DISCOVERING SYMMETRIES IN NEURAL NETWORK PARAMETER SPACES

Anonymous authors
Paper under double-blind review

ABSTRACT

Parameter space symmetries are important for understanding neural networks' loss landscape, training dynamics, and generalization. However, systematically identifying these symmetries remains a challenge. In this paper, we formalize data-dependent parameter symmetries and characterize loss invariance and the group-action axioms through infinitesimal conditions, which provide objectives for jointly learning group generators and nonlinear action maps. Our framework systematically uncovers parameter symmetries, including previously unknown ones. To study larger networks, we establish conditions under which subnetwork symmetries extend to the full model. The same construction gives an explicit family of finite-batch symmetries, providing both analytical examples and a foundation for discovery through small subnetworks. Using the infinitesimal characrization and subnetwork construction, we implement a framework for automated discovery of parameter symmetries, and successfully uncovered symmetries in various architectures, including pretrained transformer models.

1 INTRODUCTION

Parameter space symmetry, or loss-invariant transformation of parameters, influences various aspects of deep learning theory. Continuous symmetry connects groups to their orbits, revealing topological properties such as the dimension (Zhao et al., 2023) and connectedness (Zhao et al., 2025) of the minimum. Parameter symmetry also influences training dynamics through the associated conserved quantities of gradient flow (Kunin et al., 2021) and by steering stochastic gradient descent towards certain favored solutions (Ziyin, 2024). Additionally, symmetry provides a tool to perform optimization within a loss level set, with successful applications in accelerating optimization (Armenta et al., 2023; Zhao et al., 2022). Other applications of parameter symmetry include model compression (Ganev et al., 2022; Sourek et al., 2021), model alignment and merging (Zhang et al., 2025), efficient sampling in Bayesian neural networks (Wiese et al., 2023), and equivariant architectures for weight space learning (Navon et al., 2023; Zhou et al., 2023).

Despite the wide range of applications, our knowledge of parameter space symmetries remains limited. In particular, known symmetries often cannot account for all loss-invariant parameter transformations. While several frameworks have been developed to unify known symmetries, whether the symmetries in current literature are complete remains an open question (Zhao et al., 2026). The lack of a systematic approach necessitates deriving symmetries from scratch for each new architecture, creating barriers for broader application of parameter symmetries.

In this paper, we present an automated approach for discovering parameter space symmetries and their group actions in neural networks. To define the search space, we formalize the definition of data-dependent symmetries and derive infinitesimal conditions for loss invariance and the group-action axioms. These characterizations connect local parameter changes to finite transformations and provide differentiable objectives for learning nonlinear action maps. We also bound finite composition and loss errors in terms of the corresponding differential residuals.

While directly searching for symmetries in modern architectures with billions of parameters is prohibitively expensive, we show that large networks often inherit symmetries from their components or subnetworks. Analyzing these smaller networks reduces the dimension of the discovery problem. However, a symmetry of a smaller component need not automatically induce a symmetry of the full network, so this reduction requires understanding when local symmetries can be lifted. For smooth

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.

two-layer blocks, we prove that a class of output-preserving transformations can be decomposed into transformations of small subnetworks. The proof also constructs a new family of data-dependent symmetries. These results connect the structure of the network to both the construction and the decomposition of its symmetries.

Our main contributions are:

- Exact infinitesimal characterizations of data-dependent parameter symmetries, together with bounds on finite errors when the differential conditions hold approximately.
- Conditions for lifting subnetwork symmetries and decomposing a class of symmetries into small-subnetwork symmetries, together with an explicit family of finite-batch actions.
- A framework for automated discovery of parameter symmetries building on the infinitesimal characterization and subnetwork construction, with demonstrations of learned nonlinear actions in pretrained transformers and other architectures.

2 Related Work

Parameter space symmetry. Parameter symmetries are loss-invariant transformations on neural network parameters, often in the form of group actions. Symmetry often arises from equivariance of common activation functions (Godfrey et al., 2022), including invertible linear transformations in linear networks, rescaling in homogeneous networks (Badrinarayanan et al., 2015; Du et al., 2018; Petzka et al., 2020), radial rescaling in radial neural networks (Ganev et al., 2022), and translation in softmax and scaling in batchnorm functions (Kunin et al., 2021). Permutation and sign flips are often the only function-preserving transformations of tanh networks (Chen et al., 1993). ReLU networks, however, possess symmetries beyond the well-known rescaling (Grigsby et al., 2023). The existence and number of symmetries in most other architectures remain as open questions.

Data-dependent symmetry. While the above symmetries preserve the loss on all data, a relaxed definition, data-dependent symmetry, only requires loss invariance on a subset of data. Zhao et al. (2023) gave examples of such symmetries with nontrivial data dependency, though their constructions are limited to minibatches of size one and are difficult to generalize across architectures. This motivates more systematic discovery methods. The concept of a symmetry dependent on data has also appeared in adjacent fields. For example, Moskalev et al. (2023) observe that learned data invariance in neural networks is strongly conditioned on data and breaks under data distribution drift; Sonoda et al. (2023) define a joint group action on data and parameters as part of a new proof of universal approximation theory.

Discovering and measuring symmetry. Various work explores learning continuous symmetries by identifying generators of Lie groups (Krippendorf & Syvaeri, 2020; Moskalev et al., 2022; Dehmamy et al., 2021; Yang et al., 2023; Gabel et al., 2023), including cases with nonlinear group actions (Yang et al., 2024; Shaw et al., 2024; 2025; Hu et al., 2025; Ko et al., 2024). We build on this approach to discover data-dependent group action in high-dimensional parameter spaces. While learning discrete symmetry (Zhou et al., 2021; Karjol et al., 2024) and distributions of symmetry (Benton et al., 2020; Romero & Lohit, 2022; Urbano & Romero, 2023) are also relevant, they are not the primary focus of this paper.

Extracted symmetry is often evaluated locally, by measuring function changes under infinitesimal symmetry transformations (Gruver et al., 2022) or by comparing tangent spaces of orbits under the learned group and the true symmetry group (Portilheiro, 2023). We use local invariance, similar to that defined in (Gruver et al., 2022; Moskalev et al., 2022), as a preservation objective and additionally enforce the infinitesimal action law to learn consistent finite transformations.

3 Parameter Space Symmetry

In this section, we formalize the concept of data-dependent parameter symmetries and characterize them through their infinitesimal generators. The infinitesimal characterization identifies when a smooth parameter map is a loss-preserving group action, and yields the differentiable objectives used for discovery in Section 5.

3.1 Data-Dependent Group Action and Symmetry

Let Θ be the space of parameters and $\mathcal{D}$ be the space of data. In this paper, we consider loss functions of the form $L\colon \Theta \times \mathcal{D} \to \mathbb{R}$, which map parameters and a single data point to a real number. We also use the same notation when the loss is applied to multiple data points at once $L\colon \Theta \times \mathcal{D}^d \to \mathbb{R}^d$ for $d \in \mathbb{N}$. The same symmetry definition applies to vector- or matrix-valued network outputs, with invariance required componentwise.

Let G be a group. Consider a map a, which defines a map for every data batch of size $d \in \mathbb{Z}^+$:

$$\begin{aligned} a\colon \mathcal{D}^d &\to (G \times \Theta \to \Theta) \\ X &\mapsto (a_X\colon g, \theta \mapsto \theta'). \end{aligned} \tag{1}$$

The map a is a *generalized group action* on Θ if a_X is a group action for every data batch X, meaning that it satisfies the following axioms:

$$\begin{aligned} \text{identity:} \quad & a_X(I, \theta) = \theta, \qquad \forall X \in \mathcal{D}^d,\ \forall \theta \in \Theta. \\ \text{associative law:} \quad & a_X(g_2, a_X(g_1, \theta)) = a_X(g_2 g_1, \theta), \qquad \forall g_1, g_2 \in G,\ \forall X \in \mathcal{D}^d,\ \forall \theta \in \Theta. \end{aligned}$$

We introduce our first definition formalizing the notion of data-dependent symmetry. A group action a is *parameter space symmetry of* L if it additionally satisfies

$$\text{loss invariance:} \quad L(a_X(g, \theta), X) = L(\theta, X), \qquad \forall g \in G,\ \forall X \in \mathcal{D}^d,\ \forall \theta \in \Theta.$$

A function L has a *G-symmetry* if there exists a loss-invariant group action a. We refer to G as a symmetry group of L. Additionally, the action a is termed a non-trivial *data-dependent group action* or symmetry if the map (1) has a non-trivial dependency on X. That is, a is data-dependent if there exists $X_1, X_2 \in \mathcal{D}^d$, such that $a_{X_1} \neq a_{X_2}$.

3.2 Infinitesimal Symmetry

A smooth action determines vector fields describing its instantaneous changes to the parameters. We next characterize loss invariance and composition through differential conditions on these fields and the action map. Proofs and examples are given in Appendix A.

The following theorem characterizes loss invariance through the derivative of L in the directions generated by the group's infinitesimal action. We denote the Lie algebra of G by $\mathfrak{g} = T_I G$.

Theorem 3.1 (Infinitesimal characterization of loss invariance)**.** *Let G be a connected Lie group, let Θ be a smooth manifold, and let $a\colon \mathcal{D}^d \to (G \times \Theta \to \Theta)$ be a group action such that each a_X is smooth. Let $L\colon \Theta \times \mathcal{D}^d \to \mathbb{R}^d$ satisfy $L(\cdot, X) \in C^1(\Theta, \mathbb{R}^d)$ for every X. Then a is a parameter space symmetry of L if and only if*

$$D_\theta L\big|_{\theta,X}\big(D_g a_X\big|_{I,\theta}(h)\big) = 0 \tag{2}$$

for every $\theta \in \Theta$, $X \in \mathcal{D}^d$, and $h \in \mathfrak{g}$.

Here $D_\theta L|_{\theta,X}\colon T_\theta\Theta \to \mathbb{R}^d$ and $D_g a_X|_{I,\theta}\colon \mathfrak{g} \to T_\theta\Theta$.

Proof sketch. Consider the curve $\gamma(t) = a_X(\exp(th), \theta)$. Loss invariance implies (2) by differentiation at $t = 0$. Conversely, the action law gives $\dot\gamma(t) = D_g a_X|_{I,\gamma(t)}(h)$. If (2) holds at every parameter point, the chain rule shows that $L(\gamma(t), X)$ is constant. Every element of a connected Lie group is a finite product of exponentials, so the action law extends this invariance to all of G. □

Equation (2) states that the derivative of each component of L vanishes in the directions in parameter space generated by the infinitesimal action $D_g a_X\big|_{I,\theta}(h)$. Thus, moving along these directions does not change the loss to first order, reflecting the invariance of L under the group action.

We similarly characterize the group-action axioms infinitesimally. To obtain the condition for composition, we hold one group element fixed and differentiate with respect to an infinitesimal left multiplication of the other.

We use the following notation in the theorem. For $g \in G$, let $R_g(q) = qg$ denote right multiplication. For a smooth map satisfying $a_X(I, \theta) = \theta$, let $v_{X,h}(\theta) := D_g a_X\big|_{I,\theta}(h)$ denote the infinitesimal action generated by $h \in \mathfrak{g}$. As a function of θ, this is a smooth vector field on Θ.

Theorem 3.2 (Infinitesimal characterization of group actions)**.** *Let G be a connected Lie group, let Θ be a smooth manifold, and let $a\colon \mathcal{D}^d \to (G \times \Theta \to \Theta)$ be a smooth map satisfying $a_X(I, \theta) = \theta$ for every X and θ. Then a_X is a group action for every X if and only if*

$$D_g a_X\big|_{g,\theta}\big((dR_g)_I h\big) = v_{X,h}\big(a_X(g, \theta)\big) \tag{3}$$

for all $g \in G$, $\theta \in \Theta$, $X \in \mathcal{D}^d$, and $h \in \mathfrak{g}$.

Equation (3) shows that the differential of each orbit map $g \mapsto a_X(g, \theta)$ transports the right-invariant vector field generated by h on G to the vector field $v_{X,h}$ on parameter space. For a matrix Lie group, $(dR_g)_I h = hg$. At the identity, associativity implies the Lie-bracket compatibility condition $[v_{X,h_1}, v_{X,h_2}] = -v_{X,[h_1,h_2]}$, where the vector-field bracket uses the convention $[u, w]f = u(wf) - w(uf)$. Proofs and the additional global conditions are given in Appendix A.2.

4 Building New Symmetry from Known Ones

One way to identify symmetries in a large network is by examining its components or subnetworks. Despite often having billions of parameters, neural networks typically consist of a limited set of function families, such as fully connected layers, attention mechanisms, and activation functions. We first establish when symmetries of these components extend to the full network. We then give conditions under which small-subnetwork transformations generate an entire class of output-preserving maps, together with an explicit finite-batch action.

4.1 Lifting subnetwork symmetries

We identify symmetries of a larger network by preserving the outputs of selected subnetworks. A naive choice of parameter subset is not sufficient: preserving a computation on a subset of parameters need not preserve the full network if those parameters influence the surrounding computation in other ways. Proposition 4.1 formalizes when and how we can obtain symmetries of larger networks from those of their subnetworks. If a loss function L depends on a subset of the parameters solely through a subnetwork f, then any symmetry of f will also preserve L:

Proposition 4.1. *Let $L\colon \Theta \times \mathcal{D}^d \to \mathbb{R}^d$ where the parameter space Θ is a product space $\Theta = \Theta_1 \times \Theta_2$. Suppose for some spaces S and T, there exist functions $h\colon \Theta_1 \times \mathcal{D}^d \to S$, $f\colon \Theta_2 \times S \to T$ and $j\colon (\Theta_1 \times T) \times \mathcal{D}^d \to \mathbb{R}^d$, such that for every $\theta = (\theta_1, \theta_2) \in \Theta$ and $X \in \mathcal{D}^d$, $L(\theta, X) = j\big((\theta_1, f\big(\theta_2, h(\theta_1, X)\big)\big), X\big)$. If $a\colon S \to (G \times \Theta_2 \to \Theta_2)$ is a G-symmetry of f, then there is an induced G-symmetry of L, $a'\colon \mathcal{D}^d \to (G \times \Theta \to \Theta)$, defined by $a'_X(g, (\theta_1, \theta_2)) = \big(\theta_1, a_{h(\theta_1,X)}(g, \theta_2)\big)$.*

The commutative diagram below summarizes the factorization, where $p_1\colon \Theta \to \Theta_1$, $p_2\colon \Theta \to \Theta_2$ are projections, $\mathrm{id}_1\colon \Theta_1 \to \Theta_1$ and $\mathrm{id}_2\colon \Theta_2 \to \Theta_2$ are identity maps, and $X \in \mathcal{D}^d$ is a batch of data. The spaces S and T can be interpreted as intermediate feature spaces in the neural network. In this factorization, h does not depend on Θ_2, and j depends on Θ_2 only through the output of f. This effectively confines L's dependency on Θ_2 to the transformation defined by f, ensuring that any transformation on Θ_2 not altering the output of f will not affect the output of L. Consequently, symmetries identified in the smaller network f can be extrapolated to the larger network L.

$$\begin{array}{ccccc} \Theta & & \xrightarrow{\qquad\qquad L(\cdot, X) \qquad\qquad} & & \mathbb{R}^d \\ \Big\downarrow{\scriptstyle p_1 \times p_2 \times p_1} & & & & \Big\uparrow{\scriptstyle j(\cdot, X)} \\ \Theta_1 \times \Theta_2 \times \Theta_1 & \xrightarrow{\mathrm{id}_1 \times \mathrm{id}_2 \times h(\cdot, X)} & \Theta_1 \times \Theta_2 \times S & \xrightarrow{\mathrm{id}_1 \times f(\cdot, \cdot)} & \Theta_1 \times T \end{array}$$

The next two corollaries apply Proposition 4.1 to show that some symmetries are preserved as networks scale up, either through increasing the dimensionality of a layer or through adding additional

layers. We call σ row-wise if it applies a fixed function $\sigma_{row}\colon \mathbb{R}^k \to \mathbb{R}^k$ independently to each row of its input. Element-wise functions are a special case.

