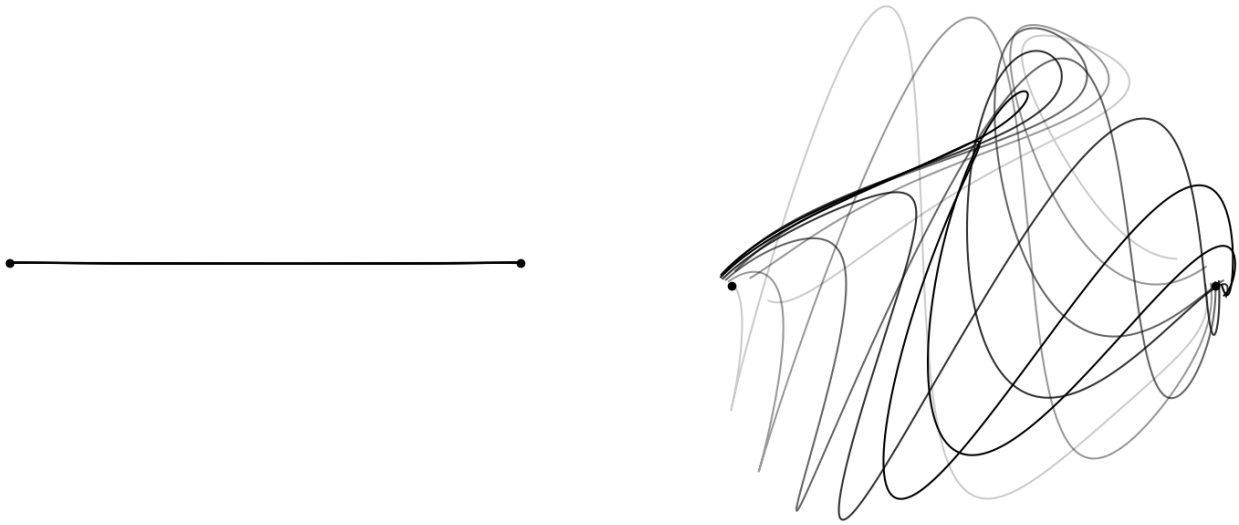
Exploring Diverse Solutions for Underdetermined Problems

Eric Volkmann [*1] **Andreas Radler** [*1] **Johannes Brandstetter** [1 2] **Arturs Berzins** [1]

Abstract

This work explores the utility of a recently proposed diversity loss in training generative, theory-informed models on underdetermined problems with multiple solutions. Unlike data-driven methods, theory-informed learning often operates in data-free settings, optimizing neural networks to satisfy objectives and constraints. We demonstrate how this diversity loss encourages the generation of diverse solutions across various example problems, effectively avoiding mode collapse and enabling exploration of the solution space.



(a) Without diversity loss.

(b) With diversity loss.

Figure 1. A neural network that is trained to represent a continuous set of parametric curves that connect two points. a) A network trained without diversity exhibits mode collapse to a simple geodesic solution. b) When applying a diversity loss, the networks learns to represent diverse curves.

1. Introduction and related work

In the past decade, there have been large advances in generative modeling (Kingma & Welling, 2013; Goodfellow et al., 2020; Rezende & Mohamed, 2015; Tomczak, 2021), made possible by both large increases in the available compute resources and dataset sizes. Generally, generative models are trained to approximate an underlying data-distribution. New samples can then be generated from the learnt distributions. The diversity of the model output is thus derived from the diversity of the training data $\mathcal{D}$.

[*]Equal contribution [1]LIT AI Lab, Institute for Machine Learning, JKU Linz, Austria [2]Emmi AI GmbH, Linz, Austria. Correspondence to: Eric Volkmann <volkmann@ml.jku.at>.

Accepted at Methods and Opportunities at Small Scale (MOSS), ICML 2025, Vancouver, Canada.

However, deep learning models are not only applied in data-rich settings such as language or computer vision but are also used in data-free (or data-sparse) theory-informed settings. Examples include Boltzmann generators (Noé et al., 2019), Physics-informed neural networks (PINNs) (Raissi et al., 2019) and Geometry-informed neural networks (GINNs) (Berzins et al., 2024).

