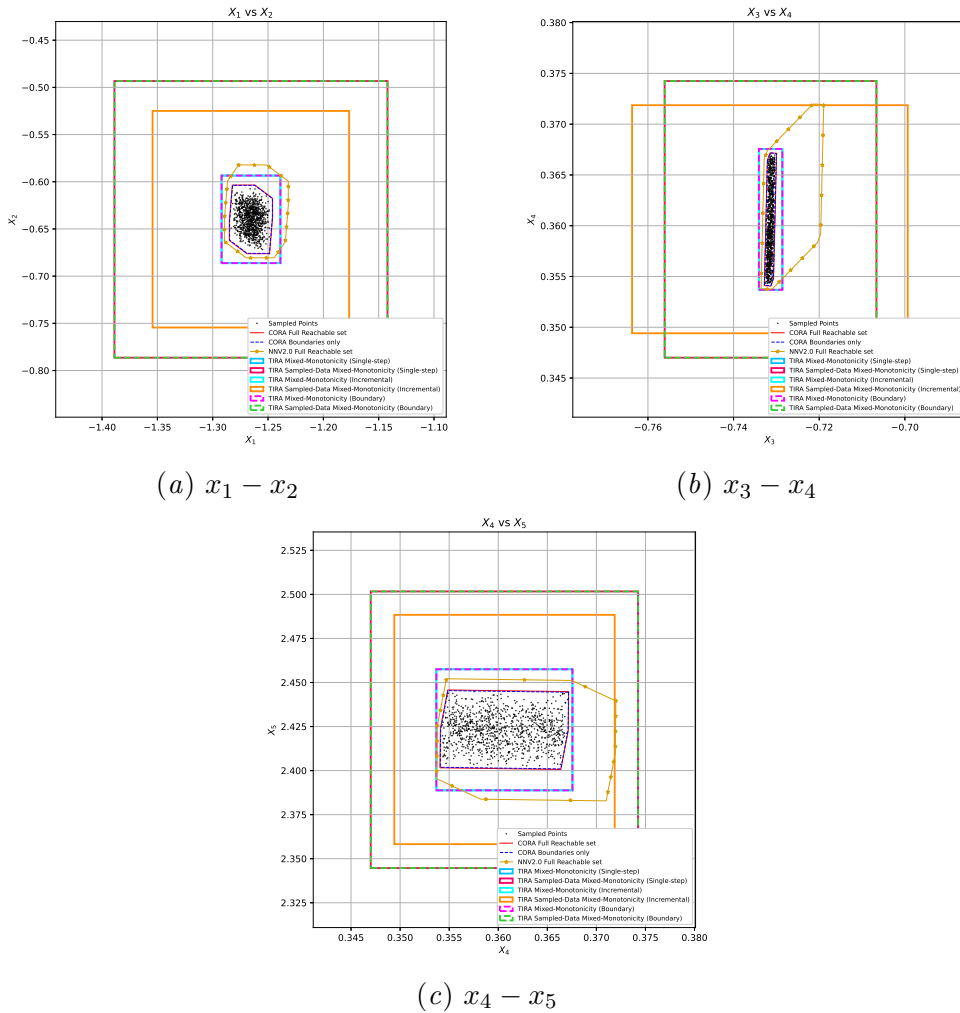

(a) $x_1 - x_2$

(b) $x_3 - x_4$

(c) $x_4 - x_5$

Figure 4: FPA Comparison between TIRA vs. CORA vs. NNV2.0 at $t = 2$

---

4. Interval boxes cannot be rotated or have their corners truncated.

Table 1: Spiral and FPA computation times (in seconds) with CORA, NNV2.0, and TIRA

Methods	Spiral @ $t = 1$	FPA @ $t = 2$
CORA Full Reachable Set	19.64	13.22
CORA Boundaries only	70.83	109.1
NNV2.0 Full Reachable Set	17.25	11.98
TIRA (single-step) Mixed-Monotonicity	0.66	0.83
TIRA (single-step) Sampled-Data Mixed-Monotonicity	0.95	1.34
TIRA (incremental) Mixed-Monotonicity	63.13	25.41
TIRA (incremental) Sampled-Data Mixed-Monotonicity	111.16	48.06
TIRA (Boundary) Mixed-Monotonicity	2.84	7.06
TIRA (Boundary) Sampled-Data Mixed-Monotonicity	4.35	12.76

5. Conclusions and Future Work

In this paper, we propose an interval-based reachability method for neural ODE, leveraging mixed monotonicity to compute over-approximations of the reachable sets. By adapting continuous-time mixed monotonicity techniques (Meyer et al., 2021), our approach efficiently computes interval-based over-approximations, both from the full initial input set and its boundaries solely, exploiting the homeomorphism property to reduce computational costs. This novel method, implemented in TIRA, provides a lightweight alternative for the reachability analysis of neural ODE, offering lower complexity and faster computation times compared with more flexible set representations such as zonotopes in CORA and star sets in NNV2.0.

Through numerical illustrations on the spiral and FPA examples, we demonstrate that our single-step, incremental, and boundary-based variants yield sound over-approximations, albeit at the cost of tightness. Ultimately, the choice between our interval-based methods and zonotope or star set based tools depends on the trade-off between over-approximation tightness and computational efficiency, making our approach particularly suitable for high-dimensional and real-time safety verification scenarios.

In future work, we plan to extend the boundary-based reachability approach by integrating the incremental reachability analysis method to compute over-approximations of the boundaries of the input set over finer incremental steps. We also aim to explore partitioning the initial input set into smaller subsets, performing reachability analysis on each subset using our interval-based method, and subsequently taking the union of all resulting over-approximations. In addition, we intend to incorporate this framework into a full verifier to check safety properties in neural ODE. Finally, evaluating our interval-based reachability method in real-world applications, such as neural ODE-based control systems, typically real-time obstacle detection, will validate the practical utility of the method and guide possible refinements for safety-critical domains.

Acknowledgments

References

Matthias Althoff. An introduction to cora 2015. In *Proc. of the workshop on applied verification for continuous and hybrid systems*, pages 120–151, 2015.

David Angeli and Eduardo D Sontag. Monotone control systems. *IEEE Transactions on automatic control*, 48(10):1684–1698, 2003.

Randall D Beer. On the dynamics of small continuous-time recurrent neural networks. *Adaptive Behavior*, 3(4):469–509, 1995.

Ricky TQ Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. *Advances in neural information processing systems*, 31, 2018.

Samuel Coogan. Mixed monotonicity for reachability and safety in dynamical systems. In *2020 59th IEEE Conference on Decision and Control (CDC)*, pages 5074–5085. IEEE, 2020.

Samuel Coogan and Murat Arcak. Efficient finite abstraction of mixed monotone systems. In *Proceedings of the 18th International Conference on Hybrid Systems: Computation and Control*, pages 58–67, 2015.

Samuel Coogan and Murat Arcak. Stability of traffic flow networks with a polytree topology. *Automatica*, 66:246–253, 2016.

Samuel Coogan, Murat Arcak, and Alexander A Kurzhanskiy. Mixed monotonicity of partial first-in-first-out traffic flow models. In *2016 IEEE 55th conference on decision and control (CDC)*, pages 7611–7616. IEEE, 2016.

