IEEE Control Systems Letters paper presented at
2023 62nd IEEE Conference on Decision and Control (CDC)
December 13-15, 2023. Marina Bay Sands, Singapore

Universal Approximation Property of Hamiltonian Deep Neural Networks

Muhammad Zakwan, Massimiliano d'Angelo and Giancarlo Ferrari-Trecate

***Abstract*— This paper investigates the universal approximation capabilities of Hamiltonian Deep Neural Networks (HDNNs) that arise from the discretization of Hamiltonian Neural Ordinary Differential Equations. Recently, it has been shown that HDNNs enjoy, by design, non-vanishing gradients, which provide numerical stability during training. However, although HDNNs have demonstrated state-of-the-art performance in several applications, a comprehensive study to quantify their expressivity is missing. In this regard, we provide a universal approximation theorem for HDNNs and prove that a portion of the flow of HDNNs can approximate arbitrary well any continuous function over a compact domain. This result provides a solid theoretical foundation for the practical use of HDNNs.**

I. INTRODUCTION

Deep Neural Networks (DNNs) have been crucial for the success of machine learning in several real-world applications like computer vision, natural language processing, and reinforcement learning. To achieve state-of-the-art performance, a common approach in machine learning is to increase the Neural Network (NN) depth. For instance, Convolutional Neural Networks (CNNs) AlexNet [1], Visual Geometric Group (VGG) network, GoogLeNet/Inception [2], Residual Network (ResNet) [3], or recently developed transformers such as ChatGPT, contain hundreds to thousands of layers. It has been empirically demonstrated that deeper networks yield better performance than single-hidden-layer NNs for large-scale and high-dimensional problems [4], [5]. However, a rigorous characterization of the approximation capabilities of complex NNs is often missing. Moreover, the understanding of how NN architectures (depth, width, and type of activation function) achieve their empirical success is an open research problem [6].

To quantify the representational power of NNs, researchers have focused on studying their Universal Approximation Properties (UAPs), namely their ability to approximate any desired continuous function with an arbitrary accuracy. To this aim, several UAP results for various classes of NNs have been proposed. The UAP of Shallow NNs (SNNs), *i.e.* with single hidden layer has proven in the seminal works of Cybenko [7] and Hornik [8]. Exploiting the latter arguments, researchers have provided several results on UAPs for DNNs. For instance, in [4] it is proved that a DNN with three hidden layers and specific types of activation functions has the UAP. The paper [5] demonstrates that a very deep ResNet, with stacked modules having one neuron per hidden layer and rectified linear unit (ReLU) activation functions, can uniformly approximate any integrable function. However, extending these results to other classes of activation functions is not straightforward. We defer the interested readers to [9] for a detailed survey on the subject.

Recently, an alternate representation of DNNs as dynamical systems has been proposed [10]. This idea was later popularized as Neural Ordinary Differential Equations (NODEs) [11]. By viewing DNNs through a dynamical perspective, researchers have been able to utilize tools from system theory in order to analyze their properties (*e.g.*, Lyapunov stability, contraction theory, and symplectic properties). Similar to DNNs, there are some contributions on UAPs for NODEs. It has been shown in [12] that capping a NODE with a single linear layer is sufficient to guarantee the UAP, but exclusively for non-invertible continuous functions. Furthermore, in [13], differential geometric tools for controllability analysis were used to provide UAPs for a class of NODEs, while in [14], the compositional properties of the flows of NODEs were exploited to obtain UAPs. In [13] certain restrictions on the choice of activation functions are present, whereas [14] impose constraints on the desired target function. Finally in [15], some interesting tools, such as composition of contractive, expansive, and sphere-preserving flow maps, have been used to prove a universal approximation theorem for the flows of dynamical systems.

Although DNNs tend to empirically perform well in general, the increasing depth can also present challenges, such as the vanishing/exploding gradient problem during the training via gradient descent algorithms. These phenomenon happen when the gradients computed during back-propagation either approach to zero or diverge. In such cases, the learning process may stop prematurely or become unstable, thereby limiting the depth of DNNs that can be utilized and consequently preventing the practical exploitation of UAP in DNNs. Practitioners have proposed several remedies to address these challenges, including skip connections in ResNet [3], batch normalization, subtle weights initialization, regularization techniques such as dropout or weight decay, and gradient clipping [16]. However, all of these ad hoc methods do not come with provable formal guarantees of non-vanishing gradients. Recently, a class of DNNs called Hamiltonian Deep Neural Networks (HDNNs)

This research is supported by the Swiss National Science Foundation under the NCCR Automation (grant agreement 51NF40 180545).

Muhammad Zakwan and Giancarlo Ferrari Trecate are with the Institute of Mechanical Engineering, Ecole Polytechnique Fédérale de Lausanne (EPFL), CH-1015 Lausanne, Switzerland {muhammad.zakwan, giancarlo.ferraritrecate}@epfl.ch.
Massimiliano d'Angelo is with Sapienza University of Rome, Rome, 00185, Italy, and with the National Research Council of Italy (IASI-CNR), Rome, 00185, Italy, massimiliano.dangelo@iasi.cnr.it.
Corresponding author: M. d'Angelo.

Copyright ©2023 IEEE

have been proposed in [17]. These DNNs stem from the discretization of Hamiltonian NODEs, and enjoy non-vanishing gradients *by design* if symplectic discretization methods [18] are used [17]. Moreover, the expressivity of HDNNs has been demonstrated empirically on several benchmarks in classification tasks. Nevertheless, the theoretical foundation on the UAP of HDNNs has yet to be explored.

