Learning Equivariant Functions via Quadratic Forms

Pavan Karjol[1] **Vivek V Kashyap**[1] **Rohan Kashyap**[2] **Prathosh A P**[1]

[1]Department of Electrical Communication Engineering, Indian Institute of Science, Bengaluru, Karnataka
[2]Computer Science, Carnegie Mellon University, Pittsburgh, Pennsylvania, USA

Abstract

In this study, we introduce a method for learning group (known or unknown) equivariant functions by learning the associated quadratic form $x^T Ax$ corresponding to the group from the data. Certain groups, known as orthogonal groups, preserve a specific quadratic form, and we leverage this property to uncover the underlying symmetry group under the assumption that it is orthogonal. By utilizing the corresponding unique symmetric matrix and its inherent diagonal form, we incorporate suitable inductive biases into the neural network architecture, leading to models that are both simplified and efficient. Our approach results in an invariant model that preserves norms, while the equivariant model is represented as a product of a norm-invariant model and a scale-invariant model, where the "product" refers to the group action.

Moreover, we extend our framework to a more general setting where the function acts on tuples of input vectors via a diagonal (or product) group action. In this extension, the equivariant function is decomposed into an angular component extracted solely from the normalized first vector and a scale-invariant component that depends on the full Gram matrix of the tuple. This decomposition captures the inter-dependencies between multiple inputs while preserving the underlying group symmetry.

We assess the effectiveness of our framework across multiple tasks, including polynomial regression, top quark tagging, and moment of inertia matrix prediction. Comparative analysis with baseline methods demonstrates that our model consistently performs strongly in both discovering the underlying symmetry and efficiently learning the corresponding equivariant function.

Proceedings of the 29[th] International Conference on Artificial Intelligence and Statistics (AISTATS) 2026, Tangier, Morocco. PMLR: Volume 300.

1 INTRODUCTION

Symmetry provides powerful inductive biases in deep learning, improving generalization and sample efficiency (Cohen, 2013; Cohen and Welling, 2014; Finzi et al., 2021; Dehmamy et al., 2021). When the underlying group is known, it can be encoded into architectures via invariant or equivariant operations, as in CNNs, which achieve translational equivariance through weight sharing.

Two main strategies exist for incorporating symmetries: (i) data augmentation with parameterized distributions (Benton et al., 2020), and (ii) architectures with intrinsic symmetry (Zaheer et al., 2017; Kicki et al., 2020; Zhou et al., 2020). In contrast, we develop a quadratic-form framework for learning G-equivariant functions, showing they canonically decompose into norm-invariant and scale-invariant terms.

Despite progress across images (Anselmi et al., 2019; Kondor and Trivedi, 2018), graphs (Han et al., 2022), and PDEs (Horie and Mitsume, 2023), challenges remain: most methods require prior group knowledge, rely on heavy computations (e.g., null-spaces, exponential maps), and scale poorly. Moreover, in many applications the group action is not directly observable, complicating symmetry exploitation.

We address these limitations by proposing a framework applicable with or without prior group knowledge. Focusing on orthogonal groups, we exploit their preservation of quadratic forms $x^\top Ax$. For $O(n)$ this corresponds to Euclidean norm preservation; for Lorentz groups it corresponds to the Minkowski norm, fundamental in physics. This geometric perspective enables

efficient equivariance and symmetry discovery. Section 4 details the mathematical framework.

1.1 Contributions

- We show that any function that is equivariant to a generalized orthogonal group (i.e., one that preserves a quadratic form $x^T Ax$) can be expressed as the product of two invariant functions: a **scale-invariant** component that acts on the input normalized by the quadratic-form norm, and a **norm-invariant** component that depends only on that norm. The product is interpreted in the context of the group's action on the input space.
- We extend this decomposition to the setting with diagonal group action on the input product space. In this case, the equivariant map splits into (i) an **angular** term computed from only the normalized first input, and (ii) a **scale-invariant** term that depends on the full Gram matrix (all pairwise quadratic-form inner products) of the inputs, thereby capturing their inter-dependencies.
- Building on this decomposition, we introduce a **symmetry discovery** procedure: by learning the underlying quadratic form directly from data, the method uncovers the preserved symmetry without assuming it in advance.
- We validate the approach across diverse tasks: synthetic polynomial regression, top-quark tagging (Lorentz-invariant classification), and prediction of moment-of-inertia matrices (rotation-equivariant regression), demonstrating accurate symmetry recovery and strong predictive performance.

2 RELATED WORK

2.1 Group Equivariance

Significant progress has been made both theoretically and practically in symmetry learning for deep learning architectures (Kondor et al., 2018; Bekkers, 2021; Finzi et al., 2021; Bronstein et al., 2021; Dehmamy et al., 2021). The classical example is the G-equivariant neural networks (G-CNNs) (Cohen and Welling, 2014) which extends CNN's to accommodate a broader class of symmetry groups beyond translations. Building on this, (Kondor and Trivedi, 2018) developed convolution formulations for arbitrary compact group actions, while (Cohen et al., 2019) further extended G-CNNs to homogeneous spaces through equivariant linear mappings.

Equivariant multi-layer perceptrons (EMLPs) for general groups were introduced by (Finzi et al., 2021), leveraging null space computations to achieve equivariance; however, scalability challenges persist, especially for higher-order groups and also requires explicit knowledge of the group to encode the group representations. Similarly, (Otto et al., 2023) proposed a linear-algebraic approach that enforces symmetries through linear constraints and convex penalties, though this approach also encounters scalability limitations.

In related work, (Park et al., 2022) employed symmetric embedding networks to learn equivariant features for downstream tasks, although their method requires prior knowledge of the symmetry group. Contrastive learning methods were also developed by (Shakerinava et al., 2022; Dangovski et al., 2022) for self-supervised equivariant representations. Symmetry learning has found practical applications in domains such as protein structure prediction, where molecular properties require invariance for 3D rotations (Yim et al., 2023).

2.2 Automatic Symmetry Discovery

Beyond architectures with fixed symmetries, several works focus on *automatic discovery* of symmetries directly from data. These approaches differ in how they parameterize transformations, enforce invariance or equivariance, and whether they yield a usable prediction model.

2.2.1 Distribution over generators

Augerino (Benton et al., 2020) learns a distribution over Lie algebra generators and samples transformations during training, applying an invariance or equivariance loss depending on the task. Similarly, LieGAN (Yang et al., 2023) learns such distributions but enforces consistency via an adversarial loss: a discriminator distinguishes whether transformed samples remain within the data distribution. A key difference is that Augerino yields a trained prediction model with soft equivariance, while LieGAN focuses on generator discovery and does not provide a predictor.

2.2.2 Post-hoc generator extraction

LieGG (Moskalev et al., 2022) extracts Lie algebra generators from trained models using linear-algebraic analysis. While useful for diagnosing symmetries, the approach is inherently post-hoc and does not construct an explicit equivariant model.

2.2.3 Vector-field approaches

Learning Infinitesimal Generators (LIG) (Ko et al., 2024) models continuous flows via Neural ODEs, extending symmetry discovery beyond affine groups.

However, enforcing equivariance would require synchronized flows in the output space, computationally non-trivial and generally infeasible. Symmetry Discovery Beyond Affine (SDBA) (Shaw et al., 2024) also identifies vector fields that annihilate functions, but is typically limited to single-generator settings and mainly yields invariants rather than full equivariant predictors. Both methods therefore provide only indirect or implicit symmetry information.

2.2.4 Discrete-group inference

Bispectral Neural Networks (BNN) (Sanborn et al., 2022) address discrete groups by reconstructing Cayley tables from bispectrum features, thus inferring the group explicitly. While highly interpretable for finite groups, no analogous bispectral construction exists for continuous symmetries.

2.2.5 Our distinction

In contrast, our framework simultaneously *discovers* the underlying symmetry and provides an explicitly *interpretable prediction model*. By learning the quadratic form preserved by an orthogonal group, we obtain a canonical decomposition into norm- and scale-invariant components. Unlike distribution-based methods (Augerino, LieGAN), which enforce equivariance only implicitly, and unlike post-hoc or indirect approaches (LieGG, LIG, SDBA), our method integrates discovery and prediction in a unified and interpretable manner, combining the clarity of BNN with applicability to continuous groups.

3 PRELIMINARIES

Definition 3.1 (**Group**). *A group $(G, \cdot)$ is a set G with a binary operation $\cdot : G \times G \to G$ satisfying: (i) Closure: $a \cdot b \in G$ for all $a, b \in G$; (ii) Associativity: $(a \cdot b) \cdot c = a \cdot (b \cdot c)$; (iii) Identity: $\exists e \in G$ s.t. $e \cdot a = a \cdot e = a$ for all $a \in G$; (iv) Inverse: $\forall a \in G$, $\exists a^{-1} \in G$ with $a \cdot a^{-1} = a^{-1} \cdot a = e$.*

Definition 3.2 (**Group Action**). *A (left)* group action *of G on a set X is a map $G \times X \to X$, $(g, x) \mapsto g \cdot x$, such that (i) $e \cdot x = x$ for all $x \in X$; (ii) $(gh) \cdot x = g \cdot (h \cdot x)$ for all $g, h \in G$, $x \in X$.*

Definition 3.3 (**Orbit**). *Given a group action of G on X, the* orbit *of $x \in X$ is*

$$G \cdot x = \{\, g \cdot x \mid g \in G \,\}. \tag{1}$$

Definition 3.4 (**General Linear Group**). *The* general linear group $\mathrm{GL}(n, \mathbb{R})$ *is the group of all invertible $n \times n$ real matrices under matrix multiplication.*

Definition 3.5 (**Equivariant Function**). *Let G act on sets X and Y. A function $f : X \to Y$ is* equivariant *if*

$$f(g \cdot x) = g \cdot f(x), \quad \forall g \in G,\ x \in X.$$

Equivariance means f commutes with the group action. An invariant function *is the special case where $g \cdot f(x) = f(x)$ for all $g \in G$.*

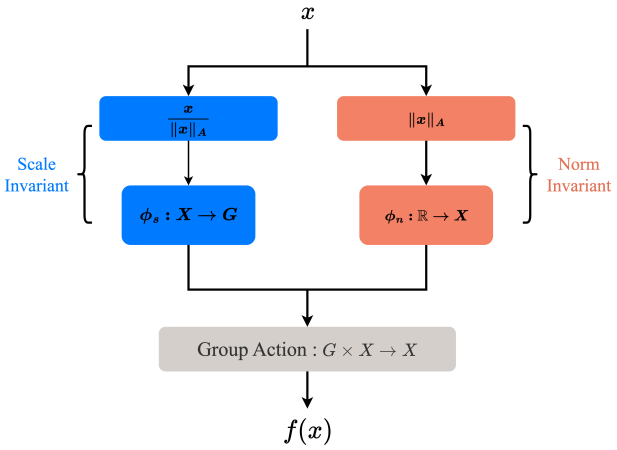


Figure 1: Schematic representation of the proposed method. ϕ_n and ϕ_s denote neural networks, where the norm-invariant network takes input norms, and the scale-invariant network takes normalized inputs where $\|x\|_A = \mathrm{sign}(x^T A x)\sqrt{|x^T A x|}$.

4 PROPOSED METHOD

We present a general framework for learning group-equivariant functions by embedding appropriate quadratic forms within our network architecture. Specifically, we focus on groups that preserve quadratic forms of the type "$x^T A x$", known as orthogonal groups. Our framework supports two cases: when the underlying group is known and when it is unknown. In the first case, we fix the quadratic form within the architecture. In the second case, we allow the quadratic form to be learned. We begin with a brief overview of quadratic forms before presenting our main contributions.

