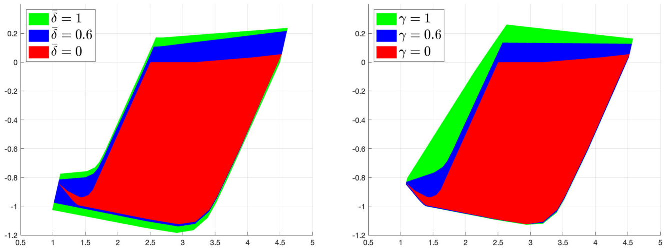
Reachability Analysis of Neural Network Control Systems With Tunable Accuracy and Efficiency

Yuhao Zhang, Hang Zhang, and Xiangru Xu, *Member, IEEE*

***Abstract*—The surging popularity of neural networks in controlled systems underscores the imperative for formal verification to ensure the reliability and safety of such systems. Existing set propagation-based approaches for reachability analysis in neural network control systems encounter challenges in scalability and flexibility. This letter introduces a novel tunable hybrid zonotope-based method for computing both forward and backward reachable sets of neural network control systems. The proposed method incorporates an optimization-based network reduction technique and an activation pattern-based hybrid zonotope propagation approach for ReLU-activated feedforward neural networks. Furthermore, it enables two tunable parameters to balance computational complexity and approximation accuracy. A numerical example is provided to illustrate the performance and tunability of the proposed approach.**

***Index Terms*—Reachable set, neural network control systems, scalability, tunability, hybrid zonotope.**



(a) $\gamma = 0, \bar{\delta} \in \{0, 0.6, 1\}$.

(b) $\bar{\delta} = 0, \gamma \in \{0, 0.6, 1\}$.

Fig. 1. An illustration of the "tunability" of the proposed method. The approximation error of reachable set computation varies with values of the tunable tolerance parameter, $\bar{\delta}$, and the tunable relaxation parameter, γ. Exact reachable sets (red) are computed when $\bar{\delta} = \gamma = 0$.

I. Introduction

NEURAL Networks (NNs) have gained widespread use in autonomous systems. However, the application of NNs in safety-critical scenarios necessitates formal verification as NNs exhibit high sensitivity to minor perturbations in the input space. To address this issue, several recent advancements have focused on reachability-based methods, primarily owing to their computational efficiency in the safety verification of Neural Network Control Systems (NNCS). By abstracting the non-linear activation functions of NNs using different set representations, the Forward Reachable Sets (FRSs) and Backward Reachable Sets (BRSs) of NNCS can be computed through set-propagation techniques to validate the safety specifications [1], [2], [3], [4], [5]. Despite these interesting results, many problems related to scalability and approximation accuracy require further exploration [6], [7].

Recently, an approach based on *Hybrid Zonotope* (HZ) was proposed to compute the *exact* FRS and BRS of NNCS with linear plant and ReLU-activated Feedforward Neural Network (FNN) controllers [8], [9], [10]. With the capability of representing non-convex sets with flat faces [11], [12], HZs enable exact abstractions of ReLU-activated FNNs through simple matrix operations. Nevertheless, the approaches mentioned above also face scalability challenges when dealing with feedback systems incorporating large NNs. This is attributed to the increasing representation complexity of the HZ reachable sets, which escalates with the number of neurons in the NNs. Heuristic complexity reduction techniques for general HZs exist [9], [11], but they don't leverage the inherent properties of NNs. Several recent works proposed output-based NN reduction algorithms by grouping neurons with similar ranges over a given input domain [13], [14]; however, these methods require predefined reduction metrics and only consider NNs in isolation.

This letter presents a novel HZ-based approach with the flexibility of balancing computational complexity and approximation accuracy. Contributions of this letter are at least twofold: i) A tunable optimization-based method is proposed for reducing the number of neurons of a given FNN while maintaining its intrinsic input-output mapping properties, with the optimal reduction metrics determined on the fly. ii) Based on the FNN reduction results, an activation pattern-based approach is presented for computing the graph set of FNNs and reachable sets of NNCS in the form of HZs. The constructed HZ representations are proved to over-approximate the exact graph and reachable sets. With the flexibility of tunable parameters, the proposed approach allows a trade-off between the set representation complexity and the approximation accuracy (see Fig. 1). The proposed approach can also restore exact

Manuscript received 8 March 2024; revised 17 May 2024; accepted 7 June 2024. Date of publication 17 June 2024; date of current version 5 July 2024. This work was supported in part by NSF under Grant CNS-2222541 and Grant CMMI-2237850. Recommended by Senior Editor A. P. Aguiar. *(Corresponding author: Xiangru Xu.)*

The authors are with the Department of Mechanical Engineering, University of Wisconsin-Madison, Madison, WI 53706 USA (e-mail: yuhao.zhang2@wisc.edu; hang.zhang@wisc.edu; xiangru.xu@wisc.edu).

Digital Object Identifier 10.1109/LCSYS.2024.3415471

reachability analysis as a special case and enable sound and complete verification for NNCS. The performance and tunability of the proposed method are demonstrated through a numerical example.

Notation: The i-th component of a vector $\boldsymbol{x}\in\mathbb{R}^n$ is denoted by x_i with $i\in[n]\triangleq\{1,\dots,n\}$. The i-th row (resp. j-th column) of a matrix $\mathbf{A}\in\mathbb{R}^{n\times m}$ is denoted by $\mathbf{A}[i,:]$ (resp. $\mathbf{A}[:,j]$). For a set $\mathcal{C}\subset[n]$ (resp. $\mathcal{C}'\subset[m]$), $\mathbf{A}[\mathcal{C},:]$ (resp. $\mathbf{A}[:,\mathcal{C}']$) denotes a submatrix of $\mathbf{A}$ with all rows $i\in\mathcal{C}$ (resp. all columns $j\in\mathcal{C}'$). The i-th column of an identity matrix $\boldsymbol{I}$ is denoted as $\boldsymbol{e}_i$. Given sets $\mathcal{X}\subset\mathbb{R}^n$, $\mathcal{Z}\subset\mathbb{R}^m$ and a matrix $\boldsymbol{R}\in\mathbb{R}^{m\times n}$, the generalized intersection of $\mathcal{X}$ and $\mathcal{Z}$ under $\boldsymbol{R}$ is $\mathcal{X}\cap_{\boldsymbol{R}}\mathcal{Z}=\{\boldsymbol{x}\in\mathcal{X}\mid\boldsymbol{R}\boldsymbol{x}\in\mathcal{Z}\}$. An interval with bounds $\underline{\boldsymbol{a}}$, $\bar{\boldsymbol{a}}\in\mathbb{R}^n$ is denoted as $[[\underline{\boldsymbol{a}},\bar{\boldsymbol{a}}]]$. The interval hull of a set $\mathcal{X}\subset\mathbb{R}^n$ is denoted as interval$(\mathcal{X})\subset\mathbb{R}^n$. The projection of a set $\mathcal{X}\subset\mathbb{R}^n$ onto a set of coordinates $\Phi=\{i_1,\dots,i_k\}\subset[n]$ is denoted as $\text{proj}_\Phi(\mathcal{X})\triangleq\{[\boldsymbol{e}_{i_1}\cdots\boldsymbol{e}_{i_k}]^\top\boldsymbol{x}\mid\boldsymbol{x}\in\mathcal{X}\}\subset\mathbb{R}^k$.

II. Preliminaries & Problem Statement

We first give the definition of hybrid zonotope.