A. Contributions

In this paper, we present a rigorous theoretical framework to prove a UAP of HDNNs. First, with a slight modification, we generalize the class of HDNNs considered in [17] without compromising the provable non-vanishing gradients property[1]. Second, we prove that a portion of the flow of HDNNs can approximate any continuous function with arbitrary accuracy. To the best of our knowledge, this is the first UAP result for a class of ResNets enjoying non-vanishing gradients which are essential for numerically well-posed training. The proof is based on three essential features *i.e.* symplectic discretization through the Semi-Implicit Euler (SIE) method, a careful choice of initial conditions, and an appropriate selection of the flow. It is important to note that general DNNs, such as deep Multi-Layered Perceptrons (MLPs) or recurrent NNs, can suffer from vanishing gradients and might fail to approximate arbitrary functions if the training stops early. Third, since DNNs arising from the discretization of ODEs are automorphic maps – they do not alter the dimension of the input data – based on the composition of functions, we extend the main result to approximate maps, where the dimensions of domain and co-domain are different. Finally, we provide a characterization of the approximation error with respect to the depth.

Organization: Section II provides preliminaries on Hamiltonian NODEs, the employed discretization scheme, definitions of UAPs, and the problem formulation. In Section III, we prove the UAP for HDNNs (Theorem 1), we investigate the case when the desired function is not an automorphic map (Corollary 1), and provide some remarks on the approximation error (Proposition 2). We discuss a numerical example in Section IV. Finally, conclusions are drawn in Section V.

B. Notation

We denote the set of non-negative reals with $\mathbb{R}_+$. For a vector $x \in \mathbb{R}^n$, its 2-norm is represented by $\|x\|$ and its 1-norm $\|x\|_1 := \sum_j |x_j|$. Given an $\mathcal{L}_2$-function $f : \mathbb{R}^n \to \mathbb{R}^n$ the $\mathcal{L}_2$ norm over the compact set $\Omega \subset \mathbb{R}^n$ is denoted by $\|f\|_{\mathcal{L}_2(\Omega)}$ and the (essential) supremum norm by $\|f\|_{\mathcal{L}_\infty(\Omega)} = \sup_{x\in\Omega} \|f(x)\|$. $|A|$ stands for the determinant of a squared matrix A. We represent with 0_n the zero vector in $\mathbb{R}^n$ and with $0_{n\times n}$ the matrix with all entries equal to zero in $\mathbb{R}^{n\times n}$. We denote the column vector of ones of dimension n with $\mathbb{1}_n$. Given $\Omega \subset \mathbb{R}^n$, $\mathcal{C}(\Omega;\mathbb{R}^n)$ stands for the space of continuous functions $f : \Omega \to \mathbb{R}^n$. Given $T \in \mathbb{R}_+$, we refer to $\mathcal{P}([0,T];\mathbb{R}^p)$ as the space of piecewise constant function $\theta : [0,T] \to \mathbb{R}^p$. Functions that cannot be represented in the form of a polynomial are referred to as *non-polynomial* functions.

II. Preliminaries and Problem Formulation

A. Hamiltonian Neural Ordinary Differential Equations

A Neural ODE [11] (NODE) is represented by the dynamical system for $t \in [0,T]$ given by

$$\dot{x}(t) = F(x(t),\theta(t)) \quad \text{with } x(0) = x_0 \in \Omega\,, \qquad (1)$$

where $x(t) \in \mathbb{R}^n$ is the state at time t of the NODE with initial condition x_0 in some compact set $\Omega \in \mathbb{R}^n$, and $F : \mathbb{R}^n \times \mathbb{R}^p \to \mathbb{R}^n$ is such that $F(x,\theta)$ is Lipschitz continuous with respect to x and measurable with respect to the weights θ. We further assume that $\theta(t) \in \mathcal{P}([0,T];\mathbb{R}^p)$. When used in machine learning tasks, the NODE is usually pre- and post-pended with additional layers, *e.g.,* $x_0 = \Psi_\alpha(z)$, with $z \in \mathbb{R}^{n_z}$ the input and Ψ_α a NNs (*e.g.* a convolutional layer) with parameters $\alpha \in \mathbb{R}^{n_\alpha}$, and the output y is computed as $y = \phi_\beta(x(T))$, where ϕ_β is a NNs with parameters $\beta \in \mathbb{R}^{n_\beta}$.

In this paper, we consider a class of NODEs inspired by Hamiltonian systems. In particular, we consider the Hamiltonian function $H : \mathbb{R}^{2n} \times \mathbb{R}_+ \to \mathbb{R}$ given by

$$H(x,t) = \tilde{\sigma}(W(t)x + b(t))^\top \mathbb{1}_n + \eta(t)^\top x\,, \qquad (2)$$

where $W : \mathbb{R}_+ \to \mathbb{R}^{2n\times 2n}$, $b : \mathbb{R}_+ \to \mathbb{R}^{2n}$, $\eta : \mathbb{R}_+ \to \mathbb{R}^{2n}$ are piece-wise constant, while $\tilde{\sigma} : \mathbb{R} \to \mathbb{R}$ is a differentiable map, applied element-wise when the argument is a matrix, and such that $\sigma(x) := \frac{\partial\tilde{\sigma}}{\partial x}(x)$ is non-polynomial and Lipschitz continuous. As explained below, σ will play the role of the so-called *activation function*. Examples that satisfy the above assumptions are provided in Table I. Note that if we set $\eta(t) = 0$ in (2), we recover DNNs proposed in [10], [17]. We define the Hamiltonian system

$$\dot{x}(t) = J(t)\frac{\partial H(x(t),t)}{\partial x}\,, \qquad (3)$$

where $J(t)$ is piecewise constant skew-symmetric matrix, namely $J(t) = -J(t)^\top$, in $\mathbb{R}^{2n} \times \mathbb{R}^{2n}$ for any $t \geq 0$. By taking into account the expression of the Hamiltonian in (2), we obtain the following dynamics

$$\dot{x}(t) = J(t)\left(W(t)^\top \sigma\big(W(t)x(t) + b(t)\big) + \eta(t)\right)\,. \qquad (4)$$

Note that the latter equation can be written in the form (1), when the weights are given by $\theta(t) = \{J(t), W(t), b(t), \eta(t)\}$ for $t \in [0,T]$.