In this setting, training data $\mathcal{D}$ is replaced with an objective o and constraints c_i. The sought solution is parametrized by a (deep) neural net f_θ and the objective $o(f_\theta)$ can then be optimized for, e.g. by minimizing the violation $o(f_\theta)$ of a partial differential equation (PDE) in the case of PINNs.

Notwithstanding the advances in theory-informed learning, many of the optimization problems of interest are underdetermined, meaning there is not a single optimal solution s_{opt} but a (discrete or continuous) set of solutions $\mathcal{S}_{opt}$ that minimize the objective o. This motivates the question of whether (and how) one can train a theory-informed generative model (generator) to sample from $\mathcal{S}_{opt}$?

First attempts at generating multiple solutions for such underdetermined problems include (Zou et al., 2025), who used deep ensembles in the PINN setting. They demonstrated that leveraging the randomness of neural network initialization can be used to discover fluid flow solutions (e.g., Allen-Cahn equation, cavity flow).

GINNs go one step further: in addition to formalizing different geometric and topological losses to train neural signed distance fields (SDFs) in a data-free manner, they introduce an explicit diversity loss to train a data-free shape generator. It is used to avoid mode collapse of the generator and explore the solution space in underdetermined settings.

In this work, we focus on this, in fact, fairly general diversity loss, and demonstrate its utility with illustrative, reproducible examples. We start with reproducing the "horseshoe" example of GINNs and compare their diversity to a formulation by Leinster (2020) on finite point sets. We then proceed with a simple generalization to neural curves in 2D and 3D.

Accompanying this work, a Jupyter Notebook is available on the official MOSS GitHub repository. This notebook provides a step-by-step walkthrough of all experiments presented, and it can be executed within minutes on a standard laptop (without a GPU or accelerator).

2. Methods

Neural representations. Neural representations (NRs) are neural networks with parameters θ that represent a function $f_\theta : x, z \mapsto y$ that maps (optional) a spatial (and/or temporal coordinate) $x \in \mathbb{R}^n$ and a latent vector $z \in \mathbb{R}^l$ to a quantity $y \in \mathbb{R}^m$. Compared to discrete representations, NRs offer superior memory efficiency and are continuous and analytically differentiable functions (Xie et al., 2022). They have achieved notable success across various domains (Park et al., 2019; Mescheder et al., 2019; Mildenhall et al., 2021; Karras et al., 2021).

NRs can easily be used to represent both discrete and continuous sets of solutions using the latent input z, also called the modulation vector. A common approach we also follow is to use a simple multi-layer perceptron (MLP) as a base architecture. Using a one-dimensional modulation code $z \in [0, 1]$ as input, we generate *neural curves* for our experiments in $\mathbb{R}^2$ and $\mathbb{R}^3$ respectively. In experiment 2, we also have an input $x \in \mathbb{R}$, which gives us families of curves.

Constrained optimization is a widely used framework for translating given problem descriptions into a mathematical formulation. The goal is to find a set of optimal solutions or *feasible set* $\mathcal{K}$ within a set of potential solutions $\mathcal{S}$. For neural representations, the potential solutions are functions f within a space $\mathcal{F}$, where f represents either a geometric shape or a solution to a partial differential equation (PDE). More formally, the set of constraints $c_i(f), i \in \{1, ..., M\}$ defines the feasible set $\mathcal{K} = \{f \in \mathcal{F} | c_i(f) = 0\}$. For notational simplicity, inequality constraints are converted to equality constraints.

$$\min o(f) \quad \text{s.t.} \quad c_i(f) = 0, \quad i \in \{1, ..., M\} \tag{1}$$

Measures of diversity. A measure of diversity should encapsulate several intrinsic properties, such as the number of elements and the dissimilarity between elements within a set. Leinster (2020) identifies such properties and rigorously demonstrates that only the Hill numbers $D_q(\mathcal{P})$, where $q \in [0, \infty]$, over a set $\mathcal{P}$ satisfy all these properties simultaneously. In our case, the set $\mathcal{P}$ equals the feasible set $\mathcal{K}$. Leinster (2020) formalizes a diversity measure based on probability distributions over the set P and, importantly, generalizes it to arbitrary pairwise similarity functions $s : \mathcal{P} \times \mathcal{P} \mapsto \mathbb{R}$ defined on elements of $\mathcal{P}$. We refer to Leinster (2020) for an in-depth discussion.

