The Role of Symmetry in Optimizing Overparameterized Networks

Kusha Sareen [1 2] **Mohammad Pedramfar** [1 2] **Sékou-Oumar Kaba** [1 2] **Mehran Shakerinava** [1 2]
Siamak Ravanbakhsh [1 2]

Abstract

Overparameterization is central to the success of deep learning, yet the mechanisms by which it improves optimization remain incompletely understood. We analyze weight-space symmetries in neural networks and show that overparameterization introduces additional symmetries that benefit optimization in two distinct ways. First, we prove that these symmetries act as a form of diagonal preconditioning on the Hessian, enabling the existence of better-conditioned minima within each equivalence class of functionally identical solutions. Second, we show that overparameterization increases the probability mass of global minima near typical initializations, making these favourable solutions more reachable. Empirically, we observe wider networks have lower top eigenvalues, smaller condition numbers and faster convergence, matching our analysis. Our analysis provides a unified framework for understanding overparameterization and width growth as a geometric transformation of the loss landscape.

1 Introduction

Overparameterized neural networks consistently outperform their smaller counterparts in optimization speed and final performance, even when the target function could be represented by much smaller architectures (Arora et al., 2018; Yang et al., 2022). A major line of theory explains this via large-width limits and linearization (Jacot et al., 2018; Allen-Zhu et al., 2019; Du et al., 2019; Zou et al., 2019; Chizat et al., 2019). Empirical studies of curvature complement this picture, showing that overparameterized networks have highly structured Hessian spectra with large near-zero subspaces and few data-dependent outliers (Sagun et al., 2017; Ghorbani et al., 2019).

[1]Mila - Quebec AI Institute, Montreal, Canada [2]Department of Computer Science, McGill University, Montreal, Canada. Correspondence to: Kusha Sareen <kusha.sareen@mila.quebec>.

Proceedings of the 43rd International Conference on Machine Learning, Seoul, South Korea. PMLR 306, 2026. Copyright 2026 by the author(s).

The loss landscape is also shaped by *symmetry*: neural networks possess large groups of function-preserving reparameterizations (permutations, rescalings, and more structured invariances) that induce equivalence classes of weights and manifolds of functionally identical minima (Simsek et al., 2021; Kunin et al., 2021; Grigsby et al., 2023; Brea et al., 2019; Martinelli et al., 2024). These symmetries complicate naive geometric notions such as "sharpness" (Dinh et al., 2017) and help explain why seemingly distinct solutions can be connected by low-loss paths (Garipov et al., 2018; Draxler et al., 2018).

This paper sits at the intersection of these two perspectives. We provide a more comprehensive characterization of the weight-space symmetries due to overparameterization, building on Fukumizu & Amari (2000); Simsek et al. (2021), and show they aid optimization through two mechanisms:

Mechanism 1: Diagonal Preconditioning. The eigenvalues of the Hessian H after a symmetry transformation are the same as those for $K^{1/2}HK^{1/2}$, where K is a diagonal matrix that depends on the symmetry transformations. The right choice of K can improve the Hessian conditioning, especially if eigenvectors are approximately axis-aligned.

Mechanism 2: Volume Growth. Overparameterization increases the total *volume* of global minima near typical initializations; specifically, the probability of initializing within ϵ of a minimum grows with width once the network is sufficiently wide. Adding parameters introduces new symmetries that expand the equivalence class, making well-conditioned, reachable minima more abundant.

Contributions.

- We characterize overparameterization symmetries via a matrix groupoid building on Simsek et al. (2021), and identify a new pre-activation splitting symmetry for ReLU networks (Sections 3 and 4).
- We prove these symmetries act as diagonal preconditioning on the Hessian, with explicit formulas for how gradients and eigenvalues transform. Under even splits, the product of nonzero eigenvalues is preserved while the trace can decrease, giving minima with better average-case conditioning (Sections 5 and 7).

- We show the probability of initializing within ϵ of a global minimum grows with width once the network is sufficiently wide (Section 8).
- Experiments on teacher-student MLPs, CNNs and Transformers, and on direct data fitting, confirm the predicted improvements in conditioning and convergence (Section 9).

2 Preliminaries

Parameter Space and Loss. Fix a depth-L feedforward architecture with layer widths $n_0 = d_{\text{in}}, n = n_1, \ldots, n_{L-1}, n_L = d_{\text{out}}$. Let Θ denote the corresponding parameter space (weights and biases). For $\theta \in \Theta$, write the realized function as $f_\theta : \mathbb{R}^{d_{\text{in}}} \to \mathbb{R}^{d_{\text{out}}}$. Given a ground-truth function f^* and an input distribution $\mathcal{D}$, we consider the population risk $\mathcal{L}(\theta) = \mathbb{E}_{x\sim\mathcal{D}}\big[\ell\big(f_\theta(x), f^*(x)\big)\big]$. Our focus is on *geometric* properties of $\mathcal{L}$ in parameter space near minima: equivalence classes of functionally identical parameters, their induced flat directions, and the curvature in directions transverse to these classes.

Functional Equivalence and Symmetry Orbits. We distinguish *functional equivalence* from *equivalence under a chosen symmetry family*. Define $\theta \sim_f \theta'$ if $f_\theta = f_{\theta'}$ (equality as functions), and define $\theta \sim_{\mathcal{T}} \theta'$ if there exists $g \in \mathcal{T}$ with $g(\theta) = \theta'$, where

$$\mathcal{T} = \{g : \Theta \to \Theta \mid f_\theta(x) = f_{g(\theta)}(x)\ \forall x, \theta\}. \tag{1}$$

For $\sim_{\mathcal{T}}$ to be an equivalence relation, we need $\mathrm{Id} \in \mathcal{T}$ and $g \in \mathcal{T} \iff g^{-1} \in \mathcal{T}$; transitivity follows from composition. When $\mathcal{T}$ contains all function-preserving transformations, $[\theta]_{\mathcal{T}} = [\theta]_f$; otherwise $[\theta]_{\mathcal{T}} \subset [\theta]_f$. To ensure closure, we define $\mathcal{T}$ through families of generators.

Realizability. We operate in a realizable setting, assuming the target function f^* can be exactly represented by a neural network of some minimal (functional) width n_f. To make this setup mathematically explicit, we often adopt the terminology of a *teacher-student* framework. A *teacher* parameter vector $\theta^* \in \Theta_{n_f}$ defines the minimal target function $f^* = f_{\theta^*}$, while a *student* network of width n is trained to match it. Importantly, this is not a restrictive assumption: universal approximation theorems imply that *any* target function can be represented by a network of sufficient minimal width.

3 Characterizing Overparameterization Symmetries

Following Simsek et al. (2021), we say a network is *overparameterized* if it has greater width than required to represent a given function. Prior work describes the overparameterized solution set primarily for *single-hidden-layer* networks and analyzes how permutation symmetries produce manifolds of minima and symmetry-induced critical points (Simsek et al., 2021). Here, we extend these results with a useful matrix viewpoint and explain what changes with depth and choice of activation. Throughout, a function-preserving symmetry is a matrix M acting on a layer such that, for a given activation σ, $M^{-1}\sigma(Mx) = \sigma(x)\,\forall x$ (e.g., M a permutation matrix, or for ReLU rescale symmetry a positive diagonal matrix). Fixed-width symmetries and the deep linear case are reviewed in Appendix O.

