DG-FDE: DYNAMIC GRAPH REPRESENTATION LEARNING VIA GRAPH NEURAL FRACTIONAL DIFFERENTIAL EQUATIONS

Anonymous authors
Paper under double-blind review

ABSTRACT

Dynamic graph representation learning learns node embeddings from evolving graphs for node property and temporal link prediction. A core challenge is that node states depend on multi-scale historical interactions, not only instantaneous graph structures. Existing continuous-time dynamic graph models rely on integer-order dynamical systems, which cannot explicitly encode long-range nonlocal history within a unified dynamics . Inspired by the nonlocal memory of fractional calculus, we propose DG-FDE, a continuous-time model that casts node representation evolution as a Caputo fractional differential equation. Its equivalent Volterra integral form uses a GNN to model time-dependent graph responses and a fractional memory kernel to aggregate historical signals. This unifies graph propagation and multi-scale temporal memory in one evolution equation. We prove finite-horizon well-posedness and derive an explicit characterization of how historical graph snapshots shape node representations. Experiments show DG-FDE outperforms its integer-order variant and strong dynamic graph baselines on standard tasks.

1 INTRODUCTION

Graphs provide an effective representation of entities and their interactions in complex systems (Khoshraftar & An, 2024). However, many real-world networks exhibit temporal dynamics. For example, user interactions in social networks, transportation networks, and relational information in knowledge graphs change over time (Kazemi et al., 2020; Longa et al., 2023). Such temporal evolution motivates dynamic graphs, which model time-varying graph structures and attributes for tasks such as node property prediction and temporal link prediction (Skarding et al., 2021).

Dynamic graphs can exhibit history-dependent evolution, in which past interactions continue to influence future structural changes across multiple temporal scales (Kazemi et al., 2020; Skarding et al., 2021; Longa et al., 2023). Temporal interaction history has therefore become an important source of information for dynamic graph representation learning; models such as JODIE, DyRep, and TGN explicitly maintain evolving states or memories to carry information across interactions (Kumar et al., 2019; Trivedi et al., 2019; Rossi et al., 2020). Without such history, node representations derived solely from the current graph structure may fail to capture temporal dependencies that persist across interactions. Despite substantial progress in modeling temporal interaction history and strong performance on downstream tasks such as node property prediction and temporal link prediction, the modeling of temporal interaction history remains far from satisfactory. **On Temporal Dependency.** Interactions in dynamic graphs may depend on historical information at multiple temporal scales (Kazemi et al., 2020; Longa et al., 2023). Although recent interactions often provide strong predictive signals, distant events may remain relevant to future relationships; accordingly, prior work has developed recurrent, memory-based, and window-based mechanisms to preserve information over time (Kumar et al., 2019; Rossi et al., 2020; Zhu et al., 2023). For example, in social networks, individuals may reconnect with old acquaintances after prolonged periods of inactivity. Methods that focus primarily on recent history may therefore overlook such long-range dependencies. Effective dynamic graph representations should capture temporal interaction histories across multiple scales as graph structures evolve. **On Structural-Temporal Coupling.** Representative dynamic graph models jointly incorporate structural and temporal information through distinct

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

modules, such as graph convolution followed by recurrent updates or separate spatial and temporal attention (Li et al., 2018; Guo et al., 2019; Liu & Zhang, 2024). Although effective, such modular designs keep structural propagation and temporal state evolution operationally separate. Consequently, the influence of evolving graph structures on node representations is typically implicit rather than being directly characterized within a unified dynamical framework. **On Historical Contribution.** Recurrent, attention-based, and memory-based dynamic graph models typically compress temporal interaction history into hidden states, embeddings, memory vectors, or evolving model parameters (Pareja et al., 2020; Sankar et al., 2020; Kumar et al., 2019; Rossi et al., 2020). Consequently, the specific contributions of different historical graph structures to the current node representation are difficult to distinguish explicitly, while their accumulation over time and persistent long-term effects also remain insufficiently characterized. Existing methods therefore still provide limited capability for explicitly modeling the contributions of different historical intervals. We therefore need a dynamic graph model that couples structural and temporal evolution while effectively capturing temporal interaction histories across multiple scales.

Fractional calculus provides a mathematical framework for systems with memory effects and nonlocal dependencies (Podlubny, 1999; Diethelm, 2010; Coelho et al., 2025; Kang et al., 2025). Unlike ordinary differential equations, whose evolution is determined by local-in-time dynamics, time-fractional differential equations allow the current state to depend explicitly on an extended history of past state changes. This property is particularly relevant to dynamic graphs, where current node states or future interactions may depend not only on recent events but also on interactions that occurred much earlier. For example, persistent relationships or recurring interaction patterns may remain informative even after long inactive periods. Fractional dynamics therefore provide a natural mechanism for aggregating historical graph-dependent responses across different temporal scales while retaining the influence of distant interactions.

Motivated by this perspective, we propose the Dynamic Graph Fractional Differential Equation model (DG-FDE), in which node representations evolve under the accumulated influence of historical graph interactions. The graph vector field drives Caputo fractional dynamics (Podlubny, 1999; Diethelm, 2010), allowing evolving graph structures to shape both the current update and its influence on subsequent representations. A fractional memory kernel carries these responses across the input window, with the fractional order regulating the relative influence of recent and distant history. Temporal dependencies thus emerge from the accumulation of graph-conditioned responses within the same continuous-time dynamics, extending the local-in-time evolution of integer-order graph differential-equation models (Poli et al., 2019; Berndt et al., 2025; Qin et al., 2026). The equivalent Volterra integral formulation makes this accumulation explicit, revealing how different historical graph intervals contribute to the current node state.

Our main contributions are as follows:

- **Fractional dynamic graph framework.** We propose a general continuous-time dynamic graph neural network framework that formulates node representation learning on dynamic graphs as a fractional dynamical process (Section 4.1).
- **Well-posedness and stability.** We provide a rigorous theoretical foundation for DG-FDE. Based on the Banach fixed-point theorem, we establish the well-posedness of the resulting neural integral equation (Section 4.2, Theorem 1 and Proposition 1).
- **Nonlocal memory characterization.** We characterize the nonlocal memory of DG-FDE through a history-dependent restart identity and a decomposition over graph intervals (Section 4.3, Propositions 2 and 3).
- **Empirical evaluation.** We evaluate DG-FDE on real-world dynamic graph datasets. The results show that graph-conditioned fractional dynamics consistently improve predictive performance over the corresponding integer-order model while surpassing specialized dynamic graph architectures (Section 5).

2 Related Work

Dynamic Graph Representation Learning. A common paradigm in dynamic graph representation learning is to model structural and temporal dependencies with separate components. DCRNN (Li et al., 2018) combines graph convolution (Kipf & Welling, 2017) with a GRU (Cho et al., 2014),

ASTGCN (Guo et al., 2019) uses separate spatial and temporal attention, and STIDGCN (Liu & Zhang, 2024) captures heterogeneous temporal patterns through interactive learning. In these models, historical information is typically propagated through successive intermediate states and compressed into embeddings or hidden representations. Long-range effects may therefore be attenuated or overwritten, and the contribution of individual historical graph intervals remains implicit. In contrast, DG-FDE jointly models graph-dependent dynamics and temporal memory within a unified dynamical system, allowing historical graph contributions to be characterized across multiple temporal scales.

Temporal point process models, such as GHP (Shang & Sun, 2019), M2DNE (Lu et al., 2019), and Hawkes-based methods (Qi et al., 2025), model the occurrence of future interaction events through history-dependent conditional intensities. Many employ exponentially decaying influence functions and thus emphasize recent interactions. Snapshot-based approaches instead use latent-variable models (Hajiramezanali et al., 2019), hyperbolic representations (Yang et al., 2021), or meta-learning (You et al., 2022), but long-range temporal effects are still represented only implicitly. While temporal point process models primarily characterize the intensity of interaction events, DG-FDE models the continuous-time evolution of node representations conditioned on an evolving graph structure. Its fractional order directly regulates the relative influence of historical graph-conditioned responses across different temporal scales.

Graph Neural Differential Equations. Graph neural differential-equation models provide a continuous-time formulation of graph representation learning. Graph Neural ODEs (Poli et al., 2019) formulate GNNs as continuous-depth systems, while Neural CDEs (Kidger et al., 2020) model irregular sequences through continuous control paths. GN-CDE (Qin et al., 2026) extends this framework to dynamic graphs using interpolated graph paths, and PENG-CDE (Berndt et al., 2025) introduces permutation-equivariant graph controlled differential equations. However, integer-order dynamics are local in time, so historical effects are mainly propagated through the latent state or control path rather than explicitly accumulated as graph-dependent historical responses. Existing fractional graph models mainly apply fractional operators to spatial propagation or feature evolution on fixed graphs. FLODE (Maskey et al., 2023) uses fractional graph operators, FROND (Kang et al., 2024) models continuous-depth dynamics, and NvoFDE (Cui et al., 2025) and FADE (Kang et al., 2025) introduce adaptive orders and attention kernels, respectively. In contrast, DG-FDE applies the Caputo fractional operator along observation time, explicitly accumulating historical graph-conditioned responses across evolving graph snapshots. This shifts the role of fractional memory from feature propagation within a fixed graph to temporal representation learning over a changing graph. To the best of our knowledge, this work provides the first rigorous well-posedness analysis for Caputo fractional graph dynamics with evolving topologies. We further characterize historical graph contributions through a restart identity and a graph-interval decomposition.

3 PRELIMINARIES

Dynamic graph. We consider a discrete-time dynamic graph observed at times $t_0 < t_1 < \cdots < t_N$, denoted by $\mathcal{G}_{0:N} := \big((t_k, \mathcal{G}_{t_k})\big)_{k=0}^{N}$, where $t_k \in \mathbb{R}$. Each snapshot is given by $\mathcal{G}_{t_k} = (\mathcal{V}, \mathcal{E}_{t_k})$, where $\mathcal{V} = \{v_1, \ldots, v_n\}$ is a fixed node set with $n = |\mathcal{V}|$, and $\mathcal{E}_{t_k} \subseteq \mathcal{V} \times \mathcal{V}$ is the edge set at time t_k. Its topology is encoded by an adjacency matrix $\mathbf{A}_{t_k} \in \{0,1\}^{n\times n}$, where $[\mathbf{A}_{t_k}]_{ij} = 1$ if $(v_i, v_j) \in \mathcal{E}_{t_k}$ and is zero otherwise. To model the graph continuously between observations, we define the piecewise-constant graph path $\mathbf{A}(t) = \mathbf{A}_{t_k}$ for $t \in [t_k, t_{k+1})$ and $k = 0, \ldots, N-1$, with $\mathbf{A}(t_N) = \mathbf{A}_{t_N}$. The goal of dynamic graph representation learning is to learn a time-dependent latent representation matrix $\mathbf{Z}(t) \in \mathbb{R}^{n\times d}$ from the observed graph sequence, where the i-th row $\mathbf{z}_i(t)^\top$ represents the evolving d-dimensional embedding of node v_i.

Neural fractional differential equations (Neural FDEs) generalize the integer-order derivative in the state equation to a fractional-order derivative. We recall two fractional operators used in the sequel.

