Geometric Neural Ordinary Differential Equations: From Manifolds to Lie Groups

Yannik Paul Wotte *, Federico Califano, Stefano Stramigioli

Posted Date: 12 June 2025

doi: 10.20944/preprints202506.1038.v1

Keywords: Neural Ordinary Differential Equations; Differential Geometry; Lie groups; Machine Learning; Optimal Control

 doi:10.20944/preprints202506.1038.v1

Article

Geometric Neural Ordinary Differential Equations: From Manifolds to Lie Groups

Yannik P. Wotte *, Federico Califano and Stefano Stramigioli

Robotics and Mechatronics, EEMCS, University of Twente (UT), Drienerlolaan 5, 7522 NB Enschede, The Netherlands
* Correspondence: y.p.wotte@utwente.nl

Abstract: Neural ordinary differential equations (neural ODEs) are a well-established tool for optimizing the parameters of dynamical systems, with applications in image classification, optimal control, and physics-learning. Although dynamical systems of interest often evolve on Lie groups and more general differentiable manifolds, theoretical results on neural ODEs are frequently phrased on $\mathbb{R}^n$. We collect recent results on neural ODEs on manifolds, and present a unifying derivation of various results that serves as a tutorial to extend existing methods to differentiable manifolds. We also extend the results to the recent class of neural ODEs on Lie groups, highlighting a non-trivial extension of manifold neural ODEs that exploits the Lie group structure.

Keywords: Neural Ordinary Differential Equations; Differential Geometry; Lie groups; Machine Learning; Optimal Control

1. Introduction

Ordinary differential equations (ODEs) are ubiquitious in the engineering sciences, from modeling and control of simple physical systems like pendulums and mass-spring dampers, or more complicated robotic arms and drones, to the description of high-dimensional spatial discretizations of distributed systems, such as fluid flows, chemical reactions or quantum oscillators. Neural ordinary differential equations (neural ODEs) [1,2] are ODEs parameterized by neural networks. Given a state x, and parameters θ representing weights and biases of a neural network, a neural ODE reads:

$$\dot{x} = f_\theta(x, t)\,, x(0) = x_0\,. \tag{1}$$

First introduced by [1] as the continuum limit of recurrent neural networks, the number of applications of neural ordinary differential equations quickly exploded beyond simple classification tasks: learning highly nonlinear dynamics of multi-physical systems from sparse data [3–5], optimal control of nonlinear systems [6], medical imaging [7] and real-time handling of irregular time-series[8], to name but a few. Discontinuous state-transitions and dynamics [9,10], time-dependent parameters [11], augmented neural ODEs [12] and physics preserving formulations [13,14] present further extensions that increase the expressivity of neural ODEs.

However, these methods are typically phrased for states $x \in \mathbb{R}^n$. For many physical systems of interest, such as robot arms, humanoid robots and drones, the state lives on differentiable manifolds and Lie groups [15,16]. More generally, the manifold hypothesis in machine learning raises the expectation that many high-dimensional data-sets evolve on intrinsically lower-dimensional, albeit more complicated manifolds [17]. Neural ODEs on manifolds [18,19] presented significant steps to address this gap, presenting first optimization methods for neural ODEs on manifolds. Yet, the general tools and approaches available on $\mathbb{R}^n$, such as including running costs, augmented states, time-dependent parameters, control-inputs or discontinuous state-transitions, are rarely addressed in a manifold context. Similar issues persist in a Lie group context, were Neural ODEs on Lie groups [20,21] were formalized.

 doi:10.20944/preprints202506.1038.v1

Our goal is to extend more of the methods for neural ODEs on $\mathbb{R}^n$ to arbitrary manifolds, and in particular Lie groups. To this end, we present a systematic approach to the design of neural ODEs on manifolds and Lie groups. Specifically, our contributions are:

- Systematic derivation of neural ODEs on manifolds and Lie groups, highlighting differences and equivalence of various approaches - for an overview, see also Table 1;
- Summarizing the state of the art on manifold and Lie group neural ODEs, by formalizing the notion of extrinsic and intrinsic neural ODEs;
- A tutorial-like introduction, to assist the reader in implementing various neural ODE methods on manifolds and Lie groups, presenting coordinate expressions alongside geometric notation.

Table 1. Summary of neural ODEs on manifolds and Lie groups presented in this article

Name of Neural ODE	Subtype	Trajectory Cost	Subsection	Originally introduced in
Neural ODEs on manifolds (Section 3)	Extrinsic	Running and final cost	Section 3.1.1	Final cost [22], running cost [21]
	Intrinsic	Running and final cost , intermittent cost	Section 3.1.2	Final cost [18], running cost [21], intermittent cost (this work)
	Augmented, time-dependent parameters	Final cost	Section 3.2.2	Augmenting $\mathcal{M}$ to $T\mathcal{M}$ [23], Augmenting $\mathcal{M}$ to $\mathcal{M} \times \mathcal{N}$ (this work)
Neural ODEs on Lie groups (Section 4)	Extrinsic	Final cost and intermittent cost	Section 4.1	In [20]
	Intrinsic, dynamics in local charts	Running and final cost	Section 4.2	In [21,24]
	Intrinsic, dynamics at Lie algebra	Running and final cost	Section 4.2	In [21]

[1] Tables may have a footer.

The remainder of the article is organized as follows. A brief state-of-the-art on neural ODEs concludes this introduction. Section 2 provides background on differentiable manifolds, Lie groups, and the coordinate-free adjoint method. Section 3 describes neural ODEs on manifolds, and derives parameter updates via the adjoint method for various common architectures and cost-functions, including time-dependent parameters, augmented neural ODEs, running costs, and intermediate cost-terms. Section 4 describes neural ODEs on matrix Lie groups, explaining merits of treating Lie groups separately from general differentiable manifolds. Both Sections 3 and 4 also classify methods into extrinsic and intrinsic approaches. We conclude with a discussion in Section 5, highlighting advantages, disadvantages, challenges and promises of the presented material. The Appendix includes background on Hamiltonian systems, which appear when transforming the adjoint method into a form that is unique to Lie groups.

1.1. Literature review

For a general introduction to neural ODEs, see [25]. Neural ODEs on $\mathbb{R}^n$ with fixed parameters were first introduced by [1], and parameter optimization via the adjoint method allowed for intermittent and final cost terms on each trajectory. The generalized adjoint method [2] also allows for running cost terms. Memory-efficient checkpointing is introduced in [26] to address stability issues of

 doi:10.20944/preprints202506.1038.v1

adjoint methods. Augmented neural ODEs [12] introduced augmented state-spaces to allow neural ODEs to express arbitrary diffeomorphisms. Time-varying parameters were introduced by [11], with similar benefits to augmented neural ODEs. Neural ODEs with discrete transitions were formulated in [9,10], with [9] also learning event-triggered transitions common in engineering application. Neural controlled differential equations (CDEs) were introduced in [27] for handling irregular time-series, and parameter updates reapply the adjoint method [1]. Neural stochastic differential equations (SDEs) were introduced in [28], relying on a stochastic variant of the adjoint method for the parameter update. The previously mentioned literature phrases dynamics of neural ODEs on $\mathbb{R}^n$.

Neural ODEs on manifolds were first introduced by [22], including an adjoint method on manifolds for final cost terms, and application to continuous normalizing flows on Riemannian manifolds, but embedding manifolds into $\mathbb{R}^n$. Neural ODEs on Riemannian manifolds are expressed in local exponential charts in [18], avoiding embedding into $\mathbb{R}^n$, and considering final cost terms in the optimization. Charts for unknown, nontrivial latent manifolds, and dynamics in local charts, are learned from high-dimensional data in [29], including also discretized solutions to partial differential equations. Parameterized equivariant Neural ODEs on manifolds are constructed in [23], also commenting on state augmentation to express arbitrary (equivariant) flows on manifolds.

Neural ODEs on Lie groups were first introduced in [30] on the Lie group $SE(3)$, to learn the port-Hamiltonian dynamics of a drone from experiment, expressing group elements on an embedding $\mathbb{R}^{12}$, and the approach was formalized to port-Hamiltonian systems on arbitrary matrix Lie groups in [20], embedding $m \times m$ matrices in $\mathbb{R}^{m^2}$.

Neural ODEs on $SE(3)$ were phrased in local exponential charts in [24] to optimize a controller for a rigid body, using a chart-based adjoint method in local expential charts. An alternative, Lie algebra based adjoint method on general Lie groups was introduced in [21], foregoing Lie group specific numerical issues of applying the adjoint method in local charts.

1.2. Notation

For a complete introduction to differential geometry see e.g, [31], and for Lie group theory see [32].

Calligraphic letters $\mathcal{M}, \mathcal{N}, \ldots$ denote smooth manifolds. For conceptual clarity, the reader may think of these manifolds as embedded in a high dimensional $\mathbb{R}^N$, e.g., $\mathcal{M} \subset \mathbb{R}^N$. The set $C^\infty(\mathcal{M},\mathcal{N})$ contains smooth functions between $\mathcal{M}$ and $\mathcal{N}$, and we define $C^\infty(\mathcal{M}) := C^\infty(\mathcal{M},\mathbb{R})$:.

The tangent space at $x \in \mathcal{M}$ is $T_x\mathcal{M}$ and the cotangent space is $T_x^*\mathcal{M}$. The tangent bundle of $\mathcal{M}$ is $T\mathcal{M}$, and the cotangent bundle of $\mathcal{M}$ is $T^*\mathcal{M}$. Then $\mathfrak{X}(\mathcal{M})$ denotes the set of vector fields over $\mathcal{M}$, and $\Omega^k(\mathcal{M})$ denotes the set of k forms, where $\Omega^1(\mathcal{M})$ are co-vector fields, and $\Omega^0(\mathcal{M}) = C^\infty(\mathcal{M})$ are smooth functions $V : \mathcal{M} \to \mathbb{R}$. The exterior derivative is denoted as $\mathrm{d} : \Omega^k(\mathcal{M}) \to \Omega^{k+1}(\mathcal{M})$. For functions $V \in C^\infty(\mathcal{M} \times \mathcal{N}, \mathbb{R})$, with $x \in \mathcal{M}$, $y \in \mathcal{N}$, we denote by $(\mathrm{d}_x V)(y) \in T_x^*\mathcal{M}$ the partial differential at $x \in \mathcal{M}$. Curves $x : \mathbb{R} \to \mathcal{M}$ are denoted as $x(t)$, and their tangent vectors are denoted as $\dot{x} \in T_{x(t)}\mathcal{M}$.