For the numerical implementation of NODE (4), we rely on the SIE discretization [18] because it can preserve the symplectic flow of time-invariant Hamiltonian systems and is crucial to prove non-vanishing gradient property of the resulting HDNNs (further details will be given in the next section). In particular, splitting the state of the Hamiltonian systems into $x = (p,q)$, we obtain the HDNN

$$\begin{bmatrix} p_{j+1} \\ q_{j+1} \end{bmatrix} = \begin{bmatrix} p_j \\ q_j \end{bmatrix} + hJ_j \begin{bmatrix} \frac{\partial H}{\partial p}(p_{j+1},q_j,t_j) \\ \frac{\partial H}{\partial q}(p_{j+1},q_j,t_j) \end{bmatrix}, \qquad (5)$$

where $h = T/N$, with $N \in \mathbb{N}$, is the integration step-size, $j = 0,\dots,N-1$ and p_j and q_j are the two state components

[1] For the sake of simplicity, we retain the same name and also refer to the proposed modified version as HDNNs.

Activation Function	$\sigma(x)$
ReLU	$\max\{x, 0\}$
Sigmoidal	$(1+\exp(-x))^{-1}$
Softplus	$\log(1+\exp(x))$
Hyperbolic Tangent	$\tanh(x)$
Radial Basis Function	$\frac{1}{\sqrt{2\pi}}\exp(-\frac{x^2}{2})$

TABLE I

EXAMPLES OF ACTIVATION FUNCTIONS.

in $\mathbb{R}^n$. Moreover, by taking into account the expression of the Hamiltonian in (2), namely the dynamics (4), we obtain the following difference equation

$$\begin{aligned}\begin{bmatrix} p_{j+1} \\ q_{j+1} \end{bmatrix} = \begin{bmatrix} p_j \\ q_j \end{bmatrix} \\ + hJ_j\left(W_j^\top \sigma\left(W_j\begin{bmatrix} p_{j+1} \\ q_j \end{bmatrix} + b_j\right) + \eta_j\right) .\end{aligned} \tag{6}$$

Clearly, the set of weights is given by $\theta_j = \{J_j, W_j, b_j, \eta_j\}$ with $j = 0, \ldots, N-1$. With a little abuse of notation we write $\theta_j \in \mathbb{R}^p$ with $j = 0, \ldots, N-1$ and appropriate $p \in \mathbb{N}$. Although, in general, one has to compute the update (p_{j+1}, q_{j+1}) of (6) through an implicit expression, it is possible to rewrite it in an explicit form, when the matrices J_j and W_j satisfy some assumptions, *e.g.*, by choosing J_j block anti-diagonal and W_j block diagonal [17].

B. Universal Approximation Property

In this section, we present some essential definitions pertaining to universal approximation properties.

Definition 1 (UAP of a function): Consider a function $g_\theta : \mathbb{R}^n \to \mathbb{R}^n$ with parameters $\theta \in \mathbb{R}^p$ and a compact subset $\Omega \subset \mathbb{R}^n$, then g_θ has the Universal Approximation Property (UAP) on $\Omega \subset \mathbb{R}^n$ if for any $f \in \mathcal{C}(\Omega; \mathbb{R}^n)$ and $\varepsilon > 0$, there exists $\theta \in \mathbb{R}^p$ such that

$$\sup_{x\in\Omega} \|f(x) - g_\theta(x)\| \leq \varepsilon \, . \tag{7}$$

We provide the following fact which descends from [19].

Proposition 1: Let $\sigma \in \mathcal{C}(\mathbb{R}; \mathbb{R})$ be non-polynomial, then for any $f \in \mathcal{C}(\Omega; \mathbb{R}^n)$, where $\Omega \subset \mathbb{R}^n$, and $\varepsilon > 0$, there exist $N \in \mathbb{N}$, $A_j, W_j \in \mathbb{R}^{n\times n}$ and $b_j \in \mathbb{R}^n$ such that the function $g : \mathbb{R}^n \to \mathbb{R}^n$ given by

$$g(x) := \sum_{j=0}^{N-1} A_j\,\sigma(W_j x + b_j) \, , \tag{8}$$

satisfies

$$\sup_{x\in\Omega} \|f(x) - g(x)\| \leq \varepsilon \, . \tag{9}$$

Some examples of activation functions σ, such that g in (8) satisfies the UAP, are given in Table I.

In the sequel, we refer to the UAP with bound $\varepsilon > 0$ to quantify the estimation error in equations (7), and (9). This value is typically a function of N, n, and the desired f, and it is characterized in Proposition 2.

