Neural Hybrid Equations: Models, Basic Properties, and Approximation Results

Daniel E. Ochoa and Ricardo G. Sanfelice

***Abstract*— This paper establishes approximation results for neural network approximators of hybrid dynamical systems. We propose neural hybrid equations, which are hybrid systems that use neural networks as the flow map and the jump map while preserving the hybrid structure. Our main result proves that solutions to neural hybrid equations, with Lipschitz non-affine activation functions, can approximate solutions of nominal hybrid systems with Lipschitz vector fields with arbitrary precision on compact time domains. For the subclass of neural hybrid equations with rectifying linear unit (ReLU) activation functions, we establish $O(1/N)$ convergence rates where N is the number of neurons. We illustrate our results with a numerical example.**

I. Introduction

Hybrid dynamical systems, coupling continuous evolution with discrete state transitions, arise across robotics, power systems, and aerospace control [1], [2]. Challenges in high-dimensional settings or when system parameters and switching logic are unknown motivate neural network methods that approximate continuous and discrete dynamics simultaneously. Universal approximation theorems provide the theoretical foundation for these approaches. Early work established that single-hidden-layer networks with sigmoid activations can approximate any continuous function on a compact set [3]. These results were later extended to continuous non-polynomial activations [4] and, more recently, to deep fixed-width neural networks using a larger class of activation functions [5]. Building on [3], [4], approximation results were extended to continuous-time dynamical systems, showing that solutions to recurrent neural networks (RNNs) can approximate solutions to nonlinear dynamical systems when using sigmoid activations [6]. Recent work extended these guarantees to hybrid systems using hybrid recurrent neural networks (HyRNNs), establishing approximation results over bounded hybrid time domains with activation functions that only vanish at zero [7].

The theoretical guarantees obtained using RNNs come with practical constraints, as this architecture faces challenges that can undermine the approximation results. For instance, discrete-time RNNs suffer from vanishing gradient problems that prevent effective learning of long-term dependencies and numerical instability when trained on long sequences of data [8], [9]. These limitations motivated renewed interest in neural ordinary differential equations (neural ODEs) [10], [11], with earlier mentions of such architectures appearing in prior work [12]. Neural ODEs parameterize the vector field of an ODE with a neural network, separating the model specification from the number of network evaluations, which is determined adaptively by the ODE solver rather than fixed by the architecture [11].

While recent approaches like the neural hybrid automata of [13] bring neural ODE ideas to study a subclass of hybrid systems by proposing a mode-conditioned neural ODE and a training procedure for learning flow and jump maps from data, rigorous results guaranteeing approximation of solutions in compact time domains such as those established in [6], [7] are not available. This paper fills this gap and provides theoretical foundations for the use of neural ODEs in hybrid dynamical systems. Our contributions are as follows: 1) we establish explicit closed-form bounds on the approximation error between solutions of a hybrid system and of a system with approximately the same flow and jump maps, improving over the implicit recursive formula of [7, Prop. 3.3.]; 2) for a given nominal hybrid system, we prove the existence of a neural hybrid equation whose solutions approximate solutions to the nominal system on compact sets of initial conditions and compact hybrid-time domains; 3) for networks with ReLU activation functions, we derive $O(1/N)$ mean-square approximation error rates, integrated over compact sets of initial conditions, where N is an upper bound on the total number of neurons. Due to space constraints, proofs will be presented elsewhere.

II. Preliminaries

A. Notation

The set of natural numbers including 0 is denoted by $\mathbb{N}$, with $\mathbb{N}_{\geq 1} \coloneqq \mathbb{N} \setminus \{0\}$. The set of real numbers is denoted by $\mathbb{R}$, with $\mathbb{R}_{\geq 0} \coloneqq \{x \in \mathbb{R} : x \geq 0\}$. We use $|x|$ to denote the Euclidean norm of the vector $x \in \mathbb{R}^n$ and use $|x|_K \coloneqq \inf_{s\in K}|x - s|$ to denote the distance from x to a closed set $K \subset \mathbb{R}^n$. Given two sets $A, B \subset \mathbb{R}^n$ closed and nonempty, the Hausdorff distance between A and B is defined as $d_H(A, B) \coloneqq \max\{\sup_{x\in B} |x|_A, \sup_{x\in A} |x|_B\}$. The closure of a set $X \subset \mathbb{R}^n$ is denoted by $\overline{X}$ and its interior is denoted by $\operatorname{int} X$. To simplify notation, for two vectors $u, v \in \mathbb{R}^n$, we write $(u, v) = [u^\top, v^\top]^\top$ to denote their concatenation. We use $x + \varepsilon\mathbb{B}$ for the closed ball centered at $x \in \mathbb{R}^n$ with radius $\varepsilon > 0$. The Euclidean inner product between two vectors a and b in $\mathbb{R}^n$ is denoted by $a^\top b$. The

D. E. Ochoa and R. G. Sanfelice are with the Department of Electrical and Computer Engineering, University of California, Santa Cruz, 95064, USA. Corresponding Author: Daniel E. Ochoa (dochoatamayo@ucsc.edu).
Research partially supported by NSF Grants no. CNS-2039054 and CNS-2111688, by AFOSR Grants nos. FA9550-23-1-0145, FA9550-23-1-0313, and FA9550-23-1-0678, by UC Alianza MX Grant no. SRPUCMX24-01, by AFRL Grant nos. FA8651-22-1-0017 and FA8651-23-1-0004, by ARO Grant no. W911NF-20-1-0253, by DoD Grant no. W911NF-23-1-0158.

vector of ones in $\mathbb{R}^n$ is denoted by $\mathbf{1}_n \in \mathbb{R}^n$. Given $\tau \in \mathbb{R}_{\geq 0}$, we let $\lfloor \tau \rfloor := \max\{l \in \mathbb{N} : l \leq \tau\}$.