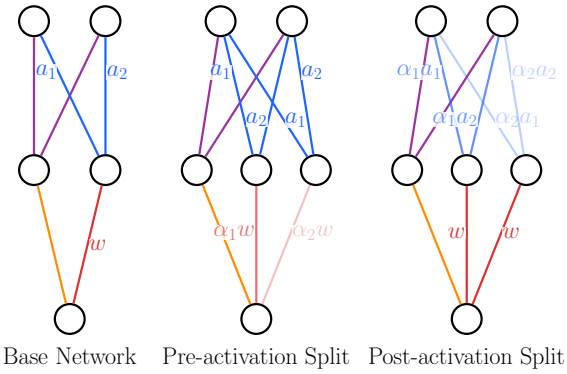

Figure 1. A neural network before and after splitting operations.

Post-activation splitting. For an arbitrary activation σ, the matrices M become highly restricted since one cannot arbitrarily mix hidden units while preserving $f_\theta(x)$ for all x. Since we make *no assumptions* about σ, the only function-preserving operations are to (1) *copy* the input and split the outgoing weights: duplicating a neuron with inputs $\vec{w}$ and outputs $\vec{a}$ into neurons $(\vec{w}, \vec{w})$ with outputs $(\vec{a}', \vec{a}'')$ satisfying $a'_i + a''_i = a_i$; or (2) add *0-type* neurons whose input weights are arbitrary and output weights sum to $\vec{0}$.

We characterize the splitting operation through a class of *duplication matrices*. We allow expansion by a matrix $M \in \mathbb{R}^{r\times n}$ for $r > n$, where M repeats rows $\vec{e}_i$ of I_n with multiplicity m_i:

$$M = \begin{bmatrix} \mathbf{1}_{m_1} \otimes \vec{e}_1^\top \\ \vdots \\ \mathbf{1}_{m_n} \otimes \vec{e}_n^\top \end{bmatrix}, \tag{2}$$

where $\mathbf{1}_{m_i}$ is an m_i-dimensional vector of ones, $\otimes$ is the Kronecker product, and the total expansion is $r = \sum_i m_i$. The contraction M^+ can be *any* pseudoinverse with $M^+M = I_n$; the freedom in M^+ corresponds to redistributing outgoing weights. Our symmetry transforms $W_l \to MW_l$ and $W_{l+1} \to W_{l+1}M^+$.

For ReLU and other piecewise-linear activations we identify an additional family of *pre-activation splitting* sym-

metries that are new to this work. The full construction, together with a strictly richer per-weight refinement of post-activation splitting and the corresponding inverse transformation, is deferred to Appendix O.

Example: Duplication and Pseudoinverse

For $M = \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}$ (duplicating the first neuron), we can have several pseudoinverses M^+:

$$M^+ = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad \begin{bmatrix} 0.9 & 0.1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad \text{etc.}$$

The freedom in M^+ corresponds to how we split the output weights.

4 The Overparameterization Groupoid

Composing splitting transformations across widths gives a *groupoid* rather than a group. Composition is only partially defined, since a transformation $\tau_{a\to b}$ can only follow another that ends at width a, and inverses do not contract below the minimal functional width n_f. Let $\mathcal{O}^L_{n_f}$ denote the resulting groupoid for L-layer networks with minimum width n_f, so that $f_\theta = f_{\theta'}$ for all $\theta' \in \theta \cdot \mathcal{O}^L_{n_f}$. Figure 15 illustrates the structure: each column represents weights with a fixed minimum width, and arrows connect equivalent parameterizations across widths. A formal characterization, including the inverse operations and partial composition, is given in Appendix G.

5 Optimization Properties

We now analyze how overparameterization symmetries affect optimization. Our goal is to show that there exist minima with favourable properties and understand the action of width growth on the loss landscape geometry.

Splitting transformations act linearly on gradients as $\nabla L_{\theta'} = B^\top \nabla L_\theta$ (Theorem P.1), where B is the rectangular block-diagonal matrix built from the per-layer expansions $\{M_l, M_l^+\}$:

$$B^\top = \operatorname{diag}\big(I \otimes (M_1^+)^\top, \ldots, M_{L-1} \otimes I\big). \tag{3}$$

Differentiating gives the Hessian transformation at a global minimum gives $H' = B^\top H B$.

5.1 Overparameterization as Diagonal Preconditioning

Theorem 5.1. *Overparameterization acts as diagonal preconditioning: the transformed Hessian H' has the same nonzero eigenvalues as HK, equivalently $K^{1/2}HK^{1/2}$, where $K = BB^\top$ is a diagonal matrix determined by the splitting structure.*

For a single split weight and its duplicated weights, $K = BB^\top$ is diagonal: split weights are scaled by m, duplicated weights with proportions $\alpha_1, \ldots, \alpha_m$ are scaled by $\sum_i \alpha_i^2$, and other weights are unchanged. The proof follows via Sylvester's identity.

6 Local Volume and Pseudo-Determinant

The ϵ-sublevel set of a minimum θ^* is locally an ellipsoid with volume $\propto (\det H)^{-1/2}$. Since splitting introduces flat symmetry directions, H' is singular, so we measure volume *modulo symmetry directions* via the pseudo-determinant $\operatorname{pdet}(H') = \prod_{i:\lambda_i>0} \lambda_i$. By Theorem 5.1, the nonzero eigenvalues of H' are exactly those of HK, so

$$\operatorname{pdet}(H') = \det(HK) = \operatorname{pdet}(H)\det(K).$$

Theorem 6.1 (Pseudo-determinant Under Splitting)**.** *Let H' be obtained from a full-rank Hessian H by an m-way split with coefficients $\alpha_1, \ldots, \alpha_m$ summing to 1, alongside the necessary duplication of subsequent layers. Then $\operatorname{pdet}(H') = m \sum_i \alpha_i^2 \cdot \operatorname{pdet}(H) \geq \operatorname{pdet}(H)$, with equality iff $\alpha_i = 1/m$ for all i.*

7 Creating Better-Conditioned Minima

The connection of symmetry to diagonal preconditioning in Section 5.1 is significant due its impact on condition numbers, which govern analysis around rates of convergence, and is a general case that applies even to the well-studied case of rescaling in ReLU network.

Classical work (Sluis, 1970) shows diagonal preconditioning, though not guaranteed to improve the worst-case condition number $\kappa(H) = \lambda_{\max}/\lambda^+_{\min}$, is particularly effective when Hessian eigenvectors are approximately axis-aligned (Figure 16).We find empirically that eigenvectors of minima converged to by standard optimizers are more axis aligned than random and this is amplified in in the largest eigenvalues (Appendix J).

Additionally, average-case conditioning has been proposed as a more robust measure of convergence by considering the entire Hessian spectrum rather than two outliers. This is captured by $\omega(H) = (\operatorname{Tr}(H)/p)/(\operatorname{pdet} H)^{1/p}$ for p positive eigenvalues (Doan & Wolkowicz, 2011; Jung et al., 2024). We show this quantity also improves. The denominator is fixed under even splits (Theorem 6.1) while the trace transforms as

Theorem 7.1 (Trace Formula)**.** *For an order-m split with split set S and duplicated set D, $\operatorname{Tr}(H') = \operatorname{Tr}(H_{\setminus S,D}) + m\sum_{s\in S} H_{ss} + \frac{1}{m}\sum_{d\in D} H_{dd}$.*

The trace decreases when $\sum_s H_{ss} < \frac{1}{m}\sum_d H_{dd}$, with optimum at $m^* = \sqrt{\sum_d H_{dd}/\sum_s H_{ss}}$.

