Comprehensive Review of Neural Differential Equations for Time Series Analysis

YongKyung Oh[1] , **Seungsu Kam**[2] , **Jonghun Lee**[3] , **Dong-Young Lim**[2,3] ,
Sungil Kim[2,3*] and **Alex A. T. Bui**[1*]

[1]Medical & Imaging Informatics (MII) Group, University of California, Los Angeles (UCLA)
[2]Department of Industrial Engineering, Ulsan National Institute of Science and Technology (UNIST)
[3]Artificial Intelligence Graduate School, Ulsan National Institute of Science and Technology (UNIST)
yongkyungoh@mednet.ucla.edu, {lewki83, jh.lee, dlim, sungil.kim}@unist.ac.kr, buia@mii.ucla.edu

Abstract

Time series modeling and analysis have become critical in various domains. Conventional methods such as RNNs and Transformers, while effective for discrete-time and regularly sampled data, face significant challenges in capturing the continuous dynamics and irregular sampling patterns inherent in real-world scenarios. Neural Differential Equations (NDEs) represent a paradigm shift by combining the flexibility of neural networks with the mathematical rigor of differential equations. This paper presents a comprehensive review of NDE-based methods for time series analysis, including neural ordinary differential equations, neural controlled differential equations, and neural stochastic differential equations. We provide a detailed discussion of their mathematical formulations, numerical methods, and applications, highlighting their ability to model continuous-time dynamics. Furthermore, we address key challenges and future research directions. This survey serves as a foundation for researchers and practitioners seeking to leverage NDEs for advanced time series analysis.

1 Introduction

The exponential growth of time series data across diverse domains has necessitated more sophisticated analytical approaches, because of the complex nonlinear nature, irregular sampling, missing values, and continuous latent dynamics. Traditional approaches [Durbin and Koopman, 2012; Box *et al.*, 2015] assume regular sampling and often linear relationships, limiting their applicability to real-world scenarios. While deep learning methods, including Recurrent Neural Network (RNN) [Rumelhart *et al.*, 1986; Medsker and Jain, 1999], Long Short-term Memory (LSTM) [Hochreiter and Schmidhuber, 1997], Gated Recurrent Unit (GRU) [Chung *et al.*, 2014], and Transformer [Vaswani *et al.*, 2017] have shown promise in capturing nonlinearity, they remain constrained by discrete-time formulations [Che *et al.*, 2018; Sun *et al.*, 2020; Weerakody *et al.*, 2021]

---
*Corresponding authors

Neural Differential Equations (NDEs) emerged as a paradigm shift, offering a principled framework for continuous-time modeling with neural networks. The seminal work on Neural Ordinary Differential Equations (NODEs) [Chen *et al.*, 2018] introduced continuous-time hidden state evolution, spawning numerous advances including Neural Controlled Differential Equations (NCDEs) [Kidger *et al.*, 2020] incorporating external control paths, and Neural Stochastic Differential Equations (NSDEs) [Han *et al.*, 2017; Tzen and Raginsky, 2019; Li *et al.*, 2020; Oh *et al.*, 2024] modeling uncertainty through stochasticity.

In this survey, we provide a comprehensive review of NDE methods in time series analysis, synthesizing advancements across several interconnected areas. We begin by exploring the base model families, including NODEs, NCDEs, and NSDEs, along with their extensions and variations. Building on this, we analyze the theoretical insights that underpin these models. We also discuss the implementation of NDE methods and analyze applications. This survey aims to provide a structured synthesis for researchers and practitioners, fostering a deeper understanding of NDEs and their potential in tackling complex time series problems.

2 Preliminaries

Time series modeling seeks to represent sequential data $x = (x_0, x_1, \ldots, x_n)$, where each $x_i \in \mathbb{R}^{d_x}$, as a continuous latent process $z(t) \in \mathbb{R}^{d_z}$ over a time domain $[0, T]$. Figure 1 illustrates the conceptual differences between conventional time series modeling and NDE-based methods. While interpolation methods simply fit curves through observations (black dots) and RNNs (often using zero-order hold) operate in discrete time steps, NDEs aim to learn the underlying continuous dynamics, $dx(t)/dt$ (or $dz(t)/dt$), parameterized by a neural network [Chen *et al.*, 2018]. This parameterization provides NDEs with a flexible framework for handling challenges common in real-world time series, such as irregular sampling, missing observations, and capturing long-horizon dependencies. Furthermore, NDEs can offer greater memory efficiency compared to standard discrete-time architectures [Chen *et al.*, 2018; Kidger *et al.*, 2020; Rubanova *et al.*, 2019; Oh *et al.*, 2024].

RNNs, including LSTMs [Hochreiter and Schmidhuber, 1997] and GRUs [Chung *et al.*, 2014], operate in discrete time, updating hidden states sequentially based on observed

(a) State of piecewise interpolations (b) State of traditional RNNs (c) State of NDE-based methods

(d) Gradient of piecewise interpolations (e) Gradient of traditional RNNs (f) Gradient of NDE-based methods
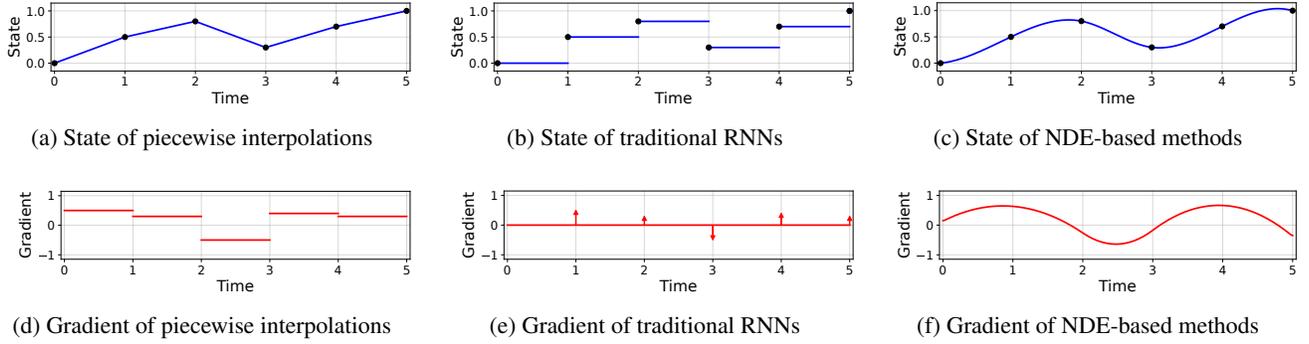
Figure 1: Conceptual comparison of time series modeling. (a) Piecewise interpolation provides continuity but lacks dynamic modeling capabilities. (b) Traditional RNNs operate on discrete steps, encountering limitations with continuous dynamics, irregularly sampled data, and memory efficiency for long sequences. (c) NDE-based methods, such as NODEs, NCDEs, and NSDEs, learn continuous dynamics, inherently accommodate irregular sampling, and offer potential memory advantages, when employing techniques like the adjoint method.

inputs. Transformer-based models [Vaswani *et al.*, 2017] offer an alternative by leveraging self-attention mechanisms, though they typically rely on fixed positional encodings and do not inherently capture continuous-time dynamics.

Various modifications have been proposed to introduce a notion of time awareness in RNNs [Che *et al.*, 2018; Rajkomar *et al.*, 2018]. One approach adjusts updates for time intervals, while another uses decay mechanisms to approximate continuous evolution. However, these heuristics do not always generalize across different settings and, in some cases, have been found to perform comparably to standard RNNs [Mozer *et al.*, 2017; Rubanova *et al.*, 2019].

2.1 Neural Ordinary Differential Equations

NODEs [Chen *et al.*, 2018] model the latent state $z(t)$ as:

$$z(t) = z(0) + \int_0^t f\big(s, z(s); \theta_f\big)\,\mathrm{d}s, \tag{1}$$

where $z(0) = h(x; \theta_h)$, $h : \mathbb{R}^{d_x} \to \mathbb{R}^{d_z}$ is a neural network parameterized by θ_h, and $f\big(t, z(t); \theta_f\big)$ approximates $\frac{\mathrm{d}z(t)}{\mathrm{d}t}$. The vector field f is typically implemented using multi-layer perceptrons or more sophisticated architectures. One can view these approaches as an infinite-layer generalization of residual networks, allowing integration over time rather than discrete stacking of layers [Chen *et al.*, 2018; Rubanova *et al.*, 2019; Dupont *et al.*, 2019; Kidger *et al.*, 2020].

2.2 Neural Controlled Differential Equations

NCDEs [Kidger *et al.*, 2020] extend NODEs by incorporating a control path $X(t)$ for updating state over time:

$$z(t) = z(0) + \int_0^t f\big(s, z(s); \theta_f\big)\,\mathrm{d}X(s), \tag{2}$$

where the integral is interpreted as a Riemann–Stieltjes integral, allowing for discontinuous paths. Piecewise-smooth control paths $X(t)$ are typically constructed using natural cubic splines [Kidger *et al.*, 2020] or Hermite cubic splines [Morrill *et al.*, 2021a]. Furthermore, Morrill *et al.* (2021b) extend Neural CDEs by incorporating rough path theory for the generalized formulation.

