EFFICIENT AND EXPRESSIVE EQUIVARIANCE DISCOVERY AND ENFORCEMENT USING VECTOR FIELDS

Anonymous authors
Paper under double-blind review

ABSTRACT

Symmetry plays an increasingly important role in the design and analysis of machine learning models, offering improvements in both generalizability and interpretability. Recent work has demonstrated the discovery and enforcement of continuous symmetries using the Lie derivative along vector fields; however, such approaches either do not explore expressive symmetries, are limited to the topic of invariance, or rely on heuristic elbow curve criteria to detect the number of symmetries, leaving nonlinear equivariance and practical scalability underexplored. In this work, we build upon the vector field approach in two significant ways. First, we introduce a PCA-enhanced formulation for discovery that improves numerical stability and computational efficiency, while replacing heuristic elbow curve selection with principled explained variance criteria. Second, we generalize the approach from invariance to equivariance, enabling nonlinear continuous discovery and enforcement in the context of equivariance. Our empirical results demonstrate a distinct computational advantage, being at least an order of magnitude faster than competing methods. We further demonstrate that vector field regularization can improve generalizability under out of sample test distributions, providing a flexible alternative to approaches that enforce equivariance architecturally. Our results thus establish the vector field approach as a scalable, practical, and flexible tool for symmetry-based learning beyond invariance.

1 INTRODUCTION

As the interest in symmetry-informed machine learning increases, so too does the need for efficient, scalable, and expressive means to dispatch symmetry discovery and symmetry enforcement. Symmetries under Lie group actions are of particular interest, and many recent methods draw upon infinitesimal generators. However, as we show, many state of the art methods are unsuitable on the basis of either computational efficiency, scalability, expressivity, or a combination thereof, motivating the introduction of improved methods. In particular, many methods of symmetry discovery appear limited to the topic of invariance, despite broader claims of applicability to equivariance.

This work builds upon the recent use of vector fields to represent the infinitesimal generators of finite Lie group actions, endowing our methods with maximal expressivity. We draw upon recent notions of symmetry discovery characterized as solutions of linear equations, and we show that past caution to solve these equations is without merit. This allows us to introduce a highly efficient method that uses Principal Component Analysis (PCA) and singular decomposition to discover symmetry. Additionally, we explicitly demonstrate the task of equivariance discovery as a separate task from invariance discovery, both from raw data as well as from functions defined by neural networks. We show empirically that our methods outperform many existing methods, and we identify the novel functionality introduced thereby.

Where past work using vector field regularization appear to be limited to the case of invariance or linear equivariance, we demonstrate both linear and nonlinear equivariance regularization. We apply vector field equivariance regularization to the task of Reynolds stress prediction in channel flow data, showing that models trained therewith are quantifiably more equivariant than baseline training. We compare with existing methods of enforcing equivariance, showing that vector field regularization is the most expressive approach, while approaches which enforce exact equivariance (such as Tensor Basis Neural Networks) are usually more quantifiably equivariant.

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

We also show that infinitesimal generators can be used to aid in the task of identifying a basis of equivariant functions by realizing the partial differential equations that define an equivariant function as equations which define a smooth, parallel section on a vector bundle. Our methods, excepting bundle connection construction, are implemented as an installable python package at `https://anonymized-for-peer-review`, which will be officially released following the conference proceedings. Throughout, we refer to our method as *SymmetryML*.

This paper is organized as follows. Section 2 provides abbreviated descriptions of existing literature which most closely relate to our methodology. Section 3 provides some theoretical background information upon which our methods rest, and Section 4 contains the detailed descriptions of our methods. We provide experimental results in Section 5, and Section 6 contains concluding remarks.

2 Related Work

Augerino was an early and insightful approach to symmetry discovery (Benton et al., 2020). A few methods followed, including Desai et al. (2022) and Forestano et al. (2023), though like Augerino, these methods focused on relatively simple symmetries. In the meantime, symmetry discovery using Lie group *infinitesimal generators* began to be more popular, and Dehmamy et al. (2021) acknowledged the vector field nature of the generators but did not explore nonlinear symmetries. Many subsequent symmetry discovery methods used matrix methods, with LieGAN perhaps being the most notable(Yang et al., 2023). LieGAN was later generalized to LaLiGAN to accommodate nonlinear symmetries by attempting to construct a latent space upon which the symmetry is linear (Yang et al., 2024). More recent work has attempted to develop specialized discovery methods when the group is restricted to SO(n) (Karjol et al., 2026). Other work began to emerge which more fully leveraged the nonlinear capabilities of vector field infinitesimal generators, with Ko et al. (2024) using the flow curves through integration and Shaw et al. (2024) avoiding integration by using the Lie derivative. The Lie derivative approach gained traction, with some methods adapting to the dynamical setting (Hu et al., 2025; Je et al., 2024). Frameworks also emerged connecting vector field discovery and enforcement (Otto et al., 2025; Shaw et al., 2025). To our knowledge, there is currently no method in existence which can discover equivariance (as opposed to invariance) for the broad class of symmetries (nonlinear) representable as flows of vector fields.

Among the most familiar constructions for enforcing rotational equivariance is the Tensor Basis Neural Network (TBNN), which are popular in Reynolds stress prediction tasks (Ling et al., 2016; Grogan et al., 2024). More recently, Equivariant Neural Networks (Kondor et al., 2018; Thomas et al., 2018; Weiler et al., 2018), such as e3nn (Geiger & Smidt, 2022), have been employed to enforce equivariance architecturally, with some authors claiming to have obtained performance improvements over traditional TBNNs (Kaszuba et al., 2025; Miller et al., 2026). Other independent works include those of Bergman (2019) and Bloem-Reddy & Teh (2020). There is also work to establish equivariance under any action, assuming reductivity (Batatia et al., 2023). The aforementioned methods dealing with vector field symmetry appear to offer symmetry enforcement for the most general symmetry types which appear in the literature; however, Otto et al. (2025) defers direct, nonlinear symmetry enforcement and discovery, and Shaw et al. (2025) addresses invariance, not equivariance. To our knowledge, there is no prior method which can enforce both invariance and equivariance for the broad class of symmetries expressible as flows of vector fields.

3 Background

Here we provide background information on invariance, equivariance, and vector field infinitesimal generators, as well as establish the notation for the paper.

3.1 Infinitesimal Generators of Left Group Actions on $\mathbb{R}^n$

Let $G \subseteq \mathbb{R}^s$ be a Lie group, and suppose G acts on $\mathbb{R}^n$: that is, for $g_1, g_2 \in G$ and for $x \in \mathbb{R}^n$, there is a function $\mu : G \times \mathbb{R}^n \to \mathbb{R}^n$ such that (notating the group operation using addition)

$$\mu(\mathbf{0}, x) = x, \qquad \mu(g_2, \mu(g_1, x)) = \mu(g_1 + g_2, x). \tag{1}$$

Let $\{v_i\}_{i=1}^s$ be a basis for the tangent space of G at $\mathbf{0}$, the group identity element. Lastly, let σ be a curve in G for which $\sigma(t_0) = \mathbf{0}$ and $\dot{\sigma}(t_0) = v_i$ for $t_0 \in \mathbb{R}$. The infinitesimal generator X_i

corresponding to v_i is given by

$$X_i = \left(\frac{d}{dt} \mu(\sigma(t), x) \right) \Big|_{t=t_0}. \tag{2}$$

For example, consider the group $G = \mathbb{R}^3$ acting on $\mathbb{R}^2$ via

$$\mu((a, b, \theta), (x, y)) = (x\cos(\theta) - y\sin(\theta) + a, x\sin(\theta) + y\cos(\theta) + b)\,.$$

Given the following three curves,

$$\sigma_a(t) = (t, 0, 0), \qquad \sigma_b(t) = (0, t, 0), \qquad \sigma_\theta(t) = (0, 0, t),$$

we find that

$$X_a = \frac{d}{dt}\,(t, 0)\,|_{t=0} = \partial_x, \qquad X_b = \frac{d}{dt}\,(0, t)\,|_{t=0} = \partial_y,$$

$$X_\theta = \frac{d}{dt}\,(x\cos(\theta) - y\sin(\theta), x\sin(\theta) + y\cos(\theta))\,|_{t=0} = -y\partial_x + x\partial_y.$$

We also note that vector field infinitesimal generators are *operators* on the space of smooth, real-valued functions: if X is an infinitesimal generator of μ, and if f is a (smooth) real-valued function on $\mathbb{R}^n$, then

$$X(f) = \sum_{i=1}^{n} \alpha^i \frac{\partial f}{\partial x^i},$$

where $\{\alpha^i\}_{i=1}^n$ denote the components of X.