C. Problem formulation

The goal of our paper can be formulated as follows. *Problem 1: Let $f : \mathbb{R}^n \to \mathbb{R}^n$ be a continuous function, $\Omega \subset \mathbb{R}^n$ be a compact set, and $\varepsilon > 0$ be the desired approximation accuracy. Find $N \in \mathbb{N}$ and weights $\theta_j = \{J_j, W_j, b_j, \eta_j\}$ with $j = 0, \ldots, N-1$ of* (6)*, such that a portion $\varphi : \mathbb{R}^n \to \mathbb{R}^n$ of the flow $\Phi_N : \mathbb{R}^{2n} \to \mathbb{R}^{2n}$ at time $N \in \mathbb{N}$ of* (6) *has the UAP on Ω.*

We recall that the flow at time $N \in \mathbb{N}$ of (6) is the corresponding unique solution at time $N \in \mathbb{N}$. In particular, $\Phi_N : \mathbb{R}^{2n} \to \mathbb{R}^{2n}$ is the flow at time N as function of the initial condition. The flow φ will be precisely defined in Theorem 1.

Moreover, motivated by real-world applications, we are also interested in approximating arbitrary continuous functions $f : \mathbb{R}^n \to \mathbb{R}^r$ where r is not necessarily equal to n. For instance, in classification tasks, typically $r < n$, as r corresponds to the number of classes to be classified and n represents the number of features. We address this problem in Corollary 1.

III. MAIN RESULTS

In this section, we present our main results whose proofs are given the Appendix. We address the Problem 1 in Theorem 1, which is a universal approximation theorem for the HDNN (5).

Theorem 1: Consider the discrete-time system (6) with initial condition $(p_0, q_0) = (\xi, 0_n)$, for some $\xi \in \Omega$ with $\Omega \subset \mathbb{R}^n$ compact. Then, the *restricted* flow $\varphi : \xi \mapsto q_N$ has the UAP on Ω.

In other words, Theorem (1) states that given the system (6) with initial condition $(p_0, q_0) = (\xi, 0_n)$, for any $f \in C(\Omega; \mathbb{R}^n)$ and $\varepsilon > 0$, there exist $N \in \mathbb{N}$ and weights $\theta_j = \{J_j, W_j, b_j, \eta_j\}$ with $j = 0, \ldots, N-1$ such that the function $\varphi : \xi \mapsto q_N$ satisfies

$$\sup_{\xi\in\Omega} \|f(\xi) - \varphi(\xi)\| \leq \varepsilon \, . \tag{10}$$

Remark 1 (Key ingredients for UAP): The proof of Theorem 1, besides exploiting arguments from [7], [8] for showing UAPs, it is based on three critical key steps: *i)* the SIE discretization scheme, *ii)* the initial condition $(p_0, q_0) = (\xi, 0_n)$, and *iii)* the focus on the *restricted* flow $\xi \mapsto q_N$, which refers to map the initial condition of the p state to the flow of the q state.

In particular, the choice of the SIE discretization scheme together with the initial condition $(p_0, q_0) = (\xi, 0_n)$ allows one to exploit the framework of Cybenko [7] to express the function $\varphi : \xi \mapsto q_N$ as (8) (see equation (21) in the Proof of Theorem 1 in Appendix B).

Remark 2 (Feature augmentation): By defining the flow Φ_N of the discrete-time system (6) (evolving in $\mathbb{R}^{2n}$), we note that (10) can be written as

$$\sup_{x\in\Omega} \|f(x) - \pi \circ \Phi_N \circ \iota(x)\| \leq \varepsilon \, , \tag{11}$$

where $\iota : \mathbb{R}^n \to \mathbb{R}^{2n}$ is the injection given by $\iota(z_1, \ldots, z_n) = (z_1, \ldots, z_n, 0, \ldots, 0)$ and $\pi : \mathbb{R}^{2n} \to \mathbb{R}^n$ is the projection $\pi(x_1, \ldots, x_n, x_{n+1}, \ldots, x_{2n}) = (x_{n+1}, \ldots, x_{2n})$. This is equivalent to the common practice in machine learning of augmenting the size of the feature space [20]. It has been demonstrated that this technique can improve DNN performance in several learning tasks. Moreover, it is also closely related to the idea of extended space [16], which suggests that by increasing the dimensionality of the feature space, one can capture more complex relationships.

We note that the UAP results in [13] do not apply in our framework because of the skew-symmetric matrix J multiplying the partial derivative of the Hamiltonian in (3). Moreover, we provide UAPs directly for implementable discrete-time layer equations (6) instead of the continuous-time NODEs. Indeed, an arbitrary discretization method may not conserve the desired properties, making it challenging to prove the UAP of discretized NODEs in general.

Untill this point, we focused on automorphisms on $\mathbb{R}^n$. The next result presents the UAP of a general map from $\Omega \subset \mathbb{R}^n$ to $\mathbb{R}^r$.

Corollary 1: Consider the discrete-time system (6) with initial condition $(p_0, q_0) = (\xi, 0_n)$, for some $\xi \in \Omega$ with $\Omega \subset \mathbb{R}^n$ compact, and the *restricted* flow $\varphi : \xi \mapsto q_N$. Let $h : \mathbb{R}^n \to \mathbb{R}^r$ be a Lipschitz continuous function such that $f(\Omega) \subseteq h(\mathbb{R}^n)$. Then, for any $\varepsilon > 0$, the function $h \circ \varphi : \Omega \to \mathbb{R}^r$, satisfies

$$\sup_{\xi \in \Omega} \| f(\xi) - h \circ \varphi(\xi) \| \leq \varepsilon. \tag{12}$$

A typical example that satisfies the necessity condition $f(\Omega) \subseteq h(\mathbb{R}^n)$ is $h(\varphi) = W_o^\top \varphi + b_o$, $W_o \in \mathbb{R}^{n \times r}$, and $b_o \in \mathbb{R}^r$, which is common in classification problems. It is straightforward to see that $h(\cdot)$ is Lipschitz continuous, surjective, and satisfies the condition $f(\Omega) \subseteq h(\mathbb{R}^n)$.