Corollary 4.2. *Consider the parameter space $\Theta(m,h,n) = \mathbb{R}^{m\times h} \times \mathbb{R}^{h\times n}$ and data space $\mathcal{D}(n,k) = \mathbb{R}^{n\times k}$. Let σ apply the same row function $\sigma_{row}\colon \mathbb{R}^k \to \mathbb{R}^k$ at every hidden width. Define $L_{mnhk}\colon \Theta(m,h,n)\times\mathcal{D}(n,k)\to\mathbb{R}^{m\times k}$ by $L_{mnhk}((U,V),X) = U\sigma(VX)$. If L_{mnhk} has a G-symmetry, then $L_{mnh'k}$ has a G-symmetry for every $h' > h$.*

Corollary 4.3. *Let $\Theta = \Theta_1 \times \ldots \times \Theta_l$ be a parameter space. Consider a list of spaces $V_0 = \mathcal{D}^d$, $V_l = \mathbb{R}^d$, and $V_1, \ldots, V_{l-1}$. Let $L\colon \Theta\times\mathcal{D}^d \to \mathbb{R}^d$ be a function defined recursively by $\{L_i\}_{i=1}^l$ with $L_i\colon \Theta_i \times V_{i-1} \to V_i$, such that $L = \phi_l$ where $\phi_i = L_i(\theta_i, \phi_{i-1}) \in V_i$ and $\phi_0 = X$. If for some $1 \le i \le l$, L_i has a G-symmetry, then L has a G-symmetry.*

Both corollaries follow by factoring and applying Proposition 4.1. The corresponding h, f, and j are deferred to Appendix B. Figure 1 (a-b) shows the subset of parameters (Θ_2) whose symmetries lift to the larger network. Figure 1 (c) shows why this need not hold for an arbitrary subset of parameters: Θ_2 omits part of the input and output fan for an intermediate neuron, so the full network depends on Θ_2 through information not captured by the subnetwork output.

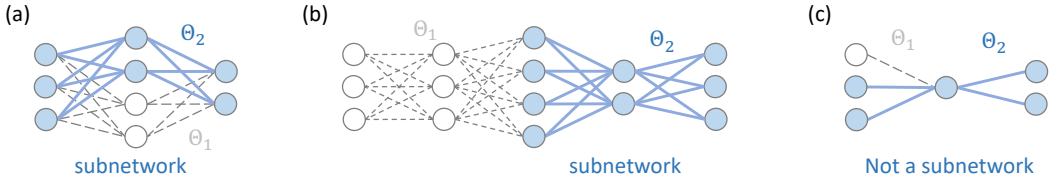


Figure 1: Examples of when symmetries in subnetworks are symmetries in the original network (a-b), and when they are not (c). Blue edges denote the transformed parameters Θ_2; gray dashed edges denote the fixed parameters Θ_1.

In addition to obtaining symmetries from those in smaller networks, we can also get symmetries for a loss function over data batches with a certain size, if we know there is a symmetry for this function over larger data batches. Concretely, an action preserving the pointwise losses on batches of size d induces an action on every smaller positive batch size, by using a fixed padding of the batch.

Proposition 4.4. *Let $L_d\colon \Theta\times\mathcal{D}^d \to \mathbb{R}^d$ be a function that is applied pointwise on each of d data points in a data batch. If L_d admits a G-symmetry, then $L_{d'}$ admits a G-symmetry for all $d' < d$.*

4.2 When small subnetworks are sufficient

The subnetwork perspective also yields a completeness result. Under the following conditions, small-subnetwork transformations generate an entire class of symmetry in the original network. This allows us to apply divide and conquer to symmetry discovery, making the problem computationally tractable.

Fix $X \in \mathbb{R}^{n\times k}$ and consider $F_X(U,V) = U\sigma(VX)$ with $U \in \mathbb{R}^{m\times H}$, $V \in \mathbb{R}^{H\times n}$, $H > k$, and smooth row-wise σ. Choose k hidden units B, let C be their complement, and assume the domain

$$\Omega_B = \{(U,V) : \det[\sigma(VX)]_B \neq 0\} \tag{4}$$

is nonempty. A hidden-unit block comprises one row of V and the matching column of U; subscripts select these blocks.

Theorem 4.5 (Decomposition into subnetwork transformations)**.** *Let $(\phi_t)_{t\in[0,1]}$ be a smooth family of diffeomorphisms of Ω_B satisfying $\phi_0 = \mathrm{id}$ and $F_X \circ \phi_t = F_X$ for every $t \in [0,1]$. Assume that all ϕ_t equal the identity outside the same compact set $K \subset \Omega_B$. Then $\phi_1 = \psi_N \circ \ldots \circ \psi_1$ for finitely many output-preserving diffeomorphisms ψ_i, each changing at most $k+1$ hidden-unit blocks and preserving the combined output of those units on X.*

When $H > k+1$, every factor ψ_i acts on a proper subnetwork. The factors may depend on unchanged parameters.

The output coordinates used in the proof additionally give an explicit family of finite-batch actions. For $P \in \mathbb{R}^{m\times(H-k)}$ and $Q \in \mathbb{R}^{(H-k)\times n}$, consider

$$V_B' = V_B, \qquad V_C' = V_C + Q, \qquad U_C' = U_C + P.$$

The units in B compensate for the output change by setting

$$U_B' = \big[F_X(U,V) - U_C'\sigma(V_C'X)\big]\big[\sigma(V_BX)\big]^{-1}.$$

This is the unique compensating value because $\sigma(V_BX)$ is invertible on Ω_B. Keeping V_B fixed preserves this domain. Changing a single entry of P or Q affects one unit in C and the compensating output weights of the k units in B.

Corollary 4.6 (An explicit data-dependent symmetry)**.** *The maps $a_X^B((P,Q),(U,V)) = (U',V')$ defined above form a smooth free action of the additive group $\mathbb{R}^{(H-k)(m+n)}$ on Ω_B and preserve F_X. This action is generated by commuting one-parameter subgroups, each changing at most $k+1$ hidden-unit blocks and preserving their combined output.*

The corollary is of independent interest. It defines a closed-form data-dependent action on Ω_B for finite batches, extending the scope of the single-input constructions in past work without requiring the activation to preserve full rank (Zhao et al., 2023). The subgraph view reveals the mechanism that enables this family of symmetry: one unit changes, k selected units compensate, and their combined output stays fixed. Proposition 4.1 then lifts each such symmetry to the full network on the corresponding invariant domain. Proofs and an illustrative example can be found in Appendix C.

5 Automatic Discovery of Parameter Space Symmetry

Using the infinitesimal characterization derived in Section 3.2, we construct an automated framework for discovering parameter space symmetries. We describe joint learning of matrix generators and a nonlinear action map, define the loss terms and regularizers, and show how infinitesimal errors control loss invariance and composition errors.

5.1 Enforcing Loss Invariance and Group Axioms

Given a function L, our goal is to learn a group and its action on $\mathbb{R}^p$. We learn r generators $h_1,\dots,h_r \in \mathbb{R}^{s\times s}$ subject to $\|h_i\|_F = 1$. These matrices define the group

$$G = \{\exp(t_m h_{i_m})\cdots\exp(t_1 h_{i_1}) : m \geq 0,\ t_j \in \mathbb{R},\ i_j \in \{1,\dots,r\}\}. \tag{5}$$

Group elements compose by matrix multiplication, and inverses reverse the factors and negate their coefficients. The matrix size s and number of generators r specify the search class.

We parameterize a candidate action using a smooth neural network b_X with a matrix input, enforcing identity exactly,

$$a_X(g,\theta) = \theta + \big(b_X(g,\theta) - b_X(I,\theta)\big). \tag{6}$$

Thus $a_X(I,\theta) = \theta$ throughout training. Write $v_{X,h}(\theta) = D_g a_X|_{I,\theta}(h)$. We sample parameter–batch pairs (θ, X), products g with varying lengths, orders, and signed coefficients, and test directions h uniformly from the learned list. If the group G is supplied, we fix h_i to a Frobenius-orthonormal basis of $\mathfrak{g}$.

We define the following infinitesimal loss terms:

$$\mathcal{L}_{\text{invariance}} = \mathbb{E}_{X,\theta,h}\big\|D_\theta L\big|_{\theta,X}[v_{X,h}(\theta)]\big\|_2^2, \tag{7}$$

$$\mathcal{L}_{\text{assoc}} = \mathbb{E}_{X,\theta,g,h}\big\|D_g a_X\big|_{g,\theta}(hg) - v_{X,h}(a_X(g,\theta))\big\|_2^2. \tag{8}$$

These terms encourage loss invariance and the composition condition in Theorem 3.2. Proposition A.1 shows that testing the learned generating directions suffices when the residuals vanish everywhere. To control finite transformations directly, we also use

$$\mathcal{L}_{\text{comp}} = \mathbb{E}_{X,\theta,g_1,g_2}\big\|a_X(g_2,a_X(g_1,\theta)) - a_X(g_2g_1,\theta)\big\|_2^2. \tag{9}$$

The batch X stays fixed within every composition. We evaluate the residuals at both original and transformed parameter points. Derivatives are computed by automatic differentiation, with sampling details in Appendix D.1.

5.2 Regularization

To prevent the learned group action from becoming trivial, we encourage infinitesimal parameter changes with mean squared norm close to β^2, where $\beta > 0$. When learning multiple generators, we also encourage distinct parameter changes (Yang et al., 2023; Ko et al., 2024). We apply these regularizers to $v_i = v_{X,h_i}(\theta)$,

$$\mathcal{L}_{\text{scale}} = \frac{1}{r}\sum_{i=1}^{r}\left(\frac{\mathbb{E}_{X,\theta}\|v_i\|_2^2}{\beta^2} - 1\right)^2, \tag{10}$$

$$\mathcal{L}_{\text{div}} = \frac{2}{r(r-1)\beta^4}\sum_{1\le i<j\le r}\big(\mathbb{E}_{X,\theta}\langle v_i, v_j\rangle\big)^2. \tag{11}$$

We set $\mathcal{L}_{\text{div}} = 0$ when $r = 1$. These terms control the average size of the induced fields and penalize correlations between them. The expectations use the original parameter and batch samples.

The base training objective, before fixed normalization, is

$$\min_{b,h_1,\dots,h_r} \quad \gamma_1\mathcal{L}_{\text{invariance}} + \gamma_2\mathcal{L}_{\text{comp}} + \gamma_3\mathcal{L}_{\text{assoc}} + \gamma_4\mathcal{L}_{\text{scale}} + \gamma_5\mathcal{L}_{\text{div}}, \tag{12}$$

where $\gamma_1, \gamma_3, \gamma_4, \gamma_5 > 0$ and $\gamma_2 \ge 0$.

We call this the hybrid objective when $\gamma_2 > 0$, since it combines the infinitesimal conditions with a direct test of finite composition. Setting $\gamma_2 = 0$ gives the infinitesimal-only objective. To assess the value of the infinitesimal conditions, we also train a finite-constraint baseline that checks output preservation and composition directly, with the same regularizers. Appendix D.1 gives the exact variants and training normalization.

5.3 Approximate Symmetry

We prove that small identity and infinitesimal errors give small loss-preservation and composition errors over bounded group paths. These bounds support using the infinitesimal characterizations as discovery objectives.

On a data batch X, let $a_X\colon G \times \mathbb{R}^p \to \mathbb{R}^p$ be a smooth map on a connected Lie subgroup $G \subseteq \mathrm{GL}(s,\mathbb{R})$, with $L(\cdot, X) \in C^1$. Suppose $\|a_X(I,\theta) - \theta\| \le \delta$ and the unsquared residuals in $\mathcal{L}_{\text{invariance}}$ and $\mathcal{L}_{\text{assoc}}$ are bounded by $\varepsilon_{\text{loss}}, \varepsilon_{\text{act}} \ge 0$, uniformly over $\theta \in \mathbb{R}^p$, $g \in G$, and unit $h \in \mathfrak{g}$. Equation (6) gives $\delta = 0$. Assume also a bounded loss derivative and Lipschitz generators, with $M, K \ge 0$ such that, for all $\theta, \vartheta \in \mathbb{R}^p$ and $h \in \mathfrak{g}$,

$$\|D_\theta L|_{\theta,X}\|_{\text{op}} \le M, \quad \text{and} \quad \|v_{X,h}(\theta) - v_{X,h}(\vartheta)\| \le K\|h\|\,\|\theta - \vartheta\|.$$

For a piecewise C^1 path $\gamma\colon [0,1] \to G$ with finitely many pieces, define its length by

$$\ell = \int_0^1 \|\dot\gamma(t)\gamma(t)^{-1}\|\,dt, \qquad F_K(\ell) = \int_0^\ell e^{Kt}\,dt.$$

The length measures the accumulated group motion. For example, $\gamma(t) = \exp(th)$ has length $\|h\|$.

Proposition 5.1 (Finite symmetry errors)**.** *Under the preceding assumptions, for every such path γ from I to g_2 and every $g_1 \in G$, $\theta \in \mathbb{R}^p$,*

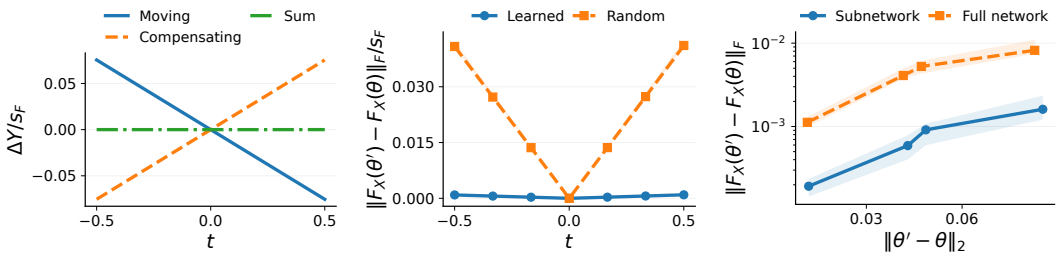
$$\|a_X(g_2, a_X(g_1,\theta)) - a_X(g_2g_1,\theta)\| \le \delta e^{K\ell} + 2\varepsilon_{\text{act}}F_K(\ell), \tag{13}$$

$$\|L(a_X(g_2,\theta), X) - L(\theta, X)\| \le M\delta + (\varepsilon_{\text{loss}} + M\varepsilon_{\text{act}})\ell. \tag{14}$$

For the learned group, the same inequalities hold for a product g_2 from (5) with $\ell = \sum_{j=1}^m |t_j|$, using the corresponding residual and Lipschitz bounds on the normalized generators h_i. Appendix A.4 gives the proofs and versions using bounds along individual paths.

6 Experiments

We use this framework to find parameter symmetries in several common architectures. We first examine known data-independent symmetries and recover the structure of higher-dimensional symmetry groups. We then study nonlinear data-dependent actions and transformations on non-contiguous layers. Finally, we use subnetworks to study larger models, including pretrained Pythia transformers.



(a) Nonlinear compensation (b) Output along tanh curves (c) Subnetwork discovery

Figure 2: Learned compensation, finite preservation, and subnetwork discovery. (a) Opposing changes $\Delta Y_j = Y_j' - Y_j$ in unit contributions $Y_j = u_j\sigma(v_jX)$ nearly cancel. s_F is the fixed root-mean-square output norm. (b) Tanh output drift is smaller compared to random translations. (c) Subnetwork discovery gives smaller full-network output error at similar displacement.

The direct-map experiments jointly learn one generator and a nonlinear action, preserving complete network or subnetwork outputs $F_X(\theta)$. Independent evaluations retain all five small-network seeds and all three Pythia seeds. Numerical summaries are medians across seeds of within-seed 95th-percentile errors; motion uses medians. Appendix D describes the sampling and additional results.

6.1 Learned data-independent symmetries

We first validate that our method can learn generators for known data-independent symmetries in neural networks. We consider a two-layer linear network $F(U, V) = UV$, with $U, V \in \mathbb{R}^{2\times2}$, and a bias-free ReLU network with widths $(2, 4, 2)$. Their reference symmetries are hidden-basis changes and positive unit rescalings. The learned maps approximately preserve outputs and compose consistently while moving the parameters. Their fields follow the reference directions, indicating successful recovery of one-parameter actions (Appendix F).

An affine-field solver, following Moskalev et al. (2022), additionally recovers symmetry algebras of up to eight dimensions without receiving the reference generators or their dimension. It recovers noncommuting transformations in linear networks, for which reversing the order of two transformations can change the final parameters. The ReLU rescalings commute. This separate specialization and exact verification of its reconstructed generators are described in Appendix F.

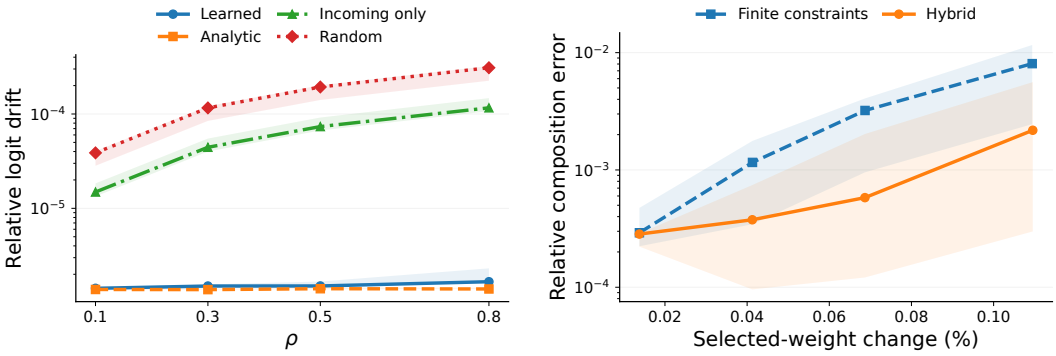
6.2 Learned data-dependent symmetries

We next study data-dependent transformations in a two-layer sigmoid network, $F_X(U, V) = U\sigma(VX)$. We fix the incoming weights of k compensating units and the outgoing weights of one moving unit. The action changes the moving unit's incoming weights and the compensators' output weights. The analytic reference (Corollary 4.6) is withheld from training.