4.1 Quadratic Forms

A quadratic form in $x \in \mathbb{R}^n$, with $A \in \mathbb{R}^{n \times n}$, is

$$q(x; A) := x^T A x. \tag{2}$$

Every quadratic form is uniquely determined by a symmetric matrix, since

$$q(x; A) = x^T A x = \tfrac{1}{2}\, x^T (A + A^T) x, \quad \forall x \in \mathbb{R}^n,$$

so without loss of generality we assume A symmetric.

A symmetric matrix A can be diagonalized by an orthogonal matrix U, i.e.

$$A = U^T D U, \tag{3}$$

where D is diagonal with the eigenvalues of A. Substituting into (2) gives

$$q(x; A) = x^T U^T D U x = (Ux)^T D (Ux) = y^T D y, \tag{4}$$

with $y = Ux$. Hence, learning $q(x; A)$ reduces to learning an orthogonal matrix U and a diagonal matrix D.

4.2 Orthogonal Group: $O(p, q)$

Consider the set defined as,

$$O(p, q) := \{ U \in GL(n, \mathbb{R}) : U^T A U = A \}, \tag{5}$$

which is a Lie subgroup of $GL(n, \mathbb{R})$ that preserves the quadratic form $q(x; A) = x^T A x$, i.e. $q(Rx; A) = q(x; A)$ for all $R \in O(p, q)$. Examples include the standard orthogonal group preserving the Euclidean norm $x^T x$, and the Lorentz group preserving the Minkowski norm $x^T \eta x$ with $\eta = \mathrm{diag}(1, -1, -1, -1)$.

In this work, we focus on learning a function $f : X \to \mathbb{R}^m$, where f is equivariant under an orthogonal group G that preserves a given or unknown quadratic form $q(x; A)$. In other words, A may be known or unknown.

We now state our first result in the following theorem, which characterizes the set of orbits under the action of an orthogonal group G. Specifically, each G-orbit corresponds to a unique value of the quadratic form.

Theorem 4.1. *Let A be a non-zero $n \times n$ matrix, and define $U := \{x \in \mathbb{R}^n : x^T A x = 0\}$. Let G be an orthogonal group that preserves the quadratic form $x^T A x$, and consider the action of G on $\mathbb{R}^n \setminus U$. For a given $c \in \mathbb{R} \setminus \{0\}$, define the set $\mathcal{O}_c := \{x \in \mathbb{R}^n : x^T A x = c\}$. Then, the set of orbits under the action of G is given by*

$$\mathcal{O}(G) = \{\mathcal{O}_c : c \in \mathbb{R}^n \setminus \{0\} \ \ \& \ \ \mathcal{O}_c \neq \varnothing\}. \tag{6}$$

Using the above result, we show that any G-equivariant function can be expressed in a canonical form that combines a norm-invariant (equivalently, G-invariant) function and a scale-invariant function. This formulation is formalized in the following theorem.

Theorem 4.2. *Let A be a non-zero $n \times n$ matrix, and define $U := \{x \in \mathbb{R}^n : x^T A x = 0\}$. Any G-equivariant function $f : \mathbb{R}^n \setminus U \to \mathbb{R}^m$, where G is an orthogonal group that preserves the quadratic form $x^T A x$, can be expressed as,*

$$f(x) = \phi_s\left(\frac{x}{\|x\|_A}\right) \cdot \phi_n\big(\|x\|_A\big), \tag{7}$$

for some functions $\phi_s : \mathbb{R}^n \to G$ and $\phi_n : \mathbb{R} \to \mathbb{R}^m$. Here, $\|z\|_A$ denotes the norm associated with the quadratic form A, defined by $\|z\|_A = \mathrm{sign}(z^T A z)\sqrt{|z^T A z|}$. Moreover, if the action of G on the output space is faithful, then ϕ_s is itself G-equivariant.

Remark 4.1 (Equivariance of ϕ_s)**.** *Under a faithful output action, the decomposition in Eq. (7) implies that ϕ_s is itself G-equivariant. As shown in Appendix C.5), this follows from the equivariance of f together with the invariance of ϕ_n. Thus, in this setting, ϕ_s inherits equivariance directly from the decomposition.*

Remark 4.2. *We would like to emphasize that $\|z\|_A$ is a pseudonorm, as it can take on negative values.*

The scale invariance part of the function decomposition of (7) is illustrated in Figure 2.

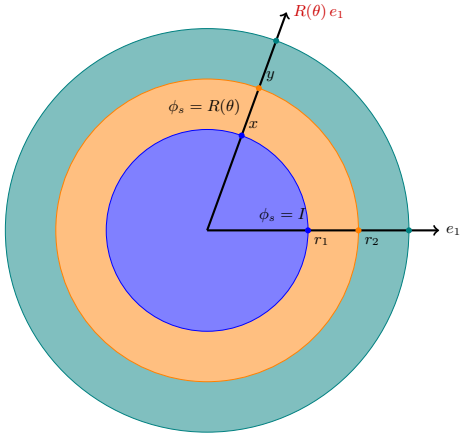


Figure 2: For an $O(2)$-equivariant function f, we have $f(x) = f\big(R(\theta) r_1 e_1\big) = \underbrace{R(\theta)}\, f(r_1 e_1)$. Similarly, $f(y) = \underbrace{R(\theta)}\, f(r_2 e_1)$; $f(z) = \underbrace{R(\theta)}\, f(r_3 e_1)$. Hence, $\phi_s(x) = \phi_s(y) = \phi_s(z) = R(\theta)$. This property holds for general orthogonal groups as well.

Remark 4.3. *In Theorem 4.2, the set U has measure zero in $\mathbb{R}^n$. Therefore, including this set U in our experiments has a negligible impact on the results, as it represents an insignificant portion of the space.*

Next, we analyze the continuity and differentiability of the G-equivariant function as follows:

Proposition 4.1. *Consider Theorem 4.2. Suppose A is either a positive or a negative definite matrix. Then, ϕ_n and ϕ_s are smooth (C^∞) whenever f is smooth (C^∞). If A is neither a positive nor a negative definite matrix, then ϕ_n and ϕ_s are continuous (C^0) whenever f is continuous (C^0).*

4.3 Diagonal group action

We now generalize our approach to the setting where the function acts on tuples of vectors rather than on individual vectors. Specifically, let X be a G-invariant subset of $\mathbb{R}^n \setminus U$, where

$$U := \{x \in \mathbb{R}^n : x^T A x = 0\},$$

and define the product space

$$X_p := \underbrace{X \times X \times \cdots \times X}_{p \text{ times}}.$$

On X_p, the group G acts diagonally, that is,

$$g \cdot (x_1, x_2, \ldots, x_p) = (g \cdot x_1,\ g \cdot x_2,\ \ldots,\ g \cdot x_p).$$

We recall the standard result (see Appendix, Classification of p-tuples up to the A-orthogonal group) which states that any two p-tuples of vectors in $\mathbb{R}^n$ that share the same Gram matrix (with respect to a non-degenerate symmetric bilinear form) are related by an isometry (Jacobson, 1985). Building upon this well-established fact, we derive a new decomposition theorem for functions equivariant under the diagonal G-action on tuples. This new result, formalized in the following theorem, extends our previous method to a broader and more general setting.

Theorem 4.3 (Extended Decomposition for G-Equivariant Functions on Tuples)**.** *Let A be a non-singular $n \times n$ real matrix and define Let G be an orthogonal group as defined in Eq. (5). Let $X \subset \mathbb{R}^n$ be a G-invariant set. For any positive integer p, consider the diagonal action of G on the product space $X_p := \underbrace{X \times X \times \cdots \times X}_{p \text{ times}}$, defined by, $\forall g \in G$*

$$g \cdot (x_1, \ldots, x_p) = (g \cdot x_1,\ \ldots,\ g \cdot x_p).$$

Then every G-equivariant function $f : X_p \to \mathbb{R}^m$ can be decomposed as

$$f(x) = \phi_s\left(\frac{x_1}{\|x_1\|_A}\right) \cdot \phi_n\big(\mathrm{Gram}(x)\big), \tag{8}$$

where $x = \Big(x_1, \ldots, x_p\Big)$ and Gram matrix

$$\mathrm{Gram}(x_1, \ldots, x_p) = \big[x_i^T A x_j\big]_{i,j \in [p]}$$

encodes all pairwise inner products with respect to the quadratic form defined by A.

Please refer to the appendix for the proofs of all the Theorems and propositions.

Remark 4.4. *When A is non-singular, Theorem 4.2 is a special case of Theorem 4.3. Notably, Theorem 4.2 remains applicable even when A is singular. In contrast, Theorem 4.3 would require further modifications to accommodate the singular case, and we leave these extensions for future work.*

4.4 Learning Equivariant Functions

Given data $\{(x_i, y_i = f(x_i))\}_{i \in [m]}$, where f is a G-equivariant function for some known or unknown orthogonal group G that preserves the quadratic form $x^T A_0 x$, we implement the functions ϕ_s and ϕ_n through neural networks. We denote these networks by the same notation, with ϕ_s and ϕ_n representing the scale-invariant and norm-invariant components of the function, respectively.

4.4.1 G is Known

When the group G is known, we fix the corresponding quadratic form in our network architecture, specifically fixing the matrix A. In this case, we aim to minimize the following loss function:

$$\arg\min_{\theta} \sum_i \mathcal{L}\left(y_i, \phi_s\left(\frac{x}{\|x\|_A}\right) \cdot \phi_n\big(\|x\|_A\big)\right) \tag{9}$$

where $\theta = \{\theta_s, \theta_n\}$ are the weights of the networks corresponding to ϕ_s and ϕ_n, respectively.

4.4.2 G is Unknown: Symmetry Discovery Framework

In this case, the goal is to discover the underlying symmetry group G by learning the appropriate matrix A. We seek to minimize the following loss function:

$$\arg\min_{A,\theta} \sum_i \mathcal{L}\left(y_i, \phi_s\left(\frac{x}{\|x\|_A}\right) \cdot \phi_n\big(\|x\|_A\big)\right), \tag{10}$$

Thus, discovering the underlying group G corresponds to learning the appropriate symmetric matrix A. However, as discussed earlier, this problem reduces to learning the appropriate orthonormal matrix U and diagonal matrix D due to the diagonalization of the symmetric matrix A. Therefore, we can rewrite the objective function in Eq. (10) as:

$$\arg\min_{U,D,\theta} \sum_i \mathcal{L}\left(y_i, \phi_s\left(\frac{\hat{x}}{\|\hat{x}\|_D}\right) \cdot \phi_n\big(\|\hat{x}\|_D\big)\right), \tag{11}$$

where $\hat{x} = Ux$, $U \in O(n)$ and D is a diagonal matrix. In this formulation, U captures the orthogonal transformation and D scales the components of the input x_i, thereby enabling the discovery of the underlying symmetry group. The block diagram of our proposed method is depicted in Figure 1.

For invariance tasks, we use only the ϕ_n network and omit ϕ_s. When the group acts on the input domain via a product action, we adopt the function decomposition from (8). This decomposition is applied both when the group is explicitly known, where we incorporate it into (9), and when the group is unknown, in which case it is used in (10).

Table 1: Examples of canonical forms for different output representations under $O(p,q)$. The required form depends explicitly on the tensor type and the induced group action. Here $P_x^A = \frac{xx^\top A}{x^\top Ax}$.

