TANGO: Graph Neural Dynamics via Learned Energy and Tangential Flows

Moshe Eliasof
University of Cambridge
United Kingdom
me532@cam.ac.uk

Eldad Haber
University of British Columbia
Canada
ehaber@eoas.ubc.ca

Carola-Bibiane Schönlieb
University of Cambridge
United Kingdom
cbs31@cam.ac.uk

Abstract

We introduce TANGO, a dynamical-systems framework for graph representation learning that steers node features via a learned energy landscape. At its core is a learnable Lyapunov function whose gradient defines an energy-decreasing direction, guaranteeing stability and convergence. To preserve flexibility, we add a learned tangential message-passing component that evolves features along energy level sets. This orthogonal decomposition—gradient descent plus tangential evolution—enables effective signal propagation even in flat or ill-conditioned regions common in graph learning, mitigates oversquashing, and remains compatible with diverse GNN backbones. Empirically, TANGO achieves strong performance across node and graph classification and regression benchmarks, validating jointly learned energy functions and tangential flows.

1 Introduction

Graph Neural Networks (GNNs) excel on graph-structured data [12] but struggle with depth and long-range interactions due to vanishing gradients [3], over-smoothing [13, 68, 79], and over-squashing [2, 22, 39, 40, 86]. Framing GNNs as continuous-time dynamical systems (*neural ODEs*) [3, 15, 30, 73] enables stability analyses via diffusion [15], energy conservation [78], antisymmetric dynamics [39], and Hamiltonian flows [46]. Meanwhile, physics-informed architectures that encode conservation or dissipation improve stability and interpretability [6, 10, 36]. The common thread is reliance on an energy functional minimized or preserved by the GNN—typically simple (e.g., Dirichlet) [79]; yet many natural processes require richer energies: protein folding exhibits rugged, multi-funnel landscapes with multiple stable conformations and transition pathways [95], while complex reactions in computational chemistry demand sophisticated potential energy surfaces [81]. *Energy-based models (EBMs)* learn energies capturing data distributions for generative modeling [25, 42, 57, 97]. In contrast, we learn a *task-driven energy* whose minimization solves downstream tasks (e.g., node/graph classification). This raises a central question: *How can we learn such an energy and leverage it within a GNN to guide representation evolution throughout layers?* Our answer decomposes feature evolution into two orthogonal flows: (i) a *gradient-descent* component

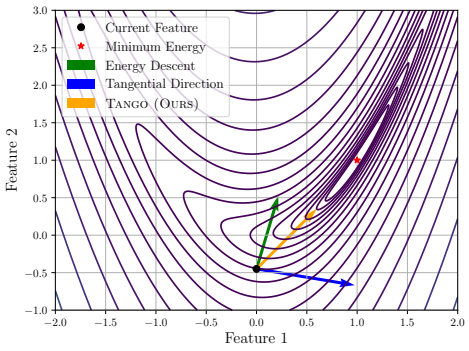


Figure 1: Illustration of TANGO dynamics in a 2D feature space. We plot level sets of a learned energy and visualize the energy-descent direction (green), the learned tangential direction (blue), and their combination (orange). The tangential component moves along level sets while the descent component reduces energy, enabling effective navigation of the learned landscape.

, TANGO: Graph Neural Dynamics via Learned Energy and Tangential Flows (Extended Abstract). Presented at the Fourth Learning on Graphs Conference (LoG 2025), Hybrid Event, December 10–12, 2025.

that decreases the learned energy, and (ii) a *tangential* component that moves along its level sets, preserving energy—promoting stability, interpretability, and mitigating over-squashing.

Our Approach. We introduce TANGO, a constrained graph-dynamics framework that embeds a learnable Lyapunov energy into message passing. Updates decompose into two GNN-parameterized flows: (i) an *energy-descent* component driving convergence to task-relevant solutions, and (ii) a *tangential, conservative* component that preserves energy while retaining flexibility. Under mild assumptions, TANGO satisfies Lyapunov conditions, ensuring stable dynamics; the tangential flow mitigates oversquashing by enabling expressive yet controlled propagation. As illustrated in Figure 1, the descent direction (green) lowers energy, the tangential direction (blue) follows level sets, and their combination (orange) defines the full update. Across standard graph benchmarks, this structured decomposition delivers strong empirical performance—competitive with, and often surpassing, widely used baselines—while maintaining controlled, stable feature dynamics.

2 Method

As discussed in Section 1, our goal is to learn a task-driven energy function and use it to improve downstream graph learning via TANgential- and Gradient-step Optimization of node features; we therefore call our method TANGO. Section 2.1 outlines the blueprint of TANGO, Appendix B details the implementation, Section 2.2 analyzes its properties, and Appendix E discusses complexity. Mathematical background and definitions of Lyapunov functions and stability are provided in Appendix A.

Notations. Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a graph with $n = |\mathcal{V}|$ nodes and $m = |\mathcal{E}|$ edges, and let $\mathbf{H}(t) = [\mathbf{h}_1(t), \dots, \mathbf{h}_n(t)]^\top \in \mathbb{R}^{n \times d}$ denote node features at continuous time t, where $\mathbf{h}_v(t) \in \mathbb{R}^d$ is the state of node v. Following dynamical-systems GNNs [3, 30, 39], in discrete architectures with finitely many layers we identify time t with depth ℓ and use $\mathbf{H}(t)$ and $\mathbf{H}^{(\ell)}$ interchangeably, as appropriate.

2.1 Optimizing Features with Energy Tangential and Gradient Steps

TANGO evolves node features via a dynamical system driven by a graph energy $V_\mathcal{G}$ with two flows:

$$\frac{d\mathbf{H}(t)}{dt} = \underbrace{-\alpha_\mathcal{G}(\mathbf{H}(t))\, \nabla_\mathbf{H} V_\mathcal{G}(\mathbf{H}(t))}_{\text{Energy Gradient Descent}} + \underbrace{\beta_\mathcal{G}(\mathbf{H}(t))\, T_{V_\mathcal{G}}(\mathbf{H}(t))}_{\text{Tangential Direction}}, \tag{1}$$

where $\alpha_\mathcal{G}, \beta_\mathcal{G} \geq 0$ weight the flows, $\nabla_\mathbf{H} V_\mathcal{G}$ is the energy gradient, and $T_{V_\mathcal{G}}(\mathbf{H}(t))$ is an update direction orthogonal to it (i.e., tangential to level sets). Many orthogonal directions are possible; Appendix B details how we learn $T_{V_\mathcal{G}}$ and implement the system. By design, the first term decreases energy, while the tangential flow preserves it.

Tangential Flow. Setting $\beta_\mathcal{G} = 0$ in Equation (1) yields pure energy–gradient flow—dissipative but often slow [9, 67] and restrictive for training. While generative models tolerate hundreds–thousands of steps, this is impractical for downstream learning because it implies as many effective layers—hard to train [71] and computationally costly. To accelerate minimization, we add a *tangential* flow along level sets of $V_\mathcal{G}$ that preserves energy. As shown in Figure 1 and analyzed in Section 2.2, coupling this tangential flow with gradient descent yields a better descent direction and faster convergence.

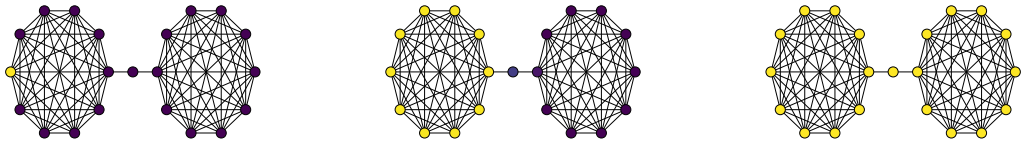
To obtain a direction orthogonal to $\nabla_\mathbf{H} V_\mathcal{G}(\mathbf{H}(t))$, let $\mathbf{M}(\mathbf{H}(t))$ be a predicted update. Define

$$T_{V_\mathcal{G}}(\mathbf{H}(t)) = \mathbf{M}(\mathbf{H}(t)) - \left\langle \mathbf{M}(\mathbf{H}(t)), \widehat{\nabla}_\mathbf{H} V_\mathcal{G}(\mathbf{H}(t)) \right\rangle \cdot \nabla_\mathbf{H} V_\mathcal{G}(\mathbf{H}(t)), \tag{2}$$

where $\widehat{\nabla}_\mathbf{H} V_\mathcal{G}(\mathbf{H}(t))$ is the normalized energy gradient. If $\nabla_\mathbf{H} V_\mathcal{G}(\mathbf{H}(t)) = 0$, set $T_{V_\mathcal{G}}(\mathbf{H}(t)) = \mathbf{M}(\mathbf{H}(t))$. The projection removes the component of $\mathbf{M}(\mathbf{H}(t))$ along the descent direction, ensuring $T_{V_\mathcal{G}}$ is orthogonal to $\nabla_\mathbf{H} V_\mathcal{G}$. In Appendix B we specify the parameterization of TANGO as a GNN.

2.2 Theoretical Properties of TANGO

We analyze the continuous-time dynamics of TANGO in (1), focusing on (i) *energy dissipation*, (ii) *feature evolution in flat energy landscapes*, and (iii) *the benefit of the tangential direction*. Proofs appear in Appendix D. Throughout our analysis, we assume that (i) The input graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is connected; (ii) $V_\mathcal{G}(\mathbf{H}(t))$ is twice differentiable and bounded below. For brevity, we omit time/layer



(a) Initial Features **(b)** Gradient Flow **(c)** TANGO

Figure 2: Comparison of propagation behaviors between gradient flow and TANGO with 50 layers. While gradient flow struggles propagating information through the bottleneck, our TANGO is effective.

superscripts and write $\mathbf{H}$ for node features when clear from the context. We start by showing that TANGO is dissipative if $\|\nabla_{\mathbf{H}} V_{\mathcal{G}}(\mathbf{H})\|^2 > 0$, and $\alpha_{\mathcal{G}} \geq 0$ (obtained by design), corresponding to the Lyapunov stability criterion from Theorem 1.

Proposition 1 (Energy Dissipation)**.** *Suppose $\alpha_{\mathcal{G}} \geq 0$ and $\|\nabla_{\mathbf{H}} V_{\mathcal{G}}(\mathbf{H})\|^2 > 0$. Then the energy $V_{\mathcal{G}}(\mathbf{H})$ is non-increasing along trajectories of Equation (1). Specifically,*

$$\frac{d}{dt} V_{\mathcal{G}}(\mathbf{H}) \quad = \quad -\alpha_{\mathcal{G}}(\mathbf{H}) \, \|\nabla_{\mathbf{H}} V_{\mathcal{G}}(\mathbf{H})\|^2 \leq 0. \tag{3}$$

We now show that unlike gradient flows, our TANGO admits evolution of node features in flat energy landscapes, a prime challenge in optimization techniques [9, 67].

Proposition 2 (TANGO can Evolve Features in Flat Energy Landscapes)**.** *Suppose $\nabla_{\mathbf{H}} V_{\mathcal{G}}(\mathbf{H}) = 0$, and $T_{V_{\mathcal{G}}}(\mathbf{H}) \neq 0$, then the* TANGO *flow in Equation (1) reads: $\frac{d\mathbf{H}}{dt} = \beta_{\mathcal{G}}(\mathbf{H}) T_{V_{\mathcal{G}}}(\mathbf{H})$. This implies that in* contrast *to gradient flows, the dynamics of* TANGO *can evolve even in regions where the energy landscape is flat.*

Theoretical Benefits of Using the Tangent Direction. TANGO combines two terms (see Equations (1) and (6)): the energy gradient $\nabla_{\mathbf{H}} V_{\mathcal{G}}(\mathbf{H}^{(\ell)})$ and the tangential direction $T_{V_{\mathcal{G}}}(\mathbf{H})$. A natural question is: *under what conditions does adding the tangential direction improve over plain gradient descent?* We address this by first recalling a classic convergence result for gradient-based minimization.

Proposition 3 (Convergence of Gradient Descent of a Scalar Function, Nocedal and Wright [67])**.** *Let $V_{\mathcal{G}}(\cdot)$ be a scalar function and let $\mathbf{H}^{(\ell+1)} = \mathbf{H}^{(\ell)} - \alpha_{\mathcal{G}}^{(\ell)}(\mathbf{H}^{(\ell)}) \nabla_{\mathbf{H}} V_{\mathcal{G}}(\mathbf{H}^{(l)})$ be a gradient-descent iteration of the energy $V_{\mathcal{G}}(\cdot)$. Then, a linear convergence is obtained, with convergence rate: $r = \frac{\lambda_{\max} - \lambda_{\min}}{\lambda_{\max} + \lambda_{\min}}$, where $\lambda_{\max}$ is the maximal eigenvalue, and in the case of problems that involve the graph Laplacian, $\lambda_{\min}$ is the second minimal eigenvalue, i.e., the first non-zero eigenvalue of the Hessian of $V_{\mathcal{G}}(\cdot)$.*

Proposition 3 shows that gradient descent struggles in ill-conditioned problems (large $\lambda_{\max}/\lambda_{\min}$), which is common in graph tasks where the Hessian inherits poor conditioning from the graph Laplacian—especially under oversquashing caused by bottlenecks [22, 38, 86]. As an alternative, we add an orthogonal flow to the gradient direction; the combined update direction is

$$\mathbf{D} = \alpha_{\mathcal{G}}(\mathbf{H}^{(\ell)}) \nabla_{\mathbf{H}} V_{\mathcal{G}}(\mathbf{H}^{(\ell)}) + \beta_{\mathcal{G}}(\mathbf{H}^{(\ell)}) T_{V_{\mathcal{G}}}(\mathbf{H}^{(\ell)}). \tag{4}$$

The following proposition demonstrates that it is possible to learn T such that $\mathbf{D}$ becomes the Newton direction, which offers quadratic convergence [67].