Our method learns to preserve outputs through nonlinear compensation. In the scalar, single-input case, the transformed parameter block changes by a median of 12%. The two units make opposing output changes (Figure 2(a)), with a very small cancellation error. The same transformed weights produce much larger changes on fresh inputs, showing that preservation is specific to the conditioning data. Appendix E gives the comparison and two-input results.

The method also learns coordinated transformations on non-contiguous layers. In a three-layer tanh network with widths $(4, 16, 16, 2)$ trained on synthetic regression, we change the first and last weight matrices while fixing the middle layer. On 24 fixed inputs, output drift along generated curves $\theta(t) = a_X(\exp(th), \theta_0)$ is much smaller than for random translations calibrated to similar parameter motion (Figure 2(b)). Here θ_0, X, and h stay fixed along each curve.

The infinitesimal conditions improve the consistency of finite transformations. We divide the discrepancy between successive transformations and their group product by the summed displacements of the two steps. In scalar sigmoid, the hybrid objective reduces this composition error from 0.306% under finite constraints to 0.037%, at similar motion. Both objectives share the architecture, regularizers, and final step budget, with separate validation-based tuning.



(a) Preserving protected outputs

(b) Composing learned transformations

Figure 3: Learned symmetry in Pythia-1B, layer 3, at the original checkpoint. (a) Relative drift $\|Z' - Z\|_F / \|Z\|_F$ of original and changed logits on one fixed 16-token prefix. ρ bounds $|t|$ in $\exp(th)$. Learned uses the hybrid objective; analytic solves for compensation; incoming-only omits it. Analytic and random references are motion-calibrated. (b) Composition error versus $100\|\theta'_S - \theta_S\|_2 / \|\theta_S\|_2$ on edited weights S.

6.3 Symmetry in subnetworks and pretrained transformers

Subnetwork discovery preserves full-network outputs more accurately than discovery over all parameters in our paired comparison. We learn an action on 10 weights incident to one hidden unit in a ReLU network with widths $(2, 8, 8, 1)$, preserving its full boundary output as required by Proposition 4.1. At similar parameter displacement, it gives 5.8 times smaller full-network output error than discovery over all 88 parameters (Figure 2(c)). Both maps use identical evaluation parameters and inputs. This compares the smaller support and boundary-output target together.

We also learn symmetries in pretrained Pythia-160M and Pythia-1B feedforward blocks. For a fixed k-token prefix, the action changes one unit's incoming weights and bias and k compensators' outgoing columns, keeping all other parameters fixed. We train on their combined boundary output, then install the edits in float32 and evaluate the complete model at the original checkpoint.

Learned compensation sharply reduces protected-output drift in Pythia-1B (Figure 3(a)). At coefficient radius $\rho = 0.5$, the selected weights change by a median of 0.069%. Protected-logit drift is approximately 129 times smaller than for a random translation at similar motion. Keeping the learned incoming update while removing compensation increases drift by 49 times, despite reducing the total displacement.

Infinitesimal training also improves composition after the weights are installed. At nearly identical motion, the hybrid objective reduces stored-weight composition error from 0.321% to 0.058% of the two-step displacement (Figure 3(b)). Each update is rounded to float32 before the next action. Pythia-160M shows the same benefits of learned compensation and improved composition (Appendix H).

7 Discussion

In this work, we introduce a framework for formalizing and discovering parameter space symmetries in neural networks. Infinitesimal characterizations and finite-error bounds provide objectives for learning nonlinear actions. Subnetwork lifting and decomposition connect this differential structure to the modular structure of neural networks. Mathematically, the existence and number of symmetries in neural network parameter spaces remain open questions. Whether the number of symmetries is affected by existence of symmetry in data or changes during training are also interesting directions. The discovered symmetries could be useful in downstream applications such as weight space learning, where architectures respecting known parameter symmetries already process neural network weights effectively (Zhang et al., 2023; Lim et al., 2024; Kalogeropoulos et al., 2024; Tran et al., 2024). Leveraging the learned symmetries via equivariant architectures (Finzi et al., 2021) or frame averaging (Puny et al., 2021) can potentially improve current approaches.

AI USE STATEMENT

In this work, we used generative AI tools for checking proof correctness, implementing experiment code, and reviewing drafts of the manuscript. We have reviewed all AI-assisted work. We take responsibility for the final content of this work.

REFERENCES

Marco Armenta, Thierry Judge, Nathan Painchaud, Youssef Skandarani, Carl Lemaire, Gabriel Gibeau Sanchez, Philippe Spino, and Pierre-Marc Jodoin. Neural teleportation. *Mathematics*, 11(2):480, 2023.

Vijay Badrinarayanan, Bamdev Mishra, and Roberto Cipolla. Symmetry-invariant optimization in deep networks. *arXiv preprint arXiv:1511.01754*, 2015.

Gregory Benton, Marc Finzi, Pavel Izmailov, and Andrew G Wilson. Learning invariances in neural networks from training data. *Advances in neural information processing systems*, 33:17605–17616, 2020.

An Mei Chen, Haw-minn Lu, and Robert Hecht-Nielsen. On the geometry of feedforward neural network error surfaces. *Neural computation*, 5(6):910–927, 1993.

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.

Simon S Du, Wei Hu, and Jason D Lee. Algorithmic regularization in learning deep homogeneous models: Layers are automatically balanced. *Neural Information Processing Systems*, 2018.

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*, pp. 3318–3328. PMLR, 2021.

Alex Gabel, Victoria Klein, Riccardo Valperga, Jeroen SW Lamb, Kevin Webster, Rick Quax, and Efstratios Gavves. Learning lie group symmetry transformations with neural networks. In *Topological, Algebraic and Geometric Learning Workshops 2023*, pp. 50–59. PMLR, 2023.

Iordan Ganev, Twan van Laarhoven, and Robin Walters. Universal approximation and model compression for radial neural networks. *arXiv preprint arXiv:2107.02550v2*, 2022.

Charles Godfrey, Davis Brown, Tegan Emerson, and Henry Kvinge. On the symmetries of deep learning models and their internal representations. *Advances in Neural Information Processing Systems*, 2022.

Elisenda Grigsby, Kathryn Lindsey, and David Rolnick. Hidden symmetries of relu networks. In *International Conference on Machine Learning*, pp. 11734–11760. PMLR, 2023.

Nate Gruver, Marc Anton Finzi, Micah Goldblum, and Andrew Gordon Wilson. The lie derivative for measuring learned equivariance. In *The Eleventh International Conference on Learning Representations*, 2022.

Stefan Haller and Josef Teichmann. Smooth perfectness through decomposition of diffeomorphisms into fiber preserving ones. *Annals of Global Analysis and Geometry*, 23(1):53–63, 2003.

Lexiang Hu, Yikang Li, and Zhouchen Lin. Explicit discovery of nonlinear symmetries from dynamic data. *arXiv preprint arXiv:2510.01855*, 2025.

Ioannis Kalogeropoulos, Giorgos Bouritsas, and Yannis Panagakis. Scale equivariant graph metanetworks. *Advances in neural information processing systems*, 37:106800–106840, 2024.

Pavan Karjol, Rohan Kashyap, Aditya Gopalan, and A. P. Prathosh. A unified framework for discovering discrete symmetries. In *Proceedings of The 27th International Conference on Artificial Intelligence and Statistics*, volume 238 of *Proceedings of Machine Learning Research*, pp. 793–801. PMLR, 2024.

Gyeonghoon Ko, Hyunsu Kim, and Juho Lee. Learning infinitesimal generators of continuous symmetries from data. *Advances in Neural Information Processing Systems*, 37:85973–86003, 2024.

Sven Krippendorf and Marc Syvaeri. Detecting symmetries with neural networks. *Machine Learning: Science and Technology*, 2(1):015010, 2020.

Daniel Kunin, Javier Sagastuy-Brena, Surya Ganguli, Daniel LK Yamins, and Hidenori Tanaka. Neural mechanics: Symmetry and broken conservation laws in deep learning dynamics. In *International Conference on Learning Representations*, 2021.

Derek Lim, Haggai Maron, Marc T. Law, Jonathan Lorraine, and James Lucas. Graph metanetworks for processing diverse neural architectures. In *The Twelfth International Conference on Learning Representations*, 2024.

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.

Artem Moskalev, Anna Sepliarskaia, Erik J Bekkers, and Arnold WM Smeulders. On genuine invariance learning without weight-tying. In *Topological, Algebraic and Geometric Learning Workshops 2023*, pp. 218–227. PMLR, 2023.

Aviv Navon, Aviv Shamsian, Idan Achituve, Ethan Fetaya, Gal Chechik, and Haggai Maron. Equivariant architectures for learning in deep weight spaces. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202, pp. 25790–25816. PMLR, 2023.

Quynh Nguyen. On connected sublevel sets in deep learning. In *International conference on machine learning*, pp. 4790–4799. PMLR, 2019.

Richard S Palais. A global formulation of the lie theory of transformtion groups. *Mem. Amer. Math. Soc.*, 22, 1957.

Henning Petzka, Martin Trimmel, and Cristian Sminchisescu. Notes on the symmetries of 2-layer relu-networks. In *Proceedings of the Northern Lights Deep Learning Workshop*, volume 1, pp. 6–6, 2020.

Vasco Portilheiro. Quantifying lie group learning with local symmetry error. In *NeurIPS 2023 Workshop on Symmetry and Geometry in Neural Representations*, 2023.

Omri Puny, Matan Atzmon, Heli Ben-Hamu, Ishan Misra, Aditya Grover, Edward J Smith, and Yaron Lipman. Frame averaging for invariant and equivariant network design. *arXiv preprint arXiv:2110.03336*, 2021.

David W Romero and Suhas Lohit. Learning partial equivariances from data. *Advances in Neural Information Processing Systems*, 35:36466–36478, 2022.

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

Ben Shaw, Sasidhar Kunapuli, Abram Magner, and Kevin R Moon. Continuous symmetry discovery and enforcement using infinitesimal generators of multi-parameter group actions. *arXiv preprint arXiv:2505.08219*, 2025.

Noam Shazeer. Glu variants improve transformer. *arXiv preprint arXiv:2002.05202*, 2020.

Sho Sonoda, Hideyuki Ishi, Isao Ishikawa, and Masahiro Ikeda. Joint group invariant functions on data-parameter domain induce universal neural networks. In *NeurIPS 2023 Workshop on Symmetry and Geometry in Neural Representations*, 2023.

Gustav Sourek, Filip Zelezny, and Ondrej Kuzelka. Lossless compression of structured convolutional models via lifting. In *International Conference on Learning Representations*, 2021.

Viet-Hoang Tran, Thieu Vo, Tho Tran-Huu, An T Nguyen, and Tan Nguyen. Monomial matrix group equivariant neural functional networks. *Advances in Neural Information Processing Systems*, 37: 48628–48665, 2024.

Gijs M Tuynman. Integrating infinitesimal (super) actions. *arXiv preprint arXiv:1302.2823*, 2013.

Alonso Urbano and David W Romero. Self-supervised detection of perfect and partial input-dependent symmetries. *arXiv preprint arXiv:2312.12223*, 2023.

Jonas Gregor Wiese, Lisa Wimmer, Theodore Papamarkou, Bernd Bischl, Stephan Günnemann, and David Rügamer. Towards efficient mcmc sampling in bayesian neural networks by exploiting symmetry. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases (ECML PKDD): Research Track*, pp. 459–474, 2023.

Jianke Yang, Robin Walters, Nima Dehmamy, and Rose Yu. Generative adversarial symmetry discovery. *International Conference on Machine Learning*, 2023.

Jianke Yang, Nima Dehmamy, Robin Walters, and Rose Yu. Latent space symmetry discovery. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235, pp. 56047–56070. PMLR, 2024.

Binchi Zhang, Zaiyi Zheng, Zhengzhang Chen, and Jundong Li. Beyond the permutation symmetry of transformers: The role of rotation for model fusion. *International Conference on Machine Learning*, 2025.

David W Zhang, Miltiadis Kofinas, Yan Zhang, Yunlu Chen, Gertjan J Burghouts, and Cees GM Snoek. Neural networks are graphs! graph neural networks for equivariant processing of neural networks. *The 2nd Annual Workshop on Topology, Algebra, and Geometry in Machine Learning (TAG-ML) at the 40th International Conference on Machine Learning*, 2023.

Yani Zhang and Helmut Bölcskei. Complete identification of deep relu neural networks by many-valued logic. *arXiv preprint arXiv:2602.00266*, 2026.

Bo Zhao, Nima Dehmamy, Robin Walters, and Rose Yu. Symmetry teleportation for accelerated optimization. *Advances in Neural Information Processing Systems*, 2022.

Bo Zhao, Iordan Ganev, Robin Walters, Rose Yu, and Nima Dehmamy. Symmetries, flat minima, and the conserved quantities of gradient flow. *International Conference on Learning Representations*, 2023.

Bo Zhao, Nima Dehmamy, Robin Walters, and Rose Yu. Understanding mode connectivity via parameter space symmetry. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267, pp. 77441–77460. PMLR, 2025.

Bo Zhao, Robin Walters, and Rose Yu. Symmetry in neural network parameter spaces. *Transactions on Machine Learning Research*, 2026. ISSN 2835-8856.

Allan Zhou, Tom Knowles, and Chelsea Finn. Meta-learning symmetries by reparameterization. *International Conference on Learning Representations*, 2021.

Allan Zhou, Kaien Yang, Kaylee Burns, Adriano Cardace, Yiding Jiang, Samuel Sokota, J Zico Kolter, and Chelsea Finn. Permutation equivariant neural functionals. *Advances in Neural Information Processing Systems*, 36, 2023.

Liu Ziyin. Symmetry leads to structured constraint of learning. In *International Conference on Machine Learning*. PMLR, 2024.

APPENDIX

A INFINITESIMAL CHARACTERIZATIONS OF PARAMETER SYMMETRY

This section proves the infinitesimal characterizations in Section 3.2, recalls the integration result used there, and proves the finite-error guarantees following the discovery objectives in Section 5.3. Examples are given in Appendix A.3.

A.1 INFINITESIMAL FORMULATION FOR LOSS INVARIANCE

Theorem 3.1. *Let G be a connected Lie group, let Θ be a smooth manifold, and let $a\colon \mathcal{D}^d \to (G \times \Theta \to \Theta)$ be a group action such that each a_X is smooth. Let $L\colon \Theta \times \mathcal{D}^d \to \mathbb{R}^d$ satisfy $L(\cdot, X) \in C^1(\Theta, \mathbb{R}^d)$ for every X. Then a is a parameter space symmetry of L if and only if*

$$D_\theta L\big|_{\theta,X}\big(D_g a_X\big|_{I,\theta}(h)\big) = 0$$

for every $\theta \in \Theta$, $X \in \mathcal{D}^d$, and $h \in \mathfrak{g}$.

Proof. Fix X and $h \in \mathfrak{g}$, and consider the smooth curve $\gamma(t) = a_X(\exp(th), \theta)$. If a is a symmetry of L, then

$$L(\gamma(t), X) = L(\theta, X) \qquad \text{for all } t \in \mathbb{R}.$$

Differentiating at $t = 0$, using $\gamma(0) = \theta$ and $\frac{d}{dt}\big|_0 \exp(th) = h$, gives

$$0 = D_\theta L\big|_{\theta,X}\left[\left.\frac{d\gamma(t)}{dt}\right|_0\right] = D_\theta L\big|_{\theta,X}\big[D_g a_X\big|_{I,\theta}(h)\big].$$

Conversely, suppose (2) holds at every parameter point. The action law gives

$$\gamma(t+s) = a_X(\exp(sh), \gamma(t)).$$

Differentiating with respect to s at 0 yields $\dot\gamma(t) = D_g a_X|_{I,\gamma(t)}(h)$. The chain rule therefore gives

$$\frac{d}{dt} L(\gamma(t), X) = D_\theta L\big|_{\gamma(t),X}\big[D_g a_X\big|_{I,\gamma(t)}(h)\big] = 0.$$

Consequently every exponential acts loss-invariantly. The subgroup generated by the image of the exponential map contains an identity neighborhood and is therefore open. Its other cosets are open as well, so the subgroup is also closed. Since G is connected, this subgroup is all of G. Every $g \in G$ is thus a finite product of exponentials. Repeated use of the action law gives $L(a_X(g,\theta), X) = L(\theta, X)$ for every g. The argument holds for every X, proving the converse. □

A.2 INFINITESIMAL FORMULATION FOR THE GROUP-ACTION AXIOMS

Throughout this subsection, Lie groups and manifolds are finite-dimensional, and manifolds are without boundary. Smoothness of a means smoothness of a_X in (g, θ) for each fixed data batch X.