Output Type	Equivariance Constraint	Canonical Form
Vector (fundamental)	$f(Rx) = Rf(x)$	$f(x) = \phi(s)\, x$
Covariant tensor	$K(Rx) = R^{-T}K(x)R^{-1}$	$K(x) = \alpha(s)\, A + \beta(s)\, (Ax)(Ax)^\top$
Contravariant tensor	$T(Rx) = R\,T(x)\,R^\top$	$T(x) = \alpha(s)\, A^{-1} + \beta(s)\, xx^\top$
Mixed tensor / conjugation	$F(Rx) = R\,F(x)\,R^{-1}$	$F(x) = \alpha(s)\, I + \beta(s)\, P_x^A$

5 DISCUSSION

5.1 Existing Canonical Forms

We emphasize that even for classical orthogonal groups such as $O(n)$, the canonical form of an equivariant function depends crucially on the *output representation* and the induced group action. This extends directly to $O(p,q)$ preserving a quadratic form $x^\top Ax$. In particular, there is no universal output-independent ansatz in the classical setting: vector, covariant, contravariant, and conjugation-equivariant outputs each require different support matrices and hence different canonical forms. We summarize representative examples for $O(p,q)$ in Table 1, where the structure changes substantially with the tensor type. Thus, existing canonical forms are inherently output-specific by construction.

This distinction is important for the tasks considered in our paper. For example, the moment-of-inertia prediction task and the synthetic matrix-valued regression problem $f(x) = 9(xx^\top A)^2 + 2(xx^\top A)$ produce matrix outputs with conjugation-type equivariance, $F(Rx) = RF(x)R^{-1}$, which require bases such as I and P_x^A as in Table 1, rather than the standard vector-output form used in EGNN-style models. More generally, our decomposition is not tied to a single output tensor type: it also accommodates the group-valued equivariant component ϕ_s, which maps a normalized input on an orbit to a group element aligning a fixed representative with that input. In this sense, while classical canonical forms remain useful for specific output actions, our framework provides a unified decomposition principle that can handle vector-valued, matrix-valued, and group-valued components within the same symmetry-discovery pipeline.

5.2 Uniqueness and Householder Reflection

The proposed canonical decomposition of a G-equivariant function is unique up to a group transformation. Furthermore, under such a transformation, the output of the scale-invariant component of the function is equivalent to a Householder reflection transformation, R. This can be expressed as:

$$R = \phi_s\left(\frac{x}{\|x\|_A}\right) = \tau\left(I - \frac{2ww^TA}{w^TAw}\right)\tau^{-1} \tag{12}$$

where τ is an appropriate transformation that maps the group G to the standard indefinite orthogonal group $O(p,q)$.

A key property of a Householder reflection is that it is its own inverse (i.e., $R = R^{-1}$). This insight leads to a significant simplification of our proposed form, particularly when the group action on the function's output is a conjugation:

$$f(g \cdot x) = g \cdot f(x) = gf(x)g^{-1} \tag{13}$$

This simplification is formalized in the following proposition.

Proposition 5.1. *Let f be a G-equivariant function where the group action on the output is a conjugation, as defined in Eq.* (13)*. Then, the function $f(x)$ can be expressed in the symmetric form:*

$$f(x) = \phi_s\left(\frac{x}{\|x\|_A}\right)\phi_n\left(\|x\|_A\right)\phi_s\left(\frac{x}{\|x\|_A}\right) \tag{14}$$

6 EXPERIMENTS

We evaluate **G-Ortho-Nets**, our proposed method for learning G-equivariant functions through decomposition into norm-invariant and scale-invariant components as given in Eq. (7). The quadratic form A is learned jointly with the decomposed functions, enabling simultaneous discovery of the underlying symmetry group $O(p,q)$ which is defined in Eq. (5).

6.1 Baseline Methods

We compare G-Ortho-Nets against two recent approaches for symmetry discovery:

Table 2: Suitability of related symmetry discovery methods for our setting. Only Augerino and LieGG provide both a usable prediction model and applicability to continuous equivariance tasks, motivating our choice of baselines. Our method further distinguishes itself by offering an **interpretable** model that trains in a **single stage**. ✓= supported, ×= not supported/limited.

Method	Prediction Model	Continuous Equivariance	Interpretable Prediction Model	Limitations/Notes
G-Ortho-Nets (Ours)	✓	✓	✓	*Interpretable prediction model, single-stage*
Augerino†	✓	✓	×	Sample averaging required at inference
LieGG†	✓	✓	×	Post-training analysis needed
LieGAN	×	✓	×	No prediction model
LIG	✓	×	×	Requires synchronized output flows, two stage
SDBA	✓	×	×	Limited to single generator, two stage
BNN	✓	×	✓	Appropriate for discrete groups

†Selected as baselines (only methods suitable for continuous equivariance with prediction models)

LieGG (Moskalev et al. (2022)) retrieves infinitesimal generators of symmetries by solving a matrix nullspace equation derived from Lie group theory. Using singular value decomposition of the polarization matrix, it extracts the Lie algebra basis and quantifies *symmetry bias* and *symmetry variance*, measuring how closely learned generators approximate true underlying symmetries.

Augerino (Benton et al. (2020)) learns invariances and equivariances by parameterizing a distribution over data augmentations. The model jointly optimizes network and augmentation parameters to discover symmetries directly from training data. Unlike LieGG's post-hoc extraction, Augerino induces soft equivariance during training.

The rationale for selecting these two methods as baselines is summarized in Table 2.

6.2 Experimental Tasks

We evaluate on three tasks spanning synthetic regression, physical simulation, and high-energy physics applications. Table 3 summarizes the tasks and their corresponding symmetry groups.

Table 3: Experimental tasks and symmetry groups.

Task	Group
Synthetic regression	$O(2,2)$
Moment of inertia	$O(3)$
Top quark tagging	$O(1,3)$

6.2.1 Task 1: Synthetic Regression

We construct a target function exhibiting equivariance under the indefinite orthogonal group $O(2,2)$:

$$f(x) = 9\left(xx^\top Axx^\top A\right) + 2\left(xx^\top A\right), \qquad (15)$$

where $A \in \mathbb{R}^{4\times4}$ is symmetric and indefinite with signature $(2,2)$—two positive and two negative eigenvalues. The function satisfies $f(g\cdot x) = g\, f(x)\, g^\top$ for all $g \in G$. This controlled setting allows precise evaluation of symmetry recovery.

6.2.2 Task 2: Moment of Inertia Prediction

Given n point masses m_i at positions $x_i \in \mathbb{R}^3$, we predict the inertia tensor:

$$f(x,m) = \sum_{i=1}^{n} m_i \left(\|x_i\|^2 I - x_i x_i^\top\right). \qquad (16)$$

This physically meaningful function is equivariant under the group $O(3)$:

$$f(g\cdot x, m) = g\, f(x,m)\, g^\top, \quad \forall g \in O(3). \qquad (17)$$

This task tests the ability to recover well-known physical symmetries from data.

6.2.3 Task 3: Top Quark Tagging

We classify jets as originating from top quarks versus light quarks using jet constituent four-vectors. The classification label is invariant under Lorentz transformations corresponding to the group

$$G = O(1,3), \quad A = \eta = \mathrm{diag}(1,-1,-1,-1). \qquad (18)$$

This high-dimensional real-world task evaluates the ability to discover Lorentz invariance in a realistic particle physics setting.

6.3 Evaluation Metrics

We assess performance using three complementary metrics:

Table 4: Regression tasks. Best results in **bold**. $\cos(A_0, A_{\text{learnt}})$ is cosine similarity between true and learned A; $d_{\text{PA}}(\mathfrak{g}_0, \mathfrak{g}_{\text{learnt}})$ is projection distance from principal angles.

Task	Method	Pred. Error	$\cos(A_0, A_{\text{learnt}})$	$d_{\text{PA}}(\mathfrak{g}_0, \mathfrak{g}_{\text{learnt}})$
Synthetic	G-Ortho-Nets	1.80×10^{-3}	**0.99**	**0.24**
	LieGG	$\mathbf{7.53 \times 10^{-4}}$	0.93	1.80
	Augerino	2.63×10^{0}	0.30	1.99
Inertia	G-Ortho-Nets	$\mathbf{1.88 \times 10^{-3}}$	**0.99**	0.15
	LieGG	2.61×10^{-3}	**0.99**	**0.06**
	Augerino	7.86×10^{-3}	0.57	1.26

Table 5: Top-tagging classification. Best results in **bold**.

Task	Method	Accuracy	$\cos(A_0, A_{\text{learnt}})$	$d_{\text{PA}}(\mathfrak{g}_0, \mathfrak{g}_{\text{learnt}})$
Top tagging	G-Ortho-Nets	**90.44**	**0.9999**	**0.0214**
	LieGG	83.10	0.9991	1.0012
	Augerino	51.80	0.5000	2.1830

Prediction Accuracy. We measure the mean squared error (MSE) for regression tasks (Tasks 1–2) and the classification accuracy for the tagging task (Task 3). This quantifies how well models approximate the target function.

Quadratic Form Recovery. We compute the cosine similarity between the learned and true quadratic forms:

$$\cos(A_0, A_{\text{learnt}}) = \frac{\langle A_{\text{learnt}}, A_0 \rangle_F}{\|A_{\text{learnt}}\|_F \, \|A_0\|_F}, \tag{19}$$

where $\langle \cdot, \cdot \rangle_F$ denotes the Frobenius inner product. For G-Ortho-Nets, A is learned directly; for LieGG and Augerino, we recover A from the learned generators X via the constraint $X^\top A + AX = 0$.

Lie Algebra Comparison. We compare the learned Lie algebra $\mathfrak{g}_{\text{learnt}}$ with the true Lie algebra $\mathfrak{g}_0$ using the principal angles. Given orthonormal bases for both subspaces, the principal angles $\theta_1 \leq \theta_2 \leq \cdots \leq \theta_k \in [0, \pi/2]$ characterize the geometric relationship between them. We compute the projection distance as:

$$d_{\text{PA}}(\mathfrak{g}_0, \mathfrak{g}_{\text{learnt}}) = \sqrt{\sum_{i=1}^{k} \sin^2(\theta_i)}, \tag{20}$$

which measures the overall misalignment between the two Lie algebra subspaces. A value of 0 indicates perfect alignment, while larger values indicate greater deviation.

6.4 Results

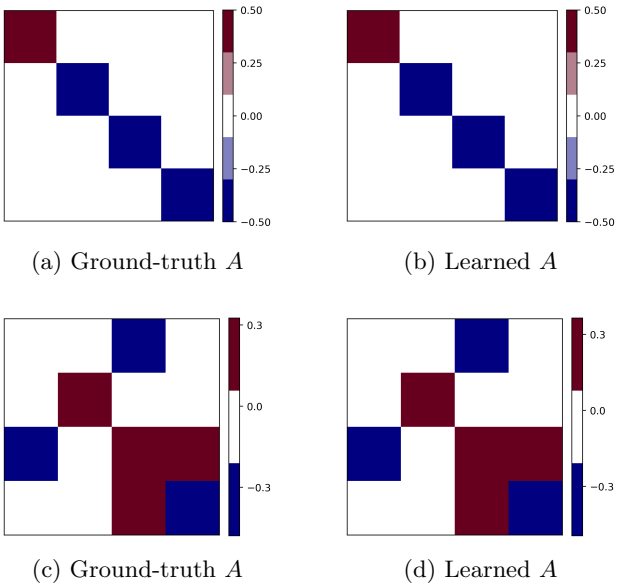
6.4.1 Quantitative results

Tables 4 and 5 summarize performance across all tasks. We evaluate prediction quality (MSE or accuracy), recovery of the quadratic form via $\cos(A_0, A_{\text{learnt}})$, and Lie-algebra alignment using projection distance $d_{\text{PA}}(\mathfrak{g}_0, \mathfrak{g}_{\text{learnt}})$ (principal angles). While LieGG achieves competitive performance in certain cases, it does not yield an interpretable prediction model and requires additional post-hoc analysis to extract Lie algebra generators.

