INTERPRETING SO(N) MULTIPLICATION: DEEP NETWORKS GENERALIZE BY LEARNING AN ALGORITHM

Anonymous authors
Paper under double-blind review

ABSTRACT

We reverse engineer networks trained on SO(n) multiplication (rotations of n-dimensional spheres) and discover that adding layers results in networks learning clean generalizing algorithms that are not learned by shallow 1- or 2-hidden layer networks. In deep networks learning SO(3), we discover this task is solved by neurons learning to activate on different axes of the sphere. These 'axial'-neurons combine information to compute the multiplicative product via Rodrigues' Rotation Formula, which is an efficient algorithm used to compute the rotation of a vector in space, given an axis and angle of rotation. The story is similar in higher dimensions, but neurons learn to rotate about planes instead of an axis. To our knowledge, this is the first Lie Group that has been reverse-engineered and interpreted, whereas prior work has focused on finite groups.

1 INTRODUCTION

The idea of studying *what* deep neural networks (DNNs) learn by constructing datasets Power et al. (2022); Nanda et al. (2023); Chughtai et al. (2023b); Gromov (2023) out of mathematical objects known as groups has evolved into giving deep learning (DL) researchers the ability to probe hypotheses under the guidance of the scientific method. This interpretability subfield began with studies on modular addition (cyclic group multiplication), originally documenting *what* is learned at the neuron level Nanda et al. (2023). Subsequently, more *finite* group tasks were studied to investigate *The Universality Hypothesis* Chughtai et al. (2023b), the idea that across random seeds, networks may learn "similar" circuitry. The field later evolved to full multi-level understandings at different levels of abstraction (inspired by Marr & Poggio (1979)), giving an understanding of neural representations (the activation manifolds of collaborating neurons between layers) Moisescu-Pareja et al. (2026); Wei et al. giving a view into *The Platonic Representation Hypothesis* Huh et al. (2024), the idea that as DNNs are scaled to be larger, they converge to one shared understanding of the data, *i.e. they learn one platonic representation*. Indeed, even the global perspective has now been studied, discovering DNNs utilize neural representations to implement efficient divide-and-conquer algorithms, yielding a new *divide-and-conquer* hypothesis McCracken et al. (2025), that DNNs would learn all *finite* group tasks via implementing some divide-and-conquer algorithm at the global level.

This work investigates an infinite group, importantly requiring a different loss function. We use a Lie Group to build datasets of multiplications of rotations of n-dimensional spheres called $SO(n)$. To our knowledge, this is the first continuous (not finite and discrete) group to be interpreted. This contrasts with prior work focusing on *finite* and discrete groups, resulting in cross entropy classification being the loss. Indeed, we see this as an important motivation to start studying Lie Group datasets: *finite* group tasks are all discrete. Thus, it could be that the reason DNNs have been found to divide-and-conquer group multiplications by splitting the group into a logarithmic number of discrete Cayley graphs (Wei et al.; Hilgert et al.) may be due to the discrete nature of the groups being studied. Our contributions follow:

1. We provide evidence that networks begin to divide-and-conquer SO(3) multiplication as depth is added (by learning the efficient Rodrigues' rotation algorithm). 2. We discover shallow networks learn significantly different representations compared to DNNs, which wasn't observed in finite groups. 3. We describe what is learned at multiple levels: what individual neurons learn as well as the global algorithms learned by DNNs trained on $SO(n)$.

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.

2 RELATED WORK

Group multiplication datasets became a go-to testing ground for studying phenomena in DL, starting with modular addition, first *how* networks learn, being training dynamics during stochastic gradient descent (SGD) Power et al. (2022); Morwani et al. (2024); Gromov (2023); He et al. (2024); Liu et al. (2022); Mohamadi et al. (2023); Mallinar et al. (2024) and later *what* networks learn Nanda et al. (2023); Chughtai et al. (2023b); Zhong et al. (2023); Stander et al. (2024); Yip et al. (2024); Doshi et al. (2023); Wu et al. (2025); McCracken et al. (2025); Tao et al. (2025); Moisescu-Pareja et al. (2026), from individual neurons, to neural representations, and even the global algorithm the network learns were all studied.

First, work pushed to study The Universality Hypothesis Olah et al. (2020); Li et al. (2015), which is the idea that DNNs will solve different datasets using similar principles. First Nanda et al. (2023) discovered cosines were learned to solve modular addition and later Chughtai et al. (2023b) claimed an algorithm was learned by DNNs solving any group task. This algorithm was that the irreducible representation matrices were being learned in DNNs and then multiplied to solve the group multiplications in finite groups, but was later shown to be false in general by two works. First Zhong et al. (2023) showed that a modified transformer could learn "pizza" representations for modular addition, while the algorithm by Nanda et al. (2023) generalized in Chughtai et al. (2023b), would have clock representations. This was followed by Moisescu-Pareja et al. (2026)'s study on modular addition with multiple architectures, finding all transformers including Nanda et al. (2023)'s learn pizza discs, which are linear projections of a universal manifold that multi-layer perceptrons (MLPs) learn to solve modular addition, being tori. This meant universality was false in a strict sense and that transformers weren't implementing Chughtai et al. (2023a)'s algorithm universally. These works all studied modular addition, so the question of universality between *different* tasks was still open.

Stander et al. (2024) showed that coset circuits, not matrix multiplications (Chughtai et al., 2023b), were being learned on networks learning alternating group multiplication. This resulted in McCracken et al. (2025) revisiting modular addition, and discovering DNNs were utilizing approximations of cosets in order to implement a global "coset intersection algorithm" known as the Chinese Remainder Theorem. This recovered universality across different finite group tasks, and gave rise to a new hypothesis: *DNNs will universally learn to divide-and-conquer to compute finite group tasks McCracken et al. (2025)*. Since then, this result has been affirmed in the dihedral (Wei et al.), homocyclic (McCracken et al., 2026) and alternating group multiplications (Hilgert et al.).

We build on this prior work by being first to interpret and present what DNNs learn on a very different type of group (compared to discrete finite groups): a continuous Lie group.

3 BACKGROUND

The special orthogonal group. $\mathrm{SO}(n) = \{R \in \mathbb{R}^{n\times n} : R^\top R = I,\ \det R = 1\}$ is the group of orientation-preserving rotations of $\mathbb{R}^n$. We focus on the $n = 3$ case. We study its multiplication map $(X, Y) \mapsto R = XY$ with $X, Y \in \mathrm{SO}(3)$, where multiplying two rotations produces their composed rotation. Every $R \in \mathrm{SO}(3)$ can be described by an angle $\theta \in [0, \pi]$ and a unit axis $u \in S^2 = \{u \in \mathbb{R}^3 : \|u\| = 1\}$: R rotates space through angle θ about u. Rodrigues' rotation formula Cheng & Gupta (1989) is the classical closed-form algorithm for constructing the corresponding rotation matrix. With I identity matrix,

$$R = I + \sin\theta[u]_\times + (1 - \cos\theta)[u]_\times^2, \qquad \text{where} \qquad [u]_\times = \begin{pmatrix} 0 & -u_3 & u_2 \\ u_3 & 0 & -u_1 \\ -u_2 & u_1 & 0 \end{pmatrix} \tag{1}$$

Product-space linear functionals. Take $\mathbb{R}^{3\times3}$ with Frobenius inner product $\langle A, R\rangle_F = \mathrm{tr}(A^\top R)$. Linear functionals on this space are represented uniquely by a matrix A: $f_A(R) = \langle A, R\rangle_F$, $(\mathbb{R}^{3\times3})^* \cong \mathbb{R}^{3\times3}$. For $R = XY$, the functional is $f_A(X, Y) = \langle A, XY\rangle_F = \sum_{i,j,k} A_{ij}X_{ik}Y_{kj}$. The function is linear in product R but bilinear in two factors X and Y.

The orthogonal decomposition. Every $A \in \mathbb{R}^{3\times3}$ decomposes uniquely as a sum of $A_0, A_1, A_2 \in \mathbb{R}^{3\times3}$, which are respectively scalar, antisymmetric, and traceless and symmetric (see App. A.3 for an example).

Axial neurons and product functionals. We call a neuron *axial* when its dependence on the product rotation $R = R(u, \theta)$ is organized around a preferred direction *and* a preferred angle about that direction. Let p be the preferred axis. We say an axial neuron has label $\ell = 1$ if it has antisymmetric dependence on $p \cdot u$, while an $\ell = 2$ axial neuron has a symmetric dependence described generally by $u^\top S u$, where $S = A_2$ is the symmetric-traceless component of its product functional f_A, in the uniaxial case, by $P_2(p \cdot u)$. Here, P_n is the nth Legendre polynomial, and $P_2(z) = (3z^2 - 1)/2$ in this case. We give the estimators used to recover preferred axes in App. A.2. The activations of an example axial neuron are shown in Fig.2.

More generally, if the preactivation h_j of neuron j is approximately linear in the product rotation, then $h_j(X, Y) \approx b_j + \langle A_j, XY \rangle_F$. The neuron is then associated with a dual element $A_j \in (\mathbb{R}^{3\times3})^* \cong \mathbb{R}^{3\times3}$, which itself decomposes as $A_j = A_{j,0} + A_{j,1} + A_{j,2}$. Axial $\ell = 1$ and $\ell = 2$ tuning can therefore be viewed as structured components of this more general product-functional description rather than as separate representational objects. This formulation distinguishes product representation from factor representation. A product-linear neuron has the form $\langle A, XY \rangle_F$, whereas a function linear in the two factors separately has the form $\langle B, X \rangle_F + \langle C, Y \rangle_F$. The former contains the multiplicative interaction defining the group product; the latter does not.