Antoine Girard. Reachability of uncertain linear systems using zonotopes. In *International workshop on hybrid systems: Computation and control*, pages 291–305. Springer, 2005.

Jean-Luc Gouzé and Karl P Hadeler. Monotone flows and order intervals. *Nonlinear World*, 1(1):23–34, 1994.

Sophie A Gruenbacher, Mathias Lechner, Ramin Hasani, Daniela Rus, Thomas A Henzinger, Scott A Smolka, and Radu Grosu. Gotube: Scalable statistical verification of continuous-depth models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 6755–6764, 2022.

Sophie Grunbacher, Ramin Hasani, Mathias Lechner, Jacek Cyranka, Scott A Smolka, and Radu Grosu. On the verification of neural odes with stochastic guarantees. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 11525–11535, 2021.

Eldad Haber and Lars Ruthotto. Stable architectures for deep neural networks. *Inverse problems*, 34(1):014004, 2017.

Akash Harapanahalli, Saber Jafarpour, and Samuel Coogan. A toolbox for fast interval arithmetic in numpy with an application to formal verification of neural network controlled systems. *arXiv preprint arXiv:2306.15340*, 2023.

Akash Harapanahalli, Saber Jafarpour, and Samuel Coogan. immrax: A parallelizable and differentiable toolbox for interval analysis and mixed monotone reachability in jax. *IFAC-PapersOnLine*, 58(11):75–80, 2024.

Luc Jaulin, Michel Kieffer, Olivier Didrit, and Eric Walter. *Applied Interval Analysis: With Examples in Parameter and State Estimation, Robust Control and Robotics.* Springer, 2001.

P Kidger. *On neural differential equations.* PhD thesis, University of Oxford, 2021.

Zhen Liang, Dejin Ren, Wanwei Liu, Ji Wang, Wenjing Yang, and Bai Xue. Safety verification for neural networks based on set-boundary analysis. In *International Symposium on Theoretical Aspects of Software Engineering*, pages 248–267. Springer, 2023.

Diego Manzanas Lopez, Sung Woo Choi, Hoang-Dung Tran, and Taylor T Johnson. Nnv 2.0: The neural network verification tool. In *International Conference on Computer Aided Verification*, pages 397–412. Springer, 2023.

Diego Manzanas Lopez, Patrick Musau, Nathaniel P Hamilton, and Taylor T Johnson. Reachability analysis of a general class of neural ordinary differential equations. In *International Conference on Formal Modeling and Analysis of Timed Systems*, pages 258–277. Springer, 2022.

William S Massey. *A basic course in algebraic topology.* Springer Science & Business Media, 1991.

Pierre-Jean Meyer. Reachability analysis of neural networks using mixed monotonicity. *IEEE Control Systems Letters*, 6:3068–3073, 2022.

Pierre-Jean Meyer, Alex Devonport, and Murat Arcak. TIRA: Toolbox for interval reachability analysis. In 22nd *ACM International Conference on Hybrid Systems: Computation and Control*, pages 224–229, 2019.

Pierre-Jean Meyer, Alex Devonport, and Murat Arcak. *Interval reachability analysis: Bounding trajectories of uncertain systems with boxes for control and verification.* Springer Nature, 2021.

P Musau and TT Johnson. Continuous-time recurrent neural networks (ctrnns)(benchmark proposal). In *5th Applied Verification for Continuous and Hybrid Systems Workshop (ARCH), Oxford, UK*, 2018. URL https://doi.org/10.29007/6czp.

YongKyung Oh, Seungsu Kam, Jonghun Lee, Dong-Young Lim, Sungil Kim, and Alex Bui. Comprehensive review of neural differential equations for time series analysis, 2025. URL https://arxiv.org/abs/2502.09885.

Ramiro Rico-Martinez, K Krischer, IG Kevrekidis, MC Kube, and JL Hudson. Discrete-vs. continuous-time nonlinear signal processing of cu electrodissolution data. *Chemical Engineering Communications*, 118(1):25–48, 1992.

Abdelrahman Sayed Sayed, Pierre-Jean Meyer, and Mohamed Ghazel. Bridging neural ode and resnet: A formal error bound for safety verification. In *International Symposium on AI Verification*, pages 97–114. Springer, 2025.

Hoang-Dung Tran, Xiaodong Yang, Diego Manzanas Lopez, Patrick Musau, Luan Viet Nguyen, Weiming Xiang, Stanley Bak, and Taylor T Johnson. Nnv: the neural network verification tool for deep neural networks and learning-enabled cyber-physical systems. In *International conference on computer aided verification*, pages 3–17. Springer, 2020.

Bai Xue, Arvind Easwaran, Nam-Joon Cho, and Martin Fränzle. Reach-avoid verification for nonlinear systems based on boundary analysis. *IEEE Transactions on Automatic Control*, 62(7):3518–3523, 2016.

Bai Xue, Qiuye Wang, Shenghua Feng, and Naijun Zhan. Over-and underapproximating reach sets for perturbed delay differential equations. *IEEE Transactions on Automatic Control*, 66(1):283–290, 2020.

Liren Yang, Oscar Mickelin, and Necmiye Ozay. On sufficient conditions for mixed monotonicity. *IEEE Transactions on Automatic Control*, 64(12):5080–5085, 2019.

Appendix A. System description

Experiments Settings: All the experiments[5] herein are run on MATLAB 2024b with the Continuous Reachability Analyzer (CORA) version 2025.1.1, the Toolbox for Interval Reachability Analysis (TIRA) version 2, and the Neural Network Verification Software Tool (NNV2.0) on an Intel (R) Core (TM) i5-1145G7 CPU@2.60 GHz and 32 GB of RAM.

A.1. Spiral

The spiral system is a 2-dimensional nonlinear dynamical system modeled as a neural ODE (Chen et al., 2018), characterized by dynamics that produce spiral trajectories in the state space, making it a valuable benchmark for studying complex, non-convergent behaviors in continuous-time systems, we consider here the following 2-dimensional neural ODE with the following dynamics

$$\dot{x} = f(x) = W_2 \tanh(W_1 x + b_1) + b_2,$$

where $x \in \mathbb{R}^2$ is the state vector, $W_1 \in \mathbb{R}^{10 \times 2}$ is the weight matrix of the first layer, $b_1 \in \mathbb{R}^{10}$ is the bias vector of the first layer, $W_2 \in \mathbb{R}^{2 \times 10}$ is the weight matrix of the second layer, $b_2 \in \mathbb{R}^2$ is the bias vector of the second layer, and $\tanh(\cdot)$ is the hyperbolic tangent activation function applied element-wise to the vector $W_1 x + b_1 \in \mathbb{R}^{10}$. The exact values of the weight matrices and bias vectors are defined within the Matlab function *spiral_non.m*.

