Finsler Geometry, Graph Neural Networks, and You

T. Mitchell Roddenberry MITCH@RICE.EDU
Richard G. Baraniuk RICHB@RICE.EDU
Rice University, Houston, TX, USA

Abstract

Graph neural network architectures based on the graph Laplacian approximate the Laplace–Beltrami operator, thus limiting their application to isotropic operators. As a nonlinear alternative to the Laplace–Beltrami operator, we consider estimates of the Finsler Laplacian on point clouds sampled from a manifold. We prove that these discrete estimates converge to the true operator on the manifold as the number of point samples grows. Moreover, we show that this operator can be expressed as a graph neural network layer, which we use to define a family of Finslerian graph neural networks constrained to express Finsler geometry. We show that Finslerian graph neural networks recover the geometry underlying nonlinear diffusion equations in practice.

1. Introduction

Graph neural networks operate on discrete collections of points with specified pairwise relations. These methods incorporate the relations into the network parameterization, often inspired by the "right" operator on a limiting object that the graph approximates. For example, the graph Laplacian is understood as a discrete approximation of the Laplace–Beltrami operator, so that graph neural networks using the Laplacian implicitly treat the graph as being sampled from a Riemannian manifold.

Using a basic operator, such as the Laplacian, graph neural networks are constructed by interleaving the operator and local nonlinearities, mimicking the composition of affine transforms and nonlinearities in feedforward neural networks. Despite the nonlinearity, these neural networks are essentially isotropic.

The natural generalization of Riemannian geometry to allow for anisotropy is *Finsler geometry*, which equips the tangent spaces of a smooth manifold with a general Minkowski norm, rather than an inner product. This allows for asymmetric, non-elliptical geometry, as pictured in Figure 1. Finsler manifolds share many properties with their Riemannian counterparts, including a Laplace operator as the Fréchet derivative of an energy functional. Defining the Laplacian on a Finsler manifold in this way yields a *nonlinear operator*, making it a viable candidate for nonlinear learning techniques that use interpretable geometric structures from the outset.

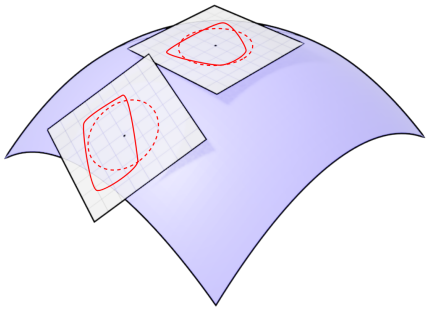

Figure 1: Riemannian (dashed) and Finsler (solid) unit balls on the tangent bundle of a manifold. The Finsler metric exhibits anisotropy, non-ellipticity, and asymmetry.

© 2026 T.M. Roddenberry & R.G. Baraniuk.

Extending the approximation of the Laplace–Beltrami operator on submanifolds by the Laplacian of a sampled graph, we consider the problem of discretely approximating nonlinear Laplacians that inherit their structure from Finsler geometries. To wit, we

1. Define the empirical Finsler Laplacian for point clouds (Section 2.1)

2. Prove that the empirical Finsler Laplacian almost surely converges to the continuüm operator (Theorem 2)

3. Express the empirical Finsler Laplacian as a graph neural network layer (Section 3), and define a class of Finslerian graph neural networks (Section 3.2)

4. Demonstrate the application of the Finslerian graph neural network to the inverse problem of recovering a Finsler metric from observed heat diffusion (Section 4).

Related Work. The convergence of graph Laplacians to the Laplace–Beltrami operator on manifolds has been studied from multiple angles (Belkin and Niyogi, 2004, 2008; Coifman and Lafon, 2006; Trillos et al., 2020) and application settings, particularly in semi-supervised learning (Trillos and Slepčev, 2018; Calder et al., 2023). The convergence of p-Laplacians has also been studied (Slepčev and Thorpe, 2019), yielding a class of methods for semi-supervised learning that are nonlinear, but still isotropic. Complementing recent results of Wang et al. (2025) establishing the convergence of graph neural networks to manifold neural networks which use convolution operators based on the Laplace–Beltrami operator and pointwise nonlinearities (such as the ReLU), the nonlinearity of the operators we study is a feature of the *geometry itself*. This offers a different motivation and approach for nonlinearities in graph neural network design.

There have been recent works on applications of Finsler geometry in computer vision and data science, with emphasis on the use of Randers metrics in particular (Weber et al., 2024; Dages et al., 2025; Gahtan et al., 2026). The work of Weber et al. (2024) is similar to ours in its use of the heat equation based on Finsler geometry, but is focused on applying the heat kernel for settings where the Finsler metric is known and computed on a manifold discretized as a *mesh*, while we study both the case where the manifold is discretized as an unstructured *point cloud* and prove convergence as the number of points grows. Additionally, we consider an inverse problem of parameterizing and recovering Finsler metrics from observed heat diffusion (Section 4), which is a counterpart to the inverse problem of recovering a Randers metric as considered by Gahtan et al. (2026). Our work differs in its focus on *heat equations* rather than *eikonal equations*, and is also not limited to Randers metrics or a particular discretization of the space.

Our discretization of the Finsler Laplacian draws inspiration from methods for solving partial differential equations on point clouds (Liang and Zhao, 2013). Similar to Randers metrics, "magnetic Laplacians" have been used in graph signal processing (Furutani et al., 2019) and subsequently for the construction of graph neural networks (Zhang et al., 2021; He et al., 2022), with the goal of modeling asymmetric systems on graphs. Indeed, Riemannian geometries under the influence of a weak magnetic field have been noted to be equivalent to Randers metrics (Tabachnikov, 2004). Our proposed Finslerian graph neural networks (Section 3) model general Finsler metrics, of which Randers metrics are a special case.

2. Finsler Laplacians

Let F be a Minkowski norm on $\mathbb{R}^D$, and let a probability measure μ supported on a closed, compact, smooth, d-dimensional submanifold $\mathcal{M} \subset \mathbb{R}^D$ be given. Let p be the density of μ with respect to the Hausdorff measure of the manifold. Under this structure, $\mathcal{M}$ is a weighted Finsler manifold (Bao et al., 2000) with metric pulled back from F via the inclusion map, with the co-Finsler metric given by the dual norm F^*. For technical background, we refer to Appendix A.

Define an energy functional for differentiable functions on $\mathcal{M}$:

$$E[f] = \int \frac{1}{2} \left[F^*(\nabla f(x))\right]^2 d\mu(x).$$

The *Finsler Laplacian* (Ge and Shen, 2000; Ohta and Sturm, 2009) is a nonlinear operator defined as the Fréchet derivative of E, or equivalently as the divergence of the gradient of the squared (co-)norm:

$$\Delta[f] = \frac{1}{p} D\{E[f]\} = \frac{1}{p} \mathrm{div}\left(pJ(\nabla f)\right),$$

where

$$J(\xi) = \nabla_\xi \frac{1}{2} \left[F^*(\xi)\right]^2 = F^*(\xi) \nabla_\xi F^*(\xi). \tag{1}$$

Observe that taking F as the Euclidean norm recovers the usual density-weighted Laplace–Beltrami operator.

Remark 1 *For the reader who likes keeping track of indices, $J : T^*\mathcal{M} \to T\mathcal{M}$ is the Legendre transformation for the cotangent/tangent bundles of the manifold.*

Example 1 *Let $\mathcal{M} \subset \mathbb{R}^3$ be a cylindrical surface. We consider the heat equation $\dot{f}(t) = -\Delta[f(t)]$, where $f(0)$ is the sum of two Dirac delta functions on $\mathcal{M}$, and Δ is either the Laplace–Beltrami operator, or the Finsler Laplacian induced by a Minkowski norm on the ambient space. In particular, for some vector v with norm less than one, let $F^*(\xi) = \|\xi\| + \langle \xi, v \rangle$, so that F^* is a Randers metric on the cotangent bundle of the ambient space. As shown in Figure 2, the Randers metric causes the diffusion of heat to "drift" according to the vector v.*

2.1. The Finsler Graph Laplacian

Let $\mathcal{X} = \{x_i\}_{i=1}^n$ be sampled *i.i.d.* from the distribution μ, and suppose we have access to the function values $f(x_i)$ for each i. We wish to estimate the gradients ∇f based on these samples, given a translation-invariant nonnegative kernel κ and $\epsilon > 0$. Assume that $\kappa : \mathbb{R}^D \to \mathbb{R}$ is smooth, compactly supported, and radially symmetric. Define $\kappa_\epsilon(z) := \epsilon^{-d} \kappa(z/\epsilon)$.