3.1 METHODS

Task and architectures studied. We study multilayer perceptrons (MLPs) trained to multiply pairs of rotations in $\mathrm{SO}(3)$. For independently Haar-distributed $X, Y \in \mathrm{SO}(3)$, networks receive concatenated matrix entries $x = [\mathrm{vec}(X), \mathrm{vec}(Y)] \in \mathbb{R}^{18}$, where vec flattens a matrix to a vector, and is trained to predict the nine entries of $R = XY$. The output is projected to $\mathrm{SO}(3)$ by normalizing the nine-dimensional output, reshaping it as a $\mathbb{R}^{3\times3}$ matrix, then applying an Singular Value Decomposition (SVD) projection Geist et al. (2024). We train DNNs across a full grid of widths $w \in \{512, 1024, 2048\}$, numbers of intermediate hidden layers $d \in \{1, 2, 3, 4, 5\}$, dataset sizes $N \in \{5000, 10000, 20000\}$, learning rates $\eta \in \{3 \times 10^{-5}, 10^{-4}, 3 \times 10^{-4}\}$, and weight decays $\lambda \in \{0, 10^{-4}, 10^{-3}\}$. The resulting 405 configurations are each trained across eight random seeds, for a total of 3,240 networks. Hidden layers use LayerNorm followed by SiLU. Networks are trained for 1001 epochs with AdamW, batch size 128, and an exponentially decaying learning rate (multiplied by 0.85 every 120 epochs). See training and analysis details in App. B.

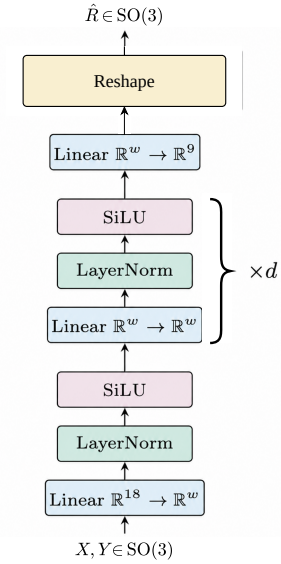

Figure 1: Network architecture. Width w and depth d vary across experiments.

Identifying product functionals. Our neuron-level analysis asks whether a hidden neuron can be described as a linear functional of the group product. For each neuron j, we fit

$$h_j(X, Y) \approx b_j + \langle A_j, XY \rangle_F, \tag{2}$$

where $A_j \in \mathbb{R}^{3\times3}$ and $\langle A, R \rangle_F = \mathrm{tr}(A^\top R)$. The coefficients are estimated on an independent fitting set and evaluated on a held-out evaluation set. As controls, we compare Eq. 2 with linear functions of X, Y, and (X, Y) separately, along with shuffled and independently sampled product matrices. All neuron-level fits below use the same fit/evaluation separation unless specified otherwise.

For each fitted functional, we decompose

$$A = \underbrace{\frac{\mathrm{tr}(A)}{3} I}_{A_0} + \underbrace{\frac{A - A^\top}{2}}_{A_1} + \underbrace{\frac{A + A^\top}{2} - A_0}_{A_2} \tag{3}$$

We quantify the contribution of each component by $\|A_\ell\|_F^2 / \|A\|_F^2$, and independently fit models restricted to the $\ell = 0$, $\ell = 1$, $\ell = 2$, and pairwise combined subspaces.

Preferred axes and Rodrigues fits. To determine whether individual neurons exhibit the axis–angle structure predicted by the decomposition above, write the product rotation as $R = R(u, \theta)$, where $u \in S^2$ is its rotation axis and $\theta \in [0, \pi]$ its rotation angle. Preferred axes are estimated using the fitting set. For a centered neuron response $y_n = h_{j,n} - \bar{h}_j$, the $\ell = 1$ preferred direction is

$$p_{1,j} = \frac{\sum_k y_k u_k}{\|\sum_k y_k u_k\|}. \tag{4}$$

For $\ell = 2$, we form

$$M_j = \frac{1}{N} \sum_n y_n \left(u_n u_n^\top - \frac{1}{3} I \right) \tag{5}$$

and define $p_{2,j}$ as the eigenvector of M_j whose eigenvalue has the largest absolute magnitude. The absolute-eigenvalue criterion captures symmetric-traceless tuning of either sign.

Let $P_1(z) = z$ and $P_2(z) = (3z^2 - 1)/2$. We fit the $\ell = 1$ Rodrigues form

$$h_j \approx a_j + b_j \sin\theta \, P_1(u^\top p_{1,j}), \tag{6}$$

and the uniaxial $\ell = 2$ form

$$h_j \approx a_j + b_j (1 - \cos\theta) \, P_2(u^\top p_{2,j}). \tag{7}$$

Here the preferred direction is estimated as above, while the intercept and Rodrigues coefficient are fitted separately for each neuron. We additionally fit the more general single-axis $\ell = 2$ model

$$h_j \approx a_j + b_j P_2(z_j) + c_j (1 - \cos\theta) + d_j (1 - \cos\theta) P_2(z_j), \qquad z_j = u^\top p_{2,j}, \tag{8}$$

along with the unrestricted five-dimensional symmetric-traceless $\ell = 2$ model. Axis-only, angle-only, and alternative-angular-factor fits serve as controls (App. B.5).

Classification of $\ell = 1$ and $\ell = 2$ neurons. For each neuron, let $R^2_{\mathrm{rod},1}$, $R^2_{\mathrm{rod},2}$, and $R^2_{\mathrm{rod},2,\mathrm{full}}$ denote the coefficients of determination for Eqs. 6, 7, and 8, respectively, and define

$$r_1 = R^2_{\mathrm{rod},1}, \qquad r_2 = \max\left(R^2_{\mathrm{rod},2}, R^2_{\mathrm{rod},2,\mathrm{full}}\right).$$

Using a fixed threshold $\tau = 0.7$, we classify

$$\mathrm{family}(j) = \begin{cases} \ell = 1, & r_1 \geq \tau \text{ and } r_1 \geq r_2, \\ \ell = 2, & r_2 \geq \tau, \\ \text{unclassified}, & \text{otherwise.} \end{cases} \tag{9}$$

This permits an independent comparison between resulting families and the $\ell = 1$ and $\ell = 2$ components of A.

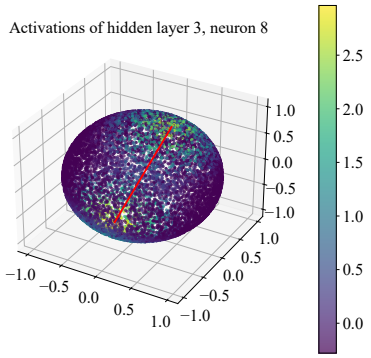

Figure 2: Activations of an example axial neuron (preferred axis in red)

Recovering the angular coefficient. We next test whether the Rodrigues angular factors themselves are recovered from the neurons. For each classified neuron its preferred axis is fixed and the neuron is fitted to

$$h_j \approx a_j + P_\ell(u^\top p_{\ell,j}) \, g_j(\theta), \tag{10}$$

with $g_j(\theta)$ independently estimated over 12 angular bins. We compare the recovered function with $\sin\theta$ for $\ell = 1$ and $1 - \cos\theta$ for $\ell = 2$, measuring their absolute cosine similarity and improvement in held-out fit obtained by replacing the corresponding Rodrigues factor with free function g_j.

Input-space and population structure. To characterize how these functionals appear in the two input rotations, we project the axes of X and Y onto a neuron's preferred direction, $x = u_X^\top p$ and $y = u_Y^\top p$, and examine the resulting two-dimensional activation maps. We additionally fit the product Legendre expansion

$$h_j(x, y) \approx \sum_{\ell,m=0}^{6} c_{\ell m} P_\ell(x) P_m(y), \qquad \text{where } P_k(x) \text{ is the } k\text{th Legendre polynomial} \tag{11}$$

and compare the empirical maps with those predicted independently by $b_j + \langle A_j, XY \rangle_F$.

At the population level, the fitted matrices $\{A_j\}$ provide coordinates for functionals represented across a layer. We analyze their singular-value spectra, effective rank, and coverage of $\ell = 0, 1, 2$ subspaces. Following the population-level Rodrigues analysis, we linearly decode

$$T_1 = \sin\theta [u]_\times, \qquad T_2 = (1 - \cos\theta)[u]_\times^2, \qquad T_1 + T_2 = R - I$$

from each layer via linear regression, comparing the full population, the classified $\ell = 1$ and $\ell = 2$ populations, their union, and size-matched random subsets.