3.2 Invariant and Equivariant Left Group Actions

Let G be a Lie group, and let $\mu_X : G \times \mathbb{R}^n \to \mathbb{R}^n$ and $\mu_Y : G \times \mathbb{R}^m \to \mathbb{R}^m$ be smooth (left) actions of G on $\mathbb{R}^n$ and $\mathbb{R}^m$, respectively. A smooth map $f : \mathbb{R}^n \to \mathbb{R}^m$ is (G, μ_X, μ_Y)-*equivariant* if

$$f\big(\mu_X(g, x)\big) \;=\; \mu_Y\big(g, f(x)\big)$$

for all $g \in G$, $x \in \mathbb{R}^n$. The function f is (G, μ_X)-*invariant* if

$$f(\mu_X(g, x)) = f(x)$$

for all $g \in G$, $x \in \mathbb{R}^n$. Often, invariance is considered to be a special case of equivariance in which μ_Y is trivial. However, we prefer to keep the concepts distinct, owing to the distinction between the construction of invariant/equivariant functions from other invariant/equivariant functions.

The set of functions which are (G, μ_X)-invariant has the structure of a ring under ordinary function addition and multiplication: that is, the sum/multiplication of two invariant functions is also invariant. Additionally, any function of an invariant function is also invariant. Therefore, an invariant model can be constructed by transforming to a feature space using a collection of known invariants of a given group action (Shaw et al., 2024).

The structure of equivariant functions is distinct from the structure of invariant functions. The set of functions which are (G, μ_x, μ_Y)-equivariant has the structure of a module over the ring of invariant functions. In particular, there is not generally a notion of composition of two equivariant functions which yields an equivariant function. Therefore, to construct equivariant models, it is not sufficient merely to obtain equivariant functions as is the case with invariance.[1] Because of the difference in constructing like-functions for symmetry enforcement in models, we maintain a distinction between invariance and equivariance in the context of machine learning.

[1] Some approaches to enforce equivariance, known as Tensor Basis models, express the predicted quantity as a linear combination of known equivariant functions, having transformed the model inputs data using known invariant functions (Grogan et al., 2024; Ling et al., 2016), .

3.3 VECTOR FIELD EQUIVARIANCE

It has been established that a (smooth) function f is (G, μ_X)-invariant if and only if it is invariant under the collection of infinitesimal generators of μ_X (Shaw et al., 2025). An analogous vector field condition can be given for equivariance. As before, let G be a Lie group, and let $\mu_X : G \times \mathbb{R}^n$ and $\mu_Y : G \times \mathbb{R}^m$ be left actions on $\mathbb{R}^n$ and $\mathbb{R}^m$, respectively. Let $\{X_i\}_{i=1}^s$ be a basis of infinitesimal generators of μ_X derived from the tangent space basis $\{v_i\}_{i=1}^s$ of G at the group identity element, and let $\{Y_i\}_{i=1}^s$ be a corresponding basis of infinitesimal generators of μ_Y. A (smooth) function $f : \mathbb{R}^n \to \mathbb{R}^m$ is (G, μ_x, μ_Y)-equivariant if and only if (component-wise)

$$X_i(f) = Y_i \circ f \tag{3}$$

for each index i. A similar framing of the vector field equivariance condition is given in Otto et al. (2025).

For example, consider the function $f(x, y) = (y, x)$, $G = \mathbb{R}$, and $\mu_X(t, (x, y)) = \mu_Y(t, (x, y)) = (xe^t, ye^t)$. We see that $s = 1$ and $m = n$, so that $X = x\partial_x + y\partial_y = Y$ is the single input/output generator. We find that (component-wise)

$$X(f) = (X(y), X(x)) = (y, x),$$

so that the associated vector field is $y\partial_x + x\partial_y$. Meanwhile, the evaluation of Y at the output of f yields

$$Y \circ f = (y)\partial_x + (x)\partial_y.$$

Thus, as the components of $X(f)$ and $Y \circ f$ are equal, f is (G, μ_X, μ_Y)-equivariant.

4 METHODS

In this section, we provide the methodology for our experimental results. Section 4.1 describes our PCA-based approach to level set estimation, though we reserve the discussion on the justification for symmetry discovery for Section 4.2.3.[2] In Section 4.2, we discuss the PCA-based approach for discovering invariance (4.2.1) as well as equivariance (4.2.2).

4.1 USING PCA TO GET THE LEVEL SET

Level set estimation refers to the task of learning a nontrivial function $f : \mathbb{R}^d \to \mathbb{R}^q$ such that $f(x) = 0$ for each data point $x \in \mathbb{R}^d$. The learned function then models the embedded submanifold upon which the data resides under the manifold assumption. In Shaw et al. (2024), this problem is formulated as selecting a basis of functions $\{h_j\}_{j=1}^m$ such that every component function $\{f_k\}_{k=1}^q$ of f is expressed as a linear combination:

$$f_k = \sum_{j=1}^{m} v_{kj} h_j.$$

Thus, this approach to level set estimation initially reduces to the problem of computing the nontrivial nullspace of the feature matrix $M_{ij} = h_j(x_i)$ for each x_i in the dataset $\{x_i\}_{i=1}^N$. However, the authors of Shaw et al. (2024) avoided evaluating the nullspace using PCA, since this approach could lead to degenerate component functions. For example, for $d = 2$ and $||x_i|| = 1$, a PCA-based approach using a cubic basis of functions could yield component functions

$$x^3 + xy^2 - x, \quad yx^2 + y^3 - y$$

in addition to the familiar $x^2 + y^2 - 1$. Without pruning, these degenerate components taint the level set estimate, and the (estimated) intrinsic dimension of the level set cannot simply be computed from the number of components of the estimated level set function. The authors of Shaw et al. (2024) attempt to navigate this difficulty by suggesting an iterative approach using elbow curve heuristics.

[2]We point out, however, that level set estimation is only one means by which symmetry discovery can take place: in general, we can examining the symmetry of any smooth function.

However, the philosophy employed herein is that, for certain applications, the fact of obtaining degenerate components in a level set estimate is not a confounding error, which we explain in the context of symmetry discovery in Section 4.2.3. Therefore, we obtain our level set estimates first by making the additive/basis assumption described above, but the nontrivial solutions to the nullspace of M are obtained using the simpler method of collecting the "least Principal Components"–that is, the Principal components with the smallest singular values. The number of components can then be "automatically" selected by specifying a threshold for singular values or else a threshold for the proportion of the sum of all singular values. Herein, we employ the latter.

4.2 PCA FOR INVARIANCE AND EQUIVARIANCE

4.2.1 INVARIANCE

The authors of Shaw et al. (2024) discover symmetries, as in obtaining level set estimation, by finding nontrivial nullspace solutions. The matrix for which nullspace solutions are found is known therein as the extended feature matrix, which can be defined as follows. Suppose a vector-valued function $F : \mathbb{R}^d \to \mathbb{R}^m$ admits a well-defined Jacobian matrix J at each of the N points x_i in a dataset, and that a chosen basis of vector fields $\{X_s\}_{s=1}^q$ is given. We define $\mathcal{J}_{i,j,k} = J_{j,k}(x_i)$, and $\mathcal{X}_{sik} = (X_s(x_i))_k$ (the kth component of X_s at the point x_i). We define

$$A_{i,j,s} = \sum_{k=1}^{d} \mathcal{J}_{i,j,k}\mathcal{X}_{sik}, \tag{4}$$

then reshape $A_{i,j,s}$ into a matrix $A_{r,s}$ by defining the index $r = (i-1)m + j$. The resulting matrix $A_{r,s}$ is the extended feature matrix, and nullspace solutions are taken to represent components v_s in solutions of the form $X = \sum_{s=1}^q v_s X_s$.

The degeneracy of vector field solutions arises from the fact that $X(f) = 0$ if and only if $hX(f) = 0$ for a smooth function f and a function h. The authors of Shaw et al. (2024) primarily avoided this degeneracy by restricting the basis to that of a Lie algebra, so that X and hX are not both present in the chosen basis.[3] We retain this approach here. However, the authors of Shaw et al. (2024; 2025) still employed elbow curve heuristics to estimate the nullspace, whereas we use the same PCA-based approach as for level set estimation. This approach then mirrors the approach taken in Hu et al. (2025), except that our vector field basis, while technically capable of being defined by arbitrary function coefficients, defines a Lie algebra basis in practice.

4.2.2 EQUIVARIANCE

A key innovation of this work is to extend this efficient methodology to the subject of equivariance. Section 3.3 makes explicit our assumption that the input vector fields X_s and the output vector fields Y_s correspond bijectively. An example of this correspondence is with diagonal group actions (see Appendix C.3), though we emphasize that SymmetryML is not restricted to these.