Riemann–Liouville Fractional Integral. Let $\alpha \in \mathbb{R}_+$ and $f \in L^1([t_0, T])$. The Riemann–Liouville fractional integral operator of order α is defined as (Podlubny, 1999; Diethelm, 2010)

$$J_{t_0}^{\alpha} f(t) := \frac{1}{\Gamma(\alpha)} \int_{t_0}^{t} (t-\tau)^{\alpha-1} f(\tau)\, d\tau, \tag{1}$$

where the integral is defined for almost every $t \in (t_0, T]$, and $\Gamma(\cdot)$ is the Gamma function. When $\alpha = 0$, we set $J_{t_0}^0 := I$, where I is the identity operator.

Caputo Fractional Derivative. Let $\alpha \in (0, 1)$ and $f \in C^1([t_0, T])$. The Caputo fractional derivative of order α is defined as (Podlubny, 1999; Diethelm, 2010)

$$ {}_{t_0}^{C}D_t^{\alpha} f(t) = \frac{1}{\Gamma(1-\alpha)} \int_{t_0}^{t} (t-\tau)^{-\alpha} f'(\tau)\, d\tau, \tag{2} $$

where $f'(\tau) = \frac{d}{d\tau} f(\tau)$ denotes the ordinary first-order derivative of f with respect to its time argument, and $t \in (t_0, T]$. For vector-valued or matrix-valued functions, these operators are applied elementwise.

4 METHODOLOGY

This section introduces DG-FDE and its mild Volterra integral formulation in Section 4.1. We establish finite-horizon well-posedness in Section 4.2 and analyze historical graph contributions and the connection to Graph Neural ODEs in Section 4.3. Finally, Sections 4.4 and 4.5 describe the numerical solver and implementation details, respectively.

4.1 DG-FDE FRAMEWORK

Given a graph path $\mathbf{A} : [t_0, T] \to \mathbb{R}^{n\times n}$, DG-FDE models the latent node representation trajectory $\mathbf{Z} : [t_0, T] \to \mathbb{R}^{n\times d}$ using the following graph-aware Caputo fractional differential equation:

$$ {}_{t_0}^{\mathrm{C}}D_t^{\alpha} \mathbf{Z}(t) = f_\theta\big(\mathbf{Z}(t), \mathbf{A}(t)\big). \tag{3} $$

Here, $t \in (t_0, T]$, the Caputo derivative is defined in section 3, $\alpha \in (0, 1)$ is the fractional order, and $f_\theta : \mathbb{R}^{n\times d} \times \mathbb{R}^{n\times n} \to \mathbb{R}^{n\times d}$ is a learnable graph neural network. The trajectory is initialized by $\mathbf{Z}(t_0) = \zeta_\theta\big(\mathbf{A}_{t_0}\big)$, where $\zeta_\theta : \mathbb{R}^{n\times n} \to \mathbb{R}^{n\times d}$ encodes the initial graph snapshot into the initial latent state.

In the dynamic graph setting considered herein, discrete changes in the graph topology may induce jump discontinuities in the associated vector field with respect to time, so the differential form of DG-FDE may fail to hold pointwise in the classical sense. By contrast, provided that the vector field is measurable and locally bounded in time (e.g., piecewise continuous), the integrability of the weakly singular kernel $(t-s)^{\alpha-1}$ ensures that the corresponding Volterra integral is well defined. We therefore use this integral equation to define continuous mild solutions of DG-FDE, thereby providing a rigorous mathematical framework for accommodating the discontinuities induced by changes in graph topology. We apply the Riemann–Liouville fractional integral of order α to a C^1 solution of equation 3 formally yields the following Volterra integral equation (Podlubny, 1999; Diethelm, 2010). All theoretical results below are established for these mild solutions.

Definition 1 (DG-FDE mild solution on a finite interval)**.** Fix $[t_0, T] \subseteq [t_0, t_N]$ and $\alpha \in (0, 1)$. A DG-FDE mild solution on $[t_0, T]$ is a continuous function $\mathbf{Z} \in C([t_0, T]; \mathbb{R}^{n\times d})$ satisfying, for every $t \in [t_0, T]$,

$$ \mathbf{Z}(t) = \mathbf{Z}_0 + \frac{1}{\Gamma(\alpha)} \int_{t_0}^{t} (t-s)^{\alpha-1} f_\theta\big(\mathbf{Z}(s), \mathbf{A}(s)\big)\, ds, \tag{4} $$

where $\mathbf{Z}_0 = \zeta_\theta(\mathbf{A}_{t_0})$.

The fractional memory kernel associated with equation 4 is $K_\alpha(t-s) = \frac{(t-s)^{\alpha-1}}{\Gamma(\alpha)}$. Using this kernel, the mild formulation can be written as

$$ \mathbf{Z}(t) = \mathbf{Z}_0 + \int_{t_0}^{t} K_\alpha(t-s) f_\theta\big(\mathbf{Z}(s), \mathbf{A}(s)\big)\, ds. \tag{5} $$

Remark 1. The fractional memory kernel $K_\alpha(t-s)$ is the key mechanism that distinguishes DG-FDE from conventional integer-order graph dynamical models. At each past time s, $f_\theta\big(\mathbf{Z}(s), \mathbf{A}(s)\big)$ characterizes the dynamical response under the graph structure active at that time, while $K_\alpha(t-s)$ assigns a decaying weight to this response according to the temporal lag. Equation 5 aggregates

historical responses over $[t_0, t]$, thereby coupling graph propagation with nonlocal temporal memory and jointly capturing short-term bursty interactions and long-term cumulative structural effects. By contrast, integer-order Graph ODEs evolve primarily according to the current state and thus have limited capacity to represent persistent memory effects.

4.2 Well-Posedness of DG-FDE

In the basic static-graph setting, Graph Neural ODE establishes that, when the vector field is Lipschitz continuous with respect to the latent state and satisfies appropriate temporal regularity conditions, the corresponding initial-value problem admits a unique absolutely continuous solution (Poli et al., 2019). GN-CDE interpolates graph snapshots into a piecewise continuously differentiable control path and guarantees existence and uniqueness when the vector field is globally Lipschitz and the interpolated path satisfies the required regularity conditions (Kidger et al., 2020; Qin et al., 2026). In contrast, DG-FDE directly employs the piecewise-constant graph path $\mathbf{A}(\cdot)$, avoiding costly smooth interpolation. Although the resulting composite vector field $t \mapsto f_\theta(\mathbf{Z}, \mathbf{A}(t))$ need not be continuous in time, it remains measurable. Provided that f_θ is uniformly Lipschitz with respect to the latent state, as specified in Assumption 1, Theorem 1 guarantees the existence of a unique continuous mild solution, while Proposition 1 establishes its stability with respect to the initial embedding. On any finite interval $[t_0, T] \subseteq [t_0, t_N]$, the graph path takes only finitely many snapshot values and is therefore measurable and uniformly bounded, with $\mathcal{A}_T = \{\mathbf{A}(t) : t \in [t_0, T]\}$.

Assumption 1 (Finite-horizon vector-field regularity)**.** On every finite interval $[t_0, T]$, the vector field f_θ is jointly measurable and uniformly Lipschitz in the latent state on bounded graph inputs. That is, there exists $L > 0$ such that, for all $\mathbf{A} \in \mathcal{A}_T$ and all $\mathbf{Z}_1, \mathbf{Z}_2 \in \mathbb{R}^{n \times d}$,

$$\|f_\theta(\mathbf{Z}_1, \mathbf{A}) - f_\theta(\mathbf{Z}_2, \mathbf{A})\|_F \leq L \, \|\mathbf{Z}_1 - \mathbf{Z}_2\|_F \, . \tag{6}$$

Theorem 1 (Finite-horizon well-posedness)**.** *Fix a finite interval $[t_0, T] \subseteq [t_0, t_N]$, a fractional order $\alpha \in (0, 1)$, and the piecewise-constant graph path $\mathbf{A}(\cdot)$ defined in section 3. If Assumption 1 holds, then for any initial embedding $\mathbf{Z}_0$, the DG-FDE Volterra equation in equation 4 has a unique mild solution $\mathbf{Z} \in C([t_0, T]; \mathbb{R}^{n \times d})$. The solution is bounded on $[t_0, T]$ and depends continuously on the initial embedding.*

Theorem 1 establishes that, under Assumption 1, any prescribed initial embedding and piecewise-constant graph path determine a unique, continuous, and bounded DG-FDE representation trajectory over every finite time interval. This result shows that abrupt changes in graph structure do not compromise the existence or uniqueness of the solution, while the nonlocal accumulation induced by the fractional kernel does not lead to finite-time blow-up of the node representations. A detailed proof is provided in Appendix A.1.

Proposition 1 (Finite-horizon stability)**.** *Under the conditions of Theorem 1, the DG-FDE solution is stable on finite horizons. In particular, for two mild solutions driven by the same graph path, there exists a finite constant $C > 0$ such that*

$$\|\mathbf{Z}(t) - \widetilde{\mathbf{Z}}(t)\|_F \leq C \|\mathbf{Z}_0 - \widetilde{\mathbf{Z}}_0\|_F, \tag{7}$$

Proposition 1 provides a rigorous robustness guarantee: over any finite time horizon, perturbations in the initial embeddings induce only bounded deviations in the resulting trajectories. Consequently, DG-FDE exhibits robust finite-horizon dynamics, maintaining stable representation trajectories even when the initial node embeddings are subject to noise or perturbations. A detailed proof is provided in Appendix A.2.

4.3 Nonlocal Memory Characterization

Proposition 2 (Restart identity)**.** *For any intermediate time $\tau \in (t_0, T)$ and any $t \in [\tau, T]$, the DG-FDE mild solution satisfies*

$$\mathbf{Z}(t) = \mathbf{Z}(\tau) + \int_\tau^t K_\alpha(t - s) f_\theta(\mathbf{Z}(s), \mathbf{A}(s)) \, ds + \mathbf{R}_{\alpha,\tau}(t), \tag{8}$$

$$\mathbf{R}_{\alpha,\tau}(t) = \int_{t_0}^\tau \left[K_\alpha(t - s) - K_\alpha(\tau - s)\right] f_\theta(\mathbf{Z}(s), \mathbf{A}(s)) \, ds. \tag{9}$$

Proposition 2 gives a history-dependent restart identity. Its derivation is provided in Appendix A.4. For a Graph Neural ODE, once the future graph path is specified, the future evolution can be restarted solely from $\mathbf{Z}(\tau)$. In contrast, the post-τ trajectory of DG-FDE generally also depends on $\mathbf{R}_{\alpha,\tau}(t)$, which retains graph-conditioned vector-field evaluations accumulated over $[t_0, \tau]$. Importantly, this residual is not introduced through an auxiliary or explicitly designed memory module; it arises intrinsically from the nonlocal Volterra fractional integral operator itself. Hence, the history dependence of DG-FDE is built directly into its governing dynamics. In the classical integer-order Caputo case $\alpha = 1$, the kernel becomes $K_1(t-s) = K_1(\tau - s) = 1$, so the residual term vanishes and the standard restart property is recovered.