6.4.2 Qualitative results

Figure 3 compares ground-truth and learned quadratic forms for the top-tagging and synthetic tasks, demonstrating faithful recovery of the target structure. G-Ortho-Nets achieve strong performance across both metrics and tasks. By explicitly parameterizing the quadratic form A, our method enables direct symmetry discovery, rather than inferring it indirectly from generators. On the top-tagging benchmark in particular, it attains state-of-the-art accuracy while most closely recovering the Lorentz structure, highlighting that encoding physically meaningful symmetries can improve generalization in high-energy physics.

Additional implementation details, experimental settings, and further analyses are provided in the appendix. In particular, implementation details of the decomposition are given in Section B.1, experimental details are provided in Section B.2, and the multi-run evaluation and noise-robustness results are reported in Section B.3 and Table 8.



(a) Ground-truth A (b) Learned A

(c) Ground-truth A (d) Learned A

Figure 3: Comparison of ground-truth and learned A matrices for top-tagging (top) and synthetic regression (bottom) using G-Ortho-Nets.

6.5 Top-tagging : Comparison with LieGAN

Some methods considered in prior work (see Table 2) are not directly comparable to our setting. In particular, LieGAN (Yang et al. (2023)) does not itself perform classification; it only learns Lie generators, and the reported "LieGAN accuracy" in prior work corresponds to a separate downstream GNN trained using the recovered metric. Likewise, LorentzNet (Gong et al. (2022)) and Clifford-based models (Ruhe et al. (2023)) assume the relevant symmetry structure is already known and hard-coded into the architecture, whereas our method must infer the symmetry directly from data. Thus, higher accuracy under such stronger prior assumptions is not inconsistent with our objectives.

To enable a fair comparison with LieGAN, we consider a matched protocol in which the learned quadratic form A from each method is used to train the same downstream GNN. The 500k-sample comparison is summarized in Table 6. Our method yields both more accurate symmetry recovery and better downstream classification performance under this matched setting. Additional details are provided in Appendix B.4, including the 100k multi-run matched comparison in Table 9, the symmetry-discovery timing analysis in Table 10, and the full wall-clock pipeline comparison at 500k samples in Table 11.

Table 6: Top-tagging results at 500k samples under a matched comparison, where the quadratic form A learned by each method is used to train the same downstream GNN.

Method	$\cos(A_0, A_{\text{learnt}})(\uparrow)$	**Accuracy** $(\uparrow)$
G-Ortho-Nets	**0.9999**	**93.75%**
LieGAN	0.9975	93.33%

6.6 Limitations

Our framework is currently limited to orthogonal groups, as these are exactly the transformations that preserve quadratic forms. While this restriction excludes more general classes such as affine or diffeomorphic symmetries, it provides a clear advantage: interpretability. By explicitly learning the quadratic form A, our model yields a physically meaningful and directly interpretable predictor, rather than only an implicit or post-hoc symmetry characterization. This trade-off mirrors the broader tension in symmetry learning between generality and interpretability. We view our approach as complementary to methods such as LieGAN, Augerino or LieGG, which target broader classes but do not produce usable equivariant predictors.

7 CONCLUSION

We presented a quadratic-form framework for learning group-equivariant functions, focusing on orthogonal groups. We proved a canonical decomposition into *norm-invariant* and *scale-invariant* components, and extended it to tuples under diagonal actions, where the equivariant map splits into an angular term (from the normalized first vector) and a scale-invariant term (from the full Gram matrix), thereby capturing interdependencies while preserving symmetry. Experiments across synthetic and physics-inspired tasks showed reliable symmetry discovery and effective equivariant learning. Future work includes extending beyond orthogonal groups and accommodating richer group actions.

References

Fabio Anselmi, Georgios Evangelopoulos, Lorenzo Rosasco, and Tomaso Poggio. Symmetry-adapted representation learning. *Pattern Recognition*, 86: 201–208, 2019.

Erik J Bekkers. B-spline cnns on lie groups, 2021. URL `https://arxiv.org/abs/1909.12057`.

Gregory Benton, Marc Finzi, Pavel Izmailov, and Andrew Gordon Wilson. Learning invariances in neural

networks. *arXiv preprint arXiv:2010.11882*, 2020. NeurIPS 2020 version available.

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

Taco Cohen. *Learning transformation groups and their invariants*. PhD thesis, PhD thesis, University of Amsterdam, 2013.

Taco Cohen and Max Welling. Learning the irreducible representations of commutative lie groups, 2014.

Taco S Cohen, Mario Geiger, and Maurice Weiler. A general theory of equivariant cnns on homogeneous spaces. *Advances in neural information processing systems*, 32, 2019.

Rumen Dangovski, Li Jing, Charlotte Loh, Seungwook Han, Akash Srivastava, Brian Cheung, Pulkit Agrawal, and Marin Soljačić. Equivariant contrastive learning, 2022. URL `https://arxiv.org/abs/2111.00899`.

Nima Dehmamy, Robin Walters, Yanchen Liu, Dashun Wang, and Rose Yu. Automatic symmetry discovery with lie algebra convolutional network. *Advances in Neural Information Processing Systems*, 34:2503–2515, 2021.

Marc Finzi, Max Welling, and Andrew Gordon Wilson. A practical method for constructing equivariant multilayer perceptrons for arbitrary matrix groups. In *International conference on machine learning*, pages 3318–3328. PMLR, 2021.

Shiqi Gong, Qi Meng, Jue Zhang, Huilin Qu, Congqiao Li, Sitian Qian, Weitao Du, Zhi-Ming Ma, and Tie-Yan Liu. An efficient lorentz equivariant graph neural network for jet tagging. *Journal of High Energy Physics*, 2022(7):30, 2022.

Jiaqi Han, Wenbing Huang, Tingyang Xu, and Yu Rong. Equivariant graph hierarchy-based neural networks, 2022. URL `https://arxiv.org/abs/2202.10643`.

Masanobu Horie and Naoto Mitsume. Physics-embedded neural networks: Graph neural pde solvers with mixed boundary conditions, 2023. URL `https://arxiv.org/abs/2205.11912`.

Nathan Jacobson. *Basic Algebra I*. W. H. Freeman and Company, New York, 2 edition, 1985.

Piotr Kicki, Mete Ozay, and Piotr Skrzypczyński. A computationally efficient neural network invariant to the action of symmetry subgroups. *arXiv preprint arXiv:2002.07528*, 2020.

Gyeonghoon Ko, Hyunsu Kim, and Juho Lee. Learning infinitesimal generators of continuous symmetries from data. *arXiv preprint arXiv:2410.21853*, 2024. NeurIPS 2024 version.

Risi Kondor and Shubhendu Trivedi. On the generalization of equivariance and convolution in neural networks to the action of compact groups. In *International Conference on Machine Learning*, pages 2747–2755. PMLR, 2018.

Risi Kondor, Zhen Lin, and Shubhendu Trivedi. Clebsch–gordan nets: a fully fourier space spherical convolutional neural network. *Advances in Neural Information Processing Systems*, 31, 2018.

Artem Moskalev, Anna Sepliarskaia, Ivan Sosnovik, and Arnold Smeulders. Liegg: Studying learned lie group generators. *Advances in Neural Information Processing Systems*, 35:25212–25223, 2022.

Samuel E Otto, Nicholas Zolman, J Nathan Kutz, and Steven L Brunton. A unified framework to enforce, discover, and promote symmetry in machine learning. *arXiv preprint arXiv:2311.00212*, 2023.

Jung Yeon Park, Ondrej Biza, Linfeng Zhao, Jan Willem van de Meent, and Robin Walters. Learning symmetric embeddings for equivariant world models, 2022. URL `https://arxiv.org/abs/2204.11371`.

David Ruhe, Johannes Brandstetter, and Patrick Forré. Clifford group equivariant neural networks. *Advances in neural information processing systems*, 36:62922–62990, 2023.

Sophia Sanborn, Christian Shewmake, Bruno Olshausen, and Christopher J. Hillar. Bispectral neural networks. *arXiv preprint arXiv:2209.03416*, 2022. ICLR 2023 version.

Mehran Shakerinava, Arnab Kumar Mondal, and Siamak Ravanbakhsh. Structuring representations using group invariants. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 34162–34174. Curran Associates, Inc., 2022.

Ben Shaw, Abram Magner, and Kevin R. Moon. Symmetry discovery beyond affine transformations. *arXiv preprint arXiv:2406.03619*, 2024. NeurIPS 2024 version.

J Yang, R Walters, N Dehmamy, and R Yu. Generative adversarial symmetry discovery. *arXiv preprint arXiv:2310.xxxx*, 2023. ICML 2023 version.

Jason Yim, Brian L. Trippe, Valentin De Bortoli, Emile Mathieu, Arnaud Doucet, Regina Barzilay, and Tommi Jaakkola. Se(3) diffusion model with application to protein backbone generation, 2023. URL `https://arxiv.org/abs/2302.02277`.

Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabas Poczos, Russ R Salakhutdinov, and Alexander J Smola. Deep sets. *Advances in neural information processing systems*, 30, 2017.

Allan Zhou, Tom Knowles, and Chelsea Finn. Meta-learning symmetries by reparameterization. *arXiv preprint arXiv:2007.02933*, 2020.

Checklist

1. For all models and algorithms presented, check if you include:
   (a) A clear description of the mathematical setting, assumptions, algorithm, and/or model. [Yes] We provide a clear mathematical formulation of the problem and describe the proposed model and assumptions
   (b) An analysis of the properties and complexity (time, space, sample size) of any algorithm. [No] We do not provide a formal complexity analysis, but we discuss empirical efficiency.
   (c) (Optional) Anonymized source code, with specification of all dependencies, including external libraries. [No] Source code is not included in this submission, but we plan to release it in the camera-ready version.

2. For any theoretical claim, check if you include:
   (a) Statements of the full set of assumptions of all theoretical results. [Yes] All assumptions underlying our theoretical results are explicitly stated
   (b) Complete proofs of all theoretical results. [Yes] Complete proofs are provided in the Appendix.
   (c) Clear explanations of any assumptions. [Yes] We provide clear explanations of all assumptions alongside the results.

3. For all figures and tables that present empirical results, check if you include:
   (a) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL). [Yes] We provide code and instructions to reproduce the main experimental results. Synthetic datasets are generated using the provided code, and all real-world datasets used in our experiments are publicly available.
   (b) All the training details (e.g., data splits, hyperparameters, how they were chosen). [Yes] Training details such as data splits and hyperparameters are described in the supplementary section.
   (c) A clear definition of the specific measure or statistics and error bars (e.g., with respect to the random seed after running experiments multiple times). [Yes] We do report evaluation metrics with error bars across multiple random seeds.
   (d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider). [Yes] Experiments were conducted on a machine equipped with two NVIDIA RTX A6000 GPUs (48GB memory each). The results can be reproduced on standard GPU hardware.

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets, check if you include:
   (a) Citations of the creator If your work uses existing assets. [Yes] We cite all assets used in our work with proper references.
   (b) The license information of the assets, if applicable. [Not Applicable] License information of the used assets is included where applicable.
   (c) New assets either in the supplemental material or as a URL, if applicable. [Not Applicable] We do not release new assets in this work.
   (d) Information about consent from data providers/curators. [Not Applicable] No data requiring consent was collected.
   (e) Discussion of sensible content if applicable, e.g., personally identifiable information or offensive content. [Not Applicable] The work does not involve sensitive or personally identifiable content.