Theorem 3.2. *Let G be a connected Lie group, let Θ be a smooth manifold, and let $a\colon \mathcal{D}^d \to (G \times \Theta \to \Theta)$ be a smooth map satisfying $a_X(I, \theta) = \theta$ for every X and θ. Define $v_{X,h}(\theta) = D_g a_X|_{I,\theta}(h)$. Then a_X is a group action for every X if and only if*

$$D_g a_X\big|_{g,\theta}\big((dR_g)_I h\big) = v_{X,h}\big(a_X(g,\theta)\big)$$

for all g, θ, X, h.

Proof. Fix X and suppress it from the notation. First suppose that a is a group action. Associativity gives

$$a(\exp(th)g, \theta) = a(\exp(th), a(g,\theta)).$$

Differentiating at $t = 0$, and using $\frac{d}{dt}\big|_0 \exp(th)g = (dR_g)_I h$, gives

$$D_g a\big|_{g,\theta}\big((dR_g)_I h\big) = D_g a\big|_{I,a(g,\theta)}(h) = v_h(a(g,\theta)).$$

Conversely, suppose that identity and the differential condition hold. For fixed h, g, and θ, define

$$u(t) = a(\exp(th)g, \theta), \qquad w(t) = a(\exp(th), a(g, \theta)).$$

Both curves are defined for every $t \in \mathbb{R}$. Since

$$\frac{d}{dt} \exp(th)g = (dR_{\exp(th)g})_I h,$$

the differential condition implies

$$\dot{u}(t) = v_h(u(t)), \qquad \dot{w}(t) = v_h(w(t)).$$

The identity axiom gives $u(0) = w(0) = a(g, \theta)$. Uniqueness of integral curves of a smooth vector field therefore yields

$$a(\exp(th)g, \theta) = a(\exp(th), a(g, \theta)) \qquad \text{for all } t \in \mathbb{R}. \tag{15}$$

No completeness assumption is needed in this step: the two integral curves are already globally defined by the given map a.

As shown in Appendix A.1, every element of a connected Lie group is a finite product of exponentials. Write $g_2 = \exp(h_m) \cdots \exp(h_1)$. Repeated application of (15) shows that both $a(g_2 g_1, \theta)$ and $a(g_2, a(g_1, \theta))$ equal the successive application of $a(\exp(h_1), \cdot), \ldots, a(\exp(h_m), \cdot)$ to $a(g_1, \theta)$. This proves associativity.

Associativity and identity imply

$$a(g^{-1}, a(g, \theta)) = \theta, \qquad a(g, a(g^{-1}, \theta)) = \theta.$$

Thus $a(g, \cdot)$ and $a(g^{-1}, \cdot)$ are mutually inverse smooth maps. Finally, both sides of the differential condition are linear in h, so checking a basis suffices. □

Every $v_{X,h}$ is complete, with flow $t \mapsto a_X(\exp(th), \theta)$. Here completeness follows from the globally defined map.

Testing generating directions. The group in Section 5.1 is determined by finite products of learned matrix exponentials. The following consequence of the preceding proof shows why testing their generating directions is sufficient.

Let G be a Lie group generated by $\{\exp(th_i) : t \in \mathbb{R},\ 1 \le i \le r\}$, with $h_i \in \mathfrak{g}$. For each fixed X, let $a_X \colon G \times \Theta \to \Theta$ be smooth with $a_X(I, \theta) = \theta$, set $v_{X,h}(\theta) = D_g a_X|_{I,\theta}(h)$, and let $L(\cdot, X)$ be C^1.

Proposition A.1 (Symmetry from generating directions)**.** *In this setup, a_X is a loss-preserving group action for every X if and only if, for every g, θ, X and $i \in \{1, \ldots, r\}$,*

$$\begin{aligned} D_g a_X\big|_{g,\theta}\big((dR_g)_I h_i\big) &= v_{X,h_i}(a_X(g, \theta)), \\ D_\theta L\big|_{\theta,X}[v_{X,h_i}(\theta)] &= 0. \end{aligned} \tag{16}$$

The selected directions may generate additional Lie algebra directions through commutators. The proposition justifies testing the selected list while the group argument ranges over all its finite products.

Proof. Fix X and suppress it from the notation. For a loss-preserving action, differentiating $a(\exp(th_i)g, \theta) = a(\exp(th_i), a(g, \theta))$ and $L(a(\exp(th_i), \theta), X) = L(\theta, X)$ at $t = 0$ gives the two conditions.

Conversely, suppose they hold. For fixed i, g, θ, define

$$u(t) = a(\exp(th_i)g, \theta), \qquad w(t) = a(\exp(th_i), a(g, \theta)).$$

The first condition gives $\dot{u} = v_{h_i}(u)$ and $\dot{w} = v_{h_i}(w)$ for all $t \in \mathbb{R}$. Identity gives $u(0) = w(0) = a(g, \theta)$, so uniqueness of integral curves implies

$$a(\exp(th_i)g, \theta) = a(\exp(th_i), a(g, \theta)).$$

For $g_2 = \exp(t_m h_{i_m}) \cdots \exp(t_1 h_{i_1})$, repeated application of this equality expresses both $a(g_2 g_1, \theta)$ and $a(g_2, a(g_1, \theta))$ as the same successive transformations of $a(g_1, \theta)$. Thus a satisfies associativity. Together with identity, this gives a group action whose inverse maps are $a(g^{-1}, \cdot)$.

The second condition and the chain rule give

$$\frac{d}{dt} L(a(\exp(th_i), \theta), X) = D_\theta L\big|_{a(\exp(th_i),\theta),X}[v_{h_i}(a(\exp(th_i), \theta))] = 0.$$

Each generating one-parameter action preserves L, and finite composition proves invariance on G. □

Lie-bracket compatibility. For smooth vector fields we use the convention $[u, w]f = u(wf) - w(uf)$. For a smooth left action, the generators satisfy

$$[v_{X,h_1}, v_{X,h_2}] = -v_{X,[h_1,h_2]}, \tag{17}$$

$$D_\theta v_{X,h_1}[v_{X,h_2}] - D_\theta v_{X,h_2}[v_{X,h_1}] = v_{X,[h_1,h_2]}, \tag{18}$$

where the second identity is in local coordinates. To see this, fix X and suppress it from the notation. Differentiate $a(g, a(\exp(th), \theta)) = a(\exp(t \operatorname{Ad}_g h), a(g, \theta))$ at $t = 0$ to obtain

$$D_\theta a\big|_{g,\theta}[v_h(\theta)] = v_{\operatorname{Ad}_g h}(a(g, \theta)). \tag{19}$$

Next take $g = \exp(th_1)$ and $h = h_2$ and differentiate at $t = 0$. Using $\frac{d}{dt}\big|_0 \operatorname{Ad}_{\exp(th_1)} h_2 = [h_1, h_2]$ gives $D_\theta v_{h_1}[v_{h_2}] = v_{[h_1,h_2]} + D_\theta v_{h_2}[v_{h_1}]$, which proves both identities. These relations describe how the learned matrix algebra acts on parameter space.

Constructing an action from infinitesimal generators. For a simply connected Lie group G, the Lie–Palais theorem integrates a linear family $h \mapsto v_{X,h}$ of smooth vector fields into a unique smooth left action if every $v_{X,h}$ is complete and (17) holds (Palais, 1957; Tuynman, 2013). Completeness means that each integral curve exists for every real time. By Theorem 3.1, the resulting action preserves a C^1 loss exactly when $D_\theta L|_{\theta,X}[v_{X,h}(\theta)] = 0$ for every θ, X, h.

An integrated action need not equal a prescribed map. For $G = (\mathbb{R}, +)$ and $\Theta = \mathbb{R}$, the map $a(t, \theta) = \theta + t + t^2$ satisfies identity and has complete, compatible generators $v_h(\theta) = h$, but $a(s + t, \theta) - a(s, a(t, \theta)) = 2st$. Integrating its generators instead gives $A(t, \theta) = \theta + t$. Thus a directly parameterized map still requires (3).

A.3 Examples

Below are examples of familiar parameter space symmetry and their infinitesimal form.

A.3.1 Linear action of matrix groups

For a linear action, the infinitesimal parameter change is a matrix–vector product. Let $G \subseteq \mathrm{GL}(n)$ be a connected matrix Lie group acting on $\Theta = \mathbb{R}^n$ by $a_x(g, \theta) = g\theta$. Then $v_{x,h}(\theta) = h\theta$, and Theorem 3.1 characterizes invariance of a C^1 scalar loss by

$$D_\theta L\big|_{\theta,x}[h\theta] = \sum_{i,j=1}^{n} \frac{\partial L}{\partial \theta_i}(\theta, x)\, h_{ij} \theta_j = 0 \tag{20}$$

for every θ, x, h. For vector-valued losses, the condition holds componentwise.

This has the same algebraic expression as the infinitesimal criterion in Theorem 3.1 of Moskalev et al. (2022), with transformations acting on parameters rather than data. Their theorem characterizes preservation of a regular zero set, whereas ours characterizes preservation of the value of L throughout parameter space.

A.3.2 HOMOGENEOUS TWO-LAYER NEURAL NETWORK

Rescaling one weight can be compensated by an inverse rescaling of the next. Consider $L((w_1, w_2), x) = w_2\sigma(w_1 x)$, where $\sigma(\alpha z) = \alpha^c \sigma(z)$ for every $\alpha > 0$ and $z \in \mathbb{R}$, with $c > 0$. The group $(\mathbb{R}_{>0}, \times)$ acts by

$$a_x(\alpha, (w_1, w_2)) = (\alpha w_1, \alpha^{-c} w_2).$$

Multiplication of the scaling factors gives the action law, and homogeneity gives $L(a_x(\alpha, (w_1, w_2)), x) = \alpha^{-c} w_2 \alpha^c \sigma(w_1 x) = L((w_1, w_2), x)$. Differentiating at $\alpha = e^t$, $t = 0$, gives

$$v_{x,1}(w_1, w_2) = (w_1, -c w_2), \qquad w_1 \frac{\partial L}{\partial w_1} - c w_2 \frac{\partial L}{\partial w_2} = 0.$$

Finite invariance uses only homogeneity. The derivative identity holds wherever $L(\cdot, x)$ is differentiable, and the global characterization applies when its stated C^1 assumption holds.

A.4 STABILITY OF FINITE TRANSFORMATIONS

We prove and refine the finite-error guarantees in Section 5.3. The candidate map may have nonzero identity and composition errors. The following estimates track the differential residuals along the paths used to compare finite transformations.

Fix X, let G be a Lie group, and let $a_X : G \times \mathbb{R}^p \to \mathbb{R}^p$ be smooth. Using the canonical identification of Euclidean tangent spaces, set $v_{X,h}(\theta) = D_g a_X|_{I,\theta}(h)$. Let $L(\cdot, X)$ be C^1. The differential errors penalized in Section 5.1 are

$$r_X(g, \theta; h) = D_g a_X\big|_{g,\theta}\big((dR_g)_I h\big) - v_{X,h}(a_X(g, \theta)), \tag{21}$$

$$s_X(\theta; h) = D_\theta L\big|_{\theta,X}[v_{X,h}(\theta)]. \tag{22}$$

For matrix groups, $(dR_g)_I h = hg$. Euclidean norms are used after flattening matrix-valued outputs.

Let $\gamma\colon [0, 1] \to G$ be a piecewise C^1 path with finitely many pieces, from I to g_2. Define $h(t)$ by $\dot\gamma(t) = (dR_{\gamma(t)})_I h(t)$ wherever the derivative exists. For fixed $g_1 \in G$ and $\theta \in \mathbb{R}^p$, write

$$u(t) = a_X(\gamma(t) g_1, \theta), \qquad w(t) = a_X(\gamma(t), a_X(g_1, \theta)).$$

The next proposition bounds the difference between applying g_1 and g_2 successively and applying their product. It only requires control along these two trajectories.

Proposition A.2 (Transport residual controls associativity error)**.** *In the preceding setup, suppose* $\lambda\colon [0, 1] \to [0, \infty)$ *is integrable and*

$$\|v_{X,h(t)}(u(t)) - v_{X,h(t)}(w(t))\| \le \lambda(t)\|u(t) - w(t)\|$$

almost everywhere. Set $\Lambda(t) = \int_0^t \lambda(s)\, ds$*. Then*

$$\begin{aligned}\|u(1) - w(1)\| \le e^{\Lambda(1)}\|u(0) - w(0)\| \\ + \int_0^1 e^{\Lambda(1) - \Lambda(s)} \Big(\|r_X(\gamma(s) g_1, \theta; h(s))\| \\ + \|r_X(\gamma(s), a_X(g_1, \theta); h(s))\| \Big)\, ds.\end{aligned}$$

The left-hand side is the finite composition error. The initial discrepancy is the identity error at $a_X(g_1, \theta)$, and the integral accumulates the transport residuals. Thus, small identity error and small transport residuals along both paths imply small composition error when the growth factor is controlled.

Proof. Right translation gives $\frac{d}{dt}(\gamma(t) g_1) = (dR_{\gamma(t) g_1})_I h(t)$. Thus, almost everywhere,

$$\begin{aligned}\dot u(t) &= v_{X,h(t)}(u(t)) + r_X(\gamma(t) g_1, \theta; h(t)), \\ \dot w(t) &= v_{X,h(t)}(w(t)) + r_X(\gamma(t), a_X(g_1, \theta); h(t)).\end{aligned}$$

The function $e(t) = \|u(t) - w(t)\|$ is absolutely continuous. The assumed growth bound and the triangle inequality imply

$$e'(t) \le \lambda(t)e(t) + \|r_X(\gamma(t)g_1, \theta; h(t))\| + \|r_X(\gamma(t), a_X(g_1, \theta); h(t))\|$$

almost everywhere. Multiplication by $e^{-\Lambda(t)}$ and integration give the asserted bound. Since $\gamma(0) = I$, we have

$$e(0) = \|a_X(I, a_X(g_1, \theta)) - a_X(g_1, \theta)\|,$$

while $\gamma(1) = g_2$ makes $e(1)$ the stated composition error. □

Loss variation along the same path. In addition to the initial loss error, variation along the path has two sources: motion of the generators across loss level sets and deviation of the candidate map from the generator dynamics. The next bound separates these contributions. Put $z(t) = a_X(\gamma(t), \theta)$.

Proposition A.3 (Infinitesimal residuals control loss variation)**.** *In the preceding setup,*

$$\begin{aligned}&\|L(a_X(g_2, \theta), X) - L(\theta, X)\| \\ &\quad \le \|L(a_X(I, \theta), X) - L(\theta, X)\| \\ &\qquad + \int_0^1 \Big(\|s_X(z(t); h(t))\| + \|D_\theta L|_{z(t),X}\|_{\mathrm{op}} \|r_X(\gamma(t), \theta; h(t))\| \Big)\, dt.\end{aligned}$$

Therefore, finite loss variation is controlled by the initial loss error, the infinitesimal loss residual, and the transport residual weighted by the loss derivative along the path. Only these trajectory-dependent quantities are needed, not a global bound on the loss derivative.

Proof. The definition of r_X gives, almost everywhere,

$$\dot{z}(t) = v_{X,h(t)}(z(t)) + r_X(\gamma(t), \theta; h(t)).$$

By the chain rule,

$$\frac{d}{dt} L(z(t), X) = s_X(z(t); h(t)) + D_\theta L\big|_{z(t),X}[r_X(\gamma(t), \theta; h(t))].$$

The composition $t \mapsto L(z(t), X)$ is piecewise C^1, hence absolutely continuous. Integrating, taking norms, and adding the initial loss discrepancy proves the bound. □

The parameterization (6) gives $u(0) = w(0)$ and $z(0) = \theta$, so both initial-error terms in the pathwise estimates vanish. We retain the general bounds below to cover other candidate maps.

Uniform bounds. We state the full assumptions for Proposition 5.1 before deriving it from the pathwise estimates. In the preceding setup, let G be connected.