Proposition 3 (Kernel-weighted graph-interval decomposition)**.** *Let $\mathbf{Z}$ be a mild solution of equation 4, and suppose that $\mathbf{A}(s) = \mathbf{A}_{t_k}$ for almost every $s \in [t_k, t_{k+1})$. If $f_\theta(\mathbf{Z}(s), \mathbf{A}_{t_k}) \approx \mathbf{F}_k$ over each interval $[t_k, t_{k+1} \wedge t]$, then, for every $t \in [t_0, T]$,*

$$\mathbf{Z}(t) \approx \mathbf{Z}_0 + \sum_{k:t_k<t} w_k^\alpha(t)\mathbf{F}_k, \qquad w_k^\alpha(t) := \frac{(t-t_k)^\alpha - \big(t - (t_{k+1} \wedge t)\big)^\alpha}{\Gamma(\alpha+1)}. \tag{10}$$

Proof. Let $a \wedge b := \min\{a, b\}$. Splitting the Volterra integral over the graph intervals and applying the interval-constant approximation gives

$$\begin{aligned} \mathbf{Z}(t) - \mathbf{Z}_0 &= \sum_{k:t_k<t} \int_{t_k}^{t_{k+1}\wedge t} K_\alpha(t-s) f_\theta\big(\mathbf{Z}(s), \mathbf{A}_{t_k}\big)\, ds \\ &\approx \sum_{k:t_k<t} \frac{\mathbf{F}_k}{\Gamma(\alpha)} \int_{t_k}^{t_{k+1}\wedge t} (t-s)^{\alpha-1}\, ds \\ &= \sum_{k:t_k<t} \frac{(t-t_k)^\alpha - \big(t-(t_{k+1}\wedge t)\big)^\alpha}{\Gamma(\alpha+1)} \mathbf{F}_k = \sum_{k:t_k<t} w_k^\alpha(t)\mathbf{F}_k, \end{aligned} \tag{11}$$

□

Proposition 3 makes the central mechanism of DG-FDE explicit by separating graph-conditioned dynamics from temporal memory. The matrix $\mathbf{F}_k$ summarizes the state update generated while $\mathbf{A}_{t_k}$ is active, whereas $w_k^\alpha(t)$ determines how strongly that interval contributes to the representation at time t. For a completed interval, defining its duration as $\Delta_k = t_{k+1} - t_k$ and its temporal lag as $\ell_k(t) = t - t_{k+1}$ gives $w_k^\alpha(t) = \big((\ell_k(t) + \Delta_k)^\alpha - \ell_k(t)^\alpha\big)/\Gamma(\alpha+1)$. For a fixed lag, the weight increases with the snapshot duration; for a fixed duration, it decreases with the lag and retains an algebraically decaying historical influence. The order α controls the shape of this memory profile. Consequently, DG-FDE incorporates irregular observation times directly through explicit, interpretable interval weights while aggregating graph-conditioned dynamics over history. Appendix A.4 bounds the resulting integral approximation error in terms of the within-interval vector-field error.

Remark 2 (Fractional order α)**.** The fractional order α controls the temporal weighting of the memory kernel. As a general framework, DG-FDE supports both fixed and learnable choices of α, and we evaluate both settings in the subsequent experiments. In the fixed setting, α is selected from a predefined candidate set; in the learnable setting, it is optimized jointly with the vector field parameters. We select the final configuration solely according to validation performance and then use it for test evaluation. Because flexible neural vector fields may achieve comparable fits under different orders (Coelho et al., 2024), we interpret α as the memory profile adopted by the model rather than a uniquely identifiable physical property of the data.

4.4 Solving DG-FDE

We solve equation 5 using the fractional-order Adams–Bashforth–Moulton predictor–corrector method (Diethelm et al., 2002; Zimmering et al., 2024). Kernel-weighted accumulation of cached vector-field evaluations captures fractional memory without repeating graph convolutions over past states. Fixed-step integration provides a predictable computational budget, with an adjustable step size to manage costs for longer input windows while preserving temporal resolution and predictive accuracy. Appendix C compares the complexity of DG-FDE with other dynamic graph neural differential equation models. Let $\tau_k = t_0 + kh$, $k = 0, \ldots, K$, denote a uniform internal solver grid

with $h = (T - t_0)/K$. This grid need not coincide with the graph observation times; the solver evaluates the piecewise-constant graph path $\mathbf{A}(\tau_k)$ at each grid point. We write $\mathbf{Z}^k \approx \mathbf{Z}(\tau_k)$ and define $\mathbf{F}^j = f_\theta(\mathbf{Z}^j, \mathbf{A}(\tau_j))$.

At step k, the predictor estimates the next state using all previous vector-field evaluations. The corrector then evaluates $\mathbf{F}_{\mathrm{P}}^k = f_\theta(\mathbf{Z}_{\mathrm{P}}^k, \mathbf{A}(\tau_k))$ and refines this estimate:

$$\mathbf{Z}_{\mathrm{P}}^k = \mathbf{Z}_0 + \frac{h^\alpha}{\Gamma(\alpha+1)} \sum_{j=0}^{k-1} b_{k-j}^{(\alpha)} \mathbf{F}^j. \tag{12}$$

$$\mathbf{Z}^k = \mathbf{Z}_0 + \frac{h^\alpha}{\Gamma(\alpha+2)} \left[\mathbf{F}_{\mathrm{P}}^k + c_k^{(\alpha)} \mathbf{F}^0 + \sum_{j=1}^{k-1} a_{k-j}^{(\alpha)} \mathbf{F}^j \right]. \tag{13}$$

The coefficients $a_m^{(\alpha)}$, $b_m^{(\alpha)}$, and $c_k^{(\alpha)}$ are obtained by integrating the fractional kernel over the solver intervals; their definitions and derivation are provided in Appendix B. After computing $\mathbf{Z}^k$, the solver stores $\mathbf{F}^k = f_\theta(\mathbf{Z}^k, \mathbf{A}(\tau_k))$ for subsequent steps.

4.5 Graph Neural Vector Field

We parameterize f_θ with an L-layer GCN. Given the graph path $\mathbf{A}(t)$, we use the normalized adjacency matrix $\widetilde{\mathbf{A}}(t) = \mathbf{D}(t)^{-1/2}\big(\mathbf{A}(t) + \mathbf{I}\big)\mathbf{D}(t)^{-1/2}$, where $\mathbf{D}(t)$ is the degree matrix of $\mathbf{A}(t) + \mathbf{I}$. Starting from $\mathbf{H}^{(0)}(t) = \mathbf{Z}(t)$, the GCN updates the hidden representations and defines the vector field as

$$\mathbf{H}^{(l)}(t) = \sigma\Big(\widetilde{\mathbf{A}}(t)\mathbf{H}^{(l-1)}(t)\mathbf{W}^{(l-1)}\Big), \quad l = 1, \dots, L, \quad f_\theta\big(\mathbf{Z}(t), \mathbf{A}(t)\big) = \mathbf{H}^{(L)}(t). \tag{14}$$

When node features are available along the dynamic graph sequence, they enter the instantaneous vector field as observed covariates. Let $\mathbf{X}(t) \in \mathbb{R}^{n \times f}$ denote the piecewise-constant feature path aligned with $\mathbf{A}(t)$. The feature-conditioned vector field keeps the same graph convolutional form, except that the first layer is replaced by $\mathbf{H}^{(1)}(t) = \sigma\Big(\widetilde{\mathbf{A}}(t)\mathbf{Z}(t)\mathbf{W}_z^{(0)} + \mathbf{X}(t)\mathbf{W}_x^{(0)}\Big)$, where $\mathbf{W}_z^{(0)} \in \mathbb{R}^{d \times d}$ and $\mathbf{W}_x^{(0)} \in \mathbb{R}^{f \times d}$ are learnable weights for the graph-propagated state and the observed node features. The subsequent layers follow equation 14 for $l = 2, \dots, L$, and the resulting vector field is written as $f_\theta(\mathbf{Z}(t), \mathbf{A}(t), \mathbf{X}(t)) = \mathbf{H}^{(L)}(t)$.

With the Lipschitz activations used in our implementation and bounded, measurable graph and feature inputs, every fixed set of finite-valued parameters induces a vector field satisfying Assumption 1 on finite observation windows. The proof for the feature-conditioned case is provided in Appendix A.3.

5 Experiments

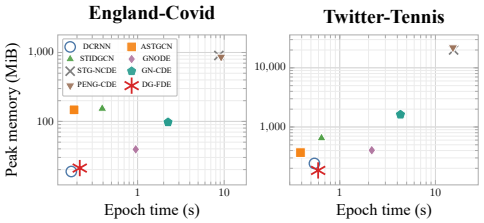
We evaluate DG-FDE on node property prediction and link prediction on graph snapshot sequences. Experiments are conducted on a Linux server equipped with eight NVIDIA GeForce RTX 3090 GPUs (24 GB each), two Intel Xeon Silver 4210 CPUs running at 2.20 GHz, and 256 GB of RAM. Additional details on experiments are provided in Appendix D.

5.1 Node Property Prediction

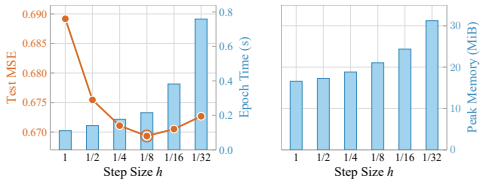
Datasets and Baselines. We evaluate DG-FDE on dynamic node property prediction using the England-Covid and Twitter-Tennis datasets from PyTorch Geometric Temporal (Rozemberczki et al., 2021). The task is to predict future node attributes from the temporal evolution of graph topology and node features. We benchmark DG-FDE against strong dynamic graph neural differential-equation baselines, includeing STG-NCDE (Choi et al., 2022), GN-CDE (Qin et al., 2026), and PENG-CDE (Berndt et al., 2025). We include Graph Neural ODE (GNODE), implemented using the graph-conditioned ODE in Eq.(7) of GN-CDE. We also include recurrent, attention, and interaction based spatiotemporal baselines, namely DCRNN (Li et al., 2018), ASTGCN (Guo et al., 2019),

Table 1: Test MSE on England-Covid and Twitter-Tennis (mean ± standard deviation over 10 random seeds; lower is better). The best and the second-best results are highlighted in red and blue, respectively.

Method	England-Covid	Twitter-Tennis
DCRNN	1.528 ± 0.150	0.231 ± 0.007
ASTGCN	1.488 ± 0.199	0.297 ± 0.025
STIDGCN	0.892 ± 0.143	0.249 ± 0.011
GNODE	0.863 ± 0.206	0.283 ± 0.026
STG-NCDE	3.969 ± 3.564	0.296 ± 0.010
GN-CDE	0.689 ± 0.159	0.293 ± 0.008
PENG-CDE	0.693 ± 0.158	0.233 ± 0.010
DG-FDE (Fixed $\alpha = 1$)	0.820 ± 0.106	0.248 ± 0.007
DG-FDE (Fixed $\alpha < 1$)	$\mathbf{0.669 \pm 0.108}$	$\mathbf{0.198 \pm 0.003}$
DG-FDE (Learned α)	$\mathbf{0.671 \pm 0.105}$	$\mathbf{0.196 \pm 0.004}$



(a) Epoch time and peak memory versus baselines.