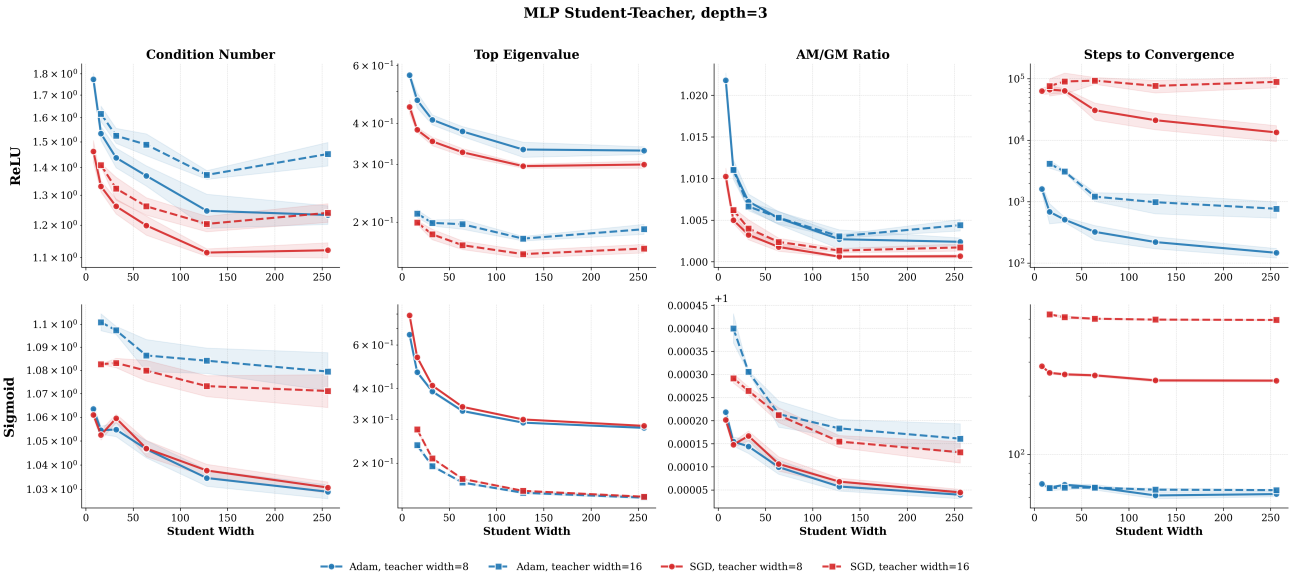

Figure 2. MLP student-teacher experiments (depth 3). Students use ReLU (top rows) and sigmoid (bottom rows), trained with Adam and SGD. Condition number, top eigenvalue, AM/GM ratio, and steps to convergence all improve monotonically with width.

8 Probability of Initializing Near Minima

For a network of width n expanded to width $n+1$ under Xavier initialization ($b_m = 1/\sqrt{m}$), the probability of initializing within ϵ of a global minimum satisfies

$$\frac{P_{n+1}}{P_n} \geq \frac{\sqrt{2}\,\ln(1+\sqrt{2})}{8} \|w^{\text{out}}\|_1 (n+1)^2 \left(\frac{n+1}{n}\right)^n (\epsilon/C)^{1/4}, \quad (4)$$

where $\|w^{\text{out}}\|_1 = \sum_{j=1}^{n} |w_j^{\text{out}}|$ is the ℓ_1 norm of the teacher's outgoing weights and $C > 0$ bounds the fourth-order coefficient of the loss along the locally flat direction introduced by splitting. The bound is derived for $d_{\text{in}} = d_{\text{out}} = 1$ in Appendix M, with the higher-dimensional case following the same template. The right-hand side exceeds 1 once $\|w^{\text{out}}\|_1 (n+1)^2 \gtrsim \epsilon^{-1/4}$, so overparameterization makes global minima strictly more reachable for sufficiently wide networks.

9 Experiments

We support our theory with three families of experiments: small-network theory validation (L-BFGS to zero loss with a specialized activation from Simsek et al. (2021)), student-teacher experiments across MLP, CNN, and Transformer architectures, and direct data fitting without a teacher. Figure 2 shows MLP student-teacher results: condition number, top eigenvalue, AM/GM ratio ω, and steps to convergence all improve monotonically with width, with the same trends across depths and on CNNs/Transformers. Our experiments on small networks match the predicted Hessian to within 0.03% in $\mathrm{Tr}(H)$, $\mathrm{pdet}(H)$, and ω. Direct data-fitting on California Housing and CIFAR-10 reproduces the conditioning improvement without a teacher suggests the result may also be relevant in mildly underparameterized regimes. Details, additional architectures, Hessian comparisons and plots are in Appendix.

10 Discussion

10.1 Implications for Optimization

Our analysis is an existence result: we characterize which minima exist and their geometric properties, rather than proving which minima gradient descent converges to. In practical optimization, early stopping within a well-conditioned basin suffices and overparameterization makes such basins more abundant. Poorly-conditioned minima with small volume may also exist, but they are both harder to reach and harder to converge to, so optimization algorithms tend to naturally bypass them. Stochastic gradient noise provides additional selection pressure, destabilizing convergence in sharp, poorly-conditioned basins.

10.2 Implications for Generalization

Our framework may also shed light on generalization. Consider noisy training data $\{x_i, y_i\}$ generated by some **simple** ground truth function f with noise: $y_i = f(x_i) + \zeta$. The width required to overfit on $\{x_i, y_i\}$ is typically much larger than the width needed to represent f. As a result, the global minima corresponding to f have larger symmetry orbits and greater volume than those corresponding to overfitting solutions. This points to a counting argument for why overparameterized networks generalize: minima are more numerous and easier to optimize. In future work, we hope to extend this to underparameterized networks through low-width subnetworks of the loss landscape (Appendix R).

References

Ainsworth, S. K., Hayase, J., and Srinivasa, S. S. Git rebasin: Merging models modulo permutation symmetries. In *International Conference on Learning Representations*, 2023.

Allen-Zhu, Z., Li, Y., and Song, Z. A convergence theory for deep learning via over-parameterization. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 242–252. PMLR, 2019.

Arora, S., Cohen, N., and Hazan, E. On the optimization of deep networks: Implicit acceleration by overparameterization. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 244–253. PMLR, 2018.

Brea, J., Simsek, B., Illing, B., and Gerstner, W. Weight-space symmetry in deep networks gives rise to permutation saddles, connected by equal-loss valleys across the loss landscape, 2019.

Chizat, L., Oyallon, E., and Bach, F. On lazy training in differentiable programming. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

Collobert, R. *Large Scale Machine Learning*. PhD thesis, Université de Paris VI, 2004.

Cooper, Y. Global minima of overparameterized neural networks. *SIAM Journal on Mathematics of Data Science*, 3(2):676–691, 2021.

Dinh, L., Pascanu, R., Bengio, S., and Bengio, Y. Sharp minima can generalize for deep nets, 2017.

Doan, X. and Wolkowicz, H. Numerical computations and the $\emptyset mega$-condition number. 01 2011.

Draxler, F., Veschgini, K., Salmhofer, M., and Hamprecht, F. A. Essentially no barriers in neural network energy landscape. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 1309–1318. PMLR, 2018.