Under the assumption of correspondence, it is clear that a linear combination $\sum_s v_s X_s$ corresponds with a linear combination $\sum_s vsY_s$. Borrowing the definitions from Section 4.2.1 (vector-valued F with Jacobian $\mathcal{J}$, as well as the Extended Feature Matrix), we write Equation (3) as

$$X_s(F) - Y_s \circ F = 0. \tag{5}$$

We define $\mathcal{Y}_{si\tilde{k}} = (Y_s(F(x_i)))_{\tilde{k}}$ (the $\tilde{k}^{th}$ component of Y_s at the point $F(x_i)$). We define $B_{r,s}$ by reshaping $\mathcal{Y}_{si\tilde{k}}$, defining $r = (i-1)m + j$. We then define the *Equivariant Residual Matrix* as

$$C_{r,s} = A_{r,s} - B_{r,s}, \tag{6}$$

where $A_{r,s}$ is the Extended Feature Matrix defined in Section 4.2.1. The non-trivial nullspace of $C_{r,s}$ define coefficients v_{sp}, which in turn define P pairs of linear combinations $\hat{X}_p = \sum_{s=1}^q v_{sp} X_s$ and $\hat{Y}_v = \sum_{s=1}^q v_{sp} Y_s$ for $1 \le p \le P$. The pairs $\{(\hat{X}_p, \hat{Y}_p)\}_{p=1}^P$ define subspaces of $\{X_s\}$ and $\{Y_s\}$ under which the function F is (G, μ_X, μ_Y)-equivariant.

In practice, the nontrivial nullspace of the Equivariant Residual Matrix is computed as with the Extended Feature Matrix and, for level set estimation, the feature matrix: identifying principal components with low variance.

[3] Examples of interesting bases include Killing vectors, conformal Killing vectors, and projective vectors.

4.2.3 On Symmetry Detection with Degenerate Level Set Components

Both the extended feature matrix (invariance) and the equivariant residual matrix (equivariance) discussed in Sections 4.2.1 and 4.2.2, respectively, require the specification of the Jacobian matrix of a (smooth) function F. In the context of learning symmetries from data via level set estimation, this raises the question implicit in previous work (Shaw et al., 2024; 2025) as to how one can expect symmetry detection to take place at all, since the level set estimate contains degenerate components when computed using PCA to approximate the nullspace of the feature matrix, as described in Section 4.1. But the fact that the level set function F contains degenerate components is precisely the condition that the Jacobian matrix is rank-deficient: therefore, the degenerate level set components do not contribute unique basis vectors to the tangent space of F, so that the space of vectors at each point that are orthogonal to the tangent space (i.e., the vectors that define symmetry of F) remains the same. Thus, though our level set estimate is not potentially as mathematically faithful to the data itself, the degenerate components of our level set estimates do not materially impact the task of symmetry discovery.

4.3 Enforcing Equivariance

4.3.1 Exact Equivariance

An established method of training machine learning models that are guaranteed to be invariant under a (continuous, multi-parameter) left group action consists of pre-computing functions that are invariant under the infinitesimal generators, then transforming the data using the invariant functions, thus defining a new feature space on which any learned model is, by construction, invariant (Shaw et al., 2024). This method operates on the assumption that a maximal set of invariant functions can be given. For equivariance, an analogous method can be pursued, which amounts to the pursuit of a basis of a module over the ring of invariant functions, as discussed in Section 3.2. This construction is not new, and the result of identifying the basis of equivariant functions allows one to construct a Tensor Basis Neural Network (Ling et al., 2016), but adapted to the specific symmetry at hand.

As in the invariant case, the identification of a basis of equivariant functions for a given set of vector field infinitesimal generators is non-trivial. In fact, there are examples of groups for which no finite basis can be given (Batatia et al., 2023). However, a vector field approach to equivariance provides additional mathematical machinery in the pursuit of equivariant bases. This is particularly true when the group action can be represented in the output space such that the infinitesimal generators have linear components. In this case, the system of equations given by (3) is linear, and it may be possible, through a process known as prolongation, to reinterpret the system as the equations that define a parallel section of a vector bundle with respect to a constructed bundle connection (Shaw, 2021; Gover & Leistner, 2018). We provide an example of this construction in Appendix B.

It should also be noted that the mechanism fails when the curvature of the bundle connection vanishes identically, giving no nonzero matrices to obtain the reduced equations (Shaw, 2021).

4.3.2 The Vectorization of the Conjugate Action of $SO(3)$ on Square Matrices of Size 3×3

Symmetry-enhanced machine learning finds concrete application in modeling turbulence closures within the field of computational fluid dynamics (Ling et al., 2016). Therein, the Reynolds stress tensor τ, whose components are representable as a 3×3 matrix τ_{ij} in a given coordinate system (which we assume is Euclidean-like), is commonly assumed to be a pointwise function of velocity gradients, represented as ∇U or, as a matrix, ∇U_{ij} (that is, the jth partial derivative of the ith velocity component): $\tau_{ij}(x) = f(u_{ij}(x))$. The machine learning goal is to learn f. However, a symmetry arising from the Navier Stokes equations is the equivariance under rotations of the Reynolds stress tensor:

$$\tau\left(Q\nabla U Q^T\right) = Q\tau\left(\nabla U\right)Q^T, \tag{7}$$

where $Q \in SO(3)$, which is to say that the learned function of the velocity gradient should be equivariant under the $SO(3)$ action of conjugation.

While vector field equivariance primarily applies to vector-valued functions, it applies to matrix-valued functions by realizing the conjugate action as a group action on the "vectorized" or "flattened"

space. For an arbitrary 3×3 matrix M with values m_{ij} (row, column), we have

$$\text{vec}(M) = \begin{bmatrix} m_{11} & m_{12} & m_{13} & m_{21} & m_{22} & m_{23} & m_{31} & m_{32} & m_{33} \end{bmatrix}^T.$$

Therefore, we find the infinitesimal generators of the action on the vectorized input space (∇U) via

$$X_i = \frac{\partial}{\partial t} \text{vec}\left(Q_i \nabla U Q_i^T\right)\Big|_{t=0},$$

where Q_i corresponds to a single-parameter subgroup of SO(3), t is the parameter for said subgroup, and $t = 0$ signifies the point at which $Q_i(0)$ is the identity matrix. The three vector field infinitesimal generators are derived explicitly in Appendix C.2.

5 Experimental Results

The first of our results benchmarks several existing methods using simple synthetic experiments. The primary purpose is to compare wall times, but it is also intended to serve as a simple benchmark in light of the notion that some existing methods fail to recover the ground truth symmetries in simple scenarios (Yang et al., 2023). The second experiment, in Section 5.2, demonstrates equivariance discovery of the 10 Pope basis tensors. In Section 5.3, we compare vector field regularization against a non-symmetric baseline, e3nn, and a Tensor Basis Neural Network (TBNN). Further experimental results can be found in Appendix A. In Appendix A.1, we vary the sample sizes for the experiment in Section 5.1, Appendix A.2 considers the task of symmetry discovery from pre-trained models, and we explore a high-dimensional example in Appendix A.3, showing that SymmetryML can be used to recover the correct symmetry group dimension even when the dimension is over 1000.

We provide code necessary to reproduce our results as supplementary files. Therein, details relating to model architectures and training hyperparameters can be directly observed. We refer the reader to the Reproducibility Statement below for additional information.

5.1 Surfaces of Revolution

In the first experiment, we benchmark SymmetryML against several others in a variety of synthetic experiments in which the ground truth symmetry is known. Each dataset or "Object" is a familiar surface of revolution. For each dataset, we obtain $40,000$ samples via uniform manifold sampling, (described in Appendix C.1). Our results are summarized in Table 1.

Table 1 contains four metrics for each dataset and method. The first metric is the recovered dimension of the symmetry group, which measures the ability of each method to recover the correct number of independent symmetries. This metric only applies to SymmetryML and LieNLSD, as LieGAN and LaLiGAN do not offer a mechanism to discover the symmetry dimension automatically and are supplied with the ground truth number. The second metric is the Grassman distance used in existing literature (Hu et al., 2025), and the Projected Frobenius metric is specially designed for distances between subspaces of different dimensions (Ye & Lim, 2016). The fourth metric is simply the wall time for each method, a key scalability metric.

LieNLSD does not natively support symmetry discovery of data merely from its distribution/shape. Instead, the authors target symmetries of trained model functions (Hu et al., 2025). However, the method of Hu et al. (2025) can be combined with the level set estimation of Shaw et al. (2024) to detect symmetry in such settings, which we do in this experiment.

The results in Table 1 demonstrate that SymmetryML is generally more accurate than existing methods, and that it is at least an order of magnitude faster.

5.2 Equivariant Discovery from the Pope Basis Tensors

We now consider the task of discovering a basis of vector fields under which each of the 10 Pope basis tensors is equivariant (Ling et al., 2016). As explained in Section 4.3.2, our convention is to vectorize the SO(3) action on matrices of size 3×3, so that we represent each Pope basis tensor as a vector-valued function $F_i : \mathbb{R}^9 \to \mathbb{R}^9$. As we seek symmetries of each, we construct a function $F : \mathbb{R}^9 \to \mathbb{R}^{90}$ which is the concatenation of all vectorized basis tensors. The search space of vector

Table 1: Surface-of-revolution symmetry discovery results at $N = 40{,}000$ samples (mean $\pm$ standard deviation). Lower is better for the Grassmann and Projected Frobenius metrics. Top scores are **bolded** with the runner-up being underlined.