Use Euclidean norms on the parameter and loss spaces, and choose a norm on $\mathfrak{g}$. Suppose the following bounds hold for all $g \in G$, $h \in \mathfrak{g}$, and $\theta, \vartheta \in \mathbb{R}^p$, with nonnegative constants $\delta, \varepsilon_{\mathrm{act}}, \varepsilon_{\mathrm{loss}}, K, M$:

$$\begin{aligned}\|a_X(I, \theta) - \theta\| \le \delta, \qquad \|r_X(g, \theta; h)\| \le \varepsilon_{\mathrm{act}}\|h\|, \\ \|s_X(\theta; h)\| \le \varepsilon_{\mathrm{loss}}\|h\|, \qquad \|D_\theta L|_{\theta,X}\|_{\mathrm{op}} \le M, \\ \|v_{X,h}(\theta) - v_{X,h}(\vartheta)\| \le K\|h\|\,\|\theta - \vartheta\|.\end{aligned} \tag{23}$$

For a piecewise C^1 path $\gamma\colon [0,1] \to G$ with finitely many pieces, define $h(t)$ by $\dot{\gamma}(t) = (dR_{\gamma(t)})_I h(t)$ wherever the derivative exists, and write

$$\ell = \int_0^1 \|h(t)\|\, dt, \qquad F_K(\ell) = \begin{cases} (e^{K\ell} - 1)/K, & K > 0, \\ \ell, & K = 0. \end{cases} \tag{24}$$

Proposition 5.1. *Under* (23)*, for every piecewise C^1 path γ with finitely many pieces from I to g_2, and every $g_1 \in G$, $\theta \in \mathbb{R}^p$,*

$$\begin{aligned}\|a_X(g_2, a_X(g_1, \theta)) - a_X(g_2 g_1, \theta)\| \le \delta e^{K\ell} + 2\varepsilon_{\mathrm{act}} F_K(\ell), \\ \|L(a_X(g_2, \theta), X) - L(\theta, X)\| \le M\delta + (\varepsilon_{\mathrm{loss}} + M\varepsilon_{\mathrm{act}})\ell.\end{aligned}$$

Proof. For the composition bound, the Lipschitz assumption gives $\lambda(t) = K\|h(t)\|$ in Proposition A.2. The initial discrepancy is at most δ, and the sum of the two transport residuals is at most $2\varepsilon_{\mathrm{act}}\|h(t)\|$. Set $q(t) = \int_0^t \|h(s)\|\, ds$. That proposition gives

$$\|u(1) - w(1)\| \le \delta e^{K\ell} + 2\varepsilon_{\mathrm{act}} \int_0^1 e^{K(\ell - q(t))}\|h(t)\|\, dt.$$

For $K > 0$, integrating the derivative of $e^{K(\ell-q(t))}$ evaluates the integral as $(e^{K\ell} - 1)/K$. For $K = 0$, the integral equals ℓ. This proves (13).

The derivative bound on $L(\cdot, X)$ makes it M-Lipschitz on $\mathbb{R}^p$, by integration along a straight line segment. Thus the initial loss discrepancy is at most $M\delta$. Substitution of the residual bounds into Proposition A.3 gives (14). □

Products of learned transformations. A path that successively applies the learned generators requires bounds only in those directions. This gives the finite-error interpretation of the group samples used in Section 5.1.

Let G be generated by $h_1, \dots, h_r$ as in (5), with $\|h_i\|_F = 1$. Retain the smooth candidate map and C^1 loss from the preceding setup. Suppose, for all $g \in G$, $\theta, \vartheta \in \mathbb{R}^p$, and i, that

$$\begin{aligned} \|a_X(I, \theta) - \theta\| &\le \delta, & \|r_X(g, \theta; h_i)\| &\le \varepsilon_{\mathrm{act}}, \\ \|s_X(\theta; h_i)\| &\le \varepsilon_{\mathrm{loss}}, & \|D_\theta L|_{\theta, X}\|_{\mathrm{op}} &\le M, \\ \|v_{X,h_i}(\theta) - v_{X,h_i}(\vartheta)\| &\le K\|\theta - \vartheta\|. \end{aligned} \tag{25}$$

All constants are nonnegative, and identity may hold approximately.

Corollary A.4 (Errors along products of learned transformations)**.** *Under* (25)*, write*

$$g_2 = \exp(t_m h_{i_m}) \cdots \exp(t_1 h_{i_1}), \qquad \ell = \sum_{j=1}^m |t_j|.$$

Then the composition and loss bounds (13)–(14) *hold for every $g_1 \in G$ and $\theta \in \mathbb{R}^p$, with $F_K(\ell) = \int_0^\ell e^{Kt}\, dt$.*

Each factor contributes $|t_j|$ to the path length. Thus the same bounds control mixed-generator compositions using residual bounds on the learned list.

Proof. For $m \ge 1$, put $b_0 = I$ and $b_j = \exp(t_j h_{i_j}) b_{j-1}$. On $[(j-1)/m, j/m]$, define

$$\gamma(t) = \exp\big((mt - j + 1) t_j h_{i_j}\big) b_{j-1}.$$

This is a piecewise smooth path from I to g_2, with right-logarithmic velocity $m t_j h_{i_j}$ on its jth interval and length ℓ. Since the residuals and $v_{X,h}$ are linear in h, the transport residuals along this interval are bounded by $m|t_j|\varepsilon_{\mathrm{act}}$, the loss residual by $m|t_j|\varepsilon_{\mathrm{loss}}$, and the growth factor in Proposition A.2 is $m|t_j|K$. That proposition therefore gives

$$\|u(1) - w(1)\| \le \delta e^{K\ell} + 2\varepsilon_{\mathrm{act}} \int_0^\ell e^{K(\ell - s)}\, ds = \delta e^{K\ell} + 2\varepsilon_{\mathrm{act}} F_K(\ell).$$

The derivative bound makes $L(\cdot, X)$ M-Lipschitz, so the initial loss discrepancy is at most $M\delta$. Substituting the same interval bounds into Proposition A.3 yields $M\delta + (\varepsilon_{\mathrm{loss}} + M\varepsilon_{\mathrm{act}})\ell$. For $m = 0$, $g_2 = I$ and both assertions follow directly from the identity-error bound. □

Inverse consistency. A further consequence bounds the error when a transformation is followed by its proposed inverse. It is obtained by applying the composition bound to the pair (g^{-1}, g).

Corollary A.5 (Error in undoing a transformation)**.** *Under* (23)*, let γ be a piecewise C^1 path with finitely many pieces from I to g^{-1}, and let ℓ be its length from* (24)*. Then*

$$\|a_X(g^{-1}, a_X(g, \theta)) - \theta\| \le \delta + \delta e^{K\ell} + 2\varepsilon_{\mathrm{act}} F_K(\ell). \tag{26}$$

This controls how well $a_X(g^{-1}, \cdot)$ undoes $a_X(g, \cdot)$; it does not assert that either candidate map is exactly invertible. Interchanging g and g^{-1} gives the reverse-composition bound, using a path from I to g.

Proof. Apply (13) with $(g_2, g_1) = (g^{-1}, g)$, and then use $\|a_X(I, \theta) - \theta\| \leq \delta$ and the triangle inequality. □

Local bounds and empirical residuals. For a particular comparison, the proof only needs transport bounds at the group–parameter pairs defining u, w, z, a growth bound between $u(t)$ and $w(t)$, and loss-residual and loss-derivative bounds along $z(t)$. To bound the initial loss error by $M\delta$, a loss-derivative bound along the segment between θ and $a_X(I, \theta)$ is enough. Thus the constants need not be finite on the entire parameter space. Small residuals at finitely many sampled points alone do not supply the trajectory bounds required by these results. Here approximate action refers to controlled identity, composition, and inverse-consistency errors on the specified domain. Closeness to an exact action and exact invertibility require additional assumptions.

B BUILDING SYMMETRIES FROM KNOWN ONES

This section proves the lifting result and its width, depth, and batch-size consequences from Section 4. The small-subnetwork generation results are proved in Appendix C.

Proposition 4.1. *Let $L\colon \Theta \times \mathcal{D}^d \to \mathbb{R}^d$ be a function, where the parameter space Θ is a product space $\Theta = \Theta_1 \times \Theta_2$, with spaces Θ_1, Θ_2. Suppose there exist functions $h\colon \Theta_1 \times \mathcal{D}^d \to S$, $f\colon \Theta_2 \times S \to T$, and $j\colon (\Theta_1 \times T) \times \mathcal{D}^d \to \mathbb{R}^d$, such that for every $\theta = (\theta_1, \theta_2) \in \Theta$ and $X \in \mathcal{D}^d$, $L(\theta, X) = j\big((\theta_1, f\big(\theta_2, h(\theta_1, X)\big)\big), X\big)$. If $a\colon S \to (G \times \Theta_2 \to \Theta_2)$ is a G-symmetry of f, then there is an induced G-symmetry of L, $a'\colon \mathcal{D}^d \to (G \times \Theta \to \Theta)$, defined by $a'_X(g, (\theta_1, \theta_2)) = \big(\theta_1, a_{h(\theta_1, X)}(g, \theta_2)\big)$.*

Proof. We need to show that a' satisfies the identity and associative law of a group action and preserves L.

Since a is a group action on Θ_2, it satisfies the identity axiom $a_{h(\theta_1,X)}(I, \theta_2) = \theta_2$. Applying this in the definition of a', we get $a'_X(I, (\theta_1, \theta_2)) = (\theta_1, a_{h(\theta_1,X)}(I, \theta_2)) = (\theta_1, \theta_2)$.

Since a is a group action on Θ_2, it satisfies the associative law $a_{h(\theta_1,X)}(g_2 g_1, \theta_2) = a_{h(\theta_1,X)}(g_2, a_{h(\theta_1,X)}(g_1, \theta_2))$, for all $g_1, g_2 \in G$. It follows that a' also satisfies the associative law: $a'_X(g_2 g_1, (\theta_1, \theta_2)) = (\theta_1, a_{h(\theta_1,X)}(g_2 g_1, \theta_2)) = (\theta_1, a_{h(\theta_1,X)}(g_2, a_{h(\theta_1,X)}(g_1, \theta_2))) = a'_X(g_2, a'_X(g_1, (\theta_1, \theta_2)))$

Finally, since a is a symmetry of f, we have $f(a_{h(\theta_1,X)}(g, \theta_2), h(\theta_1, X)) = f(\theta_2, h(\theta_1, X))$, for all $g \in G$. It follows that a' preserves the value of L: $L(a'_X(g, \theta), X) = j\big((\theta_1, f\big(a_{h(\theta_1,X)}(g, \theta_2), h(\theta_1, X)\big)\big), X\big) = j\big((\theta_1, f\big(\theta_2, h(\theta_1, X)\big)\big), X\big) = L(\theta, X)$. □

Corollary 4.2. *Consider the parameter space $\Theta(m, h, n) = \mathbb{R}^{m \times h} \times \mathbb{R}^{h \times n}$ and data space $\mathcal{D}(n, k) = \mathbb{R}^{n \times k}$. Let σ apply the same row function $\sigma_{row}\colon \mathbb{R}^k \to \mathbb{R}^k$ at every hidden width. Define $L_{mnhk}\colon \Theta(m, h, n) \times \mathcal{D}(n, k) \to \mathbb{R}^{m \times k}$ by $L_{mnhk}((U, V), X) = U\sigma(VX)$. If L_{mnhk} has a G-symmetry, then $L_{mnh'k}$ has a G-symmetry for every $h' > h$.*

Proof of Corollary 4.2. Use the same row function σ_{row} at every hidden width. Write $\theta_2 = (U_{1:h}, V_{1:h})$ for the first h hidden units and $\theta_1 = (U_{h+1:h'}, V_{h+1:h'})$ for the remaining units. Here subscripts select columns of U and rows of V. Since σ acts independently on rows,

$$L_{mnh'k}((U, V), X) = L_{mnhk}(\theta_2, X) + L_{mn(h'-h)k}(\theta_1, X).$$

Apply Proposition 4.1 with $S = \mathcal{D}(n, k)$, $T = \mathbb{R}^{m \times k}$, and

$$h(\theta_1, X) = X, \qquad f(\theta_2, X) = L_{mnhk}(\theta_2, X), \qquad j((\theta_1, Y), X) = Y + L_{mn(h'-h)k}(\theta_1, X).$$

If a_X is the given symmetry of L_{mnhk}, the induced action is

$$a'_X(g, (\theta_1, \theta_2)) = \big(\theta_1, a_X(g, \theta_2)\big).$$

It applies the original symmetry to the first h units and leaves all remaining units fixed. □

Corollary 4.3. *Let $\Theta = \Theta_1 \times ... \times \Theta_l$ be a parameter space. Consider a list of spaces $V_0 = \mathcal{D}^d$, $V_l = \mathbb{R}^d$, and V_1, ..., V_{l-1}. Let $L\colon \Theta \times \mathcal{D}^d \to \mathbb{R}^d$ be a function defined recursively by $\{L_i\}_{i=1}^l$ with $L_i\colon \Theta_i \times V_{i-1} \to V_i$, such that $L = \phi_l$ where $\phi_i = L_i(\theta_i, \phi_{i-1}) \in V_i$ and $\phi_0 = X$. If for some $1 \leq i \leq l$, L_i has a G-symmetry, then L has a G-symmetry.*

Proof. Write $\theta_{-i} = (\theta_1, \ldots, \theta_{i-1}, \theta_{i+1}, \ldots, \theta_l)$ and $L_{j,\theta_j}(z) = L_j(\theta_j, z)$. With Θ_1 in Proposition 4.1 identified with the product of the parameter spaces other than Θ_i, and Θ_2 identified with Θ_i, define

$$\begin{aligned} h(\theta_{-i}, X) &= \big(L_{i-1,\theta_{i-1}} \circ \cdots \circ L_{1,\theta_1}\big)(X), \\ f(\theta_i, z) &= L_{i,\theta_i}(z), \\ j((\theta_{-i}, z), X) &= \big(L_{l,\theta_l} \circ \cdots \circ L_{i+1,\theta_{i+1}}\big)(z). \end{aligned}$$

Empty compositions are identity maps, so $h(\theta_{-1}, X) = X$ when $i = 1$ and $j((\theta_{-l}, z), X) = z$ when $i = l$. The recursive definition of L gives

$$L(\theta, X) = j\big((\theta_{-i}, f(\theta_i, h(\theta_{-i}, X))), X\big).$$

The prefix h does not depend on θ_i, and $f = L_i$ has the assumed G-symmetry. Proposition 4.1 therefore gives a G-symmetry of L. □

Proposition 4.4. *Let $L_d\colon \Theta \times \mathcal{D}^d \to \mathbb{R}^d$ be a function that is applied pointwise on each of d data points in a data batch. If L_d admits a G-symmetry, then $L_{d'}$ admits a G-symmetry for all $d' < d$.*

Proof. Fix $1 \leq d' < d$. For $X' = (x_1, \ldots, x_{d'})$, define the deterministic padding

$$\iota(X') = (x_1, \ldots, x_{d'}, \underbrace{x_1, \ldots, x_1}_{d-d' \text{ entries}}).$$

Let a be the given G-symmetry of L_d, and set $a'_{X'}(g, \theta) = a_{\iota(X')}(g, \theta)$. For each fixed X', the padded batch is fixed, so $a'_{X'}$ inherits identity and associativity from $a_{\iota(X')}$. In particular,

$$a'_{X'}(g_2, a'_{X'}(g_1, \theta)) = a_{\iota(X')}(g_2 g_1, \theta) = a'_{X'}(g_2 g_1, \theta).$$

Since L_d and $L_{d'}$ apply the same function pointwise, projecting $L_d(\theta, \iota(X'))$ onto its first d' components gives $L_{d'}(\theta, X')$. Applying this projection to the L_d-invariance identity proves $L_{d'}(a'_{X'}(g, \theta), X') = L_{d'}(\theta, X')$. Thus a' is a G-symmetry of $L_{d'}$. □

C Proofs and further discussion of small-subnetwork symmetries

This section proves Theorem 4.5 and Corollary 4.6, and gives the explicit action and examples.

Throughout this section, we use the main-text setup and notations: $F_X(U, V) = U\sigma(VX)$, where $U \in \mathbb{R}^{m \times H}$, $V \in \mathbb{R}^{H \times n}$, $X \in \mathbb{R}^{n \times k}$, $H > k$, and σ is smooth and row-wise. Let $Z = \sigma(VX)$. The set B has k indices, C is its complement, and Z_B is invertible on the nonempty domain Ω_B in (4). Subscripts on U select columns; those on V, Z select rows.