Du, S. S., Lee, J. D., Li, H., Wang, L., and Zhai, X. Gradient descent finds global minima of deep neural networks. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 1675–1685. PMLR, 2019.

Entezari, R., Sedghi, H., Saukh, O., and Neyshabur, B. The role of permutation invariance in linear mode connectivity of neural networks. In *International Conference on Learning Representations*, 2022.

Forsythe, G. E. and Straus, E. G. On best conditioned matrices. *Proceedings of the American Mathematical Society*, 6(3):340–345, 1955.

Fukumizu, K. and Amari, S. Local minima and plateaus in hierarchical structures of multilayer perceptrons. *Neural Networks*, 13(3):317–327, 2000. ISSN 0893-6080. doi: https://doi.org/10.1016/S0893-6080(00)00009-5. URL https://www.sciencedirect.com/science/article/pii/S0893608000000095.

Garipov, T., Izmailov, P., Podoprikhin, D., Vetrov, D., and Wilson, A. G. Loss surfaces, mode connectivity, and fast ensembling of DNNs. In *Advances in Neural Information Processing Systems*, volume 31, 2018.

Ghorbani, B., Krishnan, S., and Xiao, Y. An investigation into neural net optimization via hessian eigenvalue density, 2019.

Ghosh, A., Kwon, S. M., Wang, R., Ravishankar, S., and Qu, Q. Learning dynamics of deep linear networks beyond the edge of stability, 2025.

Grigsby, J. E., Lindsey, K., and Rolnick, D. Hidden symmetries of relu networks. In *International Conference on Machine Learning*, pp. 11734–11760. PMLR, 2023.

Guille-Escuret, C., Naganuma, H., Fatras, K., and Mitliagkas, I. No wrong turns: The simple geometry of neural networks optimization paths, 2023.

Hochreiter, S. and Schmidhuber, J. Flat minima. *Neural Computation*, 9(1):1–42, 1997. doi: 10.1162/neco.1997.9.1.1.

Huang, L., Liu, X., Lang, B., and Li, B. Projection based weight normalization for deep neural networks, 2017.

Jacot, A., Gabriel, F., and Hongler, C. Neural tangent kernel: Convergence and generalization in neural networks. In *Advances in Neural Information Processing Systems*, volume 31, 2018.

Joudaki, A., Lanzillotta, G., Razlighi, M. S., Mirzadeh, I., Alizadeh, K., Hofmann, T., Farajtabar, M., and Faghri, F. Barriers for learning in an evolving world: Mathematical understanding of loss of plasticity, 2025. URL https://arxiv.org/abs/2510.00304.

Jung, W. L., Torregrosa-Belén, D., and Wolkowicz, H. The ω-condition number: Applications to optimal preconditioning and low rank generalized jacobian updating, 2024. URL https://arxiv.org/abs/2308.13195.

Kunin, D., Sagastuy-Breña, J., Ganguli, S., Yamins, D. L., and Tanaka, H. Neural mechanics: Symmetry and broken conservation laws in deep learning dynamics. In

International Conference on Learning Representations, 2021.

Martinelli, F., Şimşek, B., Gerstner, W., and Brea, J. Expand-and-cluster: parameter recovery of neural networks. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org, 2024.

Mishkin, A., Bietti, A., and Gower, R. M. Level set teleportation: An optimization perspective, 2025.

Neyshabur, B., Salakhutdinov, R., and Srebro, N. Path-SGD: Path-normalized optimization in deep neural networks, 2015.

Nguyen, Q. On connected sublevel sets in deep learning. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 4790–4799. PMLR, 2019.

Nguyen, Q. and Hein, M. The loss surface of deep and wide neural networks. In *International Conference on Machine Learning*, 2017.

Safran, I. and Shamir, O. On the quality of the initial basin in overspecified neural networks, 2015. URL https://arxiv.org/abs/1511.04210.

Sagun, L., Evci, U., Guney, V. U., Dauphin, Y., and Bottou, L. Empirical analysis of the hessian of over-parametrized neural networks, 2017.

Salimans, T. and Kingma, D. P. Weight normalization: A simple reparameterization to accelerate training of deep neural networks, 2016.

Saxe, A. M., McClelland, J. L., and Ganguli, S. Exact solutions to the nonlinear dynamics of learning in deep linear neural networks, 2014.

Simsek, B., Ged, F., Jacot, A., Spadaro, F., Hongler, C., Gerstner, W., and Brea, J. Geometry of the loss landscape in overparameterized neural networks: Symmetries and invariances, 2021.

Sluis, A. v. d. Condition numbers and equilibration of matrices. *Numerische Mathematik*, 14:14–23, 1970. URL http://eudml.org/doc/131939.

Soudry, D. and Hoffer, E. Exponentially vanishing sub-optimal local minima in multilayer neural networks, 2017. URL https://arxiv.org/abs/1702.05777.

Sussmann, H. J. Uniqueness of the weights for minimal feedforward nets with a given input-output map. *Neural Networks*, 5(4):589–593, 1992.

Yang, G., Hu, E. J., Babuschkin, I., Sidor, S., Liu, X., Farhi, D., Ryder, N., Pachocki, J., Chen, W., and Gao, J. Tensor programs V: Tuning large neural networks via zero-shot hyperparameter transfer, 2022.

Zhang, B., Zheng, Z., Chen, Z., and Li, J. Beyond the permutation symmetry of transformers: The role of rotation for model fusion, 2025a.

Zhang, T. H., Maes, L., Milligan, A., Jolicoeur-Martineau, A., Mitliagkas, I., Scieur, D., Lacoste-Julien, S., and Guille-Escuret, C. Understanding Adam requires better rotation dependent assumptions, 2025b.

Zhang, Z., Song, Y., Yu, G., Han, X., Lin, Y., Xiao, C., Song, C., Liu, Z., Mi, Z., and Sun, M. Relu^2 wins: Discovering efficient activation functions for sparse llms, 2024. URL https://arxiv.org/abs/2402.03804.

Ziyin, L. Symmetry induces structure and constraint of learning, 2023. URL https://arxiv.org/abs/2309.16932.

Zou, D., Cao, Y., Zhou, D., and Gu, Q. Stochastic gradient descent optimizes over-parameterized deep ReLU networks, 2019.

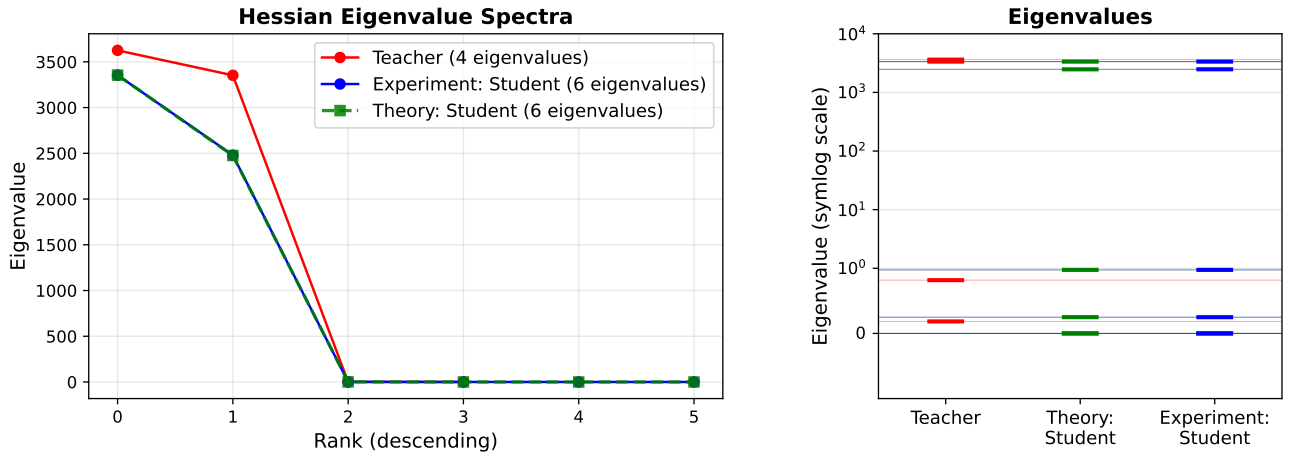

Figure 3. Empirical eigenvalue transformation for a width-2 teacher and width-3 student after optimization. *Left:* ranked eigenvalue spectra. *Right:* eigenvalue comparison. Eigenvalues align up to 0.2% with those predicted by Theorem 5.1; there are two additional flat directions from splitting.