We use $\mathrm{id} : \mathbb{R}^n \to \mathbb{R}^n$ for the identity function defined by $\mathrm{id}(x) = x$ for all $x \in \mathbb{R}^n$. For functions $f : Y \to Z$ and $g : X \to Y$, their composition $f \circ g : X \to Z$ is defined by $f \circ g\,(x) := f(g(x))$ for all $x \in X$. Given a finite collection of functions $\{h_l\}_{l=a}^{b}$ with appropriate domains and codomains, and $a, b \in \mathbb{N}$, we define the left-composition operator as $\bigcirc_{l=a}^{b} h_l := h_b \circ h_{b-1} \circ \cdots \circ h_a$, with the convention that $\bigcirc_{l=a}^{a} h_l = h_a$ and $\bigcirc_{l=a}^{b} h_l = \mathrm{id}$ when $a > b$. For functions $f : \mathbb{R}^n \to \mathbb{R}^m$ and $g : \mathbb{R}^p \to \mathbb{R}^q$, their Cartesian product $f \times g : \mathbb{R}^n \times \mathbb{R}^p \to \mathbb{R}^m \times \mathbb{R}^q$ is defined by $f \times g\,(x, y) := (f(x), g(y))$. A function $f : \mathbb{R}^n \to \mathbb{R}^n$ is said to be Lipschitz relative to a set $C \subset \mathbb{R}^n$ if there exists $L_f > 0$ such that $|f(x) - f(y)| \leq L_f |x - y|$ for all $x, y \in C$. Given $A \subset \mathbb{R}^n$, the Sobolev space $\mathcal{W}^k(A; \mathbb{R}^m)$ consists of functions $f : A \to \mathbb{R}^m$ that are k-times differentiable with square-integrable derivatives up to order k. The local Sobolev space $\mathcal{W}^k_{\mathrm{loc}}(\mathbb{R}^n; \mathbb{R}^m)$ consists of functions that belong to $\mathcal{W}^k(K; \mathbb{R}^m)$ on every compact set $K \subset \mathbb{R}^n$. For a set $\Omega \subset \mathbb{R}^n$, we denote by $\mathfrak{B}(\Omega)$ the Borel σ-algebra generated by the open sets of the subspace topology on Ω.

B. Neural Networks

In this paper, we use deep neural networks to approximate the flow map and jump map of a hybrid system.

Definition 2.1 (Deep Neural Network)**.** *Given $\overline{\sigma} : \mathbb{R} \to \mathbb{R}$, called scalar activation function, a* deep neural network *of depth $d \in \mathbb{N}_{\geq 1}$, input and output dimension $n \in \mathbb{N}_{\geq 1}$, and hidden widths $w = (w_1, \ldots, w_d) \in \mathbb{N}^d_{\geq 1}$ is a function $\hat{f} : \mathbb{R}^n \to \mathbb{R}^n$ of the formx3 $\hat{f} = \Lambda_d \circ \bigcirc_{l=1}^{d} (\sigma_l \circ \Lambda_{l-1})$, where $w_0 := n$, $w_{d+1} := n$, the maps $\{\Lambda_l\}_{l=0}^{d}$ are affine layers with $\Lambda_l(z) = W_l z + b_l$, $W_l \in \mathbb{R}^{w_{l+1} \times w_l}$, $b_l \in \mathbb{R}^{w_{l+1}}$, and the maps $\{\sigma_l\}_{l=1}^{d}$ apply $\overline{\sigma}$ component-wise, with $\sigma_l(y) := (\overline{\sigma}(y_1), \overline{\sigma}(y_2), \ldots, \overline{\sigma}(y_{w_l}))$ for all $y \in \mathbb{R}^{w_l}$. We denote the class of all such networks by $\mathcal{NN}^{\overline{\sigma}}_{n,w,d}$.*

For a broad class of activation functions, networks of sufficient depth can approximate any continuous function arbitrarily well on compact sets.

Proposition 2.2 (Universal Approximation [5, Thm. 3.2])**.** *Let $\overline{\sigma} : \mathbb{R} \to \mathbb{R}$ be a nonaffine[1] continuous function which is continuously differentiable at some $x_0 \in \mathbb{R}$, with nonzero derivative at that point. Then, for each continuous function $h : \mathbb{R}^n \to \mathbb{R}^n$, each compact set $K \subset \mathbb{R}^n$, and all $\varepsilon > 0$, there exist $d \in \mathbb{N}_{\geq 1}$, hidden widths $w = (2n + 2)\mathbf{1}_d$, and a neural network $\hat{h} \in \mathcal{NN}^{\overline{\sigma}}_{n,w,d}$ such that*

$$|h(x) - \hat{h}(x)| \leq \varepsilon \quad \forall x \in K.$$

Proposition 2.2 guarantees existence of a neural network approximator but gives no information on how the network size relates to the achieved error. Existing quantitative rates require additional regularity of the function to be approximated and restrict the activation to the rectified linear unit (ReLU), defined by $\overline{\sigma}(x) := \max\{0, x\}$ for all $x \in \mathbb{R}$.

Proposition 2.3 (Approximation Rates [14, Thm. 1], [15, Cor. 2.13])**.** *Let $\overline{\sigma}$ be the ReLU activation function and $d \in \mathbb{N}_{\geq 1}$ be a fixed depth. Then, for each compact set $K \subset \mathbb{R}^n$ and each function $h \in \mathcal{W}^k_{\mathrm{loc}}(\mathbb{R}^n; \mathbb{R}^n)$ with $k > 1/d + n/2$, there exists $c > 0$ such that, for each $N \in \mathbb{N}_{>0}$, there exist hidden widths $w \in \mathbb{N}^d_{\geq 1}$ satisfying $\mathbf{1}_d^\top w \leq N$ and a neural network $\hat{h} \in \mathcal{NN}^{\overline{\sigma}}_{n,w,d}$ such that*

$$\int_K |h(x) - \hat{h}(x)|^2 \, dx \leq \frac{c\,d}{N}.$$

III. Approximation of Hybrid Dynamical Systems via Neural Hybrid Equations

We study the approximation of hybrid dynamical systems described by the hybrid equation

$$\mathcal{H} : \begin{cases} \dot{x} = f(x) & x \in C \\ x^+ = g(x) & x \in D \end{cases}, \tag{1}$$

where $x \in \mathbb{R}^n$ is the state of the system, $C \subset \mathbb{R}^n$ is the flow set, $f : C \to \mathbb{R}^n$ is the flow map, $D \subset \mathbb{R}^n$ is the jump set, and $g : D \to \mathbb{R}^n$ is the jump map. Solutions to hybrid systems of this form are defined on hybrid time domains. A hybrid equation with the data as above will be represented by the notation $\mathcal{H} = (C, f, D, g)$.