Proposition 4 (TANGO can learn a Quadratic Convergence Direction)**.** *Assume for simplicity that $\beta_{\mathcal{G}} = 1$, and that the Hessian of $V_{\mathcal{G}}$ is invertible. Let $\mathbf{D} = \alpha_{\mathcal{G}}(\mathbf{H}^{(\ell)}) \nabla_{\mathbf{H}} V_{\mathcal{G}}(\mathbf{H}^{(\ell)}) + T_{V_{\mathcal{G}}}(\mathbf{H}^{(\ell)})$ with $\left\langle T_{V_{\mathcal{G}}}(\mathbf{H}^{(\ell)}), \widehat{\nabla}_{\mathbf{H}} V_{\mathcal{G}}(\mathbf{H}^{(\ell)}) \right\rangle = 0$. Then, it is possible to learn a direction $T_{V_{\mathcal{G}}}(\mathbf{H}^{(\ell)})$ and a step size $\alpha_{\mathcal{G}}$ such that $\mathbf{D}$ is the Newton direction, $\mathbf{N} = (\nabla^2 V_{\mathcal{G}})^{-1} \nabla V_{\mathcal{G}}$.*

Beyond improved global convergence, Newton's method has local convergence independent of the Hessian condition number [9, 67]. Hence, if the tangential flow is learned to approximate the Newton direction, TANGO can overcome slow convergence in highly ill-conditioned landscapes—an effect well known in second-order methods and their approximations, such as conjugate gradients (CG) and L-BFGS [9, 67]. *In graph learning*, Proposition 4 is especially relevant to oversquashing [2, 22]: the graph Laplacian's smallest eigenvalue is zero (for connected graphs) and the second smallest is near zero [7, 38, 50, 86], yielding poor conditioning; under these conditions, gradient-flow methods

Table 1: Test performance in five benchmarks from [29]. Shown is the mean $\pm$std of 4 runs with different random seeds. Highlighted are the top **first**, **second**, and **third** results.

Model	ZINC-12k	MNIST	CIFAR10	PATTERN	CLUSTER
	MAE↓	Accuracy↑	Accuracy↑	Accuracy↑	Accuracy↑
GCN [52]	$0.367_{\pm 0.011}$	$90.705_{\pm 0.218}$	$55.710_{\pm 0.381}$	$71.892_{\pm 0.334}$	$68.498_{\pm 0.976}$
GatedGCN [11]	$0.282_{\pm 0.015}$	$97.340_{\pm 0.143}$	$67.312_{\pm 0.311}$	$85.568_{\pm 0.088}$	$73.840_{\pm 0.326}$
EGT [49]	$0.108_{\pm 0.009}$	**98.173**$_{\pm 0.087}$	$68.702_{\pm 0.409}$	**86.821**$_{\pm 0.020}$	**79.232**$_{\pm 0.348}$
GPS [74]	**0.070**$_{\pm 0.004}$	$98.051_{\pm 0.126}$	**72.298**$_{\pm 0.356}$	$86.685_{\pm 0.059}$	$78.016_{\pm 0.180}$
GRIT [62]	**0.059**$_{\pm 0.002}$	**98.108**$_{\pm 0.111}$	**76.468**$_{\pm 0.881}$	**87.196**$_{\pm 0.076}$	**80.026**$_{\pm 0.277}$
TANGO$_{\text{GatedGCN}}$	$0.128_{\pm 0.011}$	$97.788_{\pm 0.105}$	$70.894_{\pm 0.329}$	$86.672_{\pm 0.071}$	$78.194_{\pm 0.307}$
TANGO$_{\text{GPS}}$	**0.062**$_{\pm 0.005}$	**98.197**$_{\pm 0.110}$	**75.783**$_{\pm 0.261}$	**87.182**$_{\pm 0.063}$	**80.113**$_{\pm 0.138}$

implicitly implemented by common GNNs [23] perform poorly, limiting information propagation between nodes. By enabling feature updates that approximate second-order information, TANGO actively mitigates oversquashing. We empirically validate this in Figure 2 by comparing TANGO with Dirichlet-energy minimization commonly used by baseline GNNs [23, 79]; details are in Appendix F.

3 Experiments

We consider synthetic long-range [39], LRGB [28], Dwivedi et al. [29], and heterophilic node classification [72]. TANGO consistently improves its backbones and is competitive with other GNN architectures. Experimental details are in Appendix F, and additional results are in Appendix G. Throughout our experiments, we color the top 3 performing models. In case more than one variant of our TANGO is among them, we mark only the best variant.

Graph Property Prediction. We evaluate TANGO on the three graph property prediction tasks from Gravina et al. [39]—graph diameter, single-source shortest paths (SSSP), and node eccentricity on synthetic graphs—which require propagating information beyond immediate neighbors; performance thus reflects long-range interaction ability. Table 2 reports mean test $\log_{10}(\text{MSE})$, comparing TANGO to MPNNs, DE-GNNs, and transformer-based models. Across all tasks and variants, TANGO achieves the lowest (best) error. For eccentricity, TANGO$_{\text{GPS}}$ lowers error by > 1.2 points vs. PH-DGN [46] and by > 2.0 vs. SWAN—both designed for long-radius propagation. These results validate TANGO's effectiveness at modeling long-range interactions and mitigating oversquashing. Moreover, augmenting simple backbones such as GatedGCN with TANGO consistently outperforms the baseline GatedGCN, indicating that TANGO enhances traditional MPNNs.

Table 2: Mean test set $log_{10}(\text{MSE})(\downarrow)$ and std averaged on 4 random weight initializations on Graph Property Prediction. Lower is better. **First**, **second**, and **third** best results for each task are color-coded.

Model	Diameter	SSSP	Eccentricity
MPNNs			
GatedGCN [11]	$0.1348_{\pm 0.0397}$	$-3.2610_{\pm 0.0514}$	$0.6995_{\pm 0.0302}$
GCN [52]	$0.7424_{\pm 0.0466}$	$0.9499_{\pm 0.0001}$	$0.8468_{\pm 0.0028}$
GAT [89]	$0.8221_{\pm 0.0752}$	$0.6951_{\pm 0.1499}$	$0.7909_{\pm 0.0222}$
GraphSAGE [44]	$0.8645_{\pm 0.0401}$	$0.2863_{\pm 0.1843}$	$0.7863_{\pm 0.0207}$
GIN [98]	$0.6131_{\pm 0.0990}$	$-0.5408_{\pm 0.4193}$	$0.9504_{\pm 0.0007}$
GCNII [16]	$0.5287_{\pm 0.0570}$	$-1.1329_{\pm 0.0135}$	$0.7640_{\pm 0.0355}$
DE-GNNs			
DGC [73]	$0.6028_{\pm 0.0050}$	$-0.1483_{\pm 0.0231}$	$0.8261_{\pm 0.0032}$
GRAND [15]	$0.6715_{\pm 0.0490}$	$-0.0942_{\pm 0.3897}$	$0.6602_{\pm 0.1393}$
GraphCON [78]	$0.0964_{\pm 0.0620}$	$-1.3836_{\pm 0.0092}$	$0.6833_{\pm 0.0074}$
A-DGN [39]	$-0.5188_{\pm 0.1812}$	$-3.2417_{\pm 0.0751}$	$0.4296_{\pm 0.1003}$
SWAN [40]	**-0.5981**$_{\pm 0.1145}$	$-3.5425_{\pm 0.0830}$	**-0.0739**$_{\pm 0.2190}$
PH-DGN [46]	**-0.5385**$_{\pm 0.0187}$	**-4.2993**$_{\pm 0.0721}$	**-0.9348**$_{\pm 0.2097}$
Transformers			
GPS [74]	$-0.5121_{\pm 0.0426}$	**-3.5990**$_{\pm 0.1949}$	$0.6077_{\pm 0.0282}$
Ours			
TANGO$_{\text{GatedGCN}}$	$-0.6681_{\pm 0.0745}$	$-5.0626_{\pm 0.0742}$	$-1.7419_{\pm 0.0106}$
TANGO$_{\text{GPS}}$	**-0.9772**$_{\pm 0.0518}$	**-5.5263**$_{\pm 0.0838}$	**-2.1455**$_{\pm 0.0033}$

GNN Benchmarking from Dwivedi et al. [29]. To further evaluate TANGO, we use the standard GNN benchmarks from Dwivedi et al. [29], widely used for SOTA evaluation [62]. For fair comparison, we follow Dwivedi et al. [29]'s training and evaluation protocols. Table 1 reports mean$\pm$std test metrics: MAE for ZINC-12k (regression) and accuracy(%) for all others. Across all benchmarks, TANGO consistently improves its backbone and often outperforms strong baselines.

Ablation Studies. In Appendix G.4 we provide ablation studies to understand the different design choices and components in our TANGO. In particular, we provide experiments to answers two questions. (i) Does downstream performance benefit from incorporating a tangential term even when

the underlying GNN is not the gradient of an energy function?; and (ii) Is the observed improvement due to the tangential nature of the added component, or simply due to additional parameters and network?

4 Conclusions

TANGO is a framework for graph neural dynamics that jointly models an energy-descent direction and a tangential flow. Casting message passing through Lyapunov theory and continuous dynamics, TANGO unifies task-driven energy-based modeling with learnable tangential flows that accelerate energy minimization. The tangential component sustains feature evolution in flat or ill-conditioned landscapes—unlike pure gradient flows—and mitigates oversquashing. Empirically, TANGO attains strong performance across 15 synthetic and real-world benchmarks, outperforming message-passing, diffusion-based, and attention-based GNNs. Future work: integrate higher-order differential operators into the tangential mechanism and develop analysis/regularization for the learned energy landscape.

References

[1] Sami Abu-El-Haija, Bryan Perozzi, Amol Kapoor, Nazanin Alipourfard, Kristina Lerman, Hrayr Harutyunyan, Greg Ver Steeg, and Aram Galstyan. Mixhop: Higher-order graph convolutional architectures via sparsified neighborhood mixing. In *International Conference on Machine Learning*, pages 21–29. PMLR, 2019. 17, 21

[2] Uri Alon and Eran Yahav. On the bottleneck of graph neural networks and its practical implications. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=i80OPhOCVH2. 1, 3, 14, 15

[3] Álvaro Arroyo, Alessio Gravina, Benjamin Gutteridge, Federico Barbero, Claudio Gallicchio, Xiaowen Dong, Michael Bronstein, and Pierre Vandergheynst. On vanishing gradients, over-smoothing, and over-squashing in gnns: Bridging recurrent and graph learning. *arXiv preprint arXiv:2502.10818*, 2025. URL https://arxiv.org/abs/2502.10818. 1, 2, 13

[4] Minkyung Baek, Frank DiMaio, Ivan Anishchenko, Justas Dauparas, Sergey Ovchinnikov, Gyu Rie Lee, Jue Wang, Qian Cong, Lisa N Kinch, R Dustin Schaeffer, et al. Accurate prediction of protein structures and interactions using a three-track neural network. *Science*, 373(6557):871–876, 2021. 14

[5] Ali Behrouz and Farnoosh Hashemi. Graph Mamba: Towards Learning on Graphs with State Space Models, 2024. URL https://arxiv.org/abs/2402.08678. 17

[6] Ravinder Bhattoo, Sayan Ranu, and N. M. Anoop Krishnan. Learning articulated rigid body dynamics with lagrangian graph neural networks. In *Advances in Neural Information Processing Systems*, volume 35, pages 29789–29800, 2022. URL https://arxiv.org/abs/2209.11588. 1

[7] Mitchell Black, Zhengchao Wan, Amir Nayyeri, and Yusu Wang. Understanding oversquashing in gnns through the lens of effective resistance. In *Proceedings of the 40th International Conference on Machine Learning*, pages 2528–2547. PMLR, 2023. 3

[8] Deyu Bo, Xiao Wang, Chuan Shi, and Huawei Shen. Beyond low-frequency information in graph convolutional networks. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(5):3950–3957, May 2021. doi: 10.1609/aaai.v35i5.16514. URL https://ojs.aaai.org/index.php/AAAI/article/view/16514. 17

[9] Stephen P Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004. 2, 3, 15

[10] Johannes Brandstetter, Daniel E. Worrall, and Max Welling. Message passing neural PDE solvers. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=vSix3HPYKSU. 1

[11] Xavier Bresson and Thomas Laurent. Residual Gated Graph ConvNets. *arXiv preprint arXiv:1711.07553*, 2018. 4, 13, 17, 21, 25

[12] Michael M Bronstein, Joan Bruna, Taco Cohen, and Petar Veličković. Geometric deep learning: Grids, groups, graphs, geodesics, and gauges. *arXiv preprint arXiv:2104.13478*, 2021. 1

[13] Chen Cai and Yusu Wang. A note on over-smoothing for graph neural networks. *arXiv preprint arXiv:2006.13318*, 2020. 1, 14

[14] Benjamin Chamberlain, James Rowbottom, Davide Eynard, Francesco Di Giovanni, Xiaowen Dong, and Michael Bronstein. Beltrami flow and neural diffusion on graphs. In *Advances in Neural Information Processing Systems*, volume 34, 2021. 14

[15] Benjamin Paul Chamberlain, James Rowbottom, Maria Gorinova, Stefan Webb, Emanuele Rossi, and Michael M Bronstein. GRAND: Graph neural diffusion. In *International Conference on Machine Learning (ICML)*, pages 1407–1418. PMLR, 2021. 1, 4, 13, 14, 17, 21, 25

[16] Ming Chen, Zhewei Wei, Zengfeng Huang, Bolin Ding, and Yaliang Li. Simple and Deep Graph Convolutional Networks. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 1725–1735. PMLR, 13–18 Jul 2020. 4, 17, 21, 25

[17] Tian Qi Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. In *Advances in Neural Information Processing Systems*, pages 6571–6583, 2018. 14

[18] Eli Chien, Jianhao Peng, Pan Li, and Olgica Milenkovic. Adaptive universal generalized pagerank graph neural network. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=n6jl7fLxrP. 17

[19] Jeongwhan Choi, Seoyoung Hong, Noseong Park, and Sung-Bae Cho. Gread: Graph neural reaction-diffusion networks. In *ICML*, 2023. 13, 14