5. If you used crowdsourcing or conducted research with human subjects, check if you include:
   (a) The full text of instructions given to participants and screenshots. [Not Applicable] No human participants were involved.
   (b) Descriptions of potential participant risks, with links to Institutional Review Board (IRB) approvals if applicable. [Not Applicable] No IRB approval was required.
   (c) The estimated hourly wage paid to participants and the total amount spent on participant compensation. [Not Applicable] No participant compensation was involved.

Supplementary Material

Index

A Notation and Standing Assumptions

- **Quadratic form and group definition.** Let $A \in \mathbb{R}^{n\times n}$ be a symmetric matrix. Define the generalized orthogonal group
$$G \;=\; \{\, Q \in \mathrm{GL}_n \;:\; Q^\top A Q = A \,\}.$$
- **Indefinite norm.** For $x \in \mathbb{R}^n$, define the A-norm (or pseudo-norm)
$$\|x\|_A \;=\; \mathrm{sign}(x^\top A x)\,\sqrt{|x^\top A x|}.$$
This norm generalizes the Euclidean case and may take positive or negative values depending on the signature of A.
- **Null cone.** We exclude the *null cone*
$$U \;=\; \{\, x \in \mathbb{R}^n : x^\top A x = 0 \,\},$$
which has Lebesgue measure zero. All maps and actions below are defined on $\mathbb{R}^n \setminus U$.
- **Group action on inputs.** The group G acts on the input space by the standard linear action
$$Q \cdot x \;=\; Qx.$$

- **Group action on outputs.** When the output space is matrix-valued, we consider the conjugation action

$$Q \cdot Y \;=\; QYQ^{-1},$$

which preserves the natural geometric structure of the output space.

B Additional Experiments

B.1 Implementation of the decomposition

Our method implements the decomposition

$$f(x) = \phi_s\left(\frac{x}{\|x\|_A}\right) \cdot \phi_n(\|x\|_A)$$

exactly as described in the theory. Concretely, we first compute the quadratic-form pseudo-norm

$$\|x\|_A = \operatorname{sign}(x^\top Ax)\sqrt{|x^\top Ax|},$$

then feed the normalized directional component $x/\|x\|_A$ to ϕ_s, and the scalar component $\|x\|_A$ to ϕ_n. The outputs are then combined using the appropriate group action associated with the output representation. For instance, in the vector-valued case we use

$$f(x) = Rv, \qquad R = \phi_s\left(\frac{x}{\|x\|_A}\right), \quad v = \phi_n(\|x\|_A)\,,$$

while in the matrix-valued case with conjugation action we use

$$f(x) = RQR^{-1}, \qquad R = \phi_s\left(\frac{x}{\|x\|_A}\right), \quad Q = \phi_n(\|x\|_A)\,.$$

As discussed in Section 5.2, we parameterize R using Householder reflections, for which $R^{-1} = R$, leading to improved numerical stability and reduced computational cost. Thus, the theoretical decomposition is implemented directly through standard group actions.

B.2 Experimental Details

To ensure a fair and reproducible comparison, both ϕ_s and ϕ_n are implemented as ResNets with four residual blocks and hidden dimension 128. The prediction models used for the LieGG and Augerino baselines employ the same ResNet backbone, with a comparable number of trainable parameters as our method.

All models are trained for 25 epochs using the Adam optimizer with an initial learning rate of 0.002, along with standard learning-rate decay. We vary the number of training samples and noise levels exactly as reported in the result tables. For the top-tagging task, we consider training sets of size 100k and 500k.

These design choices ensure that performance differences are not attributable to variations in model capacity or architecture. All experiments were conducted on a machine equipped with two NVIDIA RTX A6000 GPUs (48GB memory each).

Table 7: Summary of experimental setup.

Component	**Configuration**
Architecture	ResNet (4 blocks, hidden dim 128)
Baselines	Same backbone, comparable parameters
Training	Adam, lr = 0.002, 25 epochs, LR decay
Data	Varying samples and noise (see main tables)
Top-tagging	100k and 500k samples
Hardware	2× NVIDIA RTX A6000 (48GB)

B.3 Robustness Across Random Initializations

To assess whether the observed performance is stable across random initializations, we repeated the experiment **5 times for each method and configuration** on a fixed training set of size 15k, considering noise levels $\sigma \in \{0.5, 0.75, 1.0\}$. The task is regression of the matrix-valued polynomial

$$f(x) = 9(xx^\top A)^2 + 2(xx^\top A),$$

where A is the unknown quadratic form to be recovered. For each method, we report the **mean ± standard deviation** over the 5 runs.

Table 8: Multi-run results (mean ± std over 5 runs) on the matrix-valued polynomial regression task with 15k training samples. Higher cosine similarity is better, while lower test loss and projection distance are better.

Method	$\cos(A_0, A_{\text{learnt}})(\uparrow)$	**Test Loss** $(\downarrow)$	$d_{\text{PA}}(\mathfrak{g}_0, \mathfrak{g}_{\text{learnt}})(\downarrow)$
Noise level: $\sigma = 0.5$			
Ours	**0.9786 ± 0.0050**	**0.46 ± 0.24**	**0.45 ± 0.14**
LieGG	0.7606 ± 0.1011	44.16 ± 40.08	1.75 ± 0.10
Augerino	0.0063 ± 0.0141	74.74 ± 37.56	2.00 ± 0.08
Noise level: $\sigma = 0.75$			
Ours	**0.9672 ± 0.0188**	**0.48 ± 0.24**	**0.51 ± 0.15**
LieGG	0.7506 ± 0.1029	99.12 ± 89.83	1.81 ± 0.14
Augerino	0.1901 ± 0.1454	167.74 ± 84.38	2.02 ± 0.12
Noise level: $\sigma = 1.0$			
Ours	**0.9338 ± 0.0412**	**0.69 ± 0.30**	**0.82 ± 0.43**
LieGG	0.7369 ± 0.1185	176.11 ± 159.54	1.86 ± 0.10
Augerino	0.0739 ± 0.1653	297.91 ± 149.97	2.05 ± 0.07

Table 8 shows that our method consistently achieves the strongest performance across all three noise levels. In particular, it yields the highest cosine similarity, indicating more accurate recovery of the latent quadratic form, while also obtaining substantially lower test loss and smaller projection distance than both LieGG and Augerino. The standard deviations are also relatively small, suggesting that the learned symmetry and downstream predictor are stable across random restarts. Overall, these results confirm that the empirical gains of our method are not due to a favorable single run, but persist reliably across multiple independent trials.

B.4 Top-Tagging Accuracy and Timing Analysis

Some prior methods reported in the literature are not directly comparable to our setting. In particular, LieGAN does not itself output class predictions; it only learns Lie generators. Consequently, the "accuracy" often attributed to LieGAN in prior work is in fact the accuracy of a separate downstream GNN trained using the metric recovered from LieGAN, rather than an end-to-end prediction model learned by LieGAN itself. By contrast, our method jointly learns both the symmetry and the predictor in a single end-to-end model, which is a substantially more demanding task.

Similarly, methods such as LorentzNet and Clifford Nets assume the relevant symmetry structure is already known and explicitly built into the architecture. Our setting is different: the symmetry is not given and must instead be inferred from data. Therefore, comparing raw classification accuracy against models with hard-coded access to the target symmetry is not fully informative about the symmetry-discovery capability of the method.

Matched comparison using the recovered metric. To obtain a fair comparison with LieGAN, we use the following matched protocol: first learn the quadratic form A using each method, and then train the same downstream GNN using the recovered A as the metric. Under this setting, the only difference is the quality of the recovered symmetry.

Table 9 shows that our method consistently recovers a more accurate symmetry than LieGAN, as reflected in the cosine similarity values. This improved symmetry recovery also translates into better downstream classification performance. At 100k samples, our method improves over LieGAN in cosine similarity, downstream accuracy,

and downstream AUC. At 500k samples, the same trend persists, with our recovered metric again yielding higher downstream accuracy.

Table 9: Symmetry recovery and downstream performance on the top-tagging task under a matched comparison, where the quadratic form A learned by each method is used to train the same downstream GNN.

Method	$\cos(A_0, A_{\text{learnt}})(\uparrow)$	**Downstream Accuracy** ($\uparrow$)	**Downstream AUC** ($\uparrow$)
(A) 100k samples — Multi-run statistics (mean ± std)			
LieGAN	0.9912 ± 0.0056	0.9101 ± 0.0073	0.9674 ± 0.0052
Our Method	**0.9996 ± 0.0007**	**0.9261 ± 0.0028**	**0.9782 ± 0.0016**
(B) 500k samples — Single-run results			
LieGAN	0.9975	93.33%	–
Our Method	**0.9999**	**93.75%**	–

Timing analysis. We also compare the computational cost of symmetry discovery. At 100k samples, downstream GNN training takes the same amount of time for both methods, namely 0:23:11 ± 0:00:14, since the downstream architecture and training procedure are identical. The difference therefore lies in the symmetry-discovery stage. As shown in Table 10, our method is faster than LieGAN during symmetry discovery.

Table 10: Timing analysis on the top-tagging task at 100k samples.

Method	**Symmetry Discovery Time (H:M:S)**
Our Method	**0:10:05 ± 0:00:10**
LieGAN	0:17:18 ± 0:01:18

For completeness, Table 11 reports wall-clock pipeline timing at 500k samples. This comparison should be interpreted carefully, since LieGAN is inherently a two-stage pipeline whereas our method is end-to-end. In particular, LieGAN alone does not perform classification, and its final accuracy is achieved only after training an additional downstream GNN. Nevertheless, even under this broader comparison, the quadratic form recovered by our method yields better downstream accuracy when the same GNN is trained on top of it.

Table 11: Wall-clock pipeline timing and accuracy at 500k samples. The comparison should be interpreted with care, since LieGAN uses a two-stage pipeline whereas our method is trained end-to-end.

Method / Pipeline	**Time (H:M:S)**	**Accuracy** ($\uparrow$)
Our Method (joint end-to-end)	**0:37:21**	90.44%
LieGAN (symmetry only)	1:27:30	N/A
LieGAN → A-recovery → GNN	5:27:05	93.33%
Our Method → A-recovery → GNN	4:37:56	**93.75%**

Overall, these results clarify three points. First, LieGAN alone is not a classifier. Second, the commonly reported LieGAN+GNN accuracy measures the downstream classifier rather than the symmetry-discovery model itself. Third, when the comparison is matched through the same downstream GNN, our method recovers a more accurate symmetry, yields better downstream accuracy and AUC, and requires less time for symmetry discovery.

C Additional Results and Complete Proofs

Additional theoretical results and the complete proof of Theorem. 4.1 are provided here including supporting Lemmas.

C.1 Canonical Alignment in the Positive Definite Case

We begin with the positive definite case, where the quadratic form $x^\top Dx$ defines a *weighted Euclidean geometry* determined by the positive diagonal entries of D. In this setting, all directions are comparable through the group

of transformations that preserve this weighted norm. The goal of this subsection is to establish a *canonical alignment result* showing that any vector can be mapped to a fixed reference direction through an appropriate transformation in the orthogonal group $G = \{A \in \mathbb{R}^{n\times n} : A^\top DA = D\}$. This alignment construction will serve as a fundamental building block in the proof of **Theorem 4.1**, which characterizes the orbits of this group and their correspondence with quadratic-form norms.

Lemma C.1. *Let $D \in \mathbb{R}^{n\times n}$ be a diagonal matrix defined as:*

$$D = \sum_{i=1}^{n} d_i E_{i,i}, \tag{21}$$

where $d_1, \ldots, d_n > 0$, and $E_{i,j} = e_i e_j^T$ with e_i denoting the standard basis vectors in $\mathbb{R}^n$. Let $x = \sum_{i=1}^n x_i e_i \in \mathbb{R}^n$. Let G be the orthogonal group preserving the quadratic form $x^T Dx$. Then, there exists $A \in G$ such that:

$$Ax = \alpha e_1, \tag{22}$$

where α is an appropriate scaling factor such that $x^T Dx = x^T A^T DAx$.