2.3 Neural Stochastic Differential Equations

NSDEs [Han *et al.*, 2017; Tzen and Raginsky, 2019; Li *et al.*, 2020] incorporate stochasticity through:

$$z(t) = z(0) + \int_0^t f\big(s, z(s); \theta_f\big)\,\mathrm{d}s$$
$$+ \int_0^t g\big(s, z(s); \theta_g\big)\,\mathrm{d}W(s), \tag{3}$$

where $W(t)$ is a Wiener process (or Brownian motion), $f\big(\cdot; \theta_f\big)$ is the drift function, and $g\big(\cdot; \theta_g\big)$ is the diffusion function. The stochastic integral follows Itô or Stratonovich interpretations. While NODEs describe deterministic evolution through ordinary differential equations, NSDEs model uncertainty and noise through Brownian motion terms [Jia and Benson, 2019; Liu *et al.*, 2019]. This extension enables more robust modeling of real-world phenomena where randomness plays a crucial role [Kidger *et al.*, 2021b; Kidger *et al.*, 2021c; Oh *et al.*, 2024], each offering different approaches to balance stability, expressivity, and computational tractability.

3 Theoretical Considerations

3.1 Dynamical Systems Perspective

NDEs generalize the notion of discrete layers (as in RNNs or Transformers) to a continuous-time vector field. According to Eq. (1), the latent representation $z(t)$ evolves:

$$\frac{\mathrm{d}z(t)}{\mathrm{d}t} = f\big(t, z(t); \theta_f\big), \tag{4}$$

while in NCDEs, the evolution is driven by a control path:

$$\mathrm{d}z(t) = f\big(t, z(t); \theta_f\big)\,\mathrm{d}X(t). \tag{5}$$

Here, $X(t)$ is typically a spline or piecewise interpolation of observed input data x, as explained in Eq. (2).

Stochastic extensions, such as NSDEs, incorporate Brownian motion $W(t)$ or jump processes to model noise or uncertainty in the underlying dynamics, as shown in Eq. (3):

$$\mathrm{d}z(t) = f\big(t, z(t); \theta_f\big)\,\mathrm{d}t + g\big(t, z(t); \theta_g\big)\,\mathrm{d}W(t), \tag{6}$$

where $W(t)$ is a Wiener process. Formally, these methods place the latent trajectory $z(t)$ in a phase space:

$$z : [0, T] \to \mathbb{R}^{d_z}, \quad z(0) = h(x; \theta_h), \qquad (7)$$

where $h : \mathbb{R}^{d_x} \to \mathbb{R}^{d_z}$ maps raw inputs x to the initial state. By solving the equations above, one obtains a continuous phase-space trajectory that can be analyzed through classical dynamical systems tools. This continuous viewpoint also enables seamless continuous-time modeling.

Invertible Neural Networks and Flow Models. In parallel with NDEs, explicit approaches like *Neural Flows* have also been proposed [Lu *et al.*, 2018; Sonoda and Murata, 2019; Grathwohl *et al.*, 2019; Massaroli *et al.*, 2020a; Biloš *et al.*, 2021]. Rather than parameterizing the rate of change $\mathrm{d}z(t)/\mathrm{d}t$, Neural Flows directly model the solution map:

$$z(t) = \mathcal{F}(t, z(0); \theta_{\mathcal{F}}), \qquad (8)$$

where $z(0) = h(x; \theta_h)$ is the initial condition. This explicit representation bypasses the need for numerical ODE solvers, potentially offering faster computations or enhanced stability via invertibility constraints [Kobyzev *et al.*, 2020; Papamakarios *et al.*, 2021]. Such architectures align with normalizing-flow models, which are invertible by construction and allow for efficient likelihood evaluation.

Contrasting Implicit and Explicit Methods. The distinction between NODEs/NCDEs (which define $\mathrm{d}z/\mathrm{d}t$ and rely on solvers) and Neural Flows (which specify $z(t)$ directly) parallels the classic implicit versus explicit solution dichotomy in differential equations. Implicit, solver-based methods (like NODEs, NCDEs, NSDEs) can handle irregular inputs through fixed or adaptive step sizes and are more general in capturing unknown or complex dynamics. Explicit approaches, like Neural Flows [Biloš *et al.*, 2021], may yield simpler training pipelines for stable transformations, yet can suffer from constraints on network forms and from instabilities if the sought-after closed-form solution is nonexistent or approximate [Oh *et al.*, 2025].

In this survey, we limit our scope to NDEs that require implicit solvers, excluding flow-based models and other explicit parameterizations of continuous-time dynamics.

3.2 Universal Approximation Properties

NODEs. Chen *et al.* (2018) established that continuous-time models can be viewed as an infinite-depth limit of residual networks, enabling them to approximate diffeomorphic transformations under mild regularity conditions, as represented in Eq. (4). However, autonomous ODE flows may limit expressivity by constraining trajectories to remain diffeomorphic transformations of initial states, thus prohibiting intersecting paths in latent space. Dupont *et al.* (2019) addressed these limitations through augmented ODEs, which append an auxiliary variable $a(t)$ to the state:

$$\frac{\mathrm{d}}{\mathrm{d}t} \begin{pmatrix} z(t) \\ a(t) \end{pmatrix} = f\big(t, z(t), a(t); \theta_f\big). \qquad (9)$$

Augmenting the latent dimension effectively bypasses strict diffeomorphic constraints, expanding the range of representable data manifolds. Empirically, this technique improves reconstruction and expressivity in tasks where standard Neural ODEs struggle due to their trajectory overlap constraint.

NCDEs. While NODEs rely solely on a learned vector field and initial value, Kidger *et al.* (2020) showed that Neural CDEs embed a control path $X(t)$ into the model as explained in Eq. (5): By allowing data updates to arrive as increments $\mathrm{d}X(t)$ at arbitrary times, NCDEs implement a form of continuous recursion, demonstrating universal approximation for continuous paths under sufficiently rich control signals. Extensions to rough paths [Morrill *et al.*, 2021b] further expand coverage to long, irregular time series, handling signals too irregular for traditional ODE integrators.

NSDEs. Stochastic extensions achieve universal approximation for broad classes of continuous-time random processes by learning drift and diffusion terms. Han *et al.* (2017) and Tzen and Raginsky (2019) formalized how parameterizing f and g via neural networks lets these models approximate various stochastic phenomena, such as Geometric Brownian motion or Ornstein–Uhlenbeck processes. Subsequent works generalize to jump processes and specialized SDE families [Jia and Benson, 2019; Li *et al.*, 2020; Oh *et al.*, 2024; Zhang *et al.*, 2024], underscoring the versatility of NSDEs in capturing noise-driven dynamics.

3.3 Existence and Uniqueness of Solutions

Deterministic NDEs. For NODEs as explained in Eq. (4), classic results such as the Picard–Lindelöf theorem guarantees that a unique solution $z(t)$ exists if the vector field f is Lipschitz continuous in z. Formally, if

$$\|f\big(t, z(t_1); \theta_f\big) - f\big(t, z(t_2); \theta_f\big)\| \leq L\, \|z(t_1) - z(t_2)\| \tag{10}$$

for some constant $L > 0$, then the integral equation describing $z(t)$ has a unique solution starting from $z(0) = h(x)$ [Chen *et al.*, 2018]. In practice, spectral normalization or weight clipping can enforce such Lipschitz constraints, improving both existence properties and training stability [Massaroli *et al.*, 2020b; Rackauckas *et al.*, 2020].

In NCDEs as shown in Eq. (5), existence and uniqueness in this setting hinge on Lipschitz-like conditions for f and on the path $X(t)$ possessing sufficient regularity (e.g., bounded variation or spline-based interpolation) [Kidger *et al.*, 2020]. If $X(t)$ is discontinuous or highly irregular, rough path theory provides a generalized framework under which well-posedness can still be established [Morrill *et al.*, 2021a].

Stochastic NDEs. NSDEs embed stochasticity via Eq. (6), existence and uniqueness then require both f and g to satisfy Lipschitz and linear-growth conditions in z [Tzen and Raginsky, 2019; Oh *et al.*, 2024]. Common conditions for the existence and uniqueness of z are Lipschitz continuity and linear growth conditions for f and g such that if there exist constants L and C such that

$$\begin{aligned} &\|f\big(z(t_1); \theta_f\big) - f\big(z(t_2); \theta_f\big)\| \\ &\quad + \|g\big(z(t_1); \theta_g\big) - g\big(z(t_2); \theta_g\big)\| \leq L\|z(t_1) - z(t_2)\| \end{aligned} \tag{11}$$

and

$$\|f\big(z; \theta_f\big)\| + \|g\big(z; \theta_g\big)\| \leq C\,(1 + \|z\|), \qquad (12)$$

then a unique strong solution $z(t)$ exists for all finite t. Under these conditions, finite-time blow-up is prevented,

ensuring the SDE is well-posed [Jia and Benson, 2019; Li *et al.*, 2020]. Langevin-type SDEs, linear-noise SDEs, and geometric SDEs—variants of Neural SDEs—are also designed to satisfy these conditions, thereby ensuring the existence and uniqueness of their solutions [Oh *et al.*, 2024].