A Lie group is denoted by G, its elements by g, h. The group identity is $e \in G$, and I denotes the identity matrix. The Lie algebra of G is $\mathfrak{g}$, and its dual is $\mathfrak{g}^*$. Letters $\tilde{A}, \tilde{B}$ denote vectors in the Lie algebra, while letters A, B denote vectors in $\mathbb{R}^n$.

In coordinate expressions lower indices are covariant and upper indices are contravariant components of tensors. For example, for a $(0,2)$-tensor M, the components M_{ij} are covariant, and for non-degenerate M, the components of its inverse M^{-1} are M^{ij}, which are contravariant. We use the Einstein summation convention $a_i b^i := \sum_i a_i b^i$, i.e., the product of variables with repeated lower and upper indices implies a sum.

Denoting W a topological space, D the Borel σ-algebra and $\mathbb{P} : D \to [0,1]$ a probability measure, then the tuple $(W, D, \mathbb{P})$ denotes a probability space. Given a vector space L and a random variable $C : X \to L$, then the expectation of C w.r.t. $\mathbb{P}$ is $\mathbb{E}_{w\sim\mathbb{P}}(C) := \int_W C(w)\mathrm{d}\mathbb{P}(w)$.

 doi:10.20944/preprints202506.1038.v1

2. Background

2.1. Smooth Manifolds

Given an n-dimensional *manifold* $\mathcal{M}$, with $U \subset \mathcal{M}$ an open set and $Q : U \to \mathbb{R}^n$ a homeomorphism, we call (U, Q) a *chart* and we denote the *coordinates* of $x \in U$ as

$$(q^1, \dots, q^n) := Q(x), \quad x \in U \subset \mathcal{M}. \tag{2}$$

Smooth manifolds admit charts (U_1, Q_1) and (U_2, Q_2) with smooth *transition maps* $Q_{21} = Q_2 \circ Q_1^{-1}$ defined on the intersection $U_1 \bigcap U_2$, and a collection $\mathcal{A}$ of charts (U, Q) with smooth transition maps is called a (smooth) *atlas*. A *vector field* $f \in \mathfrak{X}(\mathcal{M})$ associates to any point $x \in \mathcal{M}$ a vector in $T_x\mathcal{M}$. It defines a dynamic system

$$\dot{x} = f(x)\,;\; x(0) = x_0\,, \tag{3}$$

and we denote the solution of (3) by the *flow operator*

$$\Psi_f^t : \mathcal{M} \to \mathcal{M}\,;\; \Psi_f^t(x_0) := x(t)\,. \tag{4}$$

For a real valued function $V \in C^\infty(\mathcal{M})$, its *differential* is the covector field

$$\mathrm{d}V \in \Omega^1(\mathcal{M})\,;\; \mathrm{d}V = \frac{\partial V}{\partial q^i}\mathrm{d}q^i\,. \tag{5}$$

Given additionally a smooth manifold $\mathcal{N}$ and a smooth map $\varphi : \mathcal{N} \to \mathcal{M}$, with (U, Q) and $(\bar{U}, \bar{Q})$ appropriate charts of $\mathcal{M}$ and $\mathcal{N}$, respectively. Then the *pullback* of $\mathrm{d}V$ via φ is

$$\varphi^*\mathrm{d}V \in \Omega^1(\mathcal{N})\,;\; \varphi^*\mathrm{d}V := \mathrm{d}(V \circ \varphi) = \frac{\partial \varphi^j}{\partial \bar{q}^i}\frac{\partial V}{\partial q^j}\mathrm{d}\bar{q}^i\,. \tag{6}$$

With a *Riemannian metric* M (i.e., a symmetric, non-degenerate (0,2) tensor field) on $\mathcal{M}$, the *gradient* of V is a uniquely defined vector field $\nabla V \in \mathfrak{X}(\mathcal{M})$ given by

$$\nabla V := M^{-1}\mathrm{d}V = M^{ij}\frac{\partial V}{\partial q^j}\frac{\partial}{\partial q^i}\,. \tag{7}$$

When $\mathcal{M} = \mathbb{R}^n$, we assume that M is the Euclidean metric, and pick coordinates such that the components of the gradient and differential are the same. Finally, we define the *Lie derivative* of 1-forms, which differentiates $\omega \in \Omega^1(\mathcal{M})$ along a vector field $f \in \mathfrak{X}(\mathcal{M})$ and returns $\mathcal{L}_f\omega \in \Omega^1(\mathcal{M})$:

$$\mathcal{L}_f\omega := \frac{\mathrm{d}}{\mathrm{d}t}\left(\Psi_f^{t\,*}\omega\right)_{t=0} = \omega_j\left(\frac{\partial}{\partial q^i}f^j\right)\mathrm{d}q^i + \left(\frac{\partial}{\partial q^j}\omega_i\right)f^j\mathrm{d}q^i\,. \tag{8}$$

2.2. Lie groups

Lie groups are smooth manifolds with a compatible group structure. We consider real matrix lie groups $G \subseteq GL(m, \mathbb{R})$, i.e., subgroups of the general linear group

$$GL(m, \mathbb{R}) := \{g \in \mathbb{R}^{m\times m} \mid \det(g) \neq 0\}\,, \tag{9}$$

For $g, h \in G$ the left and right translations by h are, respectively, the matrix multiplications

$$L_h(g) := hg\,, \tag{10}$$

$$R_h(g) := gh\,. \tag{11}$$

The Lie algebra of G is the vector space $\mathfrak{g} \subseteq \mathfrak{gl}(m, \mathbb{R})$, with $\mathfrak{gl}(m, \mathbb{R}) = \mathbb{R}^{m\times m}$ the Lie algebra of $GL(m, \mathbb{R})$.

 doi:10.20944/preprints202506.1038.v1

Define a basis $E := \{\tilde{E}_1, \dots, \tilde{E}_n\}$ with $\tilde{E}_i \in \mathfrak{g} \subset \mathbb{R}^{m\times m}$, and Define the (invertible, linear) map $\Lambda : \mathbb{R}^n \to \mathfrak{g}$ as[1]

$$\Lambda : \mathbb{R}^n \to \mathfrak{g};\ (A^1, \dots, A^n) \mapsto \sum_i A^i \tilde{E}_i\,. \tag{12}$$

The dual of $\mathfrak{g}$ is denoted $\mathfrak{g}^*$, and given the map Λ we call $\Lambda^* : \mathfrak{g}^* \to \mathbb{R}^n$ its dual. For $\tilde{A}, \tilde{B} \in \mathfrak{g}$ the small adjoint $\mathrm{ad}_{\tilde{A}}(\tilde{B})$ is a bilinear map, and the large Adjoint is the linear map $\mathrm{Ad}_g(\tilde{A})$ is a linear map

$$\mathrm{ad} : \mathfrak{g} \times \mathfrak{g} \to \mathfrak{g};\ \mathrm{ad}_{\tilde{A}}(\tilde{B}) = \tilde{A}\tilde{B} - \tilde{B}\tilde{A}\,, \tag{13}$$

$$\mathrm{Ad} : G \times \mathfrak{g} \to \mathfrak{g};\ \mathrm{Ad}_g(\tilde{A}) = g\tilde{A}g^{-1}\,. \tag{14}$$

In the remainder of the article, we exclusively use the adjoint representation $\mathrm{ad}_A : \mathbb{R}^n \to \mathbb{R}^n$, written without a tilde in the subscript A, and Adjoint representation $\mathrm{Ad}_g : \mathbb{R}^n \times \mathbb{R}^n$ which are obtained as

$$\mathrm{ad}_A := \Lambda^{-1}\big(\mathrm{ad}_{\Lambda(A)}\Lambda(\cdot)\big)\,, \tag{15}$$

$$\mathrm{Ad}_g := \Lambda^{-1}\big(\mathrm{Ad}_g\Lambda(\cdot)\big) \tag{16}$$

The *exponential map* $\exp : \mathfrak{g} \to G$ is a local diffeomorphism given by the matrix exponential [32, Chapter 3.7]:

$$\exp(\tilde{A}) := \sum_{n=0}^{\infty} \frac{1}{n!}\tilde{A}^n\,. \tag{17}$$

Its inverse $\log : U_{\log} \to \mathfrak{g}$ is a given by the matrix logarithm, and it is well-defined on a subset $U_{\log} \subset G$ [32, Chapter 2.3]:

$$\log(g) = \sum_{n=1}^{\infty} (-1)^{n+1} \frac{(g-I)^n}{n}\,. \tag{18}$$

Often, these infinite sums in (17) and (18) can further be reduced to a finite sums in m terms, by use of the Cayley-Hamilton Theorem [34]. A chart (U_h, Q_h) on G that assigns zero coordinates to $h \in G$ can be defined using (18) and (12)

$$U_h = \{hg \mid g \in U_{\log}\}\,, \tag{19}$$

$$Q_h :\ U_h \to \mathbb{R}^n;\ g \mapsto \Lambda^{-1}\log(h^{-1}g)\,, \tag{20}$$

$$Q_h^{-1} :\ \mathbb{R}^n \to G;\ q \mapsto h\exp\big(\Lambda(q)\big)\,. \tag{21}$$

The chart (U_h, Q_h) is called an *exponential chart*, and a collection $\mathcal{A}$ of exponential charts (U_h, Q_h) that cover the manifold is called an *exponential atlas*.

The differential of a function $V \in C^\infty(G, \mathbb{R})$ is the co-vector field $\mathrm{d}V \in \Omega^1(G)$ (see also Equation (5)). For any given $g \in G$ we further transform the co-vector $\mathrm{d}V(g) \in T_g^*G$ to a left-trivialized differential, which collects the components of the gradient expressed in $\mathfrak{g}^*$:

$$\mathrm{d}_g^L V := \Lambda^* L_g^* \mathrm{d}V(g) = \frac{\partial}{\partial q} V\Big(g\big(I + \Lambda(q)\big)\Big)_{|q=0} \in \mathbb{R}^n\,. \tag{22}$$

For a derivation of this coordinate expression, see [21, Sec. 3].