Object	Metric	SymmetryML	LieNLSD	LieGAN	LaLiGAN
Sphere	Recovered dim. (3)	3.00 ± 0.00	3.00 ± 0.00	-	-
	Grassmann	$(\mathbf{1.56 \pm 1.37}) \times \mathbf{10^{-8}}$	0.0014 ± 0.0009	0.014 ± 0.004	0.371 ± 0.641
	Proj. Frobenius	$(\mathbf{2.40 \pm 2.17}) \times \mathbf{10^{-8}}$	0.0020 ± 0.0013	0.020 ± 0.006	0.473 ± 0.816
	Time (s)	$\mathbf{2.26 \pm 0.21}$	56.77 ± 7.20	109.60 ± 4.89	392.18 ± 12.07
Circle	Recovered dim. (1)	1.00 ± 0.00	1.00 ± 0.00	-	-
	Grassmann	$\mathbf{0.000}$	0.0013 ± 0.0003	0.0050 ± 0.0023	1.219 ± 0.344
	Proj. Frobenius	$\mathbf{0.000}$	0.0019 ± 0.0004	0.0072 ± 0.0031	1.276 ± 0.187
	Time (s)	$\mathbf{2.26 \pm 0.11}$	141.99 ± 6.32	131.20 ± 15.39	398.83 ± 14.66
Cylinder	Recovered dim. (2)	2.00 ± 0.00	4.00 ± 1.00	-	-
	Grassmann	$(\mathbf{1.20 \pm 1.08}) \times \mathbf{10^{-8}}$	1.166 ± 1.046	1.383 ± 0.322	1.716 ± 0.106
	Proj. Frobenius	$(\mathbf{1.70 \pm 1.53}) \times \mathbf{10^{-8}}$	1.977 ± 0.236	1.391 ± 0.187	1.782 ± 0.057
	Time (s)	$\mathbf{0.89 \pm 0.05}$	136.64 ± 3.45	122.07 ± 2.78	218.09 ± 14.83
Torus	Recovered dim. (1)	1.00 ± 0.00	1.00 ± 0.00	-	-
	Grassmann	$\mathbf{0.000}$	0.0014 ± 0.0015	0.0154 ± 0.0007	0.989 ± 0.818
	Proj. Frobenius	$\mathbf{0.000}$	0.0019 ± 0.0021	0.0217 ± 0.0010	0.956 ± 0.762
	Time (s)	$\mathbf{3.48 \pm 0.23}$	71.25 ± 0.16	206.34 ± 3.88	185.94 ± 1.15
Mexican Hat	Recovered dim. (1)	1.00 ± 0.00	1.00 ± 0.00	-	-
	Grassmann	0.005 ± 0.000	$(\mathbf{7.9 \pm 1.0}) \times \mathbf{10^{-4}}$	0.0176 ± 0.0067	1.239 ± 0.380
	Proj. Frobenius	0.008 ± 0.000	$\mathbf{0.0011 \pm 0.0001}$	0.0249 ± 0.0096	1.275 ± 0.218
	Time (s)	$\mathbf{3.47 \pm 0.14}$	71.72 ± 2.15	110.56 ± 6.34	179.76 ± 3.02

fields on the input space is the set of all Euclidean Killing vectors in 9-dimensions. The output space by construction is 10 copies of $\mathbb{R}^9$, so that the induced action on the output space is merely the diagonal action of the input space action, giving us a bijective correspondence between input vector fields X_j and output vector fields Y_j (see Appendix C.3 for information about infinitesimal generators of diagonal group actions).

We generate 1000 data points in the input space such that each of the nine components is sampled from the uniform distribution with support $[0, 1]$. The gradient of F is not computed analytically, but rather estimated using automatic differentiation. SymmetryML discovers (a constant scaling of) the following three symmetries:

$$R_{0,2} + R_{3,5} + R_{2,8} + R_{0,6} + R_{1,7} + R_{6,8}, \quad R_{3,6} + R_{4,5} + R_{1,2} + R_{5,8} + R_{7,8} + R_{4,7},$$
$$R_{3,4} + R_{0,3} + R_{2,5} + R_{6,7} + R_{0,1} + R_{1,4},$$

which exactly align with X_2, X_3, and X_1 defined in Section 4.3.2. The singular values associated with these generators are, respectively, $7.6875 \cdot 10^{-6}$, $8.9746 \cdot 10^{-6}$, and $3.2573 \cdot 10^{-5}$, while the next smallest singular value is $2.5861 \cdot 10^{1}$. This serves as a demonstration that our symmetry discovery methodology successfully extends to equivariance.

5.3 Equivariance Enforcement

In this experiment, we consider the task of equivariance enforcement in two cases: one in which the underlying symmetry is linear, allowing us to compare with additional methods, and another in which the symmetry is nonlinear, allowing only a baseline comparison, highlighting the methodological gap that we fill with nonlinear vector field equivariance enforcement, done by using Equation (3) as the regularization penalty term. A summary of our results is given in Table 2.

The first dataset concerns that of Reynolds stress prediction from velocity gradients, which is a common application of equivariant model learning in which the group action is rotation (Ling et al., 2016). We consider a flow in a square conduit, with the flow being periodic in the streamwise direction x (McConkey et al., 2021). The periodic assumption simplifies the problem so that the input space is two-dimensional and the group action is one-dimensional, being a rotation in the (y, z) plane. We train on data for which $y > 0$ and test on data for which $y <= 0$. The infinitesimal generators used for regularization are the generators X_1, X_2, and X_3 given in Sections 5.2 and 4.3.2.

he "Square Duct" row of Table 2 shows that vector field regularization (our method) obtains a higher test score than the baseline neural network (same MLP architecture), but that the Tensor Basis Neural

Network obtains the highest score. This suggests that vector field regularization tends to yield results between exact and no equivariance enforcement in terms of performance, which is expected.[4]

We refer to the second dataset as "Projective." In this setting, we seek a function $F : \mathbb{R}^3 \to \mathbb{R}^3$ which is equivariant under the action defined by the following vectors, which are projective vectors of the Euclidean metric:

$$P_1 = x^2\partial_x + xy\partial_y + xz\partial_z, \quad P_2 = xy\partial_x + y^2\partial_y + yz\partial_z, \quad P_3 = xz\partial_x + yz\partial_y + z^2\partial_z.$$

We sample 1000 data points uniformly from a conic section defined in spherical coordinates by $1 < \rho < 1.01$, $\pi/6 < \theta < \pi/3$, and $\pi/6 < \varphi < \pi/3$, which is then transformed to cartesian coordinates (x_1, x_2, x_3). The ground truth function (f_1, f_2, f_3) is defined via

$$f_i = \frac{x_i}{\arctan(x_3/x_1)\arctan(x_2/x_1)x_1 + 1},$$

which is verifiably equivariant. For testing, 1000 points are sampled from the same distribution, but with $3 < \rho < 5$: the simulated scenario is that the training data alone does not supply a sufficient amount of information in order to inform how to extrapolate for larger ρ. The results in Table 2 demonstrate that vector field regularization is stronger than the baseline.

Table 2: Test R^2 scores (mean $\pm$ std) for equivariance enforcement tasks

Dataset	**Baseline**	**Regularization**	**e3nn**	**TBNN**
Square Duct	0.7349 ± 0.0147	0.8865 ± 0.0045	0.7785 ± 0.0001	0.9701 ± 0.0053
Projective	-3.3521 ± 0.5119	0.9178 ± 0.0598	–	–

This experiment suggests that vector field equivariance regularization, even in the nonlinear setting, can improve baseline performance. It also suggests that in tasks for which enforcing exact equivariance is both appropriate and practical, exact equivariance likely outperforms vector field regularization. Thus, vector field regularization is useful primarily when enforcing exact equivariance is either impractical or when soft enforcement is more appropriate. Vector field regularization may also be preferred when enforcing symmetry architecturally is not desirable, as in transfer learning.

6 Concluding Remarks

In this paper, we have introduced a novel and effective means of discovering and enforcing equivariance for nonlinear group actions. To our knowledge, there are no alternative methods for these tasks. For discovery, and where existing methods can be applied, SymmetryML handily outperforms all others in terms of computational efficiency (wall time), while the scoring metrics remain highly competitive. We also demonstrated that vector field regularization, both in the linear and nonlinear cases, improves generalization error over the baseline.

Future work includes improving the computational efficiency of vector field regularization, as the key computational bottleneck is the need to compute the gradient of the model with respect to the input parameters. Herein, we have used automatic differentiation to obtain model gradients. However, this introduces a severe bottleneck that we have found tends to increase training time approximately five fold. Additional future work includes the application to many more symmetry groups and nonlinear representations, for which task we have shown SymmetryML is well-equipped.