A diversity loss. The diversity loss introduced in GINNs defines a diversity measure δ on a set of functions $\{f_i\}$ as follows

$$\delta_p(\{f_i\}) = \Big(\sum_j \big[\min_{k\neq j} d(f_j, f_k)\big]^{1/p}\Big)^p . \tag{2}$$

where the function d is a dissimilarity function and $\{f_i\}$ typically represent a set of solutions. Intuitively, δ encourages diversity by maximizing the dissimilarity between each shape and its respective most similar shape in the set.

3. Experiments and Results

We demonstrate the effectiveness of the diversity loss (Eq. 2) with 3 illustrative examples. All examples are implemented in the Jupyter notebook accompanying this submission and can be run on a laptop or Google Colab in a few minutes.

Horseshoe. The goal of the horseshoe task is to distribute points evenly within a horseshoe shape. This is an example of applying the diversity loss on the finite set of points $\{Q_i\}$ with $Q_i \in \mathbb{R}^2$. We compare the nearest neighbor diversity loss (Eq. 1) with the Hill numbers defined in Leinster (2020, p. 184) with a similarity matrix $Z_{i,j} = e^{-d(f_i,f_j)}$. The exponent q is in the range $q \in [0,\infty) \setminus \{1\}$. Additionally, we assume equal probability for each element (denoted $\mathbf{p} = \mathbf{1}\frac{1}{N}$ in Leinster (2021)), where $\mathbf{1}$ is a vector where all entries are one. This leads to the hill numbers:

$$D_q^Z = \left(\sum_i^N \frac{1}{N}\big(\sum_j^N Z_{i,j}\frac{1}{N}\big)^{q-1}\right)^{1/(1-q)} . \tag{3}$$

Additionally, we ablate the exponents. The qualitative results are depicted in Figure 2. D_q^Z consistently fails to cover $\mathcal{K}$ and pushes points to the boundary.
Next, we train a neural curve to fit within the horseshoe. We show that with the diversity loss $\delta_{0.5}$, the neural curve explores the feasible set $\mathcal{K}$, as shown in Figure 3. We continue to use $\delta_{0.5}$ as diversity loss for all further experiments.

Parametric curves in 1D. We train a neural network to generate diverse curves, mapping the parameter $x \in [0,1]$ within the *design region* $[-1,1]^2$, connecting the points $(-0.8, 0)$ and $(0.8, 0)$. The diversity loss is computed via a pairwise dissimilarity measure on these curves. Figure 1 shows how the model collapses to a single curve without the diversity loss.

Parametric curve on a sphere. We train a neural network to generate a diverse set of output points $\{y \in \mathbb{R}^3 | f_\theta(z) = y\}$ lying on a sphere with radius r, enforced by a squared distance penalty $\|y\|_2^2 = r$. We use the Euclidean distance as a dissimilarity measure, locally approximating the geodesic on the sphere, and compare its performance against a finite point set optimized for even distribution on the sphere's surface.
The result is shown in Figure 4. The curve without diversity loss exhibits little diversity. In contrast, the curve with explicit diversity loss is much more diverse and wraps around the sphere several times. Since the optimization with diversity loss needs to balance with the spherical loss, the points lie further away from the surface of the sphere.