A.2. FPA

The FPA system is a 5-dimensional nonlinear dynamical system with dynamics that converge to a fixed point (an equilibrium state) under certain conditions (Beer, 1995), and the fixed-point aspect makes it a useful model for studying convergence and stability, which are important in safety-critical applications where the system must not diverge or enter unsafe states. As in the proposed benchmark in Musau and Johnson (2018), we consider here the following 5-dimensional neural ODE approximating the FPA dynamics:

$$\dot{x} = f(x) = \tau x + W \tanh(x),$$

where $x \in \mathbb{R}^5$ is the state vector, $\tau = -10^{-6}$ is a time constant for the neurons, $W \in \mathbb{R}^{5 \times 5}$ is a composite weight matrix defined as $W = \begin{pmatrix} 0_{2 \times 2} & A \\ 0_{3 \times 2} & BA \end{pmatrix}$
with $A = \begin{pmatrix} -1.20327 & -0.07202 & -0.93635 \\ 1.18810 & -1.50015 & 0.93519 \end{pmatrix}$, and $B = \begin{pmatrix} 1.21464 & -0.10502 \\ 0.12023 & 0.19387 \\ -1.36695 & 0.12201 \end{pmatrix}$.
Here, $\tanh(x)$ is the hyperbolic tangent activation function applied element-wise to the state vector x.

---

5. Code available in the following repository:
   https://github.com/ab-sayed/Mixed-Monotonicity-Reachability-Analysis-of-neural-ODE

Appendix B. Jacobian Bounds

Considering the FPA system in Appendix A.2, the Jacobian of $f(x)$ is:

$$J_x(x) = \frac{\partial f}{\partial x}(x) = \tau I_5 + W \cdot \mathrm{diag}(1 - \tanh^2(x_i)) = \tau I_5 + W \cdot \mathrm{diag}(\mathrm{sech}^2(x_i)),$$

where I_5 is a 5×5 identity matrix, and $\mathrm{diag}(1 - \tanh^2(x_i))$ is a diagonal matrix with entries $1 - \tanh^2(x_i)$ for $i = 1, \ldots, 5$, since the derivative of $\tanh(x_i)$ is $1 - \tanh^2(x_i)$. So, the goal is to find interval bounds $[\underline{J}_x, \overline{J}_x]$ such that $J_x(x) \in [\underline{J}_x, \overline{J}_x]$ for all $x \in [\underline{x}, \overline{x}]$.

It is worth noting here that interval arithmetic can be used to directly compute the range of $J_x(x)$ by evaluating the expression over the interval $[\underline{x}, \overline{x}]$. This involves bounding the nonlinear term $1 - \tanh^2(x_i)$ and combining it with the linear terms, as follows:

1. For each $i \in \{1, \ldots, 5\}$, compute the interval $[\underline{T_i}, \overline{T_i}]$ such that $1 - \tanh^2(x_i) \in [\underline{T_i}, \overline{T_i}]$ for all $x_i \in [\underline{x}_i, \overline{x}_i]$.

2. Create an interval diagonal matrix $[\underline{D}, \overline{D}] = \mathrm{diag}([\underline{T_1}, \overline{T_1}], \ldots, [\underline{T_5}, \overline{T_5}])$.

3. Compute the matrix product $W \cdot [\underline{D}, \overline{D}]$ using interval arithmetics:

$$(W \cdot [\underline{D}, \overline{D}])_{ij} = \sum_{k=1}^{n} W_{ik} \cdot [\underline{D}_{kj}, \overline{D}_{kj}] = W_{ij} \cdot [\underline{T}_j, \overline{T}_j] = \begin{cases} [W_{ij}\underline{T}_j, W_{ij}\overline{T}_j] \text{ if } W_{ij} \geq 0 \\ [W_{ij}\overline{T}_j, W_{ij}\underline{T}_j] \text{ otherwise} \end{cases}$$

4. Add the constant term τI_5 to get $[\underline{J}_x, \overline{J}_x] = \tau I_5 + W \cdot [\underline{D}, \overline{D}]$

$$J_x(x) = [\underline{J}_x, \overline{J}_x] = \begin{bmatrix} [J_{\mathrm{lb}}(1,1), J_{\mathrm{ub}}(1,1)] & [J_{\mathrm{lb}}(1,2), J_{\mathrm{ub}}(1,2)] & \cdots & [J_{\mathrm{lb}}(1,5), J_{\mathrm{ub}}(1,5)] \\ [J_{\mathrm{lb}}(2,1), J_{\mathrm{ub}}(2,1)] & [J_{\mathrm{lb}}(2,2), J_{\mathrm{ub}}(2,2)] & \cdots & [J_{\mathrm{lb}}(2,5), J_{\mathrm{ub}}(2,5)] \\ \vdots & \vdots & \ddots & \vdots \\ [J_{\mathrm{lb}}(5,1), J_{\mathrm{ub}}(5,1)] & [J_{\mathrm{lb}}(5,2), J_{\mathrm{ub}}(5,2)] & \cdots & [J_{\mathrm{lb}}(5,5), J_{\mathrm{ub}}(5,5)] \end{bmatrix}$$

Appendix C. Mixed Monotonicity

In this appendix, we adapt the continuous-time mixed monotonicity and sampled-data mixed monotonicity methods from Meyer et al. (2021), which were originally established for continuous-time dynamical systems, to neural ODE.

C.1. Continuous-Time Mixed Monotonicity

The autonomous neural ODE (1) is mixed monotone if there exists a decomposition function $g : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^n$ such that for all $x, \hat{x} \in \mathbb{R}^n$, the following conditions hold:

- g is increasing in its first argument (off-diagonally):

$$\forall i, j \in \{1, \ldots, n\}, \ j \neq i : \quad \frac{\partial g_i}{\partial x_j}(x, \hat{x}) \geq 0,$$

- g is decreasing in its second argument:

$$\forall i, j \in \{1, \ldots, n\}: \quad \frac{\partial g_i}{\partial \hat{x}_j}(x, \hat{x}) \leq 0,$$

- f is embedded in the diagonal of g:

$$g(x, x) = f(x),$$

This decomposition implies that the embedded dynamical system is evolving in $\mathbb{R}^{2n_x}$:

$$\begin{bmatrix} \dot{x} \\ \dot{\hat{x}} \end{bmatrix} = \begin{bmatrix} g(x, \hat{x}) \\ g(\hat{x}, x) \end{bmatrix} = h(x, \hat{x}),$$

is monotone with respect to the orthant $\mathbb{R}^{n_x}_+ \times \mathbb{R}^{n_x}_-$ in its state space.

Requirements and Limitations: For applicability, there must exist a matrix $L_x \in \mathbb{R}^{n_x \times n_x}$ such that $J(x) + L_x$ is sign-stable over the considered time and state ranges (where $J(x) = \frac{\partial f}{\partial x}(x)$ is the Jacobian matrix). This means that each off-diagonal element of $J(x) + L_x$ maintains a constant sign (positive or negative) for all x in the domain (e.g., the reachable tube estimate containing all trajectories over $[0, T]$).

To construct L_x while minimizing the conservatism of the over-approximation, Meyer et al. (2021) recommends shifting each off-diagonal Jacobian element based on its interval bounds $[\underline{J}_{x_{ij}}, \overline{J}_{x_{ij}}]$. Namely, for each i, j with $j \neq i$, the shifting value $y = L_{x_{ij}}$ is chosen to move the interval to the nearest sign-stable half-plane with minimal distance:

$$y = \begin{cases} \max(0, -\underline{J}_{x_{ij}}) & \text{if } |\underline{J}_{x_{ij}}| \leq |\overline{J}_{x_{ij}}|, \\ \min(0, -\overline{J}_{x_{ij}}) & \text{if } |\underline{J}_{x_{ij}}| > |\overline{J}_{x_{ij}}|. \end{cases}$$

which handles the cases where the interval is already sign-stable (no shift, $y = 0$) or crosses zero (shift by a smaller overhang).