2.3. Gradient over a flow

We will be interested in computing the gradient of functions with respect to the initial state of a flow. The adjoint sensitivity equations are a set of differential equations that achieve this. In the following, we show a derivation of the adjoint sensitivity on manifolds [21, App. A2]. Given a function

[1] Equivalently (e.g, [33]), Λ and Λ^{-1} are often denoted as operators "hat" $\wedge : \mathbb{R}^n \to \mathbb{R}^{m\times m}$ and "vee" $\vee : \mathbb{R}^{m\times m} \to \mathbb{R}^n$, respectively.

 doi:10.20944/preprints202506.1038.v1

$C : \mathcal{M} \to \mathbb{R}$, a vector field $f \in \mathfrak{X}(\mathcal{M})$, the associated flow $\Psi_f^t : \mathcal{M} \to \mathcal{M}$, and a final time $T \in \mathbb{R}$, the goal of the adjoint sensitvity method on manifolds is to compute the gradient

$$\mathrm{d}\big(C \circ \Psi_f^T\big)(x_0)\,.$$

In the adjoint method we define a co-state $\lambda(t) = \mathrm{d}(C \circ \Psi^{T-t})\big(x(t)\big) \in T^*_{x(t)}\mathcal{M}$, which represents the differential of $C\big(x(T)\big)$ with respect to $x(t)$. The adjoint sensitivity method describes its dynamics, which are integrated backwards in time from the known final condition $\lambda(T) = \mathrm{d}C\big(x(T)\big)$, see also Figure 1. The adjoint sensitivity method is stated in Theorem 1.

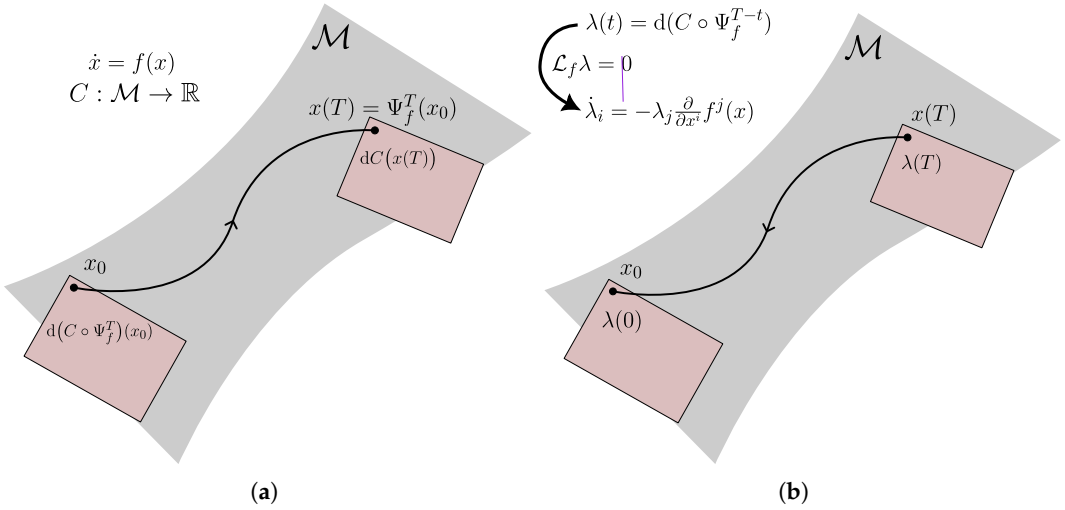

Figure 1. (**a**) The problem of computing the gradient over a flow, highlighting the cotangent spaces $\mathrm{d}C\big(x(T)\big) \in T^*_{x(T)}\mathcal{M}$ and $\mathrm{d}\big(C \circ \Psi_f^T\big)(x_0) = (\Psi_f^T)^* \mathrm{d}C\big(x(T)\big) \in T^*_{x_0}\mathcal{M}$. (**b**) In the adjoint method we set $\lambda(t) = \mathrm{d}(C \circ \Psi^{T-t})\big(x(t)\big)$, whose dynamics are uniquely determined by the property $\mathcal{L}_f \lambda = 0$, allowing to find $\lambda(0) = \mathrm{d}\big(C \circ \Psi_f^T\big)(x_0)$ by integrating $\dot{\lambda}$ backwards from $\lambda(T) = \mathrm{d}C\big(x(T)\big)$.

Theorem 1 (Adjoint sensitivity on manifolds)**.** *The gradient of a function $C \circ \Psi_f^T$ is*

$$d\big(C \circ \Psi_f^T\big)(x_0) = \lambda(0)\,, \tag{23}$$

where $\lambda(t) \in T^*_{x(t)}\mathcal{M}$ is the co-state. In a local chart (U, Q) of $\mathcal{M}$ with induced coordinates on T^*U, $x(t)$ and $\lambda(t)$ satisfy the dynamics

$$\dot{q}^j = f^j(q)\,,\ q(0) = Q(x_0)\,, \tag{24}$$

$$\dot{\lambda}_i = -\lambda_j \frac{\partial}{\partial q^i} f^j(q)\,,\ \lambda_i(T) = \frac{\partial C}{\partial q^i}\big(x(T)\big) \tag{25}$$

Proof. Define the co-state $\lambda(t) \in T^*_{x(t)}\mathcal{M}$ as

$$\lambda(t) := (\Psi_f^{T-t})^* \mathrm{d}C\big(x(T)\big)\,. \tag{26}$$

Then Equation (23) is recovered by application of Equation (6):

$$\lambda(0) = (\Psi_f^T)^* \mathrm{d}C\big(x(T)\big) = (\mathrm{d}C \circ \Psi_f^T)(x_0)\,, \tag{27}$$

 doi:10.20944/preprints202506.1038.v1

A derivation of the dynamics governing $\lambda(t)$ constitutes the remainder of this proof. By definition of $\lambda(t)$ and the Lie derivative (8), we have that $\mathcal{L}_f\lambda(t) = 0$:

$$\begin{aligned}\mathcal{L}_f\lambda(t) &= \frac{d}{ds}\big((\Psi_f^s)^*\lambda(t+s)\big)_{s=0} \\ &= \frac{d}{ds}\lambda(t) = 0\,.\end{aligned} \tag{28}$$

If we further treat λ as a 1-form $\lambda \in \Omega^1(\mathcal{M})$ (denoted as λ, by an abuse of notation), we obtain:

$$\mathcal{L}_f\lambda = \lambda_j(\frac{\partial}{\partial q^i}f^j)\mathrm{d}q^i + (\frac{\partial}{\partial q^j}\lambda_i)f^j\mathrm{d}q^i = 0\,.$$

The components satisfy the partial differential equation

$$\lambda_j\frac{\partial}{\partial q^i}f^j + f^j\frac{\partial}{\partial q^j}\lambda_i = 0\,. \tag{29}$$

Impose that $\lambda(t) = \lambda\big(\Psi_f^t(x_0)\big)$ (this defines the 1-form λ along $x(t)$), then

$$\dot{\lambda}_i = \frac{\partial\lambda_i}{\partial q^j}\dot{q}^j = \frac{\partial\lambda_i}{\partial q^j}f^j\,. \tag{30}$$

Combining Equations (29) and (30) leads to Equation (25):

$$\dot{\lambda}_i = -\lambda_j\frac{\partial}{\partial q^i}f^j\,. \tag{31}$$

Expanding the final condition $\lambda(T) = \mathrm{d}C\big(x(T)\big)$ in local coordinates (see Equation (5)) gives

$$\lambda(T) = \frac{\partial C}{\partial q^i}\big(x(T)\big)\mathrm{d}q^i = \lambda_i(T)\mathrm{d}q^i \quad \Leftrightarrow \quad \lambda_i(T) = \frac{\partial C}{\partial q^i}\big(x(T)\big)\,. \tag{32}$$

□

A fact that will become useful in Section 4, is that the equations (24) and (25) have a Hamiltonian form. Define the control Hamiltonian $H_c : T^*\mathcal{M} \to \mathbb{R}$ as

$$H_c(x,\lambda) = \lambda\big(f(x,t)\big) = \lambda_i\big(f^i(q,t)\big)\,. \tag{33}$$

Then Equation (24) and Equation (25), respectively, of Theorem 1 follow as the Hamiltonian equations on $T^*\mathcal{M}$:

$$\dot{q}^j = \frac{\partial H_c}{\partial\lambda_j} = f^j(q,t)\,, \tag{34}$$

$$\dot{\lambda}_i = -\frac{\partial H_c}{\partial q^i} = -\lambda_j\frac{\partial}{\partial q^i}f^j(q,t)\,. \tag{35}$$

For background on Hamilton's equations, see also Appendix A.1.

3. Neural ODEs on Manifolds

A neural ODE on a manifold is an NN-parameterized vector field in $\mathfrak{X}(\mathcal{M})$ – or including time dependence, it is an NN-parameterized vector field in $\mathfrak{X}(\mathcal{M}\times\mathbb{R})$, with t in the $\mathbb{R}$ slot and $\dot{t} = 1$. Given parameters $\theta \in \mathbb{R}^{n_\theta}$, we denote this parameterized vector field as $f_\theta(x,t) := f(x,t,\theta)$. This results in the dynamic system

$$\dot{x} = f_\theta(x,t)\,,\ x(0) = x_0\,, \tag{36}$$

 doi:10.20944/preprints202506.1038.v1

The key idea of neural ODEs is to tackle various flow approximation tasks, by optimising the parameters with respect to a to-be-specified optimization problem. Denote a finite time horizon T and intermittent times $T_1, T_2, \cdots < T$. Denote a general trajectory cost by

$$C^T_{f_\theta}(x_0, \theta) = F(\theta, \Psi^{T_0}_{f_\theta}(x_0), \Psi^{T_1}_{f_\theta}(x_0), \cdots, \Psi^{T}_{f_\theta}(x_0)) + \int_0^T r(\Psi^s_{f_\theta}(x_0), s)\mathrm{d}s\,, \tag{37}$$

with intermittent and final cost term F, and running cost r. Indicating a probability space $(\mathcal{M}, D, \mathbb{P})$, we define the total cost as

$$J(\theta) := \mathbb{E}_{x_0 \sim \mathbb{P}} C^T_{f_\theta}(x_0, \theta)\,. \tag{38}$$

The minimization problem takes the form

$$\min_\theta \; J(\theta)\,. \tag{39}$$

Note that (39) is not subject to any dynamic constraint - the flow already appears explicitly in the cost $C^T_{f_\theta}$.