Proof. Define $S = \sum_{i=1}^n \sqrt{d_i} e_i e_i^T$. Then, we have:

$$S^{-1} D S^{-1} = I, \tag{23}$$

where I is the identity matrix.

Consider the standard orthogonal group:

$$O(n) = \Big\{ R \in GL\big(n, \mathbb{R}\big) : R^T R = I \Big\}.$$

We know that if $y^T y = z^T z$ for any $y, z \in \mathbb{R}^n$, then there exists $R \in O(n)$ such that:

$$Ry = z.$$

Applying this result, we find $R \in O(n)$ such that:

$$RSx = \beta e_1, \tag{24}$$

where β is a scaling factor that ensures that the Euclidean norms of Sx and βe_1 are equal.

Now, define $A = S^{-1}RS$. Since S^{-1} is also a diagonal matrix, we compute:

$$\begin{aligned} Ax &= S^{-1}RSx \\ &= S^{-1}\beta e_1 \\ &= \alpha e_1, \end{aligned} \tag{25}$$

where $\alpha = \frac{\beta}{\sqrt{d_1}}$ is an appropriate scaling factor.

Next, we show that $A \in G$. Consider $A^T DA$. Since S is a diagonal matrix, $S^T = S$ and $(S^{-1})^T = S^{-1}$.

$$\begin{aligned} A^T DA &= (S^{-1}RS)^T D(S^{-1}RS) \\ &= (S^T R^T (S^{-1})^T) D(S^{-1}RS) \\ &= (SR^T S^{-1}) D(S^{-1}RS) && \text{Since } S^T = S \\ &= SR^T (S^{-1} D S^{-1}) RS && \text{By associativity} \\ &= SR^T I RS && \text{Since } S = \sqrt{D}, S^{-1}DS^{-1} = I \\ &= SR^T RS && \text{Since } R \in O(n), R^T R = I \\ &= SIS = S^2 = D. \end{aligned}$$

This shows conclusively that $A \in G$. □

Remark C.1. *The Lemma. C.1 is also applicable when $d_i < 0$, $\forall i \in [n]$. We just need to appropriately modify the matrix S.*

Lemma C.1 thus guarantees that, in the positive definite setting, every orbit under the D-orthogonal group admits a canonical representative obtained by aligning an arbitrary vector with the first coordinate axis. This serves as the foundational case for proving **Theorem 4.1**; the subsequent subsections extend this reasoning to *semi-definite* and *indefinite* quadratic forms, where certain directions may lie on null subspaces or carry opposite signs.

C.2 Canonical Alignment in the Positive Semi-Definite Case

Lemma C.2. *Let $D \in \mathbb{R}^{n\times n}$ be a diagonal matrix defined as:*

$$D = \sum_{i=1}^{k-1} d_i E_{i,i} \quad : \tag{26}$$

where $d_1, \ldots, d_{k-1} > 0$, with $k \neq 1$, and $E_{i,j} = e_i e_j^T$ with e_i denoting the standard basis vectors in $\mathbb{R}^n$. Consider a vector $y \in \mathbb{R}^n$ of the form:

$$y = ae_1 + be_k,$$

where $a, b \in \mathbb{R}$. Let $q(x; D) = x^T Dx$ be the quadratic form associated with D, and let $G = O(n, q)$ denote the orthogonal group that preserves this quadratic form, i.e.,

$$G = \{R \in GL(n, \mathbb{R}) : R^T DR = D\}.$$

Then, there exists a matrix $A \in G$ such that:

$$Ay = ae_1. \tag{27}$$

Proof. We construct A explicitly and verify the required properties. Define $R \in \mathbb{R}^{n\times n}$ as:

$$R = \sqrt{d_1}ab\, E_{k,1} - d_1 a^2\, E_{k,k} + \sum_{i\in[n],\ i\neq 1,\ i\neq k} E_{i,i}. \tag{28}$$

Define $S \in \mathbb{R}^{n\times n}$ as:

$$S = \sqrt{d_1} E_{1,1} + \sum_{i\in[n],\ i\neq 1} E_{i,i}. \tag{29}$$

Then:

$$S^{-1} = \frac{1}{\sqrt{d_1}} E_{1,1} + \sum_{i\in[n],\ i\neq 1} E_{i,i}.$$

Let $A = S^{-1}RS$. We claim that $A^T DA = D$ and $Ay = ae_1$

Step 1: Show $A^T DA = D$. Using $A = S^{-1}RS$, we compute:

$$A^T DA = SR^T S^{-1} DS^{-1} RS$$

First, compute $S^{-1}DS^{-1}$:

$$\begin{aligned} S^{-1}DS^{-1} &= \left(\frac{1}{\sqrt{d_1}} E_{1,1} + \sum_{i\in[n],\ i\neq 1} E_{i,i}\right)\left(\sum_{i=1}^{k-1} d_i E_{i,i}\right)\left(\frac{1}{\sqrt{d_1}} E_{1,1} + \sum_{i\in[n],\ i\neq 1} E_{i,i}\right) \\ &= E_{1,1} + \sum_{i=2}^{k-1} d_i E_{i,i}. \end{aligned} \tag{30}$$

Consider $RS = \left(SR^T\right)^T$,

$$\begin{aligned} RS &= \left(\sqrt{d_1}ab\,E_{k,1} - d_1a^2\,E_{k,k} + \sum_{i\in[n],\ i\neq k} E_{i,i}\right)\left(\sqrt{d_1}E_{1,1} + \sum_{i\in[n],\ i\neq 1} E_{i,i}\right) \\ &= d_1abE_{k,1} + \sqrt{d_1}E_{1,1} - d_1a^2E_{k,k} + \sum_{i\in[n],\ i\neq 1,\ i\neq k} E_{i,i}. \end{aligned} \tag{31}$$

Thus, we have,

$$\begin{aligned} A^TDA &= \left(SR^T\right)\left(S^{-1}DS^{-1}\right)(RS) \\ &= \left(d_1abE_{1,k} + \sqrt{d_1}E_{1,1} - d_1a^2E_{k,k} + \sum_{i\in[n],\ i\neq 1,\ i\neq k} E_{i,i}\right)\left(E_{1,1} + \sum_{i=2}^{k-1} d_iE_{i,i}\right)(RS) \\ &= \left(\sqrt{d_1}E_{1,1} + \sum_{i=2}^{k-1} d_iE_{i,i}\right)(RS) \\ &= \left(\sqrt{d_1}E_{1,1} + \sum_{i=2}^{k-1} d_iE_{i,i}\right)\left(d_1abE_{k,1} + \sqrt{d_1}E_{1,1} - d_1a^2E_{k,k} + \sum_{i\in[n],\ i\neq 1,\ i\neq k} E_{i,i}\right) \\ &= d_1E_{1,1} + \sum_{i=2}^{k-1} d_iE_{i,i} \\ &= \sum_{i=1}^{k-1} d_iE_{i,i} \\ &= D \end{aligned} \tag{32}$$

Consider $A = S^{-1}RS$

$$\begin{aligned} A &= S^{-1}(RS) \\ &= S^{-1}\left(d_1abE_{k,1} + \sqrt{d_1}E_{1,1} - d_1a^2E_{k,k} + \sum_{i\in[n],\ i\neq 1,\ i\neq k} E_{i,i}\right) \\ &= \left(\frac{1}{\sqrt{d_1}}E_{1,1} + \sum_{i\in[n],\ i\neq 1} E_{i,i}\right)\left(d_1abE_{k,1} + \sqrt{d_1}E_{1,1} - d_1a^2E_{k,k} + \sum_{i\in[n],\ i\neq 1,\ i\neq k} E_{i,i}\right) \\ &= E_{1,1} + d_1abE_{k,1} - d_1a^2E_{k,k} + \sum_{i\in[n],\ i\neq 1,\ i\neq k} E_{i,i}. \end{aligned} \tag{33}$$

Step 2: Verify $Ay = ae_1$

We are given the vector $y = ae_1 + be_k$. Since $k \neq 1$, ensuring that e_1 and e_k are distinct standard basis vectors. Our goal is to verify that for the constructed matrix A, the transformation yields $Ay = ae_1$.

The matrix A is given by:

$$A = E_{1,1} + d_1abE_{k,1} - d_1a^2E_{k,k} + \sum_{i\in[n],i\neq 1,i\neq k} E_{i,i}$$

Let's apply A to y by distributing the terms:

$$Ay = \left(E_{1,1} + d_1abE_{k,1} - d_1a^2E_{k,k} + \sum_{i\in[n],i\neq 1,i\neq k} E_{i,i}\right)(ae_1 + be_k)$$

We evaluate the effect of each component of A on y, noting that the elementary matrix $E_{i,j}$ acts on a basis vector e_m as $E_{i,j}e_m = \delta_{jm}e_i$, where δ_{jm} is the Kronecker delta.

1. **First Term:** $E_{1,1}(ae_1 + be_k) = aE_{1,1}e_1 + bE_{1,1}e_k = ae_1 + 0 = \boldsymbol{ae_1}$.
2. **Second Term:** $d_1abE_{k,1}(ae_1 + be_k) = ad_1abE_{k,1}e_1 + bd_1abE_{k,1}e_k = ad_1abe_k + 0 = \boldsymbol{d_1a^2be_k}$.
3. **Third Term:** $-d_1a^2E_{k,k}(ae_1 + be_k) = -ad_1a^2E_{k,k}e_1 - bd_1a^2E_{k,k}e_k = 0 - bd_1a^2e_k = \boldsymbol{-d_1a^2be_k}$.
4. **Fourth Term:** The sum $\sum_{i\in[n],i\neq 1,i\neq k} E_{i,i}$ acts as the identity on basis vectors with indices other than 1 and k. Therefore, its product with $(ae_1 + be_k)$ is zero.

Summing the results from each component:

$$\begin{aligned} Ay &= \boldsymbol{ae_1} + \boldsymbol{d_1a^2be_k} - \boldsymbol{d_1a^2be_k} \\ &= ae_1. \end{aligned}$$

Thus, we have verified that $Ay = ae_1$, as required.

Conclusion: We have constructed $A \in G$ such that $Ay = ae_1$. Therefore, the lemma is proved.

□

Remark C.2. *The Lemma. C.2 is also applicable when $d_i < 0$, $\forall i \in \{1, \ldots, k\}$. We just need to appropriately modify the matrices S and R.*

C.3 Canonical Alignment in the Indefinite Case

Lemma C.3. *Let $D \in \mathbb{R}^{n\times n}$ be a diagonal matrix defined as:*

$$D = \sum_{i=1}^{k-1} d_iE_{i,i} - \sum_{i=k}^{l} d_iE_{i,i}, \tag{34}$$

where $d_1, \ldots, d_l > 0$, $k < l$, and e_i denotes the standard basis vectors in $\mathbb{R}^n$. Let $x = ae_1 + be_k \in \mathbb{R}^n$. Let G be the orthogonal group preserving the quadratic form x^TDx. Then, there exists $A \in G$ such that:

$$Ax = \begin{cases} \alpha e_1, & x^TDx > 0, \\ \alpha e_k, & x^TDx < 0, \end{cases} \tag{35}$$

where α is an appropriate scaling factor such that $x^TDx = x^TA^TDAx$.