The decomposition function g embeds the vector field such that its i-th component is:

$$g_i(x, \hat{x}) = f_i(\xi_i) + \sum_{j=1}^n |L_{x_{ij}}|(x_j - \hat{x}_j),$$

where $\xi_i \in \mathbb{R}^n$ is defined component wise as $\xi_{ij} = x_j$ if $L_{x_{ij}} \geq 0$, else $\xi_{ij} = \hat{x}_j$

The reachable set at time T from initial interval $[\underline{x_0}, \overline{x_0}]$ is over-approximated by simulating the embedded system only once from the initial condition $\begin{bmatrix} x(0) \\ \hat{x}(0) \end{bmatrix} = \begin{bmatrix} \underline{x_0} \\ \overline{x_0} \end{bmatrix}$: the n_x first output variables of the embedded system represent the lower bound of this over-approximation; and the n_x last output variables its upper bound. This method is computationally efficient for neural ODE, although conservatism may increase with larger T.

15

Algorithm 1: Continuous-Time Mixed Monotonicity Reachability for Neural ODE

Input: Neural ODE (1), initial set $[\underline{x}_0, \overline{x}_0] \subseteq \mathbb{R}^n$, time horizon $T > 0$, **optional:**
  reachable tube estimate $[\underline{x}, \overline{x}] \subseteq \mathbb{R}^n$ containing $\{x(t) \mid t \in [0, T], x(0) \in [\underline{x}_0, \overline{x}_0]\}$

Output: Interval bounds $[\underline{x}(T), \overline{x}(T)]$ for the reachable set at time T

begin

  if $[\underline{x}, \overline{x}]$ *is not provided* **then**

    Compute Lipschitz constant L_f of f
    Set $[\underline{x}, \overline{x}] \leftarrow [\underline{x}_0 - L_f T \cdot \mathbf{1}, \overline{x}_0 + L_f T \cdot \mathbf{1}]$

  end

  Jacobian bounds:
  Compute $[\underline{J}_x, \overline{J}_x] \subseteq \mathbb{R}^{n \times n}$ for the Jacobian matrix $J_x = \frac{\partial f}{\partial x}$ over $x \in [\underline{x}, \overline{x}]$

  Shifting matrix L_x:
  Set $L_x \leftarrow 0_{n \times n}$
  for $i \leftarrow 1$ **to** n **do**

    for $j \leftarrow 1$ **to** n, $j \neq i$ **do**

      Set $L_{x_{ij}} \leftarrow y$, where:

$$y = \begin{cases} \max(0, -\underline{J}_{x_{ij}}) & \text{if } |\underline{J}_{x_{ij}}| \leq |\overline{J}_{x_{ij}}|, \\ \min(0, -\overline{J}_{x_{ij}}) & \text{if } |\underline{J}_{x_{ij}}| > |\overline{J}_{x_{ij}}|. \end{cases}$$

    end

  end

  Decomposition function $g(x, \hat{x})$:
  for $i \leftarrow 1$ **to** n **do**

    Set $\xi_i \leftarrow 0_n$
    for $j \leftarrow 1$ **to** n **do**

      if $L_{x_{ij}} \geq 0$ **then**

        Set $\xi_i^j \leftarrow x_j$

      end
      else

        Set $\xi_i^j \leftarrow \hat{x}_j$

      end

    end
    Set $g_i(x, \hat{x}) \leftarrow f_i(\xi_i) + \sum_{j=1}^n |L_{x_{ij}}|(x_j - \hat{x}_j)$

  end

  Auxiliary system:
  Simulate the system $\begin{bmatrix} \dot{x} \\ \dot{\hat{x}} \end{bmatrix} = \begin{bmatrix} g(x, \hat{x}) \\ g(\hat{x}, x) \end{bmatrix}$ from $t = 0$ to T with initial conditions $\begin{bmatrix} x_0 \\ \overline{x}_0 \end{bmatrix}$
   to obtain an output corresponding to $\begin{bmatrix} \underline{x}(T) \\ \overline{x}(T) \end{bmatrix}$;
  return *Interval bounds* $[\underline{x}(T), \overline{x}(T)]$

end

C.2. Sampled-Data Mixed Monotonicity

The main idea in this method is to model the continuous-time system as a discrete-time system by sampling at specific time points, with initial conditions $x(t_0) = x(0)$, the sampled-data version is defined as:

$$x^+ = \Phi(t_f; t_0, x(0)),$$

where $\Phi(t_f; t_0, x(0))$ is the solution of the ODE at time t_f starting from $x(0)$ at t_0. This allows us to apply discrete-time reachability methods to the continuous-time system by treating the evolution from t_0 to t_f as a single discrete step. Thus, the sampled-data approach can be directly applied by considering the solution map $\Phi(t_f; t_0, x(0))$, which for neural ODE (1) is computed by integrating the ODE defined by the neural network, as it is an autonomous system.

This method relies on the sensitivity matrix:

$$S_x(t_f; t_0, x(0)) = \frac{\partial \Phi(t_f; t_0, x(0))}{\partial x(0)},$$

which describes how small changes in the initial state $x(0)$ affect the state at time t_f. Based on Assumption 5.1 from Meyer et al. (2021), there exists a matrix L_x such that for all $i, j \in \{1, \ldots, n\}$, either:

$$S_{x_{ij}}(t_f; t_0, x_0) + L_{x_{ij}} \geq 0 \quad \forall x_0 \in [\underline{x_0}, \overline{x_0}],$$

or

$$S_{x_{ij}}(t_f; t_0, x_0) + L_{x_{ij}} \leq 0 \quad \forall x_0 \in [\underline{x_0}, \overline{x_0}],$$

which enforces a sign-stability condition similar to that used in discrete-time mixed monotonicity.

To construct L_x while minimizing the conservatism of the over-approximation, Meyer et al. (2021) recommends shifting each sensitivity element based on its interval bounds $[\underline{S}_{x_{ij}}, \overline{S}_{x_{ij}}]$. For each i, j, the shifting value $y = L_{x_{ij}}$ is chosen to move the interval to the nearest sign-stable half-plane with minimal distance:

$$y = \begin{cases} \max(0, -\underline{S}_{x_{ij}}) & \text{if } |\underline{S}_{x_{ij}}| \leq |\overline{S}_{x_{ij}}|, \\ \min(0, -\overline{S}_{x_{ij}}) & \text{if } |\underline{S}_{x_{ij}}| > |\overline{S}_{x_{ij}}|. \end{cases}$$

which handles the cases where the interval is already sign-stable or crosses zero.

The decomposition function g embeds the flow map such that its i-th component is:

$$g_i(t_0, x, \hat{x}) = \Phi_i(t_f; t_0, \xi_i) + \sum_{j=1}^{n} |L_{x_{ij}}|(x_j - \hat{x}_j).$$

Finally, the reachable set at time t_f is over-approximated by:

$$\left[g(t_0, \underline{x}, \overline{x}), g(t_0, \overline{x}, \underline{x}) \right],$$

similar to the discrete-time mixed monotonicity method, where only two evaluations of the decomposition function are needed to compute the bounds. This approach is particularly advantageous for larger time horizons, as it avoids the accumulating conservatism of continuous-time methods.