It is worth mentioning that unlike other papers [4], [5], [13], our results do not impose restrictive conditions on activation functions, which expands their potential applicability.

A. Auxiliary properties of HDNNs

In the following, we highlight a few associated properties of HDNNs. First, we provide a bound on the desired accuracy of the approximation error with respect to the depth of HDNNs. Second, we state a remark on their non-vanishing gradients property.

Let us define the first absolute moment C_f of the Fourier magnitude distribution of a desired function f. Thus, given $f : \mathbb{R}^n \to \mathbb{R}^n$, with a Fourier representation of the form $f(x) = \int_{\mathbb{R}^n} \mathrm{e}^{i\omega^\top x} \tilde{f}(\omega) \mathrm{d}x$, we define

$$C_f := \int_{\mathbb{R}^n} \|\omega\|_1 \|\tilde{f}(\omega)\| \mathrm{d}\omega \,. \tag{13}$$

The condition (13) is usually interpreted as the integrability of the Fourier transform of the gradient of the function f, and a vast list of examples for which bounds on C_f can be obtained are given in Section IX of [21].

Proposition 2: Consider the discrete-time system (6) with sigmoidal[2] σ and initial condition $(p_0, q_0) = (\xi, 0_n)$, for some $\xi \in \Omega = [-1, 1]^n$. Then, the *restricted* flow $\varphi : \xi \mapsto q_N$ has the UAP on Ω with bound $2^{\frac{n}{2}} \frac{C_f}{\sqrt{N}}$.

Proposition 2 states that for any $f \in \mathcal{C}(\Omega; \mathbb{R}^n)$ with finite C_f and $N \in \mathbb{N}$, there exist parameters $\theta_j = \{J_j, W_j, b_j, \eta_j\}$ with $j = 0, \ldots, N-1$, such that the function $\varphi : \xi \mapsto q_N$ satisfies

$$\sup_{x \in \Omega} \| f(x) - \varphi(x) \| \leq 2^{\frac{n}{2}} \frac{C_f}{\sqrt{N}} \,. \tag{14}$$

Further remarks on the evaluation/approximation of this bound can be found in [21] and [22].

As mentioned earlier, it has been shown that HDNNs considered in [17] are endowed with non-vanishing gradients or in a special case, non-exploding gradients [23], *i.e.*, they ensure numerically well-posed training. We defer the reader to those papers for a formal discussion of the non-vanishing gradients property.

Remark 3 (Non-vanishing gradients): The HDNN given by the discrete-time system (6) enjoys the non-vanishing gradients property when optimizing a generic loss function. In particular, this property is related to the Backward Sensitivity Matrix $\frac{\partial x_N}{\partial x_{N-j}} = \prod_{\ell=N-j}^{N-1} \frac{\partial x_{\ell+1}}{\partial x_\ell}$, where $x = (p, q)$, at layer $N-j$ for $j = 1, \ldots, N-1$. Although the considered Hamiltonian (2) is different from the one of [17] (because of the linear term), one is able to prove the non-vanishing gradients property (by establishing a lower bound for the Backward Sensitivity Matrix) by following the same arguments of [17, Theorem 2] which relies specifically on the symplectic property of the flow and not on the Hamiltonian structure.

IV. Numerical Example

In this example, our goal is to approximate the function $y(x) = 2(2\cos(x)^2 - 1)^2 - 1$ considered in [24]. The training set comprises 5000 datapoints generated by sampling $y(t)$ randomly for $x \in [-2\pi, 2\pi]$. We choose the mean square error as the loss function and compare the following NN architectures:

i) The SNN $\hat{y} = W_o \sigma(Wx + b)$, with $W_o \in \mathbb{R}^{1 \times N_h}$, $W \in \mathbb{R}^{N_h \times 1}$ and $b \in \mathbb{R}^{N_h}$, where N_h is the number of hidden neurons. We use the values of N_h in the set $\{400, 800, 1200, 1800, 2400\}$.

ii) an HDNN, called HDNN-1, with forward equation (6) and weight matrices (19) for $j = 0, 1, \cdots, 6$.

iii) an HDNN, called HDNN-2, with forward equation (6), where W_j is block-diagonal for $j = 0, 1, \cdots, 3$ to match the number of parameters in HDNN-1.

For HDNNs, we choose a sufficiently small step-size $h = 0.001$, and the initial conditions as $p_0, q_0 = ([x, 0_{M/2-1}], 0_{M/2})$, where M is always an even integer. Moreover, the output equation is given by $\tilde{y} = W_o q_N + b_o$, where $W_o \in \mathbb{R}^{1 \times M/2}$ and $b_o \in \mathbb{R}^{M/2}$. To have almost the same number of parameters in the chosen HDNNs, we choose M from the set $\{24, 36, 44, 54, 62\}$ for HDNN-1 and $\{26, 36, 44, 54, 64\}$ for HDNN-2, respectively.

Fig. 1 shows that the training loss decreases when more parameters are used for all three architectures. Moreover, we can see that for the same number of parameters, the block diagonal W_j matrices of HDNN-2 with half the number of

[2] The function $\sigma(x)$ is assumed to be a sigmoidal function, if it is a bounded function on the real line satisfying $\sigma(x) \to 1$ as $x \to \infty$ and $\sigma(x) \to -1$ as $x \to -\infty$ [22].