(b) Test MSE, epoch time, and peak memory versus solver step size on England-Covid.

Figure 1: Efficiency analysis of DG-FDE.

and STIDGCN (Liu & Zhang, 2024). For the fixed-order configurations, we selected α based on validation-set performance from the candidate set $\{0.1,0.2,\ldots,0.9\}$, yielding $\alpha = 0.2$ for England-Covid and $\alpha = 0.3$ for Twitter-Tennis. For the learnable-order configurations, α is initialized to 0.2 on both datasets and subsequently optimized jointly with the other model parameters.

Finding I: Fractional temporal modeling improves predictive performance. Table 1 shows that DG-FDE achieves the strongest test performance on both datasets and consistently outperforms its integer-order counterpart. The learnable-order variant remains comparable to the best manually selected fixed order, demonstrating that DG-FDE can effectively adapt its temporal memory.

Finding II: DG-FDE achieves a favorable accuracy–efficiency trade-off. Figure 1 shows that DG-FDE combines strong predictive accuracy with low runtime and memory costs compared with other continuous-time models. The step-size analysis further indicates that a moderate step size is sufficient, whereas finer discretization substantially increases computation without providing consistent accuracy gains.

5.2 Link Prediction

Datasets and Baselines. We evaluate DG-FDE on seven public dynamic graph datasets commonly used for temporal link prediction: Enron, Contact, RDS, UCI, Reddit-Title, Reddit-Body, and AS733. We compare DG-FDE with point-process-based Hawkes-GCN, Hawkes-GAT (Qi et al., 2025), GHP (Shang & Sun, 2019), and M2DNE (Lu et al., 2019); recurrent models EvolveGCN-O/H (Pareja et al., 2020), GC-LSTM (Chen et al., 2022), VGRNN (Hajiramezanali et al., 2019), ROLAND (You et al., 2022), and HTGN (Yang et al., 2021); and DySAT (Sankar et al., 2020). We additionally include GCN (Mean), a GNN-only ablation of DG-FDE. It uniformly averages graph snapshots within each input window and processes the resulting aggregated graph using the same GNN architecture employed in DG-FDE's vector field. All baselines share the same next-snapshot prediction target and evaluation protocol. For the fixed-order variant, we set the fractional order to $\alpha = 0.1$ for Contact, $\alpha = 0.4$ for Body, $\alpha = 0.3$ for Enron, $\alpha = 0.5$ for RDS and Title, $\alpha = 0.4$ for UCI, and $\alpha = 0.8$ for AS733. For the learnable-order variant, α was initialized to 0.2.

Finding III: Fractional dynamics consistently improve temporal link prediction. DG-FDE achieves the highest mean MRR@100 across all seven datasets. Both fractional variants outperform the integer-order setting ($\alpha = 1$), supporting the contribution of nonlocal temporal memory to the performance gains. The fractional-order sensitivity analysis in Appendix D further shows that suitable $\alpha < 1$ values improve performance across multiple metrics, with the preferred order varying across datasets.

Table 2: Dynamic link prediction measured by MRR@100 (%). Results are reported as mean ± standard deviation over 3 random seeds; higher is better. The best and second-best results are shown in red and blue, respectively. OOM denotes an out-of-memory result.

Method	Enron	Contact	RDS	UCI	Body	Title	AS733
GCN (Mean)	11.80 ± 0.00	12.22 ± 0.01	12.41 ± 0.00	10.84 ± 0.09	22.18 ± 0.00	36.27 ± 0.00	41.95 ± 0.00
DySAT	7.09 ± 2.22	6.17 ± 0.39	7.61 ± 0.79	19.12 ± 11.54	10.72 ± 2.78	13.72 ± 4.68	23.03 ± 0.54
EvolveGCN	11.58 ± 0.00	10.97 ± 0.01	10.77 ± 0.00	7.34 ± 0.17	18.82 ± 0.05	30.88 ± 0.00	42.06 ± 0.00
ROLAND	16.74 ± 0.81	22.51 ± 0.74	12.79 ± 7.08	18.02 ± 0.90	37.46 ± 0.06	46.85 ± 0.32	37.54 ± 6.55
WinGNN	2.74 ± 0.15	2.03 ± 0.04	2.43 ± 0.39	2.37 ± 0.13	3.01 ± 0.53	3.67 ± 1.51	2.77 ± 0.06
VGRNN	10.38 ± 0.16	12.46 ± 0.85	10.40 ± 0.59	6.96 ± 0.08	17.19 ± 0.14	OOM	42.34 ± 2.23
HTGN	16.24 ± 0.91	10.92 ± 0.19	11.01 ± 1.05	8.93 ± 1.60	11.18 ± 0.92	10.66 ± 1.24	13.55 ± 1.45
GC-LSTM	15.06 ± 0.57	13.82 ± 0.02	20.81 ± 0.15	27.28 ± 0.27	42.08 ± 0.42	54.71 ± 0.11	52.41 ± 0.55
GraphMixer	12.86 ± 0.03	15.13 ± 0.02	14.89 ± 0.03	33.62 ± 0.04	31.79 ± 0.02	38.74 ± 0.02	28.83 ± 0.03
M2DNE	11.87 ± 0.70	8.53 ± 0.10	12.40 ± 0.51	8.86 ± 0.44	5.68 ± 0.11	5.37 ± 0.01	19.43 ± 0.12
GHP	8.36 ± 0.35	9.84 ± 1.40	9.75 ± 2.02	4.15 ± 0.14	8.23 ± 1.97	16.00 ± 2.32	22.15 ± 4.88
Hawkes-GCN	15.68 ± 0.03	17.04 ± 0.17	17.51 ± 0.17	34.35 ± 1.89	34.49 ± 0.36	47.83 ± 0.27	43.83 ± 0.04
Hawkes-GAT	13.03 ± 0.20	16.47 ± 0.04	18.48 ± 0.02	**35.63 ± 1.59**	39.55 ± 0.17	50.75 ± 0.10	47.25 ± 0.29
GNODE	13.38 ± 0.65	21.23 ± 0.29	20.33 ± 0.34	27.30 ± 0.15	36.24 ± 0.73	54.86 ± 0.70	61.40 ± 0.96
STG-NCDE	10.56 ± 1.52	23.40 ± 0.39	20.98 ± 0.75	27.84 ± 0.61	OOM	OOM	OOM
GN-CDE	8.70 ± 0.35	11.91 ± 0.11	11.13 ± 0.22	24.55 ± 0.30	32.29 ± 0.27	42.41 ± 0.09	30.74 ± 0.63
PENG-CDE	8.77 ± 0.39	12.93 ± 0.72	11.21 ± 0.15	26.68 ± 0.70	OOM	OOM	26.85 ± 0.58
DG-FDE (Fixed $\alpha = 1$)	12.46 ± 1.08	29.02 ± 0.57	20.30 ± 0.03	30.28 ± 0.40	38.43 ± 0.39	53.95 ± 0.10	62.83 ± 0.98
DG-FDE (Fixed $\alpha < 1$)	**19.11 ± 0.42**	**32.38 ± 0.51**	**21.60 ± 0.15**	34.74 ± 0.81	**42.29 ± 0.44**	**56.42 ± 0.04**	**73.24 ± 1.26**
DG-FDE (Learned α)	**19.20 ± 0.73**	**30.69 ± 1.13**	**21.40 ± 0.15**	**35.65 ± 0.36**	**42.10 ± 0.31**	**56.11 ± 0.09**	**67.23 ± 0.11**

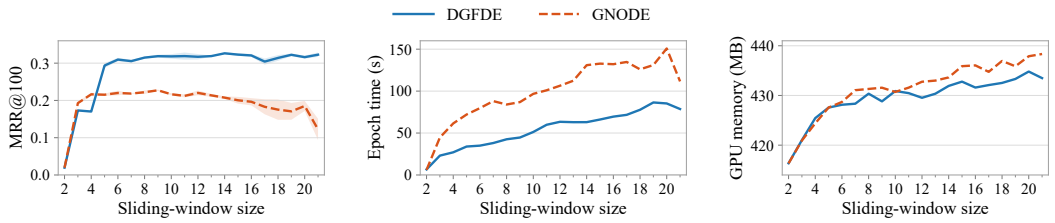


Figure 2: Effect of the sliding-window size on predictive performance and computational efficiency on the Contact dataset. DG-FDE and GNODE are compared in terms of MRR@100, average training time per epoch, and peak GPU memory usage.

Finding IV: DG-FDE provides robust and efficient long-range modeling. As the sliding window grows, DG-FDE maintains stable predictive performance, whereas GNODE gradually deteriorates. Meanwhile, DG-FDE requires less training time with comparable memory growth, demonstrating a favorable accuracy–efficiency trade-off for modeling long temporal contexts.

6 Conclusion

DG-FDE brings graph propagation and temporal memory into a unified continuous-time process, allowing historical interactions to shape node representations across multiple temporal scales. Its Volterra formulation reveals how these historical responses accumulate, making the contributions of different graph intervals explicit. This approach achieved the best performance across the evaluated node property and dynamic link prediction benchmarks, consistently outperforming its integer-order counterpart. The lower per-epoch runtime relative to the tested integer-order dynamic graph differential models further demonstrates its ability to combine predictive accuracy with computational efficiency.

Limitations and future work. Our formulation and theoretical analysis focus on a shared scalar fractional order that remains constant over time, whether prescribed or learned. A natural extension is to introduce graph-conditioned orders, such as $\alpha(t) = \alpha_\phi(\mathcal{G}(t))$, allowing temporal memory to adapt to evolving graph structures. Establishing well-posedness and stability for the resulting variable-order dynamics is an interesting theoretical direction.

AI USE STATEMENT

We used generative AI tools to assist with searching for related work, editing software code, and improving the readability of the manuscript. All AI-assisted code was reviewed, verified, and tested by the authors for correctness. All AI-assisted revisions to the manuscript were carefully reviewed by the authors. We take full responsibility for the final content of this work.

REPRODUCIBILITY STATEMENT

To facilitate reproducibility, we provide the complete source code as downloadable supplementary material. The assumptions underlying our theoretical results are stated explicitly in the main text, while complete proofs of the main theoretical claims are provided in the appendix. The appendix also includes download links for all datasets used in our experiments. In addition, the scripts for data preprocessing are included alongside the implementation in the supplementary material.

REFERENCES

Ferenc Béres, Róbert Pálovics, Anna Oláh, and András A. Benczúr. Temporal walk based centrality metric for graph streams. *Applied Network Science*, 3(1):32, 2018. doi: 10.1007/s41109-018-0080-5.

Torben Berndt, Benjamin Walker, Tiexin Qin, Jan Stühmer, and Andrey Kormilitzin. Permutation equivariant neural controlled differential equations for dynamic graph representation learning. In *Advances in Neural Information Processing Systems*, volume 38, pp. 108931–108966, 2025. doi: 10.52202/085713-3284.