AI use statement

In this work, we used generative AI tools to assist in implementing our methods in the python programming language, synthetic data generation, and cleaning/reformatting datasets. We have not used generative AI tools for the following tasks: developing theoretical models or conceptual frameworks, formulating mathematical claims, providing critical ingredients for proving mathematical claims, assisting in the writing of proofs, proposing/refining hypotheses, designing/providing feedback on research methodology or experiments, assisting with translation, supporting qualitative and thematic data analysis, or interpreting results.

[4] Although the result for e3nn is worse than expected for our attempt to employ it in this setting, it should not be taken as conclusive evidence that e3nn is not performative.

Ethics statement

We acknowledge the ICLR Code of Ethics and state that the research conducted in connection with this manuscript is adherent to the Code of Ethics.

Reproducibility statement

Our experimental results are designed to be reproducible. To this end, we have included a copy of the Jupyter notebooks and python scripts used to conduct our experiments. The subfolders entitled "LieGAN," "LieNLSD," and "LaLiGAN" contain the Section experimental results for the respective methods, which results correspond with Section 5.1 and Appendices A.1 and A.2. For these sections, scripts to reproduce the results of SymmetryML are contained in the "SymmetryML" folder, specifically under "methods/examples:" file names indicate the specific geometric object, while additional scripts and notebooks there reside which provide additional examples of SymmetryML, including unified discovery and enforcement pipelines. In "method/src," the reader will find the current (pre-release) source code for the python package which will be released following the conclusion of the conference. In the "method/Experiments_From_Papers/Current" folder are files relating to Sections 5.2 and 5.3. The "pope_discovery.ipynb" file contains the Section 5.2 experiment. The "Projective.ipynb" file contains the scripts needed to reproduce the SymmetryML + Baseline comparison for the enforcement of the projective symmetries in Section 5.3, and results appear in the subfolder "results/projective." The equivariance enforcement experiment script is contained in "enforcement_comparison.ipynb," and the data is contained in the "data" subfolder. The "results" subfolder contains the results for each method. The "scripts" subfolder contains scripts for the high dimensional hypersphere example in Appendix A.3, though results appear in "SymmetryML/results/reproduced/hypersphere_selected."

References

Ilyes Batatia, Mario Geiger, Jose M Munoz, Tess Smidt, Lior Silberman, and Christoph Ortner. A general framework for equivariant neural networks on reductive lie groups. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=3XStpETaO8.

Gregory Benton, Marc Finzi, Pavel Izmailov, and Andrew G Wilson. Learning invariances in neural networks from training data. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 17605–17616. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/cc8090c4d2791cdd9cd2cb3c24296190-Paper.pdf.

Doron L. Bergman. Symmetry constrained machine learning. In *Advances in Intelligent Systems and Computing*, pp. 501–512. Springer International Publishing, aug 2019. doi: 10.1007/978-3-030-29513-4_37. URL https://doi.org/10.1007.

Benjamin Bloem-Reddy and Yee Whye Teh. Probabilistic symmetries and invariant neural networks. *J. Mach. Learn. Res.*, 21(1), January 2020. ISSN 1532-4435.

N. Dehmamy, R. Walters, Y. Liu, D. Wang, and R. Yu. Automatic symmetry discovery with lie algebra convolutional network. *Advances in Neural Information Processing Systems*, 34:2503–2515, 2021. URL https://arxiv.org/abs/2109.07103.

Krish Desai, Benjamin Nachman, and Jesse Thaler. Symmetry discovery with deep learning. *Phys. Rev. D*, 105:096031, May 2022. doi: 10.1103/PhysRevD.105.096031. URL https://link.aps.org/doi/10.1103/PhysRevD.105.096031.

Roy T. Forestano, Konstantin T. Matchev, Katia Matcheva, Alexander Roman, Eyup B. Unlu, and Sarunas Verner. Discovering sparse representations of Lie groups with machine learning. *Phys. Lett. B*, 844:138086, 2023. doi: 10.1016/j.physletb.2023.138086.

Mario Geiger and Tess Smidt. e3nn: Euclidean neural networks, 2022. URL https://arxiv.org/abs/2207.09453.

A. Rod Gover and Thomas Leistner. Invariant prolongation of the killing tensor equation. *Annali di Matematica Pura ed Applicata (1923 -)*, 198(1):307–334, July 2018. ISSN 1618-1891. doi: 10.1007/s10231-018-0775-3. URL http://dx.doi.org/10.1007/s10231-018-0775-3.

Cody Grogan, Som Dutta, Mauricio Tano, Somayajulu L. N. Dhulipala, and Izabela Gutowska. Quantifying model uncertainty of neural network-based turbulence closures, 2024. URL https://arxiv.org/abs/2412.08818.

Nicolas Guigui, Nina Miolane, and Xavier Pennec. Introduction to riemannian geometry and geometric statistics: From basic theory to implementation with geomstats. *Foundations and Trends in Machine Learning*, 16:329–493, 2023. doi: https://doi.org/10.1561/2200000098.

Lexiang Hu, Yikang Li, and Zhouchen Lin. Explicit discovery of nonlinear symmetries from dynamic data. In *Forty-second International Conference on Machine Learning*, 2025.

Jihyeon Je, Jiayi Liu, Guandao Yang, Boyang Deng, Shengqu Cai, Gordon Wetzstein, Or Litany, and Leonidas Guibas. Robust symmetry detection via riemannian langevin dynamics. In *SIGGRAPH Asia 2024 Conference Papers*, SA '24, New York, NY, USA, 2024. Association for Computing Machinery. ISBN 9798400711312. doi: 10.1145/3680528.3687682. URL https://doi.org/10.1145/3680528.3687682.

Pavan Karjol, Vivek V Kashyap, Rohan Kashyap, and Prathosh AP. Interpretable discovery of one-parameter subgroups: A modular framework for elliptical, hyperbolic, and parabolic symmetries. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=c8WxrEWQcx.

Grzegorz Kaszuba, Tomasz Krakowski, Bartosz Ziegler, Andrzej Jaszkiewicz, and Piotr Sankowski. Implicit modeling of equivariant tensor basis with euclidean turbulence closure neural network. *Physics of Fluids*, 37(2):025137, 02 2025. ISSN 1070-6631. doi: 10.1063/5.0249490. URL https://doi.org/10.1063/5.0249490.

Gyeonghoon Ko, Hyunsu Kim, and Juho Lee. Learning infinitesimal generators of continuous symmetries from data. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 85973–86003. Curran Associates, Inc., 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/9c26a48b7ebd37bc3e979c8651fc3649-Paper-Conference.pdf.

Risi Kondor, Zhen Lin, and Shubhendu Trivedi. Clebsch-gordan nets: a fully fourier space spherical convolutional neural network, 2018. URL https://arxiv.org/abs/1806.09231.

Julia Ling, Andrew Kurzawski, and Jeremy Templeton. Reynolds averaged turbulence modelling using deep neural networks with embedded invariance. *Journal of Fluid Mechanics*, 807:155–166, 2016. doi: 10.1017/jfm.2016.615.

Ryley McConkey, Eugene Yee, and Fue-Sang Lien. A curated dataset for data-driven turbulence modelling. *Scientific Data*, 8(255), 2021. doi: 10.1038/s41597-021-01034-2. URL https://doi.org/10.1038/s41597-021-01034-2.

Aaron Miller, Sahil Kommalapati, Robert Moser, and Petros Koumoutsakos. Structure tensor reynolds-averaged navier-stokes turbulence models with equivariant neural networks, 2026. URL https://arxiv.org/abs/2511.09769.

Samuel E. Otto, Nicholas Zolman, J. Nathan Kutz, and Steven L. Brunton. A unified framework to enforce, discover, and promote symmetry in machine learning. *Journal of Machine Learning Research*, 26(248):1–83, 2025. URL http://jmlr.org/papers/v26/24-1315.html.

Ben Shaw, Abram Magner, and Kevin Moon. Symmetry discovery beyond affine transformations. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 112889–112918. Curran Associates, Inc., 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/cd04ec5aebfbe397c7fd718c35d02e0b-Paper-Conference.pdf.

Ben Shaw, Sasidhar Kunapuli, Abram Magner, and Kevin R. Moon. Continuous symmetry discovery and enforcement using infinitesimal generators of multi-parameter group actions, 2025. URL https://arxiv.org/abs/2505.08219.

Benjamin Shaw. Tractor connections for killing tensors and their generalizations. Master's thesis, Utah State University, 2021.

Nathaniel Thomas, Tess Smidt, Steven Kearnes, Lusann Yang, Li Li, Kai Kohlhoff, and Patrick Riley. Tensor field networks: Rotation- and translation-equivariant neural networks for 3d point clouds, 2018. URL https://arxiv.org/abs/1802.08219.