Algorithm 2: Sampled-Data Mixed Monotonicity Reachability for Neural ODE

Input: Neural ODE (1), initial set $[\underline{x}_0, \overline{x}_0] \subseteq \mathbb{R}^n$, time horizon $T > 0$, **optional:** reachable tube estimate $[\underline{x}, \overline{x}] \subseteq \mathbb{R}^n$ containing $\{x(t) \mid t \in [0, T], x(0) \in [\underline{x}_0, \overline{x}_0]\}$

Output: Interval bounds $[\underline{x}(T), \overline{x}(T)]$ for the reachable set over-approximation at time T

begin

   if $[\underline{x}, \overline{x}]$ *is not provided* **then**

      Compute Lipschitz constant L_f of f

      Set $[\underline{x}, \overline{x}] \leftarrow [\underline{x}_0 - L_f T \cdot \mathbf{1}, \overline{x}_0 + L_f T \cdot \mathbf{1}]$

   end

   Jacobian bounds:

   Compute $[\underline{J}_x, \overline{J}_x] \subseteq \mathbb{R}^{n \times n}$ for the Jacobian matrix $J_x = \frac{\partial f}{\partial x}$ over $x \in [\underline{x}, \overline{x}]$

   Sensitivity bounds:

   Compute $[\underline{S}_x, \overline{S}_x] \subseteq \mathbb{R}^{n \times n}$ for the sensitivity matrix $S_x(T; 0, x_0) = \frac{\partial \Phi(T; 0, x_0)}{\partial x_0}$ over $x(0) \in [\underline{x}_0, \overline{x}_0]$

   Shifting matrix L_x:

   Set $L_x \leftarrow 0_{n \times n}$

   for $i \leftarrow 1$ **to** n **do**

      for $j \leftarrow 1$ **to** n, $j \neq i$ **do**

         Set $L_{x_{ij}} \leftarrow y$, where:

$$y = \begin{cases} \max(0, -\underline{S}_{x_{ij}}) & \text{if } |\underline{S}_{x_{ij}}| \leq |\overline{S}_{x_{ij}}|, \\ \min(0, -\overline{S}_{x_{ij}}) & \text{if } |\underline{S}_{x_{ij}}| > |\overline{S}_{x_{ij}}|. \end{cases}$$

      end

   end

   Decomposition function $g(x, \hat{x})$:

   for $i \leftarrow 1$ **to** n **do**

      Set $\xi_i \leftarrow 0_n$

      for $j \leftarrow 1$ **to** n **do**

         if $L_{x_{ij}} \geq 0$ **then**

            Set $\xi_i^j \leftarrow x_j$

         end

         else

            Set $\xi_i^j \leftarrow \hat{x}_j$

         end

      end

      Solve ODE $\dot{x} = f(x)$ from $t = 0$ to T with $x(0) = \xi_i$ to obtain $\Phi(T; 0, \xi_i)$

      Set $g_i(x, \hat{x}) \leftarrow \Phi_i(T; 0, \xi_i) + \sum_{j=1}^{n} |L_{x_{ij}}|(x_j - \hat{x}_j)$

   end

   Reachability bounds:

   Compute $g(\underline{x}_0, \overline{x}_0)$ and $g(\overline{x}_0, \underline{x}_0)$

   Set $\underline{x}(T) \leftarrow g(\underline{x}_0, \overline{x}_0)$ and $\overline{x}(T) \leftarrow g(\overline{x}_0, \underline{x}_0)$

   return *Interval bounds* $[\underline{x}(T), \overline{x}(T)]$

end

C.3. Boundary-Based Mixed Monotonicity Reachability Analysis for neural ODE

The boundary-based mixed monotonicity neural ODE reachability approach, outlined in Algorithm 3, computes the over-approximation of the reachable set $\Omega(\mathcal{X}_{in})$ of the neural ODE (1) from the boundaries of the initial input set $\mathcal{X}_{in}$ instead of the entire input set.

The algorithm begins by extracting the boundaries $\mathcal{B}$ of $\mathcal{X}_{in}$ (typically the $2 \times n$ facets of the hyperplanes as illustrated in Figures 5 and 11), and the output over-approximation set $\Omega(\mathcal{X}_{in})$ is initialized as empty. For each of the extracted boundaries, the mixed monotonicity embedding is applied based on the mixed monotonicity method chosen from the two methods discussed in Sections C.1 and C.2.

Finally, the over-approximations $\Omega(\mathcal{B})$ from all boundaries are combined via a union, and we then take the interval hull of this union to ensure that it also contains the interior of the reachable set.

Algorithm 3: Boundary-Based Mixed Monotonicity Reachability for Neural ODE

Input: neural ODE (1), initial set $[\underline{x}_0, \overline{x}_0] \subseteq \mathbb{R}^n$, time horizon $T > 0$, single-step mixed monotonicity

Output: Over-approximation $\Omega(\mathcal{X}_{in})$ of the reachable set at time T

begin

    Set $\mathcal{X}_{in} \leftarrow [\underline{x}_0, \overline{x}_0]$

    Extract the boundaries $\mathcal{B}$ of $\mathcal{X}_{in}$

    Initialize $\Omega(\mathcal{X}_{in}) \leftarrow \emptyset$

    for *each boundary $\mathcal{B}$ of $\mathcal{X}_{in}$* **do**

        Apply mixed monotonicity embedding to compute decomposition function $g(x, \hat{x})$ based on Jacobian bounds

        Simulate the embedded system $\begin{bmatrix} \dot{x} \\ \dot{\hat{x}} \end{bmatrix} = \begin{bmatrix} g(x, \hat{x}) \\ g(\hat{x}, x) \end{bmatrix}$ over $[0, T]$ with initial bounds from $\mathcal{B}$

        Obtain over-approximation $\Omega(\mathcal{B})$

        Set $\Omega(\mathcal{X}_{in}) \leftarrow \Omega(\mathcal{X}_{in}) \cup \Omega(\mathcal{B})$

    end

    Set $\Omega(\mathcal{X}_{in})$ as the interval hull of itself

    return *Interval over-approximation $\Omega(\mathcal{X}_{in})$*

end

Appendix D. Numerical Results

This appendix provides separate numerical results for each of the Spiral and FPA examples for CORA, NNV2.0, and TIRA toolboxes. Tables 2 and 3 contain the tightness metrics that represent the ratio of the area of the computed reachable set over-approximations to the area spanned by the sampled successors in each 2D projection[6], as well as the computation times. The tightness metrics ratio is used to evaluate the conservatism of the over-approximation, where higher values indicate a more conservative (larger) over-approximation.