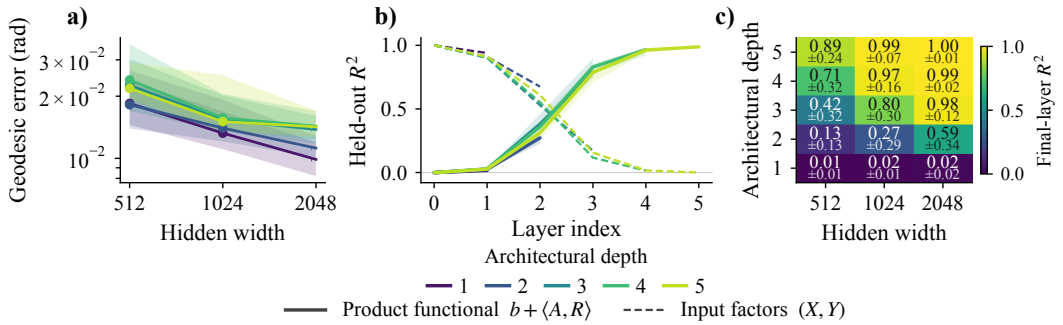

Figure 3: **Depth transforms factor representations into representations of the group product.** **(a)** Multiplication performance across network depth. **(b)** Held-out product-functional fit across hidden layers and architectural depths, compared with linear functions of the input factors. **(c)** Final-layer product-functional fit across depth and width.

4 Results

4.1 Depth produces product-functional representations

Although every network is trained on the same multiplication task, the internal representation changes systematically with depth. We first ask whether individual hidden neurons represent the two input rotations or their product. Fig. 3 shows the held-out fit of the product-functional model in Eq. 2 across hidden layers and architectures. Product-functional fit increases through depth, reaching a median R^2 of 0.988 in the final hidden layers of the deepest networks, compared with 0.017 in the corresponding layers of the shallowest networks. Over the same architectural progression, mean geodesic multiplication error worsens from 0.0175 to 0.0246.

The transition is specific to the group product. Linear functionals of $R = XY$ explain substantially more held-out activation variance in deeper layers than linear functions of X, Y, or the two factors separately. At the final hidden layer of the deepest networks ($d = 5$), the median product-functional R^2 is 0.988, compared with -0.00023 for X, -0.00022 for Y, and -0.00029 for (X, Y). The corresponding shuffled-product and independent-product controls reach -0.00121 and -0.00122. Expanded controls and the complete factorial breakdown appear in App. C.1.

These results establish the first stage of the learned computation: increasing depth reorganizes hidden activity around XY. We next ask whether these product functionals possess the internal structure predicted analytically by Rodrigues' formula.

4.2 Depth produces Rodrigues-structured $\ell = 1$ and $\ell = 2$ neurons

The decomposition of a product functional predicts distinct $\ell = 1$ and $\ell = 2$ neuron families. Fig. 5 follows these families through depth using the classification in Eq. 9. The fraction of neurons classified as $\ell = 1$ changes from 0.303 in the final hidden layers of the shallowest networks ($d = 1$) to 0.323 in the deepest networks ($d = 5$), while the corresponding $\ell = 2$ fraction changes from 0.569 to 0.574. The emergence of these families therefore accompanies the transition toward product-functional representations in Fig. 3.

The independently fitted matrices A_j recover the same structure. Neurons classified as $\ell = 1$ from their Rodrigues fits concentrate a median fraction 0.769 of $\|A_j\|_F^2$ in the antisymmetric component $A_{j,1}$, while neurons classified as $\ell = 2$ concentrate 0.728 in the symmetric-traceless component $A_{j,2}$. For $\ell = 2$ neurons, the preferred axes estimated directly from neuron activations agree almost perfectly with the dominant eigendirections extracted independently from $A_{j,2}$, with median absolute alignment 0.9999. Thus the activation-based Rodrigues classification and the independently recovered product functional identify the same decomposition.

Rodrigues' formula makes a more specific prediction about the angular dependence of each family. We therefore recover $g_j(\theta)$ freely from Eq. 10, fixing only the preferred axis and angular sector. Fig. 4 shows this recovery for a representative depth-5 network at hidden layer 4: across 42 classified $\ell = 1$ neurons, the median absolute correlation with $\sin\theta$ is 0.996, while across 899 classified $\ell = 2$

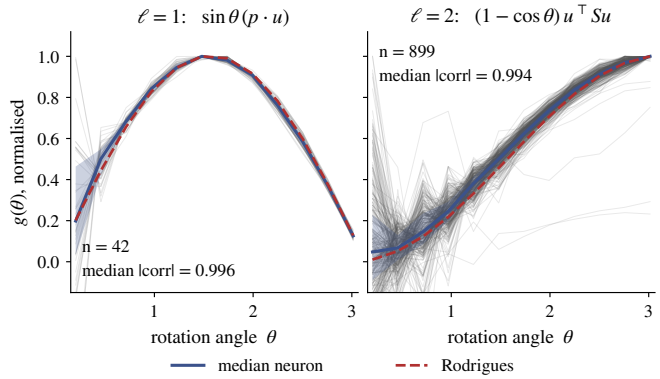

Figure 4: **Neurons recover the angular factors in Rodrigues' formula.** Freely recovered angular coefficients $g_j(\theta)$ for classified $\ell = 1$ (left) and $\ell = 2$ (right) neurons. Thin curves show individual neurons, the solid curve shows the median response, and the dashed curve shows the corresponding Rodrigues prediction. Curves are normalized and sign-aligned for comparison of shape.

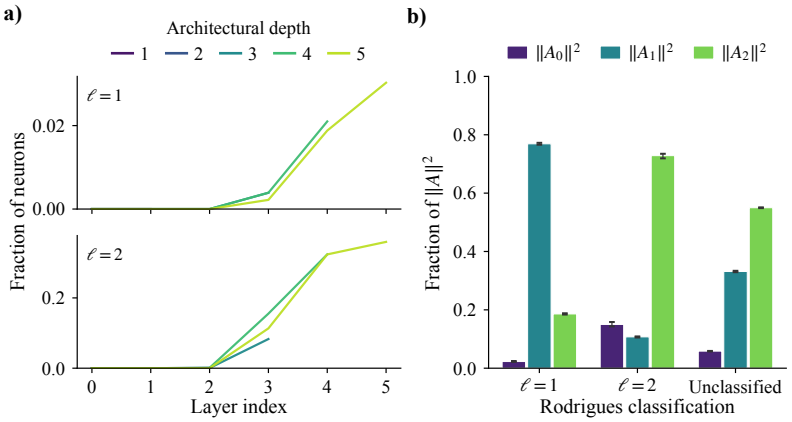

Figure 5: **Depth produces Rodrigues-structured neurons.** (a) Prevalence of Rodrigues-classified $\ell = 1$ and $\ell = 2$ neurons across hidden layers and architectural depths. (b) Independent decomposition of the fitted product functionals A_j into their $\ell = 0, 1, 2$ components. Rodrigues-classified $\ell = 1$ and $\ell = 2$ neurons concentrate their functional magnitude in the corresponding irreducible component.

neurons, the median absolute correlation with $1 - \cos\theta$ is 0.994 after sign alignment. The angular dependence predicted analytically from the product-functional decomposition therefore emerges directly from the learned neurons.

The identified neurons contribute causally to multiplication. The preceding analyses characterize correlations between individual neuron activations and the Rodrigues structure of the product. We therefore complement them with targeted neuron ablations. Fig. 6 measures task error as increasing fractions of each identified population are ablated, together with matched random populations of equal size. Ablating the $\ell = 2$ population produces the largest increase in task error and a progressively larger excess over its matched random control as the ablation fraction increases.

Depth produces population-level Rodrigues representations. If the neuron-level interpretation captures the organization of the learned representation, entire hidden layers should increasingly encode the Rodrigues terms T_1, T_2, and $T_1 + T_2$. Fig. 7 tests this prediction by linearly decoding these matrix-valued quantities from the layer activations. Held-out decoding increases through the hidden layers, reaching final-layer R^2 values of 0.99973 for T_1, 0.99973 for T_2, and 0.99973 for $T_1 + T_2$ in the deepest ($d = 5$) architectures.

The layerwise transition closely follows the neuron-level transition. Fig. 7 places product-functional strength, the prevalence of axial $\ell = 1/\ell = 2$ neurons, and population Rodrigues decoding on the same normalized layer coordinate. Product-linear neurons emerge as T_1 and T_2 become increasingly linearly represented by the population. This correspondence connects the local description of individual neurons to the representation implemented collectively by the layer.

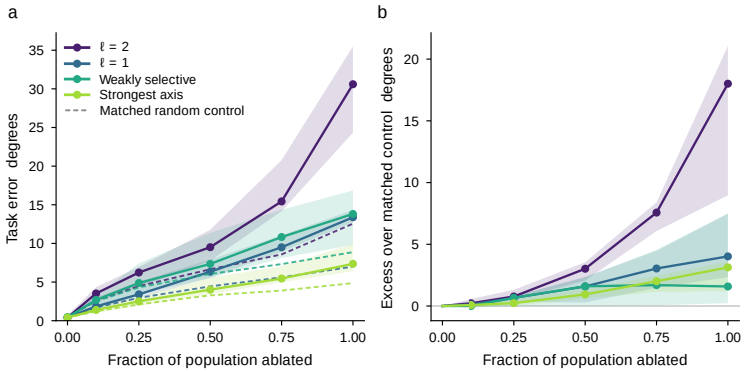

Figure 6: **Targeted ablation of Rodrigues-structured neurons degrades multiplication performance.** (a) Mean geodesic task error as increasing fractions of each neuron population are ablated; dashed curves show matched random-population controls. (b) Excess task error relative to the corresponding matched random control. Shaded regions summarize variation across runs.