C.1 Decomposing an entire output-preserving transformation

Theorem 4.5. *Let $(\phi_t)_{t \in [0,1]}$ be a smooth family of diffeomorphisms of Ω_B satisfying $\phi_0 = \mathrm{id}$ and $F_X \circ \phi_t = F_X$ for every $t \in [0, 1]$. Assume that all ϕ_t equal the identity outside the same compact set $K \subset \Omega_B$. Then $\phi_1 = \psi_N \circ \ldots \circ \psi_1$ for finitely many output-preserving diffeomorphisms ψ_i, each changing at most $k + 1$ hidden-unit blocks and preserving the combined output of those units on X.*

Proof. The output equation $Y = U_B Z_B + U_C Z_C$ uniquely determines

$$U_B = (Y - U_C Z_C) Z_B^{-1}. \tag{27}$$

Consequently,

$$\Phi(U, V) = (Y, V, U_C), \qquad Y = F_X(U, V),$$

is a diffeomorphism from Ω_B onto

$$\mathcal{O} = \mathbb{R}^{m\times k} \times \{V : \det \sigma(V_B X) \neq 0\} \times \mathbb{R}^{m\times(H-k)},$$

with inverse given by (27).

Flatten these coordinates as (y, z) with $y = Y$ and $z = (V, U_C) \in \mathbb{R}^d$, where $d = Hn + m(H - k)$. The conjugate maps $\psi_t = \Phi \circ \phi_t \circ \Phi^{-1}$ preserve y and equal identity outside a common compact subset of $\mathcal{O}$. Extend them by identity to the entire Euclidean coordinate space. These extensions are smooth diffeomorphisms because their supports stay inside $\mathcal{O}$.

First take an increment $\psi = \psi_b \circ \psi_a^{-1}$ sufficiently close to identity that $\|D(\psi - \mathrm{id})\|_\infty < 1$. Write $\psi(y, z) = (y, f_1(y, z), \ldots, f_d(y, z))$ and define

$$Q_j(y, z) = (y, f_1(y, z), \ldots, f_j(y, z), z_{j+1}, \ldots, z_d), \qquad Q_0 = \mathrm{id}, \quad Q_d = \psi.$$

The derivative of $Q_j - \mathrm{id}$ is obtained by zeroing rows of $D(\psi - \mathrm{id})$, so its operator norm is also less than one. Each Q_j is a global diffeomorphism: for any target η, the equation $x = \eta - (Q_j(x) - x)$ has a unique solution by the contraction theorem, and the nonsingular derivative gives a smooth inverse. Moreover, Q_j fixes $\mathcal{O}^c$ pointwise. Bijectivity then forces $Q_j(\mathcal{O}) = \mathcal{O}$.

Set $T_j = Q_j \circ Q_{j-1}^{-1}$. The outputs $Q_j(x)$ and $Q_{j-1}(x)$ differ only in their jth z coordinate. Hence T_j changes only z_j, keeps y fixed, and

$$\psi = T_d \circ \cdots \circ T_1.$$

If z_j is an entry of V_i, the inverse formula (27) changes only V_i and U_B. If it is an entry of U_i for $i \in C$, only U_i and U_B change. Each pulled-back factor therefore changes units in $B \cup \{i\}$, a set of at most $k + 1$ units. Since it fixes all other units and preserves F_X, it preserves this subnetwork's combined output.

Finally, smoothness of (ψ_t) and their common compact support imply that $\psi_b \circ \psi_a^{-1}$ is uniformly C^1-close to identity whenever $|b - a|$ is sufficiently small. Choose a finite partition of $[0, 1]$ with all increments satisfying the preceding bound. Factor each increment and compose the factorizations. Conjugating back by Φ proves the result. □

Relation to decomposition results. Factorization into maps preserving smaller sets of coordinates is related to fiber-preserving diffeomorphism decompositions (Haller & Teichmann, 2003). The global coordinates above give a direct proof for this network and identify the hidden-unit support of every factor. This differs from constructing a constant-output path between two parameter points (Nguyen, 2019), and from complete ReLU equivalence through rewrites that may change architecture (Zhang & Bölcskei, 2026).

C.2 An explicit compensating action

A unit can change its incoming weights while a few other units cancel its output change. The output coordinates in the preceding proof give the following family without a compact-support restriction. For $P \in \mathbb{R}^{m\times(H-k)}$ and $Q \in \mathbb{R}^{(H-k)\times n}$, prescribe

$$V_B' = V_B, \quad V_C' = V_C + Q, \quad U_C' = U_C + P, \quad U_B' = \big[F_X(U, V) - U_C' \sigma(V_C' X)\big]\big[\sigma(V_B X)\big]^{-1}.$$

Corollary 4.6. *The maps $a_X^B((P, Q), (U, V)) = (U', V')$ defined above form a smooth free action of the additive group $\mathbb{R}^{(H-k)(m+n)}$ on Ω_B and preserve F_X. This action is generated by commuting one-parameter subgroups, each changing at most $k + 1$ hidden-unit blocks and preserving their combined output.*

This family of data-dependent symmetry fixes V_B and is therefore not the entire class in Theorem 4.5. Its construction follows from the output coordinates in that theorem's proof, rather than from applying the compact-support factorization to a translation.

Proof. In the coordinates $\Phi(U, V) = (Y, V, U_C)$ established in the preceding proof, the displayed transformation fixes Y, V_B and translates (U_C, V_C) by (P, Q). Translations compose by addition, have inverses $(-P, -Q)$, act freely, and preserve Y. All group parameters are allowed because V_B, and hence the domain condition, remain unchanged. Substitution into (27) gives U'_B.

A translation of a single entry of P or Q modifies only one unit $j \in C$ and the output columns U_B. All other units remain fixed, and the total output is unchanged, so the combined contribution of $B \cup \{j\}$ is preserved. Coordinate translations commute and generate the additive group. □

Example: two sigmoid units. For a single scalar input, let $F_x = u_1\sigma(v_1x) + u_2\sigma(v_2x)$ with logistic sigmoid σ. Since $\sigma(v_1x) > 0$, any $q \in \mathbb{R}$ gives an exact transformation

$$v'_2 = v_2 + q, \qquad u'_1 = u_1 + u_2\frac{\sigma(v_2x) - \sigma((v_2 + q)x)}{\sigma(v_1x)}, \qquad u'_2 = u_2, \quad v'_1 = v_1.$$

The second unit changes its incoming weight, and the first cancels its output change.

Relation to earlier constructions. Zhao et al. (2023) give explicit nonlinear data-dependent group actions for single-input batches, while Corollary 4.6 provides an explicit larger-batch action. The action on Ω_B is constructed by fixing k units with independent batch features and using their output weights to compensate for changes in the remaining units. Keeping these units' incoming weights fixed preserves the invertibility needed for compensation, without requiring the activation to preserve rank everywhere. The subnetwork view makes both the construction and its $k+1$-unit support bound explicit. Output compensation also appears in constant-output path constructions (Nguyen, 2019), but the transformations here additionally have a specified group structure.

Application to gated feedforward blocks. The proofs use only a linear output readout and independence of each feature row from the other units' incoming parameters. They therefore apply to the smooth gated block

$$F_X = U\big[\sigma(W_gX) \odot (W_uX)\big],$$

by treating one row of W_g, one row of W_u, and the matching column of U as one unit (Shazeer, 2020). If k feature rows are independent, the same construction has support at most $k+1$ units, with an explicit additive subgroup of dimension $(H-k)(m+2n)$. An unchanged residual branch can be added without affecting either output-preservation argument. In a token-processing block, k counts all protected token positions used at this boundary; the rank condition must hold for that complete feature matrix.

D Experimental setup and validation

The one-generator small-network experiments use five optimization seeds, 101–105, using independently drawn parameter perturbations and transformations for training, checkpoint selection, recipe selection, and final testing. The task seed is 31415. Replications therefore measure optimization variability for fixed base networks and conditioning instances. The higher-dimensional protocols are given in Appendix F, and the pretrained-model protocol is given in Appendix H.

D.1 Implementation of the discovery objectives

This subsection gives the group, sampling, normalization, and regularization details for the direct-map method in Section 5. In subnetwork discovery, unchanged parameters form part of the fixed context. Affine-field extraction is described in Appendix F.

Learning the candidate group. We fix the matrix size s and generator count r, and learn normalized matrices $h_1, \ldots, h_r$ shared across data batches. Their finite exponential products define G in (5). This subgroup has a connected immersed Lie-group structure. Its Lie algebra is the smallest vector space containing the h_i and closed under matrix commutators. We use this intrinsic smooth structure when G is not closed in $\mathrm{GL}(s, \mathbb{R})$. The action network is smooth in its ambient matrix input, so its restriction to $G \times \mathbb{R}^p$ is smooth. Symmetry conditions are imposed on this restriction.

The generator count r can be smaller than the resulting algebra dimension. For example, E_{12} and E_{21} generate $[E_{12}, E_{21}] = \mathrm{diag}(1, -1)$ and hence a three-dimensional algebra. Here E_{ij} is the 2×2 matrix with a single one in position (i, j). Proposition A.1 applies to a generating list of any such group.

Exact identity. Equation (6) represents every smooth identity-preserving map at the unrestricted function level by taking $b_X(g, \theta) = a_X(g, \theta) - \theta$. Both evaluations use the same deterministic network and context. Compute their difference before adding θ. For a restricted parameter domain, the architecture must additionally keep its output in that domain.

Sampling group elements. Let μ sample the original parameter–batch pairs (θ, X). Sample a product length, generator indices, and real coefficients, and multiply their matrix exponentials to obtain g. Let π denote this distribution at the current learned matrices and let ν return each h_i with probability $1/r$. Include negative coefficients, different orders, and products of several generators. The action network receives the product matrix, so different expressions for the same matrix give the same network input. Keep the coefficient distribution fixed within each training stage. We begin with short products at moderate coefficient magnitudes and expand the range as training progresses.

For a supplied connected matrix group G, fix $h_1, \ldots, h_r$ to a Frobenius-orthonormal basis of $\mathfrak{g}$, choose π on G, and let ν be uniform on this basis. Here $r = \dim \mathfrak{g}$ and only the action map is learned.

Sampling parameter points. The differential losses use an equal mixture of original pairs from μ and pairs $(a_X(q, \theta), X)$ obtained with an independently sampled product q. This exposes training to points visited by the candidate transformations; the pathwise bounds require residual control along the full evaluated paths. When constructing these additional sample points, detach them from the sampling computation. Derivatives of the residuals are then taken at the selected points. The regularizers always use the original distribution μ so that their motion scale remains fixed.

The finite composition and finite preservation terms use the same mixture of original and transformed starting points. For $\mathcal{L}_{\mathrm{comp}}$, use a mixture of independent products (g_1, g_2) and inverse pairs (g_1, g_1^{-1}). The batch X stays fixed within each composition. Both sides of the composition residual are differentiated through in full, including the inner transformed parameter. This finite term adds direct supervision of composition over the sampled range. The infinitesimal-only variant omits both finite terms.

Automatic differentiation. Compute directional derivatives using Jacobian–vector products. In $D_g a_X|_{g,\theta}(hg)$, differentiate only the group input and hold the parameter input fixed. For the group derivative at I, the identity matrix in $b_X(I, \theta)$ remains a constant. Retain the dependence of both the direction hg and the evaluation point $a_X(g, \theta)$ when backpropagating the complete transport loss. Hold the sampled indices and coefficients fixed and differentiate through the matrix exponentials and action network. Matrix-valued outputs are flattened before taking Euclidean norms.

Preservation target. The target function is held fixed during discovery. We retain every network or subnetwork output component inside the squared norm. For ReLU targets, derivatives are interpreted away from activation boundaries, and finite preservation is checked separately. Exact reconstruction of ReLU rescalings uses positive homogeneity, including at activation boundaries (Appendix F). The finite-error theorem in Section 5.3 assumes a C^1 target.

Finite preservation and objective variants. The finite-preservation term is

$$\mathcal{L}_{\mathrm{finite}} = \mathbb{E}_{X,\theta,g} \| L(a_X(g, \theta), X) - L(\theta, X) \|_2^2. \tag{28}$$

The base hybrid objective is (12) with $\gamma_2 > 0$. The finite comparator removes $\mathcal{L}_{\mathrm{invariance}}$ and $\mathcal{L}_{\mathrm{assoc}}$ and includes a positive multiple of $\mathcal{L}_{\mathrm{finite}}$, retaining finite composition and the same regularizers. The infinitesimal-only comparator sets $\gamma_2 = 0$ and omits $\mathcal{L}_{\mathrm{finite}}$. Stronger hybrid recipes add $\mathcal{L}_{\mathrm{finite}}$ to the base objective. Each objective is tuned separately on validation data.

Fixed normalization. Section 5 writes the losses before task-specific normalization. In implementation, preservation residuals are divided by a fixed output scale s_F, and transport and composition

residuals in physical parameter coordinates are divided by $s_\theta\beta$, before squaring. The regularizers use v_i/s_θ in place of v_i, so β specifies a field magnitude in units of the fixed parameter scale s_θ. These positive constants leave the zero-residual conditions unchanged. They are fixed before optimization; their definitions for small networks and Pythia are given below and in Appendix H. Reduced-coordinate actions use the induced physical parameter norm; the Pythia coordinates are isometric for these increments. The motion-relative composition and inverse errors used in evaluation have sample-dependent denominators and are distinct from these fixed training normalizations.

Interpreting the regularizers. The scale penalty targets mean squared field magnitude β^2, while the diversity penalty discourages duplicate, opposite, or correlated fields. Both use uncentered moments so that constant translation fields count as nontrivial motion. They allow a field to vanish at individual parameter points. Pointwise independence is evaluated separately through the rank of $[v_1(\theta, X) \ \cdots \ v_r(\theta, X)]$.

Use nonzero initialization for the generators and the group-dependent part of b_X, since the squared scale penalty has zero gradient at a fully collapsed parameterization. We evaluate actual finite parameter displacements as well as generator norms. The regularization weights and reference scale are selected on validation data, balancing invariance, composition, nontrivial motion, and redundancy.

Examining the learned structure. Repeatedly adjoining commutators and taking their linear span determines the generated Lie algebra in exact arithmetic. Numerical dimension estimates should report the rank tolerance and its sensitivity. For an exact action, (18) requires the same bracket relations in the induced parameter fields. Directions whose fields vanish identically belong to the kernel of the infinitesimal action. Thus the matrix algebra, its effective action, and the rank of its fields at a parameter point describe different aspects of the learned symmetry. A nonzero derivative with respect to X shows dependence of the parameterization on data. To examine dependence of the preserved function, evaluate the fixed transformation $a_X(g, \theta)$ on a separate batch X'.

Training with bounded products and finitely many samples is assessed through held-out errors, including longer compositions. The pathwise bounds in Appendix A.4 require residual control along the evaluated paths.

D.2 One-generator experimental setup

Models and parameter sampling. Linear discovery uses 2×2 factors near identity, rejecting numerically singular samples. ReLU recovery uses a bias-free $(2, 4, 2)$ network. Focused sigmoid discovery preserves $U_B\sigma(V_BX) + u_c\sigma(v_cX)$, with V_B, u_c fixed and (U_B, v_c) transformed. The single-input case has $m = n = k = 1$; the two-input case has $m = n = k = 2$. Compensating units are selected by pivoted QR during setup and their indices remain fixed. The selected feature matrix must have condition number at most 10^4. The tanh and GELU targets are bias-free $(4, 16, 16, 2)$ students trained for $3{,}000$ steps against a fixed random teacher on Gaussian inputs. We transform the first and last matrices and hold the middle matrix fixed. Gaussian parameter perturbations have coordinate standard deviation 0.20 for linear and scalar sigmoid, 0.30 for ReLU recovery, 0.15 for two-input sigmoid, 0.05 for the deeper networks, and 0.20 for host lifting. Data-dependent fits use a fixed conditioning batch.

Training. The action network has three width-256 SiLU hidden layers and exact identity by construction. We learn one Frobenius-normalized matrix generator per action. Training uses Adam in float32, gradient clipping at 10, and equal proportions of original and transformed parameter locations for the differential residuals. One quarter of the composition pairs are inverse pairs. Field-scale regularization is active and sparsity is disabled. Each objective has a separate validation search over learning rates, batch sizes, and residual weights. Initial candidates train for $5{,}000$ steps, and the strongest candidates continue to $15{,}000$ total steps before selecting a recipe for fresh-seed replication. The selected learning rates are 10^{-3} or 3×10^{-4}, with batches of 16 or 32. The scalar-sigmoid hybrid recipe uses coefficient radius 0.8 and at most six factors; the other main-text hybrid recipes use radius 0.5 and at most three factors. The finite and infinitesimal comparators retain the same action architecture and field regularizers. Selected recipes can differ in batch size and transformation range, so the comparisons assess tuned objectives at equal final step budgets.

Finite-action tests. We keep the conditioning batch and parameter mask fixed within each comparison. Group samples are signed products $\exp(t_m h_{i_m}) \cdots \exp(t_1 h_{i_1})$. Coefficient radius ρ is the upper bound on their total absolute coefficient length $\sum_j |t_j|$, sampled uniformly on $[0.2\rho, \rho]$. It controls the group input rather than fixing the resulting parameter displacement. The test grid contains $\rho \in \{0.1, 0.3, 0.5, 0.8, 1.2\}$ and one, two, four, or eight factors, with 512 samples per cell. Tables use the declared cell $\rho = 0.5$ with one factor. Longer signed products can have smaller displacement through cancellation.