D.1. Spiral

D.1.1. CORA
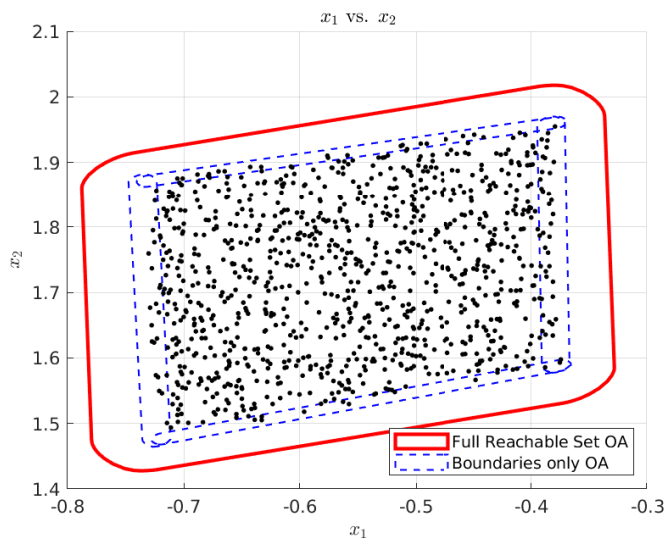
Figure 5: CORA Full Reachable Set OA vs. Boundaries only OA for Spiral at $t = 1$

D.1.2. NNV2.0

D.1.3. TIRA

---

6. The FPA example results are appearing over three subfigures, each representing the projection of two dimensions only as the FPA is a 5-dimensional example.
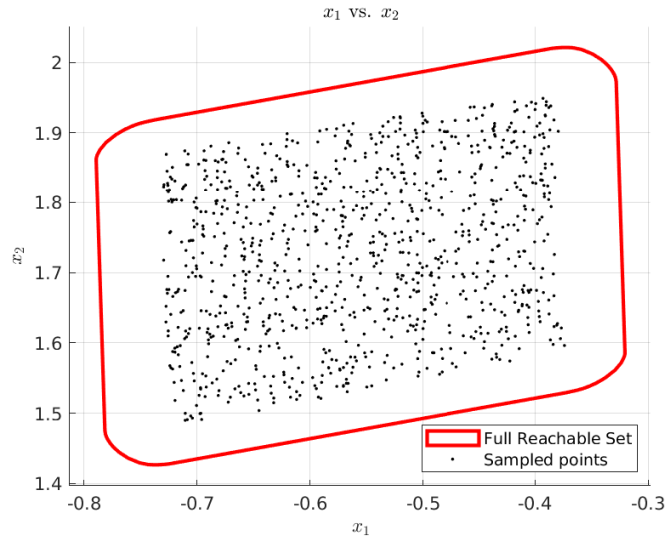
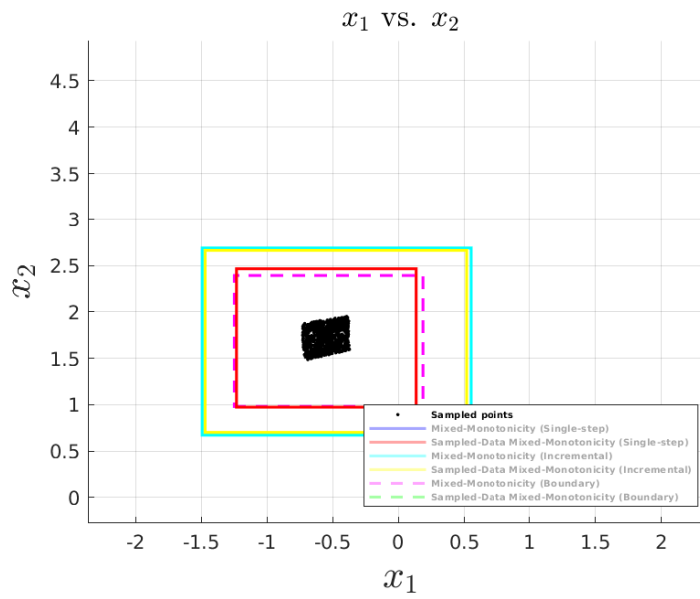Figure 6: NNV2.0 Full Reachable Set OA for Spiral at $t = 1$

Figure 7: TIRA single-step vs. incremental vs. boundary for Spiral at $t = 1$

21

Table 2: Spiral Numerical results at $t = 1$ with CORA, NNV2.0, and TIRA

| Methods	$x_1 - x_2$	Time(sec.)
CORA Full Reachable Set	1.61	19.64
CORA Boundaries only	1.15	70.83
NNV2.0 Full Reachable Set	1.71	17.25
TIRA (single-step) Mixed-Monotonicity	24.59	0.66
TIRA (single-step) Sampled-Data Mixed-Monotonicity	12.14	0.95
TIRA (incremental) Mixed-Monotonicity	24.59	63.13
TIRA (incremental) Sampled-Data Mixed-Monotonicity	23.24	111.16
TIRA (Boundary) Mixed-Monotonicity	12.05	2.84
TIRA (Boundary) Sampled-Data Mixed-Monotonicity	12.14	4.35

D.2. FPA

D.2.1. CORA

(a) $x_1 - x_2$

(b) $x_3 - x_4$

(c) $x_4 - x_5$
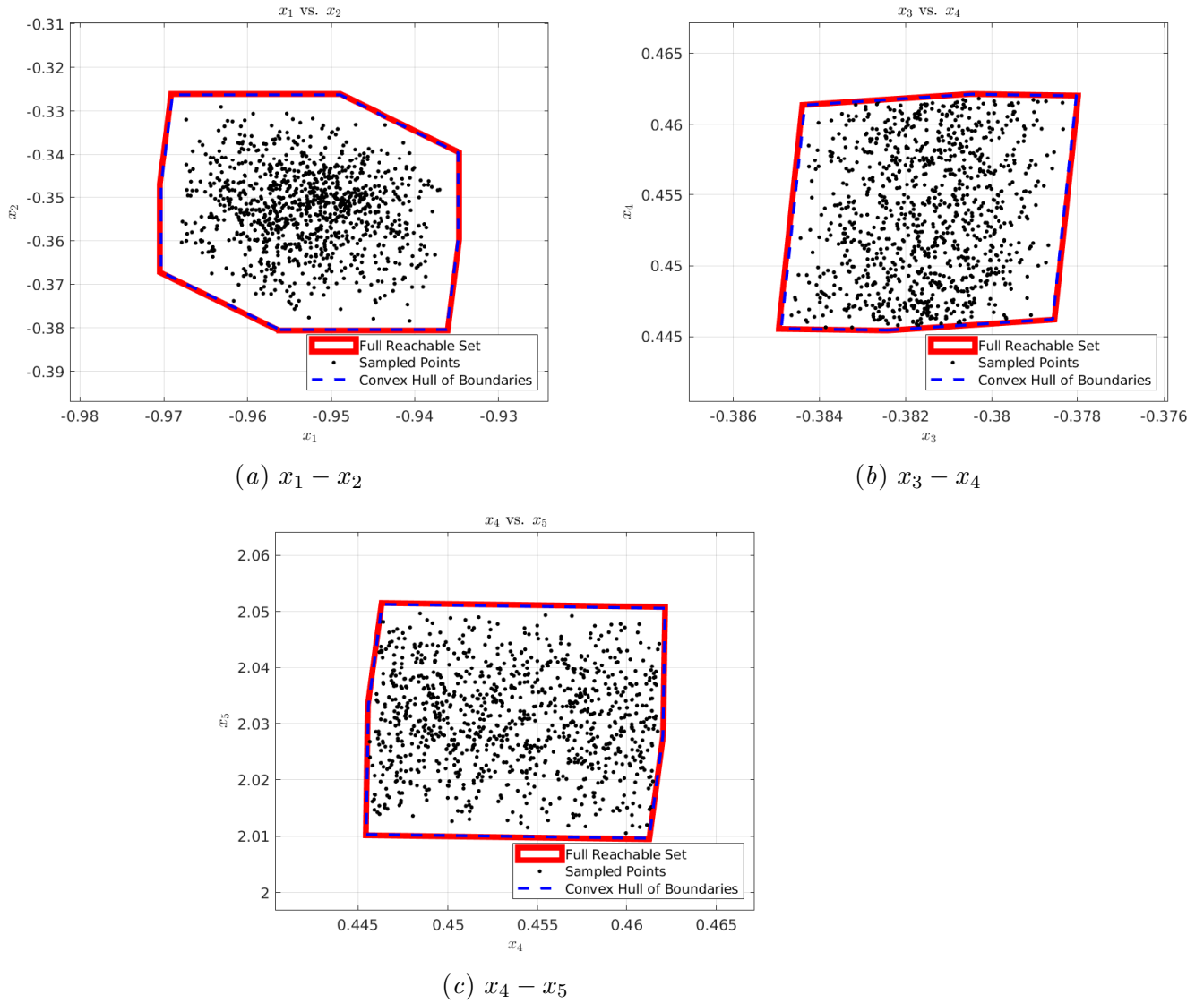
Figure 8: CORA Full Reachable Set OA vs. Boundaries only OA for FPA at $t = 2$