We use a local principal component analysis method for estimating the gradient of f. Namely, we define

$$\nabla_{\epsilon,n} f(x) = C_{\epsilon,n}^\dagger(x) b_{\epsilon,n}[f](x),$$

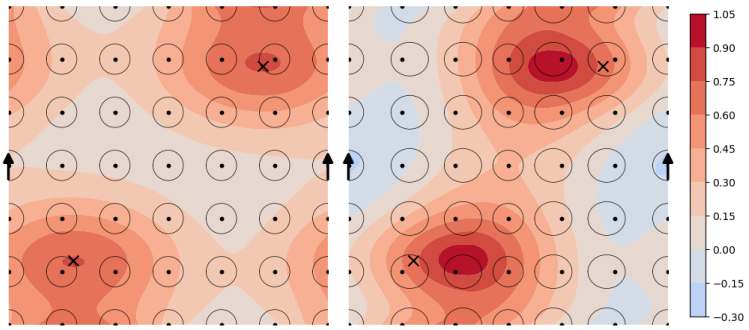

Figure 2: Heat diffusion on a cylinder (identified edges marked by arrows) with two point sources (marked ×) using the isotropic Laplace–Beltrami operator (left) and the Finsler Laplacian derived from an ambient Randers metric (right). Panels show Tissot's indicatrices for the respective metrics. The Randers heat equation exhibits "drift" in accordance with the off-center geometry.

where $C^{\dagger}_{\epsilon,n}$ is (almost) a pseudoinverse of the kernel-weighted covariance centered at x, and $b_{\epsilon,n}[f](x)$ is the kernel-weighted local variation of f near x.[1] Using the empirical gradient, we define the empirical energy functional

$$E_{\epsilon,n}[f] = \frac{1}{n}\sum_{i=1}^{n} \frac{1}{2}\left[F^*(\nabla_{\epsilon,n} f(x_i))\right]^2,$$

and the *empirical Finsler Laplacian* via the Fréchet derivative: $\Delta_{\epsilon,n}[f] = nD\{E[f]\}$. For convenience, define the (co)vector $\xi(x) = C^{\dagger}_{\epsilon,n}(x) J(\nabla_{\epsilon,n} f(x))$. This yields the formula

$$\Delta_{\epsilon,n}[f](x) = \frac{1}{n\epsilon^2}\sum_{i=1}^{n} \kappa_\epsilon\left(x - x_i\right)\langle \xi(x) + \xi(x_i), x - x_i\rangle. \tag{2}$$

We define the *Finsler graph Laplacian* $\mathcal{L}_{\epsilon,n}$ as the restriction of the empirical Finsler Laplacian to the point cloud $\mathcal{X}$. When ϵ, n are fixed or are clear from context, we simply denote the Finsler graph Laplacian by $\mathcal{L}$.

Although the Finsler Laplacian and its graph counterpart are derived from similar energy functionals, the capacity of the Finsler graph Laplacian to approximate the underlying operator does not follow immediately. Since $n \to \infty$, if $\epsilon \to 0$ at a suitable rate, the Finsler graph Laplacian converges to the continuüm Finsler Laplacian in the following sense:

Theorem 2 *Let μ be a probability measure supported on a smooth, closed, and compact d-dimensional submanifold $\mathcal{M} \subset \mathbb{R}^D$, with density p bounded away from zero on its support. Assume that f is a C^3 function on this submanifold. For each $n \geq 1$, let $\{x_i\}_{i=1}^n$ be a set of points sampled i.i.d. according to the measure μ. Then, for $\epsilon = O\left(\log n/n\right)^{1/(3d+4)}$,*

$$\lim_{n\to\infty} \max_{j\in[1,n]} \left|\Delta[f](x_j) - \mathcal{L}_{\epsilon,n}[f](x_j)\right| = 0 \quad \textit{almost surely.}$$

We prove this in Appendix D.

1. We offer more precise definitions of these objects in Appendix B.

3. The Finsler Graph Laplacian and Neural Networks

Cellular sheaves are a powerful tool for organizing operators on graphs that transform data between vector spaces (Curry, 2014; Hansen and Gebhart, 2020; Barbero et al., 2022). We cast the Finsler graph Laplacian in this framework, which motivates a new graph neural network architecture.

3.1. Cellular Sheaves for the Finsler Laplacian

Let $\mathcal{X} = \{x_i\}_{i=1}^n$ be a point cloud, with an observed function $f : \mathcal{X} \to \mathbb{R}$. After constructing a graph using $\mathcal{X}$, we construct sheaf structures over the graph such that appropriate composition of restriction maps, morphisms, and their adjoints computes the Finsler graph Laplacian. For some fixed $\epsilon > 0$, let $\mathcal{G} = (\mathcal{X}, \mathcal{E})$ be a graph so that $\mathcal{X}$ is the set of nodes and $\mathcal{E}$ consists of unordered pairs of nodes (x_i, x_j) such that $\kappa_\epsilon(x_i - x_j) \neq 0$. We define two cellular sheaves[2] on $\mathcal{G}$. The first we denote by $\mathcal{F}$, such that all vector spaces $\mathcal{F}(x)$ and $\mathcal{F}(e)$ are the real line $\mathbb{R}$. For any incident node-edge pair, the restriction map is defined as $\mathcal{F}_{x \to e} f_x = f_x$ for $f_x \in \mathcal{F}(x) \simeq \mathbb{R}$. The second sheaf is denoted by $\mathcal{T}$, and assigns a copy of the tangent space $T_x\mathbb{R}^D$ to each node $x \in \mathcal{X}$, while still assigning the real line to each edge $e \in \mathcal{E}$. For each node $x \in \mathcal{X}$ and any incident edge $(x, y) \in \mathcal{E}$, the restriction map is

$$\mathcal{T}_{x \to (x,y)} v = \frac{1}{n\epsilon^2} \kappa_\epsilon (x - y) \langle C^\dagger_{\epsilon,n}(x)(x - y), v \rangle,$$

for $v \in T_x\mathbb{R}^D \simeq \mathbb{R}^D$. We identify the tangent spaces with the cotangent spaces $T_x\mathbb{R}^D \simeq T_x^*\mathbb{R}^D$ using the ambient Euclidean metric. We define the following maps on the spaces of edge/node data:

- $\mathrm{id} : C^1(\mathcal{F}) \to C^1(\mathcal{T})$ and $\mathrm{id}^* : C^1(\mathcal{T}) \to C^1(\mathcal{F})$ are both identity maps under the obvious identification $C^1(\mathcal{F}) \simeq C^1(\mathcal{T})$,
- $J : C^0(\mathcal{T}) \to C^0(\mathcal{T})$ passes (co)vector data on the nodes through the map J as defined in (1).

The vector spaces constituting both sheaves are endowed with the usual inner product structure; the restriction maps and their adjoints induce a differential $d : C^0 \to C^1$ and a codifferential $\delta : C^1 \to C^0$ for both sheaves $\mathcal{F}, \mathcal{T}$. We model the function on the nodes as a 0-cochain $f \in C^0(\mathcal{F})$. The Finsler graph Laplacian is represented as

$$\mathcal{L}[f] = (\delta \circ \mathrm{id}^* \circ d \circ J \circ \delta \circ \mathrm{id} \circ d)[f],$$

where $d : C^0 \to C^1$ generically indicates the differential and $\delta : C^1 \to C^0$ the codifferential. Perhaps more legibly, the Finsler graph Laplacian traverses the following (non-commutative) diagram:

$$C^0(\mathcal{F}) \underset{\delta}{\overset{d}{\rightleftarrows}} C^1(\mathcal{F}) \underset{\mathrm{id}^*}{\overset{\mathrm{id}}{\rightleftarrows}} C^1(\mathcal{T}) \underset{d}{\overset{\delta}{\rightleftarrows}} C^0(\mathcal{T}) \circlearrowright J \,. \tag{3}$$

2. See Appendix C for background information on cellular (co)sheaves.

Considering the map J as an activation function, the Finsler graph Laplacian can be thought of as a layer in a sheaf neural network (Hansen and Gebhart, 2020; Barbero et al., 2022), where the nonlinearity is applied in a "lifted" space (namely, the cotangent space at each point). In contrast, the nonlinearity for a typically defined sheaf neural network is applied in the same domain as the input space, that is, as a map $\sigma : C^0(\mathcal{F}) \to C^0(\mathcal{F})$.

3.2. Finslerian Graph Neural Networks

We now consider how a graph neural network may be designed to parameterize a Finsler metric. Specifically, we will replace J with a simple neural network. For $W, V \in \mathbb{R}^{E \times D}$, and an activation function $\sigma : \mathbb{R} \to \mathbb{R}$ that is positive homogeneous of degree 1 with $\sigma' \geq 0$, define the approximation

$$\widehat{J}(\xi) = V^\top \sigma(W\xi), \tag{4}$$

where σ is applied elementwise in the standard basis. The following result gives a condition that guarantees $\widehat{J}$ to be "Finslerian," in the sense of being the gradient of a squared-norm:

Proposition 3 *If $V = W$ in (4), $\operatorname{rank}(W) = D$, and $\sigma' > 0$, then $\widehat{J}(\xi) = \nabla_\xi \frac{1}{2} \left[\widehat{F}^*(\xi)\right]^2$ where*

$$\widehat{F}^*(\xi) = \sqrt{\langle \xi, \widehat{J}(\xi) \rangle} \tag{5}$$

is an (almost) Minkowski norm. Under the further assumption that $\sigma(x) = \alpha x$ for $\alpha > 0$, the norm $\widehat{F}^*$ is Riemannian.