Maurice Weiler, Mario Geiger, Max Welling, Wouter Boomsma, and Taco Cohen. 3d steerable cnns: Learning rotationally equivariant features in volumetric data, 2018. URL https://arxiv.org/abs/1807.02547.

J. Yang, R. Walters, N. Dehmamy, and R. Yu. Generative adversarial symmetry discovery. *Proceedings of the 40th International Conference on Machine Learning*, 202(1647):39488–39508, 2023. URL https://arxiv.org/abs/2302.00236.

Jianke Yang, Nima Dehmamy, Robin Walters, and Rose Yu. Latent space symmetry discovery. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org, 2024.

Ke Ye and Lek-Heng Lim. Schubert varieties and distances between subspaces of different dimensions. *SIAM Journal on Matrix Analysis and Applications*, 37(3):1176–1197, 2016. doi: 10.1137/15M1054201. URL https://doi.org/10.1137/15M1054201.

A Additional Experimental Results

A.1 Sample Complexity Ablation Study for the Surfaces of Revolution

For each method in Section 5.1, we provide repeated experiments for sample sizes 400 and 4000 in addition to the featured sample size of $40,000$. Tables 3, 4, 5, and 6 contain results for our method, LieNLSD, LieGAN, and LaLiGAN, respectively. For LieGAN and LaLiGAN, the Recovered dimension is not actually recovered but rather is supplied as the ground truth, as explained in Section 5.1.

The results of our method are incomparable in terms of computation time and accuracy, with the exception of the Mexican hat. At 400 samples, neither our method nor LieNLSD appear to recover the correct number of dimensions consistently.

Table 3: Full sample-size sweep for SymmetryML on the surface-of-revolution symmetry discovery tasks (mean $\pm$ standard deviation).

Object	N	**Recovered dim.**	**Grassmann**	**Proj. Frobenius**	**Time (s)**
Sphere	400	3.00 ± 0.00	$(1.90 \pm 0.36) \times 10^{-8}$	$(1.99 \pm 1.72) \times 10^{-8}$	0.01 ± 0.00
	4,000	3.00 ± 0.00	0.000	0.000	1.56 ± 0.17
	40,000	3.00 ± 0.00	$(1.56 \pm 1.37) \times 10^{-8}$	$(2.40 \pm 2.17) \times 10^{-8}$	2.26 ± 0.21
Circle	400	1.00 ± 0.00	0.000	0.000	0.11 ± 0.10
	4,000	1.00 ± 0.00	0.000	0.000	1.62 ± 0.30
	40,000	1.00 ± 0.00	0.000	0.000	2.26 ± 0.11
Cylinder	400	2.00 ± 0.00	0.000	0.000	0.01 ± 0.00
	4,000	2.00 ± 0.00	$(4.97 \pm 8.60) \times 10^{-9}$	$(7.02 \pm 12.2) \times 10^{-9}$	0.61 ± 0.03
	40,000	2.00 ± 0.00	$(1.20 \pm 1.08) \times 10^{-8}$	$(1.70 \pm 1.53) \times 10^{-8}$	0.89 ± 0.05
Torus	400	1.00 ± 0.00	0.000	0.000	1.82 ± 0.14
	4,000	1.00 ± 0.00	0.000	0.000	2.71 ± 0.13
	40,000	1.00 ± 0.00	0.000	0.000	3.48 ± 0.23
Mexican Hat	400	0.33 ± 0.58	1.056 ± 0.891	0.679 ± 0.555	1.93 ± 0.10
	4,000	1.00 ± 0.00	0.016 ± 0.004	0.023 ± 0.006	2.81 ± 0.05
	40,000	1.00 ± 0.00	0.005 ± 0.000	0.008 ± 0.000	3.47 ± 0.14

Table 4: Full sample-size sweep for LieNLSD on the surface-of-revolution symmetry discovery tasks (mean $\pm$ standard deviation).

Object	N	**Recovered dim.**	**Grassmann**	**Proj. Frobenius**	**Time (s)**
Sphere	400	3.00 ± 0.00	0.0038 ± 0.0006	0.0053 ± 0.0009	0.73 ± 0.27
	4,000	3.00 ± 0.00	0.0031 ± 0.0009	0.0044 ± 0.0012	5.00 ± 0.14
	40,000	3.00 ± 0.00	0.0014 ± 0.0009	0.0020 ± 0.0013	56.77 ± 7.20
Circle	400	1.00 ± 0.00	0.0110 ± 0.0052	0.0156 ± 0.0074	1.90 ± 0.64
	4,000	1.00 ± 0.00	0.0015 ± 0.0007	0.0021 ± 0.0010	14.04 ± 0.38
	40,000	1.00 ± 0.00	0.0013 ± 0.0003	0.0019 ± 0.0004	141.99 ± 6.32
Cylinder	400	4.33 ± 0.58	0.746 ± 0.650	1.843 ± 0.104	2.06 ± 0.45
	4,000	4.67 ± 0.58	0.402 ± 0.693	1.799 ± 0.117	13.57 ± 0.40
	40,000	4.00 ± 1.00	1.166 ± 1.046	1.977 ± 0.236	136.64 ± 3.45
Torus	400	5.33 ± 0.58	0.508 ± 0.500	2.213 ± 0.176	0.85 ± 0.26
	4,000	1.00 ± 0.00	0.0021 ± 0.0010	0.0030 ± 0.0014	6.76 ± 0.29
	40,000	1.00 ± 0.00	0.0014 ± 0.0015	0.0019 ± 0.0021	71.25 ± 0.16
Mexican Hat	400	6.00 ± 0.00	0.0183 ± 0.0108	2.236 ± 0.0002	0.90 ± 0.23
	4,000	1.00 ± 0.00	0.0038 ± 0.0005	0.0053 ± 0.0007	7.19 ± 0.25
	40,000	1.00 ± 0.00	$(7.9 \pm 1.0) \times 10^{-4}$	0.0011 ± 0.0001	71.72 ± 2.15

Table 5: Full sample-size sweep for LieGAN on the surface-of-revolution symmetry discovery tasks (mean $\pm$ standard deviation).

Object	N	Recovered dim.	Grassmann	Proj. Frobenius	Time (s)
Sphere	400	3.00 ± 0.00	0.956 ± 0.190	1.189 ± 0.200	1.22 ± 0.21
	4,000	3.00 ± 0.00	0.0277 ± 0.0088	0.0392 ± 0.0125	10.44 ± 0.09
	40,000	3.00 ± 0.00	0.0143 ± 0.0041	0.0202 ± 0.0059	109.60 ± 4.89
Circle	400	1.00 ± 0.00	0.0594 ± 0.0260	0.0840 ± 0.0367	1.49 ± 0.13
	4,000	1.00 ± 0.00	0.0044 ± 0.0023	0.0062 ± 0.0032	11.22 ± 0.27
	40,000	1.00 ± 0.00	0.0050 ± 0.0023	0.0072 ± 0.0031	131.20 ± 15.39
Cylinder	400	2.00 ± 0.00	1.755 ± 0.219	1.781 ± 0.139	1.32 ± 0.20
	4,000	2.00 ± 0.00	1.562 ± 0.178	1.638 ± 0.120	11.85 ± 0.22
	40,000	2.00 ± 0.00	1.383 ± 0.322	1.391 ± 0.187	122.07 ± 2.78
Torus	400	1.00 ± 0.00	0.662 ± 0.376	0.808 ± 0.402	2.30 ± 0.28
	4,000	1.00 ± 0.00	0.0315 ± 0.0092	0.0445 ± 0.0130	20.20 ± 0.13
	40,000	1.00 ± 0.00	0.0154 ± 0.0007	0.0217 ± 0.0010	206.34 ± 3.88
Mexican Hat	400	1.00 ± 0.00	0.880 ± 0.347	1.024 ± 0.291	1.09 ± 0.17
	4,000	1.00 ± 0.00	0.0438 ± 0.0293	0.0619 ± 0.0413	9.52 ± 0.15
	40,000	1.00 ± 0.00	0.0176 ± 0.0067	0.0249 ± 0.0096	110.56 ± 6.34

Table 6: Full sample-size sweep for LaLiGAN on the surface-of-revolution symmetry discovery tasks (mean $\pm$ standard deviation).