3.4 Stability and Convergence

Deterministic Stability. NODEs and NCDEs achieve stability through Lipschitz constraints on vector fields [Haber and Ruthotto, 2017; Chen *et al.*, 2018; Kidger *et al.*, 2020]. For NODEs, spectral norm conditions on network weights keep trajectories bounded [Kidger *et al.*, 2021a]. NCDEs maintain stability by bounding $\mathrm{d}X/\mathrm{d}t$ [Morrill *et al.*, 2021a].

Stochastic Stability. NSDEs follow dynamics $\mathrm{d}\boldsymbol{z}(t) = f(\boldsymbol{z}, t; \theta_f)\,\mathrm{d}t + g(\boldsymbol{z}, t; \theta_g)\,\mathrm{d}W(t)$, where stability depends on drift f and diffusion g terms [Tzen and Raginsky, 2019; Kidger *et al.*, 2021b]. Oh *et al.* (2024) shows the stochastic stability of the proposed Neural SDEs (Langevin-type SDEs, linear-noise SDEs, and geometric SDEs) under suitable regularity conditions and relates the findings to their robustness against distribution shifts.

Optimal Transport and Convergence. The convergence of continuous-time methods can be analyzed through optimal transport theory, with the Wasserstein metric offering a principled measure of stability and generalization [Villani and others, 2009; Peyré *et al.*, 2019]. Formally, $(\Omega, \mathcal{F}, P)$ specifies a probability space for random variables in $L^p(\Omega)$ whenever stochastic components arise. For an $\mathbb{R}^d$-valued random variable X, its law $\mathcal{L}(X)$ belongs to $\mathcal{P}(\mathbb{R}^d)$. The Wasserstein distance of order p measures how far two distributions $\mu, \nu \in \mathcal{P}(\mathbb{R}^d)$ are in terms of the minimum cost coupling, such as:

$$\mathcal{W}_p(\mu, \nu) = \inf_{\Pi \in \mathcal{C}(\mu, \nu)} \left(\int_{\mathbb{R}^d}\!\!\int_{\mathbb{R}^d} |x - x'|^p\, \Pi(\mathrm{d}x, \mathrm{d}x') \right)^{1/p} \tag{13}$$

Here, $\mathcal{C}(\mu, \nu)$ denotes the set of couplings whose marginals match μ and ν. In the context of NDE models, particularly in stochastic or high-dimensional regimes, this metric provides a valuable tool for analyzing model robustness, generalization, and convergence properties [Kidger, 2021; Ruiz-Balet and Zuazua, 2023; Oh *et al.*, 2024].

Solver Convergence. NDEs rely on numerical integrators such as Euler, Runge–Kutta, and Dormand–Prince for deterministic models [Chen *et al.*, 2018; Rubanova *et al.*, 2019] and Euler–Maruyama, Milstein, or Reversible Heun for stochastic models [Kidger *et al.*, 2021c; Tzen and Raginsky, 2019]. On the other hand, systems that are stiff, exhibit jumps, or involve high-dimensional dynamics often necessitate specialized implicit solvers to ensure stability and computational efficiency [Jia and Benson, 2019; Rackauckas *et al.*, 2020; Kim *et al.*, 2021].

4 Practical Implementation

This section expands on the theoretical foundations from Section 3, focusing on the practical aspects of training, regularization, and deployment of NDEs. While discrete-time mod-

els are widely studied, continuous-time models remain underexplored despite their advantages.

4.1 Optimization of NDE-based Models

Adjoint Method. NDEs differ significantly from discrete methods in their reliance on numerical integration. During training, gradients with respect to model parameters must be backpropagated through an ODE, CDE, or SDE solver, which raises memory and stability challenges. Traditional backpropagation is conceptually straightforward but stores all intermediate states, leading to large memory usage in long sequences or high-dimensional latents.

The adjoint sensitivity method, introduced by Chen *et al.* (2018), is a pivotal technique for efficiently computing gradients in NDEs by solving an adjoint equation backward in time. This method addresses the challenge of high memory consumption in gradient-based optimization by reconstructing forward states on demand, reducing the memory complexity from $O(N)$, where N is the sequence length, to approximately $O(1)$. The adjoint state $\lambda(t)$ evolves according to a differential equation,

$$\frac{\mathrm{d}\lambda(t)}{\mathrm{d}t} = -\lambda(t)^\top \partial f / \partial \boldsymbol{z} \tag{14}$$

where $\lambda(T) = \partial \mathcal{L}/\partial \boldsymbol{z}(T)$ initializes the backward integration. This approach enables the computation of gradients via

$$\mathrm{d}\mathcal{L}/\mathrm{d}\theta = \int_0^T \lambda(t)^\top \partial f / \partial \theta\, \mathrm{d}t, \tag{15}$$

making it suitable for training models with terminal loss functions. However, the adjoint sensitivity method encounters numerical challenges, particularly in stiff or chaotic systems where reverse-time integration can amplify floating-point errors. Alternatively, checkpointing techniques [Gholami *et al.*, 2019; Zhuang *et al.*, 2020], store selected intermediate states during the forward pass, allowing localized recomputation during backward propagation.

Integral Loss Functions. Beyond terminal loss functions, NDEs have been extended to incorporate integral loss functions distributed across the entire depth domain S. In the context of optimal control [Pontryagin, 2018], the integral loss is defined as formulated by Massaroli *et al.* (2020a):

$$\mathcal{L}(\boldsymbol{z}(S)) + \int_{\mathcal{S}} \ell(t, \boldsymbol{z}(t))\, \mathrm{d}t, \tag{16}$$

where the loss combines terminal contributions $\mathcal{L}(\boldsymbol{z}(S))$ with intermediate terms $\ell(t, \boldsymbol{z}(t))$. This formulation allows the latent state $\boldsymbol{z}(t)$ to evolve through a continuum of layers, guiding the model output toward the desired trajectory over the entire depth domain S. The adjoint dynamics for such integral loss functions are modified to include an additional term, $-\partial \ell / \partial \boldsymbol{z}(t)$, accounting for the distributed loss in the backward gradients [Grathwohl *et al.*, 2018; Massaroli *et al.*, 2020a; Finlay *et al.*, 2020b]. This approach enhances the control and flexibility of trajectory-level learning, enabling improved performance in tasks that require supervision over the entire temporal or spatial domain.

Backpropagation through time. For discrete methods like RNNs, backpropagation is a common optimization strategy, where the model's recurrence is unfolded into a sequence of differentiable operations. However, these discrete methods face vanishing or exploding gradients over long sequences [Hochreiter and Schmidhuber, 1997; Pascanu *et al.*, 2013; Goodfellow *et al.*, 2016]. NDE-based variants, such as GRU-ODE [De Brouwer *et al.*, 2019], ODE-RNN [Rubanova *et al.*, 2019], ODE-LSTM [Lechner and Hasani, 2020], mitigate these issues by replacing discrete recurrence with continuous evolution, enabling robust modeling of time series while retaining mechanisms like gating for temporal dependencies.

4.2 Regularization Methods

Advanced Techniques for NDEs. While conventional regularization techniques are widely applicable to NDE-based methods, recent advances have introduced specialized approaches tailored to continuous dynamics. Regularization based on principles of optimal transport, proposed by Finlay *et al.*; Finlay *et al.* (2020b; 2020a), simplifies the dynamics of continuous normalizing flow models, thereby accelerating training. Kelly *et al.* (2020) analyzed equations with differential surrogates for the computational cost of standard solvers, leveraging higher-order derivatives of solution trajectories to reduce complexity.

Continuous-Time Modifications. Standard regularization techniques have further refined NDE-based methods; stochastic sampling of end times [Ghosh *et al.*, 2020], continuous-time dropout [Liu *et al.*, 2020], and temporal adaptive batch normalization [Zheng *et al.*, 2024]. These methods form a comprehensive toolkit for advancing the performance and scalability of NDE-based models.

4.3 Numerical Solution of NDEs

Comparison with Discrete Architectures. Discrete-time methods, including variants of RNNs and Transformers, rely on layer-wise or self-attention updates without numeric integration. Although this design simplifies backpropagation and typically scales well, these models often struggle to accommodate irregular sampling unless augmented by masking or gating mechanisms [Chung *et al.*, 2014]. RNN-based architectures can also experience vanishing or exploding gradients for lengthy sequences unless they incorporate techniques such as gradient clipping or gated units [Hochreiter and Schmidhuber, 1997; Pascanu *et al.*, 2013]. Transformers replace recurrence with attention mechanisms but depend heavily on positional encodings and expansive parameter sets, which can limit their practicality for extensive time spans with variable sampling [Vaswani *et al.*, 2017; Rae *et al.*, 2019; Wen *et al.*, 2023]. By contrast, continuous-time NDEs use solvers that naturally adapt to asynchronous events, though solver overhead and potential stiffness require careful solver selection and parameter tuning.