Jinyin Chen, Xueke Wang, and Xuanheng Xu. GC-LSTM: Graph convolution embedded LSTM for dynamic network link prediction. *Applied Intelligence*, 52:7513–7528, 2022. doi: 10.1007/s10489-021-02518-9.

Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1724–1734, Doha, Qatar, 2014. Association for Computational Linguistics. doi: 10.3115/v1/D14-1179. URL https://aclanthology.org/D14-1179/.

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*, pp. 6367–6374, 2022. doi: 10.1609/aaai.v36i6.20587.

Cecília Coelho, M. Fernanda P. Costa, and Luís L. Ferrás. Neural fractional differential equations: Optimising the order of the fractional derivative. *Fractal and Fractional*, 8(9):529, 2024. doi: 10.3390/fractalfract8090529.

Cecília Coelho, M. Fernanda P. Costa, and Luís L. Ferrás. Neural fractional differential equations. *Applied Mathematical Modelling*, 144:116060, 2025. doi: 10.1016/j.apm.2025.116060. URL https://doi.org/10.1016/j.apm.2025.116060.

Wenjun Cui, Qiyu Kang, Xuhao Li, Kai Zhao, Wee Peng Tay, Weihua Deng, and Yidong Li. Neural variable-order fractional differential equation networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 16109–16117, 2025. doi: 10.1609/aaai.v39i15.33769. URL https://ojs.aaai.org/index.php/AAAI/article/view/33769.

Kai Diethelm. *The Analysis of Fractional Differential Equations: An Application-Oriented Exposition Using Differential Operators of Caputo Type*, volume 2004 of *Lecture Notes in Mathematics*. Springer, Berlin, Heidelberg, 2010. doi: 10.1007/978-3-642-14574-2.

Kai Diethelm, Neville J. Ford, and Alan D. Freed. A predictor-corrector approach for the numerical solution of fractional differential equations. *Nonlinear Dynamics*, 29(1–4):3–22, 2002. doi: 10.1023/A:1016592219341.

Shengnan Guo, Youfang Lin, Ning Feng, Chao Song, and Huaiyu Wan. Attention based spatial-temporal graph convolutional networks for traffic flow forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 922–929, 2019. doi: 10.1609/aaai.v33i01.3301922.

Ehsan Hajiramezanali, Arman Hasanzadeh, Krishna R. Narayanan, Nick Duffield, Mingyuan Zhou, and Xiaoning Qian. Variational graph recurrent neural networks. In *Advances in Neural Information Processing Systems*, volume 32, pp. 10700–10710, 2019. URL https://papers.nips.cc/paper/2019/hash/a6b8deb7798e7532ade2a8934477d3ce-Abstract.html.

Qiyu Kang, Kai Zhao, Qinxu Ding, Feng Ji, Xuhao Li, Wenfei Liang, Yang Song, and Wee Peng Tay. Unleashing the potential of fractional calculus in graph neural networks with FROND. In *International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/377235d5cee7b104501407c0e5066c92-Abstract-Conference.html.

Qiyu Kang, Wenjun Cui, Xuhao Li, Yuxin Ma, Xueyang Fu, Wee Peng Tay, Yidong Li, and Zheng-Jun Zha. Neural fractional attention differential equations. In *Advances in Neural Information Processing Systems*, volume 38, pp. 37716–37749, 2025. doi: 10.52202/085713-1125. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/3040bf93152e90d5719379818a8664b4-Abstract-Conference.html.

Seyed Mehran Kazemi, Rishab Goel, Kshitij Jain, Ivan Kobyzev, Akshay Sethi, Peter Forsyth, and Pascal Poupart. Representation learning for dynamic graphs: A survey. *Journal of Machine Learning Research*, 21(70):1–73, 2020. URL https://jmlr.org/papers/v21/19-447.html.

Shima Khoshraftar and Aijun An. A survey on graph representation learning methods. *ACM Transactions on Intelligent Systems and Technology*, 15(1):1–55, 2024. doi: 10.1145/3633518.

Patrick Kidger, James Morrill, James Foster, and Terry Lyons. Neural controlled differential equations for irregular time series. In *Advances in Neural Information Processing Systems*, volume 33, pp. 6696–6707, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/4a5876b450b45371f6cfe5047ac8cd45-Abstract.html.

Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=SJU4ayYgl.

Srijan Kumar, Xikun Zhang, and Jure Leskovec. Predicting dynamic embedding trajectory in temporal interaction networks. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1269–1278. ACM, 2019. doi: 10.1145/3292500.3330895.

Yaguang Li, Rose Yu, Cyrus Shahabi, and Yan Liu. Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=SJiHXGWAZ.

Aoyu Liu and Yaying Zhang. Spatial-temporal dynamic graph convolutional network with interactive learning for traffic forecasting. *IEEE Transactions on Intelligent Transportation Systems*, 25 (7):7645–7660, 2024. doi: 10.1109/TITS.2024.3362145.

Antonio Longa, Veronica Lachi, Gabriele Santin, Monica Bianchini, Bruno Lepri, Pietro Liò, Franco Scarselli, and Andrea Passerini. Graph neural networks for temporal graphs: State of the art, open challenges, and opportunities. *Transactions on Machine Learning Research*, 2023. ISSN 2835-8856. URL https://openreview.net/forum?id=pHCdMat0gI.

Yuanfu Lu, Xiao Wang, Chuan Shi, Philip S. Yu, and Yanfang Ye. Temporal network embedding with micro- and macro-dynamics. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, pp. 469–478, 2019. doi: 10.1145/3357384.3357943.

Sohir Maskey, Raffaele Paolino, Aras Bacho, and Gitta Kutyniok. A fractional graph laplacian approach to oversmoothing. In *Advances in Neural Information Processing Systems*, volume 36, pp. 13022–13063, 2023. doi: 10.52202/075280-0571. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/2a514213ba899f2911723a38be8d4096-Abstract-Conference.html.

George Panagopoulos, Giannis Nikolentzos, and Michalis Vazirgiannis. Transfer graph neural networks for pandemic forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 4838–4845, 2021. doi: 10.1609/aaai.v35i6.16616.

Aldo Pareja, Giacomo Domeniconi, Jie Chen, Tengfei Ma, Toyotaro Suzumura, Hiroki Kanezashi, Tim Kaler, Tao B. Schardl, and Charles E. Leiserson. EvolveGCN: Evolving graph convolutional networks for dynamic graphs. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 5363–5370, 2020. doi: 10.1609/aaai.v34i04.5984.

Igor Podlubny. *Fractional Differential Equations*, volume 198 of *Mathematics in Science and Engineering*. Academic Press, San Diego, CA, USA, 1999. ISBN 978-0-12-558840-9.

Michael Poli, Stefano Massaroli, Junyoung Park, Atsushi Yamashita, Hajime Asama, and Jinkyoo Park. Graph neural ordinary differential equations. *arXiv preprint arXiv:1911.07532*, 2019. URL https://arxiv.org/abs/1911.07532.

QingGuo Qi, Hongyang Chen, Minhao Cheng, and Han Liu. Input snapshots fusion for scalable discrete-time dynamic graph neural networks. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.1*, pp. 1138–1149, 2025. doi: 10.1145/3690624.3709316.

Tiexin Qin, Benjamin Walker, Terry Lyons, Hong Yan, and Haoliang Li. Learning dynamic graph embeddings with neural controlled differential equations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 48(2):2096–2103, 2026. doi: 10.1109/TPAMI.2025.3617660.

Emanuele Rossi, Ben Chamberlain, Fabrizio Frasca, Davide Eynard, Federico Monti, and Michael Bronstein. Temporal graph networks for deep learning on dynamic graphs. *arXiv preprint arXiv:2006.10637*, 2020. URL https://arxiv.org/abs/2006.10637.

Benedek Rozemberczki, Paul Scherer, Yixuan He, George Panagopoulos, Alexander Riedel, Maria Astefanoaei, Oliver Kiss, Ferenc Béres, Guzmán López, Nicolas Collignon, and Rik Sarkar. PyTorch Geometric Temporal: Spatiotemporal signal processing with neural machine learning models. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, pp. 4564–4573, 2021. doi: 10.1145/3459637.3482014.

Aravind Sankar, Yanhong Wu, Liang Gou, Wei Zhang, and Hao Yang. DySAT: Deep neural representation learning on dynamic graphs via self-attention networks. In *Proceedings of the 13th International Conference on Web Search and Data Mining*, pp. 519–527. ACM, 2020. doi: 10.1145/3336191.3371845.

Jin Shang and Mingxuan Sun. Geometric Hawkes processes with graph convolutional recurrent neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 4878–4885, 2019. doi: 10.1609/aaai.v33i01.33014878.

Joakim Skarding, Bogdan Gabrys, and Katarzyna Musial. Foundations and modeling of dynamic networks using dynamic graph neural networks: A survey. *IEEE Access*, 9:79143–79168, 2021. doi: 10.1109/ACCESS.2021.3082932.

Rakshit S. Trivedi, Mehrdad Farajtabar, Prasenjeet Biswal, and Hongyuan Zha. Dyrep: Learning representations over dynamic graphs. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=HyePrhR5KX.

Jeffrey R. L. Webb. Initial value problems for Caputo fractional equations with singular nonlinearities. *Electronic Journal of Differential Equations*, 2019(117):1–32, 2019. URL https://ejde.math.txstate.edu/Volumes/2019/117/abstr.html.

Menglin Yang, Min Zhou, Marcus Kalander, Zengfeng Huang, and Irwin King. Discrete-time temporal network embedding via implicit hierarchical learning in hyperbolic space. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 1975–1985, 2021. doi: 10.1145/3447548.3467422.

Haiping Ye, Jianming Gao, and Yongsheng Ding. A generalized Gronwall inequality and its application to a fractional differential equation. *Journal of Mathematical Analysis and Applications*, 328(2):1075–1081, 2007. doi: 10.1016/j.jmaa.2006.05.061.

Jiaxuan You, Tianyu Du, and Jure Leskovec. ROLAND: Graph learning framework for dynamic graphs. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 2358–2366, 2022. doi: 10.1145/3534678.3539300.

Yifan Zhu, Fangpeng Cong, Dan Zhang, Wenwen Gong, Qika Lin, Wenzheng Feng, Yuxiao Dong, and Jie Tang. WinGNN: Dynamic graph neural networks with random gradient aggregation window. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 3650–3662, 2023. doi: 10.1145/3580305.3599551.

Bernd Zimmering, Cecılia Coelho, and Oliver Niggemann. Optimising neural fractional differential equations for performance and efficiency. *Proceedings of Machine Learning Research*, 255(1): 23, 2024.

APPENDIX

A PROOFS

A.1 WELL-POSEDNESS OF DG-FDE

Throughout Sections A.1–A.2, the model parameters θ, the fractional order $\alpha \in (0, 1)$, and the graph path are fixed. We prove the statements for the continuous mild solution in Definition 1. The constants L and B_0 below refer to this fixed vector field and finite observation interval.