Object	N	Recovered dim.	Grassmann	Proj. Frobenius	Time (s)
Sphere	400	3.00 ± 0.00	1.679 ± 0.124	1.871 ± 0.096	5.80 ± 0.08
	4,000	3.00 ± 0.00	1.399 ± 0.196	1.635 ± 0.182	38.22 ± 0.38
	40,000	3.00 ± 0.00	0.371 ± 0.641	0.473 ± 0.816	392.18 ± 12.07
Circle	400	1.00 ± 0.00	1.429 ± 0.073	1.397 ± 0.016	5.56 ± 0.20
	4,000	1.00 ± 0.00	1.356 ± 0.089	1.378 ± 0.026	38.52 ± 0.46
	40,000	1.00 ± 0.00	1.219 ± 0.344	1.276 ± 0.187	398.83 ± 14.66
Cylinder	400	2.00 ± 0.00	1.789 ± 0.138	1.859 ± 0.052	4.38 ± 0.23
	4,000	2.00 ± 0.00	1.788 ± 0.131	1.854 ± 0.048	21.76 ± 0.14
	40,000	2.00 ± 0.00	1.716 ± 0.106	1.782 ± 0.057	218.09 ± 14.83
Torus	400	1.00 ± 0.00	1.447 ± 0.083	1.400 ± 0.016	4.36 ± 0.37
	4,000	1.00 ± 0.00	1.440 ± 0.086	1.399 ± 0.018	20.66 ± 0.61
	40,000	1.00 ± 0.00	0.989 ± 0.818	0.956 ± 0.762	185.94 ± 1.15
Mexican Hat	400	1.00 ± 0.00	1.446 ± 0.084	1.400 ± 0.017	4.44 ± 0.30
	4,000	1.00 ± 0.00	1.436 ± 0.092	1.397 ± 0.019	20.01 ± 0.56
	40,000	1.00 ± 0.00	1.239 ± 0.380	1.275 ± 0.218	179.76 ± 3.02

A.2 Model Symmetry Detection

The experiment in which surfaces of revolution are examined for symmetry (Sections 5.1 and C.1), as mentioned previously, couple LieNLSD with level set estimation. However, LieNLSD was designed with model symmetry detection as the primary focus. We therefore compare the method of LieNLSD with our own in a few controlled synthetic experiments.

In the following tables, we generate data in standard Euclidean coordinates (x, y, z) from a standard normal distribution. For Tables 7 and 8, a model is trained to learn the function $\exp\left(x^2 + y^2\right)$, whereas in Tables 9 and 10, the ground truth is given by $\dfrac{2xy}{x^2 + y^2}$.

Table 7: Full sample-size sweep for SymmetryML on the model symmetry discovery task for $f(x, y, z) = \exp(x^2 + y^2)$ (mean $\pm$ standard deviation).

N	**Recovered dim.**	**Grassmann**	**Proj. Frobenius**	**Time (s)**
400	2.00 ± 0.00	0.0559 ± 0.0159	0.0395 ± 0.0112	0.90 ± 0.03
4,000	2.00 ± 0.00	0.0071 ± 0.0044	0.0050 ± 0.0031	7.43 ± 0.20
40,000	2.00 ± 0.00	0.0042 ± 0.0030	0.0029 ± 0.0021	77.20 ± 1.59

Table 8: Full sample-size sweep for LieNLSD on the model symmetry discovery task for $f(x, y, z) = \exp(x^2 + y^2)$ (mean $\pm$ standard deviation).

N	**Recovered dim.**	**Grassmann**	**Proj. Frobenius**	**Time (s)**
400	5.00 ± 0.00	0.0968 ± 0.0430	0.0432 ± 0.0192	2.96 ± 0.43
4,000	5.00 ± 0.00	0.0055 ± 0.0010	0.0025 ± 0.0004	24.63 ± 1.28
40,000	5.00 ± 0.00	0.0039 ± 0.0040	0.0018 ± 0.0018	232.36 ± 10.98

In the first case, the function is rotationally symmetric, being annihilated by vector fields ∂_z and $-y\partial_x + x\partial_y$: these are the only two Euclidean Killing vector fields under which the function is symmetric, so that a recovered dimension of 2 is sensible for our method. However, the vector fields $x\partial_z$, $y\partial_z$, and $z\partial_z$ also annihilate the ground truth function, so that a recovered dimension of 5 is sensible for LieNLSD. The results in Tables 7 and 8 suggest that our method has the computational edge, while LieNLSD has a slight edge in terms of accuracy, though this may be partially explainable by the larger search space.

Table 9: Full sample-size sweep for SymmetryML on the conformal model symmetry discovery task for $f(x, y, z) = 2xy/(x^2 + y^2)$ (mean $\pm$ standard deviation).

N	**Recovered dim.**	**Grassmann**	**Proj. Frobenius**	**Time (s)**
400	3.00 ± 0.00	1.0632 ± 0.5191	0.4658 ± 0.1492	1.12 ± 0.01
4,000	3.00 ± 0.00	0.0275 ± 0.0139	0.0159 ± 0.0080	10.25 ± 0.15
40,000	3.00 ± 0.00	0.0126 ± 0.0023	0.0073 ± 0.0014	104.03 ± 3.01

Table 10: Full sample-size sweep for LieNLSD on the conformal model symmetry discovery task for $f(x, y, z) = 2xy/(x^2 + y^2)$ (mean $\pm$ standard deviation).

N	**Recovered dim.**	**Grassmann**	**Proj. Frobenius**	**Time (s)**
400	14.00 ± 0.00	0.0612 ± 0.0148	0.8047 ± 0.0050	3.16 ± 0.39
4,000	14.00 ± 0.00	0.0054 ± 0.0021	0.8044 ± 0.0000	27.83 ± 0.48
40,000	14.00 ± 0.00	0.0028 ± 0.0007	$0.8044 \pm 1.0 \times 10^{-6}$	288.99 ± 8.63

In the second case, the function is invariant with respect to the three conformal Killing vectors of the Euclidean metric which contain quadratic components: these are the only three conformal Killing vector fields under which the function is invariant, so that a recovered dimension of 3 is sensible for our method. However, there are 14 vector fields with quadratic components which annihilate the function, so that a recovered dimension of 14 is sensible for LieNLSD. The results in Tables 9 and 10 seem to favor our method for the Projected Frobenius metric in the $N = 400$ and $N = 4,000$ cases, though LieNLSD appears to have the advantage otherwise. As before, our method is notably faster.

A.3 Recovering Symmetries of the Hypersphere in Higher Dimensions

Herein, we have claimed that our method is more scalable that existing methods. To this end, we apply our method to the recovery of the number of symmetries (Killing vectors) of the hypersphere

for higher dimensions. Current methods such as LieGAN do not appear to consider higher dimensional symmetries except in multi-particle cases, which is analogous to the diagonal action case we considered in Section 5.2. As specializing to diagonal or "per-particle" symmetries offers a very substantial reduction in the search space, such cases are insufficient to demonstrate scalability to problems that are truly high dimensional.

For a given dimension d, we generate data from a standard normal (d-dimensional) distribution. Then, we project to the $x_0 = 0$ plane. Subsequently, we normalize the data, placing the data on the level set defined by

$$x_1^2 + \cdots + x_d^2 - 1 = 0.$$

Our task is to recover the correct number of symmetries. Our results are shown in Table 11

Table 11: Results of our method on the Hypersphere

d	N	True Symmetries	Recovered Symmetries	Time (s)	Proportion
10	500	36	36 ± 0	0.064 ± 0.013	0.01
20	500	171	171 ± 0	0.678 ± 0.057	0.01
30	500	406	406 ± 0	3.536 ± 0.310	0.005
40	2000	741	741 ± 0	19.996 ± 0.227	0.005
50	2000	1176	1176 ± 0	50.421 ± 0.633	0.0045
60	2000	1711	1711 ± 0	103.418 ± 1.123	0.002

We find that our typical threshold of proportional explained variance of 0.01 must decrease as d increases, and that the sample size N must also increase, in order to recover the correct number of symmetries.

B Equivariant Function Identification using Bundle Connections

We consider the task of finding an equivariant basis of smooth functions $(u, v) \to (f_1(u, v), f_2(u, v))$ where the input vector fields are given as

$$X_1 = \ln(v)\partial_u + uv\partial_v, \qquad X_2 = \ln(v)\partial_u - uv\partial v, \qquad X_3 = u\partial_u - v\ln(v)\partial_v,$$

and corresponding output vector fields are

$$Y_1 = y\partial_x + x\partial_y, \qquad Y_2 = y\partial_x - x\partial_y, \qquad x\partial_x - y\partial_y.$$

It can be shown that

$$[X_1, X_3] = 2X_2, \qquad [Y_1, Y_3] = 2Y_2,$$

using the Lie bracket of vector fields, so that the equivariance condition need only be checked for the pairs $\{(X_1, Y_1), (X_3, Y_3)\}$. This condition is equivalent to the following system of equations, having solved Equation (3) algebraically for the first order derivatives and using $h(u, v) = \ln(v)^2 + u^2$ for brevity:

$$\begin{aligned} \frac{\partial f_1}{\partial u} &= \frac{\ln(v)f_2 + uf_1}{h(u,v)}, \qquad \frac{\partial f_1}{\partial v} = \frac{uf_2 - \ln(v)f_1}{vh(u,v)}, \\ \frac{\partial f_2}{\partial u} &= \frac{\ln(v)f_1 - uf_2}{h(u,v)}, \qquad \frac{\partial f_2}{\partial v} = \frac{\ln(v)f_2 + uf_1}{vh(u,v)}. \end{aligned} \tag{8}$$