[20] Krzysztof Choromanski, Marcin Kuczynski, Jacek Cieszkowski, Paul L. Beletsky, Konrad M. Smith, Wojciech Gajewski, Gabriel De Masson, Tomasz Z. Broniatowski, Antonina B. Gorny, Leszek M. Kaczmarek, and Stanislaw K. Andrzejewski. Performers: A new approach to scaling transformers. *Proceedings of the 37th International Conference on Machine Learning (ICML)*, pages 2020–2031, 2020. URL https://arxiv.org/abs/2009.14743. 14

[21] Krzysztof Choromanski, Han Lin, Haoxian Chen, Tianyi Zhang, Arijit Sehanobish, Valerii Likhosherstov, Jack Parker-Holder, Tamas Sarlos, Adrian Weller, and Thomas Weingarten. From block-toeplitz matrices to differential equations on graphs: towards a general theory for scalable masked transformers. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162, pages 3962–3983. PMLR, 2022. 14

[22] Francesco Di Giovanni, Lorenzo Giusti, Federico Barbero, Giulia Luise, Pietro Liò, and Michael Bronstein. On over-squashing in message passing neural networks: the impact of width, depth, and topology. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org, 2023. 1, 3, 14, 15

[23] Francesco Di Giovanni, James Rowbottom, Benjamin P. Chamberlain, Thomas Markovich, and Michael M. Bronstein. Graph neural networks as gradient flows. In *International Conference on Learning Representations (ICLR)*, 2023. URL https://arxiv.org/abs/2206.10991. 4, 14

[24] Lun Du, Xiaozhou Shi, Qiang Fu, Xiaojun Ma, Hengyu Liu, Shi Han, and Dongmei Zhang. Gbk-gnn: Gated bi-kernel graph neural networks for modeling both homophily and heterophily. In *Proceedings of the ACM Web Conference 2022*, WWW '22, page 1550–1558, New York, NY, USA, 2022. Association for Computing Machinery. ISBN 9781450390965. doi: 10.1145/3485447.3512201. URL https://doi.org/10.1145/3485447.3512201. 17

[25] Yilun Du and Igor Mordatch. Implicit generation and modeling with energy based models. In *Advances in Neural Information Processing Systems*, volume 32, 2019. 1, 14

[26] Vijay Prakash Dwivedi and Xavier Bresson. A Generalization of Transformer Networks to Graphs. *AAAI Workshop on Deep Learning on Graphs: Methods and Applications*, 2021. 17

[27] Vijay Prakash Dwivedi, Anh Tuan Luu, Thomas Laurent, Yoshua Bengio, and Xavier Bresson. Graph neural networks with learnable structural and positional representations. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=wTTjnvGphYj. 14

[28] Vijay Prakash Dwivedi, Ladislav Rampášek, Michael Galkin, Ali Parviz, Guy Wolf, Anh Tuan Luu, and Dominique Beaini. Long Range Graph Benchmark. In *Advances in Neural Information Processing Systems*, volume 35, pages 22326–22340. Curran Associates, Inc., 2022. 4, 14, 19, 20, 23

[29] Vijay Prakash Dwivedi, Chaitanya K Joshi, Anh Tuan Luu, Thomas Laurent, Yoshua Bengio, and Xavier Bresson. Benchmarking graph neural networks. *Journal of Machine Learning Research*, 24(43):1–48, 2023. 4, 14, 18, 21

[30] Moshe Eliasof, Eldad Haber, and Eran Treister. PDE-GCN: Novel architectures for graph neural networks motivated by partial differential equations. *Advances in Neural Information Processing Systems*, 34:3836–3849, 2021. 1, 2, 13, 14

[31] Moshe Eliasof, Eldad Haber, and Eran Treister. Feature transportation improves graph neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 11874–11882, 2024. 14

[32] Moshe Eliasof, Eldad Haber, Eran Treister, and Carola-Bibiane B Schönlieb. On the temporal domain of differential equation inspired graph neural networks. In *International Conference on Artificial Intelligence and Statistics*, pages 1792–1800. PMLR, 2024. 14

[33] Brandon C Fallin, Cristian F Nino, Omkar Sudhir Patil, Zachary I Bell, and Warren E Dixon. Lyapunov-based graph neural networks for adaptive control of multi-agent systems. *arXiv preprint arXiv:2503.15360*, 2025. 14

[34] Ben Finkelshtein, Xingyue Huang, Michael M. Bronstein, and Ismail Ilkan Ceylan. Cooperative Graph Neural Networks. In *Forty-first International Conference on Machine Learning*, 2024. URL https://openreview.net/forum?id=ZQcqXCuoxD. 17, 24

[35] Scott Freitas, Yuxiao Dong, Joshua Neil, and Duen Horng Chau. A large-scale database for graph representation learning. In J. Vanschoren and S. Yeung, editors, *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, volume 1, 2021. 19

[36] Han Gao, Matthew J Zahr, and Jian-Xun Wang. Physics-informed graph neural galerkin networks: A unified framework for solving pde-governed forward and inverse problems. *Computer Methods in Applied Mechanics and Engineering*, 390:114502, 2022. 1

[37] Johannes Gasteiger, Stefan Weiß enberger, and Stephan Günnemann. Diffusion Improves Graph Learning. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. 17, 21

[38] Jhony H Giraldo, Konstantinos Skianis, Thierry Bouwmans, and Fragkiskos D Malliaros. On the trade-off between over-smoothing and over-squashing in deep graph neural networks. In *Proceedings of the 32nd ACM international conference on information and knowledge management*, pages 566–576, 2023. 3

[39] Alessio Gravina, Davide Bacciu, and Claudio Gallicchio. Anti-Symmetric DGN: a stable architecture for Deep Graph Networks. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=J3Y7cgZOOS. 1, 2, 4, 13, 14, 17, 18, 21, 25

[40] Alessio Gravina, Moshe Eliasof, Claudio Gallicchio, Davide Bacciu, and Carola-Bibiane Schönlieb. On oversquashing in graph neural networks through the lens of dynamical systems. In *The 39th Annual AAAI Conference on Artificial Intelligence*, 2025. 1, 4, 14, 17, 21, 25

[41] Albert Gu, Karan Goel, and Christopher Re. Efficiently modeling long sequences with structured state spaces. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=uYLFoz1vlAC. 15

[42] Qiushan Guo, Yifan Zhang, Yifan Wang, Yizhou Wang, and Hongsheng Li. Egc: Image generation and classification via a diffusion energy-based model. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12345–12354, 2023. 1, 14

[43] Benjamin Gutteridge, Xiaowen Dong, Michael M Bronstein, and Francesco Di Giovanni. Drew: Dynamically rewired message passing with delay. In *International Conference on Machine Learning*, pages 12252–12267. PMLR, 2023. 14, 17, 19, 21, 23

[44] William L. Hamilton, Rex Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 1025–1035. Curran Associates Inc., 2017. ISBN 9781510860964. 4, 17, 25

[45] Andi Han, Dai Shi, Lequan Lin, and Junbin Gao. From continuous dynamics to graph neural networks: Neural diffusion and beyond. *arXiv preprint arXiv:2310.10121*, 2023. 14

[46] Simon Heilig, Alessio Gravina, Alessandro Trenta, Claudio Gallicchio, and Davide Bacciu. Port-Hamiltonian Architectural Bias for Long-Range Propagation in Deep Graph Networks. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=03EkqSCKuO. 1, 4, 21, 25

[47] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9 (8):1735–1780, 1997. doi: 10.1162/neco.1997.9.8.1735. 15

[48] Weihua Hu, Bowen Liu, Joseph Gomes, Marinka Zitnik, Percy Liang, Vijay Pande, and Jure Leskovec. Strategies for Pre-training Graph Neural Networks. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=HJlWWJSFDH. 17

[49] Md Shamim Hussain, Mohammed J Zaki, and Dharmashankar Subramanian. Global self-attention as a replacement for graph convolution. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 655–665, 2022. 4

[50] Adarsh Jamadandi, Celia Rubio-Madrigal, and Rebekka Burkholz. Spectral graph pruning against over-squashing and over-smoothing. In *Advances in Neural Information Processing Systems*, 2024. 3

[51] Hassan K Khalil and Jessy W Grizzle. *Nonlinear systems*, volume 3. Prentice hall Upper Saddle River, NJ, 2002. 13

[52] T. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. *Proceedings of the International Conference on Learning Representations*, 2016. 4, 17, 21, 25

[53] Kezhi Kong, Jiuhai Chen, John Kirchenbauer, Renkun Ni, C. Bayan Bruss, and Tom Goldstein. GOAT: A global transformer on large-scale graphs. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett, editors, *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 17375–17390. PMLR, 23–29 Jul 2023. URL https://proceedings.mlr.press/v202/kong23a.html. 17

[54] Devin Kreuzer, Dominique Beaini, Will Hamilton, Vincent Létourneau, and Prudencio Tossou. Rethinking graph transformers with spectral attention. *Advances in Neural Information Processing Systems*, 34:21618–21629, 2021. 21

[55] Devin Kreuzer, Dominique Beaini, Will Hamilton, Vincent Létourneau, and Prudencio Tossou. Rethinking graph transformers with spectral attention. *Advances in Neural Information Processing Systems*, 34:21618–21629, 2021. 17

[56] Sven Kreuzer, Michael Reiner, and Stefan D. D. De Villiers. Sant: Structural attention networks for graphs. *Proceedings of the 38th International Conference on Machine Learning (ICML)*, 2021. 14

[57] Yann LeCun, Sumit Chopra, Raia Hadsell, Marc'Aurelio Ranzato, and Fu-Jie Huang. A tutorial on energy-based learning. *Predicting structured data*, 1(0):1–59, 2006. 1, 14

[58] Xiang Li, Renyu Zhu, Yao Cheng, Caihua Shan, Siqiang Luo, Dongsheng Li, and Weining Qian. Finding global homophily in graph neural networks when meeting heterophily. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 13242–13256. PMLR, 17–23 Jul 2022. URL https://proceedings.mlr.press/v162/li22ad.html. 17

[59] Daniil Likhobaba, Nikita Pavlichenko, and Dmitry Ustalov. Toloker Graph: Interaction of Crowd Annotators, February 2023. URL https://doi.org/10.5281/zenodo.7620796. 19

[60] Meng Liu, Keqiang Yan, Bora Oztekin, and Shuiwang Ji. GraphEBM: Molecular graph generation with energy-based models. In *Energy Based Models Workshop - ICLR 2021*, 2021. URL https://openreview.net/forum?id=Gc51PtL_zYw. 14

[61] Sitao Luan, Chenqing Hua, Qincheng Lu, Liheng Ma, Lirong Wu, Xinyu Wang, Minkai Xu, Xiao-Wen Chang, Doina Precup, Rex Ying, Stan Z. Li, Jian Tang, Guy Wolf, and Stefanie Jegelka. The heterophilic graph learning handbook: Benchmarks, models, theoretical analysis, applications and challenges, 2024. URL https://arxiv.org/abs/2407.09618. 14, 24

[62] Liheng Ma, Chen Lin, Derek Lim, Adriana Romero-Soriano, Puneet K Dokania, Mark Coates, Philip Torr, and Ser-Nam Lim. Graph inductive biases in transformers without message passing. In *International Conference on Machine Learning*, pages 23321–23337. PMLR, 2023. 4, 18

[63] Thomas Markovich. Qdc: Quantum diffusion convolution kernels on graphs, 2023. 14

[64] Sohir Maskey, Raffaele Paolino, Aras Bacho, and Gitta Kutyniok. A fractional graph laplacian approach to oversmoothing. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=kS7ED7eE74. 14

[65] Sunil Kumar Maurya, Xin Liu, and Tsuyoshi Murata. Simplifying approach to node classification in graph neural networks. *Journal of Computational Science*, 62:101695, 2022. ISSN 1877-7503. doi: https://doi.org/10.1016/j.jocs.2022.101695. URL https://www.sciencedirect.com/science/article/pii/S1877750322000990. 17

[66] Luis Müller, Mikhail Galkin, Christopher Morris, and Ladislav Rampášek. Attending to graph transformers. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL https://openreview.net/forum?id=HhbqHBBrfZ. 24

[67] J. Nocedal and S. Wright. *Numerical Optimization*. Springer, New York, 1999. 2, 3, 15, 16

[68] Hoang Nt and Takanori Maehara. Revisiting graph neural networks: All we have is low-pass filters. *arXiv preprint arXiv:1905.09550*, 2019. 1, 14

[69] Kenta Oono and Taiji Suzuki. Graph neural networks exponentially lose expressive power for node classification. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=S1ldO2EFPr. 14

[70] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019. 17

[71] Jie Peng, Runlin Lei, and Zhewei Wei. Beyond over-smoothing: Uncovering the trainability challenges in deep graph neural networks. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, pages 1878–1887, 2024. 2

[72] Oleg Platonov, Denis Kuznedelev, Michael Diskin, Artem Babenko, and Liudmila Prokhorenkova. A critical look at the evaluation of GNNs under heterophily: Are we really making progress? In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=tJbbQfw-5wv. 4, 19, 20, 22, 24

[73] 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. 1, 4, 14, 25

[74] Petr Rampášek, Mikhail Galkin, Vijay Prakash Dwivedi, Anh Tuan Luu, Guy Wolf, and Dominique Beaini. Recipe for a general, powerful, scalable graph transformer (graphgps). In *Advances in Neural Information Processing Systems*, volume 35, pages 28877–28890, 2022. URL https://arxiv.org/abs/2205.12454. 4, 13, 21, 25

[75] Ladislav Rampášek, Mikhail Galkin, Vijay Prakash Dwivedi, Anh Tuan Luu, Guy Wolf, and Dominique Beaini. Recipe for a General, Powerful, Scalable Graph Transformer. *Advances in Neural Information Processing Systems*, 35, 2022. 14, 17

[76] Patrick Reiser, Marlen Neubert, André Eberhard, Luca Torresi, Chen Zhou, Chen Shao, Houssam Metni, Clint van Hoesel, Henrik Schopmans, Timo Sommer, et al. Graph neural networks for materials science and chemistry. *Communications Materials*, 3(1):93, 2022. 14