Definition 3.1 (Hybrid Time Domain)**.** *A compact hybrid time domain is a subset of $\mathbb{R}_{\geq 0} \times \mathbb{N}$ of the form $\bigcup_{j=0}^{J-1} [t_j, t_{j+1}] \times \{j\}$ for a sequence $0 = t_0 \leq t_1 \leq t_2 \leq \ldots$ with J finite. A set $E \subset \mathbb{R}_{\geq 0} \times \mathbb{N}$ is a hybrid time domain if it is the union of a nondecreasing sequence of compact hybrid time domains, namely, E is the union of compact hybrid time domains E_j with the property that $E_0 \subset E_1 \subset E_2 \subset \ldots \subset E_j \ldots$. Given a hybrid time domain E, we let $\sup_t E := \sup\{t : \exists j \in \mathbb{N},\ (t, j) \in E\}$, and $\sup_j E := \sup\{j : \exists t \in \mathbb{R}_{\geq 0}, (t, j) \in E\}$.*

Definition 3.2 (Solution to $\mathcal{H}$)**.** *A solution to the hybrid dynamical system $\mathcal{H}$ in (1) is a map $\phi : \mathrm{dom}\,\phi \to \mathbb{R}^n$, where $\mathrm{dom}\,\phi$ is a hybrid time domain, $t \mapsto \phi(t, j)$ is locally absolutely continuous on each interval $I^j := \{t : (t, j) \in \mathrm{dom}\,\phi\}$ for each j, and the following conditions hold:*

(S1) $\phi(0, 0) \in \overline{C} \cup D$;

(S2) for each interval I^j with nonempty interior, $\frac{d\phi}{dt}(t, j) = f(\phi(t, j))$ for almost all $t \in I^j$, and $\phi(t, j) \in C$ for all $t \in \mathrm{int}\, I^j$;

(S3) for each $(t, j) \in \mathrm{dom}\,\phi$ such that $(t, j + 1) \in \mathrm{dom}\,\phi$, $\phi(t, j) \in D$ and $\phi(t, j + 1) = g(\phi(t, j))$.

A solution ϕ to $\mathcal{H}$ is said to be maximal *if it cannot be further extended. The set of maximal solutions to $\mathcal{H}$ from a set $K \subset \mathbb{R}^n$ is denoted by $\mathcal{S}_{\mathcal{H}}(K)$.*

Definition 3.3 (Truncation of a solution)**.** *Given $\tau \geq 0$ and a solution ϕ to a hybrid system $\mathcal{H}$, the truncation of ϕ up to time τ is a function $\phi|_{\leq \tau} : \mathrm{dom}\,\phi|_{\leq \tau} \to \mathbb{R}^n$ with $\mathrm{dom}\,\phi\,|_{\leq \tau} := \{(t, j) \in \mathrm{dom}\,\phi : t + j \leq \tau\}$ and $\phi|_{\leq \tau}(t, j) := \phi(t, j)$*

[1] A function $\overline{\sigma} : \mathbb{R} \to \mathbb{R}$ is *nonaffine* if there do not exist $a, b \in \mathbb{R}$ such that $\overline{\sigma}(x) = ax + b$ for all $x \in \mathbb{R}$.

for each $(t,j) \in \operatorname{dom}\phi|_{\leq\tau}$. *The set of solutions to the hybrid system* $\mathcal{H}$ *from* $\phi_0 \in \overline{C} \cup D$ *that evolve up to time* τ *is denoted by* $\mathcal{S}_{\mathcal{H}}^{\leq\tau}(\phi_0) := \{\phi|_{\leq\tau} : \phi \in \mathcal{S}_{\mathcal{H}}(\phi_0)\}$.

In this paper, we often express solutions to hybrid equations of the form in (1) via the following representation:

Lemma 3.4. *Let* $\mathcal{H} = (C, f, D, g)$ *be a hybrid equation. Then, for any* $\phi_0 \in \overline{C} \cup D$, *and any solution* $\phi \in \mathcal{S}_{\mathcal{H}}(\phi_0)$, *it follows that*

$$\phi(t,j) = \phi_0 + \int_0^t f \circ \phi\,(s, j(s))ds + \sum_{i=0}^{j-1} (g - \mathrm{id}) \circ \phi\,(t_{i+1}, i)$$

for all $(t,j) \in \operatorname{dom}\phi$, *where* $j(s) := \max\{i : (s,i) \in \operatorname{dom}\phi\}$ *for all* $s \in [0,t]$, *and* $\{t_i\}_{i=0}^{j+1}$ *are such that* $\operatorname{dom}\phi \cap \big([0,t] \times \{0,1,\ldots,j\}\big) = \bigcup_{i=0}^{j}\big([t_i, t_{i+1}] \times \{i\}\big)$ *with* $t_{j+1} := t$.

A. Neural Hybrid Equations

In this paper, we approximate solutions to $\mathcal{H}$ in (1) by embedding neural networks within a hybrid system structure, using the class $\mathcal{NN}_{n,w,d}^{\overline{\sigma}}$ of Definition 2.1 as approximators of the flow and jump maps. The proposed approximator of $\mathcal{H}$ is denoted $\mathcal{H}_a$ and has hybrid dynamics given by

$$\mathcal{H}_a : \begin{cases} \dot{\eta} = \hat{f}(\eta) & \text{when } \mathcal{H} \text{ flows} \\ \eta^+ = \hat{g}(\eta) & \text{when } \mathcal{H} \text{ jumps} \end{cases}, \tag{2}$$

where $\eta \in \mathbb{R}^n$ is the neural network state and $\hat{f} \in \mathcal{NN}_{n,w_f,d_f}^{\overline{\sigma}_f}$, $\hat{g} \in \mathcal{NN}_{n,w_g,d_g}^{\overline{\sigma}_g}$ for some $\overline{\sigma}_f, \overline{\sigma}_g : \mathbb{R} \to \mathbb{R}$, $d_f, d_g \in \mathbb{N}_{\geq 1}$, and $w_f \in \mathbb{N}_{\geq 1}^{d_f}$, $w_g \in \mathbb{N}_{\geq 1}^{d_g}$. We refer to $\mathcal{H}_a$ as a *neural hybrid equation with activation functions* $\overline{\sigma}_f, \overline{\sigma}_g$, *depths* (d_f, d_g), *and hidden widths* (w_f, w_g), with the weight matrices and bias vectors of $\hat{f}$ and $\hat{g}$ understood as part of the architecture.