Fixed or Adaptive Solvers. NDE-based models rely on numerical integration to evolve their latent states $z(t)$ from initial conditions to final outputs. A fixed-step solver, such as explicit Euler or a basic Runge–Kutta method, updates $z(t)$ at uniform intervals and is computationally simpler but

may face instability if the dynamics change abruptly or if the time step is too large [Chen *et al.*, 2018]. Adaptive-step solvers, such as Dormand–Prince method, refine the step size Δt in response to local error estimates, offering higher accuracy for stiff or highly non-uniform dynamics [Kidger *et al.*, 2021c]; however, this adaptive nature can cause unpredictable run times in large-scale tasks [Rackauckas *et al.*, 2020]. In the stochastic setting of NSDEs, solvers like Euler–Maruyama or Milstein handle the Brownian increments for the diffusion term, balancing efficiency and stability under well-defined noise [Tzen and Raginsky, 2019; Oh *et al.*, 2024].

5 Comparison of NDE-based Methods

Table 1 summarizes the core formulations, primary tasks, and benchmark datasets associated with various NDE-based methods. These methods can be grouped into three main categories—NODE, NCDE, and NSDE—highlighting their unique characteristics and application domains.

5.1 NODE Methods

Standard form of Neural ODE [Chen *et al.*, 2018] introduced a novel approach to continuous-time modeling by parameterizing the hidden state dynamics with a neural network, allowing for the seamless handling of irregularly sampled time series data. Augmented ODE [Dupont *et al.*, 2019] extend this framework by augmenting the state space with auxiliary variables, enabling the modeling of complex trajectories and avoiding trajectory overlap. ODE^{2}VAE [Yildiz *et al.*, 2019] combines NODEs using Bayesian Neural Network (BNN) with variational autoencoders to model latent dynamics in irregular time series by learning both the continuous-time evolution and the underlying latent space.

On the other hand, GRU-ODE [De Brouwer *et al.*, 2019] adopts continuous gating mechanisms inspired by GRU cells, where the hidden state evolves dynamically based on time gaps. $u(\cdot)$, and $g(\cdot)$ are continuous counterparts of GRU. ODE-RNN [Rubanova *et al.*, 2019] combines ODE-based continuous evolution with discrete updates at observation points using an RNN cell. Similar to that, ODE-LSTM [Lechner and Hasani, 2020] incorporates the memory mechanisms of LSTM into the ODE framework, evolving the hidden state between observations using ODE solvers and performing discrete LSTM updates at observation points.

5.2 NCDE Methods

Neural CDE [Kidger *et al.*, 2020] models the latent state $z(t)$ with a piecewise-smooth control path $X(t)$. The Riemann–Stieltjes integral allows Neural CDEs to handle irregular or asynchronous time series data effectively, making them suitable for tasks like interpolation and forecasting. Neural Rough Differential Equations (Neural RDEs) [Morrill *et al.*, 2021a] extend Neural CDEs by using the log-signature of the control path $\text{LogSig}_{[r_i, r_{i+1}]}(X)$ with $X : [t_0 : t_n]$ and certain intervals $t_0 \leq r_i < r_{i+1} \leq t_n$, capturing higher-order variations. This enables Neural RDEs to handle rough paths, improving robustness for long and complex time series. Recently, Walker *et al.* (2024) further extended rough path the-

NODE Methods	State Evolution / Formulation	Applications / Tasks	Benchmarks / Datasets
Neural ODE [Chen *et al.*, 2018]	$\mathrm{d}\boldsymbol{z}(t) = f\big(t, \boldsymbol{z}(t); \theta_f\big)\,\mathrm{d}t$	Continuous normalizing flow, Image classification	Synthetic data, MNIST
Augmented ODE [Dupont *et al.*, 2019]	$\dfrac{\mathrm{d}}{\mathrm{d}t}\begin{pmatrix}\boldsymbol{z}(t)\\\boldsymbol{a}(t)\end{pmatrix} = f\left(t, \begin{pmatrix}\boldsymbol{z}(t)\\\boldsymbol{a}(t)\end{pmatrix}; \theta_f\right)$	Complex dynamics modeling, Image classification	Synthetic data, MNIST, CIFAR-10, SVHN, ImageNet
GRU-ODE [De Brouwer *et al.*, 2019]	$\dfrac{\mathrm{d}\boldsymbol{z}(t)}{\mathrm{d}t} = \big(1 - \boldsymbol{u}(t)\big) \circ \big(\boldsymbol{g}(t) - \boldsymbol{z}(t)\big)$	Complex dynamics modeling, Time series forecasting	Synthetic data, USHCN, MIMIC-III
ODE^{2}VAE [Yildiz *et al.*, 2019]	$\boldsymbol{z}(t) = (\boldsymbol{s}(t), \boldsymbol{v}(t)),\quad \dot{\boldsymbol{s}}(t) = \boldsymbol{v}(t),\quad \dot{\boldsymbol{v}}(t) = f\big(\boldsymbol{s}(t), \boldsymbol{v}(t)\big),$ $q_{\mathrm{enc}}(\boldsymbol{z}(0)\vert \boldsymbol{x}_{0:N}) = q_{\mathrm{enc}}\left(\begin{pmatrix}\boldsymbol{s}(0)\\\boldsymbol{v}(0)\end{pmatrix}\middle\vert \boldsymbol{x}_{0:N}\right)$ (variational approximation)	High-dimensional sequential data modeling, Motion prediction, Image sequence forecasting	CMU Motion Capture, MNIST, Bouncing Balls
ODE-RNN [Rubanova *et al.*, 2019]	$\bar{\boldsymbol{h}}_t = \mathrm{ODESolve}\big(f, \boldsymbol{h}_{t-1}, [t_{t-1}, t_t]\big)$ $\boldsymbol{h}_t = \mathrm{RNNCell}\big(\bar{\boldsymbol{h}}_t, \boldsymbol{x}_t\big)$	Continuous-time modeling, Interpolation and extrapolation	MuJoCo, PhysioNet mortality, Human Activity
ODE-LSTM [Lechner and Hasani, 2020]	$\bar{\boldsymbol{h}}_t = \mathrm{ODESolve}\big(f, \boldsymbol{h}_{t-1}, [t_{t-1}, t_t]\big)$ $\boldsymbol{h}_t, \boldsymbol{c}_t = \mathrm{LSTMCell}\big(\boldsymbol{c}_{t-1}, \bar{\boldsymbol{h}}_t, \boldsymbol{x}_t\big),\quad$ where internal memory $\boldsymbol{c}_t$	Continuous-time modeling, Time series classification	Synthetic data, Activity recognition, MNIST, MuJoCo physics

NCDE Methods	State Evolution / Formulation	Applications / Tasks	Benchmarks / Datasets
Neural CDE [Kidger *et al.*, 2020]	$\mathrm{d}\boldsymbol{z}(t) = f\big(\boldsymbol{z}(t); \theta_f\big)\,\mathrm{d}X(t),$ using Riemann–Stieltjes integral over control path $X(t)$	Irregular time series analysis, Time series classification	CharacterTrajectories, Speech Commands, PhysioNet Sepsis
Neural RDE [Morrill *et al.*, 2021a]	$\mathrm{d}\boldsymbol{z}(t) = f\big(\boldsymbol{z}(t); \theta_f\big)\,\mathrm{dLogSig}_{[r_i, r_{i+1}]}(X),$ where $\mathrm{LogSig}_{[r_i, r_{i+1}]}(X)$ represents the log-signature of $X(t)$	Long time series analysis, Robust feature extraction	EigenWorms, BIDMC
Neural Lad [Li *et al.*, 2023]	$\dfrac{dz(t)}{dt} = h_w(t)\,\phi_\gamma\big(z(t)\big)\,f\big(z(t)\big)\,g_v\!\left(\dfrac{dX}{dt}\right),$	Time series forecasting, Irregularly sampled time series classification	PhysioNet Sepsis, ETT(1,2), Weather, PEMS(3,4,7,8)
ANCDE [Jhin *et al.*, 2024]	$\mathrm{d}\boldsymbol{h}(t) = f\big(\boldsymbol{h}(t); \theta_f\big)\,\mathrm{d}X(t)\quad$ (attention value, $\boldsymbol{a}(t) = \sigma(\boldsymbol{h}(t))$) $\mathrm{d}\boldsymbol{z}(t) = g\big(\boldsymbol{z}(t); \theta_g\big)\,\mathrm{d}Y(t)\quad$ (attention-modulated control path, $Y(t)$)	Irregular time series classification and forecasting	CharacterTrajectories, PhysioNet Sepsis, Google stock, MuJoCo physics
DualDynamics [Oh *et al.*, 2025]	$\dfrac{\mathrm{d}\boldsymbol{z}(t)}{\mathrm{d}t} = f\big(\boldsymbol{z}(t); \theta_f\big)\,\mathrm{d}X(t) := f^*(t; \theta_f, X)\,\mathrm{d}t$ $\hat{\boldsymbol{z}}(t) = \mathcal{G}\big(t, \hat{\boldsymbol{z}}(0); \theta_{\mathcal{G}}\big)\quad$ where $\dfrac{\mathrm{d}\mathcal{G}\big(t, \hat{\boldsymbol{z}}(0)\big)}{\mathrm{d}t} = f^*(t; \theta_f, X)$	Irregular time series classification, imputation, and forecasting	UEA/UCR archive, PhysioNet mortality, PhysioNet Sepsis, Google stock, MuJoCo physics