[77] Ivan Dario Jimenez Rodriguez, Aaron Ames, and Yisong Yue. Lyanet: A lyapunov framework for training neural odes. In *International conference on machine learning*, pages 18687–18703. PMLR, 2022. 14

[78] T Konstantin Rusch, Ben Chamberlain, James Rowbottom, Siddhartha Mishra, and Michael Bronstein. Graph-coupled oscillator networks. In *International Conference on Machine Learning*, pages 18888–18909. PMLR, 2022. 1, 4, 14, 17, 21, 25

[79] T. Konstantin Rusch, Michael M. Bronstein, and Siddhartha Mishra. A Survey on Oversmoothing in Graph Neural Networks. *arXiv preprint arXiv:2303.10993*, 2023. 1, 4

[80] Lars Ruthotto and Eldad Haber. Deep neural networks motivated by partial differential equations. *Journal of Mathematical Imaging and Vision*, 62:352–364, 2020. 14

[81] Hans M Senn and Walter Thiel. Qm/mm methods for biomolecular systems. *Angewandte Chemie International Edition*, 48(7):1198–1229, 2009. 1

[82] Dai Shi, Andi Han, Lequan Lin, Yi Guo, and Junbin Gao. Exposition on over-squashing problem on gnns: Current methods, benchmarks and challenges. *arXiv preprint arXiv:2311.07073*, 2023. 15

[83] Yunsheng Shi, Zhengjie Huang, Shikun Feng, Hui Zhong, Wenjing Wang, and Yu Sun. Masked label prediction: Unified message passing model for semi-supervised classification. In Zhi-Hua Zhou, editor, *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, pages 1548–1554. International Joint Conferences on Artificial Intelligence Organization, 8 2021. doi: 10.24963/ijcai.2021/214. URL https://doi.org/10.24963/ijcai.2021/214. Main Track. 17

[84] Behzad Shirzad, Amir M. Rahmani, and Marzieh Aghaei. Exphormer: Sparse attention for graphs. *Proceedings of the 40th International Conference on Machine Learning (ICML)*, 2023. 14, 17

[85] Jan Tönshoff, Martin Ritzert, Eran Rosenbluth, and Martin Grohe. Where did the gap go? reassessing the long-range graph benchmark. In *The Second Learning on Graphs Conference*, 2023. URL https://openreview.net/forum?id=rIUjwxc5lj. 23

[86] Jake Topping, Francesco Di Giovanni, Benjamin Paul Chamberlain, Xiaowen Dong, and Michael M. Bronstein. Understanding over-squashing and bottlenecks on graphs via curvature. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=7UmjRGzp-A. 1, 3, 14, 15, 18

[87] Csaba Toth, Darrick Lee, Celia Hacker, and Harald Oberhauser. Capturing graphs with hypo-elliptic diffusions. In *Advances in Neural Information Processing Systems*, 2022. 14

[88] A. Vaswani et al. Attention is all you need. *Advances in Neural Information Processing Systems*, 30, 2017. 17

[89] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=rJXMpikCZ. 4, 17, 25

[90] Chloe Wang, Oleksii Tsepa, Jun Ma, and Bo Wang. Graph-mamba: Towards long-range graph sequence modeling with selective state spaces. *arXiv preprint arXiv:2402.00789*, 2024. 17

[91] Kun Wang, Guibin Zhang, Xinnan Zhang, Junfeng Fang, Xun Wu, Guohao Li, Shirui Pan, Wei Huang, and Yuxuan Liang. The heterophilic snowflake hypothesis: Training and empowering gnns for heterophilic graphs. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '24, page 3164–3175, New York, NY, USA, 2024. Association for Computing Machinery. ISBN 9798400704901. doi: 10.1145/3637528.3671791. 14

[92] Xiyuan Wang and Muhan Zhang. How powerful are spectral graph neural networks. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 23341–23362. PMLR, 17–23 Jul 2022. URL https://proceedings.mlr.press/v162/wang22am.html. 17

[93] Yifei Wang, Yisen Wang, Jiansheng Yang, and Zhouchen Lin. Dissecting the Diffusion Process in Linear Graph Convolutional Networks. In *Advances in Neural Information Processing Systems*, volume 34, pages 5758–5769. Curran Associates, Inc., 2021. 14, 17

[94] Yuelin Wang, Kai Yi, Xinliang Liu, Yu Guang Wang, and Shi Jin. ACMP: Allen-cahn message passing with attractive and repulsive forces for graph neural networks. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=4fZc_79Lrqs. 14

[95] Peter G Wolynes. Recent successes of the energy landscape theory of protein folding and function. *Quarterly reviews of biophysics*, 38(4):405–410, 2005. 1

[96] Louis-Pascal Xhonneux, Meng Qu, and Jian Tang. Continuous graph neural networks. In *Proceedings of the 37th International Conference on Machine Learning*, pages 10432–10441, 2020. URL https://proceedings.mlr.press/v119/xhonneux20a.html. 14

[97] Jianwen Xie, Yuting Lu, Ruiqi Gao, Honglak Zhuang, and Ying Nian Wu. A theory of generative convnet. *International Conference on Machine Learning*, pages 2635–2644, 2016. 1, 14

[98] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=ryGs6iA5Km. 4, 13, 17, 25

[99] Zhitao Ying and Jure Leskovec. Graphormer: A transformer for graphs. *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2021. 14

[100] Manzil Zaheer, Guru prasad G. H., Lihong Wang, S. V. K. N. L. Wang, Yujia Li, Jakub Konečný, Shalmali Joshi, Danqi Chen, Jennifer R. R., Zhenyu Zhang, Shalini Devaraj, and Srinivas Narayanan. Bigbird: Transformers for longer sequences. *Proceedings of the 37th International Conference on Machine Learning (ICML)*, pages 12168–12178, 2020. URL https://arxiv.org/abs/2007.14062. 14

[101] Jiong Zhu, Yujun Yan, Lingxiao Zhao, Mark Heimann, Leman Akoglu, and Danai Koutra. Beyond homophily in graph neural networks: Current limitations and effective designs. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 7793–7804. Curran Associates, Inc., 2020. 17

[102] Jiong Zhu, Ryan A. Rossi, Anup Rao, Tung Mai, Nedim Lipka, Nesreen K. Ahmed, and Danai Koutra. Graph neural networks with heterophily. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(12):11168–11176, May 2021. doi: 10.1609/aaai.v35i12.17332. URL https://ojs.aaai.org/index.php/AAAI/article/view/17332. 17

[103] Juntang Zhuang, Nicha Dvornek, Xiaoxiao Li, and James S Duncan. Ordinary differential equations on graph networks. 2020. 14

A Mathematical Background

In this section, we provide a brief overview of Lyapunov stability theory, based on the classical treatment in Khalil and Grizzle [51], which underpins the design of our TANGO. This theory originates from control systems and differential equations, offering a principled way to assess whether trajectories of a dynamical system remain bounded and converge over time.

Continuous Dynamical Systems. Let $\mathbf{h}(t) \in \mathbb{R}^d$ denote the state of a dynamical system at time $t \geq 0$, and consider a first-order ODE:

$$\frac{d\mathbf{h}(t)}{dt} = F(\mathbf{h}(t)), \tag{5}$$

where $F : \mathbb{R}^d \to \mathbb{R}^d$ is a continuous vector field. A point $\mathbf{h}^*$ is called an *equilibrium* if $F(\mathbf{h}^*) = 0$.

Definition 1 (Lyapunov Function)**.** *Let $\mathbf{h}^* \in \mathbb{R}^d$ be an equilibrium of the system in Equation (5). A continuously differentiable function $V : \mathbb{R}^d \to \mathbb{R}$ is called a* Lyapunov function *around $\mathbf{h}^*$ if:*

1. *$V(\mathbf{h}) \geq 0$ for all $\mathbf{h}$ in a neighborhood of $\mathbf{h}^*$, and $V(\mathbf{h}^*) = 0$;*
2. *$\frac{d}{dt}V(\mathbf{h}(t)) = \nabla_{\mathbf{h}} V(\mathbf{h}(t))^\top F(\mathbf{h}(t)) \leq 0$ in that neighborhood.*

The first condition ensures that V is lower-bounded by 0, i.e., that value of the Lyapunov function, sometimes also referred to as *energy* is non-negative, and the second that V does not increase along trajectories of the system.

We now recall a classical [51] stability criterion for the dynamical system in Equation (5), based on the definition of a Lyapunov function, which we will later use to characterize the stability of our approach in Section 2.2.

Theorem 1 (Lyapunov Stability)**.** *Let $\mathbf{h}^*$ be an equilibrium of Equation (5) and let V be a Lyapunov function in a neighborhood $\mathcal{N}$ of $\mathbf{h}^*$. If $\frac{d}{dt}V(\mathbf{h}(t)) \leq 0$ in $\mathcal{N}$, then $\mathbf{h}^*$ is* Lyapunov stable*.*

B Implementing TANGO Graph Neural Networks

As outlined in Section 2.1, TANGO is defined by a continuous dynamical system. To obtain a GNN, we discretize Equation (1) with a forward Euler step—standard in GNNs [3, 15, 19, 30, 39]—yielding the layer

$$\mathbf{H}^{(\ell+1)} = \mathbf{H}^{(\ell)} + \epsilon\Big(-\alpha_{\mathcal{G}}(\mathbf{H}^{(\ell)})\,\nabla_{\mathbf{H}} V_{\mathcal{G}}(\mathbf{H}^{(\ell)}) + \beta_{\mathcal{G}}(\mathbf{H}^{(\ell)})\,T_{V_{\mathcal{G}}}(\mathbf{H}^{(\ell)})\Big), \tag{6}$$

for $\ell = 0, \ldots, L-1$, where $\epsilon > 0$ is the Euler step size, $\nabla_{\mathbf{H}} V_{\mathcal{G}}(\mathbf{H}^{(\ell)})$ is the gradient of the energy in Equation (8), and $\alpha_{\mathcal{G}} \geq 0,\ \beta_{\mathcal{G}}$ are learned scalars that balance descent and tangential terms predicted by GNNs described below.

Energy Function. Given features $\mathbf{H}^{(\ell)}$, we implement $V_{\mathcal{G}}$ via

$$\tilde{\mathbf{H}}^{(\ell)} = \sigma\Big(\textsc{EnergyGNN}(\mathbf{H}^{(\ell)}; \mathcal{G})\Big) \in \mathbb{R}^{n \times d}, \tag{7}$$

where EnergyGNN is a GNN (e.g., GatedGCN [11], GPS [74]) and σ is a pointwise nonlinearity. Per-node energy scores (MLP) are given by $\tilde{V}_{\mathcal{G}}(\tilde{\mathbf{H}}^{(\ell)}) = \text{MLP}_{\text{E}}(\tilde{\mathbf{H}}^{(\ell)}) \in \mathbb{R}^{n \times 1}$, and the graph-level energy is defined as:

$$V_{\mathcal{G}}(\mathbf{H}^{(\ell)}) = \frac{1}{n} \sum_{v \in \mathcal{V}} \tilde{V}_{\mathcal{G}}(\tilde{\mathbf{H}}^{(\ell)})_v^2 \in \mathbb{R}_{\geq 0}. \tag{8}$$

We also compute a bounded non-negative energy descent coefficient $\alpha_{\mathcal{G}}$ via global sum pooling [98] of $\tilde{\mathbf{H}}^{(\ell)}$, followed by an MLP and sigmoid:

$$\alpha_{\mathcal{G}}(\mathbf{H}^{(\ell)}) = \textsc{Sigmoid}\Big(\text{MLP}_\alpha\Big(\textsc{SumPool}(\tilde{\mathbf{H}}^{(\ell)})\Big)\Big). \tag{9}$$

Tangential Update. We compute $T_{V_{\mathcal{G}}}(\mathbf{H}^{(\ell)})$ with a dedicated GNN, TangentGNN. Given $\mathbf{H}^{(\ell)}$, it predicts a node-update direction

$$\mathbf{M}^{(\ell)} = \sigma\Big(\textsc{TangentGNN}(\mathbf{H}^{(\ell)}; \mathcal{G})\Big). \tag{10}$$

The energy-tangential component coefficient is then obtained via the orthogonal projection in Equation (2). The tangential term is scaled by

$$\beta_{\mathcal{G}}(\mathbf{H}^{(\ell)}) = \mathrm{MLP}_{\beta}\Big(\textsc{SumPool}(\mathbf{M}^{(\ell)})\Big)\,. \tag{11}$$

Notably, while the depths of TANGENTGNN and ENERGYGNN can be different, for simplicity of hyper-parameter tuning, throughout our experiments, we use the same number of layers for both.

C Related Work

Deep GNNs and Dynamical Systems. A growing body of work interprets GNN layers as iterative updates in a dynamical system, providing a principled framework to analyze stability, control diffusion, and inform architectural design. Poli et al. [73] introduced Graph Neural ODEs, inspired by neural ODEs [17, 80], modeling node feature evolution via continuous-depth ODEs aligned with graph structure, enabling adaptive computation and improved performance in dynamic settings. Similarly, Xhonneux et al. [96] proposed Continuous GNNs, where feature channels evolve by differential equations, mitigating over-smoothing via infinite-depth limits. Follow-up works such as GODE [103], GRAND [15], PDE-GCN$_{\mathrm{D}}$ [30], and DGC [93] view GNN layers as discrete integration steps of the heat equation to control oversmoothing [13, 68, 69]. Extensions like PDE-GCN$_{\mathrm{M}}$ [30] and GraphCON [78] add oscillatory components to preserve feature energy, while others leverage heat-kernel attention [21], anti-symmetry [39, 40], reaction-diffusion [19, 94], advection-reaction-diffusion [31] to enhance long-range or directional flow, and higher-order graph neuro ODE models [32]. A comprehensive overview is given in Han et al. [45]. Closely related, Di Giovanni et al. [23] interpret GNN layer updates as gradient flows of the Dirichlet energy, aligning message passing with energy minimization. In contrast, our TANGO learns a graph-adaptive, task-specific energy and introduces a novel descent mechanism combining energy gradients with a learnable tangential component, enabling more expressive dynamics than pure gradient flows.