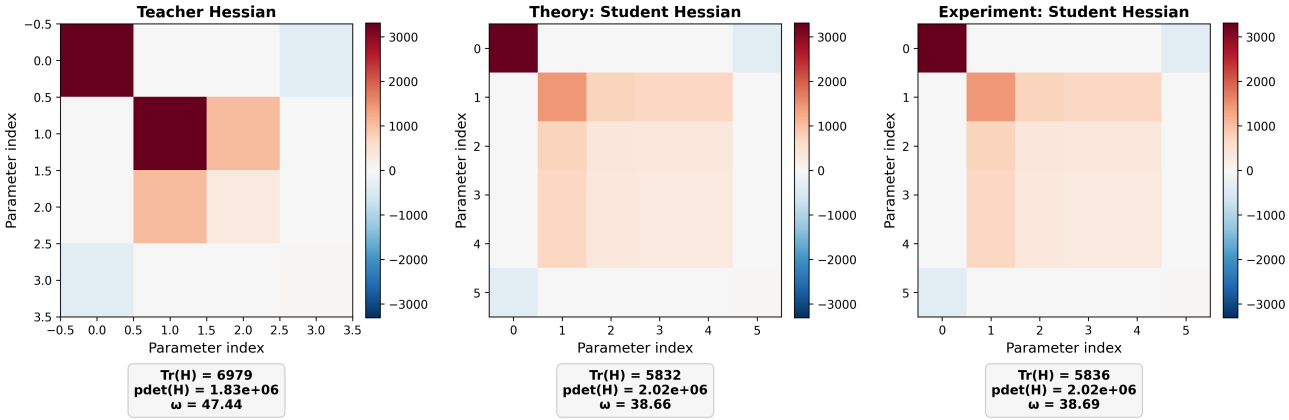

Figure 4. Theoretical and empirical Hessian comparison for a teacher-to-student expansion computed after optimizing to zero loss. *Left:* teacher Hessian. *Centre:* student Hessian predicted by Theorem 6.1. *Right:* empirically measured student Hessian after training to zero loss. The theoretical prediction matches the experiment to within 0.03% in $\mathrm{Tr}(H)$, $\mathrm{pdet}(H)$, and ω.

A Theory Checking and Direct Data Fitting Experiments

We directly test Theorem 6.1 by training small networks to zero loss with L-BFGS, guaranteeing convergence to a global minimum. To rule out global minima outside the symmetry orbit of the teacher, we use the activation $\sigma_{\alpha,\gamma}(x) = \sigma_{\mathrm{soft}}(x) + \alpha\,\sigma_{\mathrm{sig}}(\gamma x)$ from Simsek et al. (2021), for which all global minima of the teacher–student loss are related by symmetry. Figure 3 shows the empirical eigenvalue spectrum of the student. Figure 4 shows the Hessian predicted by Theorem 6.1 matches the empirically measured Hessian to within 0.03% in $\mathrm{Tr}(H)$, $\mathrm{pdet}(H)$, and ω. Post-optimization weights exhibit the predicted clustering structure (Figure 7; see also Figure 8).

Direct data fitting. To test whether the improvements depend on the teacher-student setup, we train networks directly on data. Figure 5 shows MLPs on California Housing regression and ConvNets on CIFAR-10 classification; the model cannot fully overfit in the latter setting. Conditioning improves with width in both cases.

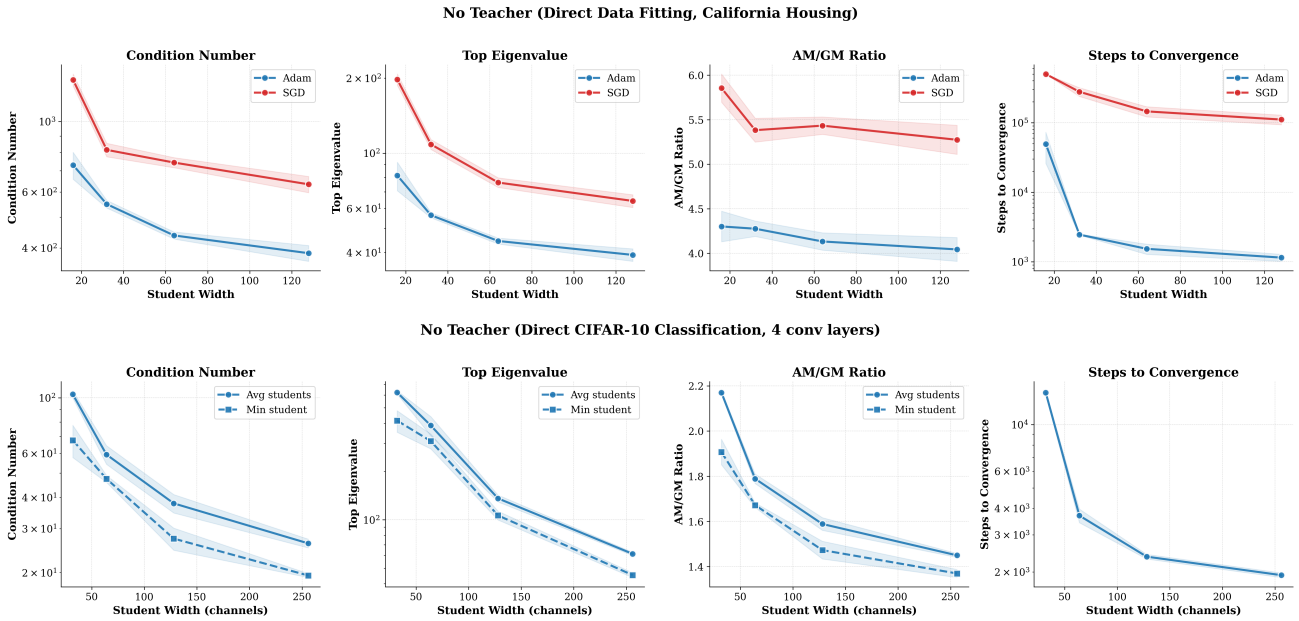

Figure 5. Top: MLPs trained directly on California Housing regression (no teacher). *Bottom:* ConvNets trained directly on CIFAR-10 classification (model cannot fully overfit). Conditioning improves with width in both settings.