Remark 3.5 (On Deep ResNets)**.** *Applying the forward Euler discretization scheme to the flow dynamics of the neural hybrid equation* $\mathcal{H}_a$ *with step size* h *yields the update rule* $\eta_{k+1} = \eta_k + h\,\hat{f}(\eta_k)$, *which is a residual neural network (ResNet) block with residual function* $\hat{f}$ *implemented as a deep sub-network [16]. This ResNet structure emerges naturally when coupling neural hybrid equations with standard numerical solvers, providing an interpretation of neural hybrid equations as a mechanism to specify deep ResNets, where the residual architecture is not designed directly but follows from a discretization scheme applied to the continuous dynamics.*

Similar to approximators for continuous-time and discrete-time systems, it leads to an interconnection

$$\mathcal{H}_{\text{int}} : \left\{ \begin{array}{rl} \dot{x} &= f(x) \\ \dot{\eta} &= \hat{f}(\eta) \end{array} \right\} (x,\eta) \in C \times \mathbb{R}^n \quad \left\{ \begin{array}{rl} x^+ &= g(x) \\ \eta^+ &= \hat{g}(\eta) \end{array} \right\} (x,\eta) \in D \times \mathbb{R}^n \tag{3}$$

For this interconnection, existence of solutions depends on the data specifying $\mathcal{H}$ and $\mathcal{H}_a$. The result below characterizes the solutions to $\mathcal{H}_{\text{int}}$ under the assumption that the activation function satisfies a Lipschitz condition.

Lemma 3.6 (Existence of Solutions)**.** *Assume* f *is Lipschitz relative to* C *and* $\overline{\sigma}_f$ *is Lipschitz. Then, for each* $\phi \in \mathcal{S}_{\mathcal{H}}(\phi_0)$, *there exists a unique* $\psi \in \mathcal{S}_{\mathcal{H}_a}(\phi_0)$ *such that* $\operatorname{dom}\phi = \operatorname{dom}\psi$, *and* (ϕ, ψ) *is a solution to* $\mathcal{H}_{int}$ *with* $(\phi(0,0), \psi(0,0)) = (\phi_0, \phi_0)$.

Remark 3.7 (Implications of the Interconnection Structure)**.** *This paper assumes the flow and jump sets,* C *and* D *respectively, of the hybrid equation* $\mathcal{H} = (C, f, D, g)$ *are known. This enables the construction of* $\mathcal{H}_a$ *through the interconnection structure in* (3), *which enforces synchronization of flows and jumps between* $\mathcal{H}_a$ *and* $\mathcal{H}$, *and prevents the emergence of* hybrid peaking behavior *[17].*

To show that, under appropriate conditions, solutions to $\mathcal{H}_a$ in (2) approximate the solutions to $\mathcal{H}$ in (1), we first present a general result that bounds the distance between solutions of two hybrid systems whose flow and jump maps are uniformly close on compact sets. This result is not specific to neural network approximators. A similar bound appears in [7, Prop. 3.3], where the error is expressed implicitly via a recursive formula. The result below provides an explicit closed-form bound in terms of the Lipschitz constants of f and g and the approximation errors ε_f and ε_g.

Proposition 3.8 (Hybrid Tube-Bound Over Compact Time Domains)**.** *Given a hybrid system* $\mathcal{H} = (C, f, D, g)$ *and a compact set* $K \subset \mathbb{R}^n$, *suppose that*
(i) f *is Lipschitz relative to* $C \cap K$ *with constant* L_f;
(ii) g *is Lipschitz relative to* $D \cap K$ *with constant* L_g.
Suppose there exist $\varepsilon_f, \varepsilon_g > 0$ *and* $\hat{f}, \hat{g} : \mathbb{R}^n \to \mathbb{R}^n$ *such that*

$$|f(x) - \hat{f}(x)| \leq \varepsilon_f \quad \forall x \in C \cap K; \tag{4a}$$
$$|g(x) - \hat{g}(x)| \leq \varepsilon_g \quad \forall x \in D \cap K. \tag{4b}$$

Then, for each solution ϕ *to* $\mathcal{H}$, *each solution* ψ *to*

$$\hat{\mathcal{H}} : \begin{cases} \dot{\hat{x}} = \hat{f}(\hat{x}) & \text{when } \mathcal{H} \text{ flows} \\ \hat{x}^+ = \hat{g}(\hat{x}) & \text{when } \mathcal{H} \text{ jumps} \end{cases}, \tag{5}$$

and each $\tau > 0$ *such that* $\operatorname{rge}\phi|_{\leq\tau} \subset K$ *and* $\operatorname{rge}\psi|_{\leq\tau} \subset K$,

$$\begin{aligned} |\phi(t,j) - \psi(t,j)| \leq\ & |\phi(0,0) - \psi(0,0)| \exp\left(L_f t\right)(1+L_g)^j \\ & + \varepsilon_f \int_0^t \exp\left(L_f(t-s)\right)(1+L_g)^{j-i(s)}\, ds \\ & + \varepsilon_g \sum_{i=0}^{j-1} \exp\left(L_f(t-t_{i+1})\right)(1+L_g)^{j-(i+1)} \end{aligned} \tag{6}$$

for all $(t,j) \in \operatorname{dom}\phi|_{\leq\tau}$. *Additionally, when* $\phi(0,0) = \psi(0,0)$, *for all* $(t,j) \in \operatorname{dom}\phi|_{\leq\tau}$, *it follows that*

$$|\phi(t,j) - \psi(t,j)| \leq (1+L_g)^{\lfloor\tau\rfloor}\, e^{L_f\tau}\, \tau\, \max\{\varepsilon_f, \varepsilon_g\}.$$

Proposition 3.8 bounds the worst-case deviation between solutions of two hybrid systems whose flow and jump maps are uniformly close on a compact set. To leverage this bound for approximation of solutions via neural hybrid equations, we use the following assumption:

Assumption 3.9 (A Class of Activation Functions)**.** *Given $\overline{\sigma} : \mathbb{R} \to \mathbb{R}$, suppose $\overline{\sigma}$ is Lipschitz, nonaffine, and continuously differentiable at some $x_0 \in \mathbb{R}$ with $\overline{\sigma}'(x_0) \neq 0$.*

Under this condition, by combining Propositions 2.2 and 3.8 with a compactness argument for reachable sets [1, Lem. 6.16(b)] we obtain the following result.