Learning Energy Functions in Neural Networks. Energy-based models (EBMs) provide a flexible framework in deep learning by learning an energy function whose low-energy regions correspond to areas with high probability for the data. They have been widely used in generative tasks such as image synthesis [25, 42, 57, 97] and graph generation [60, 76]. In contrast to these typically unsupervised settings, our work focuses on learning a *task-driven* energy function tailored to predictive objectives like node or graph classification. Here, inference corresponds to descending the learned energy landscape, whose minima align with correct outputs. Relatedly, Lyapunov functions—classical tools from control theory—have been used in neural networks to ensure stable learning or inference dynamics, e.g., by enforcing stability in Neural ODEs [77] or GNN-based controllers [33]. However, such approaches typically assume a fixed or implicit energy function rather than learning one. Our method, TANGO, bridges and extends these perspectives by learning a graph-adaptive, task-specific energy and introducing a novel optimization scheme. Crucially, our TANGO incorporates a learnable tangential component that accelerates energy minimization and enhances performance in graph learning tasks.

Oversquashing in Graph Learning. Graph neural networks (GNNs) typically operate through message-passing mechanisms, aggregating information from local neighborhoods. While effective in capturing short-range dependencies, this design often leads to *oversquashing*, a phenomenon where signals from distant nodes are compressed into fixed-size representations, impeding the flow of long-range information [2, 22, 86]. This limitation poses a challenge in domains that demand rich global context, such as bioinformatics [4, 28] and heterophilic graphs [61, 91]. A range of strategies have been proposed to mitigate oversquashing. *Graph rewiring* approaches, such as SDRF [86], densify the graph to enhance connectivity prior to training. In contrast, methods like GRAND [15], BLEND [14], and DRew [43] adjust the graph structure dynamically based on node features. *Transformer-based models* offer another promising route by leveraging global attention to enable direct, long-range message passing. Examples include SAN [56], Graphormer [99], and GPS [75], which incorporate positional encodings, such as Laplacian eigenvectors [29] and random walk structural embeddings [27] to preserve structural identity. However, the quadratic complexity of full attention in these models raises scalability concerns, motivating interest in sparse attention mechanisms [20, 84, 100]. An alternative line of work explores *non-local dynamics* to enhance expressivity without relying solely on attention. FLODE [64] employs fractional graph operators, QDC [63] uses quantum diffusion processes, and G2TN [87] models explicit diffusion paths to propagate information more

effectively. While these approaches address the oversquashing bottleneck, they often come with increased computational demands due to dense propagation operators. For a broader overview of these techniques, see Shi et al. [82]. We note that the challenge of modeling long-range dependencies also arises in other domains, such as sequential architectures [41, 47].

Optimization Techniques. The formulation of TANGO draws parallel with concepts that have been explored in the optimization literature, particularly in the design of dynamical systems that balance expressivity and convergence. While traditional gradient descent provides a robust and interpretable mechanism for minimizing energy functions, its convergence rate can be limited in poorly conditioned settings [9, 67], which frequently arise in graph-based problems due to structural bottlenecks [2, 86]. Second-order approaches, such as Newton's method, are known to accelerate convergence by incorporating curvature information, albeit at increased computational cost. The combination of energy gradient descent and a learned tangential component in TANGO suggests a learnable departure from purely first-order schemes. Rather than explicitly computing or approximating the Hessian, our framework enables the model to learn corrective update directions that are orthogonal to the descent path. This design implicitly aligns with the motivations behind quasi-Newton techniques like conjugate gradients and LBFGS [67], which aim to improve convergence by leveraging directional information that complements the gradient. From this perspective, TANGO can be viewed as embedding optimization-inspired dynamics within graph learning frameworks. This is particularly relevant in scenarios affected by oversquashing [22], where effective feature transmission often requires departing from strictly local, gradient-driven updates. By allowing energy-preserving tangential flows, TANGO introduces flexibility reminiscent of structured optimization methods, adapted to the graph learning domain.

D Proofs of Theoretical Results

In this section, we restate the theoretical results from Section 2.2 and provide their proofs. As in the main text, we assume the following throughout: (i) the input graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is connected; (ii) the energy function $V_{\mathcal{G}}(\mathbf{H}(t))$ is twice differentiable and bounded from below. For simplicity of notation, throughout this section, we omit the time or layer scripts and use the term $\mathbf{H}$ to denote node features when possible.

Proposition 1 (Energy Dissipation)**.** *Suppose* $\alpha_{\mathcal{G}} \geq 0$ *and* $\|\nabla_{\mathbf{H}} V_{\mathcal{G}}(\mathbf{H})\|^2 > 0$*. Then the energy* $V_{\mathcal{G}}(\mathbf{H})$ *is non-increasing along trajectories of Equation* (1)*. Specifically,*

$$\begin{aligned} \frac{d}{dt} V_{\mathcal{G}}(\mathbf{H}) &= -\alpha_{\mathcal{G}}(\mathbf{H}) \left\|\nabla_{\mathbf{H}} V_{\mathcal{G}}(\mathbf{H})\right\|^2 + \beta_{\mathcal{G}}(\mathbf{H}) \langle T_{V_{\mathcal{G}}}(\mathbf{H}), \nabla_{\mathbf{H}} V_{\mathcal{G}}(\mathbf{H}) \rangle \\ &= -\alpha_{\mathcal{G}}(\mathbf{H}) \left\|\nabla_{\mathbf{H}} V_{\mathcal{G}}(\mathbf{H})\right\|^2 \leq 0. \end{aligned}$$

Proof. By the chain rule,

$$\frac{d}{dt} V_{\mathcal{G}}(\mathbf{H}) = \left\langle \nabla_{\mathbf{H}} V_{\mathcal{G}}(\mathbf{H}),\, \frac{d\mathbf{H}}{dt} \right\rangle.$$

Substituting the dynamics of Equation (1):

$$\begin{aligned} \frac{d}{dt} V_{\mathcal{G}}(\mathbf{H}) &= \langle \nabla_{\mathbf{H}} V_{\mathcal{G}}(\mathbf{H}),\, -\alpha_{\mathcal{G}}(\mathbf{H})\, \nabla_{\mathbf{H}} V_{\mathcal{G}}(\mathbf{H}) + \beta_{\mathcal{G}}(\mathbf{H})\, T_{V_{\mathcal{G}}}(\mathbf{H}) \rangle \\ &= -\alpha_{\mathcal{G}}(\mathbf{H})\, \left\|\nabla_{\mathbf{H}} V_{\mathcal{G}}(\mathbf{H})\right\|^2 + \beta_{\mathcal{G}}(\mathbf{H})\, \langle T_{V_{\mathcal{G}}}(\mathbf{H}),\, \nabla_{\mathbf{H}} V_{\mathcal{G}}(\mathbf{H}) \rangle. \end{aligned}$$

As discussed in Section 2, we have by design, that

$$\langle T_{V_{\mathcal{G}}}(\mathbf{H}),\, \nabla_{\mathbf{H}} V_{\mathcal{G}}(\mathbf{H}) \rangle = 0.$$

Therefore,

$$\frac{d}{dt} V_{\mathcal{G}}(\mathbf{H}) = -\alpha_{\mathcal{G}}(\mathbf{H})\, \left\|\nabla_{\mathbf{H}} V_{\mathcal{G}}(\mathbf{H})\right\|^2.$$

Because $\alpha_{\mathcal{G}}(\mathbf{H}) \geq 0$ by design, the energy is non-increasing, and assuming $\alpha_{\mathcal{G}}(\mathbf{H}) > 0$, the system is dissipative, i.e., its energy is decreasing. □

Proposition 2 (TANGO can Evolve Features in Flat Energy Landscapes)**.** *Suppose* $\nabla_{\mathbf{H}} V_{\mathcal{G}}(\mathbf{H}) = 0$*, and* $T_{V_{\mathcal{G}}}(\mathbf{H}) \neq 0$*, then the* TANGO *flow in Equation* (1) *reads:*

$$\frac{d\mathbf{H}}{dt} = \beta_{\mathcal{G}}(\mathbf{H}) T_{V_{\mathcal{G}}}(\mathbf{H}).$$

This implies that in contrast *to gradient flows, the dynamics of* TANGO *can evolve even in regions where the energy landscape is flat.*

Proof. Because $\nabla_{\mathbf{H}} V_{\mathcal{G}}(\mathbf{H}) = 0$, the first term in Equation (1) vanishes, and the TANGO dynamical system reads:

$$\frac{d\mathbf{H}}{dt} = \beta_{\mathcal{G}}(\mathbf{H}) T_{V_{\mathcal{G}}}(\mathbf{H}),$$

Assuming that $T_{V_{\mathcal{G}}}(\mathbf{H}) \neq 0$, TANGO can continue evolving node features also in cases where $\nabla_{\mathbf{H}} V_{\mathcal{G}}(\mathbf{H}) = 0$, i.e., where the energy landscape is flat. □

Proposition 3 (Convergence of Gradient Descent of a Scalar Function, Nocedal and Wright [67])**.** *Let* $V_{\mathcal{G}}(\cdot)$ *be a scalar function and let* $\mathbf{H}^{(\ell+1)} = \mathbf{H}^{(\ell)} - \alpha_{\mathcal{G}}^{(\ell)}(\mathbf{H}^{(\ell)}) \nabla_{\mathbf{H}} V_{\mathcal{G}}(\mathbf{H}^{(l)})$ *be a gradient-descent iteration of the energy* $V_{\mathcal{G}}(\cdot)$*. Then, a linear convergence is obtained, with convergence rate:*

$$r = \frac{\lambda_{\max} - \lambda_{\min}}{\lambda_{\max} + \lambda_{\min}},$$

where $\lambda_{\max}$ *is the maximal eigenvalue, and in the case of problems that involve the graph Laplacian,* $\lambda_{\min}$ *is the second minimal eigenvalue, i.e., the first non-zero eigenvalue of the Hessian of* $V_{\mathcal{G}}(\cdot)$*.*

Proposition 4 (TANGO can learn a Quadratic Convergence Direction)**.** *Assume for simplicity that* $\beta_{\mathcal{G}} = 1$*, and that the Hessian of* $V_{\mathcal{G}}$ *is invertible. Let* $\mathbf{D} = \alpha_{\mathcal{G}}(\mathbf{H}^{(\ell)}) \nabla_{\mathbf{H}} V_{\mathcal{G}}(\mathbf{H}^{(\ell)}) + T_{V_{\mathcal{G}}}(\mathbf{H}^{(\ell)})$ *with* $\left\langle T_{V_{\mathcal{G}}}(\mathbf{H}^{(\ell)}), \widehat{\nabla}_{\mathbf{H}} V_{\mathcal{G}}(\mathbf{H}^{(\ell)}) \right\rangle = 0$*. Then, it is possible to learn a direction* $T_{V_{\mathcal{G}}}(\mathbf{H}^{(\ell)})$ *and a step size* $\alpha_{\mathcal{G}}$ *such that* $\mathbf{D}$ *is the Newton direction,* $\mathbf{N} = (\nabla^2 V_{\mathcal{G}})^{-1} \nabla V_{\mathcal{G}}$*.*

Proof. We aim to construct a direction $\mathbf{D} = \alpha_{\mathcal{G}}(\mathbf{H})\, \nabla_{\mathbf{H}} V_{\mathcal{G}}(\mathbf{H}) + T_{V_{\mathcal{G}}}(\mathbf{H})$ that matches the Newton direction:

$$\mathbf{N} = \left(\nabla^2_{\mathbf{H}} V_{\mathcal{G}}(\mathbf{H})\right)^{-1} \nabla_{\mathbf{H}} V_{\mathcal{G}}(\mathbf{H}).$$

Recall that by design, we have that $T_{V_{\mathcal{G}}}(\mathbf{H})$ is orthogonal to the energy gradient, i.e., $\langle T_{V_{\mathcal{G}}}(\mathbf{H}),\, \nabla_{\mathbf{H}} V_{\mathcal{G}}(\mathbf{H}) \rangle = 0$. Then, we can express a Newton direction by the decomposition:

$$\mathbf{N} = \alpha_{\mathcal{G}}(\mathbf{H})\, \nabla_{\mathbf{H}} V_{\mathcal{G}}(\mathbf{H}) + T_{V_{\mathcal{G}}}(\mathbf{H}).$$

Solving for the orthogonal component yields:

$$T_{V_{\mathcal{G}}}(\mathbf{H}) = \mathbf{N} - \alpha_{\mathcal{G}}(\mathbf{H})\, \nabla_{\mathbf{H}} V_{\mathcal{G}}(\mathbf{H}).$$

To enforce orthogonality, we require:

$$\langle \mathbf{N} - \alpha_{\mathcal{G}}(\mathbf{H})\, \nabla_{\mathbf{H}} V_{\mathcal{G}}(\mathbf{H}),\, \nabla_{\mathbf{H}} V_{\mathcal{G}}(\mathbf{H}) \rangle = 0.$$

Expanding and simplifying, we find:

$$\langle \mathbf{N},\, \nabla_{\mathbf{H}} V_{\mathcal{G}}(\mathbf{H}) \rangle - \alpha_{\mathcal{G}}(\mathbf{H})\, \left\| \nabla_{\mathbf{H}} V_{\mathcal{G}}(\mathbf{H}) \right\|^2 = 0,$$

and the optimal step size is given by:

$$\alpha_{\mathcal{G}}(\mathbf{H}) = \frac{\langle \mathbf{N},\, \nabla_{\mathbf{H}} V_{\mathcal{G}}(\mathbf{H}) \rangle}{\left\| \nabla_{\mathbf{H}} V_{\mathcal{G}}(\mathbf{H}) \right\|^2},$$

showing that it is possible to learn a Newton direction, i.e., a quadratic energy convergence direction. □