D.2.2. NNV2.0

(a) $x_1 - x_2$

(b) $x_3 - x_4$

(c) $x_4 - x_5$
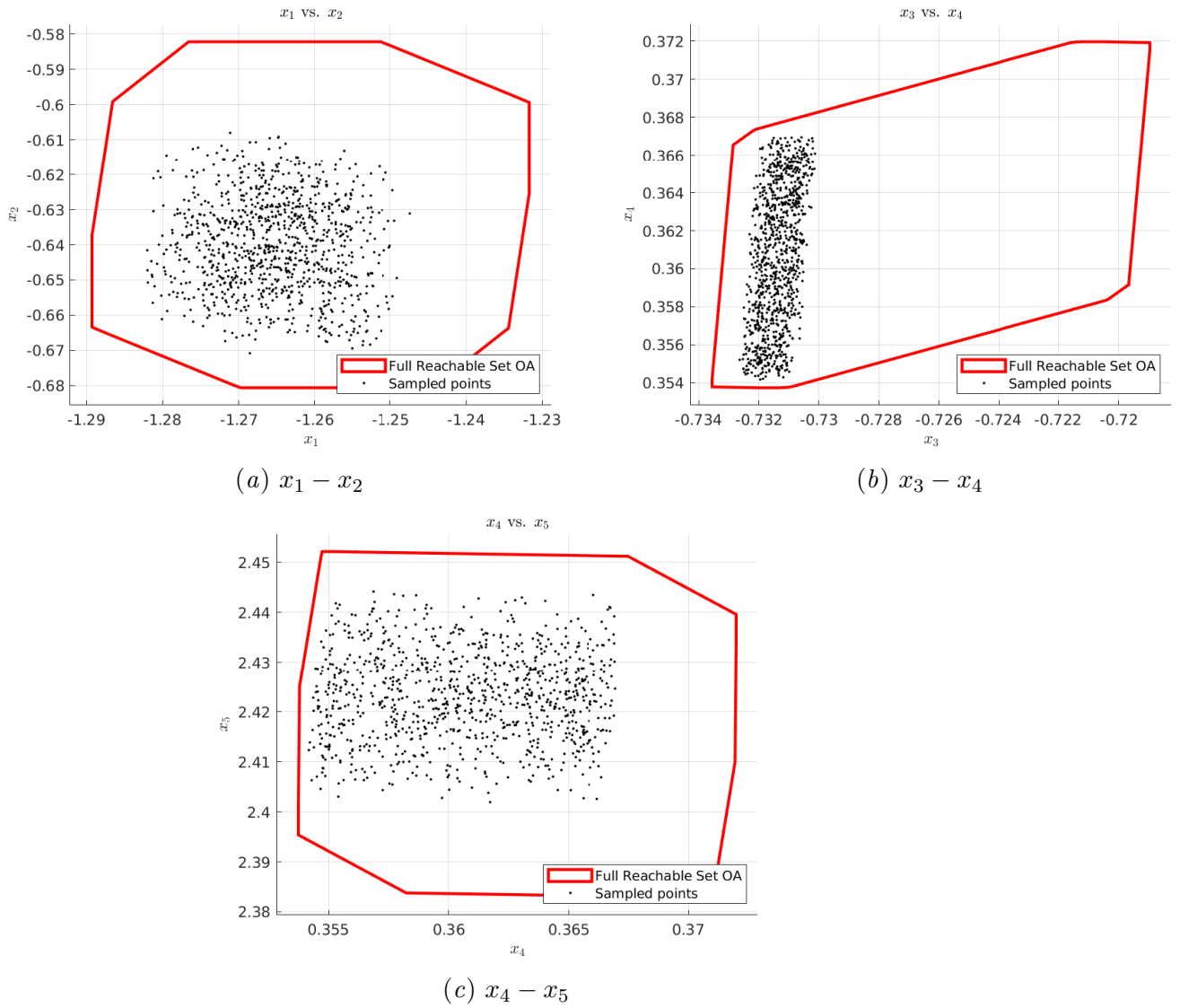
Figure 9: NNV2.0 Full Reachable Set OA at $t = 2$