Proof. We will prove the result for the case $x^TDx > 0$. Similar steps follow for the other case. The value of x^TDx is:

$$x^TDx = d_1a^2 - d_kb^2. \tag{36}$$

Since $d_1a^2 - d_kb^2 > 0$, define the scalar β as:

$$\beta = \frac{1}{\sqrt{d_1a^2 - d_kb^2}}. \tag{37}$$

Define an $n \times n$ matrix S as:

$$S = \sqrt{d_1}E_{1,1} + \sqrt{d_k}E_{k,k} + \sum_{i\in[n],\ i\neq 1,k} E_{i,i}. \tag{38}$$

Now define another $n \times n$ matrix R as:

$$R = \beta\left(\sqrt{d_1}aE_{1,1} - \sqrt{d_k}bE_{1,k} - \sqrt{d_k}bE_{k,1} + \sqrt{d_1}aE_{k,k} + \sum_{i\in[n],\ i\neq 1,k} E_{i,i}\right). \tag{39}$$

Set $A = S^{-1}RS$. Our goal is to show that $A^TDA = D$ and $Ax = \alpha e_1$.

Step 1: Show that $A^TDA = D$. Consider A^TDA:

$$\begin{aligned} A^TDA &= \big(SR^TS^{-1}\big)D\big(S^{-1}RS\big) \\ &= (RS)^T\big(S^{-1}DS^{-1}\big)(RS). \end{aligned} \tag{40}$$

We note that:

$$S^{-1}DS^{-1} = E_{1,1} - E_{k,k} + \sum_{i=2}^{k-1} d_i E_{i,i} - \sum_{i=k+1}^{l} d_i E_{i,i}, \tag{41}$$

and:

$$RS = \beta\left(d_1 a E_{1,1} - d_k b E_{1,k} - \sqrt{d_1 d_k} b E_{k,1} + \sqrt{d_1 d_k} a E_{k,k} + \sum_{i\in[n],\ i\neq 1,k} E_{i,i}\right). \tag{42}$$

Substituting (41) and (42) into (40), we get:

$$\begin{aligned} A^TDA &= \beta^2\left[\frac{1}{\beta^2}d_1E_{1,1} - \frac{1}{\beta^2}d_kE_{k,k} + \sum_{i=2}^{k-1} d_iE_{i,i} - \sum_{i=k+1}^{l} d_iE_{i,i}\right] \\ &= \sum_{i=1}^{k-1} d_iE_{i,i} - \sum_{i=k}^{l} d_iE_{i,i} \\ &= D. \end{aligned} \tag{43}$$

Step 2: Show that $Ax = \alpha e_1$. From (42) and (38), we compute:

$$\begin{aligned} A &= S^{-1}RS \\ &= \beta\left(\sqrt{d_1}aE_{1,1} - \frac{d_k}{\sqrt{d_1}}bE_{1,k} - \sqrt{d_1}bE_{k,1} + \sqrt{d_1}aE_{k,k} + \sum_{i\in[n],\ i\neq 1,k} E_{i,i}\right). \end{aligned} \tag{44}$$

Thus, the value of Ax is:

$$\begin{aligned} Ax &= \beta\left[\left(\sqrt{d_1}a^2 - \frac{d_k}{\sqrt{d_1}}b^2\right)e_1\right] \\ &= \frac{\beta}{\sqrt{d_1}}\Big[\big(d_1a^2 - d_kb^2\big)e_1\Big] \\ &= \frac{\beta}{\sqrt{d_1}}\left[\frac{1}{\beta^2}e_1\right] \\ &= \frac{1}{\beta\sqrt{d_1}}e_1. \end{aligned} \tag{45}$$

Hence, $Ax = \alpha e_1$, where $\alpha = \frac{1}{\beta\sqrt{d_1}} = \frac{\sqrt{d_1a^2 - d_kb^2}}{\sqrt{d_1}}$. □

C.4 Orbit Space and Quadratic-Form Norm Correspondence

Building upon Lemmas C.1, C.2, and C.3, we now establish Theorem 4.1, which characterizes the structure of group orbits under quadratic-form–preserving transformations.

Theorem 4.1. *Let A be a non-zero $n\times n$ matrix, and define $U := \big\{x \in \mathbb{R}^n : x^TAx = 0\big\}$. Let G be an orthogonal group that preserves the quadratic form x^TAx, and consider the action of G on $\mathbb{R}^n \setminus U$. For a given $c \in \mathbb{R}\setminus\{0\}$, define the set $\mathcal{O}_c := \big\{x \in \mathbb{R}^n : x^TAx = c\big\}$. Then, the set of orbits under the action of G is given by*

$$\mathcal{O}(G) = \big\{\mathcal{O}_c : c \in \mathbb{R}^n \setminus \{0\} \ \ \&\ \ \mathcal{O}_c \neq \varnothing\big\}. \tag{6}$$

Proof. We have to show that each set $\mathcal{O}_c$ is a unique orbit. Suppose $x \in \mathcal{O}_c$, then by definition of orbit under the action of given orthogonal group, the orbit of x is a subset of $\mathcal{O}_c$, i.e., $G \cdot x \subset \mathcal{O}_c$. Now, we need to show the subset relation in reverse direction, i.e., $\mathcal{O}_c \subset G \cdot x$. This is equivalent to proving : if $x^T Ax = y^T Ay$, then x and y must lie in the same orbit.

To achieve this, we first diagonalize the symmetric matrix A. Let $UAU^T = D$, where D is a diagonal matrix. By selecting an appropriate permutation matrix P, we further transform D into a canonical form which partitions D into the positive, negative, and zero eigenvalues of A. Without loss of generality, we assume this structure for A and proceed with the proof.

Let $x \in \mathbb{R}^n$. We partition the diagonal matrix D into positive, negative, and zero components:

$$D = \begin{bmatrix} D_p & & \\ & D_n & \\ & & D_z \end{bmatrix},$$

where $D_z = \mathbf{0}$ (zero matrix). Correspondingly, partition x as:

$$x = \begin{bmatrix} x_p \\ x_n \\ x_z \end{bmatrix},$$

so that

$$\begin{aligned} \|x\|_D^2 &= \left(x^T Dx\right) \\ &= \left(x_p^T D_p x_p\right) + \left(x_n^T D_n x_n\right) + \left(x_z^T D_z x_z\right) \\ &= \|x_p\|_{D_p}^2 + \|x_n\|_{D_n}^2 + \|x_z\|_{D_z}^2. \end{aligned} \tag{46}$$

Since $\|x_z\|_{D_z}^2 = 0$ because $D_z = \mathbf{0}$, we have $\left(x_z^T D_z x_z\right) = 0$.

Step 1: Positive and Negative diagonal entries

Apply Lemma C.1 to $\left(D_p, x_p\right)$ and $\left(D_n, x_n\right)$, yielding matrices A_p and A_n such that:

$$A_p^T D_p A_p = D_p, \quad A_p x_p = \alpha e_1,$$

and

$$A_n^T D_n A_n = D_n, \quad A_n x_n = \beta_1 e_1.$$

Step 2: Zero diagonal entries

Since $\|x_z\|_{D_z}^2 = 0$, for any $y \in \mathbb{R}^{n-k-l}$, we can find an orthonormal matrix A_z (i.e., $A_z^T A_z = I$) such that:

$$A_z^T D_z A_z = D_z = \mathbf{0}, \quad A_z x_z = \beta_2 e_1.$$

Define A_1 as:

$$A_1 = \begin{bmatrix} A_p & & \\ & A_n & \\ & & A_z \end{bmatrix}.$$

Since $A_1^T D A_1 = D$, we have $A_1 \in G$. Moreover,

$$A_1 x = \begin{bmatrix} A_p & & \\ & A_n & \\ & & A_z \end{bmatrix} \begin{bmatrix} x_p \\ x_n \\ x_z \end{bmatrix} = \begin{bmatrix} A_p x_p \\ A_n x_n \\ A_z x_z \end{bmatrix} = \alpha e_1 + \beta_1 e_{k+1} + \beta_2 e_{l+1}.$$

Step 3: Combining Negative and Zero Components

Concatenate x_n and x_z:

$$x_{nz} = \beta_1 e_1 + \beta_2 e_{l-k+1}.$$

Combine D_n and D_z as:

$$D_{nz} = \begin{bmatrix} D_n & \\ & D_z \end{bmatrix}.$$

Applying Lemma C.2 to $\big(D_{nz}, x_{nz}\big)$, we find A_{nz} such that:

$$A_{nz}^T D_{nz} A_{nz} = D_{nz}, \quad A_{nz} x_{nz} = \beta e_1.$$

Define A_2 as:

$$A_2 = \begin{bmatrix} I & \\ & A_{nz} \end{bmatrix}.$$

Since $A_2^T D A_2 = D$, we have $A_2 \in G$. Now:

$$A_2 A_1 x = \alpha e_1 + \beta e_{k+1}.$$

Step 4: Aligning with Canonical Basis

Applying Lemma C.3 to $\big(D, A_2 A_1 x\big)$ (for the case of $x^T D x > 0$), we find A_3 such that:

$$A_3^T D A_3 = D, \quad A_3(A_2 A_1 x) = \gamma e_1. \tag{47}$$

Similar steps hold in the case of $x^T D x < 0$, i.e., we can find A_3 such that $A_3^T D A_3 = D \quad \& \quad A_3(A_2 A_1 x) = \gamma e_{k+1}$. Since $A_1, A_2, A_3 \in G$, their product $A_3 A_2 A_1 \in G$. Let, $W = A_3 A_2 A_1$. Then, we can write,

$$W^T D W = D, \quad W x = \gamma e_1. \tag{48}$$

Conclusion

Since, the Eq. (48) holds for any $x \in \mathbb{R}^n$ having the same value for $\|x\|_D$, we can conclude that $x, y \in \mathbb{R}^n$ lie in the same orbit if $\big(\|x\|_D\big) = \big(\|y\|_D\big)$.

□

C.5 Decomposition into Scale- and Norm-Invariant Components

Building upon the orbit characterization established in Theorem 4.1, we now derive our main result: a canonical decomposition of any G-equivariant function into *scale-invariant* and *norm-invariant* components. The orbit-space analysis reveals that the orbit of each input vector is uniquely determined by its quadratic-form norm. In the following result, we further show that the combination of this norm and the corresponding normalized direction completely characterizes each input, thereby enabling the desired factorization of equivariant functions.

The following theorem formalizes this decomposition result.

Theorem 4.2. *Let A be a non-zero $n \times n$ matrix, and define $U := \big\{x \in \mathbb{R}^n : x^T A x = 0\big\}$. Any G-equivariant function $f : \mathbb{R}^n \setminus U \to \mathbb{R}^m$, where G is an orthogonal group that preserves the quadratic form $x^T A x$, can be expressed as,*

$$f(x) = \phi_s\left(\frac{x}{\|x\|_A}\right) \cdot \phi_n\big(\|x\|_A\big), \tag{7}$$

for some functions $\phi_s : \mathbb{R}^n \to G$ and $\phi_n : \mathbb{R} \to \mathbb{R}^m$. Here, $\|z\|_A$ denotes the norm associated with the quadratic form A, defined by $\|z\|_A = \operatorname{sign}(z^T A z)\sqrt{|z^T A z|}$. Moreover, if the action of G on the output space is faithful, then ϕ_s is itself G-equivariant.

Proof. Since $f(x)$ is equivariant under the action of G, we have

$$f\big(g \cdot x\big) = g \cdot f(x), \quad \forall g \in G, x \in \mathbb{R}^n. \tag{49}$$

This can be rewritten in terms of a canonical form as,

$$f(y) = \phi_1(y) \cdot \phi_2\big(r(y)\big), \tag{50}$$

where,

$$y = \phi_1(y) \cdot r(y). \tag{51}$$

Here, $r(y) \in \mathbb{R}^n \setminus U$ is a representative element chosen from the entire orbit of y, denoted by $G \cdot y$. The function $\phi_1 : \mathbb{R}^n \setminus U \to G$ maps each point y to an element of G that aligns y with its orbit representative $r(y)$.