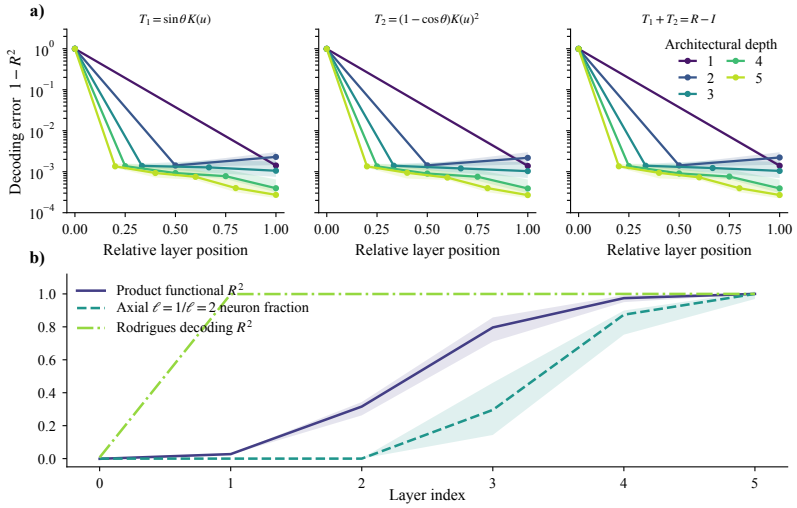

Figure 7: Rodrigues structure emerges across hidden layers. **(a)** Held-out population decoding of T_1, T_2, and $T_1 + T_2$ across hidden layers and architectural depths. **(b)** Product-functional fit, prevalence of $\ell = 1/\ell = 2$ neurons, and Rodrigues decodability aligned by normalized layer position.

Wide late-layer populations provide sufficient linear capacity to span the low-dimensional Rodrigues targets across several neuron subsets. The noteworthy result is the progression through depth and its correspondence with the emergence of product-functional and Rodrigues-structured neurons.

4.3 The continuous solution reduces to Fourier structure on finite cyclic subgroups

The product-functional interpretation makes a direct prediction when the continuous task is restricted to an embedded finite group. For a fixed axis α, consider the cyclic subgroup elements

$$X_a = R_\alpha\left(\frac{2\pi a}{m}\right), \qquad Y_b = R_\alpha\left(\frac{2\pi b}{m}\right), \qquad X_a Y_b = R_\alpha\left(\frac{2\pi(a+b)}{m}\right).$$

Here, $R_\alpha(\theta)$ is the group element given by rotation about the axis α by angle θ. m is the cardinality of the cyclic subgroup, while $a, b \in \{0, 1, \ldots, m-1\}$ Restricting a learned product functional to this subgroup yields

$h_j(a,b) \approx c_{0,j} + c_{1,j}\cos\left(\frac{2\pi(a+b)}{m}\right) + c_{2,j}\sin\left(\frac{2\pi(a+b)}{m}\right)$. The product-functional interpretation therefore predicts both the activation grid on $C_m(\alpha) \times C_m(\alpha)$ and its Fourier structure. Fig. 8 tests this prediction directly: activations predicted independently from A_j reproduce the observed diagonal modular-sum structure, while the Fourier spectrum concentrates on the predicted $p = q$ modes. Across architectures, the fraction of power on these modes rises with depth and approaches unity in the deepest networks. This connects the continuous and finite-group descriptions directly:

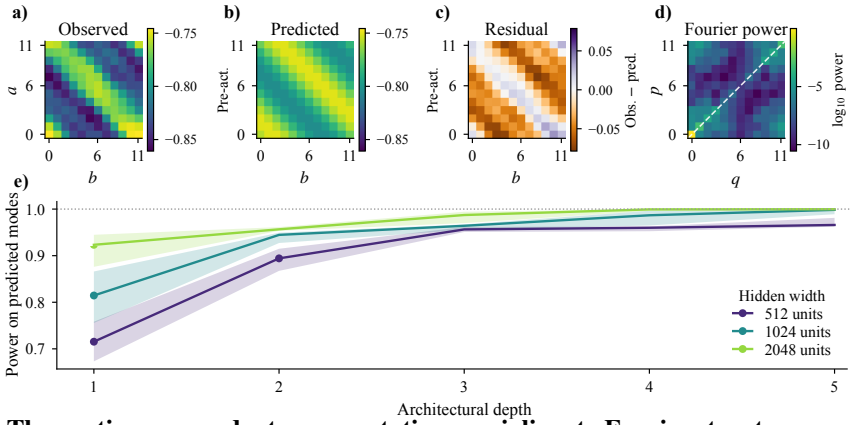

Figure 8: **The continuous product representation specializes to Fourier structure on an embedded finite cyclic group.** (a) Observed neuron activation on $C_m(\alpha) \times C_m(\alpha)$, where $C_m(\alpha)$ is the cyclic group of rotations about axis α of cardinality m. (b) Activation predicted independently from the fitted product functional A_j. (c) Residual. (d) Two-dimensional Fourier power. (e) Population fraction of Fourier power concentrated on the predicted $p = q$ modes, where p, q are discrete Fourier indices.

the same learned product functional has a Rodrigues description on $\mathrm{SO}(3)$ and, under restriction to C_m, a modular-sum Fourier description.

4.4 The learned functionals collectively span the product representation

The preceding results describe the learned solution at the level of individual neurons, complete hidden layers, and finite subgroup restrictions. We finally examine how the collection of learned functionals $\{A_j\}$ occupies the product-functional space itself. Fig. 9 shows the singular spectra and effective-rank coverage of the complete matrix space and its $\ell = 1$ and $\ell = 2$ components. Coverage remains near complete across depth, reaching 0.994 of the three-dimensional $\ell = 1$ space and 0.987 of the five-dimensional $\ell = 2$ space in the deepest ($d = 5$) architectures.

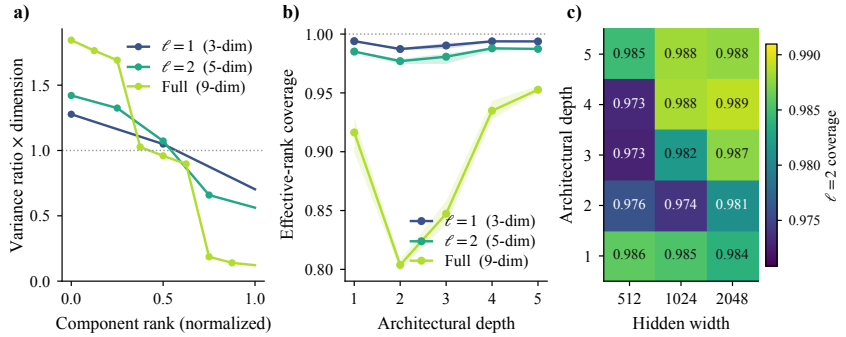

Figure 9: Learned product functionals form a distributed representation of product space. Singular spectra and normalized effective-rank coverage of the complete product-functional space and its $\ell = 1$ and $\ell = 2$ components across hidden layers and architectural depths.

Together, these experiments reveal a coherent depth-dependent transition across several levels of description. Additional layers increasingly organize hidden activity around the product XY; individual product functionals separate into the $\ell = 1$ and $\ell = 2$ structures predicted by Rodrigues' formula and recover their corresponding angular coefficients; complete hidden layers increasingly encode the Rodrigues decomposition; and restriction to embedded cyclic subgroups recovers modular-sum Fourier structure. The population of learned functionals simultaneously expands through the available product-functional subspaces. These results connect architectural depth to the emergence of a structured internal algorithm for continuous group multiplication.

In higher dimensions. We run similar analyses on DNNs trained on $SO(n)$-multiplication for $n > 3$ and observe an analogue of Rodrigues' rotation formula (Gallier, 2003) being learned. Here, DNNs are no longer rotating about an axis on a sphere. Instead, a rotation in $SO(2m), m \geq 2$ is determined by choosing m mutually orthogonal planes in $\mathbb{R}^{2m}$, and m angles to rotate those planes

by. For $SO(2m+1)$, there is a single fixed axis in addition to the m rotated planes. The $SO(4)$ Rodrigues rotation formula is:

$$R = I + \underbrace{\sin(\theta_1)F_1}_{T_1^{(4)}} + \underbrace{(1-\cos(\theta_1))F_1^2}_{T_2^{(4)}} + \underbrace{\sin(\theta_2)F_2}_{T_3^{(4)}} + \underbrace{(1-\cos(\theta_2))F_2^2}_{T_4^{(4)}}, \tag{12}$$

where F_1 and F_2 are skew-symmetric matrices determined by the orthogonal planes of rotation of R, and θ_1, θ_2 are the respective angles of rotation about those planes (Gallier, 2003).

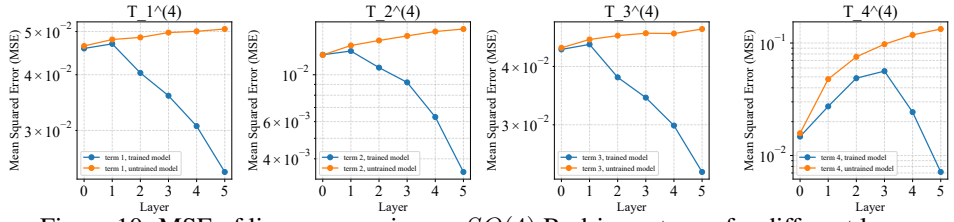

Figure 10: MSE of linear regression on $SO(4)$ Rodrigues terms for different layers

We run linear regression to decode the terms in the equation as in Appendix B.9 and find evidence that the neuronal population in the later layers linearly represent the terms Fig.10. Fig. 11 extends the product-functional picture to $\mathrm{SO}(4)$: with depth, the antisymmetric components of learned product functionals become increasingly concentrated on simple bivectors (a representation of the oriented plane spanned by two linearly independent vectors) corresponding to individual rotation planes.