Definition 1 [11]: The set $\mathcal{Z}\subset\mathbb{R}^n$ is a hybrid zonotope if there exist $\mathbf{c}\in\mathbb{R}^n$, $\mathbf{G}^c\in\mathbb{R}^{n\times n_g}$, $\mathbf{G}^b\in\mathbb{R}^{n\times n_b}$, $\mathbf{A}^c\in\mathbb{R}^{n_c\times n_g}$, $\mathbf{A}^b\in\mathbb{R}^{n_c\times n_b}$, $\mathbf{b}\in\mathbb{R}^{n_c}$ such that $\mathcal{Z}=\{\mathbf{G}^c\boldsymbol{\xi}^c+\mathbf{G}^b\boldsymbol{\xi}^b+\mathbf{c}\mid\boldsymbol{\xi}^c\in\mathcal{B}_\infty^{n_g},\boldsymbol{\xi}^b\in\{-1,1\}^{n_b},\mathbf{A}^c\boldsymbol{\xi}^c+\mathbf{A}^b\boldsymbol{\xi}^b=\mathbf{b}\}$ where $\mathcal{B}_\infty^{n_g}=\{\boldsymbol{x}\in\mathbb{R}^{n_g}\mid\|\boldsymbol{x}\|_\infty\le 1\}$ is the unit hypercube in $\mathbb{R}^{n_g}$. The *HCG*-representation of the HZ is given by $\mathcal{Z}=\langle\mathbf{G}^c,\mathbf{G}^b,\mathbf{c},\mathbf{A}^c,\mathbf{A}^b,\mathbf{b}\rangle$, where $\mathbf{c}$ is called the center, the columns of $\mathbf{G}^b$ are called the *binary generators*, and the columns of $\mathbf{G}^c$ are called the *continuous generators*.

The representation complexity of $\mathcal{Z}$ is determined by n_g, n_b, and n_c. HZs are closed under commonly used set operations such as linear map, intersection, and union. For an HZ $\mathcal{Z}\subset\mathbb{R}^n$, interval$(\mathcal{Z})$ can be obtained by solving $2n$ Mixed Integer Linear Programs (MILPs) [11], [12].

Next we define notations related to FNNs. Let $\boldsymbol{\pi}:\mathbb{R}^n\to\mathbb{R}^m$ be an ℓ-layer FNN with weight matrices $\{\boldsymbol{W}^{(k-1)}\}_{k\in[\ell]}$ and bias vectors $\{\boldsymbol{v}^{(k-1)}\}_{k\in[\ell]}$. Denote $\boldsymbol{x}^{(k)}\in\mathbb{R}^{n_k}$ as the neurons of the k-th layer. Then, $\boldsymbol{x}^{(k)}=\boldsymbol{\phi}(\boldsymbol{W}^{(k-1)}\boldsymbol{x}^{(k-1)}+\boldsymbol{v}^{(k-1)})$, $\forall k\in[\ell-1]$, where $\boldsymbol{x}^{(0)}=\boldsymbol{x}$ is the input of the FNN $\boldsymbol{\pi}$ and $\boldsymbol{\phi}$ is the vector-valued activation function constructed by component-wise repetition of the activation function $\sigma(\cdot)$, i.e., $\boldsymbol{\phi}(\boldsymbol{z})\triangleq[\sigma(z_1)\ \cdots\ \sigma(z_{n_k})]^\top$. In the last layer, only the linear map is applied, i.e., $\boldsymbol{\pi}(\boldsymbol{x})=\boldsymbol{x}^{(\ell)}=\boldsymbol{W}^{(\ell-1)}\boldsymbol{x}^{(\ell-1)}+\boldsymbol{v}^{(\ell-1)}$. Although only ReLU activation functions are considered in this letter, the proposed methods can be easily extended to other types of activation functions by using their HZ approximation as in [15]. Given an input set $\mathcal{Z}\subset\mathbb{R}^n$ of the FNN $\boldsymbol{\pi}$, the image set of $\mathcal{Z}$ is defined as $\boldsymbol{\pi}(\mathcal{Z})=\{z\in\mathbb{R}^m\mid z=\boldsymbol{\pi}(\boldsymbol{x}),\boldsymbol{x}\in\mathcal{Z}\}$ and the graph of $\boldsymbol{\pi}$ over $\mathcal{Z}$ is defined as $\mathcal{G}_{\boldsymbol{\pi}}(\mathcal{Z})\triangleq\{(\boldsymbol{x},z)\in\mathbb{R}^{n+m}\mid z=\boldsymbol{\pi}(\boldsymbol{x}),\boldsymbol{x}\in\mathcal{Z}\}$.

The plant considered in this letter is given as

$$\boldsymbol{x}(t+1)=\boldsymbol{A}_d\boldsymbol{x}(t)+\boldsymbol{B}_d\boldsymbol{u}(t) \tag{1}$$

where $\boldsymbol{x}(t)\in\mathcal{X}\subset\mathbb{R}^n$, $\boldsymbol{u}(t)\in\mathbb{R}^m$ are the state and the control input, respectively. The control input is $\boldsymbol{u}(t)=\boldsymbol{\pi}(\boldsymbol{x}(t))$ where $\boldsymbol{\pi}$ is a given ℓ-layer FNN. The NNCS consisting of system (1) and the controller $\boldsymbol{\pi}$ is a closed-loop system:

$$\boldsymbol{x}(t+1)=\boldsymbol{f}_{cl}(\boldsymbol{x}(t))\triangleq\boldsymbol{A}_d\boldsymbol{x}(t)+\boldsymbol{B}_d\boldsymbol{\pi}(\boldsymbol{x}(t)). \tag{2}$$

Given an initial set $\mathcal{X}_0\subset\mathcal{X}$ for the NNCS (2), its T-step FRS is defined as $\mathcal{R}_T(\mathcal{X}_0)\triangleq\{\boldsymbol{x}(T)\in\mathcal{X}\mid x(t)=\boldsymbol{f}_{cl}(\boldsymbol{x}(t-1)),x(0)\in\mathcal{X}_0,t\in[T]\}$; given a target set $\mathcal{T}\subset\mathcal{X}$, its T-step BRS is defined as $\mathcal{P}_T(\mathcal{T})\triangleq\{\boldsymbol{x}(0)\in\mathcal{X}\mid x(t)=\boldsymbol{f}_{cl}(\boldsymbol{x}(t-1)),x(T)\in\mathcal{T},t\in[T]\}$. We assume the state set $\mathcal{X}$, target set $\mathcal{T}$, and initial set $\mathcal{X}_0$ are all represented as HZs.

In this letter, we aim to develop a systematic, HZ-based approach for computing the FRS and BRS of the NNCS (2) with a tunable trade-off between computational efficiency and approximation accuracy.

III. Tunable FNN Reduction

In this section, we present a flexible optimization-based approach for reducing the number of neurons of a given FNN while preserving its important input-output mapping property. Specifically, given an ℓ-layer FNN $\boldsymbol{\pi}$ and an input domain $\mathcal{Z}$, we aim to construct a new FNN $\tilde{\boldsymbol{\pi}}$ that has a reduced number of neurons than $\boldsymbol{\pi}$ and that over-approximates the original FNN $\boldsymbol{\pi}$ over the domain $\mathcal{Z}$, i.e., $\boldsymbol{\pi}(\mathcal{Z})\subseteq\tilde{\boldsymbol{\pi}}(\mathcal{Z})$. The main idea of our reduction approach is to group "similar" neurons in each layer of the FNN according to *variable merge buckets* that is defined below.

Definition 2: Given an ℓ-layer FNN π, an input set $\mathcal{Z}$, an interval $\mathcal{I}^{(k)}\triangleq[[\boldsymbol{\alpha}^{(k)},\boldsymbol{\beta}^{(k)}]]\subset\mathbb{R}^{n_k}$ that bounds the ranges of the neurons in the k-th layer where $k\in[\ell-1]$, a set of scalar-valued centers $\{c_j^{(k)}\}_{j=1}^p$, and a set of scalar-valued tolerances $\{\delta_j^{(k)}\}_{j=1}^p$, then a variable merge bucket of the k-th layer is defined as

$$\mathcal{B}^{(k)}\triangleq\mathcal{B}_1^{(k)}\cup\mathcal{B}_2^{(k)}\cup\cdots\cup\mathcal{B}_p^{(k)}\subseteq[n_k], \tag{3}$$

with $\mathcal{B}_j^{(k)}\triangleq\{i\in[n_k]\mid[[\alpha_i^{(k)},\beta_i^{(k)}]]\subseteq[[c_j^{(k)}-\delta_j^{(k)},c_j^{(k)}+\delta_j^{(k)}]]\}$ for $j\in[p]$, where each neuron index $i\in[n_k]$ can only belong to at most one bucket in $\mathcal{B}^{(k)}$.