We prove Proposition 3 in Appendix A.1. Based on this, for a (potentially constant) learnable matrix $W(t)$, we define a *Finslerian graph neural network* under the framework of graph neural diffusion (Chamberlain et al., 2021; Thorpe et al., 2022; Bodnar et al., 2022); that is, an ordinary differential equation of the form

$$\dot{f}(t) = -\widehat{\mathcal{L}}_{W(t)}[f(t)], \tag{6}$$

where $\widehat{\mathcal{L}}_{W(t)}$ is the Finsler graph Laplacian defined using $\widehat{J}(\xi) = W^\top(t)\sigma(W(t)\xi)$. If $W(t)$ varies with t, this models a setting where the ambient *Minkowski norm* changes over time. Otherwise, the ambient norm is constant so that the heat diffusion process is time-shift invariant. We take W to be constant in our experiments.

The constraint $V = W$ assumed in Proposition 3 and realized in (6) is akin to enforcing shift-invariance in convolutional neural networks or permutation-invariance in graph neural networks. The more general unconstrained version is obviously capable of expressing a superset of the functions realizable under the symmetry constraint, but there is no guarantee that learning algorithms in practice will "discover" this symmetric property. So, we enforce the desired symmetries to aid generalization and interpretability.

4. Numerical Experiments

We demonstrate the Finsler graph Laplacian and Finslerian graph neural networks on a collection of numerical examples, corroborating our convergence result (Theorem 2) and the utility of a symmetric nonlinear network for representing Minkowski norms (Proposition 3). Implementation details can be found in Appendix E, and source code is available [here].

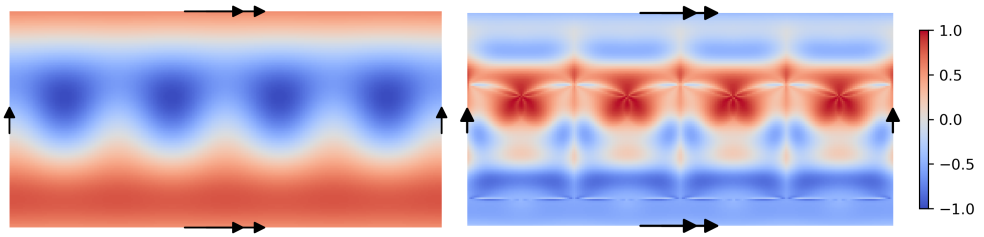

Figure 3: The function $f(x, y, z) = \cos(x) + \cos(y) + z$ restricted to the torus (left) and the Finsler Laplacian $\Delta[f]$ defined so that $F^*(\xi) = \|\xi\|_3$ (right). Identified edges of torus marked by arrows.

4.1. Convergence of the Finsler Graph Laplacian

We first validate the convergence result stated in Theorem 2. Let $\mathcal{M} \subset \mathbb{R}^3$ be a torus with major radius 2 and minor radius 1 oriented along the xy-plane. We consider the restriction of a function $f(x, y, z) = \cos(x) + \cos(y) + z$ on the ambient space to $\mathcal{M}$, and consider the Finsler Laplacian defined by the ℓ_3-norm, that is, where $F^*(\xi) = \left(\sum_i |\xi_i|^3\right)^{1/3}$. The function f and the application of the Finsler Laplacian $\Delta[f]$ are pictured in Figure 3. We sample points on the torus uniformly according to the ambient (Hausdorff) measure, and evaluate the Finsler graph Laplacian constructed using varying numbers of sampled points n and bandwidths ϵ. We then measure the cosine similarity between the true Finsler Laplacian evaluated at the sampled points and the Finsler graph Laplacian, to account for scaling. The results of this are shown in Figure 4 (left). As expected, the similarity between the true and graph Laplacian map increases as n grows from 250 to 9500, with the corresponding "best" value of ϵ decreasing at a modest rate as the point cloud becomes denser. This corroborates the convergence result established by Theorem 2.

4.2. Learning Geometry using Finslerian Graph Neural Networks

We consider a semisupervised geometric inverse problem of recovering a Finsler metric from an observed instance of heat diffusion. Namely, we sample a point cloud of $n = 600$ points uniformly on the unit square, and construct a graph using a gaussian kernel with a bandwidth of $\epsilon = 0.2$. In the graph construction, we periodize the domain, giving the manifold the topology of a torus. We generate a function $f(0)$ on the domain as a superposition of 24 Fourier modes of varying frequency, phase, and amplitude, then sample it to yield a function on the point cloud. Then, we run the heat equation using a Finsler Laplacian defined using a Randers (co)metric (see Example 1) with a drift vector $v = [0.25, 0.25]$. We observe the state $f(T)$ for $T = 1.0$, and use 20% of the labeled nodes for training.

Given this single input-output sample of the heat equation over one time unit, we apply multiple techniques to model the heat diffusion. The first two are Finslerian graph neural networks (FGNNs), both with hidden dimension $E = 64$, where one uses the ReLU activation and the other uses a linear activation. The third and fourth learn maps on the node signal space that more closely resembles a graph neural network (GNN). The initial input $f(0)$ is linearly lifted to $\mathbb{R}^E$, and heat diffusion is performed for node features in $\mathbb{R}^E$ where $\dot{f}(t) = \text{MLP}(Lf(t))$ with L denoting the standard graph Laplacian, and MLP a single-

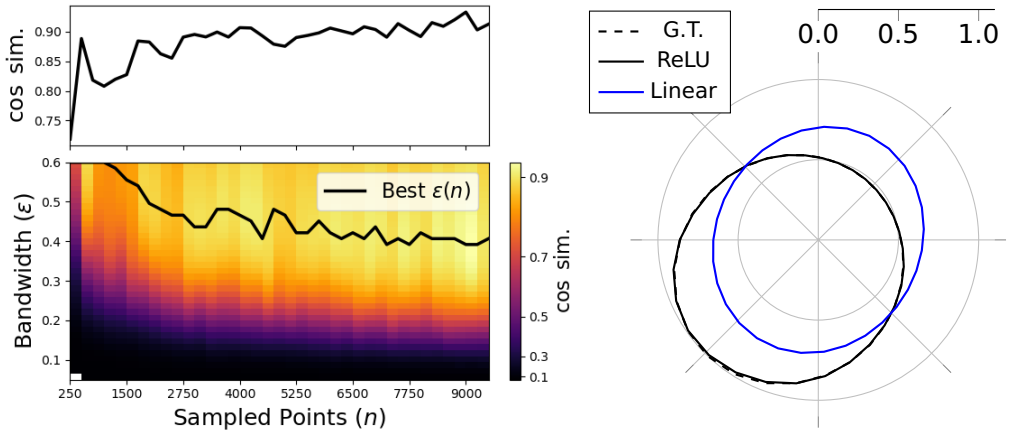

Figure 4: (Bottom left) Similarity between the Finsler graph Laplacian and the true Finsler Laplacian as a function of n and ϵ. (Top left) Cosine similarity for "best" $\epsilon(n)$. (Right) Recovery of a Randers metric by Finslerian graph neural networks.

hidden-layer neural network with either ReLU or linear activations, mapping $\mathbb{R}^E$ to $\mathbb{R}^E$ on each node. The output of the diffusion is then linearly projected back to $\mathbb{R}$.

We train all methods to map the initial input $f(0)$ to predict the labeled 20% of nodes at $f(T)$. We then measure the relative mean-squared error (MSE) over the remaining 80% of nodes, recorded under the column "Train Graph" in Table 1. The Finslerian graph neural networks strongly outperform the standard graph neural networks. Moreover, the nonlinear (ReLU) Finslerian graph neural network performs an order of magnitude better than the linear one, because the linear method is only capable of *Riemannian* diffusion, by Proposition 3. To test generalization, we resample the graph and generate 16 new initial conditions $\{f_j(0)\}_{j=1}^{16}$, and compare the trained models to the true Finsler heat equation on the new, unseen graph. As shown in Table 1 under the column "Test Graph," the Finslerian graph neural networks suffer no performance loss, reflecting the generalization suggested by Theorem 2. On the other hand, the graph neural networks appear to overfit to the training graph, performing much worse on the test graph.

Table 1: MSE of learned approximations to Finsler heat flow. Standard deviation of MSE over test signals shown for test graph.