A.1.1 AUXILIARY VOLTERRA REGULARITY RESULT

Lemma 1 (Bounded forcing and continuity of the fractional integral)**.** *Under Assumption 1, let* $\mathcal{A}_T = \{\mathbf{A}(t) : t \in [t_0, T]\}$. *Then*

$$B_0 := \max_{\mathbf{A}\in\mathcal{A}_T} \|f_\theta(\mathbf{0}, \mathbf{A})\|_F < \infty, \qquad \|f_\theta(\mathbf{Z}, \mathbf{A})\|_F \le B_0 + L\|\mathbf{Z}\|_F. \tag{15}$$

For a bounded measurable trajectory $\mathbf{Z}$*, the function* $g(t) = f_\theta(\mathbf{Z}(t), \mathbf{A}(t))$ *is measurable and bounded. Writing* $J^\alpha = J^\alpha_{t_0}$ *within this appendix, we have*

$$\|J^\alpha g\|_\infty \le \frac{(T - t_0)^\alpha}{\Gamma(\alpha + 1)}\|g\|_\infty, \qquad J^\alpha g \in C([t_0, T]; \mathbb{R}^{n\times d}). \tag{16}$$

Here $\|g\|_\infty$ *is the essential supremum of its Frobenius norm.*

Proof. The graph path takes finitely many values, and each value of f_θ is a finite matrix, so B_0 is finite. The second inequality in equation 15 follows by adding and subtracting $f_\theta(\mathbf{0}, \mathbf{A})$ and applying the Lipschitz condition. For each fixed snapshot, $f_\theta(\cdot, \mathbf{A})$ is continuous; composing it with $\mathbf{Z}(t)$ on the finitely many graph intervals makes g measurable. The same inequality gives $\|g\|_\infty \le B_0 + L\|\mathbf{Z}\|_\infty$.

For every $t \in [t_0, T]$, direct kernel integration yields

$$\|(J^\alpha g)(t)\|_F \le \frac{\|g\|_\infty}{\Gamma(\alpha)} \int_{t_0}^{t} (t - s)^{\alpha-1}\, ds = \frac{\|g\|_\infty (t - t_0)^\alpha}{\Gamma(\alpha + 1)}. \tag{17}$$

Thus the integral exists at every time, including $t = t_0$, where it is zero. To establish continuity across graph jumps, take $t_0 \le u < t \le T$ and split the integral difference:

$$\begin{aligned}(J^\alpha g)(t) - (J^\alpha g)(u) = {} & \frac{1}{\Gamma(\alpha)} \int_u^t (t - s)^{\alpha-1} g(s)\, ds \\ & + \frac{1}{\Gamma(\alpha)} \int_{t_0}^{u} [(t - s)^{\alpha-1} - (u - s)^{\alpha-1}] g(s)\, ds.\end{aligned} \tag{18}$$

The first term has norm at most $\|g\|_\infty (t - u)^\alpha / \Gamma(\alpha + 1)$. Since $r^{\alpha-1}$ decreases for $r > 0$, the absolute kernel difference in the second term satisfies

$$\begin{aligned}\int_{t_0}^{u} |(t - s)^{\alpha-1} - (u - s)^{\alpha-1}|\, ds &= \frac{(u - t_0)^\alpha - (t - t_0)^\alpha + (t - u)^\alpha}{\alpha} \\ &\le \frac{(t - u)^\alpha}{\alpha}.\end{aligned} \tag{19}$$

The last inequality uses $(u - t_0)^\alpha \le (t - t_0)^\alpha$. Combining the two terms, and exchanging t and u when necessary, gives

$$\|(J^\alpha g)(t) - (J^\alpha g)(u)\|_F \le \frac{2\|g\|_\infty}{\Gamma(\alpha + 1)} |t - u|^\alpha. \tag{20}$$

This proves continuity on the whole closed interval. It is the bounded-forcing case of the fractional-integral regularity result in Webb (2019, Proposition 3.2(3)). □

A.1.2 Proof of Theorem 1

Proof. Work in the Banach space $\mathcal{C} = C([t_0, T]; \mathbb{R}^{n\times d})$ and define

$$(\mathcal{T}\mathbf{Z})(t) = \mathbf{Z}_0 + \frac{1}{\Gamma(\alpha)} \int_{t_0}^{t} (t-s)^{\alpha-1} f_\theta(\mathbf{Z}(s), \mathbf{A}(s))\, ds. \tag{21}$$

Each continuous candidate is bounded on $[t_0, T]$, so Lemma 1 shows that $\mathcal{T}$ maps $\mathcal{C}$ into itself and $(\mathcal{T}\mathbf{Z})(t_0) = \mathbf{Z}_0$. For $\beta > 0$, equip $\mathcal{C}$ with the weighted norm

$$\|\mathbf{Z}\|_\beta = \max_{t\in[t_0,T]} e^{-\beta(t-t_0)} \|\mathbf{Z}(t)\|_F, \qquad e^{-\beta(T-t_0)}\|\mathbf{Z}\|_\infty \le \|\mathbf{Z}\|_\beta \le \|\mathbf{Z}\|_\infty. \tag{22}$$

The displayed norm equivalence preserves completeness. For any $\mathbf{Z}, \mathbf{Y} \in \mathcal{C}$, Assumption 1 and $\|\mathbf{Z}(s) - \mathbf{Y}(s)\|_F \le e^{\beta(s-t_0)}\|\mathbf{Z} - \mathbf{Y}\|_\beta$ imply

$$\begin{aligned} e^{-\beta(t-t_0)}\|(\mathcal{T}\mathbf{Z})(t) - (\mathcal{T}\mathbf{Y})(t)\|_F &\le \frac{L}{\Gamma(\alpha)} \int_{t_0}^{t} e^{-\beta(t-t_0)}(t-s)^{\alpha-1}\|\mathbf{Z}(s) - \mathbf{Y}(s)\|_F\, ds \\ &\le \frac{L\|\mathbf{Z} - \mathbf{Y}\|_\beta}{\Gamma(\alpha)} \int_{t_0}^{t} e^{-\beta(t-s)}(t-s)^{\alpha-1}\, ds. \end{aligned} \tag{23}$$

Substitute $r = t - s$ and use the Gamma integral:

$$\int_0^{t-t_0} e^{-\beta r} r^{\alpha-1}\, dr \le \int_0^{\infty} e^{-\beta r} r^{\alpha-1}\, dr = \Gamma(\alpha)\beta^{-\alpha}. \tag{24}$$

Taking the maximum over t in equation 23 gives

$$\|\mathcal{T}\mathbf{Z} - \mathcal{T}\mathbf{Y}\|_\beta \le L\beta^{-\alpha}\|\mathbf{Z} - \mathbf{Y}\|_\beta. \tag{25}$$

Choose $\beta > L^{1/\alpha}$. Banach's fixed-point theorem then gives a unique $\mathbf{Z} \in \mathcal{C}$ satisfying $\mathbf{Z} = \mathcal{T}\mathbf{Z}$ at every time. This is exactly the mild solution in equation 4. Conversely, every continuous mild solution is a fixed point, so uniqueness holds in the stated solution class. Boundedness follows from continuity on $[t_0, T]$. The initial-state dependence asserted in the theorem follows from the estimate in Section A.2, whose proof uses only the integral equation and Assumption 1. □

The argument establishes continuous Volterra mild solutions, as specified in Definition 1. Relating these solutions to a classical Caputo derivative requires the corresponding derivative domain; continuity alone does not give C^1 regularity (Webb, 2019).

A.2 Finite-Horizon Stability

Proof of Proposition 1. Let $\mathbf{Z}$ and $\widetilde{\mathbf{Z}}$ be mild solutions for the same graph path, fixed parameters, and fractional order, with initial states $\mathbf{Z}_0$ and $\widetilde{\mathbf{Z}}_0$. Define $u(t) = \|\mathbf{Z}(t) - \widetilde{\mathbf{Z}}(t)\|_F$ and $\delta = \|\mathbf{Z}_0 - \widetilde{\mathbf{Z}}_0\|_F$. Subtracting their integral equations gives

$$\begin{aligned} \mathbf{Z}(t) - \widetilde{\mathbf{Z}}(t) &= \mathbf{Z}_0 - \widetilde{\mathbf{Z}}_0 \\ &\quad + \frac{1}{\Gamma(\alpha)} \int_{t_0}^{t} (t-s)^{\alpha-1}[f_\theta(\mathbf{Z}(s), \mathbf{A}(s)) - f_\theta(\widetilde{\mathbf{Z}}(s), \mathbf{A}(s))]\, ds. \end{aligned} \tag{26}$$

Taking norms, using the nonnegative kernel, and applying the uniform Lipschitz condition yields

$$u(t) \le \delta + \frac{L}{\Gamma(\alpha)} \int_{t_0}^{t} (t-s)^{\alpha-1} u(s)\, ds. \tag{27}$$

Here u is continuous and nonnegative. The fractional Grönwall inequality (Ye et al., 2007) therefore applies and gives

$$u(t) \le \delta E_\alpha(L(t-t_0)^\alpha), \qquad E_\alpha(x) = \sum_{j=0}^{\infty} \frac{x^j}{\Gamma(\alpha j + 1)}. \tag{28}$$

The series converges to a finite value for every finite x and has nonnegative coefficients, so E_α is nondecreasing on $[0, \infty)$. Consequently the constant in Proposition 1 can be taken as

$$C = E_\alpha(L(T-t_0)^\alpha), \qquad \max_{t\in[t_0,T]} \|\mathbf{Z}(t) - \widetilde{\mathbf{Z}}(t)\|_F \le C\|\mathbf{Z}_0 - \widetilde{\mathbf{Z}}_0\|_F. \tag{29}$$

This proves uniform continuous dependence on the initial embedding. □

A.3 GCN-Parameterized Vector-Field Regularity

Proposition 4 (GCN regularity)**.** *Let the graph path be as in Section 3, the weights be fixed and finite, and σ be an entrywise L_σ-Lipschitz activation. Define*

$$M_{\widetilde{A}} := \max_{t\in[t_0,T]} \|\widetilde{\mathbf{A}}(t)\|_2, \qquad L^\star := (L_\sigma M_{\widetilde{A}})^L \prod_{l=1}^{L} \|\mathbf{W}^{(l-1)}\|_2. \tag{30}$$

Then the graph-only GCN in equation 14 satisfies

$$\|f_\theta(\mathbf{Z}_1, \mathbf{A}(t)) - f_\theta(\mathbf{Z}_2, \mathbf{A}(t))\|_F \le L^\star \|\mathbf{Z}_1 - \mathbf{Z}_2\|_F. \tag{31}$$

Consequently, it satisfies Assumption 1 with any positive Lipschitz constant not smaller than $L^\star$.

Proof. The adjacency entries are nonnegative and self-loops are added, so every diagonal entry of $\mathbf{D}(t)$ is at least one. Thus each normalized adjacency is a finite matrix. The graph path has finitely many values, which makes $M_{\widetilde{A}}$ finite without requiring a graph-symmetry assumption.