E Complexity and Runtimes

Complexity. Each step of TANGO requires computing the gradient of the learned energy function $V_{\mathcal{G}}(\mathbf{H}^{(\ell)})$, that is defined in Equation (8). This involves two main operations: (i) forward and backward passes through the energy network ENERGYGNN, which contains L_{energy} message-passing layers and an MLP; and (ii) automatic differentiation to compute $\nabla_{\mathbf{H}} V_{\mathcal{G}}(\mathbf{H}^{(\ell)})$ with respect to the input node features. In parallel, the tangential flow direction $T_{V_{\mathcal{G}}}(\mathbf{H}^{(\ell)})$ is obtained by projecting the vector field $\mathbf{M}^{(\ell)}$ computed by a separate TANGENTGNN with L_{tangent} layers onto the orthogonal complement of the normalized energy gradient, as shown in Equation (2). This projection is of computational cost of $O(nd)$ per step, where $n = |\mathcal{V}|$ and d is the feature dimensionality. In addition, scalar coefficients $\alpha_{\mathcal{G}}$ and $\beta_{\mathcal{G}}$ are computed from pooled node features using MLPs (Equations (9) and (11)). Assuming both ENERGYGNN and TANGENTGNN are message-passing architectures with linear complexity in the number of nodes and edges, and setting $L_{\text{energy}} = L_{\text{tangent}}$, the total complexity per layer becomes $O(L_{\text{gnn}} \cdot (n+m) \cdot d)$, where L_{gnn} is the number of GNN layers used in each subnetwork and $m = |\mathcal{E}|$ is the number of edges. Unrolling the dynamics over L steps, the overall computational complexity of TANGO is:

$$O\left(L \cdot L_{\text{gnn}} \cdot (|\mathcal{V}| + |\mathcal{E}|) \cdot d\right).$$

Runtimes. We measure the runtimes of our TANGO using two backbones, GatedGCN and GPS, and compare it with the baseline backbone runtimes. In addition, we consider other methods like FAGCN [8] and CO-GNN [34] for a broad comparison of the runtimes of TANGO. For reference, we also refer to Table 7, where we compare the obtained downstream performance, which shows in many cases significant improvement using our TANGO variants compared with other considered methods. We measure the runtimes on the Questions dataset, using the same major hyperparameters for all methods (256 channels, 8 layers) to ensure fairness. The measurements were conducted on an NVIDIA RTX6000 Ada GPU with 48GB of memory.

Table 3: Training runtimes (milliseconds per epoch) on the Questions dataset using an 8-layer network with 256 channels on an NVIDIA RTX6000 Ada GPU.

Method	GCN	CO-GNN	FAGCN	GatedGCN	GAT	GPS(GatedGCN)	$\text{TANGO}_{\text{GatedGCN}}$	$\text{TANGO}_{\text{GPS}}$
Runtime (ms/epoch)	69.77	210.32	103.94	129.92	112.40	429.08	184.98	694.27

F Experimental Details

In this section, we provide additional experimental details.

Computational Resources. Our experiments are run on NVIDIA RTX6000 Ada with 48GB of memory. Our code is implemented in PyTorch [70], and will be publicly released upon acceptance.

Baselines. We consider different classical and state-of-the-art GNN baselines. Specifically:

- Classical MPNNs, i.e., GCN [52], GraphSAGE [44], GAT [89], GatedGCN [11], GIN [98], GINE [48], GCNII [16], and CoGNN [34];
- Heterophily-specific models, i.e., H2GCN [101], CPGNN [102], FAGCN [8], GPR-GNN [18], FSGNN [65], GloGNN [58], GBK-GNN [24], and JacobiConv [92];
- DE-DGNs, i.e., DGC [93], GRAND [15], GraphCON [78], A-DGN [39], and SWAN [40];
- Graph Transformers, i.e., Transformer [26, 88], GT [83], SAN [55], GPS [75], GOAT [53], and Exphormer [84];
- Higher-Order DGNs, i.e., DIGL [37], MixHop [1], and DRew [43].
- SSM-based GNN, i.e., Graph-Mamba [90], GMN [5], and GPS+Mamba [5]

F.1 Synthetic Example from Figure 2

In the synthetic example in Figure 2, we demonstrate the effectiveness of TANGO in overcoming the oversquashing issue in GNNs. To do that, we consider a Barbell graph, where all node features are set to 0, besides the left-most node in the graph, which is set to 1, as shown in Figure 2(a). The goal is to

allow the information to propagate through all nodes effectively. We do this by considering a gradient flow process of the Dirichlet energy using 50 layers (steps), as shown in Figure 2(b), where it is noticeable that the information is now flowing to the right part in the graph, because of the bottleneck between the two cliques. However, as we show in Figure 2(c), by considering our TANGO, which utilizes both an energy flow as well as a tangential flow, it is possible to effectively propagate the information through all the nodes in the graphs.

F.2 Graph Property Prediction

Dataset. We construct our benchmark following the protocol introduced by Gravina et al. [39]. Graph instances are synthetically generated from a variety of canonical topologies, including Erdős–Rényi, Barabasi-Albert, caveman, tree, and grid models. Each graph consists of 25 to 35 nodes, with node features initialized as random identifiers sampled uniformly from the interval $[0, 1)$. The prediction targets encompass several structural tasks: computing the shortest paths from a source node, estimating node eccentricity, and determining graph diameter. The complete dataset contains 7,040 graphs, split into 5,120 for training, 640 for validation, and 1,280 for testing. These tasks inherently demand capturing long-range dependencies, as they involve global graph computations such as shortest path inference. As highlighted in Gravina et al. [39], traditional algorithms like Bellman-Ford or Dijkstra's method require multiple rounds of message propagation, which motivates the need for expressive graph models. The benchmark graph families, such as caveman, tree, line, star, caterpillar, and lobster, frequently include structural bottlenecks that are known to induce oversquashing effects [86], posing additional challenges for message-passing-based GNNs.

Experimental Setup. We adopt the same evaluation framework as Gravina et al. [39], including datasets, training routines, and hyperparameter spaces. Model training is conducted using the Adam optimizer for up to 1500 epochs, with early stopping triggered after 100 consecutive epochs of no improvement on the validation Mean Squared Error (MSE). Hyperparameters are selected via grid search, and performance is averaged over 4 independent runs with different random seeds for weight initialization. A summary of the hyperparameter grid used in our experiments is provided in Table 5.

F.3 Graph Benchmarks from Dwivedi et al. [29]

Dataset. To comprehensively assess the capabilities of TANGO, we evaluate its performance on a diverse set of graph learning benchmarks curated by Dwivedi et al. [29]. The benchmark suite includes: *ZINC-12k*, a molecular regression dataset containing chemical compounds, where the goal is to predict the constrained solubility of each molecule. Graphs represent molecular structures, with atoms as nodes and chemical bonds as edges. Node and edge features encode atom types and bond types, respectively. *MNIST* and *CIFAR-10* superpixels are graph-structured versions of standard image classification datasets, where images are converted into sparse graphs of superpixels. Each superpixel forms a node, and edges are based on spatial adjacency. The tasks involve classifying digits (MNIST) and natural objects (CIFAR-10) based on graph-structured representations. *CLUSTER* and *PATTERN* are synthetic datasets designed to assess the relational inductive biases of graph neural networks. Both datasets are generated from a set of stochastic block models (SBMs). In *CLUSTER*, the task is to group nodes by community, while *PATTERN* involves identifying specific structural patterns within each graph. These datasets span a variety of domains: chemical, image, and synthetic graphs, and are commonly used to benchmark architectural innovations in GNNs [62]. We follow the official training, validation, and test splits provided by Dwivedi et al. [29], ensuring consistency in evaluation across models.

Experimental Setup. We adhere to the training and evaluation protocol established in Dwivedi et al. [29]. For each dataset, we perform hyperparameter tuning via grid search, optimizing the corresponding evaluation metrics: Mean Absolute Error (MAE) for *ZINC-12k*, and classification accuracy for the remaining tasks. We use the AdamW optimizer and train all models for up to 300 epochs, with early stopping based on validation performance. To ensure comparability with prior work, we respect the same parameter budgets used in the original benchmark and maintain the architectural constraints defined for fair evaluation. Each configuration is trained with three random seeds, and we report the average and standard deviation of the results. Hyperparameter ranges used in this set of experiments are summarized in Table 5.

F.4 Long Range Graph Benchmark

Dataset. To evaluate model performance on real-world graphs with significant long-range dependencies, we utilize the *Peptides-func* and *Peptides-struct* benchmarks introduced in Dwivedi et al. [28]. These datasets represent peptide molecules as graphs, where nodes correspond to heavy (non-hydrogen) atoms, and edges denote chemical bonds. *Peptides-func* is a multi-label classification task with 10 functional categories, including antibacterial, antiviral, and signaling-related properties. In contrast, *Peptides-struct* focuses on regression, targeting physical and geometric attributes such as molecular inertia (weighted by atomic mass and valence), atom pair distance extremes, sphericity, and average deviation from a best-fit plane. Together, the two datasets comprise 15,535 peptide graphs and roughly 2.3 million nodes. We adopt the official train/validation/test partitions from Dwivedi et al. [28] and report mean and standard deviation across three different random seeds for each experiment.

Experimental Setup. We follow the evaluation protocol established in Dwivedi et al. [28], including dataset usage, training strategy, and model capacity constraints. Hyperparameter tuning is carried out via grid search, optimizing for Average Precision (AP) in the classification task and Mean Absolute Error (MAE) in the regression task. All models are trained using the AdamW optimizer for up to 300 epochs, with early stopping based on validation performance. To ensure fairness and comparability, all models adhere to the 500K parameter limit, in line with the settings of Dwivedi et al. [28] and Gutteridge et al. [43]. Each configuration is run three times with different weight initializations, and results are averaged. Details of the hyperparameter ranges considered can be found in Table 5.

F.5 Heterophilic Node Classification

Dataset. For evaluating performance in heterophilic graph settings, we consider five benchmark tasks introduced by Platonov et al. [72]: *Roman-Empire*, *Amazon-Ratings*, *Minesweeper*, *Tolokers*, and *Questions*. These datasets span a diverse range of domains and graph topologies. *Roman-Empire* is constructed from the Wikipedia article on the Roman Empire, where nodes represent words and edges capture either sequential adjacency or syntactic relations. The task is node classification with 18 syntactic categories, and the underlying graph is sparse and chain-structured, suggesting the presence of long-range dependencies. *Amazon-Ratings* originates from Amazon's product co-purchasing graph. Nodes correspond to products, linked if they are frequently bought together. The classification task involves predicting discretized average product ratings (five classes), with node features derived from fastText embeddings of product descriptions. *Minesweeper* is a synthetic dataset modeled as a 100×100 grid. Nodes represent individual cells, with edges connecting adjacent cells. A random 20% of nodes are labeled as mines, and the objective is to classify mine-containing cells based on one-hot features that encode the number of neighboring mines. *Tolokers* is based on the Toloka crowdsourcing platform [59], where each node is a worker (toloker), and edges indicate co-participation on the same project. The task involves binary classification to detect whether a worker has been banned, using node features from user profiles and performance metrics. *Questions* draws from user interaction data on Yandex Q, a question-answering forum. Nodes represent users, and edges capture answering interactions. The goal is to identify users who remain active, with input features derived from user-provided descriptions. A summary of dataset statistics is provided in Table 4.

Table 4: Statistics of the heterophilic node classification datasets.

	Roman-empire	Amazon-ratings	Minesweeper	Tolokers	Questions
N. nodes	22,662	24,492	10,000	11,758	48,921
N. edges	32,927	93,050	39,402	519,000	153,540
Avg degree	2.91	7.60	7.88	88.28	6.28
Diameter	6,824	46	99	11	16
Node features	300	300	7	10	301
Classes	18	5	2	2	2
Edge homophily	0.05	0.38	0.68	0.59	0.84

Experimental Setup. Our experimental procedure aligns with that of Freitas et al. [35] and Platonov et al. [72]. We conduct a grid search to optimize model performance, using classification accuracy for the *Roman-Empire* and *Amazon-Ratings* tasks, and ROC-AUC for *Minesweeper*, *Tolokers*, and *Questions*. Each model is trained using the AdamW optimizer for a maximum of 300 epochs. Our experiments follow the official dataset splits provided by Platonov et al. [72]. For each model

configuration, we perform multiple training runs with different random seeds and report the mean and standard deviation of the results. The hyperparameter grid explored in these experiments is summarized in Table 5.

F.6 Hyperparameters

In Table 5, we summarize the hyperparameter grids used for tuning our TANGO across different benchmarks. Alongside standard training hyperparameters such as learning rate, weight decay, and batch size, our method introduces several additional components. These include the number of unrolled steps L (corresponding to the depth of the energy-based dynamics), the hidden dimension d of node features, and the number of message-passing layers L_{gnn} used within the internal ENERGYGNN and TANGENTGNN modules. In all experiments, we share the architecture depth between ENERGYGNN and TANGENTGNN. We also tune the step size ϵ used in the forward Euler update (Equation (6)), which controls the integration scale of the continuous dynamics. We explore multiple values of L to assess how the number of dynamical steps impacts long-range propagation across different tasks. Details of the complete hyperparameter grid can be found in Table 5.

Table 5: Hyperparameter grids used during model selection for the different benchmark categories: *GraphPropPred* (Diameter, SSSP, Eccentricity), *LRGB* (Peptides-func/struct), *Graph Benchmarks* (ZINC-12k, MNIST, CIFAR-10, CLUSTER, PATTERN), and *Node Classification* (Roman-Empire, Amazon-Ratings, Minesweeper, Tolokers, Questions).