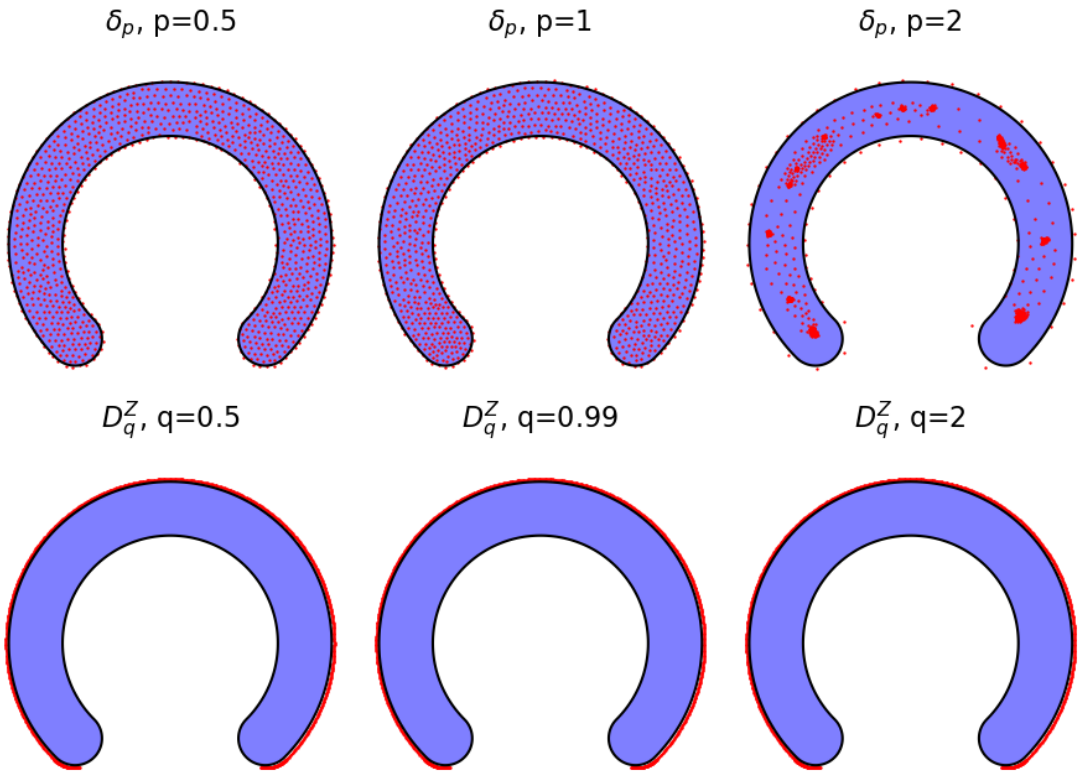
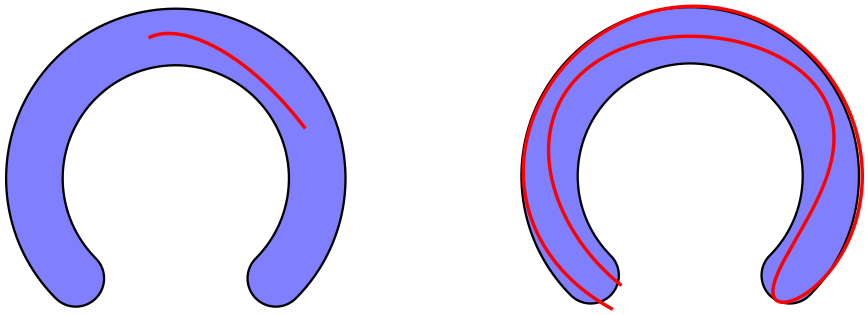
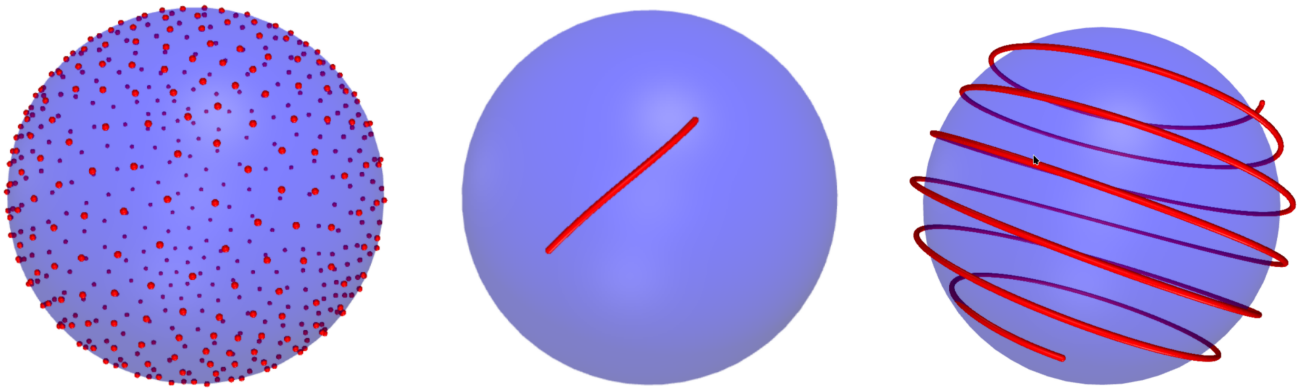


Figure 2. Visual comparison of diversity losses optimizing points Q_i within a horseshoe-shaped feasible set $\mathcal{K}$. The top row shows δ_{min} (Eq. 2), while the bottom row uses D_{sum}^Z (Eq. 3), both with varying exponents $p \in \{0.5, 1, 2\}$. For $0 \leq p \leq 1$, δ is concave, promoting uniform coverage of $\mathcal{K}$, whereas δ_{min}'s convexity for $p \geq 1$ leads to clustering. Conversely, D_{sum}^Z consistently pushes points to $\mathcal{K}$'s boundary.