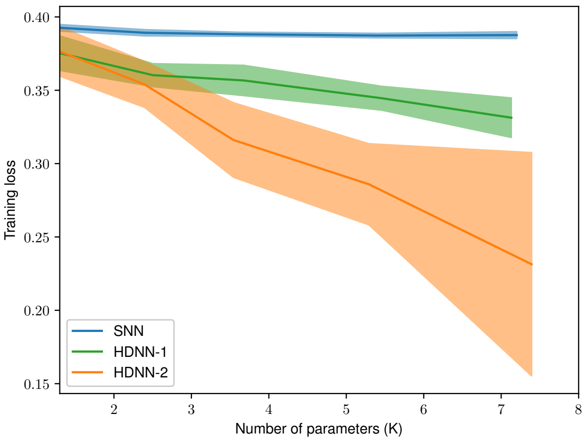

Fig. 1. Averaged training loss and standard deviations over multiple experiments for three architectures.

layers can be leveraged to further improve the performance over HDNN-1.

V. Conclusion and Future Work

We demonstrated the universal approximation property of Hamiltonian Deep Neural Networks (HDNNs) that also enjoy non-vanishing gradients during training. This result affirms both the practicality and theoretical foundation of HDNNs. In particular, we have demonstrated that a portion of the flow of HDNNs can approximate any continuous function in a compact domain. Also, we provide some insights on the approximation error with respect to the depth of neural network.

Our work opens doors to quantifying the expressivity of other physics-inspired neural networks with special properties, such as [25]. Future research will focus on leveraging differential geometric tools [13] to establish universal approximation properties for HDNNs, where the Hamiltonian function is parameterized by an arbitrary neural network.

Appendix

A. A preliminary lemma

In order to prove Theorem 1, we introduce a key auxiliary result which relaxes the necessity of full-rank weight matrices W_j in (8) assumed in [26, Theorem 2.6]. During the training of NN (8), some entries of W_j in (8) might vanish and this assumption cannot be satisfied. Therefore, the result in [26] might not be of practical use. However, the following Lemma shows that even if W_j in (8) are not full-rank, we can still construct an approximation with full-rank matrices and apply the results of [7], [19].

Lemma 1: Let g be the function in (8) with the UAP on Ω. For any $\tilde{\varepsilon} > 0$ we can find $\tilde{A}_j, \tilde{W}_j \in \mathbb{R}^{n\times n}$, with $\tilde{W}_j$ full rank, and $\tilde{b}_j \in \mathbb{R}^n$ for $j = 0, \ldots, N-1$, such that $\tilde{g}(x) := \sum_{j=0}^{N} \tilde{A}_j \, \sigma(\tilde{W}_j x + \tilde{b}_j)$ satisfies $\|\tilde{g} - g\|_{\mathcal{L}_\infty(\Omega)} \leq \tilde{\varepsilon}$.

In other words, Lemma 1 allows us to assume, without loss of generality, that the function g in (8) can be arbitrarily well-approximated by using full-rank matrices W_j for any $j = 0, \ldots, N-1$.

Proof: Given the function g in (8) with the UAP on Ω, we consider the case in which there exists the set $K = \{\kappa \in \{0, \ldots, N-1\} : |W_\kappa| = 0\}$ non-empty with cardinality $\tilde{n}$. For $\kappa \in K$, let $\operatorname{rank}(W_\kappa) = n - r_\kappa$, with $r_\kappa > 0$ the number of dependent column vectors of W_κ so that, up to a row permutation, assume W_κ is partitioned as

$$W_\kappa = \left(w_\kappa^{(1)}, \ldots, w_\kappa^{(r_\kappa)}, w_\kappa^{(r_\kappa+1)}, \ldots, w_\kappa^{(n)}\right)^\top, \tag{15}$$

with the last $n - r_\kappa$ vectors linearly independent. Then, the parameters $\tilde{A}_j, \tilde{W}_j \in \mathbb{R}^{n\times n}$ and $\tilde{b}_j \in \mathbb{R}^n$ of the function $\tilde{g}(x) = \sum_{j=0}^{N} \tilde{A}_j \, \sigma(\tilde{W}_j x + \tilde{b}_j)$ can be selected as follows. We set $\tilde{A}_j = A_j$, $\tilde{b}_j = b_j$ for all $j = 0, \ldots, N-1$. Moreover, $\tilde{W}_j = W_j$ for all $j \notin K$ and, for $\kappa \in K$, $\tilde{W}_\kappa = W_\kappa + \Lambda_\kappa$, where $\Lambda_\kappa = \left(\tilde{w}_\kappa^{(1)}, \ldots, \tilde{w}_\kappa^{(r_\kappa)}, 0_n, \ldots, 0_n\right)^\top$, and the vectors $\tilde{w}_\kappa^{(\ell)}$, $\ell = 1, \ldots, r_\kappa$, are selected such that $|\tilde{W}_\kappa| \neq 0$ and

$$\|\tilde{w}_\kappa^{(\ell)}\| \leq \frac{\tilde{\varepsilon}}{r_\kappa \, \tilde{n} \, \sqrt{n} \, L_\sigma \, \|x\|_{\mathcal{L}_\infty(\Omega)} \, \max_{1\leq p\leq n} \|a_\kappa^{(p)}\|}, \tag{16}$$

where L_σ is the Lipschitz constant of function σ and ${a_\kappa^{(p)}}^\top$, $p = 1, \ldots, n$, are the rows of the matrix A_κ[3]. By noticing that for $x \in \Omega$ we have

$$\left\|(\tilde{W}_\kappa - W_\kappa)x\right\| \leq \|x\|_{\mathcal{L}_\infty(\Omega)} \sum_{\ell=1}^{r_\kappa} \|\tilde{w}_\kappa^{(\ell)}\|, \tag{17}$$

and by looking at the p-th component of the difference $\tilde{g} - g$, by inequality (16), for $x \in \Omega$, we have

$$\begin{aligned} \left|\tilde{g}^{(p)}(x) - g^{(p)}(x)\right| &\leq L_\sigma \, \|x\|_{\mathcal{L}_\infty(\Omega)} \sum_{\kappa\in K} \left(\|a_\kappa^{(p)}\| \sum_{\ell=1}^{r_\kappa} \|w_\kappa^{(\ell)}\| \right) \\ &\leq \sum_{\kappa\in K} \frac{\tilde{\varepsilon}}{\tilde{n}\sqrt{n}} \leq \frac{\tilde{\varepsilon}}{\sqrt{n}}, \end{aligned} \tag{18}$$

from which we obtain $\|\tilde{g} - g\|_{\mathcal{L}_\infty(\Omega)} \leq \tilde{\varepsilon}$. ■