Hyperparameter	***GraphPropPred***	***LRGB***	***Graph Benchmarks***	***Node Classification***
Unrolled steps L	{1,5,10,20}	{2,4,8,16,32}	{2,4,8,16,32}	{2,4,8,16,32}
GNN layers L_{gnn}	{1,2,4,8,16}	{1,2,4,8,16}	{1,2,4,8,16}	{1,2,4,8,16}
Feature dimension d	{10, 20, 30}	{64, 128,256}	{64, 128, 256}	{64, 128, 256}
Step size ϵ	{0.001, 0.1, 1.0}	{0.001, 0.1, 1.0}	{0.001, 0.1, 1.0}	{0.001, 0.1, 1.0}
Learning rate	{1e-3, 1e-4}	{1e-3, 1e-4}	{1e-3, 1e-4}	{1e-3, 1e-4}
Weight decay	{0,1e-6, 1e-5}	{0, 1e-6, 1e-5}	{0, 1e-6, 1e-5}	{0, 1e-6, 1e-5}
Activation function (σ)	ReLU	ELU, GELU, ReLU	ELU, GELU, ReLU	ELU, GELU, ReLU
Batch size	{32,64,128}	{32,64,128}	{32, 64,128}	N/A

G Additional Results and Comparisons

G.1 Long-Range Benchmark

We assess the performance of our method on the real-world long-range graph benchmark (LRGB) from [28], focusing on the *Peptides-func* and *Peptides-struct* datasets. We follow the experimental setting in [28], including the 500K parameter budget. All transformer baselines include positional and structural encodings. TANGO does not use additional encodings. The datasets consist of large molecular graphs derived from peptides, where the structure and function of a peptide depend on interactions between distant parts of the graph. Therefore, relying on short-range interactions, such as those captured by local message passing in GNNs, may not be sufficient to excel at this task.

Table 6 provides a comparison of our TANGO model with a wide range of baselines. A broader comparison is presented in Table 10. The results indicate that TANGO outperforms standard MPNNs, transformer-based GNNs, DE-GNNs, and most Multi-hop GNNs.

G.2 Heterophilic Node Classification

We report and compare the performance of our TANGO with other recent benchmarks on the heterophilic node classification datasets from Platonov et al. [72], in Table 7. As can be seen from the Table, TANGO offers strong performance that is similar or better than recent state-of-the-art methods, further demonstrating its effectiveness.

G.3 Additional Comparisons

The comparisons made in Section 3 offer a focused comparison with directly related methods as well as baseline backbones. In addition to that, we now provide a more comprehensive comparison in Table 10 and Table 11, to further facilitate a comprehensive comparison with recent methods. As can

Table 6: Results for Peptides-func and Peptides-struct (3 training seeds). The **first**, **second**, and **third** best scores are colored.

Model	Peptides-func AP ↑	Peptides-struct MAE ↓
MPNNs		
GCN [52]	$59.30_{\pm 0.23}$	$0.3496_{\pm 0.0013}$
GINE [29]	$54.98_{\pm 0.79}$	$0.3547_{\pm 0.0045}$
GCNII [16]	$55.43_{\pm 0.78}$	$0.3471_{\pm 0.0010}$
GatedGCN [11]	$58.64_{\pm 0.77}$	$0.3420_{\pm 0.0013}$
Multi-hop GNNs		
DIGL+MPNN+LapPE [37]	$68.30_{\pm 0.26}$	$0.2616_{\pm 0.0018}$
MixHop-GCN+LapPE [1]	$68.43_{\pm 0.49}$	$0.2614_{\pm 0.0023}$
DRew-GCN+LapPE [43]	**71.50**$_{\pm 0.44}$	$0.2536_{\pm 0.0015}$
Transformers		
Transformer+LapPE [29]	$63.26_{\pm 1.26}$	$0.2529_{\pm 0.0016}$
SAN+LapPE [54]	$63.84_{\pm 1.21}$	$0.2683_{\pm 0.0043}$
GPS+LapPE [74]	$65.35_{\pm 0.41}$	$0.2500_{\pm 0.0005}$
DE-GNNs		
GRAND [15]	$57.89_{\pm 0.62}$	$0.3418_{\pm 0.0015}$
GraphCON [78]	$60.22_{\pm 0.68}$	$0.2778_{\pm 0.0018}$
A-DGN [39]	$59.75_{\pm 0.44}$	$0.2874_{\pm 0.0021}$
SWAN [40]	$67.51_{\pm 0.39}$	**0.2485**$_{\pm 0.0009}$
PH-DGN [46]	**70.12**$_{\pm 0.45}$	**0.2465**$_{\pm 0.0020}$
Ours		
TANGO$_{\text{GATEDGCN}}$	$68.92_{\pm 0.40}$	$0.2451_{\pm 0.0006}$
TANGO$_{\text{GPS}}$	**70.21**$_{\pm 0.43}$	**0.2422**$_{\pm 0.0014}$

be seen, also under these comparisons, our TANGO offers strong performance. We also report the results for Table 2 without applying log_{10} in Table 12

G.4 Ablation Study

Setup. We conduct two key ablation studies to better understand the contributions of the energy function and the tangential flow in TANGO. Specifically, we aim to answer the following questions:

(i) *Does downstream performance benefit from incorporating a tangential term even when the underlying GNN is not the gradient of an energy function?*

(ii) *Is the observed improvement due to the tangential nature of the added component, or simply due to additional parameters and network?*

To address these questions, we design two controlled experiments. For comprehensive coverage, we evaluate one representative dataset from each benchmark group: ZINC-12k, Roman-empire, Peptides-func, and Diameter. All experiments are run with two backbone architectures, GatedGCN and GPS. For reference, we also report the performance of the original backbones.

Results. For ablation (i), we compare TANGO against a variant we call TANGO-NON-ENERGY, in which the gradient-based energy descent term $\nabla_{\mathbf{H}} V_{\mathcal{G}}(\mathbf{H}^{(\ell)})$ in Equation (6) is replaced by intermediate node features from the same GNN backbone, as detailed in Equation (7). These features are computed using the same architecture but are not guaranteed to correspond to the gradient of any scalar energy function. This setup ensures fairness in capacity while removing the energy-based structure. As shown in Table 8, although both variants benefit from the inclusion of the tangential component, the full TANGO consistently outperforms TANGO-NON-ENERGY, confirming that leveraging a valid energy gradient contributes meaningfully to downstream performance.

For ablation (ii), we isolate the effect of the tangential nature of the added direction. In this variant, denoted TANGO-NON-TANGENT, we use the same output from the tangential network as in Equation (10) but omit the orthogonal projection step defined in Equation (2). Thus, while we still introduce an additional GNN term into the dynamics, it is not explicitly orthogonal to the energy gradient. Our results in Table 9 show that while this variant improves the performance compared with the baseline backbone, it also results in a drop in performance compared to the full TANGO. This highlights the importance of the tangential constraint, and its contribution towards improving the utilization of the learned energy function, as discussed in Section 2.2. Together, these ablations underscore the importance of both components in our design: (i) the principled learned energy descent, and (ii) the structured tangential update, as crucial for effective and flexible feature evolution.

Table 7: Mean test set score and std averaged over the splits from Platonov et al. [72]. **First**, **second**, and **third** best results for each task are color-coded. We mark each method once – if two variants are among the leading methods, we mark the best-performing variant.

Model	Roman-empire Acc ↑	Amazon-ratings Acc ↑	Minesweeper AUC ↑	Tolokers AUC ↑	Questions AUC ↑
MPNNs					
GAT	80.87±0.30	49.09±0.63	92.01±0.68	83.70±0.47	77.43±1.20
GAT-sep	88.75±0.41	52.70±0.62	**93.91**±0.35	83.78±0.43	76.79±0.71
Gated-GCN	74.46±0.54	43.00±0.32	87.54±1.22	77.31±1.14	76.61±1.13
GCN	73.69±0.74	48.70±0.63	89.75±0.52	83.64±0.67	76.09±1.27
CO-GNN(Σ, Σ)	**91.57**±0.32	51.28±0.56	95.09±1.18	83.36±0.89	**80.02**±0.86
CO-GNN(μ, μ)	91.37±0.35	**54.17**±0.37	**97.31**±0.41	**84.45**±1.17	76.54±0.95
SAGE	85.74±0.67	**53.63**±0.39	93.51±0.57	82.43±0.44	76.44±0.62
Graph Transformers					
Exphormer	**89.03**±0.37	53.51±0.46	90.74±0.53	83.77±0.78	73.94±1.06
NAGphormer	74.34±0.77	51.26±0.72	84.19±0.66	78.32±0.95	68.17±1.53
GOAT	71.59±1.25	44.61±0.50	81.09±1.02	83.11±1.04	75.76±1.66
$\text{GPS}_{\text{GAT+Performer}}$ (RWSE)	87.04±0.58	49.92±0.68	91.08±0.58	**84.38**±0.91	77.14±1.49
GT	86.51±0.73	51.17±0.66	91.85±0.76	83.23±0.64	77.95±0.68
GT-sep	87.32±0.39	52.18±0.80	92.29±0.47	82.52±0.92	78.05±0.93
Heterophily-Designated GNNs					
FAGCN	65.22±0.56	44.12±0.30	88.17±0.73	77.75±1.05	77.24±1.26
FSGNN	79.92±0.56	52.74±0.83	90.08±0.70	82.76±0.61	**78.86**±0.92
GBK-GNN	74.57±0.47	45.98±0.71	90.85±0.58	81.01±0.67	74.47±0.86
GloGNN	59.63±0.69	36.89±0.14	51.08±1.23	73.39±1.17	65.74±1.19
GPR-GNN	64.85±0.27	44.88±0.34	86.24±0.61	72.94±0.97	55.48±0.91
JacobiConv	71.14±0.42	43.55±0.48	89.66±0.40	68.66±0.65	73.88±1.16
Ours					
$\text{TANGO}_{\text{GatedGCN}}$	**91.89**±0.30	52.60±0.53	98.32±0.59	85.51±0.98	80.39±1.04
$\text{TANGO}_{\text{GPS}}$	91.08±0.57	**53.83**±0.32	**98.39**±0.54	**85.66**±1.01	**80.32**±1.07

Table 8: Ablation study on the importance of using a gradient of an energy term in Equation (6).

Model	ZINC-12k MAE ↓	Roman-empire Acc. ↑	Peptides-func AP ↑	Diameter $\log_{10}$(MSE) ↓
GatedGCN	0.282±0.015	74.46±0.54	58.64±0.77	0.1348±0.0397
$\text{TANGO-NON-ENERGY}_{\text{GatedGCN}}$	0.138±0.014	86.94±0.43	68.07±0.45	-0.5992±0.0831
$\text{TANGO}_{\text{GatedGCN}}$	**0.128**±0.011	**91.89**±0.30	**68.92**±0.40	**-0.6681**±0.0745
GPS	0.070±0.004	87.04±0.58	65.35±0.41	-0.5121±0.0426
$\text{TANGO-NON-ENERGY}_{\text{GPS}}$	0.067±0.004	89.00±0.61	67.58±0.39	-0.7178±0.0729
$\text{TANGO}_{\text{GPS}}$	**0.062**±0.005	**91.08**±0.57	**70.21**±0.43	**-0.9772**±0.0518

Table 9: The importance of using a tangential term to the energy term in Equation (6).

Model	ZINC-12k MAE ↓	Roman-empire Acc. ↑	Peptides-func AP ↑	Diameter $\log_{10}$(MSE) ↓
GatedGCN	0.282±0.015	74.46±0.54	58.64±0.77	0.1348±0.0397
$\text{TANGO-NON-TANGENT}_{\text{GatedGCN}}$	0.186±0.016	83.59±0.48	68.01±0.52	-0.2193±0.0899
$\text{TANGO}_{\text{GatedGCN}}$	**0.128**±0.011	**91.89**±0.30	**68.92**±0.40	**-0.6681**±0.0745
GPS	0.070±0.004	87.04±0.58	65.35±0.41	-0.5121±0.0426
$\text{TANGO-NON-TANGENT}_{\text{GPS}}$	0.066±0.010	88.57±0.72	67.33±0.59	-0.2916±0.0404
$\text{TANGO}_{\text{GPS}}$	**0.062**±0.005	**91.08**±0.57	**70.21**±0.43	**-0.9772**±0.0518

Table 10: Results for Peptides-func and Peptides-struct averaged over 3 training seeds. Baseline results are taken from [28] and [43]. Re-evaluated methods employ the 3-layer MLP readout proposed in [85]. Note that all MPNN-based methods include structural and positional encoding. ‡ means 3-layer MLP readout and residual connections are employed based on [85]. This table is an extended version of the focused Table 6.

Model	Peptides-func AP ↑	Peptides-struct MAE ↓
MPNNs		
GCN	$59.30_{\pm 0.23}$	$0.3496_{\pm 0.0013}$
GINE	$54.98_{\pm 0.79}$	$0.3547_{\pm 0.0045}$
GCNII	$55.43_{\pm 0.78}$	$0.3471_{\pm 0.0010}$
GatedGCN	$58.64_{\pm 0.77}$	$0.3420_{\pm 0.0013}$
Multi-hop GNNs		
DIGL+MPNN	$64.69_{\pm 0.19}$	$0.3173_{\pm 0.0007}$
DIGL+MPNN+LapPE	$68.30_{\pm 0.26}$	$0.2616_{\pm 0.0018}$
MixHop-GCN	$65.92_{\pm 0.36}$	$0.2921_{\pm 0.0023}$
MixHop-GCN+LapPE	$68.43_{\pm 0.49}$	$0.2614_{\pm 0.0023}$
DRew-GCN	$69.96_{\pm 0.76}$	$0.2781_{\pm 0.0028}$
DRew-GCN+LapPE	$71.50_{\pm 0.44}$	$0.2536_{\pm 0.0015}$
DRew-GIN	$69.40_{\pm 0.74}$	$0.2799_{\pm 0.0016}$
DRew-GIN+LapPE	$71.26_{\pm 0.45}$	$0.2606_{\pm 0.0014}$
DRew-GatedGCN	$67.33_{\pm 0.94}$	$0.2699_{\pm 0.0018}$
DRew-GatedGCN+LapPE	$69.77_{\pm 0.26}$	$0.2539_{\pm 0.0007}$
Transformers		
Transformer+LapPE	$63.26_{\pm 1.26}$	$0.2529_{\pm 0.0016}$
SAN+LapPE	$63.84_{\pm 1.21}$	$0.2683_{\pm 0.0043}$
GraphGPS+LapPE	$65.35_{\pm 0.41}$	$0.2500_{\pm 0.0005}$
Modified and Re-evaluated‡		
GCN	$68.60_{\pm 0.50}$	$0.2460_{\pm 0.0007}$
GINE	$66.21_{\pm 0.67}$	$0.2473_{\pm 0.0017}$
GatedGCN	$67.65_{\pm 0.47}$	$0.2477_{\pm 0.0009}$
GraphGPS	$65.34_{\pm 0.91}$	$0.2509_{\pm 0.0014}$
DE-GNNs		
GRAND	$57.89_{\pm 0.62}$	$0.3418_{\pm 0.0015}$
GraphCON	$60.22_{\pm 0.68}$	$0.2778_{\pm 0.0018}$
A-DGN	$59.75_{\pm 0.44}$	$0.2874_{\pm 0.0021}$
SWAN	$67.51_{\pm 0.39}$	$0.2485_{\pm 0.0009}$
Graph SSMs		
Graph-Mamba	$67.39_{\pm 0.87}$	$0.2478_{\pm 0.0016}$
GMN	$70.71_{\pm 0.83}$	$0.2473_{\pm 0.0025}$
Ours		
$\text{TANGO}_{\text{GATEDGCN}}$	$68.92_{\pm 0.40}$	$0.2451_{\pm 0.0006}$
$\text{TANGO}_{\text{GPS}}$	$70.21_{\pm 0.43}$	$0.2422_{\pm 0.0014}$

Table 11: Mean test set score and std averaged over the splits from Platonov et al. [72]. This table is an extended version of the focused Table 7. Baseline results are reported from [34, 61, 66, 72].