Since, $r(y)$ is the same for every element in the orbit of y, it follows that $\phi_2 \circ r$ is G-invariant. Therefore, applying Theorem 4.1, we can express this as

$$\left(\phi_2 \circ r\right)(x) = \phi_n\left(\|x\|_A\right), \tag{52}$$

for some function ϕ_n.

Next, we select a representative element $r(y)$ such that it satisfies the homogeneity condition $r(\alpha y) = \alpha\, r(y)$ for any scalar $\alpha \in \mathbb{R}$ with $\alpha > 0$. This can be achieved by setting $r(y) = \alpha e_i$, where $\alpha = \|y\|_A$ represents the norm of y with respect to the matrix A, and the choice of the standard basis vector e_i depends on the sign of $\|y\|_A$, denoted by $\text{sign}\left(\|y\|_A\right)$.

This specific selection of $r(y)$ implies that if we define,

$$\phi_1(\alpha y) = \phi_1(y), \tag{53}$$

then the following two equations, derived from Eq. (51) and the chosen form of $r(y)$, will hold consistently:

$$\alpha y = \alpha\big(\phi_1(y) \cdot r(y)\big) = \phi_1(y) \cdot \alpha r(y), \tag{54}$$

$$\alpha y = \phi_1\big((\alpha y)\big) \cdot \alpha r(y). \tag{55}$$

Thus, the Eq. (53) shows that ϕ_1 is scale-invariant. Hence, we can write,

$$\phi_1(x) = \phi_s\left(\frac{x}{\|x\|_A}\right), \tag{56}$$

for some function ϕ_s.

Finally, substituting Eq. (52) and (56) into Eq. (50), we obtain,

$$f(x) = \phi_s\left(\frac{x}{\|x\|_A}\right) \cdot \phi_n\big(\|x\|_A\big).$$

It remains to show the equivariance of ϕ_s under the additional assumption that the output action of G is faithful. Let $z = x/\|x\|_A$. Since G preserves the quadratic form, we have

$$\|g \cdot x\|_A = \|x\|_A,$$

and since $\phi_n(\|x\|_A)$ depends only on the invariant norm, it is G-invariant. Using the equivariance of f, we obtain

$$\phi_s(g \cdot z) \cdot \phi_n(\|x\|_A) = f(g \cdot x) \tag{57}$$

$$= g \cdot f(x) \tag{58}$$

$$= g \cdot [\phi_s(z) \cdot \phi_n(\|x\|_A)] \tag{59}$$

$$= (g \cdot \phi_s(z)) \cdot \phi_n(\|x\|_A). \tag{60}$$

Since both sides act on the same $\phi_n(\|x\|_A)$, and the output action is faithful, it follows that

$$\phi_s(g \cdot z) = g \cdot \phi_s(z).$$

Thus ϕ_s is G-equivariant.

□

C.6 Regularity conditions

Proposition 4.1. *Consider Theorem 4.2. Suppose A is either a positive or a negative definite matrix. Then, ϕ_n and ϕ_s are smooth (C^∞) whenever f is smooth (C^∞). If A is neither a positive nor a negative definite matrix, then ϕ_n and ϕ_s are continuous (C^0) whenever f is continuous (C^0).*

Proof. The proof relies on analyzing the regularity of the components in the rearranged decomposition from Theorem 4.2.

1. From the proof of Theorem 4.2, we can isolate the norm-invariant component ϕ_n by applying the inverse of the group element returned by ϕ_s.

$$\phi_n(\|x\|_A) = \left(\phi_s\left(\frac{x}{\|x\|_A}\right)\right)^{-1} f(x) \tag{61}$$

   This equation shows that the regularity of ϕ_n depends on the regularity of f and the map $x \mapsto \phi_s\left(\frac{x}{\|x\|_A}\right)$. Since ϕ_s maps to the Lie group G, and both group action and group inversion are smooth operations, the regularity of the expression is determined by the map $x \mapsto \frac{x}{\|x\|_A}$ and the function $f(x)$ itself.

2. **Case 1: A is Positive or Negative Definite**
   If matrix A is positive or negative definite, the quadratic form x^TAx is non-zero for all non-zero $x \in \mathbb{R}^n$. This means the set $U := \{x \in \mathbb{R}^n : x^TAx = 0\}$ contains only the zero vector. The domain of analysis is $\mathbb{R}^n \setminus U = \mathbb{R}^n \setminus \{0\}$.

   The norm is defined as $\|x\|_A = \text{sign}(x^TAx)\sqrt{|x^TAx|}$. When A is definite, $\text{sign}(x^TAx)$ is constant for all $x \neq 0$. The expression inside the square root, $|x^TAx|$, is a smooth and strictly positive polynomial for $x \neq 0$. The square root of a smooth, strictly positive function is smooth.

   Therefore, the map $x \mapsto \|x\|_A$ is smooth on $\mathbb{R}^n \setminus \{0\}$. Consequently, the map $x \mapsto \frac{x}{\|x\|_A}$ is a composition of smooth functions and is itself smooth[cite: 622]. Since the group action is smooth, the function $x \mapsto \left(\phi_s\left(\frac{x}{\|x\|_A}\right)\right)^{-1}$ is also smooth.

   If we assume f is smooth (C^∞), then the right-hand side of Eq. (61) is a product of smooth functions, which is smooth. Thus, ϕ_n must also be smooth[cite: 624].

3. **Case 2: A is Indefinite**
   If A is neither positive nor negative definite, the set U contains non-zero vectors. The function $g(z) = \sqrt{|z|}$ is continuous everywhere but not differentiable at $z = 0$. Because x^TAx can be zero for non-zero x, the map $x \mapsto \|x\|_A$ is continuous on $\mathbb{R}^n$ but is not differentiable on the set U. The map $x \mapsto \frac{x}{\|x\|_A}$ is therefore continuous on its domain $\mathbb{R}^n \setminus U$.

   If we assume f is continuous (C^0), then the right-hand side of Eq. (61), being a product and composition of continuous functions, is continuous. It follows that ϕ_n must also be continuous.

This covers both cases and completes the proof. □

C.7 Extension to Diagonal action

Theorem C.4 (Classification of p-tuples up to the A-orthogonal group, cf. (Jacobson, 1985, Chapter 6))**.** *Let A be a non-degenerate real symmetric $n \times n$ matrix, and consider the bilinear form $\langle x, y\rangle_A = x^\top A\, y$ on $\mathbb{R}^n$. Suppose $(x_1, \ldots, x_p)$ and $(y_1, \ldots, y_p)$ are two ordered p-tuples in $\mathbb{R}^n$. Then the following are equivalent:*

1. *They have the same A-Gram matrix, i.e.*

$$x_i^\top A\, x_j \;=\; y_i^\top A\, y_j \quad \textit{for all } 1 \le i, j \le p.$$

2. *There exists an invertible linear map $Q \in \mathrm{GL}(n, \mathbb{R})$ such that*

$$Q^\top A\, Q \;=\; A \quad \textit{and} \quad Q\, x_i \;=\; y_i \quad \textit{for all } i.$$

Equivalently, $(x_1, \ldots, x_p)$ *and* $(y_1, \ldots, y_p)$ *lie in the same orbit under the action of the group*

$$G \;=\; \{\, Q \in \mathrm{GL}(n, \mathbb{R}) : Q^\top A\, Q = A\}.$$

Theorem 4.3 (Extended Decomposition for G-Equivariant Functions on Tuples)**.** *Let* A *be a non-singular* $n \times n$ *real matrix and define Let* G *be an orthogonal group as defined in Eq.* (5)*. Let* $X \subset \mathbb{R}^n$ *be a* G*-invariant set. For any positive integer* p*, consider the diagonal action of* G *on the product space* $X_p := \underbrace{X \times X \times \cdots \times X}_{p \text{ times}}$*, defined by,* $\forall g \in G$

$$g \cdot (x_1, \ldots, x_p) = (g \cdot x_1,\, \ldots,\, g \cdot x_p).$$

Then every G*-equivariant function* $f : X_p \to \mathbb{R}^m$ *can be decomposed as*

$$f(x) = \phi_s\left(\frac{x_1}{\|x_1\|_A}\right) \cdot \phi_n\big(\mathrm{Gram}(x)\big), \tag{8}$$

where $x = \Big(x_1, \ldots, x_p\Big)$ *and Gram matrix*

$$\mathrm{Gram}(x_1, \ldots, x_p) = \left[x_i^T A x_j\right]_{i,j \in [p]}$$

encodes all pairwise inner products with respect to the quadratic form defined by A*.*

Proof. In the proof of Theorem 4.2, we utilized the orbit correspondence result (Theorem 4.1) to establish the norm invariance part (ϕ_n) of the function decomposition. In the setting of the diagonal action on tuples, the invariance part (ϕ_n) of the decomposition follows immediately from Theorem C.4. The remainder of the proof proceeds analogously to that of Theorem 4.2. □

C.8 Householder reflection

Proposition 5.1. *Let* f *be a* G*-equivariant function where the group action on the output is a conjugation, as defined in Eq.* (13)*. Then, the function* $f(x)$ *can be expressed in the symmetric form:*

$$f(x) = \phi_s\left(\frac{x}{\|x\|_A}\right) \phi_n\left(\|x\|_A\right) \phi_s\left(\frac{x}{\|x\|_A}\right) \tag{14}$$

Proof. The proof combines the canonical decomposition from Theorem 4.2 with the specific properties of the group action and the scale-invariant component.

1. **Canonical Decomposition:** According to Theorem 4.2, any G-equivariant function $f : X \to Y$ can be written as the action of a scale-invariant component on a norm-invariant component:

$$f(x) = \phi_s\left(\frac{x}{\|x\|_A}\right) \cdot \phi_n(\|x\|_A) \tag{62}$$

   where $\phi_s : X \to G$ and $\phi_n : \mathbb{R} \to Y$. The symbol '$\cdot$' denotes the group action of G on the output space Y.

2. **Group Action as Conjugation:** The proposition states that the group action on the output is conjugation. For a group element $g \in G$ and an output object $Y \in Y$, this action is defined as $g \cdot Y = gYg^{-1}$. Applying this definition to our decomposition in Eq. (62), the action of the group element $\phi_s(\ldots)$ on the output $\phi_n(\ldots)$ is:

$$f(x) = \phi_s\left(\frac{x}{\|x\|_A}\right) \phi_n(\|x\|_A) \left(\phi_s\left(\frac{x}{\|x\|_A}\right)\right)^{-1} \tag{63}$$

3. **Householder Reflection Property:** As discussed in Section 5.2, the scale-invariant component $\phi_s(\frac{x}{\|x\|_A})$ is equivalent to a Householder reflection. A key property of a Householder reflection matrix R is that it is its own inverse, i.e., $R = R^{-1}$. Therefore, we have:

$$\phi_s\left(\frac{x}{\|x\|_A}\right) = \left(\phi_s\left(\frac{x}{\|x\|_A}\right)\right)^{-1} \tag{64}$$

4. **Final Form:** By substituting the property from Eq. (64) into Eq. (63), we replace the inverse term with the term itself:

$$\begin{aligned} f(x) &= \phi_s\left(\frac{x}{\|x\|_A}\right)\phi_n(\|x\|_A)\left(\phi_s\left(\frac{x}{\|x\|_A}\right)\right)^{-1} \\ &= \phi_s\left(\frac{x}{\|x\|_A}\right)\phi_n(\|x\|_A)\phi_s\left(\frac{x}{\|x\|_A}\right) \end{aligned}$$

This yields the symmetric form of the function as stated in the proposition. □