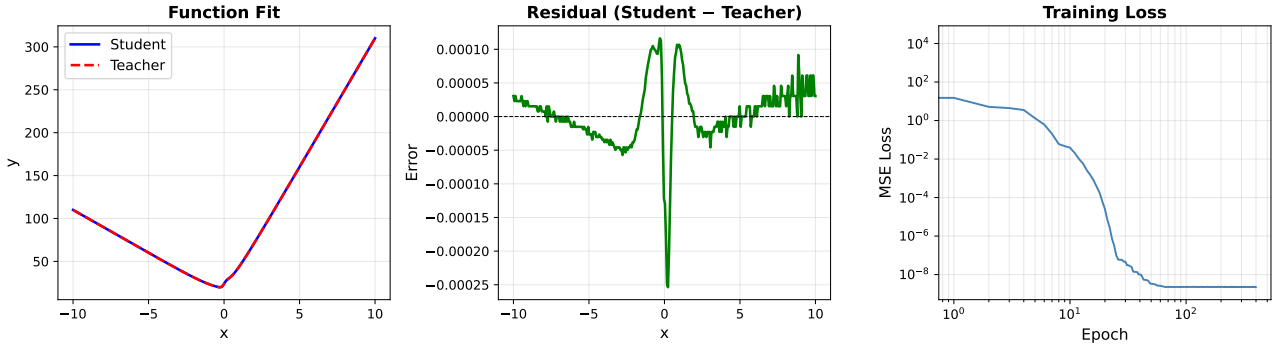

Figure 6. Training a width-3 student trained to match a width-2 teacher to zero loss via L-BFGS. *Left:* Student aligns exactly with teacher. *Centre:* There is a small residual error. *Right:* Loss curve using L-BFGS.

B Experimental Appendix

B.1 Architecture

For the depth-1 sweep, networks are single-hidden-layer MLPs ($L = 1$) with input and output dimension equal to the teacher width ($d_{\text{in}} = d_{\text{out}} = 8$). Deeper sweeps and other experiment families use the architectures listed in Table 1. The teacher has hidden width $n_f = 8$; student widths are $n \in \{8, 16, 32, 64\}$, corresponding to overparameterization ratios $n/n_f \in \{1, 2, 4, 8\}$. We test both ReLU and sigmoid activations.

B.2 Training

Synthetic MLP student-teacher. For each configuration, we train 5 independent student networks in parallel to match the frozen teacher's outputs. At each training step, a fresh batch of inputs is sampled from $\mathcal{N}(0, I)$ and the teacher generates targets; each student sees the same batch. The loss is the mean squared error between student and teacher outputs.

We use maximal update parameterization μP (Yang et al., 2022) for width-consistent learning dynamics, with base width 8. We test two optimizers:

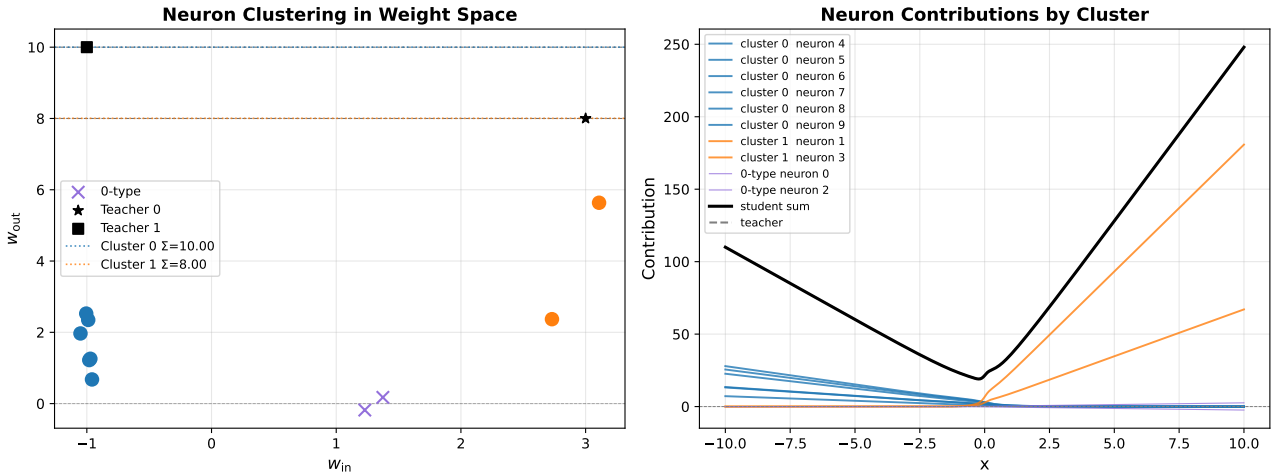

Figure 7. Neuron clustering at a global minimum of a width-10 student trained to match a width-2 teacher (zero loss via L-BFGS). *Left:* student weights in $(w_{\text{in}}, w_{\text{out}})$ space. Neurons cluster into two groups, each with the same w_{in} as the corresponding teacher neuron and outgoing weights summing to the teacher's value ($\Sigma = 10$ and $\Sigma = 8$). Two neurons are 0-type (output weights cancel). *Right:* per-cluster neuron contributions confirm each cluster implements one teacher neuron via splitting.

- **SGD**: learning rate 0.1, maximum 50,000 epochs.
- **Adam**: learning rate 0.01, maximum 5,000 epochs.

Training terminates early if the loss falls below 5×10^{-5} or if no improvement is observed for 20,000 consecutive epochs.

MLP on California Housing. As a no-teacher counterpart to the synthetic experiment, students are trained directly to predict median house value from the 8 standardised California Housing features, on a fixed 200-sample training subset. All other hyperparameters — μP, optimizer pair, learning rates, epoch budgets, early-stopping — match the synthetic experiment.

CNN on CIFAR-10. The architecture is a 4-block ConvNet (Conv $\rightarrow$ BN $\rightarrow$ ReLU, with 2×2 MaxPool every two blocks) followed by global average pooling and a linear head. Channel widths $\{32, 64, 128, 256\}$ are swept. Two settings share hyperparameters except for the loss: *teacher–student*, matching the logits of a pretrained width-32 teacher under MSE (early-stop 5×10^{-3}), and *direct*, predicting CIFAR-10 labels under cross-entropy (early-stop 0.4). Both use Adam at 10^{-3}, batch size 256, maximum 5,000 epochs.

Character-level transformer on Alice in Wonderland. The teacher is a 4-layer, 4-head encoder-only transformer with d_{model}=32, learned absolute positional embeddings, and a 73-character vocabulary, pretrained on the Alice corpus tokenised into 128-character windows. Students share the architecture but vary $d_{\text{model}} \in \{32, 64, 128\}$ and minimise per-token KL divergence to the teacher's logits. We use Adam at 10^{-3}, batch size 256, up to 50,000 epochs, and compute the top 50 Hessian eigenvalues (rather than 100) per student.

B.3 Compute Resources

All experiments are run on NVIDIA L40S GPUs. A single seed of a single configuration takes between roughly 5 minutes (smallest synthetic MLP runs) and 12 hours (largest Transformer student-teacher runs).