We call a neuron reducible if it is contained in a bucket of $\mathcal{B}^{(k)}$ with other neurons or if its range is a singleton (i.e., $\alpha_i^{(k)}=\beta_i^{(k)}$). By definition, each bucket $\mathcal{B}_j^{(k)}$ contains the indices of similar neurons whose output ranges fall into an interval with center $c_j^{(k)}$ and radius $\delta_j^{(k)}$. To balance the number of remaining neurons and approximation accuracy of the reduced FNN, we formulate the following MILP to identify the optimal centers $\{c_j^{(k)}\}_{j=1}^p$ and tolerances $\{\delta_j^{(k)}\}_{j=1}^p$ of the variable merge bucket $\mathcal{B}^{(k)}$. Note that the superscript, k, is dropped in the MILP for better readability.

$$\min_{\{c_j\},\{\delta_j\},\{b_{i,j}\},\{d_j\}}\ \lambda\sum_{j=1}^p\delta_j-\sum_{i=1}^{n_k}\sum_{j=1}^p b_{i,j}-\sum_{j=1}^p d_j \tag{4a}$$

$$\text{s.t.}\quad 0\le\delta_j\le\bar{\delta},\ \ b_{i,j}\in\{0,1\},\ \ d_j\in\{0,1\},c_j\in\mathbb{R}, \tag{4b}$$

$$\sum_{j=1}^p b_{i,j}\le 1, \tag{4c}$$

$$\alpha_i-c_j+\delta_j\ge -M\big(1-b_{i,j}\big), \tag{4d}$$

$$\beta_i-c_j-\delta_j\le M\big(1-b_{i,j}\big), \tag{4e}$$

$$\sum_{i=1}^{n_k}b_{i,j}\le M\big(1-d_j\big),\ \forall i\in[n_k],\ \forall j\in[p]. \tag{4f}$$

In (4), the k-th layer interval bounds $\mathcal{I}^{(k)}=[[\boldsymbol{\alpha}^{(k)},\boldsymbol{\beta}^{(k)}]]$, the number of buckets $p\in\mathbb{Z}_{>0}$, the *tunable tolerance parameter*

$\bar{\delta} \in \mathbb{R}_{\geq 0}$, and a sufficiently large positive constant M are all given. The binary variable $b_{i,j}$ indicates whether the i-th neuron is in the j-th bucket through constraints (4d)-(4e); the binary variable d_j indicates whether the j-th bucket is empty through constraint (4f); constraint (4c) indicates that a neuron is assigned to at most one bucket; the objective function (4a) is formulated to maximize the number of neurons to be reduced while minimizing the total sizes of the tolerances; the weight parameter λ in (4a) is used to balance the sizes of buckets and the number of neurons to be reduced. It's easy to check that the MILP (4) is always feasible.

Remark 1: Compared with other existing works on NN reduction such as [13], [14], our method can abstract the FNN with the reduction metrics determined on the fly in an optimal manner. Given $\mathcal{I}$, $p \in \mathbb{Z}_{>0}$, $\bar{\delta} > 0$ and $0 < \lambda < \frac{1}{p\bar{\delta}}$ in the MILP (4), let $\mathcal{B}^*$ be the variable merge bucket corresponding to the optimal solution of MILP (4). It's easy to check that all the neurons in $\mathcal{B}^*$ are reducible. Moreover, guided by the objective function (4a), $\mathcal{B}^*$ contains the maximum number of reducible neurons with the least number of non-empty buckets.

After creating the variable merge bucket by solving (4), all neurons contained in the variable merge bucket will be removed and the induced approximation error will be added to the next layer to ensure an over-approximation of the original FNN. This is summarized in the following lemma; the proof of this lemma is similar to that of [13, Proposition 4] and is omitted due to the space limitation.

Lemma 1: For the k-th layer of an FNN $\boldsymbol{\pi}$, $k \in [\ell - 1]$, given the interval bounds $\mathcal{I}^{(k)} \subset \mathbb{R}^{n_k}$ for the neurons in the k-th layer and the variable merge bucket $\mathcal{B}^{(k)}$, a reduced network $\tilde{\boldsymbol{\pi}}$ is constructed by adjusting the weights and bias of the $(k-1)$-th and k-th layers as follows:

$$\begin{aligned} \tilde{\boldsymbol{W}}^{(k-1)} &= \boldsymbol{W}^{(k-1)}\left[\overline{\mathcal{B}}^{(k)}, :\right], \ \tilde{\boldsymbol{v}}^{(k-1)} = \boldsymbol{v}^{(k-1)}\left[\overline{\mathcal{B}}^{(k)}, :\right], \\ \tilde{\boldsymbol{W}}^{(k)} &= \boldsymbol{W}^{(k)}\left[:, \overline{\mathcal{B}}^{(k)}\right], \ \tilde{\boldsymbol{v}}^{(k)} = \boldsymbol{v}^{(k)} + \boldsymbol{\varepsilon}^{(k)}, \end{aligned} \tag{5}$$

where $\overline{\mathcal{B}}^{(k)} \triangleq [n_k] \backslash \mathcal{B}^{(k)}$ denotes the index set of remaining neurons and $\tilde{\boldsymbol{v}}^{(k)}$ includes the approximation error $\boldsymbol{\varepsilon}^{(k)} \triangleq \sum_{j=1}^{p} \boldsymbol{W}^{(k)}[:, B_j^{(k)}] \cdot \operatorname{proj}_{B_j^{(k)}}(\mathcal{I}^{(k)})$. Then, $\tilde{\boldsymbol{\pi}}$ over-approximates $\boldsymbol{\pi}$ over the domain $\mathcal{Z}$, i.e., $\boldsymbol{\pi}(\mathcal{Z}) \subseteq \tilde{\boldsymbol{\pi}}(\mathcal{Z})$.

The reduced network $\tilde{\boldsymbol{\pi}}$ can be computed by applying Lemma 1 layer-by-layer as summarized in Algorithm 1. Specifically, for the k-th layer of $\tilde{\boldsymbol{\pi}}$, the output set $\mathcal{X}^{(k)}$ is computed through the function `propagate` in Line 4, which represents FNN output computation algorithms such as [9, Algorithm 1]. In Line 5, the interval hull of the HZ set $\mathcal{X}^{(k)}$ can be calculated exactly by solving a set of $2n_k$ MILPs to find the upper and lower bounds in the n_k cardinal directions, as detailed in [16, Proposition 3.2.10]. Based on the interval bounds, a set of valid buckets is created by solving MILP (4) in Line 6. Finally, in Line 7, weights and bias are adjusted according to Lemma 1.

Proposition 1: Given an ℓ-layer FNN $\boldsymbol{\pi}$ and an input domain $\mathcal{Z}$, Algorithm 2 returns a reduced FNN $\tilde{\boldsymbol{\pi}}$, such that $\boldsymbol{\pi}(\mathcal{Z}) \subseteq \tilde{\boldsymbol{\pi}}(\mathcal{Z})$ and $\mathcal{G}_{\pi}(\mathcal{Z}) \subseteq \mathcal{G}_{\tilde{\pi}}(\mathcal{Z})$. Moreover, $\boldsymbol{\pi}(\mathcal{Z}) = \tilde{\boldsymbol{\pi}}(\mathcal{Z})$ and $\mathcal{G}_{\pi}(\mathcal{Z}) = \mathcal{G}_{\tilde{\pi}}(\mathcal{Z})$ when $\bar{\delta} = 0$.