Method	Train Graph	Test Graph (std.)
ReLU FGNN	1.2×10^{-7}	4.1×10^{-8} (1.2×10^{-8})
Linear FGNN	3.6×10^{-4}	7.8×10^{-5} (2.2×10^{-5})
ReLU GNN	4.7×10^{-3}	1.3×10^{-1} (1.3×10^{-1})
Linear GNN	2.6×10^{-3}	1.6×10^{-1} (1.6×10^{-1})

We also examine the qualitative properties of the Finslerian graph neural networks after training. Since both methods amount to learning $\widehat{J}(\xi) = \nabla_\xi \frac{1}{2}[\widehat{F}^*(\xi)]^2$, we plot the unit balls of the norms $\widehat{F}^*$ in Figure 4 (right). The ReLU model yields a metric that is extremely close to the ground truth (G.T.) metric. The model with linear activation learns a metric whose unit ball is an ellipse, due to its restriction to Riemannian metrics by Proposition 3.

5. Conclusion

We have developed the topological and algebraic tools for approximating Finsler geometric Laplacian operators on sampled point clouds. The proposed empirical Finsler Laplacian is shown to converge in a uniform pointwise sense, demonstrating its utility both for computations in PDEs and in semisupervised learning problems arising in data science. Beyond the applications, the empirical Finsler Laplacian demonstrates the limitations of graph signal processing and graph neural network methods that rely too heavily on isotropy. By linking the structure of the Finsler Laplacian to a class of nonlinear graph neural networks, we hope to promote study into how nonlinearities in neural architectures implicitly impose geometric structures on the domain. By tying the neural architecture design to a known geometric structure, such as a Minkowski norm, solutions found by learning algorithms become interpretable via the induced geometry.

Acknowledgments

This work was supported by ONR grant N00014-23-1-2714, DOE grant DE-SC0020345, DOI grant 140D0423C0076, and a Google Cloud Computing Award.

References

David Bao, S-S Chern, and Zhongmin Shen. *An Introduction to Riemann-Finsler Geometry*. Springer Science & Business Media, 2000.

Federico Barbero, Cristian Bodnar, Haitz Sáez de Ocáriz Borde, Michael Bronstein, Petar Veličković, and Pietro Liò. Sheaf neural networks with connection Laplacians. In *Topological, Algebraic and Geometric Learning Workshops*, 2022.

Mikhail Belkin and Partha Niyogi. Semi-supervised learning on Riemannian manifolds. *Machine learning*, 56(1), 2004.

Mikhail Belkin and Partha Niyogi. Towards a theoretical foundation for Laplacian-based manifold methods. *Journal of Computer and System Sciences*, 74(8), 2008.

Cristian Bodnar, Francesco Di Giovanni, Benjamin Chamberlain, Pietro Liò, and Michael Bronstein. Neural sheaf diffusion: A topological perspective on heterophily and oversmoothing in GNNs. *Advances in Neural Information Processing Systems*, 35, 2022.

James Bradbury et al. JAX: composable transformations of Python+NumPy programs, 2018. URL http://github.com/jax-ml/jax.

Dmitri Burago and Sergey Ivanov. Isometric embeddings of Finsler manifolds. *Algebra i Analiz*, 5(1), 1993. English: St. Petersburg Math Journal 5.1 (1994).

Jeff Calder, Dejan Slepčev, and Matthew Thorpe. Rates of convergence for Laplacian semi-supervised learning with low labeling rates. *Research in the Mathematical Sciences*, 10 (1):10, 2023.

Ben Chamberlain, James Rowbottom, Maria I Gorinova, Michael Bronstein, Stefan Webb, and Emanuele Rossi. GRAND: Graph neural diffusion. In *International Conference on Machine Learning*, 2021.

Ronald R Coifman and Stéphane Lafon. Diffusion maps. *Applied and Computational Harmonic Analysis*, 21(1), 2006.

Justin Michael Curry. *Sheaves, cosheaves and applications*. University of Pennsylvania, 2014.

Thomas Dages, Simon Weber, Ya-Wei Eileen Lin, Ronen Talmon, Daniel Cremers, Michael Lindenbaum, Alfred M Bruckstein, and Ron Kimmel. Finsler multi-dimensional scaling: Manifold learning for asymmetric dimensionality reduction and embedding. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025.

James F Davis, Benjamin R Edwards, and V Alan Kosteleckỳ. Characteristic tensors for almost Finsler manifolds. *The Journal of Geometric Analysis*, 36(4), 2026.

DeepMind et al. The DeepMind JAX Ecosystem, 2020. URL http://github.com/google-deepmind.

Satoshi Furutani, Toshiki Shibahara, Mitsuaki Akiyama, Kunio Hato, and Masaki Aida. Graph signal processing for directed graphs based on the Hermitian Laplacian. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, 2019.

Barak Gahtan, Jacob Shpund, and Alex M Bronstein. Wildfire simulation with differentiable Randers-Finsler eikonal solvers. *arXiv:2603.00035*, 2026.

Yuxin Ge and Zhongmin Shen. Eigenvalues and eigenfunctions of metric measure manifolds. *Proceedings of the London Mathematical Society*, 82(3), 2000.

Jakob Hansen and Thomas Gebhart. Sheaf neural networks. In *NeurIPS Workshop on Topological Data Analysis and Beyond*, 2020.

Yixuan He, Michael Perlmutter, Gesine Reinert, and Mihai Cucuringu. MSGNN: A spectral graph neural network based on a novel magnetic signed Laplacian. In *Learning on Graphs Conference*, 2022.

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

Patrick Kidger and Cristian Garcia. Equinox: neural networks in JAX via callable PyTrees and filtered transformations. *Differentiable Programming workshop at Neural Information Processing Systems*, 2021.

Jian Liang and Hongkai Zhao. Solving partial differential equations on point clouds. *SIAM Journal on Scientific Computing*, 35(3), 2013.

Shin-ichi Ohta and Karl-Theodor Sturm. Heat flow on Finsler manifolds. *Communications on Pure and Applied Mathematics*, 62(10), 2009.

Dejan Slepčev and Matthew Thorpe. Analysis of p-Laplacian regularization in semisupervised learning. *SIAM Journal on Mathematical Analysis*, 51(3), 2019.

Serge Tabachnikov. Remarks on magnetic flows and magnetic billiards, Finsler metrics and a magnetic analog of Hilbert's fourth problem. In *Modern dynamical systems and applications*. Cambridge University Press, 2004.

Matthew Thorpe, Tan Minh Nguyen, Hedi Xia, Thomas Strohmer, Andrea Bertozzi, Stanley Osher, and Bao Wang. GRAND++: Graph neural diffusion with a source term. In *International Conference on Learning Representations*, 2022.

Nicolás García Trillos and Dejan Slepčev. A variational approach to the consistency of spectral clustering. *Applied and Computational Harmonic Analysis*, 45(2), 2018.

Nicolás García Trillos, Moritz Gerlach, Matthias Hein, and Dejan Slepčev. Error estimates for spectral convergence of the graph Laplacian on random geometric graphs toward the Laplace–Beltrami operator. *Foundations of Computational Mathematics*, 20(4), 2020.

Charalampos T. Tsitouras. Runge–Kutta pairs of order 5 (4) satisfying only the first column simplifying assumption. *Computers & Mathematics with Applications*, 62(2), 2011.

Zhiyang Wang, Juan Cervino, and Alejandro Ribeiro. A manifold perspective on the statistical generalization of graph neural networks. In *International Conference on Machine Learning*, 2025.

Simon Weber, Thomas Dages, Maolin Gao, and Daniel Cremers. Finsler-Laplace-Beltrami operators with application to shape analysis. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.

Xitong Zhang, Yixuan He, Nathan Brugnone, Michael Perlmutter, and Matthew Hirn. MagNet: A neural network for directed graphs. In *Advances in Neural Information Processing Systems*, 2021.

Appendix A. Finsler Geometry

We treat the ambient space $\mathbb{R}^D$ as an affine space, so that the tangent spaces $T_x\mathcal{M}$ are all isomorphic to $\mathbb{R}^D$, and the ambient space has no curvature. A *Minkowski norm* on $\mathbb{R}^D$ is a continuous function $F : \mathbb{R}^D \to \mathbb{R}^{\geq 0}$ with the following properties for all $v, w \in \mathbb{R}^D, \lambda \geq 0$:

Smoothness: F is smooth away from zero.

Strong convexity: $\nabla^2[F^2(v)] \succ 0$ for $v \neq 0$.

Positive homogeneity: $F(\lambda v) = \lambda F(v)$.

Positive definiteness: $F(v) > 0$ unless $v = 0$.

Unlike a typical norm, we only require homogeneity to hold for $\lambda \geq 0$, rather than for all $\lambda \in \mathbb{R}$; this allows for F to be asymmetric.

If the conditions for the Minkowski norm F only hold on a subset $\mathbb{R}^D \setminus S$ where S is a closed, conic subset of $\mathbb{R}^D$, we say that F is an *almost Minkowski norm* (Davis et al., 2026, Definition 4).