NSDE Methods	State Evolution / Formulation	Applications / Tasks	Benchmarks / Datasets
Neural SDE [Tzen and Raginsky, 2019]	$\mathrm{d}\boldsymbol{z}(t) = f\big(t, \boldsymbol{z}(t); \theta_f\big)\,\mathrm{d}t + g\big(t, \boldsymbol{z}(t); \theta_g\big)\,\mathrm{d}W(t)$	Stochastic dynamics modeling with variational inference framework	Theoretical analysis
Neural Jump SDE [Jia and Benson, 2019]	$\mathrm{d}\boldsymbol{z}(t) = f\big(t, \boldsymbol{z}(t); \theta_f\big)\,\mathrm{d}t + w(t, \boldsymbol{z}(t), \boldsymbol{k}(t); \theta_w) \cdot \mathrm{d}N(t)$	Point process modeling, Event feature prediction	Synthetic data, Stack Overflow, MIMIC-II, Earthquake
Latent SDE [Li *et al.*, 2020]	$\mathrm{d}\tilde{\boldsymbol{z}}(t) = h_\theta\big(t, \tilde{\boldsymbol{z}}(t)\big)\,\mathrm{d}t + \sigma\big(t, \tilde{\boldsymbol{z}}(t)\big)\,\mathrm{d}W(t)\quad$ (prior) $\mathrm{d}\boldsymbol{z}(t) = h_\phi\big(t, \boldsymbol{z}(t)\big)\,\mathrm{d}t + \sigma\big(t, \boldsymbol{z}(t)\big)\,\mathrm{d}W(t)\quad$ (approximate posterior)	Stochastic adjoint sensitivity method	Synthetic data, Motion capture data
Neural SDEs as GANs [Kidger *et al.*, 2021b]	$\mathrm{d}\boldsymbol{x}(t) = \mu_\theta(t, \boldsymbol{x}(t))\,\mathrm{d}t + \sigma_\theta(t, \boldsymbol{x}(t))\,\mathrm{d}W(t)\quad$ (generator) $\mathrm{d}\boldsymbol{h}(t) = f_\phi(t, \boldsymbol{h}(t))\,\mathrm{d}t + g_\phi(t, \boldsymbol{h}(t))\,\mathrm{d}Y(t)\quad$ (discriminator)	Generative adversarial network, Time series classification and prediction	Synthetic data, Google/Alphabet stock, Beijing air quality
Stable Neural SDEs [Oh *et al.*, 2024]	**Neural LSDE:** $\quad\mathrm{d}\boldsymbol{z}(t) = \gamma\big(\overline{\boldsymbol{z}}(t); \theta_\gamma\big)\,\mathrm{d}t + \sigma(t; \theta_\sigma)\,\mathrm{d}W(t)$ **Neural LNSDE:** $\quad\mathrm{d}\boldsymbol{z}(t) = \gamma\big(t, \overline{\boldsymbol{z}}(t); \theta_\gamma\big)\,\mathrm{d}t + \sigma(t; \theta_\sigma)\boldsymbol{z}(t)\,\mathrm{d}W(t)$ **Neural GSDE:** $\quad\dfrac{\mathrm{d}\boldsymbol{z}(t)}{\boldsymbol{z}(t)} = \gamma\big(t, \overline{\boldsymbol{z}}(t); \theta_\gamma\big)\,\mathrm{d}t + \sigma(t; \theta_\sigma)\,\mathrm{d}W(t)$	Irregular time series classification, imputation, and forecasting	UEA/UCR archive, Speech Commands, PhysioNet mortality, PhysioNet Sepsis, MuJoCo physics

Table 1: Comparison of NDE Methods: Formulations, Applications, and Benchmarks

ory to Log-NCDEs. These approaches leverage continuous-time dynamics, enabling flexible modeling of time series.

Recent works [Jhin *et al.*, 2022; Jhin *et al.*, 2023] emphasize training-based approaches for constructing control paths in NCDEs. For instance, Attentive Neural Controlled Differential Equation (ANCDE) [Jhin *et al.*, 2024] extend this line of research by incorporating attention mechanisms for dynamic path construction. This is implemented using two coupled NCDEs: one computes attention values, $\boldsymbol{a}(t) = \sigma(\boldsymbol{h}(t))$ using an activation σ, and the other applies the attention-modulated control path, $Y(t)$, which is a combination of vector $a(t)$ and $X(t)$. Similarly, Li *et al.* (2023) propose Neu-

ral LAD (Latent dynamics), which incorporates an attention-based transformation of the input path gradient $g_v \frac{dX}{dt}$ for continuous-time latent dynamics. In addition, Neural LAD employs a seasonality-trend transformation $h_w(t)$ to capture periodic and slowly varying patterns, and uses $\phi_\gamma(z(t))$ to learn structural relationships within the latent state.

DualDynamics [Oh *et al.*, 2025] combines explicit and implicit mechanisms within a unified framework to improve the modeling of irregular time series data. By integrating Neural ODEs for explicit time evolution and learnable implicit updates for latent state transitions, DualDynamics achieves a balance between interpretability and robustness. These variations of NCDEs have been extensively applied to irregularly-sampled time series or data with missing observations, demonstrating their effectiveness in a variety of tasks including classification, imputation, and forecasting.

5.3 NSDE Methods

Neural SDEs, parameterized by neural networks for both drift and diffusion terms, provide a flexible framework for modeling complex stochastic processes, with foundational contributions from Han *et al.* (2017) and Tzen and Raginsky (2019). Neural Jump SDE [Jia and Benson, 2019] incorporates Poisson jump processes to model discontinuous state changes. Zhang *et al.* (2024) extended this to temporal point processes. While requiring specialized solvers, this approach effectively captures rare, discontinuous events in NSDEs.

Latent SDE [Li *et al.*, 2020] extends the encoding-decoding paradigm of Latent ODE [Rubanova *et al.*, 2019] by incorporating stochastic dynamics, allowing for more flexible modeling of uncertainty and variability. Latent SDE defines a prior process $\tilde{z}(t)$ and an approximate posterior process $z(t)$, both parameterized as Neural SDEs with h_θ, h_ϕ, and σ. During the training, the Kullback–Leibler (KL) divergence measures and minimizes the prior-posterior discrepancy while maintaining uncertainty. On the other hand, Kidger *et al.* (2021b) extends Wasserstein generative adversarial network (GAN) to continuous-time stochastic processes, where both generator and discriminator are formulated as Neural SDEs.

Stable Neural SDEs, introduced in Oh *et al.* (2024), extend Neural SDEs to address stability and robustness in complex systems: Neural Langevin-type SDE (LSDE) with additive noise for invariant measures, Neural Linear Noise SDE (LNSDE) with multiplicative noise for scale-dependent fluctuations, and Neural Geometric SDE (GSDE) with state-dependent noise for exponential dynamics. Note that, these formulations utilize the augmented states with control path, $\overline{z}(t) = \zeta(t, z(t), X(t); \theta_\zeta)$, in order to ensure stability for irregular time series analysis. Therefore, NSDEs extend NODEs by incorporating stochasticity, enabling robust modeling of complex, uncertain, and discontinuous dynamics.

5.4 Empirical Comparison

We present a consolidated summary of diverse empirical results from Oh *et al.* (2024) and Oh *et al.* (2025), encompassing tasks such as interpolation on PhysioNet Mortality, forecasting on MuJoCo, and classification on PhysioNet Sepsis.

We summarize interpolation experiments on the 2012 PhysioNet Mortality dataset [Silva *et al.*, 2012], which contains 37 ICU variables recorded over 48 hours. Following Rubanova *et al.* (2019), timestamps were rounded to the nearest minute, resulting in up to 2880 intervals. As in Shukla and Marlin (2021), observation rates varied from 50% to 90%, using 8000 instances to predict the unobserved values. Benchmark models included RNN-VAE [Chen *et al.*, 2018], mTAND-Full [Shukla and Marlin, 2021], L-ODE-RNN [Chen *et al.*, 2018], L-ODE-ODE [Rubanova *et al.*, 2019], DualDynamics [Oh *et al.*, 2025], Latent SDE [Li *et al.*, 2020], and variants of Neural SDEs [Oh *et al.*, 2024]. Well-designed NDE methods outperform conventional models in interpolation performance, as shown in Table 2.