Proof: By construction, after adjusting the weights and bias of each layer in Line 7 of Algorithm 2, $\tilde{\boldsymbol{\pi}}$ over-approximates $\boldsymbol{\pi}$ according to Lemma 1. When $\bar{\delta} = 0$, all the bucket tolerances are forced to be 0 and no approximation error will be propagated through the reduced FNN (i.e., $\boldsymbol{\varepsilon}^{(k)} = 0$). Thus, $\boldsymbol{\pi}(z) = \tilde{\boldsymbol{\pi}}(z)$ and $\mathcal{G}_{\pi}(\mathcal{Z}) \subseteq \mathcal{G}_{\tilde{\pi}}(\mathcal{Z})$. ■

Algorithm 1: Optimization-Based FNN Reduction

```
Input: input domain $\mathcal{Z}$, FNN $\boldsymbol{\pi}$ with weight matrices
       $\{\boldsymbol{W}^{(k-1)}\}_{k=1}^{\ell}$ and bias vectors $\{\boldsymbol{v}^{(k-1)}\}_{k=1}^{\ell}$, the
       number of buckets $p \in \mathbb{Z}_{>0}$, a sufficiently large
       number $M > 0$, tunable tolerance bound $\bar{\delta} \geq 0$,
       weight parameter $0 < \lambda < \frac{1}{p\bar{\delta}}$
Output: reduced FNN $\tilde{\boldsymbol{\pi}}$ with weight matrices
        $\{\tilde{\boldsymbol{W}}^{(k-1)}\}_{k=1}^{\ell}$ and bias vectors $\{\tilde{\boldsymbol{v}}^{(k-1)}\}_{k=1}^{\ell}$
1 $\mathcal{X}^{(0)} \leftarrow \mathcal{Z}$;
2 $\tilde{\boldsymbol{W}}^{(0)} \leftarrow \boldsymbol{W}^{(0)}$; $\tilde{\boldsymbol{v}}^{(0)} \leftarrow \boldsymbol{v}^{(0)}$;
3 for $k \in \{1, 2, \dots, \ell - 1\}$ do
4     $\mathcal{X}^{(k)} \leftarrow$ propagate($\boldsymbol{\phi}$, $\tilde{\boldsymbol{W}}^{(k-1)}$, $\tilde{\boldsymbol{v}}^{(k-1)}$, $\mathcal{X}^{(k-1)}$);
5     $\mathcal{I}^{(k)} \leftarrow$ interval($\mathcal{X}^{(k)}$);    // Using [16, Prop.
      3.2.10]
6     $\mathcal{B}^{(k)}, \overline{\mathcal{B}}^{(k)} \leftarrow$ solving MILP (4) with $\mathcal{I}^{(k)}$;
7     $\tilde{\boldsymbol{W}}^{(k-1)}, \tilde{\boldsymbol{v}}^{(k-1)}, \tilde{\boldsymbol{W}}^{(k)}, \tilde{\boldsymbol{v}}^{(k)} \leftarrow$ (5) in Lemma 1;
8 return $\{\tilde{\boldsymbol{W}}^{(k-1)}\}_{k=1}^{\ell}$, $\{\tilde{\boldsymbol{v}}^{(k-1)}\}_{k=1}^{\ell}$
```

Note that the input domain $\mathcal{Z}$ can be any set representations as long as the interval bounds in Line 5 of Algorithm 1 can be computed. However, as shown in [10], the input-output mapping of ReLU-activated FNNs can be represented *exactly* by HZs, and the tightest interval bounds can be computed by the interval hull of HZs. So we will use HZ as the set representation for FNNs and NNCS in the following.

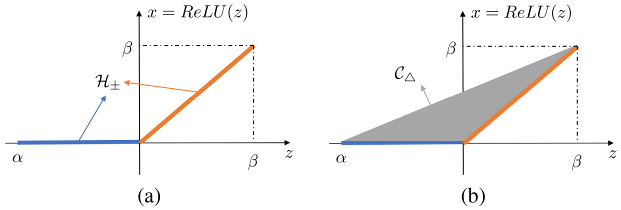
IV. Tunable HZ Propagation of FNNs and NNCS

In this section, we first present an approach for propagating an HZ through a given FNN $\boldsymbol{\pi}$ by approximating its graph $\mathcal{G}_{\boldsymbol{\pi}}$ with a tunable trade-off between computational efficiency and approximation accuracy, and based on that, compute the FRSs and BRSs of the NNCS (2).

Motivated by the fact that ReLU-activated FNNs usually observe limited numbers of activation patterns [17], we first propose a novel graph computation approach to construct a relaxed over-approximation of the graph $\mathcal{G}_{\boldsymbol{\pi}}$.

Consider the graph of a scalar-valued univariate ReLU function $x = \mathit{ReLU}(z)$ over an interval $[[\alpha, \beta]] \subset \mathbb{R}$, i.e., $\mathcal{G}_{ReLU}([[\alpha, \beta]]) \triangleq \{(z, x) \in \mathbb{R}^2 \mid x = \mathit{ReLU}(z), z \in [[\alpha, \beta]]\}$. Depending on the activation pattern of *ReLU*, the graph can be represented as the line segment in the first quadrant $\mathcal{H}_+$, the line segment on the negative z-axis $\mathcal{H}_-$, or the union of two line segments $\mathcal{H}_\pm$ (see Fig. 2). Specifically,

$$\mathcal{G}_{ReLU}([[\alpha, \beta]]) \\ = \begin{cases} \mathcal{H}_+ \triangleq \langle \begin{bmatrix} \frac{\beta-\alpha}{2} \\ \frac{\beta-\alpha}{2} \end{bmatrix}, \emptyset, \begin{bmatrix} \frac{\beta+\alpha}{2} \\ \frac{\beta+\alpha}{2} \end{bmatrix}, \emptyset, \emptyset, \emptyset \rangle, \text{ if } 0 \leq \alpha \leq \beta, \\ \mathcal{H}_- \triangleq \langle \begin{bmatrix} \frac{\beta-\alpha}{2} \\ 0 \end{bmatrix}, \emptyset, \begin{bmatrix} \frac{\beta+\alpha}{2} \\ 0 \end{bmatrix}, \emptyset, \emptyset, \emptyset \rangle, \text{ if } \alpha \leq \beta \leq 0, \\ \quad \mathcal{H}_\pm \triangleq \langle \mathbf{G}_h^c, \mathbf{G}_h^b, \mathbf{c}_h, \mathbf{A}_h^c, \mathbf{A}_h^b, \mathbf{b}_h \rangle, \text{ if } \alpha < 0 < \beta, \end{cases}$$



(a) (b)

Fig. 2. The graph of a ReLU function over $[[\alpha, \beta]]$ can be represented as an HZ. (a) Graph $\boldsymbol{\mathcal{H}}_{\pm}$ with $\alpha < 0 < \beta$, (b) The triangle over-approximation of the graph $\boldsymbol{\mathcal{C}}_{\triangle} \supset \boldsymbol{\mathcal{H}}_{\pm}$.

where expressions of $\mathbf{G}_h^c, \mathbf{G}_h^b, \mathbf{c}_h, \mathbf{A}_h^c, \mathbf{A}_h^b, \mathbf{b}_h$ can be found in [10, eq. (3)]. The next lemma generalizes our previous result in [10] and presents an HZ representation for the graph of the vector-valued ReLU function $\boldsymbol{\phi}$ over an HZ domain.

Lemma 2: Given a domain represented as an HZ $\mathcal{Z} \subset \mathbb{R}^{n_k}$ and its interval hull $\mathcal{I} \triangleq [[\boldsymbol{\alpha}, \boldsymbol{\beta}]] = \text{interval}(\mathcal{Z})$, then the graph of the k-th layer's vector-valued activation function $\boldsymbol{\phi} : \mathbb{R}^{n_k} \to \mathbb{R}^{n_k}$ over $\mathcal{Z}$ can be exactly represented as

$$\mathcal{G}_{\boldsymbol{\phi}}(\mathcal{Z}) = (\boldsymbol{P} \cdot \mathcal{G}_{ReLU}(\mathcal{I})) \cap_{[\boldsymbol{I}\ \mathbf{0}]} \mathcal{Z} \tag{6}$$

where $\boldsymbol{P} = [\boldsymbol{e}_2\ \boldsymbol{e}_4\ \cdots\ \boldsymbol{e}_{2n_k}\ \boldsymbol{e}_1\ \boldsymbol{e}_3\ \cdots\ \boldsymbol{e}_{2n_k-1}]^T \in \mathbb{R}^{2n_k \times 2n_k}$ is a permutation matrix and $\mathcal{G}_{ReLU}(\mathcal{I}) = \mathcal{G}_{ReLU}([[\alpha_1, \beta_1]]) \times \cdots \times \mathcal{G}_{ReLU}([[\alpha_{n_k}, \beta_{n_k}]])$.