For a linear subspace $U \subset \mathbb{R}^D$, we pullback the Minkowski norm to U via the inclusion map $\mathrm{id} : U \hookrightarrow \mathbb{R}^D$ as $F(u) := (F \circ \mathrm{id})(u)$ for $u \in U$. One can check that this defines a Minkowski norm on U. For a smooth submanifold $\mathcal{M} \subset \mathbb{R}^D$, endowing the tangent bundle of $\mathcal{M}$ with the pullback of the Minkowski norm on $\mathbb{R}^D$ defines a *Finsler manifold* (Bao et al., 2000).

Notably, as shown by Burago and Ivanov (1993, Theorem 3.2), if the Minkowski norm F is odd and $\mathcal{M}$ is compact, then there exists a Minkowski norm F' on $\mathbb{R}^{2d}$ such that $\mathcal{M}$ can be isometrically embedded in $\mathbb{R}^{2d}$.

A.1. Proof of Proposition 3

For $\widehat{J}$ to be the gradient of a scalar-valued energy functional on the cotangent bundle, the Jacobian of $\widehat{J}$ must be a symmetric tensor, since that computes the Hessian of the functional. Specifically,

$$\nabla \widehat{J}(\xi) = W^\top \operatorname{diag}(\sigma'(W\xi))W.$$

Observe that enforcing $V = W^\top$ guarantees that $\nabla \widehat{J}(\xi)$ is symmetric. Moreover, if $\operatorname{rank}(W) = D$ and $\sigma' > 0$, it also holds that the Jacobian of $\widehat{J}$ is positive definite. Moreover, since σ is positive homogeneous of degree 1, the norm (5) is also positive homogeneous of degree 1. If σ is nonsmooth at zero (for instance, if σ is the Leaky ReLU), then we only have smoothness of F at points where none of the entries of $W\xi$ are equal to zero: the set of such ξ is a closed, conic subset of $\mathbb{R}^D$ with measure zero. Hence, we have established the conditions for $\widehat{F}^*$ to be an almost Minkowski norm, as desired.

Appendix B. Empirical Gradient Estimation

We develop a suitable estimate of the gradient of a function f given samples over a finite set of points $\{x_i\}_{i=1}^n$. A first approach is the moving least-squares estimate:

$$\nabla_{\epsilon,n} f(x) = \arg\min_{\xi} \sum_{i=1}^{n} \kappa_\epsilon (x - x_i) \left(f(x) - f(x_i) - \langle \xi, x - x_i \rangle\right)^2.$$

In other words,

$$\nabla_{\epsilon,n} f(x) = \arg\min_{\xi} G_{\epsilon,n}(x, \xi),$$

where

$$G_{\epsilon,n}(x,\xi) = \frac{1}{\epsilon^2} \sum_{i=1}^{n} \kappa_\epsilon (x - x_i) \left(f(x) - f(x_i) - \langle \xi, x - x_i \rangle\right)^2.$$

Setting $\nabla_\xi G_{\epsilon,n}(x,\xi) = 0$ yields the solution

$$\nabla_{\epsilon,n} f(x) = (C_{\epsilon,n}(x))^{-1} b_{\epsilon,n}[f](x)$$

$$C_{\epsilon,n}(x) = \frac{1}{n\epsilon^2} \sum_{i=1}^{n} \kappa_\epsilon (x - x_i) (x - x_i) \otimes (x - x_i)$$

$$b_{\epsilon,n}[f](x) = \frac{1}{n\epsilon^2} \sum_{i=1}^{n} \kappa_\epsilon (x - x_i) \left(f(x) - f(x_i)\right)(x - x_i).$$

We call $\nabla_{\epsilon,n}$ the empirical gradient, $C_{\epsilon,n}$ the empirical covariance, and $b_{\epsilon,n}$ the empirical cogradient.

To account for this, we define

$$\nabla_{\epsilon,n} f(x) = h_\epsilon\left(C_{\epsilon,n}(x)\right) b_{\epsilon,n}[f](x)$$

where

$$h_\epsilon(A) = (A^4 + \epsilon^4 I)^{-1} A^3. \tag{7}$$

This choice of h_ϵ approximates the matrix inverse for parts of the spectrum far from zero, while remaining close to zero for parts of the spectrum close to zero; in other words, it acts as a soft Moore-Penrose pseudoinverse. Hence, we denote $C^\dagger_{\epsilon,n}(x) := h_\epsilon(C_{\epsilon,n}(x))$.

Remark 4 *We show in Appendix D that the covariance matrices $C_\epsilon, C_{\epsilon,n}$ concentrate near weighted projection matrices onto local tangent planes, yielding a rank-d structure. Because of this, h_ϵ was designed to mitigate the effect of small eigenvalues of $C_{\epsilon,n}(x)$ that arise due to sampling effects and the curvature of the manifold at x. In practice, one could replace h_ϵ with a spectral thresholding function, or with a rank-d approximation.*

Appendix C. Cellular (Co)sheaf Laplacians

We provide background on cellular (co)sheaves. For deeper coverage, we suggest the thesis of Curry (2014). A *cellular sheaf* $\mathcal{F}$ on the graph $\mathcal{G}$ assigns

- to each node $x \in \mathcal{X}$ a vector space $\mathcal{F}(x)$,
- to each edge $e \in \mathcal{E}$ a vector space $\mathcal{F}(e)$, and
- to each incident vertex-edge pair (x, e) a linear *restriction map* $\mathcal{F}_{x \to e} : \mathcal{F}(x) \to \mathcal{F}(e)$.

The *dual cosheaf* $\mathcal{F}^*$ on the graph $\mathcal{G}$ assigns

- to each node $x \in \mathcal{X}$ the dual vector space $\mathcal{F}^*(x)$,
- to each edge $e \in \mathcal{E}$ the dual vector space $\mathcal{F}^*(e)$, and
- to each incident vertex-edge pair (x, e) a linear *extension map* $\mathcal{F}^*_{x \to e} : \mathcal{F}^*(e) \to \mathcal{F}^*(x)$.

For a given cellular (co)sheaf $\mathcal{F}$, the space of 0-cochains $C^0(\mathcal{F})$ constitutes the set of all assignments of vectors to each node according to the sheaf structure. That is, a 0-cochain $f \in C^0(\mathcal{F})$ is a map sending each $x \in \mathcal{X}$ to an element of $\mathcal{F}(x)$.

The space of 1-cochains is defined similarly, with additional information to account for the orientation of edges. A 1-cochain $\beta \in C^1(\mathcal{F})$ is a map sending each *oriented edge* $\overrightarrow{e}$ to an element of $\mathcal{F}(e)$, with the property that reversing the orientation changes the sign: $\beta(\overleftarrow{e}) = -\beta(\overrightarrow{e})$.

This yields the differential map from $C^0(\mathcal{F})$ to $C^1(\mathcal{F})$ for a sheaf $\mathcal{F}$ as follows. Let $\overrightarrow{e} = [x, y]$ be an oriented edge. Then, for $f \in C^0(\mathcal{F})$, put

$$(df)(\overrightarrow{e}) = \mathcal{F}_{y \to e}(f(y)) - \mathcal{F}_{x \to e}(f(x)).$$

Similarly, for the dual cosheaf $\mathcal{F}^*$, we have the codifferential map from $C^1(\mathcal{F}^*)$ to $C^0(\mathcal{F}^*)$ defined for $\beta \in C^1(\mathcal{F}^*)$ as

$$(\delta\beta)(x) = \sum_{y \in \mathcal{X}:(x,y)\in\mathcal{E}} \mathcal{F}^*_{x\to e}(\beta([x,y])).$$

We now repeat the construction of the Finsler Laplacian. Let $\mathcal{F}$ and $\mathcal{T}$ be the respective cellular sheaves of *functions* and *vectors* defined in section 3, with corresponding dual cosheaves $\mathcal{F}^*$ and $\mathcal{T}^*$ of *cofunctions* and *covectors*. Let $\mathrm{id} : C^1(\mathcal{F}) \to C^1(\mathcal{T}^*), \mathrm{id}^* : C^1(\mathcal{T}) \to C^1(\mathcal{F}^*)$ be the identity maps under the obvious identification of the spaces. Given that J maps covectors to vectors, it can be extended to a map $J : C^0(\mathcal{T}^*) \to C^0(\mathcal{T})$. Then, we have that the Finsler graph Laplacian is a map $\mathcal{L} : C^0(\mathcal{F}) \to C^0(\mathcal{F}^*)$ defined by composition

$$\mathcal{L}[f] = (\delta \circ \mathrm{id}^* \circ d \circ J \circ \delta \circ \mathrm{id} \circ d)[f],$$

or, as a diagram,

$$\begin{array}{ccccc} C^0(\mathcal{F}) & & C^0(\mathcal{T}^*) & \xrightarrow{J} & C^0(\mathcal{T}) & & C^0(\mathcal{F}^*) \\ \downarrow{\scriptstyle d} & & \uparrow{\scriptstyle \delta} & & \downarrow{\scriptstyle d} & & \uparrow{\scriptstyle \delta} \\ C^1(\mathcal{F}) & \xrightarrow{\mathrm{id}} & C^1(\mathcal{T}^*) & & C^1(\mathcal{T}) & \xrightarrow{\mathrm{id}^*} & C^1(\mathcal{F}^*) \end{array} . \tag{8}$$

Of course, if the vector spaces for the sheaves are inner product spaces, we can identify them with their duals. This allows for the simplification of (8) to the diagram (3).