B.4 Hessian Computation

After convergence, we compute the Hessian eigenspectrum using the Lanczos algorithm with full reorthogonalization, which efficiently approximates the eigenvalue distribution via Hessian-vector products without forming the full matrix. For small networks used in theory validation we additionally compute the full Hessian exactly via finite differences to verify the Lanczos estimates. We extract: the complete eigenvalue spectrum, trace, top eigenvalue, condition number ($\lambda_{\max}/\lambda^{+}_{\min}$),

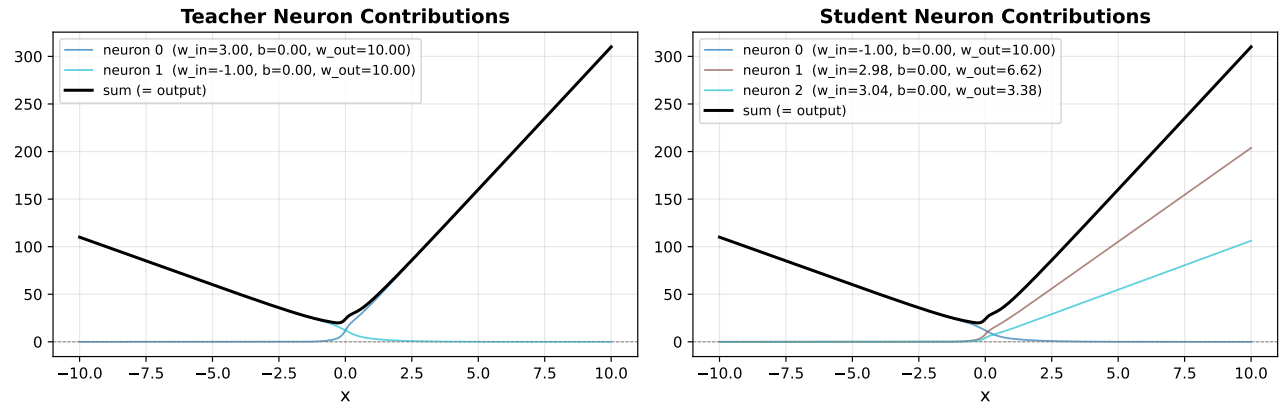

Figure 8. Weight structure at a global minimum of a width-3 student trained to match a width-2 teacher (zero loss via L-BFGS). *Left:* teacher neuron contributions. *Right:* student neuron contributions. Student neuron 0 replicates teacher neuron 1 exactly ($w_{\text{in}} = -1.00$), while neurons 1 and 2 split teacher neuron 0 ($w_{\text{in}} \approx 3.00$) with outgoing weights $6.62 + 3.38 = 10.00$, matching the predicted splitting structure.

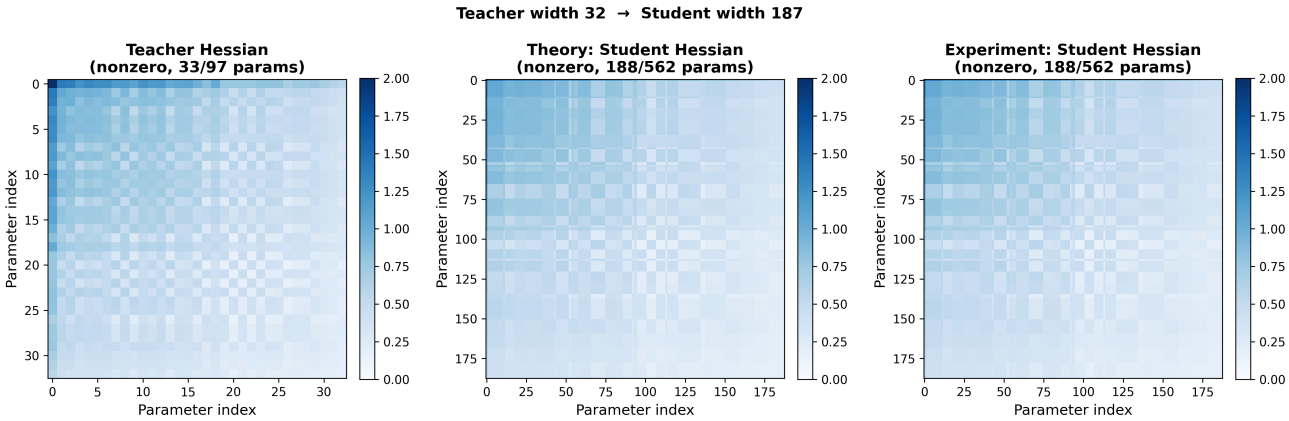

Figure 9. Theoretical and empirical Hessian comparison at larger scale computed by direct application of a symmetry transformation (teacher width 32 → student width 187, sigmoid activation). *Left:* teacher Hessian. *Centre:* student Hessian predicted by Theorem 6.1. *Right:* empirically measured student Hessian. Hessian entries agree nearly exactly, implying the prediction scales to larger networks.

pseudo-determinant (product of nonzero eigenvalues), and the AM/GM ratio of positive eigenvalues. To handle Lanczos numerical noise, eigenvalues below a small magnitude threshold are set to zero. Gradient norms are also recorded at the final iterate.

B.5 Hyperparameters

C Proof Details

C.1 Proof of Theorem P.1 (Gradient Scaling)