Methods	Observed %				
	50%	60%	70%	80%	90%
RNN-VAE	13.418	12.594	11.887	11.133	11.470
	±0.008	±0.004	±0.005	±0.007	±0.006
mTAND-Full	4.139	4.018	4.157	4.410	4.798
	±0.029	±0.048	±0.053	±0.149	±0.036
L-ODE-RNN	8.132	8.140	8.171	8.143	8.402
	±0.020	±0.018	±0.030	±0.025	±0.022
L-ODE-ODE	6.721	6.816	6.798	6.850	7.142
	±0.109	±0.045	±0.143	±0.066	±0.066
DualDynamics	3.631	3.659	3.463	3.224	3.114
	±0.049	±0.028	±0.032	±0.044	±0.050
Latent SDE	8.862	8.864	8.686	8.716	8.435
	±0.036	±0.058	±0.122	±0.032	±0.077
Neural SDE	8.592	8.591	8.540	8.318	8.252
	±0.055	±0.052	±0.051	±0.010	±0.023
Neural LSDE	3.799	3.584	3.457	3.262	3.111
	±0.055	±0.055	±0.078	±0.032	±0.076
Neural LNSDE	3.808	3.617	3.405	3.269	3.154
	±0.078	±0.129	±0.089	±0.057	±0.084
Neural GSDE	3.824	3.667	3.493	3.287	3.118
	±0.088	±0.079	±0.024	±0.070	±0.065

Table 2: Interpolation performance (Test MSE $\times 10^{-3}$) versus percent observed time points on PhysioNet Mortality dataset

For forecasting and classification task, we consider a variety of NDE-related methods, including GRU-Δt [Choi *et al.*, 2016], GRU-D [Che *et al.*, 2018], GRU-ODE [De Brouwer *et al.*, 2019], ODE-RNN [Rubanova *et al.*, 2019], Latent-ODE [Rubanova *et al.*, 2019], Augmented-ODE [Dupont *et al.*, 2019], Attentive co-evolving neural ordinary differential equations (ACE-NODE) [Jhin *et al.*, 2021], Neural CDE [Kidger *et al.*, 2020], ANCDE [Jhin *et al.*, 2024], EXtrapolation and InTerpolation-based model (EXIT) [Jhin *et al.*, 2022], LEArnable Path-based model (LEAP) [Jhin *et al.*, 2023], DualDynamics [Oh *et al.*, 2025], and variants of Neural SDEs [Oh *et al.*, 2024] including Neural LSDE, Neural LNSDE, and Neural GSDE.

Table 3 summarizes forecasting results, which used the MuJoCo dataset [Todorov *et al.*, 2012] with the Hopper setup from the DeepMind control suite [Tassa *et al.*, 2018]. Following Jhin *et al.*; Jhin *et al.* (2022; 2023), models were trained to forecast 10 future steps from 50 observed points under 30%, 50%, and 70% missingness. Similar to the previous results, NDE models demonstrate both superior prediction performance under varying missingness and greater memory efficiency in general, in MuJoCo forecasting.

Table 4 summarizes classification results, which used the PhysioNet Sepsis dataset [Reyna *et al.*, 2019], containing

Methods	Test MSE				Memory (MB)
	0% dropped	30% dropped	50% dropped	70% dropped	
GRU-Δt	0.223 ±0.020	0.198 ±0.036	0.193 ±0.015	0.196 ±0.028	533
GRU-D	0.578 ±0.042	0.608 ±0.032	0.587 ±0.039	0.579 ±0.052	569
GRU-ODE	0.856 ±0.016	0.857 ±0.015	0.852 ±0.015	0.861 ±0.015	146
ODE-RNN	0.328 ±0.225	0.274 ±0.213	0.237 ±0.110	0.267 ±0.217	115
Latent-ODE	0.029 ±0.011	0.056 ±0.001	0.055 ±0.004	0.058 ±0.003	314
Augmented-ODE	0.055 ±0.004	0.056 ±0.004	0.057 ±0.005	0.057 ±0.005	286
ACE-NODE	0.039 ±0.003	0.053 ±0.007	0.053 ±0.005	0.052 ±0.006	423
Neural CDE	0.028 ±0.002	0.027 ±0.000	0.027 ±0.001	0.026 ±0.001	52.1
ANCDE	0.026 ±0.001	0.025 ±0.001	0.025 ±0.001	0.024 ±0.001	79.2
EXIT	0.026 ±0.000	0.025 ±0.004	0.026 ±0.000	0.026 ±0.001	127
LEAP	0.022 ±0.002	0.022 ±0.001	0.022 ±0.002	0.022 ±0.001	144
DualDynamics	0.006 ±0.001	0.008 ±0.001	0.008 ±0.001	0.008 ±0.001	318
Neural SDE	0.028 ±0.004	0.029 ±0.001	0.029 ±0.001	0.027 ±0.000	234
Neural LSDE	0.013 ±0.000	0.014 ±0.001	0.013 ±0.000	0.013 ±0.001	249
Neural LNSDE	0.012 ±0.001	0.014 ±0.001	0.013 ±0.001	0.014 ±0.000	273
Neural GSDE	0.013 ±0.001	0.013 ±0.001	0.013 ±0.000	0.014 ±0.000	306

Table 3: Forecasting MSE and memory usage versus percent dropped time points on MuJoCo dataset

40,335 patients and 34 temporal variables. Following Kidger *et al.* (2020), we compared classification with and without observation intensity (OI), an index reflecting patient severity. Classification performance was measured by AUROC. On the PhysioNet Sepsis task, NDE-based models consistently achieve higher AUROC with significantly lower memory usage compared to conventional baselines.

Methods	Test AUROC		Memory (MB)	
	OI	No OI	OI	No OI
GRU-Δt	0.878 ±0.006	0.840 ±0.007	837	826
GRU-D	0.871 ±0.022	0.850 ±0.013	889	878
GRU-ODE	0.852 ±0.010	0.771 ±0.024	454	273
ODE-RNN	0.874 ±0.016	0.833 ±0.020	696	686
Latent-ODE	0.787 ±0.011	0.495 ±0.002	133	126
ACE-NODE	0.804 ±0.010	0.514 ±0.003	194	218
Neural CDE	0.880 ±0.006	0.776 ±0.009	244	122
ANCDE	0.900 ±0.002	0.823 ±0.003	285	129
EXIT	0.913 ±0.002	0.836 ±0.003	257	127
DualDynamics	0.918 ±0.003	0.873 ±0.004	453	233
Neural SDE	0.799 ±0.007	0.796 ±0.006	368	240
Neural LSDE	0.909 ±0.004	0.879 ±0.008	373	436
Neural LNSDE	0.911 ±0.002	0.881 ±0.002	341	445
Neural GSDE	0.909 ±0.001	0.884 ±0.002	588	280

Table 4: AUROC and memory usage on PhysioNet Sepsis

Across diverse time series related tasks, NDE-based methods consistently achieve better performance compared to conventional models. Their advantages are particularly pronounced in dynamic prediction settings and irregularly sampled data, confirming their suitability for real-world time series modeling in a continuous-time manner.

6 Discussion and Future Directions

Computational Challenges. NDEs face scalability challenges with continuous-time solvers. Fixed-step integrators or adaptive integrators, while accurate, can lead to unpredictable computation times in stiff regions [Chen *et al.*, 2018; Kim *et al.*, 2021; Kidger, 2021]. Specialized techniques like parallel-in-time integration and GPU acceleration offer promising solutions [Rackauckas *et al.*, 2020; Gholami *et al.*, 2019]. Future research should focus on developing comprehensive integration schemes that can automatically balance computational efficiency with numerical accuracy, particularly for large-scale applications in real-time settings.

Theoretical Development. Current theoretical understanding requires expansion, particularly for non-stationary and noisy data. While stability guarantees exist for NODEs and NCDEs through Lipschitz constraints [Massaroli *et al.*, 2020b], similar guarantees for NSDEs remain incomplete [Oh *et al.*, 2024]. Drift-diffusion analysis under strong noise conditions needs further investigation [Tzen and Raginsky, 2019; Li *et al.*, 2020]. Development of rigorous frameworks for analyzing convergence properties and error bounds in these stochastic settings would significantly advance both theoretical foundations and practical applications of NDEs.

Physics-Informed NDEs. Physics-Informed NDEs (or Physics-Informed Neural Networks) incorporate domain knowledge, such as Partial Differential Equations (PDEs) constraints and conservation laws, to align modeled dynamics with physical systems. Recent advances in physics-informed architectures have demonstrated remarkable success in capturing multi-scale phenomena and handling noisy measurements in complex physical systems [Rudy *et al.*, 2017; Raissi *et al.*, 2019; Cuomo *et al.*, 2022]. By embedding structural priors, these approaches ensure interpretability and reliability in real-world tasks.

Hybrid Architectures. Hybrid architectures combine NDEs with models like graph neural networks or Transformers to capture complex time series patterns. Graph-based NDEs handle spatio-temporal data [Poli *et al.*, 2019; Choi *et al.*, 2022], while attention mechanisms address long-range dependencies and irregular sampling [Jhin *et al.*, 2024; Li *et al.*, 2024]. Furthermore, Oh *et al.* (2025) integrates explicit and implicit mechanisms for irregular time series analysis. Future research directions should focus on developing more efficient training algorithms for these hybrid architectures, and exploring applications in real-world domains where both temporal dynamics and structural relationships are important.

7 Conclusion

This paper has provided a focused and comprehensive survey of NDEs, specifically for time series analysis. While existing literature includes reviews on broader topics or specific niches within this domain, a systematic consolidation and comparative analysis of the core architectural families and their variants, tailored to the unique challenges of temporal data, has been largely absent. Our work addresses this gap,

offering a structured synthesis of advancements and outlining future research directions in this new paradigm.

NDEs represent a significant advancement in time series modeling, offering a principled approach to handling continuous dynamics. Through various formulations, including NODEs, NCDEs, and NSDEs, they provide flexible frameworks for complex temporal data, including cases with irregular sampling or missing data. While challenges remain in computational efficiency and theoretical understanding, ongoing developments in solver techniques, stability analysis, and hybrid architectures continue to enhance their capabilities. Beyond current applications in classification, interpolation, and forecasting, NDEs show potential in anomaly detection, reinforcement learning, and multi-agent systems. As the field evolves, NDEs are increasingly positioned to address sophisticated time series challenges across diverse applications.