Appendix D. Proof of Laplacian Convergence

In this section, we prove a collection of results towards Theorem 2. Throughout, we will assume that ϵ is sufficiently small so that it is less than the reach of the manifold $\mathcal{M} \subset \mathbb{R}^D$. For fixed $\epsilon > 0$, it will be useful to consider the empirical Finsler Laplacian in the limiting regime $n \to \infty$. Under mild conditions on the kernel κ and density p, and assuming f is appropriately bounded, we have

$$\begin{aligned} \nabla_\epsilon f(x) &= h_\epsilon\left(C_\epsilon(x)\right)\left(b_\epsilon[f](x)\right) \\ C_\epsilon(x) &= \frac{1}{\epsilon^{d+2}} \int \kappa\left(\frac{x-y}{\epsilon}\right)(x-y)\otimes(x-y)d\mu(y) \\ b_\epsilon[f](x) &= \frac{1}{\epsilon^{d+2}} \int \kappa\left(\frac{x-y}{\epsilon}\right)(f(x)-f(y))(x-y)d\mu(y). \end{aligned} \tag{9}$$

Using this smoothed estimator of the gradient $\nabla_\epsilon f$, we define the smoothed asymptotic energy functional

$$E_\epsilon[f] = \int \frac{1}{2}\left[F^*(\nabla_\epsilon f(x))\right]^2 d\mu(x),$$

and the smoothed Laplacian by taking the Fréchet derivative

$$\Delta_\epsilon[f] = \frac{1}{p} DE_\epsilon[f]. \tag{10}$$

Under the assumption that the kernel κ is radially symmetric (isotropic with respect to the Euclidean norm) about the origin, we evaluate the smoothed Laplacian (10) to

$$\begin{aligned}\Delta_\epsilon[f](x) = \frac{1}{\epsilon^{d+2}} \int \kappa\left(\frac{x-y}{\epsilon}\right) \Big(h_\epsilon(C_\epsilon(x)) J(\nabla_\epsilon f(x)) \\ + h_\epsilon(C_\epsilon(y)) J(\nabla_\epsilon f(y))\Big) \cdot (x-y) d\mu(y).\end{aligned} \tag{11}$$

The strategy will amount to a bound between the difference of $\Delta_{\epsilon,n}[f]$ and $\Delta[f]$, intermediated by $\Delta_\epsilon[f]$ via the triangle inequality. In particular, we will bound the sum

$$|\Delta[f] - \Delta_{\epsilon,n}[f]| \leq |\Delta[f] - \Delta_\epsilon[f]| + |\Delta_\epsilon[f] - \Delta_{\epsilon,n}[f]|.$$

The first term measures the approximation error due to the smoothing parameter ϵ, and the second measures the error due to discretization.

D.1. Local Kernel Moments

Let $x \in \mathcal{M}$ be given. Through a rigid transformation, we choose coordinates such that $x = 0$ and the tangent plane at x occupies the first d coordinates, with the normal plane occupying the remaining $D-d$ coordinates. Denoting the tangent coordinates by $u \in T_x\mathcal{M}$, we represent $\mathcal{M}$ locally as a graph $y_x(u) = (u, g_x(u)) \in \mathbb{R}^D$. The normal coordinates $g_x(u)$ satisfy

$$g_x(u) = \Pi_x[u,u] + D_x[u,u,u] + O(u^4), \tag{12}$$

where Π_x is a symmetric tensor given by the second fundamental form at x, D_x is a third-order tensor, and $O(u^4)$ denotes some fourth-order tensor expression of u. Denoting the tensor product of a vector v with itself k times by $v^{\otimes k}$, we consider expressions of the form

$$M(x,k,\epsilon) = \frac{1}{\epsilon^{d+2}} \int \kappa\left(\frac{y_x(u)}{\epsilon}\right) y_x(u)^{\otimes k} du.$$

First, we carry out the change of variables $v = u/\epsilon$, yielding

$$M(x,k,\epsilon) = \frac{1}{\epsilon^2} \int \kappa\left(\frac{y_x(\epsilon v)}{\epsilon}\right) y_x(\epsilon v)^{\otimes k} dv. \tag{13}$$

We assume that $\kappa(z)$ is a smooth, compactly supported function of $\|z\|^2$, *i.e.*, $\kappa(z) = \kappa(\|z\|^2)$. By a Taylor series expansion centered at a given z, for $w \perp z$, we have

$$\begin{aligned}\kappa(z+w) &= \kappa(\|z\|^2 + \|w\|^2) \\ &= \kappa(\|z\|^2) + \frac{\kappa'(\|z\|^2)}{2}\|w\|^2 + O(\|w\|^4).\end{aligned} \tag{14}$$

D.1.1. Flattening of the Graph

With the preliminary definitions handled, we begin by considering the behavior of $\frac{y(\epsilon v)}{\epsilon}$ as $\epsilon \to 0$. Observe that $\frac{y(\epsilon v)}{\epsilon} = (v, g_x(\epsilon v)/\epsilon)$, so the main quantity of interest is $g_x(\epsilon v)$. By direct substitution into (12), we have

$$g_x(\epsilon v) = \epsilon^2 \Pi_x[v,v] + \epsilon^3 D_x[v,v,v] + \epsilon^4 O(v^4). \tag{15}$$

Note that $g_x(\epsilon v) \perp v$, and the squared-norm satisfies

$$\|g_x(\epsilon v)\|^2 = \epsilon^4 \|\Pi_x[v,v]\|^2 + 2\epsilon^5 \langle \Pi_x[v,v], D_x[v,v,v]\rangle + \epsilon^6 O(v^6).$$

We conclude, by substitution into (14), that

$$\kappa\left(\frac{y(\epsilon v)}{\epsilon}\right) = \kappa(\|v\|^2) + \kappa'(\|v\|^2)\left(\epsilon^2 \|\Pi_x[v,v]\|^2 + 2\epsilon^3 \langle \Pi_x[v,v], D_x[v,v,v]\rangle\right) + O(\epsilon^4), \quad (16)$$

recalling that the compact support of κ allows us to assume $v^6 = O(1)$. That is, $\kappa(y(\epsilon v)/\epsilon) = \kappa(\|v\|^2) + O(\epsilon^2)$, so that the normal coordinates become negligible as $\epsilon \to 0$.

D.1.2. Parity of Local Kernel Moment Integrals

Let $T : (T_x\mathcal{M})^k \to \mathbb{R}$ be a multilinear form for some odd value of k, and consider the integral of $\kappa(\|v\|^2)T[v,\ldots,v]$. Since k is odd, $T[v,\ldots,v]$ is odd, so the integral vanishes by radial symmetry, *i.e.*,

$$\int \kappa(\|v\|^2)T[v,\ldots,v]dv = 0.$$

A similar property holds for κ', so that

$$\int \kappa'(\|v\|^2)T[v,\ldots,v]dv = 0.$$

We define a few quantities of interest for integrals of tensor forms with even parity. The first is the second moment of $\kappa(\|v\|^2)$:

$$m_\kappa I_d = \int \kappa(\|v\|^2)v^{\otimes 2}dv,$$

where I_d denotes the $d \times d$ identity matrix. The second is defined based on the curvature at x:

$$m_{\kappa,x} = \int \kappa(\|v\|^2)\Pi_x[v,v]dv.$$

Note that m_κ is a scalar, but $m_{\kappa,x}$ is a vector normal to x.