Let s_θ be the root-mean-square parameter-coordinate magnitude and s_F the root-mean-square output norm, estimated from 64 calibration samples and fixed before optimization. Write $\theta_1 = a_X(g_1, \theta)$ and $\theta_{12} = a_X(g_2, \theta_1)$. We measure

$$E_{\text{out}} = \frac{\|F_X(\theta_1) - F_X(\theta)\|}{s_F},$$
$$E_{\text{comp}} = \frac{\|\theta_{12} - a_X(g_2 g_1, \theta)\|}{\|\theta_1 - \theta\| + \|\theta_{12} - \theta_1\|},$$
$$E_{\text{inv}} = \frac{\|a_X(g_1^{-1}, \theta_1) - \theta\|}{\|\theta_1 - \theta\|}.$$

For example, a composition error of 10^{-3} is a discrepancy of 0.1% of the two-step displacement. An output error of 10^{-3} is 0.1% of the fixed calibration scale s_F, rather than a percentage change in prediction accuracy. The implementation adds 10^{-12} to denominators after scaling parameter norms by s_θ. Transport is normalized by the corresponding field norm. Subdivision compares one exponential transformation with four applications at one quarter of its coefficient, normalized by the one-step motion. Cancellation is $\|\Delta Y_B + \Delta Y_C\| / (\|\Delta Y_B\| + \|\Delta Y_C\| + 10^{-12})$. Each one-generator small-network table entry aggregates the within-seed statistic by its median across all five seeds.

Joint numerical tolerances. The error and motion requirements are fixed before final testing. The 95th-percentile output error must be at most 10^{-3}; composition, inverse, and subdivision errors must be at most 10^{-2}; and relative transport error must be at most 2×10^{-2}. These requirements apply to both one- and two-factor tests at radius 0.5. One-factor median motion must be at least $0.05 s_\theta$. Compensation additionally requires median moving-output change of at least $0.005 s_F$ and cancellation-error 95th percentile at most 10^{-2}. Columns headed "Fits" count how many fitted actions satisfy every requirement. The reported errors and motion include every seed, regardless of that count. These tolerances summarize the sampled tests and are not global guarantees.

Table 1 reports the complete one-generator results, including all seeds in each setting.

Setting	Motion (%)	Output	Composition	Inverse	Fits
Linear	5.25	4.73×10^{-4}	3.68×10^{-3}	3.27×10^{-3}	5/5
ReLU	3.74	8.70×10^{-4}	3.07×10^{-3}	6.14×10^{-3}	3/5
Sigmoid, $k=1$	11.95	5.23×10^{-5}	3.71×10^{-4}	5.16×10^{-4}	5/5
Sigmoid, $k=2$	6.35	1.46×10^{-4}	1.48×10^{-3}	2.16×10^{-3}	4/5
Tanh, separated layers	1.50	9.57×10^{-4}	3.29×10^{-3}	3.53×10^{-3}	2/5
GELU, separated layers	1.50	1.07×10^{-3}	4.41×10^{-3}	3.77×10^{-3}	2/5
ReLU subnetwork	4.81	7.51×10^{-4}	4.29×10^{-3}	6.62×10^{-3}	5/5

Table 1: Learned one-generator approximate actions in the small-network experiments. Errors are medians across five seeds of within-seed 95th percentiles over 512 samples at coefficient radius 0.5 and one factor. Output error uses a fixed calibration scale; composition and inverse errors are relative to attempted motion. Motion is the median relative change in the transformed parameter block. The final column counts fits satisfying every tolerance in Appendix D, including two-factor tests, transport, subdivision, and nontrivial motion.

Generated curves. We additionally evaluate $\theta(t) = a_X(\exp(th), \theta_0)$ at 61 equally spaced times in $[-0.5, 0.5]$, using 32 fresh parameter points per seed in float64. Each trajectory keeps θ_0, X, and h fixed. The horizontal coordinate t is the signed exponential coefficient and $t = 0$ is the original parameter point. The plotted output drift is $\|F_X(\theta(t)) - F_X(\theta_0)\|_F / s_F$. Its random reference

is a fixed parameter translation $\theta_0 + tz$, with direction drawn once and scale calibrated to match root-mean-square physical motion at $t = 0.3$ on separate validation parameters. Aggregate curves retain all five seeds. Illustrations use seed 101 and the first sampled parameter point, selected before evaluating these curves. These are descriptive trajectory evaluations; the randomized finite-action tests and their joint numerical criterion are unchanged.

Main-text curve summaries. In Figure 2(a), each unit's contribution is $Y_j = u_j\sigma(v_jX)$, and its signed change is $\Delta Y_j = Y_j(\theta(t)) - Y_j(\theta_0)$. These changes and their sum are divided by s_F. This panel uses the prespecified illustration above. Panel (b) takes the median output drift over the 32 initial parameter points separately in each seed, then the median across seeds. In panel (c), the horizontal coordinate is the seed median raw parameter displacement and the vertical coordinate is the seed 95th-percentile full-network output error; each is then aggregated by its median across five seeds. Shaded regions span the corresponding seed summaries. The host comparison uses identical complete network parameters and inputs for the two methods, as detailed in Appendix G.

E Objective comparisons and batch-specific preservation

Table 2 compares the objectives on the one-generator small-network settings. The hybrid objective gives lower median preservation, composition, and inverse errors than the finite comparator in each paired setting. On scalar sigmoid, the infinitesimal-only objective also produces accurate preserving actions. Adding finite composition to the tuned recipe reduces its median composition error from 2.42×10^{-3} to 3.71×10^{-4}.

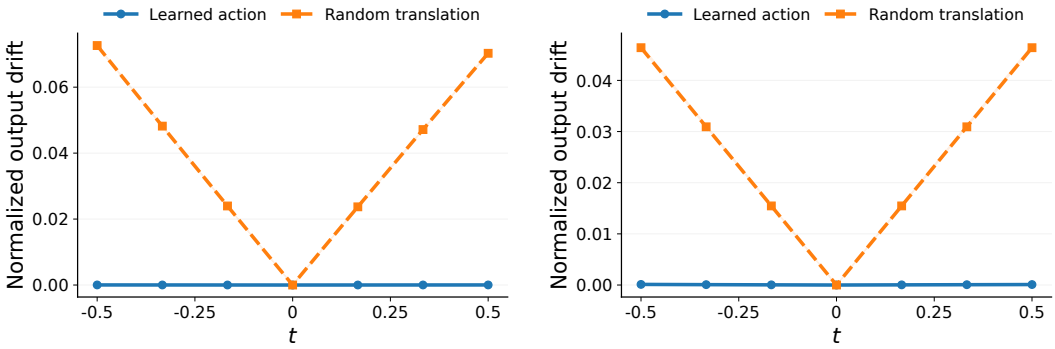
Setting	Objective	Output	Comp.	Inverse	Transport	Fits
Linear	Hybrid	4.73×10^{-4}	3.68×10^{-3}	3.27×10^{-3}	3.92×10^{-3}	5/5
Linear	Finite	1.31×10^{-3}	7.33×10^{-3}	7.84×10^{-3}	1.27×10^{-2}	0/5
ReLU	Hybrid	8.70×10^{-4}	3.07×10^{-3}	6.14×10^{-3}	6.55×10^{-3}	3/5
ReLU	Finite	2.56×10^{-3}	1.10×10^{-2}	1.50×10^{-2}	1.87×10^{-2}	0/5
Sigmoid, $k=1$	Hybrid	5.23×10^{-5}	3.71×10^{-4}	5.16×10^{-4}	8.88×10^{-4}	5/5
Sigmoid, $k=1$	Finite	1.07×10^{-4}	3.06×10^{-3}	3.60×10^{-3}	4.24×10^{-3}	5/5
Sigmoid, $k=1$	Infinitesimal	5.16×10^{-5}	2.42×10^{-3}	1.43×10^{-3}	1.45×10^{-3}	5/5
Sigmoid, $k=2$	Hybrid	1.46×10^{-4}	1.48×10^{-3}	2.16×10^{-3}	2.03×10^{-3}	4/5
Sigmoid, $k=2$	Finite	2.50×10^{-4}	5.43×10^{-3}	4.15×10^{-3}	9.40×10^{-3}	3/5
Sigmoid, $k=2$	Infinitesimal	1.71×10^{-4}	1.70×10^{-3}	1.66×10^{-3}	2.36×10^{-3}	3/5
Tanh	Hybrid	9.57×10^{-4}	3.29×10^{-3}	3.53×10^{-3}	5.23×10^{-3}	2/5
Tanh	Finite	2.40×10^{-3}	2.36×10^{-2}	9.14×10^{-3}	3.27×10^{-2}	0/5
GELU	Hybrid	1.07×10^{-3}	4.41×10^{-3}	3.77×10^{-3}	7.52×10^{-3}	2/5
GELU	Finite	2.75×10^{-3}	2.53×10^{-2}	9.75×10^{-3}	3.69×10^{-2}	0/5
Local ReLU host	Hybrid	7.51×10^{-4}	4.29×10^{-3}	6.62×10^{-3}	7.66×10^{-3}	5/5

Table 2: Objective comparisons for the one-generator small-network settings. Errors are medians across all five seeds of their within-seed 95th percentiles at radius 0.5 and one factor. The joint numerical criterion additionally includes the two-factor cell, subdivision, and motion. Each objective is tuned independently.

E.1 Transformations across non-contiguous layers

We also learn one-generator transformations in tanh and GELU networks with widths $(4, 16, 16, 2)$, trained on synthetic teacher regression. Discovery changes the first and last weight matrices while keeping the middle layer fixed, giving a 96-parameter search space. With 24 protected inputs, median relative parameter motion is approximately 1.5%. Two of five fits for each activation satisfy all the joint numerical tolerances in Appendix D; all five are retained in the tables and figures.

To examine finite preservation, we evaluate the output along curves $\theta(t) = a_X(\exp(th), \theta_0)$ with θ_0, X, and h fixed along each curve. Compared with random parameter translations calibrated to comparable motion, the learned curves have much smaller output variation (Figure 2(b)). The hybrid objective also reduces the median composition error from 2.36×10^{-2} to 3.29×10^{-3} for



(a) One protected input

(b) Two protected inputs

Figure 4: Output variation along sigmoid trajectories with one or two protected inputs. The horizontal axis is the signed coefficient t in $\theta(t) = a_X(\exp(th), \theta_0)$. Normalized output drift is $\|F_X(\theta(t)) - F_X(\theta_0)\|_F / s_F$, with s_F fixed before training. Learned action follows this curve; random translation follows $\theta_0 + tz$ with z fixed and scaled on validation parameters to comparable motion at $t = 0.3$. Curves take medians over 32 initial points, then across all five seeds; shading shows the minimum and maximum seed medians.

tanh and from 2.53×10^{-2} to 4.41×10^{-3} for GELU. These measurements describe coordinated approximate transformations on the tested parameter neighborhoods. The first/last-layer support is specified before training.

E.2 Preservation on conditioning and fresh batches

To examine batch-specific preservation, we first construct $\theta' = a_X(g, \theta)$ using the conditioning batch, and then evaluate these same weights on an independently sampled X'. Table 3 shows small errors on X and larger errors on X'. These measurements use 128 parameter samples and one- to three-factor products at radius 0.5. They describe one frozen transformation evaluated on different inputs.

Setting	Conditioning batch	Fresh batch, same weights
Sigmoid, $k = 1$	4.46×10^{-5}	1.11×10^{-1}
Sigmoid, $k = 2$	1.14×10^{-4}	1.13×10^{-2}
Tanh	8.06×10^{-4}	3.58×10^{-3}
GELU	9.23×10^{-4}	5.48×10^{-3}

Table 3: Preservation on conditioning and fresh batches using the same transformed weights. Entries are medians across five seeds of within-seed 95th-percentile errors, using the original calibration scale. Both columns use exactly the same frozen weights.

F Recovered directions and higher-dimensional actions

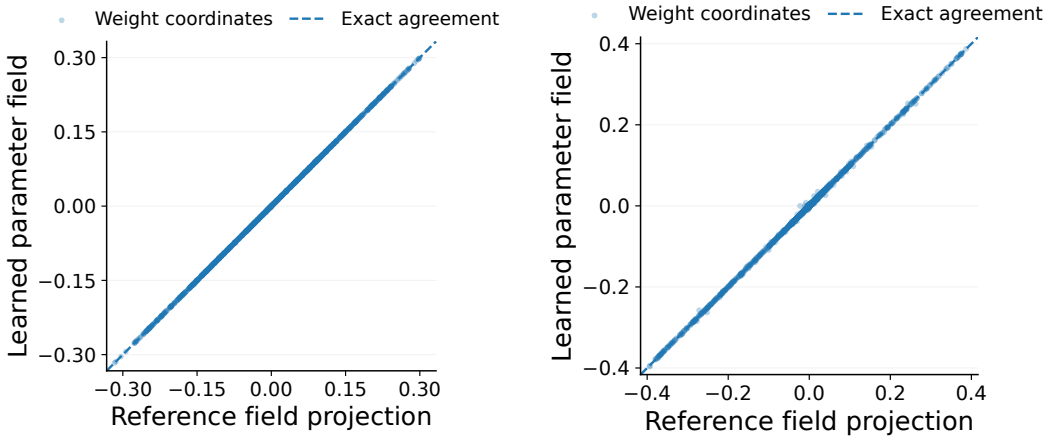
F.1 One-parameter reference directions

We compare the induced field $v_{X,h}(\theta) = D_g a_X|_{I,\theta}[h]$ with reference generator values at the same parameter points. The containment error is the normalized distance of the learned tangent span from the reference span. Table 4 uses 24 parameter samples per seed at a fixed context and relative rank tolerance 10^{-4}. The one-generator matrix algebra is one-dimensional and abelian by construction. Its induced field has sampled rank one in these fits. The small containment errors demonstrate agreement with the reference directions. Each experiment targets a single one-parameter subgroup.

Figure 5 visualizes the reference agreement directly. Each point compares one coordinate of the learned field with the same coordinate of its pointwise projection onto the analytic reference span.

Setting	Field dim.	Orbit rank	Reference-span error
Linear	1	1	1.92×10^{-3}
ReLU	1	1	7.14×10^{-3}
Sigmoid, $k=1$	1	1	1.35×10^{-4}
Sigmoid, $k=2$	1	1	9.93×10^{-4}
Local ReLU host	1	1	3.91×10^{-3}

Table 4: Induced-field agreement for the reported one-generator actions. Containment entries are medians across five seeds of mean normalized reference-span distances. Field dimension and sampled orbit rank equal one in each seed.



(a) Linear network

(b) ReLU network

Figure 5: Induced parameter fields and known symmetry directions. Each dot (weight coordinates in the legend) represents one coordinate of $v_{X,h}(\theta) = D_g a_X|_{I,\theta}[h]$, the instantaneous parameter change per unit coefficient. Its horizontal value is that coordinate after orthogonal projection onto the reference tangent span at the same θ; its vertical value is the learned field coordinate. The dashed exact-agreement line is $y = x$. All coordinates from 32 parameter configurations and five seeds are shown. The projection is pointwise, so the plots measure tangent-direction agreement.

This projection allows the coefficients to depend on the parameter point. The plot therefore illustrates tangent-direction agreement, while the finite-action tests evaluate composition and preservation separately.

F.2 Affine-field extraction and exact verification

We search for parameter fields $v(\theta) = A\theta + b$ without providing a reference basis or its dimension. For each network, we sample 512 complete parameter vectors and four Gaussian inputs per vector. An entry in a weight matrix with input width d has distribution $\mathcal{N}(0, 1/d)$. With $J = D_\theta F_X(\theta)$ and $\bar{\theta} = (\theta^\top, 1)^\top$, the constraints are

$$J[A\ b]\bar{\theta} = 0.$$

They are linear in the $p(p+1)$ unknown coefficients of $[A\ b]$. We stack the componentwise equations, normalize their rows, and take the right singular vectors below 10^{-8} times the largest singular value. This is an affine extension of the linear-generator extraction used by Moskalev et al. (2022). We repeat extraction with seeds 101–103. The recovered nullities are unchanged at relative thresholds 10^{-4}, 10^{-6}, and 10^{-8} in every run.

The affine-field solver enforces the sampled infinitesimal preservation equation directly; it does not optimize the neural objective in (12).