Theorem 3.10 (Universal Approximation via Neural Hybrid Equations)**.** *Given a hybrid system $\mathcal{H} = (C, f, D, g)$ with $C \subset \mathbb{R}^n$ and $D \subset \mathbb{R}^n$ closed, suppose that*
(i) f is Lipschitz relative to C;
(ii) g is Lipschitz relative to D.
Then, for each $\tau > 0$, each compact set $K \subset \mathbb{R}^n$, each $\varepsilon > 0$, and each pair of activation functions $\overline{\sigma}_f, \overline{\sigma}_g$ satisfying Assumption 3.9, there exist $d_f, d_g \in \mathbb{N}_{\geq 1}$ and a neural hybrid equation $\mathcal{H}_a$ as in (2), with activation functions $\overline{\sigma}_f, \overline{\sigma}_g$, depths (d_f, d_g), and hidden widths $w_f = (2n+2)\mathbf{1}_{d_f}$, $w_g = (2n+2)\mathbf{1}_{d_g}$, such that for each $\phi \in \mathcal{S}^{\leq\tau}_{\mathcal{H}}(K)$ there exists $\psi \in \mathcal{S}^{\leq\tau}_{\mathcal{H}_a}(K)$ satisfying (7).

$$|\phi(t,j) - \psi(t,j)| \leq \varepsilon \quad \forall (t,j) \in \operatorname{dom}\phi. \tag{7}$$

The neural ODE literature [10], [11] primarily employs differential equations to implicitly parameterize deep neural network architectures for learning tasks, an approach analogous to how discretization of neural hybrid equations gives rise to ResNets as discussed in Remark 3.5. Theorem 3.10 addresses the converse problem, namely, how to use neural networks to approximate solutions of dynamical systems. Our results align with recent work on universal approximation for continuous-time systems [18], extending these guarantees to hybrid dynamics and allowing for deep neural network approximations of the flow and jump maps. For hybrid systems, the recent work in [7] establishes approximation results using hybrid recurrent neural networks (HyRNNs) under the assumption that activation functions satisfy $\overline{\sigma}(x) = 0$ only if $x = 0$. This condition is satisfied by many common activation functions such as tanh, leaky ReLU, and ELU, but excludes the ReLU activation function, which is widely used in practice and for which sharp theoretical approximation results are available, including the rates of Proposition 2.3. Theorem 3.10 replaces the restriction of [7] with Assumption 3.9, which the ReLU activation function satisfies. The next section specializes to this type of activation functions and establishes explicit convergence rates under additional regularity on the nominal system.

IV. Approximation Rates Using ReLU Activation Functions

While Theorem 3.10 guarantees that solutions of a neural hybrid equation can approximate solutions of a nominal hybrid system to arbitrary pointwise precision, it gives no information on how the neural network size relates to the achieved error. This section addresses that gap by establishing $O(1/N)$ mean-square convergence rates when using neural hybrid equations with ReLU activation functions, where N upper bounds the total number of neurons across all hidden layers, and the error is measured by integrating the worst-case graphical distance between nominal and neural solutions over initial conditions.

To bound this integral, we select, for each initial condition, a solution pair to the interconnected system $\mathcal{H}_{\text{int}}$ in (3) that achieves the worst-case graphical distance, bound the graphical distance along that pair, and integrate the result over initial conditions. This selection is trivial when solutions are unique. However, when $C \cap D \neq \varnothing$ solutions to the system need not be unique even when f is Lipschitz relative to C and the activation functions of the neural hybrid equation are Lipschitz, since a solution may either flow or jump from the same initial condition. The following lemma establishes that a suitable measurable selection still exists in such a case.

Lemma 4.1 (Measurable Maximal Selection)**.** *Let $\mathcal{H} = (C, f, D, g)$ be a hybrid system with $C, D \subset \mathbb{R}^n$ closed, f Lipschitz relative to C, and g Lipschitz relative to D. Let $\hat{f}, \hat{g} : \mathbb{R}^n \to \mathbb{R}^n$ be Lipschitz continuous, and consider $\mathcal{H}_{int} = (C \times \mathbb{R}^n, f \times \hat{f}, D \times \mathbb{R}^n, g \times \hat{g})$. Then, for each $\tau > 0$ and each compact set $K \subset C \cup D$, there exists a measurable selector $\kappa : K \to \mathcal{S}^{\leq\tau}_{\mathcal{H}_{int}}(K \times K)$, written $\kappa(\phi_0) = (\kappa_{\mathcal{H}}(\phi_0), \kappa_{\mathcal{H}_a}(\phi_0))$ with $\kappa_{\mathcal{H}}(\phi_0) \in \mathcal{S}^{\leq\tau}_{\mathcal{H}}(\phi_0)$ and $\kappa_{\mathcal{H}_a}(\phi_0) \in \mathcal{S}^{\leq\tau}_{\mathcal{H}_a}(\phi_0)$ for each $\phi_0 \in K$, such that*

$$\kappa(\phi_0) \in \operatorname*{argmax}_{(\tilde{\phi},\tilde{\psi}) \in \mathcal{S}^{\leq\tau}_{\mathcal{H}_{int}}(\phi_0,\phi_0)} d_H(\operatorname{gph}\tilde{\phi}, \operatorname{gph}\tilde{\psi})^2,$$

and the map $K \ni \phi_0 \mapsto d_H(\operatorname{gph}\kappa_{\mathcal{H}}(\phi_0), \operatorname{gph}\kappa_{\mathcal{H}_a}(\phi_0))^2 \in \mathbb{R}_{\geq 0}$ is $(\mathfrak{B}(K), \mathfrak{B}(\mathbb{R}_{\geq 0}))$-measurable.

The following result combines Proposition 2.3 with the measurability of the worst-case solution pair in Lemma 4.1 to establish mean-square approximation rates for neural hybrid equations using ReLU activation functions.