Acknowledgments

References

[Biloš *et al.*, 2021] Marin Biloš, Johanna Sommer, Syama Sundar Rangapuram, Tim Januschowski, and Stephan Günnemann. Neural flows: Efficient alternative to neural odes. *Advances in Neural Information Processing Systems*, 34:21325–21337, 2021.

[Box *et al.*, 2015] George EP Box, Gwilym M Jenkins, Gregory C Reinsel, and Greta M Ljung. *Time series analysis: forecasting and control*. John Wiley & Sons, 2015.

[Che *et al.*, 2018] Zhengping Che, Sanjay Purushotham, Kyunghyun Cho, David Sontag, and Yan Liu. Recurrent neural networks for multivariate time series with missing values. *Scientific reports*, 8(1):6085, 2018.

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

[Choi *et al.*, 2016] Edward Choi, Mohammad Taha Bahadori, Andy Schuetz, Walter F Stewart, and Jimeng Sun. Doctor ai: Predicting clinical events via recurrent neural networks. In *Machine learning for healthcare conference*, pages 301–318. PMLR, 2016.

[Choi *et al.*, 2022] Jeongwhan Choi, Hwangyong Choi, Jeehyun Hwang, and Noseong Park. Graph neural controlled differential equations for traffic forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, volume 36, pages 6367–6374, 2022.

[Chung *et al.*, 2014] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

[Cuomo *et al.*, 2022] Salvatore Cuomo, Vincenzo Schiano Di Cola, Fabio Giampaolo, Gianluigi Rozza, Maziar Raissi, and Francesco Piccialli. Scientific machine learning through physics–informed neural networks: Where we are and what's next. *Journal of Scientific Computing*, 92(3):88, 2022.

[De Brouwer *et al.*, 2019] Edward De Brouwer, Jaak Simm, Adam Arany, and Yves Moreau. Gru-ode-bayes: Continuous modeling of sporadically-observed time series. *Advances in neural information processing systems*, 32, 2019.

[Dupont *et al.*, 2019] Emilien Dupont, Arnaud Doucet, and Yee Whye Teh. Augmented neural odes. *Advances in Neural Information Processing Systems*, 32, 2019.

[Durbin and Koopman, 2012] James Durbin and Siem Jan Koopman. *Time series analysis by state space methods*, volume 38. OUP Oxford, 2012.

[Finlay *et al.*, 2020a] Chris Finlay, Augusto Gerolin, Adam M Oberman, and Aram-Alexandre Pooladian. Learning normalizing flows from entropy-kantorovich potentials. *arXiv preprint arXiv:2006.06033*, 2020.

[Finlay *et al.*, 2020b] Chris Finlay, Jörn-Henrik Jacobsen, Levon Nurbekyan, and Adam Oberman. How to train your neural ode: the world of jacobian and kinetic regularization. In *International conference on machine learning*, pages 3154–3164. PMLR, 2020.

[Gholami *et al.*, 2019] Amir Gholami, Kurt Keutzer, and George Biros. Anode: Unconditionally accurate memory-efficient gradients for neural odes. *arXiv preprint arXiv:1902.10298*, 2019.

[Ghosh *et al.*, 2020] Arnab Ghosh, Harkirat Behl, Emilien Dupont, Philip Torr, and Vinay Namboodiri. Steer: Simple temporal regularization for neural ode. *Advances in Neural Information Processing Systems*, 33:14831–14843, 2020.

[Goodfellow *et al.*, 2016] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

[Grathwohl *et al.*, 2018] Will Grathwohl, Ricky TQ Chen, Jesse Bettencourt, Ilya Sutskever, and David Duvenaud. Ffjord: Free-form continuous dynamics for scalable reversible generative models. *arXiv preprint arXiv:1810.01367*, 2018.

[Grathwohl *et al.*, 2019] Will Grathwohl, Ricky T. Q. Chen, Jesse Bettencourt, and David Duvenaud. Scalable reversible generative models with free-form continuous dynamics. In *International Conference on Learning Representations*, 2019.

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

[Han *et al.*, 2017] Jiequn Han, Arnulf Jentzen, et al. Deep learning-based numerical methods for high-dimensional parabolic partial differential equations and backward stochastic differential equations. *Communications in mathematics and statistics*, 5(4):349–380, 2017.

[Hochreiter and Schmidhuber, 1997] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[Jhin *et al.*, 2021] Sheo Yon Jhin, Minju Jo, Taeyong Kong, Jinsung Jeon, and Noseong Park. Ace-node: Attentive co-evolving neural ordinary differential equations. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pages 736–745, 2021.

[Jhin *et al.*, 2022] Sheo Yon Jhin, Jaehoon Lee, Minju Jo, Seungji Kook, Jinsung Jeon, Jihyeon Hyeong, Jayoung Kim, and Noseong Park. Exit: Extrapolation and interpolation-based neural controlled differential equations for time-series classification and forecasting. In *Proceedings of the ACM Web Conference 2022*, pages 3102–3112, 2022.

[Jhin *et al.*, 2023] Sheo Yon Jhin, Minju Jo, Seungji Kook, and Noseong Park. Learnable path in neural controlled differential equations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 8014–8022, 2023.

[Jhin *et al.*, 2024] Sheo Yon Jhin, Heejoo Shin, Sujie Kim, Seoyoung Hong, Minju Jo, Solhee Park, Noseong Park, Seungbeom Lee, Hwiyoung Maeng, and Seungmin Jeon. Attentive neural controlled differential equations for time-series classification and forecasting. *Knowledge and Information Systems*, 66(3):1885–1915, 2024.

[Jia and Benson, 2019] Junteng Jia and Austin R Benson. Neural jump stochastic differential equations. *Advances in Neural Information Processing Systems*, 32, 2019.

[Kelly *et al.*, 2020] Jacob Kelly, Jesse Bettencourt, Matthew J Johnson, and David K Duvenaud. Learning differential equations that are easy to solve. *Advances in Neural Information Processing Systems*, 33:4370–4380, 2020.

[Kidger *et al.*, 2020] Patrick Kidger, James Morrill, James Foster, and Terry Lyons. Neural controlled differential equations for irregular time series. *Advances in Neural Information Processing Systems*, 33:6696–6707, 2020.

[Kidger *et al.*, 2021a] Patrick Kidger, Ricky T. Q. Chen, and Terry J. Lyons. "hey, that's not an ode": Faster ode adjoints via seminorms. *International Conference on Machine Learning*, 2021.

[Kidger *et al.*, 2021b] Patrick Kidger, James Foster, Xuechen Li, Harald Oberhauser, and Terry Lyons. Neural SDEs as Infinite-Dimensional GANs. *International Conference on Machine Learning*, 2021.

[Kidger *et al.*, 2021c] Patrick Kidger, James Foster, Xuechen Chen Li, and Terry Lyons. Efficient and accurate gradients for neural sdes. *Advances in Neural Information Processing Systems*, 34:18747–18761, 2021.

[Kidger, 2021] Patrick Kidger. *On Neural Differential Equations*. PhD thesis, University of Oxford, 2021.

[Kim *et al.*, 2021] Suyong Kim, Weiqi Ji, Sili Deng, Yingbo Ma, and Christopher Rackauckas. Stiff neural ordinary differential equations. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 31(9), 2021.

[Kobyzev *et al.*, 2020] Ivan Kobyzev, Simon JD Prince, and Marcus A Brubaker. Normalizing flows: An introduction and review of current methods. *IEEE transactions on pattern analysis and machine intelligence*, 43(11):3964–3979, 2020.

[Lechner and Hasani, 2020] Mathias Lechner and Ramin Hasani. Learning long-term dependencies in irregularly-sampled time series. *arXiv preprint arXiv:2006.04418*, 2020.

[Li *et al.*, 2020] Xuechen Li, Ting-Kam Leonard Wong, Ricky TQ Chen, and David Duvenaud. Scalable gradients for stochastic differential equations. In *International Conference on Artificial Intelligence and Statistics*, pages 3870–3882. PMLR, 2020.

[Li *et al.*, 2023] Jianguo Li, Zhanxing Zhu, et al. Neural lad: a neural latent dynamics framework for times series modeling. *Advances in Neural Information Processing Systems*, 36:17345–17356, 2023.

[Li *et al.*, 2024] Jianguo Li, Zhanxing Zhu, et al. Neural lad: a neural latent dynamics framework for times series modeling. *Advances in Neural Information Processing Systems*, 36, 2024.

[Liu *et al.*, 2019] Xuanqing Liu, Tesi Xiao, Si Si, Qin Cao, Sanjiv Kumar, and Cho-Jui Hsieh. Neural sde: Stabilizing neural ode networks with stochastic noise. *arXiv preprint arXiv:1906.02355*, 2019.

[Liu *et al.*, 2020] Xuanqing Liu, Tesi Xiao, Si Si, Qin Cao, Sanjiv Kumar, and Cho-Jui Hsieh. How does noise help robustness? explanation and exploration under the neural sde framework. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 282–290, 2020.