To display the dependence on the input state while retaining the notation from Section 4.5, write $\mathbf{H}^{(l)}(t; \mathbf{Z}_i)$ for the hidden state generated by $\mathbf{Z}_i$, where $\mathbf{H}^{(0)}(t; \mathbf{Z}_i) = \mathbf{Z}_i$. Entrywise Lipschitz continuity gives the same constant for the Frobenius norm. Hence, for $l = 1, \dots, L$,

$$\begin{aligned} \|\mathbf{H}^{(l)}(t; \mathbf{Z}_1) - \mathbf{H}^{(l)}(t; \mathbf{Z}_2)\|_F &\le L_\sigma \|\widetilde{\mathbf{A}}(t)(\mathbf{H}^{(l-1)}(t; \mathbf{Z}_1) - \mathbf{H}^{(l-1)}(t; \mathbf{Z}_2))\mathbf{W}^{(l-1)}\|_F \\ &\le L_\sigma M_{\widetilde{A}} \|\mathbf{W}^{(l-1)}\|_2 \|\mathbf{H}^{(l-1)}(t; \mathbf{Z}_1) - \mathbf{H}^{(l-1)}(t; \mathbf{Z}_2)\|_F. \end{aligned} \tag{32}$$

Iterating this estimate through all L layers and using $f_\theta(\mathbf{Z}_i, \mathbf{A}(t)) = \mathbf{H}^{(L)}(t; \mathbf{Z}_i)$ proves equation 31. The finite network is continuous in the state for each snapshot; over the finite graph-input set this also gives joint measurability. Finally, $B_0 = \max_{\mathbf{A}\in\mathcal{A}_T} \|f_\theta(\mathbf{0}, \mathbf{A})\|_F < \infty$ by finiteness of that set. These are the properties used in Lemma 1 and the well-posedness proof. □

Feature-conditioned extension. When the first layer also uses node features, their time dependence must be included in the field being analyzed. With fixed input paths $\mathbf{A}(t)$ and $\mathbf{X}(t)$, the first layer is

$$\mathbf{H}^{(1)}(t) = \sigma\Big(\widetilde{\mathbf{A}}(t)\mathbf{Z}\mathbf{W}_z^{(0)} + \mathbf{X}(t)\mathbf{W}_x^{(0)}\Big). \tag{33}$$

Assume additionally that $\mathbf{X}$ is measurable and bounded. For two states at the same time and with the same features, the additive feature term cancels, so the first-layer estimate and its iteration above give

$$\begin{aligned} &\|f_\theta(\mathbf{Z}_1, \mathbf{A}(t), \mathbf{X}(t)) - f_\theta(\mathbf{Z}_2, \mathbf{A}(t), \mathbf{X}(t))\|_F \\ &\le (L_\sigma M_{\widetilde{A}})^L \|\mathbf{W}_z^{(0)}\|_2 \prod_{l=2}^{L} \|\mathbf{W}^{(l-1)}\|_2 \|\mathbf{Z}_1 - \mathbf{Z}_2\|_F. \end{aligned} \tag{34}$$

The closure of the bounded feature range is compact in its finite-dimensional matrix space. For each of the finitely many graph snapshots, the network output at $\mathbf{Z} = \mathbf{0}$ is continuous in these features; hence $\sup_t \|f_\theta(\mathbf{0}, \mathbf{A}(t), \mathbf{X}(t))\|_F < \infty$. Measurable input paths and continuous layer maps also make the feature-conditioned vector field jointly measurable in $(t, \mathbf{Z})$. These facts provide the same bounded-forcing and uniform state Lipschitz estimates used above. Thus the existence and uniqueness argument applies to this feature-conditioned Volterra equation, and the stability estimate applies when both solutions use the same graph *and feature* paths. This extension explicitly requires the feature conditions stated here.

A.4 Details for the Memory Identities

Restart identity (Proposition 2). Let $g(s) = f_\theta(\mathbf{Z}(s), \mathbf{A}(s))$. By Lemma 1, g is bounded, so the kernel integrals below are absolutely convergent. Subtract equation 5 at τ from the same equation at $t \ge \tau$, and split the latter integral at τ:

$$\mathbf{Z}(t) - \mathbf{Z}(\tau) = \int_\tau^t K_\alpha(t-s) g(s)\, ds + \int_{t_0}^{\tau} [K_\alpha(t-s) - K_\alpha(\tau-s)] g(s)\, ds. \tag{35}$$

The second term is precisely $\mathbf{R}_{\alpha,\tau}(t)$ in equation 9, proving equation 8. Values of the singular kernel at isolated integration endpoints do not affect the integrals. At $t = \tau$ both terms vanish; for the boundary case $\alpha = 1$, $K_1 = 1$ and the history correction vanishes as stated.

Interval approximation (Proposition 3). The symbol $\approx$ in Proposition 3 represents a piecewise-constant approximation of the vector field along the trajectory. We quantify this integral approximation as follows. Fix $t \in (t_0, T]$ and let $I_k(t) = [t_k, \min\{t_{k+1}, t\}]$ for $k \in \{0, \dots, N-1\}$ with $t_k < t$. Choose the interval values $\mathbf{F}_k$ from that proposition, and define

$$\varepsilon_k(t) = \operatorname*{ess\,sup}_{s \in I_k(t)} \|g(s) - \mathbf{F}_k\|_F, \qquad e(t) = \mathbf{Z}(t) - \mathbf{Z}_0 - \sum_{k:t_k<t} w_k^\alpha(t)\mathbf{F}_k. \tag{36}$$

Since $w_k^\alpha(t) = \int_{I_k(t)} K_\alpha(t-s)\,ds \geq 0$, splitting the Volterra integral over these intervals gives the exact error identity and bound

$$e(t) = \sum_{k:t_k<t} \int_{I_k(t)} K_\alpha(t-s)[g(s) - \mathbf{F}_k]\,ds, \qquad \|e(t)\|_F \leq \sum_{k:t_k<t} w_k^\alpha(t)\varepsilon_k(t). \tag{37}$$

If $\varepsilon_k(t) \leq \varepsilon$ on all contributing intervals, their weights sum to $(t-t_0)^\alpha/\Gamma(\alpha+1)$, and therefore

$$\|e(t)\|_F \leq \frac{\varepsilon(t-t_0)^\alpha}{\Gamma(\alpha+1)} \leq \frac{\varepsilon(T-t_0)^\alpha}{\Gamma(\alpha+1)}. \tag{38}$$

At $t = t_0$ the sum and error are zero. If the field equals $\mathbf{F}_k$ almost everywhere on each interval, the error vanishes and equation 10 becomes an equality. A piecewise-constant graph alone does not imply this condition, since $\mathbf{Z}(s)$ can vary within each interval.

B Fractional Solver Details

Coefficients. For $m \geq 1$ and $k \geq 1$, the coefficients used in equation 12–equation 13 are

$$\begin{aligned} b_m^{(\alpha)} &= m^\alpha - (m-1)^\alpha, \\ a_m^{(\alpha)} &= (m+1)^{\alpha+1} - 2m^{\alpha+1} + (m-1)^{\alpha+1}, \\ c_k^{(\alpha)} &= (k-1)^{\alpha+1} - (k-1-\alpha)k^\alpha. \end{aligned} \tag{39}$$

These coefficients result from integrating the constant and linear temporal interpolants used by the predictor and corrector against the fractional kernel.

Fractional-order setting. In the main experiments, α is selected using the validation set and remains fixed during training. Only in the learnable-order ablation do we optimize an unconstrained parameter α_{raw} through

$$\alpha = \varepsilon + (1-2\varepsilon)\,\text{sigmoid}(\alpha_{\text{raw}}), \qquad \varepsilon = 10^{-4}. \tag{40}$$

The solver coefficients are constructed from α inside the forward computation, allowing automatic differentiation through the solver in this ablation.

C Computational Complexity.

Let n denote the number of nodes, d the hidden dimension, and W the number of input snapshots per window. For a common comparison, we assume explicit fixed-step integration over the same time interval with K integration steps and a constant number of vector-field evaluations per step. Network depths and input feature dimensions are treated as constants, and all methods use dense graph operations. We consider the additive-fusion GN-CDE implementation and the STG-NCDE configuration with two Chebyshev support terms. Table 3 separates interpolation construction from numerical integration.

For DG-FDE, each vector-field evaluation requires a graph convolution, while the kth predictor–corrector step aggregates k previously cached vector-field evaluations. Summing these contributions over the window yields the additional $O(K^2nd)$ history-aggregation cost and an $O(Knd)$ history buffer. Importantly, historical graph convolutions are not recomputed during this aggregation.

Table 3: Per-window forward time complexity under a common fixed-step integration setting. Interpolation coefficients are constructed once per window and may be cached across training epochs. The total cost is the sum of the last two columns.

Method	Interpolation construction	Numerical integration
GNODE (Qin et al., 2026)	–	$O\big(K(n^2d + nd^2)\big)$
GN-CDE (Qin et al., 2026)	$O(Wn^2)$	$O\big(K(n^2d + nd^2)\big)$
PENG-CDE (Berndt et al., 2025)	$O(Wn^2)$	$O\big(K(n^2d + nd^2)\big)$
STG-NCDE (Choi et al., 2022)	$O(Wn)$	$O\big(K(n^2d + nd^3)\big)$
DG-FDE	–	$O\big(K(n^2d + nd^2) + K^2nd\big)$

Practical efficiency. The dense implementations of GN-CDE and PENG-CDE interpolate the evolving adjacency matrices, requiring $O(Wn^2)$ preprocessing and coefficient storage. This overhead becomes substantial as the number of nodes increases; caching avoids repeated coefficient construction but does not remove their storage cost. STG-NCDE instead interpolates node features and constructs an adaptive graph. DG-FDE directly uses piecewise-constant graph observations and therefore avoids constructing differentiable control paths.

Actual runtime also depends on the numerical solver. For an adaptive GNODE solver, the factor K in Table 3 must be replaced by its actual number of vector-field evaluations, which includes multistage evaluations and rejected steps. When directly differentiating through the solver, retained intermediate activations can further increase training memory, although the precise cost depends on the differentiation and checkpointing strategy. In contrast, our fixed-step fractional predictor–corrector scheme performs one correction per step and has a predictable evaluation budget. Over the evaluated windows, the additional history aggregation can be outweighed by savings in vector-field evaluations and solver-control overhead. Consistent with this trade-off, our experiments show lower training time than the evaluated graph ODE/CDE baselines while maintaining strong predictive performance (Figure 1).

Our theoretical analysis establishes finite-horizon stability with respect to perturbations in the initial embedding. The practical effectiveness of the numerical scheme is supported separately by the step-size ablation: on England-Covid, a moderate step size provides a favorable accuracy–efficiency trade-off, whereas finer discretization increases computation without improving predictive accuracy. For longer input windows, the step size can be adjusted to control the integration budget. This flexibility limits practical computational costs, while the direct history aggregation retains its quadratic dependence on the number of integration steps.

D Experimental Details

D.1 Implementation Details for Node Property Prediction

England-Covid. England-Covid contains daily mobility networks and COVID-19 case counts for 129 English NUTS-3 regions from March 13 to May 12, 2020 (Panagopoulos et al., 2021; Rozemberczki et al., 2021). Directed edge weights represent interregional mobility, node features contain the standardized case counts from the preceding eight days, and the target is the standardized case count on the following day. We normalize the adjacency matrices, construct non-overlapping windows of length 5, and split them chronologically into training, validation, and test sets using a $60\%/20\%/20\%$ ratio.