Normally, the optimization problem is solved by means of a stochastic gradient descent optimization algorithm [35]. In this, a batch of N initial conditions x_i is sampled from the probability distribution corresponding to the probability measure $\mathbb{P}$. Writing $C_i = C^T_{f_\theta}(x_i, \theta)$, the parameter gradient $\frac{\partial}{\partial \theta} J(\theta)$ is approximated as

$$\frac{\partial}{\partial \theta} J(\theta) = \mathbb{E}_{x_0 \sim \mathbb{P}} \frac{\partial}{\partial \theta} C^T_{f_\theta}(x_0) \approx \frac{1}{N} \sum_{i=0}^{N} \frac{\partial}{\partial \theta} C_i\,. \tag{40}$$

In this section, we show how to optimize the parameters θ for various choices of neural ODEs and cost functions, with (37) the most general case of a cost, and highlight similarities in the various derivations. In the following, the gradient $\frac{\partial}{\partial \theta} C_i$ is computed via the adjoint method on manifolds, for various scenarios. The advantage of the adjoint method over e.g., automatic differentiation of C_i / back-propagation through an ODE solver, is that it has a constant memory efficiency with respect to the network depth T.

3.1. Constant parameters, running and final cost

Here we consider neural ODEs of the form (36), with constant parameters θ, and cost functions of the form

$$C^T_{f_\theta}(x_0, \theta) = F(\Psi^T_{f_\theta}(x_0), \theta) + \int_0^T r(\Psi^s_{f_\theta}(x_0), \theta, s)\mathrm{d}s\,, \tag{41}$$

with a final cost term F and a running cost term r. This generalizes [1,2] to manifolds. Compared to existing manifold methods for neural ODES [18,19], the running cost is new.

The parameter gradient's components $\frac{\partial}{\partial \theta} C^T_{f_\theta}((x_0, t_0), \theta) \in \mathbb{R}^{n_\theta}$ are then computed by Theorem 2 (see also [21]):

Theorem 2 (Generalized Adjoint Method on Manifolds)**.** *Given the dynamics* (36) *and the cost* (41)*, the parameter gradient's components* $\frac{\partial}{\partial \theta} C^T_{f_\theta}((x_0, t_0), \theta) \in \mathbb{R}^{n_\theta}$ *are computed by*

$$\frac{\partial}{\partial \theta} C^T_{f_\theta}((x_0, t_0), \theta) = (\frac{\partial F}{\partial \theta})(x(T), \theta) + \int_0^T \frac{\partial}{\partial \theta}(\lambda_j f^j_\theta(q(s)) + r(q(s), \theta, s))ds\,. \tag{42}$$

 doi:10.20944/preprints202506.1038.v1

where the state $x(s) \in \mathcal{M}$ and co-state $\lambda(s) \in T^*_{x(s)}\mathcal{M}$ satisfy, in a local chart (U, Q) with $q(t) = Q(x(t))$, $\lambda(t) = \lambda_i(t)dq^i$

$$\dot{q}^j = f^j_\theta(q,t)\,,\ q(0) = Q(x_0)\,,\ t(0) = t_0\,, \tag{43}$$

$$\dot{\lambda}_i = -\lambda_j \frac{\partial}{\partial q^i} f^j_\theta(q,t) - \frac{\partial r}{\partial q^i}\,,\ \lambda_i(T) = \frac{\partial F}{\partial q^i}(x(T),\theta)\,. \tag{44}$$

Proof. Define the augmented state space as $\mathcal{M}' = \mathcal{M} \times \mathbb{R}^{n_\theta} \times \mathbb{R} \times \mathbb{R}$, to include the original state $x \in \mathcal{M}$, parameters $\theta \in \mathbb{R}^{n_\theta}$, accumulated running cost $L \in \mathbb{R}$ and time $t \in \mathbb{R}$ in the augmented state $x' := (x, \theta, L, t) \in \mathcal{M}'$. In addition, define the augmented dynamics $f_{\text{aug}} \in \mathfrak{X}(\mathcal{M}')$ as

$$\dot{x}' = f_{\text{aug}}(x') = \begin{pmatrix} f_\theta(x,t) \\ 0 \\ r(x,\theta,t) \\ 1 \end{pmatrix}\,,\ x'(0) = x'_0 := \begin{pmatrix} x_0 \\ \theta \\ 0 \\ t_0 \end{pmatrix}\,. \tag{45}$$

This is an autonomous system with final state $x'(T) = (x(T), \theta, \int_0^T r(x,\theta,s)\mathrm{d}s, T)$. Next, define the cost $C_{\text{aug}} : \mathcal{M}' \to \mathbb{R}$ on the augmented space:

$$C_{\text{aug}}(x') = F(x,\theta) + L\,. \tag{46}$$

Then Equation (41) can be rewritten as the evaluation of a terminal cost $C_{\text{aug}}(x'(T))$:

$$C^T_{f_\theta}(x_0) = (C_{\text{aug}} \circ \Psi^T_{f_{\text{aug}}})(x'_0)\,. \tag{47}$$

By Theorem 1, the gradient $\mathrm{d}\left(C_{\text{aug}} \circ \Psi^T_{f_{\text{aug}}}\right)$ is given by

$$\mathrm{d}(C_{\text{aug}} \circ \Psi^T_{f_{\text{aug}}})(x'_0) = \lambda(0)\,, \tag{48}$$

and by Equation (25), the componentents of $\lambda(s)$ satisfy

$$\dot{\lambda}_i = -\lambda_j \frac{\partial}{\partial q^{i'}} f^j_{\text{aug}}\,,\ \lambda_i(T) = \frac{\partial}{\partial q^{i'}} C_{\text{aug}}(x'(T)) \tag{49}$$

Split the co-state into $\lambda_q, \lambda_\theta, \lambda_L, \lambda_t$, then their components' dynamics are:

$$\dot{\lambda}_{q,i} = -\frac{\partial}{\partial q^i}\left(\lambda_{q,j} f^j_\theta(q,t) + \lambda_L r(q,\theta,t)\right)\,,\ \lambda_q(T) = \frac{\partial F}{\partial q}(x(T),\theta)\,, \tag{50}$$

$$\dot{\lambda}_{\theta,i} = -\frac{\partial}{\partial \theta^i}\left(\lambda_{q,j} f^j_\theta(q,t) + \lambda_L r(q,\theta,t)\right)\,,\ \lambda_\theta(T) = \frac{\partial F}{\partial \theta}(x(T),\theta) \tag{51}$$

$$\dot{\lambda}_L = 0\,,\ \lambda_L(T) = \frac{\partial}{\partial L} C_{\text{aug}}(x(T),\theta) = 1 \tag{52}$$

$$\dot{\lambda}_t = -\frac{\partial}{\partial t}\left(\lambda_{q,j} f^j_\theta(q,t) + \lambda_L r(q,\theta,t)\right)\,,\ \lambda_t(T) = \frac{\partial}{\partial t} C_{\text{aug}}(x(T),\theta) = 0\,. \tag{53}$$

The component $\lambda_L = 1$ is constant, so equation (50) coincides with (44). Integrating (51) from $s = 0$ to $s = T$ recovers Equation (42). λ_t does not appear in any of the other equations, such that Equation (53) may be ignored. □

In summary, the above proof dependeds on identifying a suitable augmented manifold $\mathcal{M}'$, with the goal that augmented dynamics $f_{\text{aug}} \in \mathfrak{X}(\mathcal{M}')$ are autonomous, and that the cost-function $C_{\text{aug}} : \mathcal{M}' \to \mathbb{R}$ on the augmented manifold rephrases the cost (41) as a final cost $C_{\text{aug}}(x(T))$, which allows to apply Theorem 1, and express any gradients of the original cost. In later sections (Sec. 3.2),

 doi:10.20944/preprints202506.1038.v1

this process will be the main technical tool for generalizations of Theorem (2). The next sections describe common special cases of (36) and Theorem 2.

3.1.1. Vanilla neural ODEs and extrinsic neural ODEs on manifolds

The case of neural ODEs on $\mathbb{R}^n$ (e.g., [1,2]) is obtained by setting $\mathcal{M} = \mathbb{R}^n$. In this case, one global chart can be used to represent all quantities.

This overlaps with extrinsic neural ODEs on manifolds (described, for instance, in [22]), which optimize the neural ODE on an embedding space $\mathbb{R}^N$. We denote this embedding as $\iota : \mathcal{M} \to \mathbb{R}^N$, let $x \in \mathcal{M}$ and $y \in \mathbb{R}^N$. Optimizing the neural ODE on $\mathbb{R}^N$ requires extending the dynamics $f_\theta(x,t) \in \mathfrak{X}(\mathcal{M})$ to a vector field $f_\theta^\uparrow(y,t) \in \mathfrak{X}(\mathbb{R}^N)$, such that

$$\iota_* f_\theta(x,t) = f_\theta^\uparrow(\iota(x),t) \tag{54}$$

The dynamics $f_\theta^\uparrow(y,t)$ are then used in Theorem 2, and also the costate lives in $T^*\mathbb{R}^N$.

As shown in [22], the resulting parameter gradients are equivalent to those resulting from an application in local charts, as long as it can be guaranteed that the integral curves of $f^\uparrow(y,t) \in X(\mathbb{R}^N)$ remain within $\iota(\mathcal{M}) \subset \mathbb{R}^N$, i.e., are geometrically preserving.

A strong upside of an extrinsic formulation is that existing neural ODE packages (e.g., [36]) can be applied directly. Possible downsides of extrinsic neural ODEs are that finding $f^\uparrow(y,t) \in X(\mathbb{R}^N)$ may not be immediate, a geometrically preserving integration has to be guaranteed separately, and that N is larger than the intrinsic dimension $n = \dim \mathcal{M}$, leading to computational overhead.

3.1.2. Intrinsic neural ODEs on manifolds

The intrinsic case of neural ODEs on manifolds [18] is described by integrating the dynamics in local charts. Given a chart-transition from a chart (U_1, Q_1) to a chart (U_2, Q_2) chart transitions of the state and costate components ($q_1^i, \lambda_{1,i}$ and $q_2^i, \lambda_{2,i}$, respectively) are given by

$$q_2^i = Q_2^i \circ Q_1^{-1}(q_1)\,, \tag{55}$$

$$\lambda_{i,2} = A_i^j \lambda_{j,1}\,. \tag{56}$$

with $A_i^j = \partial_i (Q_1^j \circ Q_2^{-1})$. The advantage of intrinsic neural ODEs on manifolds over extrinsic neural ODEs on manifolds is that the dimension of the resulting equations is as low as possible, for the given manifold. A disadvantage lies in having to determine charts, and chart-switching procedures. In available state-of-the-art packages for neural ODEs, these are phrased as discontinuous dynamics with state transitions $Q_1^j \circ Q_2^{-1} : \mathbb{R}^n \to \mathbb{R}^n$. For details on chart-switching methods see [18,21,29].