[Lu *et al.*, 2018] Yiping Lu, Aoxiao Zhong, Quanzheng Li, and Bin Dong. Beyond finite layer neural networks: Bridging deep architectures and numerical differential equations. In *International Conference on Machine Learning*, pages 3276–3285. PMLR, 2018.

[Massaroli *et al.*, 2020a] Stefano Massaroli, Michael Poli, Michelangelo Bin, Jinkyoo Park, Atsushi Yamashita, and Hajime Asama. Stable neural flows. *arXiv preprint arXiv:2003.08063*, 2020.

[Massaroli *et al.*, 2020b] Stefano Massaroli, Michael Poli, Jinkyoo Park, Atsushi Yamashita, and Hajime Asama. Dissecting neural odes. *Advances in Neural Information Processing Systems*, 33:3952–3963, 2020.

[Medsker and Jain, 1999] Larry Medsker and Lakhmi C Jain. *Recurrent neural networks: design and applications*. CRC press, 1999.

[Morrill *et al.*, 2021a] James Morrill, Patrick Kidger, Lingyi Yang, and Terry Lyons. Neural controlled differential equations for online prediction tasks. *arXiv preprint arXiv:2106.11028*, 2021.

[Morrill *et al.*, 2021b] James Morrill, Cristopher Salvi, Patrick Kidger, and James Foster. Neural rough differential equations for long time series. In *International Conference on Machine Learning*, pages 7829–7838. PMLR, 2021.

[Mozer *et al.*, 2017] Michael C Mozer, Denis Kazakov, and Robert V Lindsey. Discrete event, continuous time rnns. *arXiv preprint arXiv:1710.04110*, 2017.

[Oh *et al.*, 2024] YongKyung Oh, Dongyoung Lim, and Sungil Kim. Stable neural stochastic differential equations in analyzing irregular time series data. In *The Twelfth International Conference on Learning Representations*, 2024.

[Oh *et al.*, 2025] YongKyung Oh, Dongyoung Lim, and Sungil Kim. Dualdynamics: Synergizing implicit and explicit methods for robust irregular time series analysis. In *The 39th Annual AAAI Conference on Artificial Intelligence*, 2025.

[Papamakarios *et al.*, 2021] George Papamakarios, Eric Nalisnick, Danilo Jimenez Rezende, Shakir Mohamed, and Balaji Lakshminarayanan. Normalizing flows for probabilistic modeling and inference. *The Journal of Machine Learning Research*, 22(1):2617–2680, 2021.

[Pascanu *et al.*, 2013] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. On the difficulty of training recurrent neural networks. In *Proceedings of the 30th International Conference on International Conference on Machine Learning-Volume 28*, pages III–1310, 2013.

[Peyré *et al.*, 2019] Gabriel Peyré, Marco Cuturi, et al. Computational optimal transport: With applications to data science. *Foundations and Trends® in Machine Learning*, 11(5-6):355–607, 2019.

[Poli *et al.*, 2019] Michael Poli, Stefano Massaroli, Junyoung Park, Atsushi Yamashita, Hajime Asama, and Jinkyoo Park. Graph neural ordinary differential equations. *arXiv preprint arXiv:1911.07532*, 2019.

[Pontryagin, 2018] Lev Semenovich Pontryagin. *Mathematical theory of optimal processes*. Routledge, 2018.

[Rackauckas *et al.*, 2020] Christopher Rackauckas, Yingbo Ma, Julius Martensen, Collin Warner, Kirill Zubov, Rohit Supekar, Dominic Skinner, Ali Ramadhan, and Alan Edelman. Universal differential equations for scientific machine learning. *arXiv preprint arXiv:2001.04385*, 2020.

[Rae *et al.*, 2019] Jack W Rae, Anna Potapenko, Siddhant M Jayakumar, and Timothy P Lillicrap. Compressive transformers for long-range sequence modelling. *arXiv preprint arXiv:1911.05507*, 2019.

[Raissi *et al.*, 2019] Maziar Raissi, Paris Perdikaris, and George E Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational physics*, 378:686–707, 2019.

[Rajkomar *et al.*, 2018] Alvin Rajkomar, Eyal Oren, Kai Chen, Andrew M Dai, Nissan Hajaj, Michaela Hardt, Peter J Liu, Xiaobing Liu, Jake Marcus, Mimi Sun, et al. Scalable and accurate deep learning with electronic health records. *NPJ digital medicine*, 1(1):1–10, 2018.

[Reyna *et al.*, 2019] Matthew A Reyna, Chris Josef, Salman Seyedi, Russell Jeter, Supreeth P Shashikumar, M Brandon Westover, Ashish Sharma, Shamim Nemati, and Gari D Clifford. Early prediction of sepsis from clinical data: the physionet/computing in cardiology challenge 2019. In *2019 Computing in Cardiology (CinC)*, pages Page–1. IEEE, 2019.

[Rubanova *et al.*, 2019] Yulia Rubanova, Ricky TQ Chen, and David K Duvenaud. Latent ordinary differential equations for irregularly-sampled time series. *Advances in neural information processing systems*, 32, 2019.

[Rudy *et al.*, 2017] Samuel H Rudy, Steven L Brunton, Joshua L Proctor, and J Nathan Kutz. Data-driven discovery of partial differential equations. *Science advances*, 3(4):e1602614, 2017.

[Ruiz-Balet and Zuazua, 2023] Domenec Ruiz-Balet and Enrique Zuazua. Neural ode control for classification, approximation, and transport. *SIAM Review*, 65(3):735–773, 2023.

[Rumelhart *et al.*, 1986] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by backpropagating errors. *nature*, 323(6088):533–536, 1986.

[Shukla and Marlin, 2021] Satya Narayan Shukla and Benjamin M Marlin. Multi-time attention networks for irregularly sampled time series. *arXiv preprint arXiv:2101.10318*, 2021.

[Silva *et al.*, 2012] Ikaro Silva, George Moody, Daniel J Scott, Leo A Celi, and Roger G Mark. Predicting in-hospital mortality of icu patients: The physionet/computing in cardiology challenge 2012. In *2012 Computing in Cardiology*, pages 245–248. IEEE, 2012.

[Sonoda and Murata, 2019] Sho Sonoda and Noboru Murata. Transport analysis of infinitely deep neural network. *The Journal of Machine Learning Research*, 20(1):31–82, 2019.

[Sun *et al.*, 2020] Chenxi Sun, Shenda Hong, Moxian Song, and Hongyan Li. A review of deep learning methods for irregularly sampled medical time series data. *arXiv preprint arXiv:2010.12493*, 2020.

[Tassa *et al.*, 2018] Yuval Tassa, Yotam Doron, Alistair Muldal, Tom Erez, Yazhe Li, Diego de Las Casas, David Budden, Abbas Abdolmaleki, Josh Merel, Andrew Lefrancq, et al. Deepmind control suite. *arXiv preprint arXiv:1801.00690*, 2018.

[Todorov *et al.*, 2012] Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ international conference on intelligent robots and systems*, pages 5026–5033. IEEE, 2012.

[Tzen and Raginsky, 2019] Belinda Tzen and Maxim Raginsky. Neural stochastic differential equations: Deep latent gaussian models in the diffusion limit. *arXiv preprint arXiv:1905.09883*, 2019.

[Vaswani *et al.*, 2017] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[Villani and others, 2009] Cédric Villani et al. *Optimal transport: old and new*, volume 338. Springer, 2009.

[Walker *et al.*, 2024] Benjamin Walker, Andrew Donald McLeod, Tiexin Qin, Yichuan Cheng, Haoliang Li, and Terry Lyons. Log neural controlled differential equations: The lie brackets make a difference. In *Forty-first International Conference on Machine Learning*, 2024.

[Weerakody *et al.*, 2021] Philip B Weerakody, Kok Wai Wong, Guanjin Wang, and Wendell Ela. A review of irregular time series data handling with gated recurrent neural networks. *Neurocomputing*, 441:161–178, 2021.

[Wen *et al.*, 2023] Qingsong Wen, Tian Zhou, Chaoli Zhang, Weiqi Chen, Ziqing Ma, Junchi Yan, and Liang Sun. Transformers in time series: a survey. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence*, pages 6778–6786, 2023.

[Yildiz *et al.*, 2019] Cagatay Yildiz, Markus Heinonen, and Harri Lahdesmaki. Ode2vae: Deep generative second order odes with bayesian neural networks. *Advances in Neural Information Processing Systems*, 32, 2019.

[Zhang *et al.*, 2024] Shuai Zhang, Chuan Zhou, Yang Aron Liu, Peng Zhang, Xixun Lin, and Zhi-Ming Ma. Neural jump-diffusion temporal point processes. In *Forty-first International Conference on Machine Learning*, 2024.

[Zheng *et al.*, 2024] Su Zheng, Zhengqi Gao, Fan-Keng Sun, Duane S Boning, Bei Yu, and Martin Wong. Improving neural ode training with temporal adaptive batch normalization. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

[Zhuang *et al.*, 2020] Juntang Zhuang, Nicha Dvornek, Xiaoxiao Li, Sekhar Tatikonda, Xenophon Papademetris, and James Duncan. Adaptive checkpoint adjoint method for gradient estimation in neural ode. In *International Conference on Machine Learning*, pages 11639–11649. PMLR, 2020.