B. Proof of Theorem 1

We prove the result by showing that the function $\varphi : \xi \mapsto q_N$ can be written in the form (8), and thus, satisfying Proposition 1, it has the UAP on Ω. In fact, by restricting the parameter space as follows

$$\begin{aligned} J_j &= \begin{bmatrix} 0_{n\times n} & -X \\ X & 0_{n\times n} \end{bmatrix} & W_j &= \begin{bmatrix} \tilde{W}_j & 0_{n\times n} \\ 0_{n\times n} & 0_{n\times n} \end{bmatrix}, \\ b_j &= \begin{bmatrix} \tilde{b}_j \\ 0_n \end{bmatrix} & \eta_j &= \begin{bmatrix} 0_n \\ -\tilde{\eta}_j \end{bmatrix}, \end{aligned} \tag{19}$$

[3] The case $A_\kappa = 0_{n\times n}$ is trivial since one can select any $\tilde{w}_\kappa^{(\ell)}$ such that $|\tilde{W}_\kappa| \neq 0$ and $\|\tilde{g} - g\|_{\mathcal{L}_\infty(\Omega)} = 0$.

where $X \in \mathbb{R}^{n\times n}$, $\tilde{W}_j : \mathbb{R}_+ \to \mathbb{R}^{n\times n}$, $\tilde{b}_j : \mathbb{R}_+ \to \mathbb{R}^n$, $\tilde{\eta}_j : \mathbb{R}_+ \to \mathbb{R}^n$, one can write (6) as

$$\begin{aligned}\begin{bmatrix} p_{j+1} \\ q_{j+1} \end{bmatrix} &= \begin{bmatrix} p_j \\ q_j \end{bmatrix} + h \begin{bmatrix} X^\top \tilde{\eta}_j \\ X \tilde{W}_j^\top \sigma(\tilde{W}_j p_{j+1} + \tilde{b}_j) \end{bmatrix} \\ &= \begin{bmatrix} p_j \\ q_j \end{bmatrix} + \begin{bmatrix} \tilde{\gamma}_j \\ \tilde{A}_j \sigma(\tilde{W}_j p_{j+1} + \tilde{b}_j) \end{bmatrix}\end{aligned} \tag{20}$$

for $j = 0, 1, \cdots, N-1$, where $\tilde{\gamma}_j = hX^\top \tilde{\eta}_j$, and $\tilde{A}_j = hX\tilde{W}_j^\top$, respectively. From the initial condition $(p_0, q_0) = (\xi, 0_n)$, $\xi \in \Omega$, and by substituting the expression of p_{j+1} into the second equation of (20) we have that

$$\begin{aligned} q_N &= \sum_{j=0}^{N-1} \tilde{A}_j \sigma(\tilde{W}_j(p_j + \tilde{\gamma}_j) + \tilde{b}_j) \\ &= \sum_{j=0}^{N-1} \tilde{A}_j \sigma(\tilde{W}_j \xi + \tilde{d}_j) =: \varphi(\xi) \ , \end{aligned} \tag{21}$$

where $\tilde{d}_j = \tilde{W}_j \tilde{r}_j + \tilde{b}_j$ with $\tilde{r}_j = \tilde{r}_{j-1} + \tilde{\gamma}_j$ and $\tilde{r}_0 = \tilde{\gamma}_0$. Notice that, because of Lemma 1, we can assume, without loss of generality, that $\tilde{W}_j$ in (21) are full-rank. Consequently, one can freely choose $\tilde{A}_j$ by setting $X = \frac{1}{h}\tilde{A}_j \tilde{W}_j^{-\top}$ for all $j = 0, \dots, N-1$, while $\tilde{d}_j$ is free by construction due to the parameter $\tilde{b}_j$. Thus, the map (21) has the UAP on Ω (Proposition 1), *i.e.* $\|\varphi(\xi) - g\|_{\mathcal{L}_\infty(\Omega)} \leq \tilde{\varepsilon}$, with g in (8). ■

Note that the zero patterns of matrices, *i.e.* W, η, b in (19) is only assumed for proving Theorem 1. However, since using more parameters[4] in (19) cannot compromise UAPs, the structure of the weight matrices in (19) is never used in practice.