The proof of Lemma 2 follows the same procedures as [10, Lemma 2] and thus is omitted. As $\mathcal{H}_+$ and $\mathcal{H}_-$ are degenerated HZs without binary generators, the set complexity of the constructed HZ using (6) is reduced by exploiting the activation patterns of each neuron.

To further reduce the complexity of the computed HZ-represented graph sets, the set $\mathcal{H}_{\pm}$ can be over-approximated by its triangle-shaped convex hull $\mathcal{C}_{\triangle}$ as shown in Fig. 2, i.e., $\mathcal{H}_{\pm} \subset \mathcal{C}_{\triangle} \triangleq \langle \big[\mathbf{G}_h^c\ \mathbf{G}_h^b\big], \emptyset, \mathbf{c}_h, \big[\mathbf{A}_h^c\ \mathbf{A}_h^b\big], \emptyset, \mathbf{b}_h \rangle$. To limit the conservatism on the over-approximated graphs, we introduce a *tunable relaxation parameter* $0 \leq \gamma \leq 1$ such that $\mathcal{H}_{\pm}$ is only replaced by the relaxed set $\mathcal{C}_{\triangle}$ when the ratio between α and β is large enough, i.e.,

$$\tilde{\mathcal{G}}_{ReLU}([[\alpha, \beta]]) = \begin{cases} \mathcal{C}_{\triangle}, \text{ if } (\alpha < 0 < \beta) \wedge \left(\frac{|\alpha|}{\beta} \leq \gamma \vee \frac{\beta}{|\alpha|} \leq \gamma\right), \\ \mathcal{G}_{ReLU}([[\alpha, \beta]]), \quad \text{otherwise.} \end{cases}$$

Based on the relaxed formulation above, Lemma 2 can be naturally extended to over-approximate the graph of the vector-valued activation function $\boldsymbol{\phi}$ over an HZ domain.

Proposition 2: Given an HZ $\mathcal{Z} \subset \mathbb{R}^{n_k}$, its interval hull $\mathcal{I} = \text{interval}(\mathcal{Z})$ and the tunable relaxation parameter $0 \leq \gamma \leq 1$, the graph of the activation function $\boldsymbol{\phi} : \mathbb{R}^{n_k} \to \mathbb{R}^{n_k}$ over $\mathcal{Z}$ can be over-approximated by the following HZ:

$$\tilde{\mathcal{G}}_{\boldsymbol{\phi}}(\mathcal{Z}) = \Big(\boldsymbol{P} \cdot \tilde{\mathcal{G}}_{ReLU}(\mathcal{I})\Big) \cap_{[\boldsymbol{I}\ \mathbf{0}]} \mathcal{Z} \supseteq \mathcal{G}_{\boldsymbol{\phi}}(\mathcal{Z}) \tag{7}$$

where $\boldsymbol{P}$ and $\tilde{\mathcal{G}}_{ReLU}(\mathcal{I})$ are defined similarly as in Lemma 2. Moreover, when $\gamma = 0$, $\tilde{\mathcal{G}}_{\boldsymbol{\phi}}(\mathcal{Z}) = \mathcal{G}_{\boldsymbol{\phi}}(\mathcal{Z})$.

With the increase of the tunable relaxation parameter γ, more graphs of individual neurons represented by $\mathcal{H}_{\pm}$ will be approximated by the relaxed set $\mathcal{C}_{\triangle}$, resulting in a larger over-approximated graph set of the activation function $\boldsymbol{\phi}$. In the extreme case $\gamma = 1$, all the sets $\mathcal{H}_{\pm}$ will be relaxed into $\mathcal{C}_{\triangle}$ and $\tilde{\mathcal{G}}_{\boldsymbol{\phi}}(\mathcal{Z})$ will become a degenerated HZ without binary generators. On the other hand, when $\gamma = 0$, no relaxation is performed and therefore, $\mathcal{G}_{\boldsymbol{\phi}}(\mathcal{Z}) = \tilde{\mathcal{G}}_{\boldsymbol{\phi}}(\mathcal{Z})$.

Algorithm 2: Tunable Graph Over-Approximation of FNN

```
Input: HZ input set $\mathcal{Z}$, original FNN $\boldsymbol{\pi}$ with weight
       matrices $\{\boldsymbol{W}^{(k-1)}\}_{k=1}^{\ell}$ and bias vectors
       $\{\boldsymbol{v}^{(k-1)}\}_{k=1}^{\ell}$, number of buckets $p \in \mathbb{Z}_{>0}$, large
       number $M > 0$, tolerance bound $\bar{\delta} \geq 0$, weight
       parameter $\lambda \geq 0$, relaxation parameter $0 \leq \gamma \leq 1$
Output: Over-approximated graph $\tilde{\mathcal{G}}_{\boldsymbol{\pi}}$ as an HZ
1  $\mathcal{X}^{(0)} \leftarrow \mathcal{Z} = \langle \mathbf{G}_z^c, \mathbf{G}_z^b, \mathbf{c}_z, \mathbf{A}_z^c, \mathbf{A}_z^b, \mathbf{b}_z \rangle$;
2  $\tilde{\boldsymbol{W}}^{(0)} \leftarrow \boldsymbol{W}^{(0)}$; $\tilde{\boldsymbol{v}}^{(0)} \leftarrow \boldsymbol{v}^{(0)}$;
3  for $k \in \{1, 2, \ldots, \ell - 1\}$ do
4      $\mathcal{Z}^{(k-1)} \leftarrow \tilde{\boldsymbol{W}}^{(k-1)} \mathcal{X}^{(k-1)} + \tilde{\boldsymbol{v}}^{(k-1)}$;
5      $\mathcal{I}_z^{(k-1)} = [[\boldsymbol{\alpha}^{(k-1)}, \boldsymbol{\beta}^{(k-1)}]] \leftarrow \text{interval}(\mathcal{Z}^{k-1})$;
6      $\mathcal{I}_x^{(k)} \leftarrow \boldsymbol{\phi}(I_z^{k-1})$;
7      $\mathcal{B}^{(k)}, \overline{\mathcal{B}}^{(k)} \leftarrow$ solving MILP (4) with $\mathcal{I}_x^{(k)}$;
8      $\tilde{\boldsymbol{W}}^{(k-1)}, \tilde{\boldsymbol{v}}^{(k-1)}, \tilde{\boldsymbol{W}}^{(k)}, \tilde{\boldsymbol{v}}^{(k)} \leftarrow$ (5) in Lemma 1;
9      $\tilde{\mathcal{Z}}^{(k-1)} \leftarrow \text{proj}_{\overline{\mathcal{B}}^{(k)}}(\mathcal{Z}^{(k-1)})$;        // Linear map
10     $\tilde{\mathcal{I}}_z^{(k-1)} \leftarrow \text{proj}_{\overline{\mathcal{B}}^{(k)}}(\mathcal{I}_z^{(k-1)})$;        // Linear map
11     $\tilde{\mathcal{G}}^{(k)} \leftarrow (\boldsymbol{P} \cdot \tilde{\mathcal{G}}_{ReLU}(\tilde{\mathcal{I}}_z^{(k-1)})) \cap_{[\boldsymbol{I}\ \mathbf{0}]} \tilde{\mathcal{Z}}^{(k-1)}$;      // HZ
       intersection using [11, Prop. 7]
12     $\mathcal{X}^{(k)} \leftarrow [\mathbf{0}\ \boldsymbol{I}] \cdot \tilde{\mathcal{G}}^{(k)}$;      // Next layer input
13 $\langle \mathbf{G}^c, \mathbf{G}^b, \mathbf{c}, \mathbf{A}^c, \mathbf{A}^b, \mathbf{b} \rangle \leftarrow \tilde{\boldsymbol{W}}^{(\ell-1)} \mathcal{X}^{(\ell-1)} + \tilde{\boldsymbol{v}}^{(\ell-1)}$;
14 $\tilde{\mathcal{G}}_{\boldsymbol{\pi}} \leftarrow \langle \begin{bmatrix} \mathbf{G}_z^c & \mathbf{0} \\ \mathbf{G}^c \end{bmatrix}, \begin{bmatrix} \mathbf{G}_z^b & \mathbf{0} \\ \mathbf{G}^b \end{bmatrix}, \begin{bmatrix} \mathbf{c}_z \\ \mathbf{c} \end{bmatrix}, \mathbf{A}^c, \mathbf{A}^b, \mathbf{b} \rangle$;
15 return $\tilde{\mathcal{G}}_{\boldsymbol{\pi}}$
```

To construct an HZ over-approximation of the graph $\mathcal{G}_{\boldsymbol{\pi}}$ for $\boldsymbol{\pi}$, we can propagate the input set as an HZ through the reduced FNN $\tilde{\boldsymbol{\pi}}$ layer-by-layer using Proposition 2 and linear map operations of HZs as summarized in Algorithm 2.