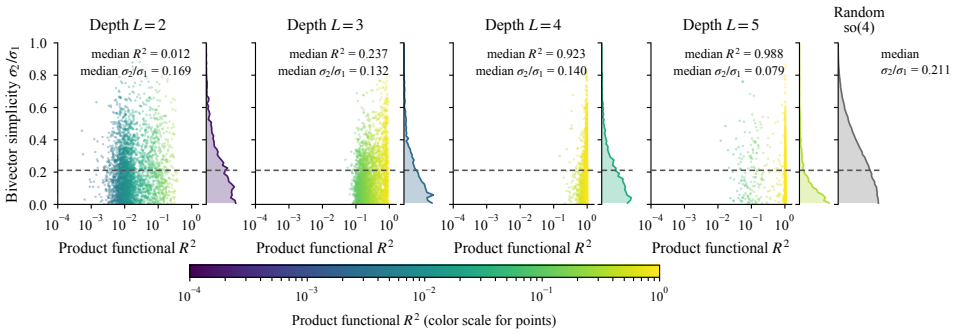

Figure 11: **SO(4) product functionals develop simple-plane bivector structure.** Bivector simplicity across hidden layers, quantified by the canonical-value ratio σ_2/σ_1, with smaller values indicating greater concentration on a single rotation plane. Learned functionals become increasingly simple-plane structured across hidden layers, separating from the random SO(4) baseline.

5 Conclusion

We present evidence DNNs learning $\mathrm{SO}(3)$ learn Rodrigues' rotation formula – an efficient algorithm to generate $\mathrm{SO}(3)$ matrices provided an axis and angle of rotation. Neurons specialize to detect product rotations about specific axes. The collective activations of neurons in the deeper layers linearly represent terms in Rodrigues' rotation formula, providing support for the divide-and-conquer conjecture on a new class and type of group: the continuous Lie group $\mathrm{SO}(n)$. **Limitations.** Our work only studies the $\mathrm{SO}(n)$ Lie group class, but we actually see that as a strength. Our goal was to provide an in-depth case study on a new type of group multiplication and describe a novel phenomenon that doesn't occur in discrete groups. We succeed, finding that depth allows for the efficient Rodrigues' rotation formula to be learned, and believe our expedition into a new class of group will inspire future work on other Lie groups and provide diversity in the choice of testing grounds for scientific hypotheses.

AI use statement

In this work, we used generative AI tools to search the literature for similar work, to create plotting scripts, to optimize training code, and to make refinements to the text. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

Ethics and Reproducibility statement

This work adheres to the ICLR Code of Ethics. In this study, no human subjects or animal experimentation were involved. All datasets used, including *the* SO(n) datasets, were sourced in compliance with relevant usage guidelines, ensuring no violation of privacy. We have taken care to avoid any biases or discriminatory outcomes in our research process. No personally identifiable information was used, and no experiments were conducted that could raise privacy or security concerns. We are committed to maintaining transparency and integrity throughout the research process.

We have made every effort to ensure that the results presented in this paper are reproducible. Code and datasets of the main training and analysis pipeline are included as anonymized supplementary materials to facilitate replication and verification. We believe these measures will enable other researchers to reproduce our work and further advance the field.

References

Hui Cheng and Krishna C Gupta. An historical note on finite rotations. 1989.

Bilal Chughtai, Lawrence Chan, and Neel Nanda. Neural networks learn representation theory: Reverse engineering how networks perform group operations. In *ICLR 2023 Workshop on Physics for Machine Learning*, 2023a.

Bilal Chughtai, Lawrence Chan, and Neel Nanda. A toy model of universality: Reverse engineering how networks learn group operations. In *International Conference on Machine Learning*, pp. 6243–6267. PMLR, 2023b.

Darshil Doshi, Aritra Das, Tianyu He, and Andrey Gromov. To grok or not to grok: Disentangling generalization and memorization on corrupted algorithmic datasets. *arXiv preprint arXiv:2310.13061*, 2023.

Jean Gallier. Computing exponentials of skew-symmetric matrices and logarithms of orthogonal matrices. *International Journal of Robotics and Automation*, 2003.

A René Geist, Jonas Frey, Mikel Zhobro, Anna Levina, and Georg Martius. Learning with 3d rotations, a hitchhiker's guide to so (3). *arXiv preprint arXiv:2404.11735*, 2024.

Andrey Gromov. Grokking modular arithmetic. *arXiv preprint arXiv:2301.02679*, 2023.

Tianyu He, Darshil Doshi, Aritra Das, and Andrey Gromov. Learning to grok: Emergence of in-context learning and skill composition in modular arithmetic tasks. *arXiv preprint arXiv:2406.02550*, 2024.

Arthur Ayestas Hilgert, Sihui Wei, Doina Precup, Gabriela Moisescu-Pareja, and Gavin McCracken. Toward a general understanding of neural representations learned by deep neural networks on group multiplications. In *Topology, Algebra, and Geometry in Data Science*.

Minyoung Huh, Brian Cheung, Tongzhou Wang, and Phillip Isola. The platonic representation hypothesis, 2024. URL https://arxiv.org/abs/2405.07987.

Yixuan Li, Jason Yosinski, Jeff Clune, Hod Lipson, and John Hopcroft. Convergent learning: Do different neural networks learn the same representations? *arXiv preprint arXiv:1511.07543*, 2015.

Ziming Liu, Ouail Kitouni, Niklas S Nolte, Eric Michaud, Max Tegmark, and Mike Williams. Towards understanding grokking: An effective theory of representation learning. *Advances in Neural Information Processing Systems*, 35:34651–34663, 2022.

Neil Mallinar, Daniel Beaglehole, Libin Zhu, Adityanarayanan Radhakrishnan, Parthe Pandit, and Mikhail Belkin. Emergence in non-neural models: grokking modular arithmetic via average gradient outer product. *arXiv preprint arXiv:2407.20199*, 2024.

D Marr and T Poggio. From understanding computation to understanding neural circuitry. *Neuroscience Research Program Bulletin*, 15(3):470–488, 1979.

Gavin McCracken, Gabriela Moisescu-Pareja, Vincent Letourneau, Doina Precup, and Jonathan Love. Uncovering a universal abstract algorithm for modular addition in neural networks, 2025. URL https://arxiv.org/abs/2505.18266.

Gavin McCracken, Arthur Ayestas Hilgert, Sihui Wei, Gabriela Moisescu-Pareja, Zhaoyue Wang, and Jonathan Love. Interpreting deep neural networks trained on elementary p groups reveals algorithmic structure. In *Topology, Algebra, and Geometry in Data Science*, 2026.

Mohamad Amin Mohamadi, Zhiyuan Li, Lei Wu, and Danica Sutherland. Grokking modular arithmetic can be explained by margin maximization. In *NeurIPS 2023 Workshop on Mathematics of Modern Machine Learning*, 2023. URL https://openreview.net/forum?id=QPMfCLnIqf.

Gabriela Moisescu-Pareja, Gavin McCracken, Harley Wiltzer, Colin Daniels, Vincent Létourneau, Doina Precup, and Jonathan Love. On the geometry and topology of representations: the manifolds of modular addition. In *International Conference on Learning Representations*, volume 2026, pp. 402–422, 2026.

Depen Morwani, Benjamin L. Edelman, Costin-Andrei Oncescu, Rosie Zhao, and Sham M. Kakade. Feature emergence via margin maximization: case studies in algebraic tasks. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=i9wDX850jR.

Neel Nanda, Lawrence Chan, Tom Lieberum, Jess Smith, and Jacob Steinhardt. Progress measures for grokking via mechanistic interpretability. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=9XFSbDPmdW.

Chris Olah, Nick Cammarata, Ludwig Schubert, Gabriel Goh, Michael Petrov, and Shan Carter. Zoom in: An introduction to circuits. *Distill*, 5(3):e00024–001, 2020.

Alethea Power, Yuri Burda, Harri Edwards, Igor Babuschkin, and Vedant Misra. Grokking: Generalization beyond overfitting on small algorithmic datasets, 2022. URL https://arxiv.org/abs/2201.02177.

Dashiell Stander, Qinan Yu, Honglu Fan, and Stella Biderman. Grokking group multiplication with cosets. In *Forty-first International Conference on Machine Learning*, 2024.

Tao Tao, Darshil Doshi, Dayal Singh Kalra, Tianyu He, and Maissam Barkeshli. (how) can transformers predict pseudo-random numbers? In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=asDx9sPAUN.

Sihui Wei, Gavin McCracken, Gabriela Moisescu-Pareja, Harley Wiltzer, Doina Precup, Irina Rish, and Jonathan Love. Deep neural networks divide and conquer dihedral multiplication. In *Forty-third International Conference on Machine Learning*.

Wilson Wu, Louis Jaburi, Jason Gross, et al. Towards a unified and verified understanding of group-operation networks. In *International Conference on Learning Representations*, volume 2025, pp. 61683–61714, 2025.