Theorem 4.2. *Given a hybrid system $\mathcal{H} = (C, f, D, g)$ with $C, D \subset \mathbb{R}^n$ closed, ReLU activation functions $\overline{\sigma}_f, \overline{\sigma}_g : \mathbb{R} \to \mathbb{R}$, and depths $d_f, d_g \in \mathbb{N}_{\geq 1}$, suppose that*
(i) $f, g \in \mathcal{W}^k_{loc}(\mathbb{R}^n)$ for some $k > \max\{1/d_f, 1/d_g\} + n/2$;
(ii) f is Lipschitz relative to C;
(iii) g is bi-Lipschitz[2] relative to D.
Then, for each $\tau > 0$ and each compact set $K \subset \mathbb{R}^n$, there exists $c > 0$ such that for each $N \in \mathbb{N}_{>0}$, there exist hidden widths $w_f \in \mathbb{N}^{d_f}_{\geq 1}$ and $w_g \in \mathbb{N}^{d_g}_{\geq 1}$ with $\max\{\mathbf{1}^\top_{d_f} w_f, \mathbf{1}^\top_{d_g} w_g\} \leq N$, and a neural hybrid equation $\mathcal{H}_a$ as in (2) with $\hat{f} \in \mathcal{NN}^{\overline{\sigma}_f}_{n,w_f,d_f}$ and $\hat{g} \in \mathcal{NN}^{\overline{\sigma}_g}_{n,w_g,d_g}$, such that

$$\int_K \max_{(\phi,\psi) \in \mathcal{S}^{\leq\tau}_{\mathcal{H}_{int}}(\phi_0,\phi_0)} d_{\text{gph}}(\phi,\psi)^2 \, d\phi_0 \leq c\frac{\max\{d_f, d_g\}}{N}, \tag{8}$$

where $d_{\text{gph}}(\phi,\psi) := d_H(\operatorname{gph}\phi, \operatorname{gph}\psi)$.

Theorem 4.2 establishes $O(1/N)$ mean-square convergence of solutions to $\mathcal{H}_a$ to solutions to $\mathcal{H}$, integrated over initial conditions taken from a compact set $K \subset \mathbb{R}^n$. In

[2] A map $g : \mathbb{R}^n \to \mathbb{R}^n$ is bi-Lipschitz relative to D if there exist constants $0 < L_{g^{-1}} < L_g$ such that $L_{g^{-1}}|x-y| \leq |g(x)-g(y)| \leq L_g|x-y|$ for all $x, y \in D$. This ensures g is a homeomorphism with Lipschitz inverse.

particular, for fixed network depths d_f and d_g, any target precision is achievable by taking N large enough. Deeper networks offer more flexibility, since larger depths relax the Sobolev condition in (i), yet the constant $\max\{d_f, d_g\}$ in the rate grows accordingly, so more neurons are needed to reach the same precision.

V. Training of a Neural Hybrid Equation

We illustrate our results by training a neural hybrid equation that approximates a bouncing ball system with state $x = (x_1, x_2) \in \mathbb{R}^2$, where x_1 denotes the height and x_2 the vertical velocity, and whose dynamics are given by

$$\mathcal{H} : \begin{cases} \dot{x} = f(x) \coloneqq (x_2, -\gamma) & x \in C \\ x^+ = g(x) \coloneqq (x_1, -\lambda x_2) & x \in D \end{cases},$$

where $\gamma > 0$ is the gravitational acceleration, $\lambda \in (0, 1)$ is the coefficient of restitution, and the flow and jump sets are $C \coloneqq \{x \in \mathbb{R}^2 : x_1 \geq 0 \text{ or } x_2 \geq 0\}$ and $D \coloneqq \{x \in \mathbb{R}^2 : x_1 \leq 0 \text{ and } x_2 \leq 0\}$, respectively.

For the neural hybrid equation we fix depths $d_f = d_g = d \in \mathbb{N}_{\geq 1}$ and verify that the bouncing ball system satisfies the hypotheses of Theorems 3.10 and 4.2. The flow map f is Lipschitz relative to C with constant $L_f = 1$. The jump map g is bi-Lipschitz relative to D, satisfying $\lambda|x - \tilde{x}| \leq |g(x) - g(\tilde{x})| \leq |x - \tilde{x}|$ for all $x, \tilde{x} \in D$. Since f and g are polynomials, $f, g \in \mathcal{W}^k_{\text{loc}}(\mathbb{R}^2)$ for all $k \geq 0$, which satisfies condition (i) of Theorem 4.2 for any $d \geq 1$. Thus, the bouncing ball system satisfies the hypotheses of both theorems, and the existence of an approximating neural hybrid equation and the $O(1/N)$ mean-square error scaling follow directly from Theorems 3.10 and 4.2.

To numerically illustrate these guarantees, we consider a neural hybrid equation as in (2) with ReLU activations and widths $w_f = w_g = w\,\mathbf{1}_d$, with $w \in \mathbb{N}_{\geq 1}$. The trainable parameters of $\hat{f}$ and $\hat{g}$, namely the weight matrices $\{W^f_l\}_{l=0}^d$, $\{W^g_l\}_{l=0}^d$ and bias vectors $\{b^f_l\}_{l=0}^d$, $\{b^g_l\}_{l=0}^d$ of Definition 2.1, are vectorized and stacked into a single vector $\vartheta \in \mathbb{R}^P$, where $P = 2\,[w(n+1) + (d-1)w(w+1) + n(w+1)]$. We use $\mathcal{H}^\vartheta_a$ to denote the neural hybrid equation parameterized by ϑ, and $\mathcal{H}^\vartheta_{\text{int}}$ to denote the interconnected system obtained from $\mathcal{H}$ and $\mathcal{H}^\vartheta_a$ with the structure in (3).

Given a compact set $K \subset C \cup D$ and fixed hybrid time horizon $\tau > 0$, we train the parameter ϑ, by generating $M \in \mathbb{N}_{>0}$ solution samples $\{\phi^{(m)}\}_{m=1}^M$ with $\phi^{(m)} \in \mathcal{S}^{\leq\tau}_{\mathcal{H}}(K)$, and sample their values at continuous times during flows via adaptive numerical integration, and before and after each jump. To update ϑ, we minimize a loss function $\vartheta \mapsto \mathcal{L}(\vartheta)$ that measures the error between nominal and neural solutions along pairs $(\phi^{(m)}, \psi^{(m)}_\vartheta) \in \mathcal{S}^{\leq\tau}_{\mathcal{H}^\vartheta_{\text{int}}}(\phi^{(m)}(0,0), \phi^{(m)}(0,0))$, using a Julia-based hybrid equations solver with autodifferentiable numerical integration[3], which enables backpropagation of gradients through the flows and jumps of $\mathcal{H}^\vartheta_a$. The architecture, compact set K, and remaining hyperparameters used in the numerical example are specified next.