To incorporate the FNN reduction method presented in Section III with the activation pattern-based graph computation, both the pre-activation interval bounds and the post-activation interval bounds of neurons in each layer are needed in Algorithm 2. Instead of solving two sets of MILPs to get the interval hulls of pre-activated HZ $\mathcal{Z}^{(k-1)}$ and post-activated HZ $\boldsymbol{\phi}(\mathcal{Z}^{(k-1)})$, we only compute the interval hull once in Line 5 by solving a set of $2n_k$ MILPs, similar to Line 5 in Algorithm 1. The pre-activation interval bounds are then propagated through the activation in Line 6 to get post-activation interval bounds. For monotonic activation functions like ReLU, the interval propagation can be computed efficiently and exactly without introducing any conservatism; in other words, interval$(\boldsymbol{\phi}(\mathcal{Z})) = \boldsymbol{\phi}(\text{interval}(\mathcal{Z}))$ holds. Since the size of the FNN $\boldsymbol{\pi}$ decreases after applying Lemma 1 for each iteration in Line 8, the HZ $\mathcal{Z}^{(k-1)}$ and the interval $\mathcal{I}_z^{(k-1)}$ computed from the original FNN are projected onto the set of coordinates corresponding to the remaining neurons (i.e., $\overline{\mathcal{B}}^{(k)}$) in Line 9 and Line 10.

The following theorem shows that the graph over-approximation in Algorithm 2 is sound.

Theorem 1: Given an ℓ-layer ReLU-activated FNN $\boldsymbol{\pi} : \mathbb{R}^n \to \mathbb{R}^m$ and an HZ $\mathcal{Z} \subset \mathbb{R}^n$, the output of Algorithm 2

$\tilde{\mathcal{G}}_{\boldsymbol{\pi}}$ is an HZ that over-approximates the exact graph of $\boldsymbol{\pi}$ over $\mathcal{Z}$, i.e., $\tilde{\mathcal{G}}_{\boldsymbol{\pi}} \supseteq \mathcal{G}_{\boldsymbol{\pi}}(\mathcal{Z})$. Furthermore, $\tilde{\mathcal{G}}_{\boldsymbol{\pi}} = \mathcal{G}_{\boldsymbol{\pi}}(\mathcal{Z})$ when $\bar{\delta} = 0$ and $\gamma = 0$.

Proof: A reduced FNN $\tilde{\boldsymbol{\pi}}$ is constructed using (5) in Line 8 of Algorithm 2. In Line 9-12, the over-approximated input set $\tilde{\mathcal{Z}}^{(k-1)}$, graph $\tilde{\mathcal{G}}^{(k)}$ and output set $\mathcal{X}^{(k)}$ of the k-th layer of the reduced FNN is computed iteratively for $k \in [\ell - 1]$. Thus, the over-approximation properties are preserved through the propagation of each hidden layer. Only a linear map is applied to the last layer in Line 13 and $\tilde{\mathcal{G}}_{\boldsymbol{\pi}}$ stacks the input set and the over-approximated output set of $\tilde{\boldsymbol{\pi}}$ as $\tilde{\mathcal{G}}_{\boldsymbol{\pi}} \supseteq \mathcal{G}_{\tilde{\boldsymbol{\pi}}}(\mathcal{Z})$. As $\tilde{\boldsymbol{\pi}}$ over-approximates $\boldsymbol{\pi}$ over $\mathcal{Z}$, $\tilde{\mathcal{G}}_{\boldsymbol{\pi}} \supseteq \mathcal{G}_{\tilde{\boldsymbol{\pi}}}(\mathcal{Z}) \supseteq \mathcal{G}_{\boldsymbol{\pi}}(\mathcal{Z})$. Since $\mathcal{Z}$ is an HZ and HZs are closed under all the set operations involved in Algorithm 2, $\tilde{\mathcal{G}}_{\boldsymbol{\pi}}$ is also an HZ by construction. When $\bar{\delta} = \gamma = 0$, $\tilde{\boldsymbol{\pi}}$ preserves the same input-output relationship of $\boldsymbol{\pi}$ as shown in Proposition 1, and the constructed graph set in Line 11 is exact for each layer by Proposition 2. Thus, $\tilde{\mathcal{G}}_{\boldsymbol{\pi}} = \mathcal{G}_{\boldsymbol{\pi}}(\mathcal{Z})$. ■

Remark 2: Given the calculated graph $\tilde{\mathcal{G}}_{\boldsymbol{\pi}}$ of Algorithm 2, for any HZ-represented input set $\mathcal{Z}_{in} \subseteq \mathcal{Z}$, the image set of $\boldsymbol{\pi}$ can be over-approximated by $\tilde{\mathcal{Z}}_{image} = [\mathbf{0}_{m\times n} \;\; \boldsymbol{I}_m] \cdot (\tilde{\mathcal{G}}_{\boldsymbol{\pi}} \cap_{[\boldsymbol{I}_n \; \mathbf{0}_{n\times m}]} \mathcal{Z}_{in})$; for any HZ-represented output set $\mathcal{Z}_{out} \subseteq \boldsymbol{\pi}(\mathcal{Z})$, the preimage set of $\boldsymbol{\pi}$ can be over-approximated by $\tilde{\mathcal{Z}}_{pre} = [\boldsymbol{I}_n \;\; \mathbf{0}_{n\times m}] \cdot (\tilde{\mathcal{G}}_{\boldsymbol{\pi}} \cap_{[\mathbf{0}_{m\times n} \; \boldsymbol{I}_m]} \mathcal{Z}_{out})$.

Finally, we give the following theorem that computes the one-step over-approximated FRS and BRS for the NNCS (2).

Theorem 2: Consider NNCS (2) and any given HZ $\mathcal{Z} \subset \mathbb{R}^n$. Let $\tilde{\mathcal{G}}_{\boldsymbol{\pi}}$ be the over-approximated graph set of the FNN $\boldsymbol{\pi}$ over the domain $\mathcal{Z}$ using Algorithm 2, i.e., $\tilde{\mathcal{G}}_{\boldsymbol{\pi}} \supseteq \mathcal{G}_{\boldsymbol{\pi}}(\mathcal{Z})$.

(i) For any initial set $\mathcal{X}_0 \subseteq \mathcal{Z}$ represented by an HZ, the one-step FRS of the NNCS (2) can be over-approximated by the HZ $\tilde{\mathcal{R}}(\mathcal{X}_0) = [\boldsymbol{A}_d \; \boldsymbol{B}_d] \cdot (\tilde{\mathcal{G}}_{\boldsymbol{\pi}} \cap_{[\boldsymbol{I}_n \; \mathbf{0}_{n\times m}]} \mathcal{X}_0) \supseteq \mathcal{R}(\mathcal{X}_0)$.

(ii) For any target set $\mathcal{T} \subset \mathbb{R}^n$ represented by an HZ, the one-step BRS of the NNCS (2) in the domain $\mathcal{Z}$ can be over-approximated by the HZ $\tilde{\mathcal{P}}(\mathcal{T}) = [\boldsymbol{I}_n \; \mathbf{0}_{n\times m}] \cdot (\tilde{\mathcal{G}}_{\boldsymbol{\pi}} \cap_{[\boldsymbol{A}_d \; \boldsymbol{B}_d]} \mathcal{T}) \supseteq \mathcal{P}(\mathcal{T}) \cap \mathcal{Z}$.

(iii) The over-approximation in (i) and (ii) becomes exact when $\bar{\delta} = \gamma = 0$.