Chun Hei Yip, Rajashree Agrawal, Lawrence Chan, and Jason Gross. Modular addition without black-boxes: Compressing explanations of mlps that compute numerical integration, 2024. URL https://arxiv.org/abs/2412.03773.

Ziqian Zhong, Ziming Liu, Max Tegmark, and Jacob Andreas. The clock and the pizza: Two stories in mechanistic explanation of neural networks. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=S5wmbQc1We.

A Additional Mathematical Background

A.1 Axis–angle conventions

For a non-identity $R \in \mathrm{SO}(3)$, the rotation axis u is an eigenvector of R with eigenvalue 1. Using signed angles,

$$(u, \theta) \sim (-u, -\theta).$$

At $\theta = 0$ the axis is undefined, while at $\theta = \pi$ the axes u and $-u$ represent the same rotation. Analyses involving preferred directions therefore use sign-invariant quantities where appropriate or impose an explicit sign convention.

A.2 Preferred-axis estimation

Let h denote a neuron's response and

$$y = h - \mathbb{E}[h].$$

For an $\ell = 1$-dominated neuron, the preferred antisymmetric direction is estimated by

$$\tilde{p}_1 = \mathbb{E}[yu], \qquad p_1 = \frac{\tilde{p}_1}{\|\tilde{p}_1\|}.$$

The sign of p_1 is arbitrary and can be absorbed into the fitted coefficient.

For an $\ell = 2$-dominated neuron, define

$$Q(u) = uu^\top - \frac{1}{3} I$$

and

$$M = \mathbb{E}[yQ(u)].$$

We eigendecompose M and take the eigenvector corresponding to the eigenvalue of largest absolute magnitude as the dominant axis p_2. Its remaining sign ambiguity is fixed by convention.

Unlike the $\ell = 1$ case, p_2 is only a summary of the corresponding functional. The complete $\ell = 2$ object is the symmetric and traceless tensor S.

A.3 Irreducible representations of $\mathrm{SO}(3)$

The finite-dimensional irreducible representations of $\mathrm{SO}(3)$ are indexed by non-negative integers ℓ and have dimension $2\ell + 1$. Writing the ℓth representation as D^ℓ,

$$D^\ell(XY) = D^\ell(X) D^\ell(Y).$$

The $\ell = 0$ representation is scalar, $\ell = 1$ is the standard three-dimensional representation, and $\ell = 2$ acts on the five-dimensional space of symmetric-traceless rank-two tensors.

Two constructions used in this work should be distinguished. The product matrix $R = XY$ is the matrix of the standard $\ell = 1$ representation. Separately, the conjugation action on the ambient matrix space decomposes as

$$\mathbb{R}^{3\times 3} \cong V_0 \oplus V_1 \oplus V_2,$$

with dimensions $1 + 3 + 5$. The latter decomposition is used for the dual matrix A associated with a product-linear neuron.

Every $A \in \mathbb{R}^{3\times 3}$ decomposes uniquely and orthogonally as

$$A = A_0 + A_1 + A_2,$$

where

$$A_0 = \frac{\operatorname{tr} A}{3} I, \qquad A_1 = \frac{A - A^\top}{2}, \qquad A_2 = \frac{A + A^\top}{2} - \frac{\operatorname{tr} A}{3} I.$$

These scalar, antisymmetric, and symmetric-traceless spaces have dimensions 1, 3, and 5, respectively, and transform as the $\ell = 0, 1, 2$ irreducible components under conjugation by $\mathrm{SO}(3)$.

A.4 Irreducible representations and Rodrigues' formula

For an antisymmetric A_1, there is a vector p such that, up to scale,

$$\langle A_1, R\rangle_F \propto \sin\theta\,(p \cdot u).$$

For a symmetric-traceless $A_2 = S$,

$$\langle S, R\rangle_F = (1 - \cos\theta)u^\top S u.$$

Thus the $\ell = 1$ component is antisymmetric, with angular factor $\sin\theta$, whereas the $\ell = 2$ component is symmetric and with a trace of zero, with angular factor $1 - \cos\theta$.

In the uniaxial $\ell = 2$ case,

$$S \propto pp^\top - \frac{1}{3}I,$$

so

$$\langle S, R\rangle_F \propto (1 - \cos\theta)P_2(p \cdot u),$$

where $P_2(z) = \frac{1}{2}(3z^2 - 1)$. A general S, however, has five degrees of freedom and need not be describable by a single preferred axis.

A.5 Haar measure on $\mathrm{SO}(3)$

The compact group $\mathrm{SO}(3)$ has a unique normalized Haar probability measure invariant under left and right multiplication. Under Haar sampling, the rotation axis is uniformly distributed on S^2, while the axis–angle density satisfies

$$p(\theta) \propto 1 - \cos\theta, \qquad \theta \in [0, \pi].$$

Thus rotations close to the identity are relatively sparse, which is relevant when estimating angular tuning functions by binning in θ.

A.6 Finite cyclic subgroups

For a fixed unit axis $p \in S^2$ and integer $m \geq 1$, define

$$C_m(p) = \left\{ R\left(p, \frac{2\pi k}{m}\right) : k = 0, \ldots, m-1 \right\}.$$

This is a cyclic subgroup of $\mathrm{SO}(3)$ with multiplication

$$R\left(p, \frac{2\pi a}{m}\right) R\left(p, \frac{2\pi b}{m}\right) = R\left(p, \frac{2\pi(a+b)}{m}\right),$$

where addition is modulo m.

Restricting a trained $\mathrm{SO}(3)$ multiplication network to

$$C_m(p) \times C_m(p)$$

therefore embeds an exact finite cyclic multiplication problem within the continuous group. We use these restrictions as controlled probes of hidden responses along fixed rotation axes.

A.7 Population decoding of Rodrigues components

For a hidden-layer activation vector

$$H^{(k)}(X, Y) \in \mathbb{R}^{d_k},$$

population-level linear probes can test whether the layer spans the Rodrigues components

$$T_1 = \sin\theta[u]_\times, \qquad T_2 = (1 - \cos\theta)[u]_\times^2.$$

Specifically, we fit

$$W_1^{(k)}H^{(k)} + b_1^{(k)} \approx \mathrm{vec}(T_1),$$
$$W_2^{(k)}H^{(k)} + b_2^{(k)} \approx \mathrm{vec}(T_2),$$

and

$$W_{12}^{(k)}H^{(k)} + b_{12}^{(k)} \approx \mathrm{vec}(R - I).$$

Here, vec flattens a matrix into a vector with the same dimension. These probes concern the information linearly represented by a population and should not be interpreted as evidence that an individual neuron implements either Rodrigues term.

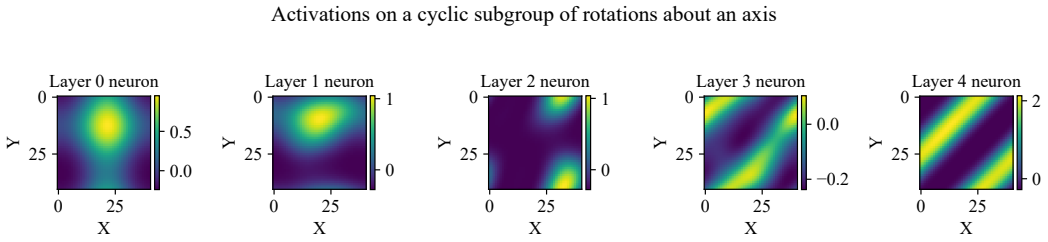

Figure 12: The emergence of product invariance shown using the activations of neurons on a finite cyclic subgroup of rotations.

B Additional Methods

B.1 Dataset construction and training

For each network seed s, training and validation examples were generated independently. Each example consists of two independent Haar-distributed rotations $X, Y \in \mathrm{SO}(3)$ and the exact matrix product $R = XY$. Training and validation data were generated once and held fixed throughout training. A nominal training fraction of 0.9 determined the training-set size, with validation examples generated independently.

For a network of width w and d intermediate hidden layers, the architecture contains an initial hidden layer mapping 18 input features to w units, followed by d hidden layers of width w and a final linear layer mapping the hidden representation to nine output coordinates. Each hidden affine transformation is followed by LayerNorm ($\epsilon = 10^{-6}$) and SiLU. Every affine layer includes a bias term. Parameters are initialized using LeCun-normal initialization.

The complete factorial sweep was

$$\begin{aligned} w &\in \{512, 1024, 2048\}, \\ d &\in \{1, 2, 3, 4, 5\}, \\ N &\in \{5000, 10000, 20000\}, \\ \eta &\in \{3 \times 10^{-5}, 10^{-4}, 3 \times 10^{-4}\}, \\ \lambda &\in \{0, 10^{-4}, 10^{-3}\}. \end{aligned}$$

All 405 configurations were trained with eight seeds, yielding 3,240 independently initialized networks.

Optimization used AdamW for 1001 epochs with batch size 128. Training proceeded for the complete 1001-epoch schedule for every run. The learning rate followed an exponential schedule with multiplicative decay 0.85 every 120 epochs. Weight decay was applied to all learned parameters.

The network's nine-dimensional output v was first normalized by its Euclidean norm and reshaped as a matrix $M \in \mathbb{R}^{3\times3}$. If $M = U\Sigma V^\top$, the predicted rotation was

$$\widehat{R} = U \ \mathrm{diag}\left(1, 1, \det(UV^\top)\right) V^\top.$$

Training minimized the mean entrywise squared error between $\widehat{R}$ and R. Rotation error was additionally measured by the geodesic angle between $\widehat{R}$ and R.