[3]See https://bit.ly/neuralHyEQCode for a repository with the code.

A. Numerical Example

For the numerical experiment, we set $\gamma = 9.8$, $\lambda = 0.8$, depths $(d_f, d_g) = (2, 2)$ and widths $(w_f, w_g) = (16, 16)$ for the neural hybrid equation, giving $P = 708$ trainable parameters, let $K \coloneqq [0.5, 10] \times [-3.0, 3.0]$, and fix the hybrid-time horizon $\tau = 10$. We use $M = 8$ solution samples, whose values are measured during flows via numerical integration of the interconnected system $\mathcal{H}^\vartheta_{\text{int}}$ with step size $\Delta t_{\max} = 0.1$, and before and after each jump. For each $\vartheta \in \mathbb{R}^P$ and $m \in \{1, 2, \ldots, M\}$, we define the training loss by

$$\mathcal{L}^{(m)}(\vartheta) \coloneqq \alpha \mathcal{L}^{(m)}_{\text{flow}}(\vartheta) + \beta \mathcal{L}^{(m)}_{\text{jump}}(\vartheta) + \delta|\vartheta|^2, \tag{9}$$

where $\alpha, \beta, \delta > 0$ are tunable parameters, and

$$\mathcal{L}^{(m)}_{\text{flow}}(\vartheta) \coloneqq \sum_{(t,j) \in E^{(m)}} |\phi^{(m)}(t,j) - \psi^{(m)}_\vartheta(t,j)|^2$$

$$\mathcal{L}^{(m)}_{\text{jump}}(\vartheta) \coloneqq \sum_{i=0}^{\bar{\jmath}^{(m)}-1} \Big| [\phi^{(m)}(t_{i+1}, i+1) - \phi^{(m)}(t_{i+1}, i)] - [\psi^{(m)}_\vartheta(t_{i+1}, i+1) - \psi^{(m)}_\vartheta(t_{i+1}, i)] \Big|^2.$$

where $\bar{\jmath}^{(m)} \coloneqq \sup_j \operatorname{dom} \phi^{(m)}$ for each m, and $E^{(m)} \subset \operatorname{dom} \phi^{(m)} = \bigcup_{i=0}^{\bar{\jmath}^{(m)}-1} [t_i, t_{i+1}] \times \{i\}$ denotes the set of sampled hybrid times from the nominal solution. The term $|\vartheta|^2$ in (9) is a regularization term, used to prevent the weights from growing unbounded, which would otherwise drive the Lipschitz constants of $\hat{f}$ and $\hat{g}$ large, increasing the constant c in Theorem 4.2 and loosening the $O(1/N)$ bound.

The loss $\mathcal{L}^{(m)}$ is minimized with respect to ϑ using the Adam optimizer with learning rate 5×10^{-4} over 10,000 iterations. At each block of 25 consecutive iterations, one solution from $\{\phi^{(m)}\}_{m=1}^M$ is selected at random and held fixed for gradient computation, after which a new sample is selected. This cycling procedure is implemented to prevent overfitting to individual solutions. Loss weights are set to $\alpha = 1.0$, $\beta = 2.0$, and $\delta = 0.1$.

Figure 1 shows the training progression and compares nominal and neural solutions at initial and final training iterations for the test initial condition $\phi_0 = (10, 3) \in K$, $\phi_0 \notin \{\phi^{(m)}(0,0)\}_{m=1}^M$, which was not used during training. The top row shows the hybrid solutions plotted against both continuous time t and discrete time j. In the plots against discrete time, each index j displays the values of x_1 on the j-th flow interval $[t_j, t_{j+1}]$, so the vertical line at each j reflects continuous evolution between jumps.

The bottom row of Figure 1 shows the loss evolution during training. The loss decreases and plateaus, with residual spikes visible in the plateau region. These spikes are consistent with the behavior of the Adam algorithm, which keeps a running estimate of the gradient magnitude to adapt the step size. In the plateau, this estimate decays toward small values, and when a new training sample $\phi^{(m)}$ is selected whose gradient magnitude significantly exceeds those values, the update step is transiently large before the estimate catches up [19, Sec. 2.2]. Despite these spikes, the training procedure yields a neural hybrid equation that captures the behavior