Model	Roman-empire Acc ↑	Amazon-ratings Acc ↑	Minesweeper AUC ↑	Tolokers AUC ↑	Questions AUC ↑
MPNNs					
GAT	$80.87_{\pm 0.30}$	$49.09_{\pm 0.63}$	$92.01_{\pm 0.68}$	$83.70_{\pm 0.47}$	$77.43_{\pm 1.20}$
GAT-sep	$88.75_{\pm 0.41}$	$52.70_{\pm 0.62}$	$93.91_{\pm 0.35}$	$83.78_{\pm 0.43}$	$76.79_{\pm 0.71}$
GAT (LapPE)	$84.80_{\pm 0.46}$	$44.90_{\pm 0.73}$	$93.50_{\pm 0.54}$	$84.99_{\pm 0.54}$	$76.55_{\pm 0.84}$
GAT (RWSE)	$86.62_{\pm 0.53}$	$48.58_{\pm 0.41}$	$92.53_{\pm 0.65}$	$85.02_{\pm 0.67}$	$77.83_{\pm 1.22}$
GAT (DEG)	$85.51_{\pm 0.56}$	$51.65_{\pm 0.60}$	$93.04_{\pm 0.62}$	$84.22_{\pm 0.81}$	$77.10_{\pm 1.23}$
Gated-GCN	$74.46_{\pm 0.54}$	$43.00_{\pm 0.32}$	$87.54_{\pm 1.22}$	$77.31_{\pm 1.14}$	$76.61_{\pm 1.13}$
GCN	$73.69_{\pm 0.74}$	$48.70_{\pm 0.63}$	$89.75_{\pm 0.52}$	$83.64_{\pm 0.67}$	$76.09_{\pm 1.27}$
GCN (LapPE)	$83.37_{\pm 0.55}$	$44.35_{\pm 0.36}$	$94.26_{\pm 0.49}$	$84.95_{\pm 0.78}$	$77.79_{\pm 1.34}$
GCN (RWSE)	$84.84_{\pm 0.55}$	$46.40_{\pm 0.55}$	$93.84_{\pm 0.48}$	$85.11_{\pm 0.77}$	$77.81_{\pm 1.40}$
GCN (DEG)	$84.21_{\pm 0.47}$	$50.01_{\pm 0.69}$	$94.14_{\pm 0.50}$	$82.51_{\pm 0.83}$	$76.96_{\pm 1.21}$
CO-GNN(Σ, Σ)	$91.57_{\pm 0.32}$	$51.28_{\pm 0.56}$	$95.09_{\pm 1.18}$	$83.36_{\pm 0.89}$	$80.02_{\pm 0.86}$
CO-GNN(μ, μ)	$91.37_{\pm 0.35}$	$54.17_{\pm 0.37}$	$97.31_{\pm 0.41}$	$84.45_{\pm 1.17}$	$76.54_{\pm 0.95}$
SAGE	$85.74_{\pm 0.67}$	$53.63_{\pm 0.39}$	$93.51_{\pm 0.57}$	$82.43_{\pm 0.44}$	$76.44_{\pm 0.62}$
Graph Transformers					
Exphormer	$89.03_{\pm 0.37}$	$53.51_{\pm 0.46}$	$90.74_{\pm 0.53}$	$83.77_{\pm 0.78}$	$73.94_{\pm 1.06}$
NAGphormer	$74.34_{\pm 0.77}$	$51.26_{\pm 0.72}$	$84.19_{\pm 0.66}$	$78.32_{\pm 0.95}$	$68.17_{\pm 1.53}$
GOAT	$71.59_{\pm 1.25}$	$44.61_{\pm 0.50}$	$81.09_{\pm 1.02}$	$83.11_{\pm 1.04}$	$75.76_{\pm 1.66}$
GPS	$82.00_{\pm 0.61}$	$53.10_{\pm 0.42}$	$90.63_{\pm 0.67}$	$83.71_{\pm 0.48}$	$71.73_{\pm 1.47}$
$\text{GPS}_{\text{GCN+Performer}}$ (LapPE)	$83.96_{\pm 0.53}$	$48.20_{\pm 0.67}$	$93.85_{\pm 0.41}$	$84.72_{\pm 0.77}$	$77.85_{\pm 1.25}$
$\text{GPS}_{\text{GCN+Performer}}$ (RWSE)	$84.72_{\pm 0.65}$	$48.08_{\pm 0.85}$	$92.88_{\pm 0.50}$	$84.81_{\pm 0.86}$	$76.45_{\pm 1.51}$
$\text{GPS}_{\text{GCN+Performer}}$ (DEG)	$83.38_{\pm 0.68}$	$48.93_{\pm 0.47}$	$93.60_{\pm 0.47}$	$80.49_{\pm 0.97}$	$74.24_{\pm 1.18}$
$\text{GPS}_{\text{GAT+Performer}}$ (LapPE)	$85.93_{\pm 0.52}$	$48.86_{\pm 0.38}$	$92.62_{\pm 0.79}$	$84.62_{\pm 0.54}$	$76.71_{\pm 0.98}$
$\text{GPS}_{\text{GAT+Performer}}$ (RWSE)	$87.04_{\pm 0.58}$	$49.92_{\pm 0.68}$	$91.08_{\pm 0.58}$	$84.38_{\pm 0.91}$	$77.14_{\pm 1.49}$
$\text{GPS}_{\text{GAT+Performer}}$ (DEG)	$85.54_{\pm 0.58}$	$51.03_{\pm 0.60}$	$91.52_{\pm 0.46}$	$82.45_{\pm 0.89}$	$76.51_{\pm 1.19}$
$\text{GPS}_{\text{GCN+Transformer}}$ (LapPE)	OOM	OOM	$91.82_{\pm 0.41}$	$83.51_{\pm 0.93}$	OOM
$\text{GPS}_{\text{GCN+Transformer}}$ (RWSE)	OOM	OOM	$91.17_{\pm 0.51}$	$83.53_{\pm 1.06}$	OOM
$\text{GPS}_{\text{GCN+Transformer}}$ (DEG)	OOM	OOM	$91.76_{\pm 0.61}$	$80.82_{\pm 0.95}$	OOM
$\text{GPS}_{\text{GAT+Transformer}}$ (LapPE)	OOM	OOM	$92.29_{\pm 0.61}$	$84.70_{\pm 0.56}$	OOM
$\text{GPS}_{\text{GAT+Transformer}}$ (RWSE)	OOM	OOM	$90.82_{\pm 0.56}$	$84.01_{\pm 0.96}$	OOM
$\text{GPS}_{\text{GAT+Transformer}}$ (DEG)	OOM	OOM	$91.58_{\pm 0.56}$	$81.89_{\pm 0.85}$	OOM
GT	$86.51_{\pm 0.73}$	$51.17_{\pm 0.66}$	$91.85_{\pm 0.76}$	$83.23_{\pm 0.64}$	$77.95_{\pm 0.68}$
GT-sep	$87.32_{\pm 0.39}$	$52.18_{\pm 0.80}$	$92.29_{\pm 0.47}$	$82.52_{\pm 0.92}$	$78.05_{\pm 0.93}$
Heterophily-Designated GNNs					
CPGNN	$63.96_{\pm 0.62}$	$39.79_{\pm 0.77}$	$52.03_{\pm 5.46}$	$73.36_{\pm 1.01}$	$65.96_{\pm 1.95}$
FAGCN	$65.22_{\pm 0.56}$	$44.12_{\pm 0.30}$	$88.17_{\pm 0.73}$	$77.75_{\pm 1.05}$	$77.24_{\pm 1.26}$
FSGNN	$79.92_{\pm 0.56}$	$52.74_{\pm 0.83}$	$90.08_{\pm 0.70}$	$82.76_{\pm 0.61}$	$78.86_{\pm 0.92}$
GBK-GNN	$74.57_{\pm 0.47}$	$45.98_{\pm 0.71}$	$90.85_{\pm 0.58}$	$81.01_{\pm 0.67}$	$74.47_{\pm 0.86}$
GloGNN	$59.63_{\pm 0.69}$	$36.89_{\pm 0.14}$	$51.08_{\pm 1.23}$	$73.39_{\pm 1.17}$	$65.74_{\pm 1.19}$
GPR-GNN	$64.85_{\pm 0.27}$	$44.88_{\pm 0.34}$	$86.24_{\pm 0.61}$	$72.94_{\pm 0.97}$	$55.48_{\pm 0.91}$
H2GCN	$60.11_{\pm 0.52}$	$36.47_{\pm 0.23}$	$89.71_{\pm 0.31}$	$73.35_{\pm 1.01}$	$63.59_{\pm 1.46}$
JacobiConv	$71.14_{\pm 0.42}$	$43.55_{\pm 0.48}$	$89.66_{\pm 0.40}$	$68.66_{\pm 0.65}$	$73.88_{\pm 1.16}$
Graph SSMs					
GMN	$87.69_{\pm 0.50}$	$54.07_{\pm 0.31}$	$91.01_{\pm 0.23}$	$84.52_{\pm 0.21}$	–
GPS + Mamba	$83.10_{\pm 0.28}$	$45.13_{\pm 0.97}$	$89.93_{\pm 0.54}$	$83.70_{\pm 1.05}$	–
Ours					
$\text{TANGO}_{\text{GatedGCN}}$	$91.89_{\pm 0.30}$	$52.60_{\pm 0.53}$	$98.32_{\pm 0.59}$	$85.51_{\pm 0.98}$	$80.39_{\pm 1.04}$
$\text{TANGO}_{\text{GPS}}$	$91.08_{\pm 0.57}$	$53.83_{\pm 0.32}$	$98.39_{\pm 0.54}$	$85.66_{\pm 1.01}$	$80.32_{\pm 1.07}$

Table 12: Mean test set MSE (↓) and std averaged on 4 random weight initializations on Graph Property Prediction. Lower is better. **First**, **second**, and **third** best results for each task are color-coded.

Model	Diameter	SSSP	Eccentricity
MPNNs			
GatedGCN [11]	$1.363955_{\pm 0.124683}$	$5.483\text{e-}04_{\pm 6.489e-05}$	$5.006106_{\pm 0.348115}$
GCN [52]	$5.525862_{\pm 0.592928}$	$8.910457_{\pm 0.002052}$	$7.027486_{\pm 0.045308}$
GAT [89]	$6.638959_{\pm 1.149565}$	$4.955643_{\pm 1.710477}$	$6.178741_{\pm 0.315841}$
GraphSAGE [44]	$7.319813_{\pm 0.675865}$	$1.933303_{\pm 0.820429}$	$6.113642_{\pm 0.291398}$
GIN [98]	$4.102986_{\pm 0.935300}$	$0.287872_{\pm 0.277933}$	$8.920722_{\pm 0.014379}$
GCNII [16]	$3.378314_{\pm 0.443395}$	$0.073638_{\pm 0.002289}$	$5.807644_{\pm 0.474727}$
DE-GNNs			
DGC [73]	$4.006822_{\pm 0.046130}$	$0.710722_{\pm 0.037803}$	$6.700389_{\pm 0.049370}$
GRAND [15]	$4.693534_{\pm 0.529556}$	$0.805008_{\pm 0.722347}$	$4.572987_{\pm 1.466786}$
GraphCON [78]	$1.248533_{\pm 0.178241}$	$0.041343_{\pm 8.758e-04}$	$4.822808_{\pm 0.082176}$
A-DGN [39]	$0.302831_{\pm 0.126350}$	$5.732\text{e-}04_{\pm 9.912e-05}$	$2.689057_{\pm 0.621036}$
SWAN [40]	**0.252290**$_{\pm 0.066515}$	$2.867\text{e-}04_{\pm 5.480e-05}$	**0.843529**$_{\pm 0.425363}$
PH-DGN [46]	**0.289401**$_{\pm 0.012461}$	**5.020e-05**$_{\pm 8.334e-06}$	**0.116198**$_{\pm 0.056107}$
Transformers			
GPS [74]	$0.307539_{\pm 0.030167}$	**2.518e-04**$_{\pm 1.130e-04}$	$4.052285_{\pm 0.263127}$
Ours			
TANGO$_{\text{GATEDGCN}}$	$0.214734_{\pm 0.036836}$	$8.658\text{e-}06_{\pm 1.479e-06}$	$0.018118_{\pm 4.422e-04}$
TANGO$_{\text{GPS}}$	**0.105390**$_{\pm 0.012570}$	**2.976e-06**$_{\pm 5.743e-07}$	**0.007153**$_{\pm 5.435e-05}$