B.2 Independent analysis samples

Interpretability analyses used independent samples for parameter estimation and evaluation. For every trained network, independent Haar samples were generated for fitting and evaluation, with up to 8,000 examples used for each analysis set. Preferred axes, regression coefficients, and other fitted quantities were estimated exclusively from the fitting set. Fit quality was evaluated on the held-out evaluation set.

Analyses were applied to every hidden layer and to multiple checkpoints through training. Competing models for a given network, checkpoint, and layer used the same independently generated fit/evaluation samples unless specified otherwise.

B.3 PRODUCT-FUNCTIONAL CONTROLS

In addition to the product-functional model

$$h = b + \langle A, R \rangle_F, \qquad R = XY,$$

we fit the operand controls

$$h = b + \langle B, X \rangle_F, \qquad h = b + \langle C, Y \rangle_F,$$

and

$$h = b + \langle B, X \rangle_F + \langle C, Y \rangle_F.$$

Two null controls replace R with either a random permutation of the observed product matrices or independently generated Haar rotations. We additionally fit the $\ell = 0$, $\ell = 1$, $\ell = 2$, $\ell = 0 + 1$, $\ell = 0 + 2$, and $\ell = 1 + 2$ restrictions of the product representation.

As representation controls, we fit the degree-two representation $D^2(R)$ and the corresponding operand-side model using $D^2(X)$ and $D^2(Y)$. All ordinary product-functional and control regressions include an explicit intercept and are solved by least squares.

The identity

$$\langle A, XY \rangle_F = \sum_{i,a,j} A_{ij} X_{ia} Y_{aj}$$

relates the product-functional description directly to a bilinear function of the two operands.

B.4 RELATION BETWEEN THE MATRIX DECOMPOSITION AND RODRIGUES' FORMULA

Let

$$A = \alpha I + A_1 + S, \qquad \alpha = \frac{\mathrm{tr}(A)}{3},$$

where A_1 is antisymmetric and S is symmetric and traceless. Writing the rotation as

$$R = I + \sin\theta [u]_\times + (1 - \cos\theta)[u]_\times^2$$

expresses a product functional as scalar, antisymmetric, and symmetric-traceless contributions with angular dependencies $1 + 2\cos\theta$, $\sin\theta$, and $1 - \cos\theta$, respectively. In particular, the $\ell = 1$ contribution is proportional to

$$\sin\theta \, (a^\top u),$$

where a is the axial vector of A_1, while the $\ell = 2$ contribution is proportional to

$$(1 - \cos\theta) \, u^\top S u.$$

For the uniaxial restriction

$$S = \gamma \left(pp^\top - \frac{1}{3} I \right),$$

this becomes proportional to

$$(1 - \cos\theta) P_2(u^\top p),$$

which motivates equation 7. The Rodrigues analyses and the matrix-functional analyses therefore provide two parameterizations of the same underlying product structure.

B.5 RODRIGUES AND AXIS–ANGLE CONTROLS

For every neuron, the preferred axes p_1 and p_2 are estimated before evaluating the following models. In addition to equation 6–equation 8, we fit the axis-only controls

$$h = a + bP_1(u^\top p_1), \qquad h = a + bP_2(u^\top p_2),$$

the angle-only controls

$$h = a + b\sin\theta, \qquad h = a + b(1 - \cos\theta),$$

and the crossed controls

$$h = a + b(1 - \cos\theta)P_1(u^\top p_1),$$
$$h = a + b\sin\theta P_2(u^\top p_2).$$

The crossed controls exchange the Rodrigues angular factors associated with the $\ell = 1$ and $\ell = 2$ components.

The unrestricted $\ell = 2$ comparison is

$$h = a + \langle S, R\rangle_F, \qquad S = S^\top, \quad \mathrm{tr}(S) = 0,$$

which contains five independent $\ell = 2$ coefficients. The single-axis Rodrigues model corresponds to a uniaxial restriction of this general symmetric-traceless functional.

B.6 FLEXIBLE ANGULAR RECOVERY

For equation 10, the observed fitting-set range of θ was divided into 12 equal-width bins. The coefficient $g_j(\theta)$ was taken to be constant within each bin and jointly fitted with the intercept. Evaluation samples were assigned using the bin boundaries determined from the fitting set. Empty fitting bins were assigned coefficient zero.

For $\ell = 1$ neurons, the recovered coefficients were compared with $\sin\theta$ evaluated at the bin centers; for $\ell = 2$ neurons they were compared with $1 - \cos\theta$. We report

$$\frac{|\langle g, g^\star\rangle|}{\|g\|\,\|g^\star\|}$$

and the held-out gain

$$\Delta R^2 = R^2_{\text{flex}} - R^2_{\text{Rodrigues}}.$$

The absolute cosine accounts for the arbitrary overall sign of the preferred-axis parameterization.

B.7 TWO-DIMENSIONAL LEGENDRE ANALYSIS

For a preferred axis p, we define

$$x = u_X^\top p, \qquad y = u_Y^\top p,$$

where u_X and u_Y are the rotation axes of the two operands. We fit equation 11 directly to individual fitting samples through degree six in each variable, yielding 49 product-Legendre terms. Coefficients are evaluated on independent samples.

To compare contributions across degrees, the power associated with coefficient $c_{\ell m}$ is

$$\mathcal{P}_{\ell m} = \frac{c^2_{\ell m}}{(2\ell + 1)(2m + 1)}.$$

Nonconstant power fractions are computed over components with $(\ell, m) \neq (0, 0)$. We distinguish operand-only terms ($\ell = 0$ or $m = 0$) from interaction terms ($\ell > 0$ and $m > 0$), and separately quantify low-order ($\ell, m < 3$) and higher-order contributions.

For visualization, empirical activations are binned over the (x, y) plane. The same samples are used to construct the prediction $b + \langle A, XY\rangle_F$ and its residual. Visualization retains empty bins as undefined entries.

B.8 Population geometry

Each fitted A_j is represented in orthonormal coordinates of the full nine-dimensional matrix space and separately in its one-, three-, and five-dimensional $\ell = 0, 1, 2$ components. For the resulting neuron-by-coordinate matrix V, let σ_i denote its singular values. We define

$$p_i = \frac{\sigma_i}{\sum_k \sigma_k}, \qquad r_{\text{eff}} = \exp\left(-\sum_i p_i \log p_i\right).$$

Subspace coverage is r_{eff} divided by the dimension of the corresponding subspace.

B.9 Population Rodrigues decoding

To distinguish single-neuron tuning from information available linearly at the population level, we regress the matrix-valued targets

$$T_1 = \sin\theta[u]_\times, \qquad T_2 = (1-\cos\theta)[u]_\times^2, \qquad R - I = T_1 + T_2$$

on the complete hidden-layer activation vector. The regressions are repeated using the complete population, neurons classified as $\ell = 1$, neurons classified as $\ell = 2$, their union, and random subsets matched to the number of classified neurons. This analysis characterizes population-level linear decodability, while the neuron-wise Rodrigues fits define the families in equation 9.

B.10 Numerical validation

The analysis pipeline includes numerical validation of the identities and invariances used throughout the interpretation. These checks cover reconstruction of $R = XY$; Rodrigues reconstruction; orthogonality and reconstruction of $A = A_0 + A_1 + A_2$; the bilinear identity $\langle A, XY\rangle_F$; orthonormality of the representation bases; agreement between the uniaxial symmetric-traceless and single-axis $\ell = 2$ forms; and strict separation of fitting and evaluation samples.

Axis–angle extraction near $\theta = \pi$ uses the corresponding near-π recovery branch for numerical stability. Population probes use a consistent target parameterization within each comparison, including a consistent treatment of the regression intercept.

C Additional Results

C.1 Controls for product-functional specificity

The main-text product-functional analysis is expanded here using the complete set of controls from Appendix B.3. Figure 13 compares linear functionals of R, X, Y, and (X, Y) with the shuffled-product, independent-product, and degree-two representation controls. The comparison is shown across hidden layers and architectural depths so that the emergence of product specificity can be followed through the complete network.

C.2 Independent agreement between Rodrigues families and the decomposition of A

The family assignment in equation 9 and the matrix decomposition in equation 3 are estimated through separate analyses. Figure 14 shows their correspondence at the neuron level. The $\ell = 1$ and $\ell = 2$ functional norm fractions are reported conditionally on Rodrigues family, together with the complete distribution across the decomposition simplex.

For $\ell = 2$ neurons, we additionally compare the preferred axis obtained from Eq. 5 with the dominant eigendirection of the symmetric-traceless component $A_{j,2}$. Their agreement is summarized by the absolute axis inner product. Biaxiality is shown alongside this comparison to characterize the progression from approximately uniaxial neurons to general five-dimensional $\ell = 2$ functionals.

Figure 15 reports the complete flexible angular recovery from equation 10. Recovered $g_j(\theta)$ curves are aggregated across classified neurons, with distributions of angular-shape similarity and ΔR^2 reported separately for $\ell = 1$ and $\ell = 2$ families and stratified by layer and depth.