D.1.3. Computation of Local Kernel Moments

With these tools in place, computing the moments $M(x,k,\epsilon)$ becomes a trivial exercise. When $k=1$, substituting (16) and (15) into (13) and zeroing all terms with odd parity, we have

$$M(x,1,\epsilon) = (0, m_{\kappa,x}) + O(\epsilon^2).$$

Similarly, when $k=2$, we have

$$M(x,2,\epsilon) = \begin{bmatrix} m_\kappa I_d & 0 \\ 0 & 0 \end{bmatrix} + O(\epsilon^2).$$

D.2. Smoothed Gradient

We consider now the deviation between the smoothed gradient $\nabla_\epsilon f$ and the true gradient ∇f. As before, let $x \in \mathcal{M}$ be given, and apply a rigid transformation so that $x = 0$ and the tangent plane occupies the first d coordinates. We pull back the density p and the function f to $T_x\mathcal{M}$ so that the relevant integrals may locally be defined as integrals over the tangent plane. We rewrite (9) as

$$\begin{aligned} \nabla_\epsilon f(x) &= h_\epsilon\left(C_\epsilon(x)\right) b_\epsilon[f](x) \\ C_\epsilon(x) &= \frac{1}{\epsilon^{d+2}} \int_{T_x\mathcal{M}} \kappa\left(\frac{y_x(u)}{\epsilon}\right) y_x(u)^{\otimes 2} p_x(u) du \\ b_\epsilon[f](x) &= \frac{1}{\epsilon^{d+2}} \int_{T_x\mathcal{M}} \kappa\left(\frac{y_x(u)}{\epsilon}\right) y_x(u)(f_x(u) - f_x(0)) p_x(u) du. \end{aligned} \tag{17}$$

D.2.1. Local Taylor Series

It will be useful to gather some local approximations of p_x and f_x by their Taylor series representations. Taking a Taylor expansion of p_x, we have

$$p_x(u) = p_x(0) + \langle \nabla p_x(0), u\rangle + \frac{1}{2}\nabla^2 p_x(0)[u,u] + \frac{1}{6}\nabla^3 p_x(0)[u,u,u] + O(u^4). \tag{18}$$

Similarly, carrying out a Taylor series expansion of $f_x(u) - f_x(0)$ centered at zero, we have

$$f_x(u) - f_x(0) = \nabla f_x(0)[u] + \frac{1}{2}\nabla^2 f_x(0)[u,u] + \frac{1}{6}\nabla^3 f_x(0)[u,u,u] + O(u^4). \tag{19}$$

D.2.2. Smoothed Covariance

Performing the substitution $v = u/\epsilon$ in (17) yields

$$C_\epsilon(x) = \frac{1}{\epsilon^2} \int_{T_x\mathcal{M}} \kappa\left(\frac{y_x(\epsilon v)}{\epsilon}\right) y_x(\epsilon v)^{\otimes 2} p_x(\epsilon v) dv.$$

Substituting in the Taylor series expansion of the density (18) yields

$$C_\epsilon(x) = \begin{bmatrix} p_x(0) m_\kappa I_d & 0 \\ 0 & 0 \end{bmatrix} + O(\epsilon^2).$$

That is, $C_\epsilon(x)$ is approximately $p(x) m_\kappa I_d$ in the tangent coordinates, apart from an $O(\epsilon^2)$ error term.

D.2.3. Smooth Cogradient

We carry out the same substitution as before, $v = u/\epsilon$, yielding

$$b_\epsilon[f](0) = \frac{1}{\epsilon^2} \int \kappa\left(\frac{y(\epsilon v)}{\epsilon}\right) y(\epsilon v)(f_x(\epsilon v) - f_x(0)) p_x(\epsilon v) dv.$$

Substituting the Taylor series expansions (18) and (19), we have

$$b_\epsilon[f](x) = C_\epsilon(x)\nabla f(x) + O(\epsilon^2). \tag{20}$$

D.2.4. Combined Error

By (20), and using h_ϵ from (7), we have

$$\begin{aligned}\nabla_\epsilon f(x) &= h_\epsilon(C_\epsilon(x))C_\epsilon(x)\nabla f(x) + O(\epsilon^2)h_\epsilon(C_\epsilon(x)) \\ &= \nabla f(x) + O(\epsilon),\end{aligned}$$

where $\nabla f(x) \in \mathbb{R}^D$ by the obvious embedding of the tangent plane. It follows that $J(\nabla_\epsilon f(x)) = J(\nabla f(x)) + O(\epsilon)$.

Since $p(x)$ is bounded away from zero, for sufficiently small $\epsilon > 0$ we have $h_\epsilon(p(x)m_\kappa I_d) \approx \frac{1}{p(x)m_\kappa} I_d$ and $h_\epsilon(O(\epsilon^2)) = O(\epsilon^2)$. In other words,

$$h_\epsilon(C_\epsilon(x)) = \left(\begin{bmatrix} \frac{1}{p(x)m_\kappa} I_d & 0 \\ 0 & 0 \end{bmatrix}\right) + O(\epsilon^2).$$

Putting the two bounds together, this implies

$$h_\epsilon(C_\epsilon(x))J(\nabla_\epsilon f(x)) = m_\kappa^{-1} p^{-1}(x)\nabla f(x) + O(\epsilon). \tag{21}$$

D.3. Smoothed Finsler Laplacian

We begin by bounding the difference between $\Delta[f](x)$ and $\Delta_\epsilon[f](x)$. For fixed x, we apply a rigid transformation as before, so that $x = 0$ and the manifold is a local graph of the tangent plane $y_x(u) = (u, g_x(u))$. For convenience, define the vector field on the tangent plane

$$Q_x[f](u) = \frac{J(\nabla f(y_x(u)))}{p(x)}$$

so that by (21) we have

$$h_\epsilon(C_\epsilon(y_x(u)))J(\nabla_\epsilon f(y_x(u))) = m_\kappa^{-1}\left(Q_x[f](u) + O(\epsilon)\right). \tag{22}$$

Substituting (22) into (11) and rewriting the integral as one over the tangent plane yields, after a change of variables,

$$\Delta_\epsilon[f](0) = \frac{1}{m_\kappa \epsilon^2}\int \kappa\left(\frac{y(\epsilon v)}{\epsilon}\right)\left(Q_x[f](\epsilon v) + Q_x[f](0) + O(\epsilon)\right)\cdot y(\epsilon v)p_x(\epsilon v)dv.$$

We perform a Taylor expansion on the product $(Q_x[f](\epsilon v) + Q_x[f](0) + O(\epsilon))p_x(\epsilon v)$, and applying the same parity arguments as before, simplify this expression to

$$\begin{aligned}\Delta_\epsilon[f](x) &= \operatorname{Tr}\left(p_x(0)\nabla Q_x[f](0) + 2Q_x[f](0)\otimes\nabla p_x(0)\right) + O(\epsilon) \\ &= 2\frac{\langle J(\nabla f(x)), \nabla p(x)\rangle}{p(x)} + p(x)\operatorname{div}\left(\frac{J(\nabla f(x))}{p(x)}\right) + O(\epsilon) \\ &= \frac{1}{p(x)}\operatorname{div}\left(p(x)J(\nabla f(x))\right) + O(\epsilon) \\ &= \Delta[f](x) + O(\epsilon).\end{aligned} \tag{23}$$

Thus, for fixed x and f, $\Delta_\epsilon[f](x)$ is an $O(\epsilon)$-approximant of $\Delta[f](x)$.

D.4. Concentration Inequalities

We now bound the difference between $\Delta_\epsilon[f](x)$ and $\Delta_{\epsilon,n}[f](x)$ where x is assumed to be a sampled point. The expression for the empirical Laplacian (2) closely resembles a Monte Carlo estimate of the smoothed Laplacian (11), with the exception of the smoothed gradients $\nabla_{\epsilon,n} f$ coupling the terms in the sum. To account for this, we introduce an "oracle" empirical Laplacian that has access to the true (smoothed) gradients $\nabla_\epsilon f$, and then bound the difference in two steps via the triangle inequality. In particular, defining

$$\widetilde{\Delta}_{\epsilon,n}[f](x) = \frac{1}{n\epsilon^{d+2}} \sum_{i=1}^{n} \kappa\left(\frac{x - x_i}{\epsilon}\right)\Big(h_\epsilon(C_\epsilon(x))J(\nabla_\epsilon f(x)) + h_\epsilon(C_\epsilon(x_i))J(\nabla_\epsilon f(x_i))\Big)(x - x_i)$$

we consider the bound

$$\Big|\Delta_\epsilon[f](x) - \Delta_{\epsilon,n}[f](x)\Big| \le \Big|\Delta_\epsilon[f](x) - \widetilde{\Delta}_{\epsilon,n}[f](x)\Big| + \Big|\widetilde{\Delta}_{\epsilon,n}[f](x) - \Delta_{\epsilon,n}[f](x)\Big|. \tag{24}$$

D.4.1. Sampling Error

We consider the first term in the bound (24), which is the error due to discretization without the effects of gradient estimation. Observe that $\widetilde{\Delta}_{\epsilon,n}[f](x_j)$ is an average of *i.i.d.* terms bounded by $O(\epsilon^{-(d+1)})$, with variance $O(\epsilon^{-(d+2)})$, and whose expectation is equal to $\Delta_\epsilon[f](x)$. Recalling that J is assumed Lipschitz, Bernstein's inequality for bounded random variables yields the concentration inequality for fixed x:

$$\mathbb{P}\Big\{|\Delta_\epsilon[f](x) - \widetilde{\Delta}_{\epsilon,n}[f](x)| < \frac{t}{\epsilon}\Big\} > 1 - 2\exp\Big(-t^2 n\epsilon^d/3\Big). \tag{25}$$