C. Proof of Corollary 1

In [14, Proposition 3.8] it is shown that there exists a continuous function $\psi(\xi) = \sum_{i=1}^N z_i \psi_i(\xi)$ for $\xi \in \Omega$, where $z_i \in h^{-1}(F_i)$, with $\{F_i\}_{i=1}^N$ a partition of $f(\Omega)$, and continuous functions $\psi_i : \Omega \to [0,1]$ such that $\psi_i = 1$ on A_i and $\psi_i = 0$ on $\cup_{j\neq i} A_j$. The sets $A_i \subset \Omega_i$, with $\{\Omega_i\}_{i=1}^N$ a partition of Ω, such that $h \circ \psi$ has the UAP on Ω (provided that the desired function f is such that $f(\Omega) \subseteq h(\mathbb{R}^n)$). Now, take ψ such that $\|f - h\circ\psi\|_{\mathcal{L}_\infty(\Omega)} \leq \varepsilon/2$ and, by Theorem 1, take $\varphi : \xi \mapsto q_N$ such that $\|\psi - \varphi\|_{\mathcal{L}_\infty(\Omega)} \leq \varepsilon/(2L_h)$. Then, for any $\xi \in \Omega$ we have

$$\begin{aligned} \|f - h\circ\varphi(\xi)\| &\leq \|f(\xi) - h\circ\psi(\xi)\| \\ &\qquad + \|h(\xi)\circ\psi(\xi) - h\circ\varphi(\xi)\| \\ &\leq \frac{\varepsilon}{2} + L_h \|\psi(\xi) - \varphi(\xi)\| \leq \varepsilon \ , \end{aligned}$$

and the proof is completed. ■

D. Proof of Proposition 2

The proof follows from [22, Theorem 1] by noting that the function φ in (21) is the NN considered in [22], by selecting the probability measure $\tilde{\lambda}(\cdot) := \frac{1}{2^n}\lambda(\cdot)$ where λ is the Lebesgue measure on Ω, and by recalling the norm inequality $\|\cdot\|_\infty \leq \|\cdot\|_2$. ■

References

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, 2017.

[2] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1–9.

[3] 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.

[4] Z. Shen, H. Yang, and S. Zhang, "Neural network approximation: Three hidden layers are enough," *Neural Networks*, vol. 141, pp. 160–173, 2021.

[5] H. Lin and S. Jegelka, "Resnet with one-neuron hidden layers is a universal approximator," *Advances in neural information processing systems*, vol. 31, 2018.

[6] M. Raghu, B. Poole, J. Kleinberg, S. Ganguli, and J. Sohl-Dickstein, "On the expressive power of deep neural networks," in *international conference on machine learning*. PMLR, 2017, pp. 2847–2854.

[7] G. Cybenko, "Approximation by superpositions of a sigmoidal function," *Mathematics of control, signals and systems*, vol. 2, no. 4, pp. 303–314, 1989.

[8] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural networks*, vol. 2, no. 5, pp. 359–366, 1989.

[9] I. Gühring, M. Raslan, and G. Kutyniok, "Expressivity of deep neural networks," *arXiv preprint arXiv:2007.04759*, 2020.

[10] E. Haber and L. Ruthotto, "Stable architectures for deep neural networks," *Inverse problems*, vol. 34, no. 1, p. 014004, 2017.

[11] R. T. Chen, Y. Rubanova, J. Bettencourt, and D. K. Duvenaud, "Neural ordinary differential equations," in *Advances in neural information processing systems*, vol. 31, 2018.

[12] H. Zhang, X. Gao, J. Unterman, and T. Arodz, "Approximation capabilities of neural odes and invertible residual networks," in *International Conference on Machine Learning*. PMLR, 2020, pp. 11 086–11 095.

[13] P. Tabuada and B. Gharesifard, "Universal approximation power of deep residual neural networks through the lens of control," *IEEE Transactions on Automatic Control*, 2022, doi: 10.1109/TAC.2022.3190051.

[14] Q. Li, T. Lin, and Z. Shen, "Deep learning via dynamical systems: An approximation perspective," *Journal of the European Mathematical Society*, 2022.

[15] E. Celledoni, D. Murari, B. Owren, C.-B. Schönlieb, and F. Sherry, "Dynamical systems' based neural networks," *arXiv preprint arXiv:2210.02373*, 2022.

[16] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.

[17] C. L. Galimberti, L. Furieri, L. Xu, and G. Ferrari-Trecate, "Hamiltonian deep neural networks guaranteeing non-vanishing gradients by design," *IEEE Transactions on Automatic Control*, pp. 1–8, 2023, doi: 10.1109/TAC.2023.3239430.

[18] E. Hairer, M. Hochbruck, A. Iserles, and C. Lubich, "Geometric numerical integration," *Oberwolfach Reports*, vol. 3, no. 1, pp. 805–882, 2006.

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

[20] E. Dupont, A. Doucet, and Y. W. Teh, "Augmented Neural ODEs," *Advances in neural information processing systems*, vol. 32, 2019.

[21] A. R. Barron, "Universal approximation bounds for superpositions of a sigmoidal function," *IEEE Transactions on Information theory*, vol. 39, no. 3, pp. 930–945, 1993.

[22] ——, "Approximation and estimation bounds for artificial neural networks," *Machine learning*, vol. 14, pp. 115–133, 1994.

[23] M. Zakwan, L. Xu, and G. Ferrari-Trecate, "Robust classification using contractive Hamiltonian neural odes," *IEEE Control Systems Letters*, vol. 7, pp. 145–150, 2022.

[24] H. Mhaskar, Q. Liao, and T. Poggio, "Learning functions: when is deep better than shallow," *arXiv preprint arXiv:1603.00988*, 2016.

[25] M. Zakwan, L. Di Natale, B. Svetozarevic, P. Heer, C. N. Jones, and G. F. Trecate, "Physically consistent Neural ODEs for learning multi-physics systems," *arXiv preprint arXiv:2211.06130*, 2022.

[26] Y. Aizawa and M. Kimura, "Universal approximation properties for odenet and resnet," *arXiv preprint arXiv:2101.10229*, 2020.

[4] We recall that J_j should keep the sparsity structure (19) to maintain the non-vanishing gradients property (see [17, Theorem 2]).