(a) Without diversity loss. (b) With diversity loss.

Figure 3. A neural network trained to distribute a parametric curve within the horseshoe feasible set $\mathcal{K}$. a) Without diversity, the network produces a short, localized curve. b) With diversity, the network learns to spread the curve throughout $\mathcal{K}$.



(a) Finite point set with diversity. (b) Neural curve w/o diversity loss. (c) Neural curve with diversity loss.

Figure 4. Discover diverse points on a sphere. a) A finite point set optimized for diversity on the sphere surface. a) A network trained without diversity only covers a small area of the sphere's surface. b) With diversity loss, the resulting curve covers a wider area of the sphere's surface, but this is at the expense of fitting the surface less well.

4. Conclusion

In this work, we demonstrated the practical utility of a diversity loss δ in training theory-informed generative models for underdetermined problems. We successfully encourage the exploration of the solution space and generation of discrete and continuous solution sets across examples ranging from point distributions to neural curves.

Our findings underscore the importance of explicitly incorporating diversity mechanisms in data-free learning paradigms, particularly when the objective function admits a manifold of optimal solutions. Future work will focus on extending these principles to more complex problem settings, such as neural fields on manifolds, more efficient optimization of the diversity δ, and learning structured latent spaces, which have gained a prominent role in data-driven generative modeling.

References

Berzins, A., Radler, A., Volkmann, E., Sanokowski, S., Hochreiter, S., and Brandstetter, J. Geometry-informed neural networks. *arXiv preprint arXiv:2402.14009*, 2024.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020.

Karras, T., Aittala, M., Laine, S., Härkönen, E., Hellsten, J., Lehtinen, J., and Aila, T. Alias-free generative adversarial networks. In Ranzato, M., Beygelzimer, A., Dauphin, Y., Liang, P., and Vaughan, J. W. (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 852–863. Curran Associates, Inc., 2021.

Kingma, D. P. and Welling, M. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Leinster, T. Entropy and diversity: The axiomatic approach. *arXiv preprint arXiv:2012.02113*, 2020.

Leinster, T. *Entropy and diversity: the axiomatic approach*. Cambridge university press, 2021.

Mescheder, L., Oechsle, M., Niemeyer, M., Nowozin, S., and Geiger, A. Occupancy networks: Learning 3d reconstruction in function space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4460–4470, 2019.

Mildenhall, B., Srinivasan, P. P., Tancik, M., Barron, J. T., Ramamoorthi, R., and Ng, R. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1):99–106, 2021.

Noé, F., Olsson, S., Köhler, J., and Wu, H. Boltzmann generators: Sampling equilibrium states of many-body systems with deep learning. *Science*, 365(6457):eaaw1147, 2019.

Park, J. J., Florence, P., Straub, J., Newcombe, R., and Lovegrove, S. DeepSDF: Learning continuous signed distance functions for shape representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 165–174, 2019.

Raissi, M., Perdikaris, P., and Karniadakis, G. E. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational physics*, 378: 686–707, 2019.

Rezende, D. and Mohamed, S. Variational inference with normalizing flows. In *International conference on machine learning*, pp. 1530–1538. PMLR, 2015.

Tomczak, J. M. Why deep generative modeling? In *Deep Generative Modeling*, pp. 1–12. Springer, 2021.

Xie, Y., Takikawa, T., Saito, S., Litany, O., Yan, S., Khan, N., Tombari, F., Tompkin, J., Sitzmann, V., and Sridhar, S. Neural fields in visual computing and beyond. *Computer Graphics Forum*, 2022. ISSN 1467-8659.

Zou, Z., Wang, Z., and Karniadakis, G. E. Learning and discovering multiple solutions using physics-informed neural networks with random initialization and deep ensemble. *arXiv preprint arXiv:2503.06320*, 2025.