Twitter-Tennis. Twitter-Tennis contains 120 hourly mention-network snapshots from Roland-Garros 2017 (Béres et al., 2018; Rozemberczki et al., 2021). We retain the 1,000 most popular accounts; node features encode degree and local transitivity, and the target is the log-transformed number of mentions received in the next snapshot. We normalize the adjacency matrices, construct non-overlapping windows of length 10, and apply the same chronological $60\%/20\%/20\%$ split.

Model configurations. DG-FDE uses a two-layer ReLU multilayer perceptron to map the input features into a 32-dimensional latent space, vector field consists of a single GCN layer with ReLU

activation followed by a linear readout layer. All methods are trained with AdamW using a learning rate of 10^{-2} and weight decay of 10^{-4}. DG-FDE and the continuous-time baselines are trained for at most 200 epochs, with validation every five epochs after the first 20 epochs. Discrete-time baselines are trained for at most 2,000 epochs, with validation every 25 epochs after the first 200 epochs. We retain the checkpoint with the lowest validation MSE. GNCDE, STG-NCDE, and Perm-Equiv GNCDE use cubic interpolation and the Tsit5 solver.

Loss Function and Evaluation Metric. All models use the mean squared error (MSE) as both the training objective and evaluation metric. Let $\mathcal{D}$ contain $|\mathcal{D}|$ temporal windows, each with N nodes. The MSE is defined as

$$\mathrm{MSE} = \frac{1}{|\mathcal{D}|N} \sum_{s \in \mathcal{D}} \sum_{i=1}^{N} \left(\hat{y}_{s,i} - y_{s,i}\right)^2, \tag{41}$$

where $\hat{y}_{s,i}$ and $y_{s,i}$ denote the predicted and ground-truth values, respectively, for node i in the s-th temporal window.

D.2 Implementation Details for Link Prediction

In our experiments, we evaluate DG-FDE on seven publicly available dynamic graph datasets that are widely used for temporal link prediction. For the link-prediction baselines, we adopt the dataset-specific best-performing configurations established in the benchmark of Qi et al. (2025). We apply a consistent evaluation protocol across all methods.

Enron.[1] Enron is a temporal email communication network derived from the publicly released Enron email corpus. The nodes represent employees or email accounts, while each directed temporal edge indicates that an email was sent from one account to another at a particular time. The dataset captures the evolution of organizational communication and is used to predict future email interactions between users.

Contact.[2] Contact is a physical-proximity network derived from the Copenhagen Networks Study. The nodes correspond to university students carrying study-provided smartphones, and temporal edges represent proximity events detected through Bluetooth scans. This dataset provides fine-grained observations of evolving human contact patterns and is used to predict which pairs of individuals will interact in future time intervals.

Radoslaw Email.[3] (RDS for short) RDS records internal email communication among employees of a mid-sized manufacturing company. Each node represents an employee, and a directed temporal edge from one node to another corresponds to an individual email sent by the former to the latter. The evolving communication network is used to forecast future email interactions between employees.

UC Irvine Messages.[4] (UCI for short) UCI is an online social network constructed from private messages exchanged by students at the University of California, Irvine. Nodes denote users, and each directed temporal edge indicates that one user sent a private message to another at a specific time. Temporal link prediction on this dataset aims to identify user pairs that are likely to communicate in subsequent time steps.

Reddit-Title and Reddit-Body.[5] These two datasets describe directed hyperlink interactions between communities on Reddit. Their nodes represent subreddits, while a directed temporal edge is created when a post in a source subreddit contains a hyperlink to a post in a target subreddit. Reddit-Title contains hyperlinks extracted from post titles, whereas Reddit-Body contains those extracted from post bodies. The prediction task is to forecast future hyperlink connections between subreddits.

Autonomous Systems.[6] (AS733 for short) AS733 is a sequence of daily communication graphs constructed from Border Gateway Protocol records. Each node represents an autonomous system on

[1] https://www.cs.cmu.edu/~enron/

[2] https://doi.org/10.6084/m9.figshare.7267433

[3] https://networkrepository.com/ia-radoslaw-email.php

[4] https://snap.stanford.edu/data/CollegeMsg.html

[5] https://snap.stanford.edu/data/soc-RedditHyperlinks.html

[6] https://snap.stanford.edu/data/as-733.html

Table 4: Statistics of the dynamic graph snapshot datasets. The last column reports the number of snapshots in the chronological training/validation/test splits.

Dataset	# Nodes	# Edges	# Time Steps (Train/Val/Test)
Enron	184	10,472	30 / 5 / 10
Contact	694	463,558	470 / 67 / 135
RDS	167	82,927	70 / 10 / 20
UCI	1,899	59,835	61 / 9 / 17
Title	54,075	571,927	122 / 17 / 35
Body	35,776	286,562	122 / 17 / 35
AS733	7,716	1,167,892	70 / 10 / 20

the Internet, and an edge denotes a communication or peering relationship between two autonomous systems. It is used to predict future connections in the evolving Internet topology.

Table 4 summarizes the number of nodes, temporal edges, and chronological data splits. For every dataset, graph snapshots are ordered by time and divided chronologically into training, validation, and test intervals.

Hyperparameter. For a fair comparison, all methods follow a common experimental protocol. Specifically, we employ the Adam optimizer with an initial learning rate of 1×10^{-3} and a weight decay of 5×10^{-5}. The learning rate is adjusted using a cosine annealing schedule. Each model is trained for at most 50 epochs with an early-stopping patience of 20 epochs. We use the preceding 9 graph snapshots as historical input, search the hidden dimension over $\{64, 128, 256\}$ and the number of GCN layers in the vector field over $\{1, 2, 3\}$, and apply a dropout rate of 0.1. During training, one negative edge is sampled for each positive edge, whereas 100 negative edges per positive edge are used for evaluation. Gradient clipping is applied with a maximum ℓ_2-norm of 2.0.

Prediction Head. Given the node representations $\mathbf{h}_u^{(t)}$ and $\mathbf{h}_v^{(t)}$, we construct an edge representation using the Hadamard product and employ a two-layer MLP to estimate the link probability:

$$\hat{y}_{uv}^{(t)} = \sigma\Big(\mathbf{W}_2 \,\mathrm{Dropout}\Big(\mathrm{ReLU}\Big(\mathbf{W}_1\Big(\mathbf{h}_u^{(t)} \odot \mathbf{h}_v^{(t)}\Big) + \mathbf{b}_1\Big)\Big) + b_2\Big). \tag{42}$$

Loss Function. Let $\mathcal{E}_t^+$ and $\mathcal{E}_t^-$ denote the positive and sampled negative link sets, respectively. The model is optimized using a class-balanced binary cross-entropy loss:

$$\mathcal{L}_t = -\frac{\alpha \sum_{(u,v)\in\mathcal{E}_t^+} \log(\hat{y}_{uv}^{(t)} + \epsilon) + \sum_{(u,v)\in\mathcal{E}_t^-} \log(1 - \hat{y}_{uv}^{(t)} + \epsilon)}{|\mathcal{E}_t^+| + |\mathcal{E}_t^-|}, \qquad \alpha = \frac{|\mathcal{E}_t^-|}{|\mathcal{E}_t^+|}, \tag{43}$$

where $\epsilon = 10^{-15}$ ensures numerical stability. During training, one negative link is sampled for each positive link, yielding $\alpha = 1$.

Evaluation Metric. We adopt mean reciprocal rank (MRR) as the primary evaluation metric. For each positive link, we sample $M = 100$ negative links. Let s_i^+ and s_{ij}^- denote the scores of the i-th positive link and its j-th negative candidate, respectively. Using a tie-aware ranking protocol, MRR is computed as

$$r_i = 1 + \sum_{j=1}^{M} \mathbb{I}[s_{ij}^- > s_i^+] + \frac{1}{2}\sum_{j=1}^{M} \mathbb{I}[s_{ij}^- = s_i^+], \qquad \mathrm{MRR} = \frac{1}{N}\sum_{i=1}^{N} \frac{1}{r_i}, \tag{44}$$

where N is the number of positive test links. A higher MRR indicates that positive links are ranked above negative candidates more consistently. The checkpoint with the highest validation MRR is used for final testing.

Sensitivity to the fractional order. Figure 3 reports the link-prediction results of DG-FDE with fixed fractional orders $\alpha \in \{0.1, 0.2, \ldots, 1.0\}$ on all seven datasets. The results show that the preferred order varies across datasets and evaluation metrics, while $\alpha = 1$ corresponds to the integer-order setting.

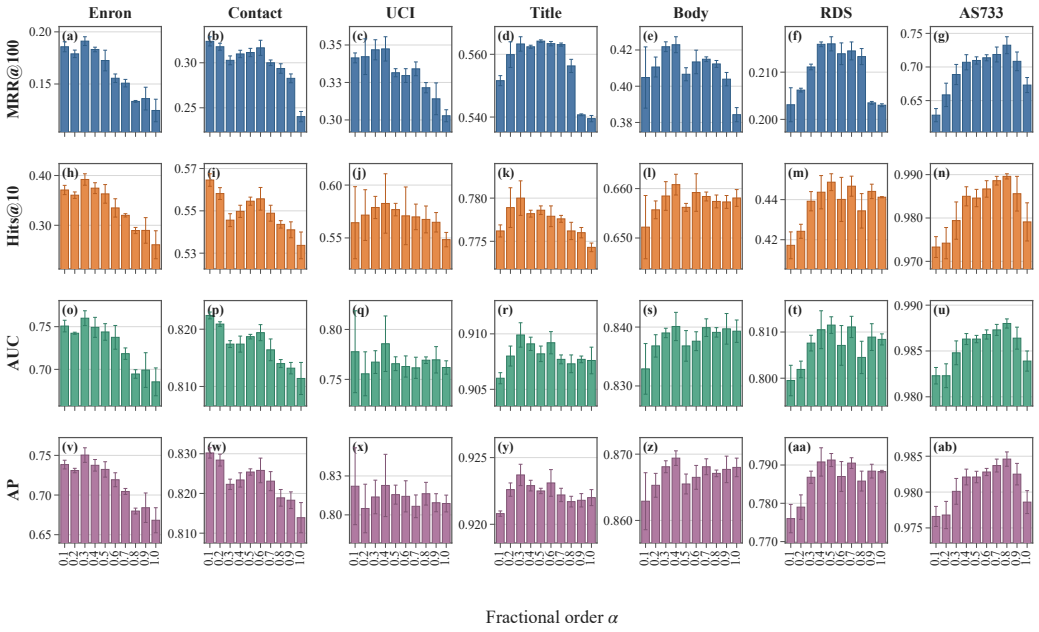


Figure 3: Sensitivity of DG-FDE to the fixed fractional order α on the seven temporal link-prediction datasets. Columns correspond to datasets and rows report MRR@100, Hits@10, AUC, and AP. The setting $\alpha = 1$ is the integer-order counterpart.