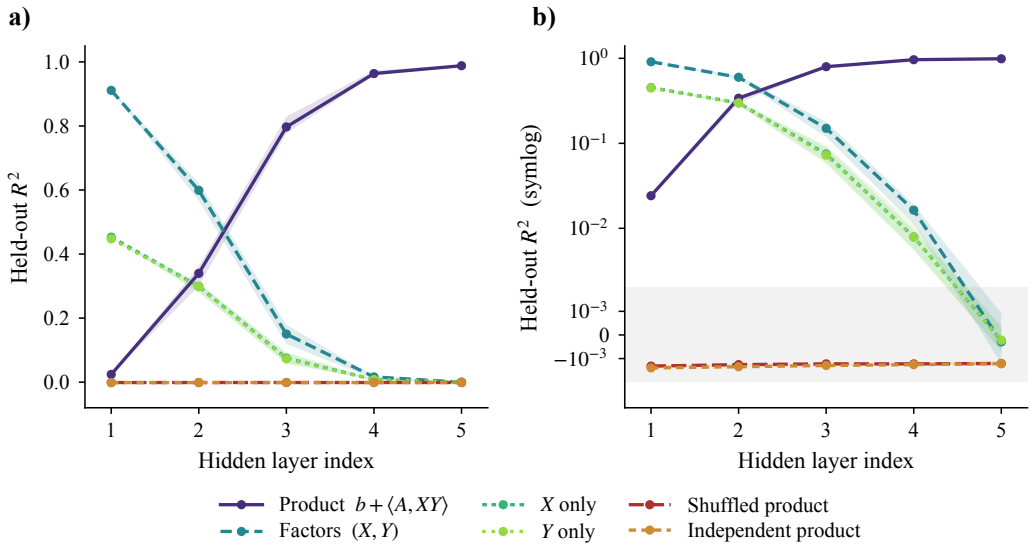

Figure 13: Product-functional specificity across hidden layers. Held-out fits are shown for the product, operand, shuffled-product, independent-product, and representation-control models described in Appendix B.3.

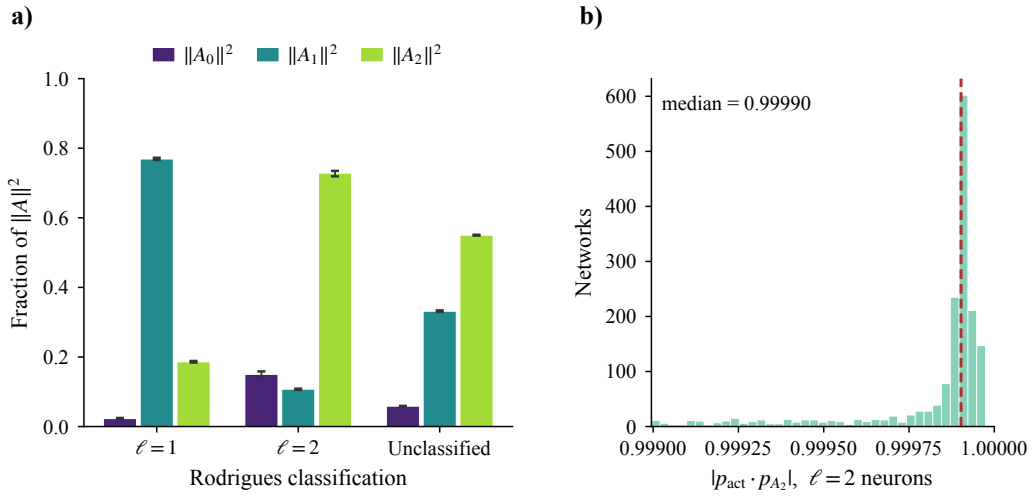

Figure 14: Agreement between the independently estimated Rodrigues families and product-functional decomposition. Panels show $\ell = 1/\ell = 2$ functional norm fractions conditioned on neuron family, the decomposition simplex, preferred-axis agreement, and $\ell = 2$ biaxiality.

C.3 Robustness across architecture and optimization settings

Figure 16 stratifies the principal measurements by every axis of the factorial sweep. The central depth-dependent progression appears across widths, dataset sizes, learning rates, and weight decays. The panels report task performance, product-functional fit, $\ell = 1/\ell = 2$ family prevalence, functional-space composition, and population Rodrigues decoding, with the eight seeds associated with each configuration providing the between-run variation.

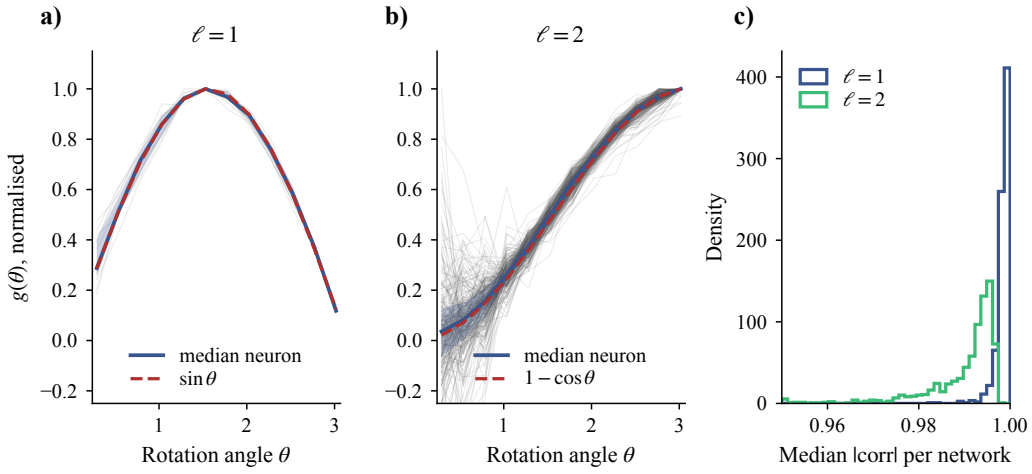

Figure 15: Flexible recovery of the angular coefficient. Recovered $g_j(\theta)$ curves are shown for $\ell = 1$ and $\ell = 2$ neurons together with $\sin\theta$ and $1 - \cos\theta$, respectively. Additional panels report absolute cosine similarity and the held-out change in fit associated with the flexible angular model.

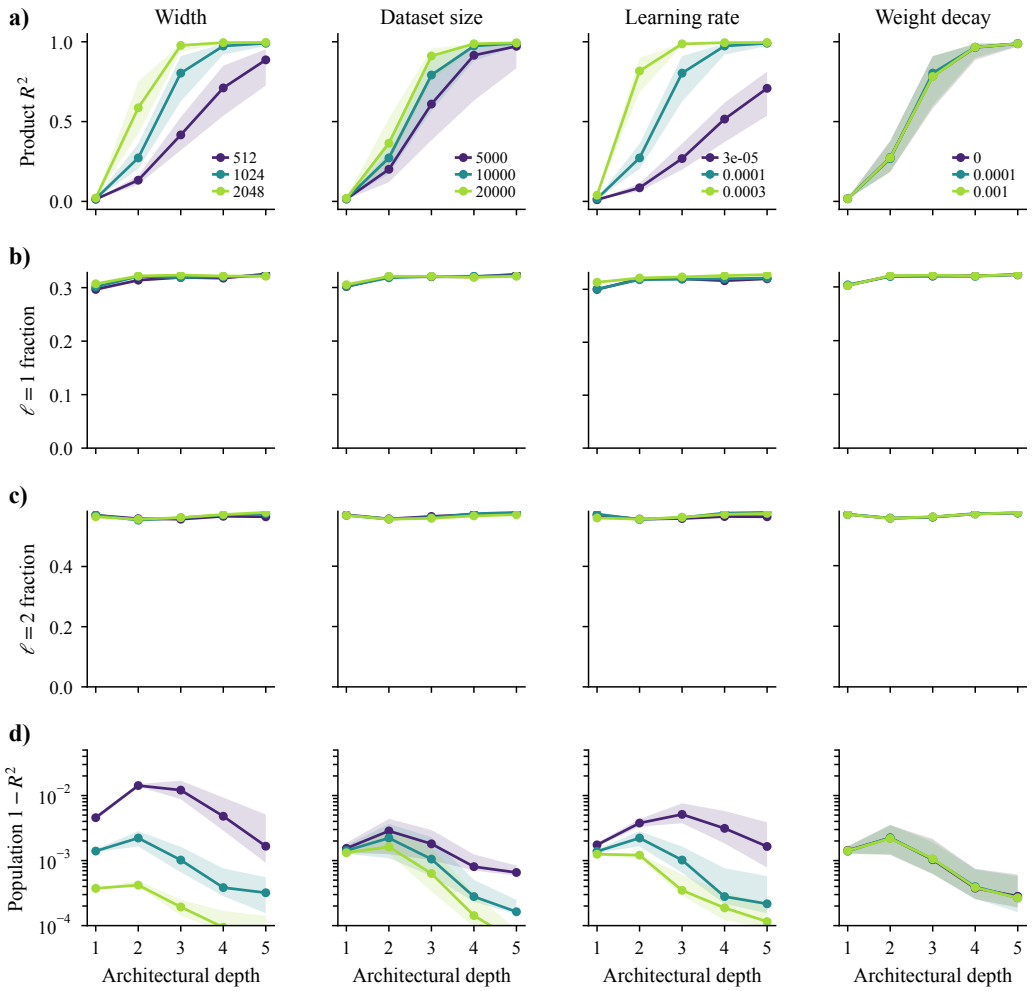

Figure 16: Principal measurements across the factorial sweep. Results are stratified by depth, width, dataset size, learning rate, and weight decay, with variation across eight independently initialized networks summarized within each configuration.