The corresponding bundle connection $\tilde{\nabla}$ (with two-dimensional fibers) is given by

$$\begin{aligned} \tilde{\nabla}_{\partial u} E_1 &= -\frac{u}{h(u,v)}E_1 - \frac{\ln(v)}{h(u,v)}E_2, \quad \tilde{\nabla}_{\partial_u} E_2 = -\frac{\ln(v)}{h(u,v)}E_1 + \frac{u}{h(u,v)}E_2, \\ \tilde{\nabla}_{\partial_v} E_1 &= \frac{\ln(v)}{vh(u,v)}E_1 - \frac{u}{vh(u,v)}E_2, \quad \tilde{\nabla}_{\partial_v} E_2 = -\frac{u}{vh(u,v)}E_1 - \frac{\ln(v)}{vh(u,v)}E_2, \end{aligned} \tag{9}$$

where $\{E_1, E_2\}$ signifies the fiber basis vectors: solutions are expressed as $f_1(u, v)E_1 + f_2(u, v)E_2$. The number of solutions–that is, the number of elements in the equivariant basis–is given by the nullspace of the following matrix, which is derived using the bundle connection curvature and higher-order covariant derivatives, following Shaw (2021):

$$\frac{1}{v^2 h(u,v)^3} \begin{bmatrix} -4uv\ln(v)h(u,v) & 4u^2 vh(u,v) \\ -4v\ln(v)^2 h(u,v) & 4uv\ln(v)h(u,v) \\ 16u^2 v\ln(v) & -16u^3 v \\ 16uv\ln(v)^2 & -16u^2 v\ln(v) \\ 4u\ln(v)\left(h(u,v) + 4\ln(v)\right) & -4u^2\ln(v)\left(h(u,v) + 4\ln(v)\right) \\ 4\ln(v)\left(h(u,v) + 4\ln(v)\right) & -4u\ln(v)\left(h(u,v) + 4\ln(v)\right) \end{bmatrix}.$$

One such basis (over the module of smooth functions) is $\left\{\frac{u}{\ln(v)}E_1 + E_2\right\}$, meaning the dimension of the space of equivariant functions is precisely 1 (Shaw, 2021). To identify the basis explicitly, one can solve the reduced equations by constructing $S = f(u, v)\left(\frac{u}{\ln(v)}E_1 + E_2\right)$ and solving $\tilde{\nabla}S = 0$. This results in the following equations for the unknown function f, after removing degenerate equations:

$$\frac{\partial f}{\partial u} = 0, \qquad \frac{f_v v\ln(v) - f}{v\ln(v)} = 0. \tag{10}$$

Besides the deliverance of the precise number of equivariant basis functions, the tractor connection is useful for reducing the original system of equations, as seen by comparing Equations (8) and (10). We find that $f(u, v)$ is a scaling of $\ln(v)$, so that the basis of equivariant functions sought is $\{u\mathbb{E}_1 + \ln(v)E_2\}$.

This particular example is contrived, given the low dimensionality along with the fact that the identification of X_i with Y_i is made through the diffeomorphism $u(x, y) = x$, $v(x, y) = e^y$ (for $v > 0$). However, the example demonstrates that when the group action on the output space is represented such that the vector fields have linear components, the construction of affine bundle connections may help to identify equivariant functions exactly, without regard to reductivity (Batatia et al., 2023).

C Additional Theoretical Context and Results

C.1 Surfaces of Revolution

Herein, we describe the method by which uniform sampling for surfaces of revolution occurs, and we provide the precise formulas for each surface of revolution named in 5.1. The method of uniform sampling is merely that of uniform sampling on Riemannian manifolds equipped with a metric tensor. A surface of revolution can be parameterized as follows, where (x, y, z) are the ordinary Euclidean coordinates:

$$x = F(u)\sin(v), \quad y = F(u)\sin(v), \quad z = G(u)\cos(v).$$

That is, the surface is realized by rotating the parametric curve $(F(u), G(u))$ in the (x, z) plane about the z-axis, with the coordinate v being the ordinary angle defined on the (x, y) plane. This parameterization induces a metric tensor g in (u, v) coordinates which is independent of v, so that u and v are sampled independently. The components of g can be represented as the following matrix:

$$g = \begin{bmatrix} F^2 + G^2 & 0 \\ 0 & F^2 \end{bmatrix}.$$

Explicitly, v is sampled from Uniform$(0, 2\pi)$. The variable u is sampled according to the determinant of g, as described in Guigui et al. (2023).

C.2 The Infinitesimal Generators of the Vectorized Conjugate Action of SO(3) on Matrices of Size 3×3

In this Section, we complete the derivation outlined in Section 4.3.2. Taking Q_1, Q_2, and Q_3 to be the Euler rotations

$$Q_1 = \begin{bmatrix} \cos(t) & \sin(t) & 0 \\ -\sin(t) & \cos(t) & 0 \\ 0 & 0 & 1 \end{bmatrix}, Q_2 = \begin{bmatrix} \cos(t) & 0 & \sin(t) \\ 0 & 1 & 0 \\ -\sin(t) & 0 & \cos(t) \end{bmatrix}, Q_3 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(t) & \sin(t) \\ 0 & -\sin(t) & \cos(t) \end{bmatrix},$$

we obtain the following infinitesimal generators in coordinates defined by vec(M):

$$\begin{aligned} X_1 = &-(m_{21} + m_{12})\partial_{m_{11}} + (m_{11} - m_{22})\partial_{m_{12}} - m_{23}\partial_{m_{13}} + (m_{11} - m_{22})\partial_{m_{21}} \\ &+(m_{21} + m_{12})\partial_{m_{22}} + m_{13}\partial_{m_{23}} - m_{32}\partial_{m_{31}} + m_{31}\partial_{m_{32}}, \\ X_2 = &-(m_{31} + m_{13})\partial_{m_{11}} - m_{32}\partial_{m_{12}} + (m_{11} - m_{33})\partial_{m_{13}} - m_{23}\partial_{m_{21}} + m_{21}\partial_{m_{23}} \\ &+(m_{11} - m_{33})\partial_{m_{31}} + m_{12}\partial_{m_{32}} + (m_{31} + m_{13})\partial_{m_{33}}, \\ X_3 = &-m_{13}\partial_{m_{12}} + m_{12}\partial_{m_{13}} - m_{31}\partial_{m_{21}} - (m_{32} + m_{23})\partial_{m_{22}} + (m_{22} - m_{33})\partial_{m_{23}} \\ &+m_{21}\partial_{m_{31}} + (m_{22} - m_{33})\partial_{m_{32}} + (m_{32} + u_{23})\partial_{m_{33}}. \end{aligned}$$

It is through these generators that equivariance discovery and enforcement take place for the experiments related to turbulence closures (Sections 5.2 and 5.3). Parenthetically, we also note that they are 9-dimensional Killing vectors of the Euclidean metric tensor.

C.3 Diagonal Group Actions and their Infinitesimal Generators

Let G be a group acting on a set Y. The action of G on Y induces an action on the set Y^m, where a group element $g \in G$ acts on $(y_1, y_2, \ldots, y_m) \in Y^m$ by acting on each component y_i separately and in the manner in which g acts on $y \in Y$. It is said that the action of G on Y^m is a diagonal action.

If $\Psi(t, \mathbf{x})$ is a flow on $\mathbb{R}^n$, the induced diagonal action on $\mathbb{R}^n \times \mathbb{R}^n \times \cdots \times \mathbb{R}^n = (\mathbb{R}^n)^m$ is defined:

$$\tilde{\Psi}(t, (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m)) = (\Psi(t, \mathbf{x}_1), \Psi(t, \mathbf{x}_2), \ldots, \Psi(t, \mathbf{x}_m)).$$

If X is the vector field infinitesimal generator for $\Psi(t, \mathbf{x})$, we denote the infinitesimal generator for $\Psi(t, \mathbf{x}_i)$ by X_i, so that the infinitesimal generator $\tilde{X}$ for $\tilde{\Psi}$ is given as

$$\tilde{X} = X_1 + X_2 + \cdots + X_m.$$

For example, let $\Psi(t, (x, y)) = (x + t, ty)$, so that $X = \partial_x + y\partial_y$, the level curves for which can be written as $y = ce^x$. The infinitesimal generator for the diagonal action of Ψ on $\mathbb{R}^2 \times \mathbb{R}^2 \times \mathbb{R}^2$ is

$$\tilde{X} = \partial_{x_1} + y_1\partial_{y_1} + \partial_{x_2} + y_2\partial_{y_2} + \partial_{x_3} + y_3\partial_{y_3}.$$