3.2. Extensions

The proof of Theorem 2 depended on identifying a suitable augmented manifold $\mathcal{M}'$, autonomous augmented dynamics $f_{\text{aug}} \in \mathfrak{X}(\mathcal{M}')$ and augmented cost-function $C_{\text{aug}} : \mathcal{M}' \to \mathbb{R}$ that rephrases the cost (41) as a final cost $C_{\text{aug}}(x(T))$, to apply Theorem 1. This approach generalizes to various other scenarios, including different cost-terms, augmented neural ODEs on manifolds and time-dependent parameters, presented in the following.

3.2.1. Nonlinear and intermittent cost terms

We consider here the case of neural ODEs on manifolds of the form (36) with cost (37). For the final and intermittent cost term $F_\theta : \mathcal{M} \times \mathcal{M} \times \cdots \times \mathcal{M} \to \mathbb{R}$ we denote by $\mathrm{d}_k F_\theta \in T_x^*\mathcal{M}$ the differential w.r.t. the k-th slot, and denote θ as a subscript to avoid confusion. The components of $\mathrm{d}_k F$ will be denoted $\frac{\partial F}{\partial^k q^i}$. In this case, the parameter gradient is determined by repeated application of Theorem 2:

 doi:10.20944/preprints202506.1038.v1

Theorem 3 (Generalized Adjoint Method on Manifolds). *Given the dynamics (36) and the cost (37), the parameter gradient's components* $\frac{\partial}{\partial\theta}C^T_{f_\theta}((x_0,t_0),\theta) \in \mathbb{R}^{n_\theta}$ *are computed by*

$$\begin{aligned}\frac{\partial}{\partial\theta}C^T_{f_\theta}((x_0,t_0),\theta) =& \left(\frac{\partial F}{\partial\theta}\right)(\theta, x(T_1), x(T_2), \cdots, x(T)) \\ &+ \int_0^T \frac{\partial}{\partial\theta}\left(\lambda_j f^j_\theta(q(s)) + r(q(s),\theta,s)\right)ds\,.\end{aligned} \tag{57}$$

where the state $x(s) \in \mathcal{M}$ *satisfies (43) and the co-state* $\lambda(s) \in T^*_{x(s)}\mathcal{M}$ *satisfies dynamics with discrete updates at times* $T_1, \cdots, T_{N-1}$ *given by*

$$\dot\lambda_{q,i} = \frac{\partial}{\partial q^i}\left(\lambda_{q,j} f^j_\theta(q,t) + r(q_N,\theta,t)\right);\ \lambda_{q,i}(T) = \frac{\partial F_\theta}{\partial^N q^i}(x(T_1),\cdots,x(T)) \tag{58}$$

$$\lambda_i(T_{k,-}) = \lambda_i(T_{k,+}) + \frac{\partial F_\theta}{\partial^k q^i}\left(x(T_1),\cdots,x(T)\right), \tag{59}$$

with $T_{k,-}$ *the instance after a discrete update at time* T_k *(recall that co-state dynamics are integrated backwards, so* $T_{k,-} < T_k < T_{k,+}$*), and* $T_{k,+}$ *the instance before.*

Proof. We introduce an augmented manifold $\mathcal{M}' = \mathcal{M}\times\cdots\times\mathcal{M}\times\mathbb{R}^{n_\theta}\times\mathbb{R}\times\mathbb{R}$, to include N copies of the original state $x \in \mathcal{M}$, parameters $\theta \in \mathbb{R}^{n_\theta}$, accumulated running cost $L \in \mathbb{R}$ and time $t \in \mathbb{R}$ in the augmented state $x' := (x_1,\cdots,x_N,\theta,L,t) \in \mathcal{M}'$. Let

$$\varrho_{T_i}(t) = \begin{cases} 1 & t \le T_i \\ 0 & t > T_i \end{cases}, \tag{60}$$

and define the augmented dynamics $f_{\text{aug}} \in \mathfrak{X}(\mathcal{M}')$ as

$$\dot x' = f_{\text{aug}}(x') = \begin{pmatrix} \varrho_{T_1}(t) f_\theta(x_1,t) \\ \vdots \\ \varrho_{T_{N-1}}(t) f_\theta(x_{N-1},t) \\ f_\theta(x_N,t) \\ 0 \\ r(x_N,\theta,t) \\ 1 \end{pmatrix},\ x'(0) = x'_0 := \begin{pmatrix} x_0 \\ \vdots \\ x_0 \\ x_0 \\ \theta \\ 0 \\ t_0 \end{pmatrix}. \tag{61}$$

This is an autonomous system with final state

$$x'(T) = (x(T_1),\cdots,x(T_{N-1}),x(T),\theta,\int_0^T r(x,\theta,s)\mathrm{d}s,T)\,. \tag{62}$$

Next, define the cost $C_{\text{aug}} : \mathcal{M}' \to \mathbb{R}$ on the augmented space:

$$C_{\text{aug}}(x') = F_\theta(x_1,\cdots,x_N) + L\,. \tag{63}$$

Then Equation (37) can be rewritten as the evaluation of a terminal cost $C_{\text{aug}}(x'(T))$:

$$C^T_{f_\theta}(x_0) = (C_{\text{aug}} \circ \Psi^T_{f_{\text{aug}}})(x'_0)\,. \tag{64}$$

 doi:10.20944/preprints202506.1038.v1

Apply Equation (25), and split the co-state into $\lambda_{q_1}, \cdots, \lambda_{q_N}, \lambda_\theta, \lambda_L, \lambda_t$, then their components' dynamics are:

$$\dot{\lambda}_{q_1,i} = -\frac{\partial}{\partial q^i}\left(\lambda_{q_1,j}\varrho_{T_1}(t) f_\theta^j(q_1,t)\right), \ \lambda_{q_1}(T) = \frac{\partial F_\theta}{\partial^1 q}(x(T_1),\cdots,x(T)), \tag{65}$$

$$\vdots$$

$$\dot{\lambda}_{q_N,i} = -\frac{\partial}{\partial^N q^i}\left(\lambda_{q_N,j} f_\theta^j(q_N,t) + \lambda_L r(q_N,\theta,t)\right), \ \lambda_{q_N}(T) = \frac{\partial F_\theta}{\partial^N q}(x(T_1),\cdots,x(T)), \tag{66}$$

$$\dot{\lambda}_{\theta,i} = -\frac{\partial}{\partial \theta^i}\left(\lambda_{q_1,j}\varrho_{T_1}(t) f_\theta^j(q_1,t) + \cdots + \lambda_{q_N,j} f_\theta^j(q_N,t) + \lambda_L r(q,\theta,t)\right), \tag{67}$$

$$\lambda_\theta(T) = \frac{\partial F_\theta}{\partial \theta}(x(T_1),\cdots,x(T)).$$

We excluded the dynamics of λ_t, which does not appear in any of the other equations, and the constant $\lambda_L = 1$. Finally, define the cumulative co-state

$$\lambda_q = \varrho_{T_1}(t)\lambda_{q_1} + \cdots + \lambda_{q_N}. \tag{68}$$

Its dynamics at $t \in [0,T]$ $T_1, \cdot, T_{N-1}$ are given by the sum of (65) to (66), letting $q = q_N$:

$$\dot{\lambda}_{q,i} = \dot{\lambda}_{q_1,i} + \cdots + \dot{\lambda}_{q_N,i} \tag{69}$$

$$= \frac{\partial}{\partial q^i}\left(\lambda_{q,j} f_\theta^j(q,t) + r(q_N,\theta,t)\right) \tag{70}$$

$$\lambda_q(T) = \frac{\partial F_\theta}{\partial^N q}(x(T_1),\cdots,x(T)), \tag{71}$$

with discrete jumps (58) accounting for the final conditions of $\lambda_{q_1}, \cdots, \lambda_{q_N}$, and the dynamics (67) can be rewritten as

$$\dot{\lambda}_{\theta,i} = \frac{\partial}{\partial \theta^i}\left(\lambda_{q,j} f_\theta^j(q,t) + r(q,\theta,t)\right); \ \lambda_\theta(T) = \frac{\partial F_\theta}{\partial \theta}(x(T_1),\cdots,x(T)). \tag{72}$$

Integrating this from $s = 0$ to $s = T$ recovers Equation (57). □

Cost terms of this form are interesting for optimiziation of e.g., periodic orbits [37] or trajectories on manifolds, where conditions at multiple checkpoints $\Psi_{f_\theta}^{T_i}(x_0)$ may appear in the cost.

3.2.2. Augmented neural ODEs on manifolds and time-dependent parameters

With state $x \in \mathcal{M}$, augmented state $\alpha \in \mathcal{N}$ (not to be confused with $x' \in \mathcal{M}'$), and parameterized $\varphi_\theta : \mathcal{M} \to \mathcal{N}$, augmented neural ODEs on manifolds are neural ODEs on the manifold $\mathcal{M} \times \mathcal{N}$ of the form

$$\begin{pmatrix} \dot{x} \\ \dot{\alpha} \end{pmatrix} = \begin{pmatrix} f_\theta(x,\alpha) \\ g_\theta(x,\alpha) \end{pmatrix}; \ \begin{pmatrix} x(0) \\ \alpha(0) \end{pmatrix} = \begin{pmatrix} x_0 \\ \varphi_\theta(x_0) \end{pmatrix}. \tag{73}$$