Embed each recovered field as $H = \left[\begin{smallmatrix} A & b \\ 0 & 0 \end{smallmatrix}\right]$. The generated matrices act linearly on $(\theta, 1)$, so the finite composition law is supplied by matrix multiplication. Preservation is tested on 128 independently

sampled parameter/input pairs per seed. Each group sample is a product of four exponentials with independently sampled coefficients in $[-0.18, 0.18]$. The output norm is divided by a fixed root-mean-square output norm on these test points. Composition and inverse errors are relative to physical motion. This experiment measures extraction of preserving generators within the affine class.

The solver recovers four- and eight-dimensional hidden-basis algebras in the two- and three-layer linear networks and three-, six-, and eight-dimensional rescaling algebras in the ReLU networks (Table 5). All three extraction seeds identify the corresponding reference span. Finite transformations of the floating-point estimates preserve the outputs to approximately 10^{-15} in float64, with median relative parameter changes of 4.2%–6.4%.

Network	Widths	Dimension	Center	Derived	Output error
Linear	$(2, 2, 2)$	4	1	3	1.10×10^{-15}
Linear	$(2, 2, 2, 2)$	8	2	6	1.47×10^{-15}
ReLU	$(2, 3, 2)$	3	3	0	1.13×10^{-15}
ReLU	$(2, 3, 3, 1)$	6	6	0	1.16×10^{-15}
ReLU	$(2, 2, 2, 2, 2, 1)$	8	8	0	1.26×10^{-15}

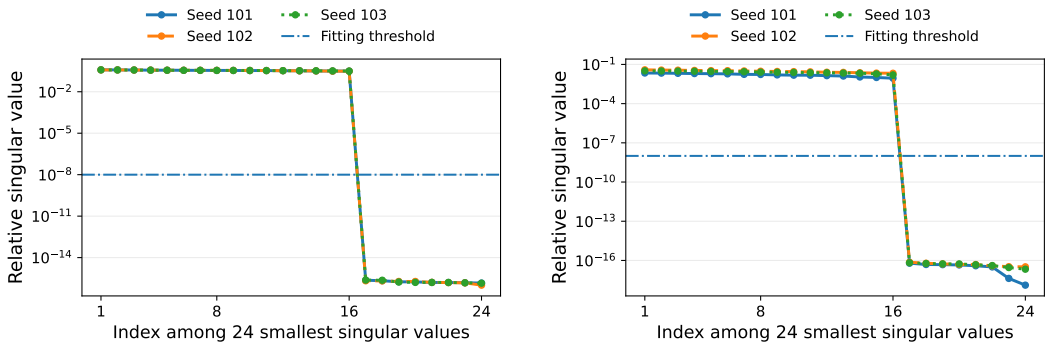
Table 5: Higher-dimensional recovery by the affine-field solver. Widths include input and output dimensions. The dimension is inferred from the sampled constraints; all three extraction seeds agree. The center consists of generators commuting with every generator, and the derived algebra is the span of their commutators. Exact reconstruction confirms the reference-span identification and the displayed algebra dimensions. Output errors are medians across seeds of within-seed 95th percentiles over 128 independent parameter/input samples and four-factor transformations.

Exact reconstruction and interpretation. We use pivoted QR to choose coordinates for the recovered span and reconstruct its basis with rational coefficients of denominator at most 32. These reconstructed matrices are distinct from the floating-point estimates. Exact arithmetic verifies their linear independence, preservation, equality with the reference span, and bracket closure. All 15 reconstructed families satisfy these exact checks. For linear networks, every component of $D_\theta F_X(\theta)[A\theta + b]$ vanishes as a polynomial in the weights. For ReLU networks, the reconstructed fields are exact linear combinations of hidden-unit rescalings. Positive homogeneity then gives preservation at all inputs and parameters, including activation boundaries.

For example, in the three-layer linear network the reconstructed span agrees with the generators of

$$(W_1, W_2, W_3) \longmapsto (G_1 W_1,\ G_2 W_2 G_1^{-1},\ W_3 G_2^{-1}), \qquad G_1, G_2 \in \mathrm{GL}^+(2).$$

The product of the three matrices is unchanged. This identifies the recovered algebra as $\mathfrak{gl}(2) \oplus \mathfrak{gl}(2)$, with dimension eight, center dimension two, and derived dimension six. In the five-layer ReLU network, the eight hidden-unit rescalings give an eight-dimensional abelian algebra. These identifications concern the recovered reference families; the sampled nullspace calculation alone is not a classification of every symmetry of the architecture.



(a) Three-layer linear network

(b) Five-layer ReLU network

Figure 6: Singular values of the stacked affine-field constraints. The horizontal axis orders the 24 smallest singular values from largest to smallest; each vertical value is divided by the largest singular value of the complete constraint matrix. The fitting-threshold line is 10^{-8}, below which a direction is retained. Seed labels identify three independently sampled extraction datasets. Each has eight retained directions in both networks, separated by a large gap. Table 5 gives their different bracket structures.

G Subnetwork discovery and lifting

Paired host evaluation. Local and full discovery are evaluated on the same 128 complete host parameter vectors and inputs per seed, with identical coefficient draws and two factors per group sample. Their supports contain 10 and 88 parameters. Table 6 reports raw displacements and absolute host-output errors. At radius 0.5, the per-seed error ratios favor local discovery by 4.7–9.9 times. The preservation targets and supports differ, so this experiment measures the combined subnetwork strategy.

Radius	Local motion	Full motion	Local output	Full output
0.1	0.0119	0.0115	1.92×10^{-4}	1.12×10^{-3}
0.3	0.0430	0.0416	5.91×10^{-4}	4.09×10^{-3}
0.5	0.0486	0.0472	9.12×10^{-4}	5.26×10^{-3}
0.8	0.0853	0.0828	1.61×10^{-3}	8.20×10^{-3}

Table 6: Paired local and full-parameter discovery in the $(2, 8, 8, 1)$ ReLU host. Motion is the median across seeds of within-seed median raw displacement. Output is the median across seeds of within-seed 95th-percentile absolute host error.

H Subnetwork actions in pretrained language models

We fit a separate action for each fixed prefix from the training split of WikiText-103. The reported sites are Pythia-160M layer 6 with eight tokens and Pythia-1B layer 3 with 16 tokens, using zero-based layer indices. Each result includes all three seeds 101–103. Nearby-parameter tests and original-checkpoint installation are separate evaluations.

H.1 Parameterization and training

Let $\widetilde{X}$ contain the block inputs at all protected positions, augmented with a bias row. The selected boundary contribution is $U_B Z_B + U_C \sigma(\widetilde{V}_C \widetilde{X})$, where C contains one moving unit and B contains k compensating units. We hold $\widetilde{V}_B, U_C$, all other units, and the output bias fixed. The compensator feature matrix Z_B is chosen by pivoted QR and remains fixed. The complete augmented input has rank k, and the accepted Z_B has condition number at most 10^4.

The action uses coordinates (D, W) defined by

$$\widetilde{X} = QR, \qquad U_C = OR_U, \qquad \widetilde{V}_C = \widetilde{V}_{C,0} + DQ^\top, \qquad U_B = U_{B,0} + OW,$$

where Q and O have orthonormal columns. With $P_0 = \widetilde{V}_{C,0}\widetilde{X}$, we train on the reduced output

$$f_X(D, W) = WZ_B + R_U\sigma(P_0 + DR).$$

Here f_X is the output target $L(\cdot, X)$ used in the discovery losses. Boundary-output changes equal $O\Delta f_X$ and satisfy $\|O\Delta f_X\|_F = \|\Delta f_X\|_F$. Likewise, $\|\Delta DQ^\top\|_F^2 + \|O\Delta W\|_F^2 = \|\Delta D\|_F^2 + \|\Delta W\|_F^2$. The $2k$ coordinates thus retain the physical output and motion norms while restricting incoming changes to directions visible on the prefix. Each action call receives both current coordinates.

We use the width-256, three-hidden-layer SiLU decoder, exact identity, and one learned Frobenius-normalized 2×2 generator. Training uses float32 Adam, batch size 32, gradient clipping at 10, and no sparsity. The sampled state is the original checkpoint with probability 0.2 and a Gaussian coordinate perturbation otherwise, with standard deviation 0.2 times the original selected-weight root-mean-square magnitude. Original and transformed residual locations are mixed equally, and one quarter of composition pairs are inverse pairs. The training radius is 0.5 with up to three signed exponential factors.

For each objective, three recipes train for $5{,}000$ steps and the strongest two continue to $15{,}000$. Validation selects a recipe for three fresh-seed fits of $15{,}000$ steps. Base recipes use learning rates 10^{-3} or 3×10^{-4}, preservation weight 10, composition weight 1, and hybrid transport weight 1. The stronger recipe uses learning rate 3×10^{-4}, preservation weight 30, composition weight 3, and hybrid transport weight 3 plus finite preservation weight 30. Scale weight is 1. Final tests use an independent random stream. Objectives are tuned separately at equal step and search budgets.

H.2 Nearby-parameter validation

Local tests use 512 samples per radius/factor cell and the one-generator numerical criterion in Appendix D. The fixed parameter scale is the original selected-weight root-mean-square magnitude; the output scale is the norm of the original moving unit's contribution on the prefix. Table 7 gives the three-seed Pythia-160M eight-token results.

Objective	Output	Composition	Inverse	Transport	Cancellation	Fits
Finite	6.00×10^{-4}	8.00×10^{-3}	3.16×10^{-3}	1.40×10^{-2}	2.43×10^{-2}	0/3
Hybrid	2.18×10^{-4}	1.93×10^{-3}	1.12×10^{-3}	1.72×10^{-3}	9.98×10^{-3}	2/3

Table 7: Pythia-160M, layer 6, eight protected tokens. Errors are medians across three seeds of within-seed 95th percentiles at radius 0.5 and one factor. The final column counts fits satisfying every tolerance, including two factors, subdivision, and visible compensation.

H.3 Installed weights and full-model outputs

Installation begins at the original checkpoint. We change the selected incoming row, bias, and compensating output columns, cast to float32, run the complete model, and restore the weights. Logit measurements test how preservation of the trained boundary output carries through the remaining computation. The prefix stays fixed within fitting and action-law tests.

For each coefficient radius $\rho \in \{0.1, 0.3, 0.5, 0.8\}$, we evaluate $g = \exp(th)$ with $\|h\|_F = 1$ at 16 signed coefficients. Eight magnitudes are sampled uniformly on $[0.2\rho, \rho]$ and paired with their negatives. The same normalized magnitudes $|t|/\rho$ are reused across radii, and methods share coefficient draws within each seed. Thus ρ bounds the exponential coefficient, not the physical weight displacement. Let Z and Z' denote the original and changed logits on the protected positions. The reported relative error is $\|Z' - Z\|_F/(\|Z\|_F + 10^{-12})$, and the absolute root-mean-square error is $\|Z' - Z\|_F/\sqrt{N}$ for N logit entries. Cancellation compares the sum of the moving and compensating output changes with the sum of their norms. Motion is measured on the selected weights after casting. Selected-weight change is $100\|\theta'_S - \theta_S\|_2/\|\theta_S\|_2$, computed on the raw

selected weights S at the original checkpoint. Composition and inverse tests round after each action and retain the resulting stored weights for the next application. Their relative errors use the same two-step and one-step displacement denominators defined in Appendix D. TF32 is disabled.

The analytic comparator translates the incoming coordinates and solves for compensating output coefficients. Its direction and motion scale are fixed on validation data, as are those of the random additive translation. Incoming-only retains the learned incoming update and omits compensation, giving less total motion. The learned decoder receives neither compensation labels nor a solve.

To examine data dependence, we evaluate the same frozen edits on the protected prefix, its teacher-forced continuation, and three reserved test-article windows. In the 1B example, the learned relative-logit error tails are 1.50×10^{-6}, 2.12×10^{-5}, and 2.24×10^{-5}, respectively. The fresh-text statistic averages the three article errors for each edit before computing its within-seed percentile. These measurements describe preservation on the fitted prefix and behavior of the same changed weights on unprotected inputs. All full-model results in this subsection concern the original checkpoint.

Model	Transformation	Motion (%)	Relative logits	RMS logits	Cancellation
160M	Learned	0.1541	8.82×10^{-7}	7.13×10^{-4}	4.54×10^{-3}
160M	Analytic	0.1523	8.36×10^{-7}	6.76×10^{-4}	1.95×10^{-5}
160M	Incoming only	0.0231	1.84×10^{-6}	1.48×10^{-3}	1.00×10^{0}
160M	Random	0.1520	3.09×10^{-6}	2.50×10^{-3}	6.56×10^{-1}
1B	Learned	0.0687	1.50×10^{-6}	5.37×10^{-6}	3.29×10^{-3}
1B	Analytic	0.0695	1.41×10^{-6}	5.02×10^{-6}	5.22×10^{-4}
1B	Incoming only	0.0105	7.35×10^{-5}	2.62×10^{-4}	1.00×10^{0}
1B	Random	0.0688	1.93×10^{-4}	6.90×10^{-4}	8.70×10^{-1}

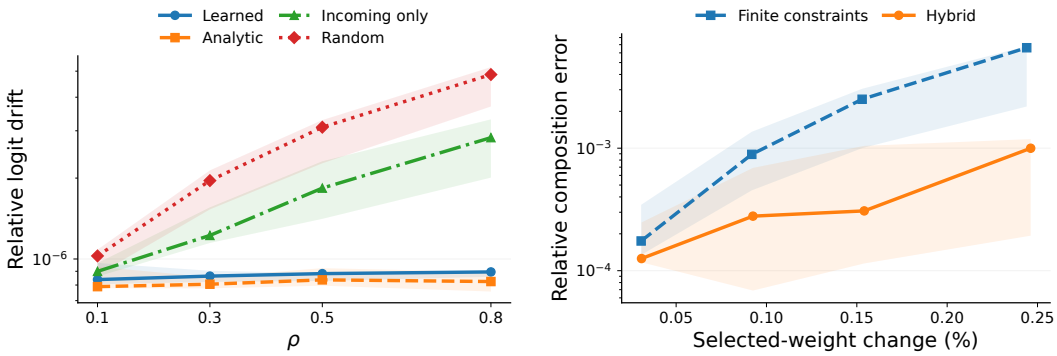
Table 8: Installed transformations at radius 0.5, starting from the original checkpoint. Motion is the median percentage change in the selected parameter block. Error columns are medians across all three seeds of within-seed 95th percentiles over 16 transformations. Analytic and random controls are calibrated to similar physical motion; incoming-only removes the learned compensation.

Figure 7 gives the Pythia-160M counterpart to Figure 3. At layer 6 with eight protected tokens, learned compensation keeps protected-logit drift near the analytic reference across the displayed coefficient range. At radius 0.5, hybrid training reduces stored-weight composition error from 2.51×10^{-3} to 3.08×10^{-4}, with similar raw displacement. These plots evaluate the installed edits at the original checkpoint; Table 7 separately describes nearby-parameter tests.

The analytic reference has a measured finite-precision logit error of 8.36×10^{-7} in the 160M example and 1.41×10^{-6} in the 1B example. The learned transformations reach similar protected-logit errors, with larger cancellation residuals. Table 9 separately reports composition and inverse consistency of the stored-weight transformations. At the 1B site, hybrid training reduces the median stored composition error from 3.21×10^{-3} to 5.81×10^{-4} at nearly identical displacement.

Model	Objective	Raw motion	Composition	Inverse
160M	Finite	2.94×10^{-3}	2.51×10^{-3}	1.91×10^{-3}
160M	Hybrid	2.97×10^{-3}	3.08×10^{-4}	5.92×10^{-4}
1B	Finite	2.39×10^{-3}	3.21×10^{-3}	2.86×10^{-3}
1B	Hybrid	2.39×10^{-3}	5.81×10^{-4}	9.00×10^{-4}

Table 9: Stored-weight action consistency at radius 0.5. Both learned objectives use the same selected support and separately tuned recipes. Error entries are medians across three seeds of within-seed 95th percentiles. These evaluations start at the original checkpoint.



(a) Preserving protected outputs

(b) Composing learned transformations

Figure 7: Pythia-160M, layer 6, at the original checkpoint and one fixed eight-token prefix. (a) Coefficient radius ρ bounds $|t|$ in $g = \exp(th)$; magnitudes lie in $[0.2\rho, \rho]$. Relative logit drift is $\|Z' - Z\|_F / \|Z\|_F$ on the prefix. Learned denotes the hybrid action; analytic solves for compensation; random is a motion-calibrated translation; incoming-only removes the learned compensation. (b) Selected-weight change is $100\|\theta'_S - \theta_S\|_2 / \|\theta_S\|_2$ on edited parameters S. Relative composition error compares sequential and combined edits, divided by their total two-step displacement, with rounding after each update. Finite constraints and hybrid use separately tuned objectives. Each curve aggregates within-seed statistics across all three seeds by their median; error statistics are 95th percentiles over 16 signed transformations per radius, and motion statistics are medians. Shading spans the seed statistics.