D.2.3. TIRA

(a) $x_1 - x_2$

(b) $x_3 - x_4$

(c) $x_4 - x_5$
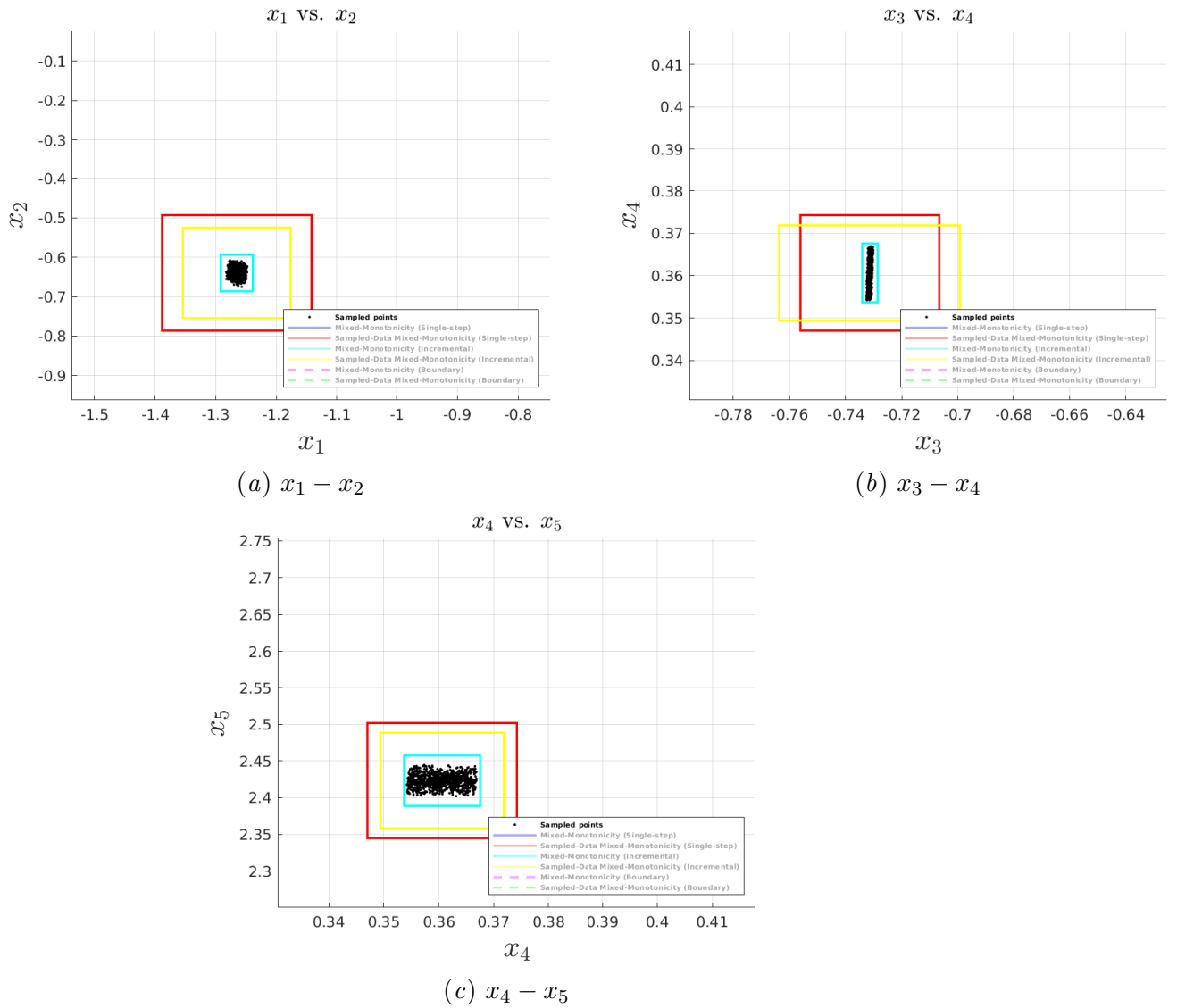
Figure 10: TIRA single-step vs. incremental vs. boundary for FPA at $t = 2$

(a) $x_1 - x_2$

(b) $x_3 - x_4$

(c) $x_4 - x_5$

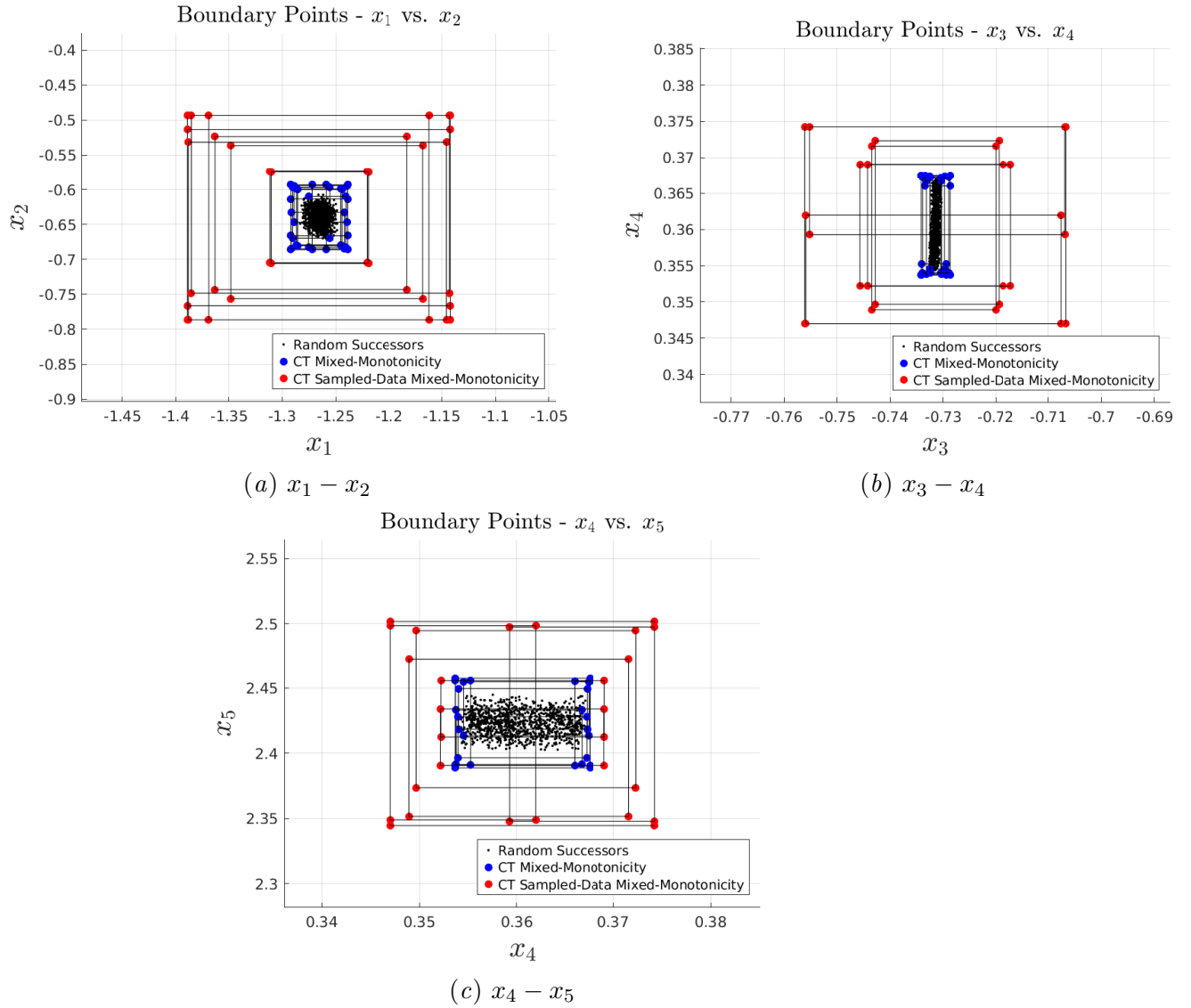Figure 11: FPA 10 Boundaries

Table 3: FPA Numerical results at $t = 2$ with CORA, NNV2.0, and TIRA

Methods	$x_1 - x_2$	$x_3 - x_4$	$x_4 - x_5$	Time(sec.)
CORA Full Reachable Set	1.33	1.11	1.13	13.22
CORA Boundaries only	1.18	0.99	1.08	109.1
NNV2.0 Full Reachable Set	2.52	8.74	2.43	11.98
TIRA (single-step) Mixed-Monotonicity	2.29	2.30	1.79	0.83
TIRA (single-step) Sampled-Data Mixed-Monotonicity	33.57	40.67	8.05	1.34
TIRA (incremental) Mixed-Monotonicity	2.29	2.30	1.79	25.41
TIRA (incremental) Sampled-Data Mixed-Monotonicity	18.92	43.64	5.50	48.06
TIRA (Boundary) Mixed-Monotonicity	2.29	2.30	1.79	7.06
TIRA (Boundary) Sampled-Data Mixed-Monotonicity	33.57	40.67	8.05	12.76