Time t is not included explicitly in these dynamics, since it can be included in α. This case also includes the scenario of time-dependent parameters $\bar{\theta}(t)$ as part of α. As the trajectory cost, we take a final cost

$$C_{f_\theta,g_\theta}^T(x_0,\theta) = F(\Psi_{f_\theta,g_\theta}^T(x_0,\varphi_\theta(x_0)),\theta). \tag{74}$$

 doi:10.20944/preprints202506.1038.v1

Theorem 4 (Adjoint Method for Augmented Neural ODEs on Manifolds)**.** *Given the dynamics* (73) *and the cost* (74)*, the parameter gradient's components* $\frac{\partial}{\partial\theta}C^T_{f_\theta}\big((x_0,t_0),\theta\big)\in\mathbb{R}^{n_\theta}$ *are computed by*

$$\begin{aligned}\frac{\partial}{\partial\theta}C^T_{f_\theta,g_\theta}\big((x_0,\varphi(x_0)),\theta\big) =&\big(\frac{\partial F}{\partial\theta}\big)(x(T),\alpha(T),\theta)+\frac{\partial\varphi^j}{\partial\theta}\lambda_{\alpha,j}(0)\\ &+\int_0^T\frac{\partial}{\partial\theta}\big(\lambda_{x,j}f^j_\theta(q(s))+\lambda_{\alpha,j}g^j_\theta(q(s))\big)ds\,.\end{aligned}\tag{75}$$

where the states $x(s)\in\mathcal{M},\alpha(s)\in\mathcal{N}$ *satisfy* (73) *and co-states* $\lambda_x(s)\in T^*_{x(s)}\mathcal{M},\lambda_\alpha(s)\in T^*_{\alpha(s)}\mathcal{N}$*, satisfy, in a local chart* (U,Q) *on* $\mathcal{M}$ *and* $\bar U,\bar Q$ *on* $\mathcal{N}$*:*

$$\dot\lambda_{x,i}=-\frac{\partial}{\partial q^i}\big(\lambda_{x,j}f^j_\theta(q,\bar q,t)+\lambda_{\alpha,j}g^j_\theta(q,\bar q,t)\big)\,,\ \lambda_{x,i}(T)=\frac{\partial F}{\partial q^i}\big(x(T),\alpha(T),\theta\big)\,,\tag{76}$$

$$\dot\lambda_{\alpha,i}=-\frac{\partial}{\partial \bar q^i}\big(\lambda_{x,j}f^j_\theta(q,\bar q,t)+\lambda_{\alpha,j}g^j_\theta(q,\bar q,t)\big)\,,\ \lambda_{\alpha,i}(T)=\frac{\partial F}{\partial \bar q^i}\big(x(T),\alpha(T),\theta\big)\,.\tag{77}$$

Proof. Define the augmented state space as $\mathcal{M}'=\mathcal{M}\times\mathcal{N}\times\mathbb{R}^{n_\theta}$, to include the states $x\in\mathcal{M},\alpha(s)\in\mathcal{N}$ and parameters $\theta\in\mathbb{R}^{n_\theta}$ in the augmented state $x':=(x,\alpha,\theta)\in\mathcal{M}'$. In addition, define the augmented dynamics $f_{\text{aug}}\in\mathfrak{X}(\mathcal{M}')$ as

$$\dot x'=f_{\text{aug}}(x')=\begin{pmatrix}f_\theta(x,\alpha)\\ g_\theta(x,\alpha)\\ 0\end{pmatrix}\,,\ x'(0)=x'_0:=\begin{pmatrix}x_0\\ \varphi_\theta(x_0)\\ \theta\end{pmatrix}\,.\tag{78}$$

This is an autonomous system with final state $x'(T)=(x(T),\alpha(T),\theta)$. Next, define the cost $C_{\text{aug}}:\mathcal{M}'\to\mathbb{R}$ on the augmented space:

$$C_{\text{aug}}(x')=F(x,\alpha,\theta)\,.\tag{79}$$

Then Equation (41) can be rewritten as the evaluation of a terminal cost $C_{\text{aug}}\big(x'(T)\big)$. The gradient $\mathrm{d}\big(C_{\text{aug}}\circ\Psi^T_{f_{\text{aug}}}\big)$ is given by an application of Equation (25). Split the co-state into $\lambda_x,\lambda_\alpha,\lambda_\theta$, then their components' dynamics are:

$$\dot\lambda_{x,i}=-\frac{\partial}{\partial q^i}\big(\lambda_{x,j}f^j_\theta(q,\bar q,t)+\lambda_{\alpha,j}g^j_\theta(q,\bar q,t)\big)\,,\ \lambda_x(T)=\frac{\partial F}{\partial q}(x(T),\alpha(T),\theta)\,,\tag{80}$$

$$\dot\lambda_{\alpha,i}=-\frac{\partial}{\partial \bar q^i}\big(\lambda_{x,j}f^j_\theta(q,\bar q,t)+\lambda_{\alpha,j}g^j_\theta(q,\bar q,t)\big)\,,\ \lambda_{\alpha,i}(T)=\frac{\partial F}{\partial \bar q^i}\big(x(T),\alpha(T),\theta\big)\,.\tag{81}$$

$$\dot\lambda_{\theta,i}=-\frac{\partial}{\partial \theta^i}\big(\lambda_{x,j}f^j_\theta(q,\bar q,t)+\lambda_{\alpha,j}g^j_\theta(q,\bar q,t)\big)\,,\ \lambda_\theta(T)=\frac{\partial F}{\partial \theta}(x(T),\alpha(T),\theta)\tag{82}$$

Since $\alpha(0)=\varphi_\theta(x_0)$ also depends on θ, the total gradient of the cost w.r.t. θ is given by

$$\frac{\partial}{\partial\theta^i}C^T_{f_\theta}\big((x_0,\varphi_\theta(x_0)),\theta\big)=\lambda_{\theta,i}(0)+\frac{\partial\varphi^j}{\partial\theta^i}\lambda_{\alpha,j}(0)\,.\tag{83}$$

This recovers Equation (75). □

A further degenerate application of Theorem 4 is obtained by removing x. Then both dynamics $g_\theta(\alpha)$ and initial condition $\alpha(0)=\varphi_\theta(0)$ are parameterized by θ, allowing joint optimization of parameters and initial condition. This is interesting for joint optimization and numerical continuation, e.g. [37].

 doi:10.20944/preprints202506.1038.v1

4. Neural ODEs on Lie Groups

Just as a neural ODE on a manifold is an NN-parameterized vector field in $\mathfrak{X}(\mathcal{M})$ (or, including time, $\mathfrak{X}(\mathcal{M} \times \mathbb{R})$), a neural ODE on a Lie group can be seen as a parameterized vector field in $\mathfrak{X}(G)$ (or $\mathfrak{X}(G \times \mathbb{R})$, respectively). Similarly to Equation (36), this results in a dynamic system

$$\dot{g} = f_\theta(g, t)\,,\ g(0) = g_0\,, \tag{84}$$

Yet, Lie groups offers more structure than manifolds: the Lie algebra $\mathfrak{g}$ provides a canonical space to represent tangent vectors, and its dual $\mathfrak{g}^*$ provides a canonical space to represent the co-state. Similarly, canonical (exponential) charts offer structure for integrating dynamic systems [38]. Frequently, dynamics on a Lie group induced dynamics on a manifold $\mathcal{M}$: by means of an action

$$\Phi : G \times \mathcal{M} \to \mathcal{M}\,;\ (g, x) \mapsto \Phi(g, x)\,, \tag{85}$$

evolutions $g(t)$ induce evolutions $x(t) = \Phi(g(t), x_0)$ on $\mathcal{M}$. This makes neural ODEs on Lie groups interesting in their own right.

In this section, we describe optimizing (39) for the cost

$$C^T_{f_\theta}(g_0, \theta) = F(\Psi^T_{f_\theta}(g_0), \theta) + \int_0^T r(\Psi^s_{f_\theta}(g_0), \theta, s)\mathrm{d}s\,, \tag{86}$$

with a final cost term F and a running cost term r. We highlight the extrinsic approach, and two intrinsic approaches, where one of the latter is peculiar to Lie groups.

4.1. Extrinsic neural ODEs on Lie groups

The extrinsic formulation of neural ODEs on Lie groups was first introduced by [20], and applies ideas of [19] (see also Section 3.1.1). Given $G \subset GL(m, \mathbb{R})$, this formulation treats the dynamic system (84) as a dynamic system on $\mathbb{R}^{m^2}$. Denote $\mathrm{vec} : \mathbb{R}^{m\times m} \to \mathbb{R}^{m^2}$ an invertible map that stacks the components of an input matrix matrix into a component vector[2] and let $\mathrm{proj}_G : \mathbb{R}^{m\times m} \to G$ be a projection onto $G \subset \mathbb{R}^{m\times m}$. Further denote $A_y = \mathrm{vec}^{-1}(y)$ and $g_y = \mathrm{proj}_G(A_y)$. A lift $f^\uparrow_\theta(y, t)$ can then be defined as

$$f^\uparrow_\theta(y, t) = \mathrm{vec}\big(A_y g_y^{-1} f(g_y, \theta, t)\big)\,. \tag{87}$$

As was the case for extrinsic neural ODEs on manifolds, the cost-gradient resulting from this optimization is well-defined and equivalent to any intrinsically defined procedure. However, dimension m^2 of the vectorization can be significantly larger than the intrinsic dimension of the Lie group.

4.2. Intrinsic neural ODEs on Lie groups

Theorem 2 directly applies to optimization of neural ODEs on Lie groups, given the local exponential charts (19), (20) on G. This does not make full use of the available structure on Lie groups. Frequently, dynamical systems are of a left-invariant form (88) or a right-invariant form (89)

$$\dot{g} = g\Lambda\big(\rho^L_\theta(g, t)\big)\,, \tag{88}$$
$$\dot{g} = \Lambda\big(\rho^R_\theta(g, t)\big)g\,. \tag{89}$$

Denoting $K(q) : T_q\mathbb{R}^n \to \mathbb{R}^n$ the derivative of the exponential map (see [21] for details). Then the chart-representatives f^i_θ in a local exponential chart (U_h, Q_h) are

$$f^{L,i}_\theta(q, t) = (K^{-1})^i_j(q)\rho^{L,j}(Q_h^{-1}(q))\,, \tag{90}$$
$$f^{R,i}_\theta(q, t) = (K^{-1})^i_j(q)\mathrm{Ad}_{Q_h^{-1}(q)}\rho^{R,j}(Q_h^{-1}(q))\,. \tag{91}$$

[2] in canonical coordinates on $\mathbb{R}^{m\times m}$ and $\mathbb{R}^{m^2}$, though this choice is not required.

 doi:10.20944/preprints202506.1038.v1

Application of Theorem 2 then requires computing $\frac{\partial}{\partial q^j} f_\theta^{L,i}(q,t)$ or $\frac{\partial}{\partial q^j} f_\theta^{R,i}(q,t)$. But this leads to significant computational overhead due differentiation of the terms $(K^{-1})^i_j(q)$ (see [21]). Instead of applying Theorem 2, i.e., expressing dynamics in local charts, the dynamics can also be expressed at the Lie algebra $\mathfrak{g}$. Theorem 25 has a Hamiltonian form, which can be directly transformed into Hamiltonian equations on a Lie group (see also Appendix A.1). Applying this reasoning to Theorem 2, we arrive at the following form, which foregoes differentiating $(K^{-1})^i_j(x)$:

Theorem 5 (Left Generalized Adjoint Method on Matrix Lie Groups)**.** *Given are the dynamics* (88) *and the cost* (86)*, or the dynamics* (89) *with* $\rho_\theta^L(g,t) = Ad_{g^{-1}}\rho_\theta^R(g,t)$*. Then the parameter gradient* $\frac{\partial}{\partial\theta} C_{f_\theta}^T(g_0)$ *of the cost is given by the integral equation*

$$\frac{\partial}{\partial\theta} C_{f_\theta}^T(g_0) = \frac{\partial F}{\partial\theta}(g(T),\theta) + \int_0^T \frac{\partial}{\partial\theta}\left(\lambda_g^\top \rho_\theta^L(g,s) + r(g,\theta,s)\right) ds\,, \tag{92}$$

where the state $g(t) \in G$ *and co-state* $\lambda_g(t) \in \mathbb{R}^n$ *are the solutions of the system of equations*

$$\dot{g} = f_\theta(g,t)\,,\ g(0) = g_0\,, \tag{93}$$

$$\dot{\lambda}_g = -d_g^L\left(\lambda_g^\top \rho_\theta^L(g,s) + r(g,\theta,s)\right) + ad^\top_{\rho_\theta^L(g,t)}\lambda_g\,,\ \lambda_g(T) = d_g^L F(g(T),\theta)\,. \tag{94}$$

Proof. This is proven in two steps. First, define the time-and-parameter-dependent control-Hamiltonian $H_c : T^*\mathcal{M} \times \mathbb{R}^{n_\theta} \times \mathbb{R} \to \mathbb{R}$ as

$$H_c(x,\lambda,\theta,t) = \lambda\left(f_\theta(x,t)\right) + r(x,\theta,t) = \lambda_i\left(f_\theta^i(q,t)\right) + r(q,\theta,t)\,. \tag{95}$$

The equations for the state- and co-state dynamics (43) and (44), respectively, of Theorem 2 follow as the Hamiltonian equations on $T^*\mathcal{M}$:

$$\dot{q}^j = \frac{\partial H_c}{\partial\lambda_j} = f_\theta^j(q,t)\,, \tag{96}$$

$$\dot{\lambda}_i = -\frac{\partial H_c}{\partial q^i} = -\lambda_j \frac{\partial}{\partial q^i} f_\theta^j(q,t) - \frac{\partial r}{\partial q^i}\,. \tag{97}$$

And the integral equation (42) reads

$$\frac{\partial}{\partial\theta} C_{f_\theta}^T\left((x_0,t_0),\theta\right) = \frac{\partial F}{\partial\theta}(x(T),\theta) + \int_0^T \frac{\partial H_c}{\partial\theta}\mathrm{d}t\,. \tag{98}$$

Second, rewrite the control Hamiltonian (95) on a Lie group G, i.e., $H_c : T^*G \times \times \mathbb{R}^{n_\theta} \times \mathbb{R} \to \mathbb{R}$. By substituting $\lambda_g(t) = \Lambda^* L_g^* \lambda(t)$ (see also Equation (A6)), this induces $\mathcal{H}_c : G \times \mathfrak{g}^* \times \mathbb{R}^{n_\theta} \times \mathbb{R} \to \mathbb{R}$

$$\mathcal{H}_c(g,\lambda_g,\theta,t) = \lambda_g^\top \rho_\theta^L(g,t) + r(g,\theta,t)\,. \tag{99}$$

Finally Hamilton's equations (96), (97) are rewritten in their form on a matrix Lie group by means of (A7), (A8), which recovers equations (93) and (94):

$$\dot{g} = g\Lambda\left(\frac{\partial\mathcal{H}_c}{\partial\lambda_g}\right), \tag{100}$$

$$\dot{\lambda}_g = -\mathrm{d}_g^L\mathcal{H}_c + \mathrm{ad}^\top_{\frac{\partial\mathcal{H}_c}{\partial\lambda_g}}\lambda_g\,, \tag{101}$$

To find the final condition for λ_g, use that $\lambda_g(t) = \Lambda^* L_g^* \lambda(t)$:

$$\lambda_g(T) = \Lambda^* L_g^* \lambda(T) = \Lambda^* L_g^* \mathrm{d}F(g(T),\theta) = d_g^L F(g(T),\theta)\,. \tag{102}$$

 doi:10.20944/preprints202506.1038.v1

□

Similar equations also hold on abstract (non-matrix) Lie groups, see [21]. Compared to the extrinsic method of Section 4.1, Theorem 5 has the advantage that the dimension of the co-state λ_g is as low as possible. Compared to the chart-based approach on Lie groups, Theorem 5 foregoes differentiating through the terms $K^i_j(q)$, avoiding overhead. Compared to a chart-based approach on manifolds, also the choice of charts is canonical on Lie groups. Although the Lie-group approach foregoes many of the pitfalls of intrinsic neural ODEs on manifolds, implementation in existing neural ODE packages is currently cumbersome: the adjoint-sensitivity equations (94) have a non-standard form, requiring an adapted dynamics of the co-state λ, but these equations are rarely intended for modification, in existing packages. Packages for geometry-preserving integrators on Lie groups, such as [38] are also not readily available for arbitrary Lie groups.

4.3. Extensions

The proof of Theorem 5 relied on finding a control-Hamiltonian formulation for Theorem 2. This approach generalizes to methods in Section 3.2, which rely on the use of Theorem 1. That is because Theorem 1 itself has a Hamiltonian form ([19,21]).

5. Discussion

We discuss advantages and disadvantages of the main flavors of the presented formulations for manifold neural ODEs, expanding on the previous sections. We focus on extrinsic (embedding dynamics in $\mathbb{R}^N$) and intrinsic (integrating in local charts) formulations. Summarizing prior comments:

- the extrinsic formulation is readily implemented if the low-dimensional manifold $\mathcal{M}$ and an embedding into $\mathbb{R}^N$ are known. This comes at the possible cost of geometric inexactness, and a higher dimension of the co-state and sensitivity equations
- the co-state in the intrinsic formulation has a generally lower dimension, which reduces the dimension of the sensitivity equations. The chart-based formulation also guarantees geometrically exact integration of dynamics. This comes at the mild cost of having to define local charts and chart-transitions.

This dimensionality reduction is unlikely to have a high impact when the manifold $\mathcal{M}$ is known and low dimensional, e.g., for the sphere $\mathcal{M} = S^2$ or similar manifolds. However, when applying the manifold hypothesis to high-dimensional data, there might be non-trivial latent manifolds for which the embedding is not immediate, and where the latent manifold is of a much lower dimension than the embedding data-manifold. Then the intrinsic method becomes difficult to avoid. If geometric exactness of the integration is desired, local charts need to be defined also for the extrinsic approach, in which case the intrinsic approach may offer further advantages.

In order to derive neural ODEs on Lie groups, three approaches were possible: the extrinsic and intrinsic formulations on manifolds directly carry over to matrix Lie groups, embedding $G \subset GL(m, \mathbb{R})$ in $\mathbb{R}^{m^2}$ or using local exponential charts, respectively. A third option a novel intrinsic method for neural ODEs on matrix Lie groups, which made full use of the Lie group structure by phrasing dynamics on $\mathfrak{g}$ (as is more common on Lie groups), and the co-state on $\mathfrak{g}^*$, avoiding difficulties of the chart-based formalism in differentiating extra terms.

Summarizing prior comments on advantages and disadvantages of these flavors:

- the extrinsic formulation on matrix Lie groups can come at much higher cost than that on manifolds, since the intrinsic dimension of G can be much lower than m^2, and a higher dimension of the co-state and sensitivity equations. Geometrically exact integration procedures are more readily available for matrix Lie groups, integrating $\dot{g}$ in local exponential charts.
- the chart-based formulation on matrix Lie groups struggles when are not naturally phrased in local charts. This is common, dynamics are often more naturally phrased on $\mathfrak{g}$. This was alleviated by an algebra-based formulation on matrix Lie groups. Both are intrinsic approaches, that feature

 doi:10.20944/preprints202506.1038.v1

a co-state dynamics that are as low as posssible. However, the algebra-based approach still misses readily available software implementation.

The authors believe that the algebra-based formulation is more convenient, in principle, and consider software implementations of the algebra-based approach as possible future work.

In summary, we presented a unified, geometric approach to extend various methods for neural ODEs on $\mathbb{R}^N$ to neural ODEs on manifolds and Lie groups. Optimization of neural ODEs on manifolds was based on the adjoint method on manifolds. Given a novel cost-function C and neural ODE architecture f, the strategy to present the results in a unified fashion was to identify a suitable augmented manifold $\mathcal{M}_{\text{aug}}$, augmented dynamics $f_{\text{aug}} \in \mathfrak{X}(\mathcal{M}_{\text{aug}})$, and cost $C_{\text{aug}} : \mathcal{M}_{\text{aug}} \to \mathbb{R}$ such that the original cost-function can be rephrased as $C = C_{\text{aug}} \circ \Psi^T_{f_{\text{aug}}}$. To further derive optimization of intrinsic neural ODEs on Lie groups was based on finding a Hamiltonian formulation of the adjoint method on manifolds, and to subsequently transformed them into the Hamiltonian equations on a matrix Lie group.

Author Contributions: Conceptualization, Y.P.W.; methodology, Y.P.W.; software, Y.P.W.; validation, Y.P.W.; formal analysis, Y.P.W.; investigation, Y.P.W.; resources, S.S.; data curation, Y.P.W.; writing—original draft preparation, Y.P.W.; writing—review and editing, Y.P.W., F.C., S.S.; visualization, Y.P.W.; supervision, F.C., S.S.; project administration, S.S.; funding acquisition, S.S. All authors have read and agreed to the published version of the manuscript.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflicts of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; or in the decision to publish the results.

Appendix A. Additional Material

Appendix A.1. Hamiltonian dynamics on Lie groups

We briefly review Hamiltonian systems on manifolds and matrix Lie groups (see also [21, App. A1]).