D.4.2. Covariance Estimation

Since $C_{\epsilon,n}(x)$ is the average of n *i.i.d.* random matrices with operator norm uniformly bounded by $O(\epsilon^{-d})$ and expectation $C_\epsilon(x)$, the matrix Hoeffding inequality implies that for $t \ge 0$,

$$\mathbb{P}\Big\{\|C_{\epsilon,n}(x) - C_\epsilon(x)\| < t\Big\} > 1 - D \cdot \exp\Big(-t^2 n\epsilon^d/2\Big).$$

For $t = O(\epsilon^2)$, this implies

$$\mathbb{P}\Big\{\|h_\epsilon(C_{\epsilon,n}(x)) - h_\epsilon(C_\epsilon(x))\| < t\Big\} > 1 - D \cdot \exp\Big(-t^2 n\epsilon^d/2\Big). \tag{26}$$

D.4.3. Cogradient Estimation

As a consequence of a previous approximation (20), we have

$$\|b_\epsilon[f](x)\| \le p(x) m_\kappa \|\nabla f(x)\| + O(\epsilon^2).$$

For fixed x, the empirical cogradient $b_{\epsilon,n}[f](x)$ is the mean of n *i.i.d.* random vectors with norm uniformly bounded by $O(\epsilon^{-d})$, expected value $b_\epsilon[f](x)$, and variance bounded by $O(\epsilon^{-d})$. The vector Bernstein inequality yields

$$\mathbb{P}\Big\{\|b_{\epsilon,n}[f](x) - b_\epsilon[f](x)\| < t\Big\} > 1 - 2\exp\Big(-t^2 n\epsilon^d\Big). \tag{27}$$

D.4.4. Gradient Approximation Error

We now consider the second term in the bound (24), which is the error in the empirical Laplacian due to estimation of the gradients at the sampled points. By a union bound over (26) and (27), the empirical gradient satisfies

$$\mathbb{P}\Big\{\|\nabla_{\epsilon,n} f(x) - \nabla_\epsilon f(x)\| < O(t)\Big\} > 1 - (2+D)\exp\left(-t^2 n\epsilon^d/2\right),$$

where we assume $t = O(\epsilon^2)$. Since J is assumed Lipschitz, these conditions imply our desired bound. Namely,

$$\mathbb{P}\Big\{\|h_\epsilon(C_{\epsilon,n}(x))J(\nabla_{\epsilon,n} f(x)) - h_\epsilon(C_\epsilon(x))J(\nabla_\epsilon f(x))\| < O(t)\Big\} > 1 - (2+D)\exp\left(-t^2 n\epsilon^d/2\right). \tag{28}$$

D.5. Proof of Theorem 2

For convenience, define

$$\Phi_\epsilon(x) = h_\epsilon(C_\epsilon(x))J(\nabla_\epsilon f(x))$$
$$\Phi_{\epsilon,n}(x) = h_\epsilon(C_{\epsilon,n}(x))J(\nabla_{\epsilon,n} f(x)).$$

For fixed index i, (28) coupled with the law of total probability implies

$$\mathbb{P}\Big\{|\Phi_\epsilon(x_i) - \Phi_{\epsilon,n}(x_i)| < O(t)\Big\} > 1 - (2+D)\exp\left(-t^2(n-1)\epsilon^d/2\right).$$

By a union bound over the indices i, we have

$$\mathbb{P}\Big\{\max_i |\Phi_\epsilon(x_i) - \Phi_{\epsilon,n}(x_i)| < O(t)\Big\} > 1 - (2+D)n\exp\left(-t^2(n-1)\epsilon^d/2\right).$$

These bounds taken together imply

$$|\widetilde{\Delta}_{\epsilon,n}[f](x_j) - \Delta_{\epsilon,n}[f](x_j)| = O(t\epsilon^{-(d+1)})$$

for all indices $1 \le j \le n$ with probability at least $1 - (2+D)n\exp(-t^2(n-1)\epsilon^d/2)$. Taken together with (25), we have

$$|\Delta_\epsilon[f](x_j) - \Delta_{\epsilon,n}[f](x_j)| = O(t\epsilon^{-(d+1)})$$

with probability at least $1 - (n(2+D)+1)\exp(-t^2(n-1)\epsilon^d/2)$. We choose t, ϵ varying with n according to

$$\epsilon_n = \left((4+2\alpha)\frac{\log n}{n}\right)^{\frac{1}{3d+4}}$$
$$t_n = \epsilon_n^{d+2}$$

for some $\alpha > 0$, so that

$$\mathbb{P}\left\{\max_j |\Delta_\epsilon[f](x_j) - \Delta_{\epsilon,n}[f](x_j)| < O(\epsilon_n)\right\} > 1 - O\left(\frac{D}{n^{1+\alpha}}\right).$$

Then, by the bound (23), we have

$$\mathbb{P}\left\{\max_j |\Delta[f](x_j) - \Delta_{\epsilon,n}[f](x_j)| < O(\epsilon_n)\right\} > 1 - O\left(\frac{D}{n^{1+\alpha}}\right).$$

Noting that the empirical estimates are assumed independent, applying the Borel-Cantelli Lemma establishes Theorem 2, as desired.

Table 2: MSE of learned approximations to Finsler heat flow. Standard deviation of MSE over test signals shown for test graph.

Method	Train Graph	Test Graph (std.)
ReLU FGNN	1.9×10^{-5}	4.7×10^{-6} (1.5×10^{-6})
Linear FGNN	3.7×10^{-4}	7.5×10^{-5} (2.2×10^{-5})
ReLU GNN	4.3×10^{-3}	1.3×10^{-1} (1.3×10^{-1})
Linear GNN	2.4×10^{-3}	1.6×10^{-1} (1.6×10^{-1})

Appendix E. Implementation Details and Additional Experiments

In this section, we detail the specifics of the experiments in Section 4. All experiments are implemented in Python using the JAX, Equinox, Optax, and Diffrax libraries (Bradbury et al., 2018; DeepMind et al., 2020; Kidger and Garcia, 2021; Kidger, 2021). We also include additional experiments to show the ability of Finslerian graph neural networks to learn general Finsler metrics.

E.1. Convergence on the Finsler Graph Laplacian

For a torus with major radius $R = 2.0$ and minor radius $r = 1.0$, points are sampled via rejection sampling. Azimuthal and poloidal angles (u, v) are uniformly proposed, and accepted according to the true area (Hausdorff) measure $dA = r(R + r\cos(v))dudv$. We compute $C^{\dagger}_{\epsilon,n}(x)$ by taking the Moore-Penrose pseudoinverse of the best rank-2 approximant of the local covariance matrix $C_{\epsilon,n}(x)$.

E.2. Learning Geometry using Finslerian Graph Neural Networks

The initial conditions $f(0)$ are generated as a superposition of 24 random Fourier modes with frequencies sampled uniformly from $[1.0, 10.0]$ and phases from $[0, 2\pi]$. The drift vector defining the Randers metric is set to $v = [0.25, 0.25]$. The parameterizations of $\widehat{J}(\xi)$ are constructed using a neural network with a single hidden layer of with $E = 64$. The graph neural diffusion equations are integrated from $t = 0$ to $t = 1$. using the Tsitouras 5/4 solver (Tsitouras, 2011), with a time increment of $\Delta t = 0.05$. Models are trained using the Adam optimizer with a learning rate of 3×10^{-4} for 5000 steps, with weight decay. The loss function is the relative Mean Squared Error (MSE) computed exclusively over 20% of the nodes selected uniformly at random.

E.3. Learning a General Finsler Metric

We repeat the experiments in Section 4.2 using a different Finsler metric. Namely, we define $F^*(\xi)$ to be

$$F^*(\xi) = \|\xi\|_3 + \langle \xi, v \rangle,$$

where the drift vector v is the same as before. That is, we replace the usual Euclidean norm in the Randers metric with the ℓ^3 norm. We show the results of this in Table 2, noting similar trends as the original version of the experiment. Additionally, we plot the learned

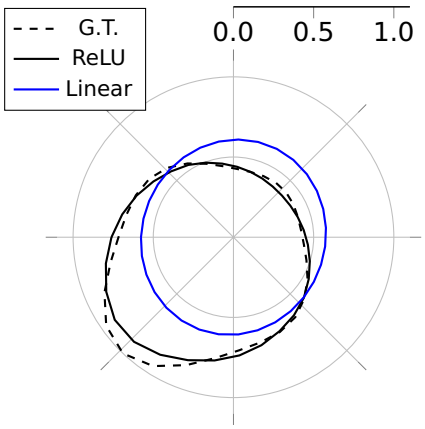

Figure 5: Recovery of a Finsler metric by Finslerian graph neural networks.

unit balls of the Finsler metrics for the nonlinear and linear Finslerian graph neural networks in Figure 5. We note that the metric ball learned by the ReLU network is not as close as observed when learning a Randers metric (Figure 4), which we attribute to the smoothing effect of the estimator mollifying the underlying metric in the data generating process.