Proof: (i) Since $\tilde{\mathcal{G}}_{\boldsymbol{\pi}} \supseteq \mathcal{G}_{\boldsymbol{\pi}}(\mathcal{Z}) = \{(\boldsymbol{x}, \boldsymbol{u}) \in \mathbb{R}^{n+m} \mid \boldsymbol{x} \in \mathcal{Z}, \boldsymbol{u} = \boldsymbol{\pi}(\boldsymbol{x})\}$ and $\mathcal{R}(\mathcal{X}_0) = \{\boldsymbol{A}_d\boldsymbol{x} + \boldsymbol{B}_d\boldsymbol{u} \mid \boldsymbol{x} \in \mathcal{X}_0, \boldsymbol{u} = \boldsymbol{\pi}(\boldsymbol{x})\}$, we have $\tilde{\mathcal{R}}(\mathcal{X}_0) \supseteq [\boldsymbol{A}_d \;\; \boldsymbol{B}_d] \cdot (\mathcal{G}_{\boldsymbol{\pi}}(\mathcal{Z}) \cap_{[\boldsymbol{I} \; \mathbf{0}]} \mathcal{X}_0) = \{\boldsymbol{A}_d\boldsymbol{x} + \boldsymbol{B}_d\boldsymbol{u} \mid \boldsymbol{x} \in (\mathcal{Z} \cap \mathcal{X}_0), \boldsymbol{u} = \boldsymbol{\pi}(\boldsymbol{x})\} = \mathcal{R}(\mathcal{X}_0)$.

(ii) Since $\tilde{\mathcal{G}}_{\boldsymbol{\pi}} \supseteq \mathcal{G}_{\boldsymbol{\pi}}(\mathcal{Z}) = \{(\boldsymbol{x}, \boldsymbol{u}) \in \mathbb{R}^{n+m} \mid \boldsymbol{x} \in \mathcal{Z}, \boldsymbol{u} = \boldsymbol{\pi}(\boldsymbol{x})\}$, we have $\tilde{\mathcal{P}}(\mathcal{T}) \supseteq [\boldsymbol{I}_n \; \mathbf{0}_{n\times m}] \cdot (\mathcal{G}_{\boldsymbol{\pi}}(\mathcal{Z}) \cap_{[\boldsymbol{A}_d \; \boldsymbol{B}_d]} \mathcal{T}) = \{\boldsymbol{x} \in \mathbb{R}^n \mid \boldsymbol{x} \in \mathcal{Z}, \boldsymbol{u} = \boldsymbol{\pi}(\boldsymbol{x}), \boldsymbol{A}_d\boldsymbol{x} + \boldsymbol{B}_d\boldsymbol{u} \in \mathcal{T}\} = \mathcal{P}(\mathcal{T}) \cap \mathcal{Z}$.

(iii) The results follow from $\tilde{\mathcal{G}}_{\boldsymbol{\pi}} = \mathcal{G}_{\boldsymbol{\pi}}(\mathcal{Z})$ if $\bar{\delta} = \gamma = 0$. ■

Note that multi-step FRSs or BRSs can be computed by applying this theorem iteratively, and the size of the reduced FNNs might vary with each iteration as the reduction is performed locally.

Similar to the analysis for isolated FNNs, the over-approximated FRSs and BRSs of the NNCS (2) become exact when $\bar{\delta} = \gamma = 0$ is selected in Algorithm 2. Given an unsafe set $\mathcal{O}$ as an HZ, sufficient safety verification conditions for the NNCS (2) can be formulated as MILPs by checking whether the intersection between the computed FRSs/BRSs and the set $\mathcal{O}$ is empty [9], [10].

Remark 3: The tunable parameters $\bar{\delta}$ and γ in Algorithm 2 govern bucket tolerances and HZ representation complexity for FNNs and NNCS. In general, increasing $\bar{\delta}$ reduces FNN sizes while larger γ relaxes HZ representations, which will reduce the computation time at the expense of larger approximation errors, resulting in a sound but incomplete verification. In practice, the values of $\bar{\delta}$ and γ should be adjusted for desired approximation accuracy. When $\bar{\delta} = \gamma = 0$, the reachability analysis and safety verification results become exact. Compared to existing HZ-based methods emphasizing exact reachability analysis [9], [10], the proposed tunable approach provides more flexibility in balancing computational efficiency and approximation accuracy. This tunability offers a powerful tool for the HZ-based method to handle NNCS whose HZ set complexity arises from both the size of FNNs and the error propagation during system evolution.

Remark 4: State-of-the-art NN verifiers like α, β-CROWN [18] and Marabou [19] offer efficient analysis for standalone NNs but introduce conservatism when directly applied to NNCS. Compared with other set representations (e.g., zonotopes [20], constrained zonotopes [3], and polynotopes [21]) that have also been used for NNCS reachability analysis, HZs can represent arbitrary non-convex and disconnected sets with flat faces and their set operations can be efficiently computed using simple identities. These features make HZ better suited for investigating NNCS reachability problems that usually involve non-convex polytopic sets.

The method of this letter can be potentially extended to NNCS with nonlinear plant and other types of activation functions by incorporating the nonlinear reachability algorithms in [15], [22]. The computational efficiency of the proposed method may be also further improved by leveraging the linear bound propagation techniques in [18].

V. Simulation Results

The following example demonstrates the computation of FRSs and BRSs using Theorems 1 and 2. Results are obtained in MATLAB R2022a on a desktop with an Intel Core i9-12900k CPU and 32GB of RAM.

Consider the following linearized ground robot model: $\boldsymbol{x}(t+1) = \begin{bmatrix} \boldsymbol{I}_2 & \boldsymbol{I}_2 \\ \mathbf{0} & \boldsymbol{I}_2 \end{bmatrix}\boldsymbol{x}(t) + \begin{bmatrix} 0.5 \cdot \boldsymbol{I}_2 \\ \boldsymbol{I}_2 \end{bmatrix}\boldsymbol{u}(t)$, where the state $\boldsymbol{x} = [x, y, \dot{x}, \dot{y}]^\top$ consists of $x - y$ position and velocity, and the input $\boldsymbol{u}(t) = \boldsymbol{\pi}(\boldsymbol{x}(t))$ is a ReLU-activated FNN with 100 neurons trained from a dataset generated by an MPC controller.

First, we compute FRSs $\tilde{\mathcal{R}}_1(\mathcal{X}_0), \ldots, \tilde{\mathcal{R}}_5(\mathcal{X}_0)$ using Theorem 2 iteratively with a given initial set $\mathcal{X}_0 = [[2.5, 3]] \times [[2.5, 3]] \times [[-0.3, -0.1]] \times [[-0.3, -0.1]]$ and tunable parameters $\gamma = 0$ and $\bar{\delta} \in \{0, 0.04d_{max}, 0.06d_{max}\}$, where d_{max} is the largest range of neurons in each layer. The computation takes 47.36 sec for $\bar{\delta} = 0$, 62.78 sec for $\bar{\delta} = 0.04d_{\max}$, and 70.15 sec for $\bar{\delta} = 0.06d_{\max}$. When $\bar{\delta} = 0.04d_{max}$, the reduced FNNs at $t = 1, \ldots, 5$ have 4, 3, 7, 13, and 9 neurons, respectively; when $\bar{\delta} = 0.06d_{max}$, the reduced FNNs at $t = 1, \ldots, 5$ have 1, 5, 5, 10, and 11 neurons, respectively. We compare our method with NNV [5], ReachLP and ReachLP-Partition with the default Greedy Sim-Guided partition [2]. The projections of the FRSs onto the $x - y$ plane are shown in Fig. 3. It can be seen that all FRSs in our method with $\bar{\delta} = 0$ coincide with the exact FRSs computed by [10]. For

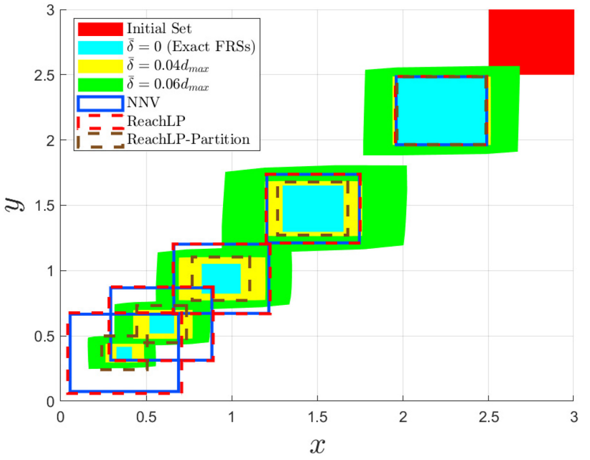


Fig. 3. The 5-step FRSs (projected on the $x - y$ plane) that are computed via the proposed method, NNV, ReachLP, and ReachLP-Partition.