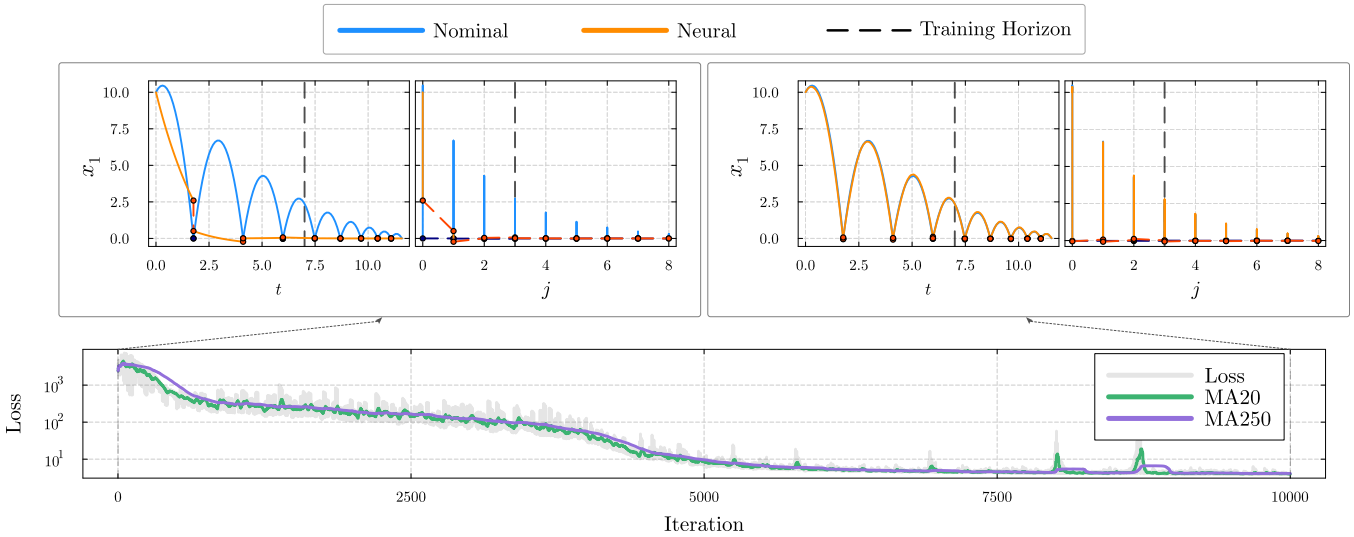

Fig. 1: Training progression of the neural hybrid system approximating a bouncing ball. *Top row:* Comparison of the state x_1 corresponding to a nominal solution ϕ and neural solution ψ starting from the test initial condition $\phi_0 = (10, 3) \in K$, which was not included in the training set. Parameters from initial (left) and final (right) training iterations are evaluated, shown in both continuous-time t and discrete-time j. The dashed vertical line marks the training horizon at $t_{\max}$ and $j_{\max}$ such that $\operatorname{dom} \psi|_{\leq t_{\max}+j_{\max}} \subset \operatorname{dom} \psi|_{\leq \tau}$ with $\tau = 10$. *Bottom:* Loss evolution during training with moving averages (MA20 and MA250) showing convergence. The neural system achieves accurate approximation of the nominal dynamics within the training domain and demonstrates reasonable extrapolation beyond it.

of the system for the unseen initial condition $\phi_0 = (10, 3)$ within and beyond the training hybrid-time horizon $\tau = 10$.

VI. Conclusions

This paper establishes approximation results for hybrid dynamical systems via neural hybrid equations. The main contributions are a closed-form hybrid tube bound, a universal approximation theorem for solutions on compact hybrid time domains, and $O(1/N)$ mean-square convergence rates when using ReLU activation functions, where N is an upper bound on the total number of neurons. These results assume the flow and jump sets C and D are known. When neural networks must also approximate C and D, temporal misalignment between flows and jumps of $\mathcal{H}_a$ and $\mathcal{H}$ can cause solutions to diverge even with perfect vector field approximation, and quantifying this effect via set-valued analysis of perturbed reachable sets is left for future work.

References

[1] R. Goebel, R. G. Sanfelice, and A. R. Teel, *Hybrid Dynamical Systems: Modeling, Stability, and Robustness*. Princeton, N.J: Princeton University Press, 2012.

[2] J. Lygeros, K. Johansson, S. Simic, Jun Zhang, and S. Sastry, "Dynamical properties of hybrid automata," *IEEE Trans. Automat. Contr.*, vol. 48, no. 1, pp. 2–17, Jan. 2003.

[3] G. Cybenko, "Approximation by superpositions of a sigmoidal function," *Math. Ctrl. Signal Sys.*, vol. 2, no. 4, pp. 303–314, Dec. 1989.

[4] A. Pinkus, "Approximation theory of the MLP model in neural networks," *Acta Numer.*, vol. 8, pp. 143–195, Jan. 1999.

[5] P. Kidger and T. Lyons, "Universal Approximation with Deep Narrow Networks," in *Proc. Thirty Third Conf. Learn. Theory*. PMLR, Jul. 2020, pp. 2306–2327.

[6] K.-i. Funahashi and Y. Nakamura, "Approximation of dynamical systems by continuous time recurrent neural networks," *Neural Networks*, vol. 6, no. 6, pp. 801–806, Jan. 1993.

[7] R. G. Sanfelice, "HyRNN: Hybrid Recurrent Neural Networks for Approximating Hybrid Dynamical Systems," in *40th Annual AAAI Conference on Artificial Intelligence*, 2026.

[8] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE Trans. Neural Netw.*, vol. 5, no. 2, pp. 157–166, Mar. 1994.

[9] R. Pascanu, T. Mikolov, and Y. Bengio, "On the difficulty of training recurrent neural networks," in *Proc. 30th Int. Conf. Mach. Learn.* PMLR, May 2013, pp. 1310–1318.

[10] R. T. Q. Chen, Y. Rubanova, J. Bettencourt, and D. K. Duvenaud, "Neural Ordinary Differential Equations," in *Adv. Neural Info Proc Sys*, vol. 31. Curran Associates, Inc., 2018.

[11] P. Kidger, J. Morrill, J. Foster, and T. Lyons, "Neural Controlled Differential Equations for Irregular Time Series," in *Adv. Neural Inf. Process. Syst.*, vol. 33. Curran Associates, Inc., 2020, pp. 6696–6707.

[12] R. Rico-Martínez, K. Krischer, I. Kevrekidis, M. Kube, and J. Hudson, "Discrete- vs. Continuous-time nonlinear signal processing of cu electrodissolution data," *Chem. Eng. Commun.*, vol. 118, no. 1, pp. 25–48, Nov. 1992.

[13] M. Poli, S. Massaroli, L. Scimeca, S. Chun, S. J. Oh, A. Yamashita, H. Asama, J. Park, and A. Garg, "Neural Hybrid Automata: Learning Dynamics With Multiple Modes and Stochastic Transitions," in *Adv. Neural Inf. Process. Syst.*, vol. 34. Curran Associates, Inc., 2021, pp. 9977–9989.

[14] G. Bresler and D. Nagaraj, "Sharp representation theorems for ReLU networks with precise dependence on depth," in *Adv. Neural Inf. Process. Syst.*, vol. 33. Curran Associates, Inc., 2020, pp. 10 697–10 706.

[15] Y. Liao and P. Ming, "Spectral Barron Space for Deep Neural Network Approximation," *SIAM Journal on Mathematics of Data Science*, vol. 7, no. 3, pp. 1053–1076, Sep. 2025.

[16] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.

[17] R. G. Sanfelice, J. B. Biemond, N. Van De Wouw, and W. H. Heemels, "An embedding approach for the design of state-feedback tracking controllers for references with jumps," *Intl J Robust & Nonlinear*, vol. 24, no. 11, pp. 1585–1608, Jul. 2014.

[18] Z. Li, K. Liu, L. Liverani, and E. Zuazua, "Universal Approximation of Dynamical Systems by Semi-Autonomous Neural ODEs and Applications," Jul. 2024, comment: 22 pages, 16 pages.

[19] T. Huang, Z. Zhu, G. Jin, L. Liu, Z. Wang, and S. Liu, "SPAM: Spike-aware Adam with Momentum Reset for Stable LLM training," in *Thirteen. Int. Conf. Learn. Represent.*, 2025.