Given a manifold $\mathcal{Q}$ with coordinate maps $Q^i : \mathcal{Q} \to \mathbb{R}$ and p_i in the basis $\mathrm{d}q^i$ on $T^*_q\mathcal{Q}$, we define the symplectic form $\omega \in \Omega^2(T^*\mathcal{M})$ as

$$\omega = \mathrm{d}p_i \wedge \mathrm{d}q^i \,. \tag{A1}$$

Let $Y \in \mathfrak{X}(T^*\mathcal{Q})$, then a Hamiltonian $H \in C^\infty(T^*\mathcal{Q}, \mathbb{R})$ implicitly defines a unique vector field $X_H \in \mathfrak{X}(T^*\mathcal{Q})$ by

$$\mathrm{d}H(Y) = \omega(X_H, Y) \,. \tag{A2}$$

In coordinates, X_H has the components

$$\dot{q}^i = \frac{\partial H}{\partial p_i} \,, \tag{A3}$$

$$\dot{p}_i = -\frac{\partial H}{\partial q^i} \,. \tag{A4}$$

On a Lie group G, the group structure allows the identification $T^*G \equiv G \times \mathfrak{g}^* \equiv G \times \mathbb{R}^n$. E.g., using the pull back ${L_g}^* : T^*_g G \to \mathfrak{g}^*$ of left-translation map $L_g : G \to G$, and $\Lambda^* : \mathfrak{g} \to \mathbb{R}^n$, to define $P_g \in \mathbb{R}^n$ as

$$P_g = \Lambda^* L_g^* P \,. \tag{A5}$$

Then the left Hamiltonian $\mathcal{H}^L : G \times \mathfrak{g}^* \to \mathbb{R}$ is defined in terms of $H : T^*G \to \mathfrak{g}$ as

$$\mathcal{H}^L(g, P_g) = H\big(g, P)\big) \,. \tag{A6}$$

 doi:10.20944/preprints202506.1038.v1

For a matrix Lie group the left Hamiltonian equations read:

$$\dot{g} = g\Lambda(\frac{\partial \mathcal{H}^L}{\partial P}), \tag{A7}$$

$$\dot{P} = -\mathrm{d}_g^L \mathcal{H}^L + \mathrm{ad}^\top_{\frac{\partial \mathcal{H}^L}{\partial P}} P, \tag{A8}$$

with $\Lambda : \mathbb{R}^n \to \mathfrak{g}$ as in (12) and $\mathrm{d}_g^L \mathcal{H} \in \mathbb{R}^n$ as in (22).

References

1. Chen, R.T.Q.; Rubanova, Y.; Bettencourt, J.; Duvenaud, D. Neural Ordinary Differential Equations. *CoRR* **2018**, *abs/1806.07366*, [1806.07366].
2. Massaroli, S.; Poli, M.; Park, J.; Yamashita, A.; Asama, H. Dissecting neural ODEs. *Advances in Neural Information Processing Systems* **2020**, *2020-Decem*, [2002.08071].
3. Zakwan, M.; Natale, L.D.; Svetozarevic, B.; Heer, P.; Jones, C.; Trecate, G.F. Physically Consistent Neural ODEs for Learning Multi-Physics Systems*. *IFAC-PapersOnLine* **2023**, *56*, 5855–5860. 22nd IFAC World Congress, https://doi.org/https://doi.org/10.1016/j.ifacol.2023.10.079.
4. Sholokhov, A.; Liu, Y.; Mansour, H.; Nabi, S. Physics-informed neural ODE (PINODE): embedding physics into models using collocation points. *Scientific Reports 2023 13:1* **2023**, *13*, 1–13. https://doi.org/10.1038/s41598-023-36799-6.
5. Ghanem, P.; Demirkaya, A.; Imbiriba, T.; Ramezani, A.; Danziger, Z.; Erdogmus, D. Learning Physics Informed Neural ODEs With Partial Measurements. *AAAI-25* **2024**, [arXiv:cs.LG/2412.08681].
6. Massaroli, S.; Poli, M.; Califano, F.; Park, J.; Yamashita, A.; Asama, H. Optimal Energy Shaping via Neural Approximators. *SIAM Journal on Applied Dynamical Systems* **2022**, *21*, 2126–2147, [https://doi.org/10.1137/21M1414279]. https://doi.org/10.1137/21M1414279.
7. Niu, H.; Zhou, Y.; Yan, X.; Wu, J.; Shen, Y.; Yi, Z.; Hu, J. On the applications of neural ordinary differential equations in medical image analysis. *Artificial Intelligence Review* **2024**, *57*, 1–32. https://doi.org/10.1007/S10462-024-10894-0/TABLES/5.
8. Oh, Y.; Kam, S.; Lee, J.; Lim, D.Y.; Kim, S.; Bui, A.A.T. Comprehensive Review of Neural Differential Equations for Time Series Analysis **2025**.
9. Poli, M.; Massaroli, S.; Yamashita, S.A.J.O.N.L.S.C.N.A.L.A.; Asama, H.; Garg, A. Neural Hybrid Automata: Learning Dynamics with Multiple Modes and Stochastic Transitions **2021**.
10. Chen, R.T.Q.; Amos, B.; Nickel, M. Learning Neural Event Functions for Ordinary Differential Equations **2021**.
11. Davis, J.Q.; Choromanski, K.; Varley, J.; Lee, H.; Slotine, J.J.; Likhosterov, V.; Weller, A.; Makadia, A.; Sindhwani, V. Time Dependence in Non-Autonomous Neural ODEs **2020**. [arXiv:cs.LG/2005.01906].
12. Dupont, E.; Doucet, A.; Teh, Y.W. Augmented Neural ODEs **2019**. [arXiv:stat.ML/1904.01681].
13. Chu, H.; Miyatake, Y.; Cui, W.; Wei, S.; Furihata, D. Structure-Preserving Physics-Informed Neural Networks With Energy or Lyapunov Structure, 2024, [arXiv:cs.LG/2401.04986].
14. Kütük, M.; Yücel, H. Energy dissipation preserving physics informed neural network for Allen–Cahn equations. *Journal of Computational Science* **2025**, *87*, 102577. https://doi.org/https://doi.org/10.1016/j.jocs.2025.102577.
15. Bullo, F.; Murray, R.M. Tracking for fully actuated mechanical systems: a geometric framework. *Automatica* **1999**, *35*, 17–34. https://doi.org/10.1016/S0005-1098(98)00119-8.
16. Marsden, J.E.; Ratiu, T.S. *Introduction to Mechanics and Symmetry*; Vol. 17, Springer New York, 1999. https://doi.org/10.1007/978-0-387-21792-5.
17. Whiteley, N.; Gray, A.; Rubin-Delanchy, P. Statistical exploration of the Manifold Hypothesis **2025**. [arXiv:stat.ME/2208.11665].
18. Lou, A.; Lim, D.; Katsman, I.; Huang, L.; Jiang, Q.; Lim, S.N.; De Sa, C. Neural Manifold Ordinary Differential Equations. *Advances in Neural Information Processing Systems* **2020**, *2020-Decem*, [2006.10254].
19. Falorsi, L.; Davidson, T.R.; Berkeley, A.U.C.; Forré, P.; Mar, M.L. Reparameterizing Distributions on Lie Groups **2019**. *89*, [arXiv:1903.02958v1].
20. Duong, T.; Altawaitan, A.; Stanley, J.; Atanasov, N. Port-Hamiltonian Neural ODE Networks on Lie Groups for Robot Dynamics Learning and Control. *IEEE Transactions on Robotics* **2024**, *40*, 3695–3715. https://doi.org/10.1109/TRO.2024.3428433.

 doi:10.20944/preprints202506.1038.v1

21. Wotte, Y.P.; Califano, F.; Stramigioli, S. Optimal potential shaping on SE(3) via neural ordinary differential equations on Lie groups. *The International Journal of Robotics Research* **2024**, *43*, 2221–2244. https://doi.org/10.1177/02783649241256044.
22. Falorsi, L.; Forré, P. Neural Ordinary Differential Equations on Manifolds, 2020, [2006.06663].
23. Andersdotter, E.; Persson, D.; Ohlsson, F. Equivariant Manifold Neural ODEs and Differential Invariants **2024**.
24. Wotte, Y. Optimal Potential Energy Shaping on SE(3) via Neural Approximators. *University of Twente Archive* **2021**.
25. Pau, B.S. An introduction to neural ordinary differential equations **2024**.
26. Gholami, A.; Keutzer, K.; Biros, G. ANODE: Unconditionally Accurate Memory-Efficient Gradients for Neural ODEs **2019**. [arXiv:cs.LG/1902.10298].
27. Kidger, P.; Morrill, J.; Foster, J.; Lyons, T.J. Neural Controlled Differential Equations for Irregular Time Series. *CoRR* **2020**, *abs/2005.08926*, [2005.08926].
28. Li, X.; Wong, T.K.L.; Chen, R.T.Q.; Duvenaud, D. Scalable Gradients for Stochastic Differential Equations **2020**. [arXiv:cs.LG/2001.01328].
29. Floryan, D.; Graham, M.D. Data-driven discovery of intrinsic dynamics. *Nature Machine Intelligence* **2022**, *4*, 1113–1120. https://doi.org/10.1038/s42256-022-00575-4.
30. Duong, T.; Atanasov, N. Hamiltonian-based Neural ODE Networks on the SE(3) Manifold For Dynamics Learning and Control **2021**. [2106.12782v3].
31. Isham, C.J. *Modern Differential Geometry for Physicists*; 1999. https://doi.org/https://doi.org/10.1142/3867.
32. Hall, B.C. *Lie Groups, Lie Algebras, and Representations: An Elementary Introduction*; Graduate Texts in Mathematics (GTM, volume 222), Springer, 2015.
33. Solà, J.; Deray, J.; Atchuthan, D. A micro Lie theory for state estimation in robotics, 2021, [arXiv:cs.RO/1812.01537].
34. Visser, M.; Stramigioli, S.; Heemskerk, C. Cayley-Hamilton for roboticists. *IEEE International Conference on Intelligent Robots and Systems* **2006**, *1*, 4187–4192. https://doi.org/10.1109/IROS.2006.281911.
35. Robbins, H.; Monro, S. A Stochastic Approximation Method. *The Annals of Mathematical Statistics* **1951**, *22*, 400 – 407. https://doi.org/10.1214/aoms/1177729586.
36. Poli, M.; Massaroli, S.; Yamashita, A.; Asama, H.; Park, J. TorchDyn: A Neural Differential Equations Library **2020**. [2009.09346].
37. Wotte, Y.P.; Dummer, S.; Botteghi, N.; Brune, C.; Stramigioli, S.; Califano, F. Discovering efficient periodic behaviors in mechanical systems via neural approximators. *Optimal Control Applications and Methods* **2023**, *44*, 3052–3079. https://doi.org/https://doi.org/10.1002/oca.3025.
38. Munthe-Kaas, H. High order Runge-Kutta methods on manifolds. *Applied Numerical Mathematics* **1999**, *29*, 115–127. Proceedings of the NSF/CBMS Regional Conference on Numerical Analysis of Hamiltonian Differential Equations, https://doi.org/https://doi.org/10.1016/S0168-9274(98)00030-0.