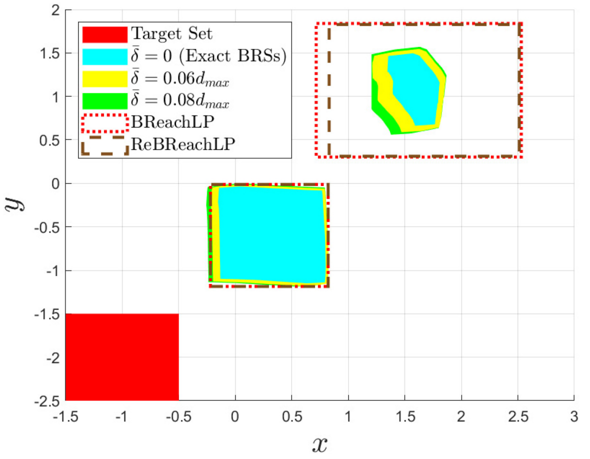


Fig. 4. The 2-step BRSs (projected on the $x - y$ plane) that are computed via the proposed method, BReachLP and ReBReachLP.

two other values of $\bar{\delta}$, the FRSs computed using our method over-approximate the exact FRSs, where a larger $\bar{\delta}$ results in a larger FRS as expected. Moreover, FRSs computed using our method do not aggravate the conservativeness as t increases, while those computed by NNV and ReachLP tend to be more conservative with an increasing t.

Next, we compute two-step BRSs $\tilde{\mathcal{P}}_1(\mathcal{T})$ and $\tilde{\mathcal{P}}_2(\mathcal{T})$ with a given target set $\mathcal{T} = [[-1.5, -0.5]] \times [[-2.5, -1.5]] \times [[-1.1, -0.9]] \times [[-1.1, -0.9]]$ and tunable parameters $\gamma = 0.1$ and $\bar{\delta} \in \{0, 0.06d_{max}, 0.08d_{max}\}$ using Theorem 2 iteratively. A prior set enclosing the BRS is chosen as the input set of Algorithm 2. For comparison, we use BReachLP and ReBReachLP [4] to compute over-approximations of the BRSs. Fig. 4 shows the projections of the computed BRSs onto the $x - y$ plane. Similar to the FRS case, our calculated BRSs become more conservative with a larger $\bar{\delta}$. Nevertheless, they are more accurate than the BRSs computed by BReachLP and ReBReachLP at $t = 2$.

VI. Conclusion

We introduced a tunable HZ-based approach that integrates an optimization-based FNN reduction technique with an activation pattern-based HZ propagation of FNNs. With two tunable parameters, our method can generate HZ over-approximations for the BRSs and FRSs of NNCS, allowing for a flexible balance between set complexity and approximation accuracy. Moreover, the proposed approach was shown to revert to exact reachability analysis as a special case.

References

[1] C. Huang, J. Fan, W. Li, X. Chen, and Q. Zhu, "ReachNN: Reachability analysis of neural-network controlled systems," *ACM Trans. Embed. Comput. Syst.*, vol. 18, no. 5, pp. 1–22, 2019.

[2] M. Everett, G. Habibi, C. Sun, and J. P. How, "Reachability analysis of neural feedback loops," *IEEE Access*, vol. 9, pp. 163938–163953, 2021.

[3] Y. Zhang and X. Xu, "Safety verification of neural feedback systems based on constrained zonotopes," in *Proc. IEEE Conf. Decision Control*, 2022, pp. 2737–2744.

[4] N. Rober, M. Everett, and J. P. How, "Backward reachability analysis for neural feedback loops," in *Proc. IEEE 61st Conf. Decision Control*, 2022, pp. 2897–2904.

[5] D. M. Lopez, S. W. Choi, H.-D. Tran, and T. T. Johnson, "NNV 2.0: The neural network verification tool," in *Proc. Int. Conf. Comput.-Aided Verificat.*, 2023, pp. 397–412.

[6] C. Brix, S. Bak, C. Liu, and T. T. Johnson, "The fourth international verification of neural networks competition (VNN-COMP 2023): Summary and results," 2023, *arXiv:2312.16760*.

[7] D. Manzanas Lopez, M. Althoff, M. Forets, T. T. Johnson, T. Ladner, and C. Schilling, "ARCH-COMP23 category report: Artificial Intelligence and Neural Network Control Systems (AINNCS) for continuous and hybrid systems plants," in *EPiC Series in Computing*, Manchester, U.K.: EasyChair, 2023.

[8] J. A. Siefert, T. J. Bird, J. P. Koeln, N. Jain, and H. C. Pangborn, "Successor sets of discrete-time nonlinear systems using hybrid zonotopes," in *Proc. Am. Control Conf.*, 2023, pp. 1383–1389.

[9] Y. Zhang and X. Xu, "Reachability analysis and safety verification of neural feedback systems via hybrid zonotopes," in *Proc. Am. Control Conf.*, 2023, pp. 1915–1921.

[10] Y. Zhang, H. Zhang, and X. Xu, "Backward reachability analysis of neural feedback systems using hybrid zonotopes," *IEEE Control Syst. Lett.*, vol. 7, pp. 2779–2784, 2023.

[11] T. J. Bird, H. C. Pangborn, N. Jain, and J. P. Koeln, "Hybrid zonotopes: A new set representation for reachability analysis of mixed logical dynamical systems," *Automatica*, vol. 154, Aug. 2023, Art. no. 111107.

[12] T. J. Bird and N. Jain, "Unions and complements of hybrid zonotopes," *IEEE Control Syst. Lett.*, vol. 6, pp. 1778–1783, 2021.

[13] T. Ladner and M. Althoff, "Specification-driven neural network reduction for scalable formal verification," 2023, *arXiv:2305.01932*.

[14] Y. Y. Elboher, J. Gottschlich, and G. Katz, "An abstraction-based framework for neural network verification," in *Proc. Int. Conf. Comput.-Aided Verificat.*, 2020, pp. 43–65.

[15] H. Zhang, Y. Zhang, and X. Xu, "Hybrid zonotope-based backward reachability analysis for neural feedback systems with nonlinear system models," 2024, *arXiv:2310.06921*.

[16] T. J. Bird, "Hybrid zonotopes: A mixed-integer set representation for the analysis of hybrid systems," Ph.D. dissertation, Dept. Mech. Eng., Purdue Univ. Grad. School, West Lafayette, IN, USA, 2022.

[17] B. Hanin and D. Rolnick, "Deep ReLU networks have surprisingly few activation patterns," in *Proc. 33rd Conf. Neural Inf. Process. Syst.*, 2019, pp. 1–10.

[18] S. Wang et al., "Beta-CROWN: Efficient bound propagation with per-neuron split constraints for neural network robustness verification," in *Proc. Conf. Neural Inf. Process. Syst.*, 2021, pp. 29909–29921.

[19] G. Katz et al., "The marabou framework for verification and analysis of deep neural networks," in *Proc. 31st Int. Conf. Comput.-Aided Verificat.*, 2019, pp. 443–452.

[20] G. Singh, T. Gehr, M. Mirman, M. Püschel, and M. Vechev, "Fast and effective robustness certification," in *Proc. 32nd Conf. Neural Inf. Process. Syst.*, 2018, pp. 1–12.

[21] C. Trapiello, C. Combastel, and A. Zolghadri, "Verification of neural network control systems using symbolic zonotopes and polynotopes," 2023, *arXiv:2306.14619*.

[22] J. A. Siefert, T. J. Bird, J. P. Koeln, N. Jain, and H. C. Pangborn, "Reachability analysis of nonlinear systems using hybrid zonotopes and functional decomposition," 2023, *arXiv:2304.06827*.