Using the standard notation

$$a^{(l)} = w^{(l)} h^{(l-1)} + b^{(l)}, \qquad h^{(l)} = \sigma(a^{(l)})$$

for the pre-activations and activations at layer l. Error backpropagates as

$$\delta^{(l-1)} = \sigma' \circ (W^l)^T \cdot \delta^{(l)}.$$

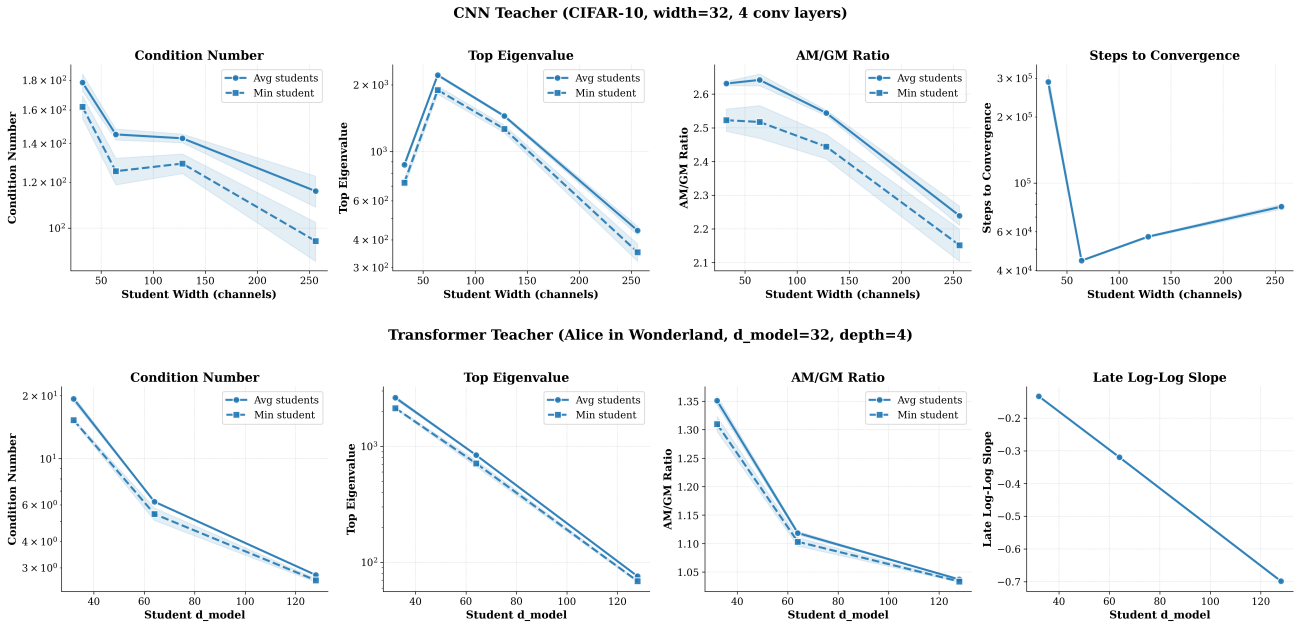

Figure 10. Top: CNN student-teacher experiment. A ConvNet teacher (width 32, 4 conv layers) is pre-trained on CIFAR-10. Student ConvNets of increasing channel width are trained to match the teacher's output logits. *Bottom:* Transformer student-teacher experiment. A character-level decoder-only Transformer teacher ($d_{\text{model}} = 32$, depth 4) is pre-trained on Alice in Wonderland. Student Transformers of increasing width are trained to match the teacher's output distribution via KL divergence. In both settings, condition number, top eigenvalue, and AM/GM ratio improve with width, alongside convergence behavior (steps to convergence for the CNN, late log-log slope for the Transformer).

The gradient of the weights are written

$$\frac{\partial L}{\partial w_{ji}^{(l)}} = \delta_j^{(l)} \, h_i^{(l-1)}.$$

For post-activation splitting, we duplicate the activations in layer l and split the outgoing weights in layer $l+1$. The gradient of the outgoing split weights is unaffected since $\delta_j^{(l+1)}$ and $h_i^{(l)}$ are fixed. The gradient of the incoming duplicated weights in layer l is divided by m since $\delta_j^{(l)}$ is split.

For pre-activation splitting, we split the activations in layer l and duplicate the outgoing weights in layer $l+1$. The gradient of the incoming split weights is unaffected since $\delta_j^{(l)}$ and $h_i^{(l-1)}$ are fixed.

The gradient of the outgoing duplicated weights in layer $l+1$ is divided by m since $h_i^{(l)}$ is split.

Finally, we can see downstream gradients are not impacted since $M^T \cdot \sigma' \cdot M^{+T} = \sigma'$:

$$\delta^{(l-1)} = \sigma' \circ (MW^l)^T \cdot \sigma' \circ (W^{l+1}M^+)^T \cdot \delta^{(l+1)} = \sigma' \circ (W^l)^T \cdot \sigma' \circ (W^{l+1})^T \cdot \delta^{(l+1)}.$$

C.2 Proof of Theorem 5.1 (Diagonal Preconditioning)

Applying Sylvester's Identity and letting $K = BB^\top \in \mathbb{R}^{a\times a}$, the characteristic polynomial of H' is

$$\det(H' - \lambda I_b) = (-\lambda)^{b-a} \det(HK - \lambda I_a) = (-\lambda)^{b-a} \det(K^{1/2}HK^{1/2} - \lambda I_a) \tag{5}$$

by similarity, since K is positive-definite for a valid expansion. This shows that H' has exactly $b-a$ zero eigenvalues, and its remaining a nonzero eigenvalues are identical to those of the full-rank $a \times a$ matrix HK.

C.3 Proof of Theorem 6.1 and Associated Theorems

To evaluate the change in the pseudo-determinant under expansion, we first explicitly characterize the diagonal matrix $K = BB^\top \in \mathbb{R}^{a\times a}$. As established in the main text, the expanded Hessian is $H' = B^\top HB$, and the nonzero eigenvalues of H' match those of HK.

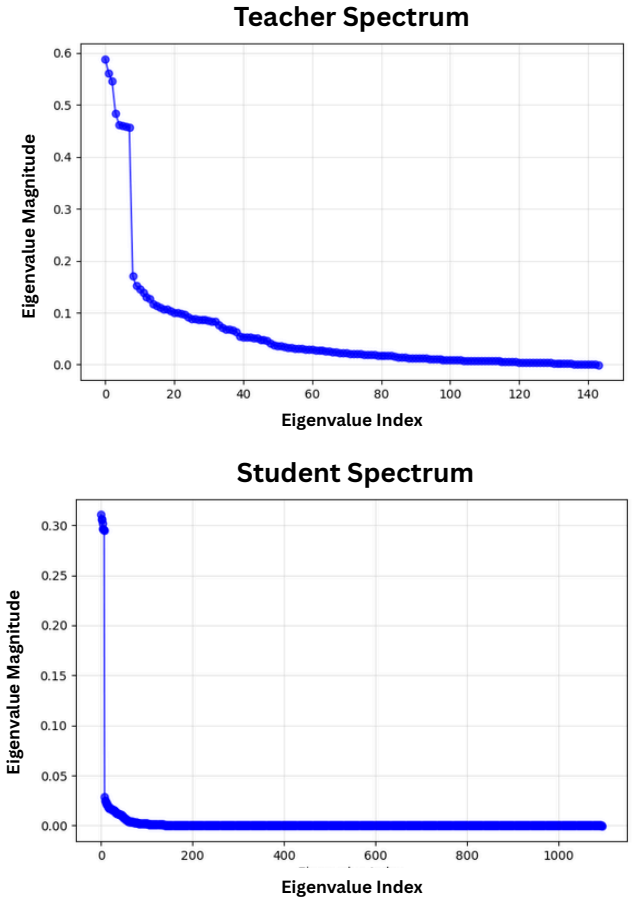

Figure 11. Hessian eigenvalue spectrum for a teacher network (top, width $n_f = 8$) and an overparameterized student (bottom, width $n = 64$). The teacher spectrum decays smoothly, while the student spectrum concentrates its curvature in a small number of leading eigenvalues with the vast majority near zero, corresponding to the flat directions introduced by overparameterization symmetries.

The matrix $B \in \mathbb{R}^{a \times b}$ maps the gradients of the b expanded parameters back to the a original parameters. Because $K = BB^\top$, the diagonal entries of K correspond exactly to the inner products of the rows of B:

- **Untouched weights:** The gradient mapping is a standard basis vector. The inner product is 1, yielding a diagonal entry of 1.
- **Split weights ($s \in S$):** a weight whose split copies have values α_i times the original (with $\sum_i \alpha_i = 1$). By Theorem P.1, each copy has *unchanged* gradient, so the row in B contains m ones, yielding a diagonal entry of m.
- **Duplicated weights ($d \in D$):** a weight whose copies are *identical* to the original, but whose copy gradients are scaled by the splitting proportions $\alpha_{d,i}$ (Theorem P.1). The row in B contains the $\alpha_{d,i}$ values, yielding a diagonal entry of $\sum_{i=1}^{m} \alpha_{d,i}^2$.

Because K is a diagonal matrix, its determinant is simply the product of these diagonal entries. Assuming the expansion involves an equal number of split and duplicated weights ($|S| = |D|$), we can pair the scaling factors of each split weight with a duplicated weight. The determinant simplifies to:

$$\det(K) = 1^{|O|} \cdot \prod_{s \in S} m \cdot \prod_{d \in D} \left(\sum_{i=1}^{m} \alpha_{d,i}^2 \right) = \prod_{d \in D} \left(m \sum_{i=1}^{m} \alpha_{d,i}^2 \right)$$

More generally, the first form $\det(K) = m^{|S|} \prod_{d \in D} (\sum_i \alpha_{d,i}^2)$ gives $\det(K) = m^{|S|-|D|}$ under even splitting, so volume is preserved only when $|S| = |D|$.

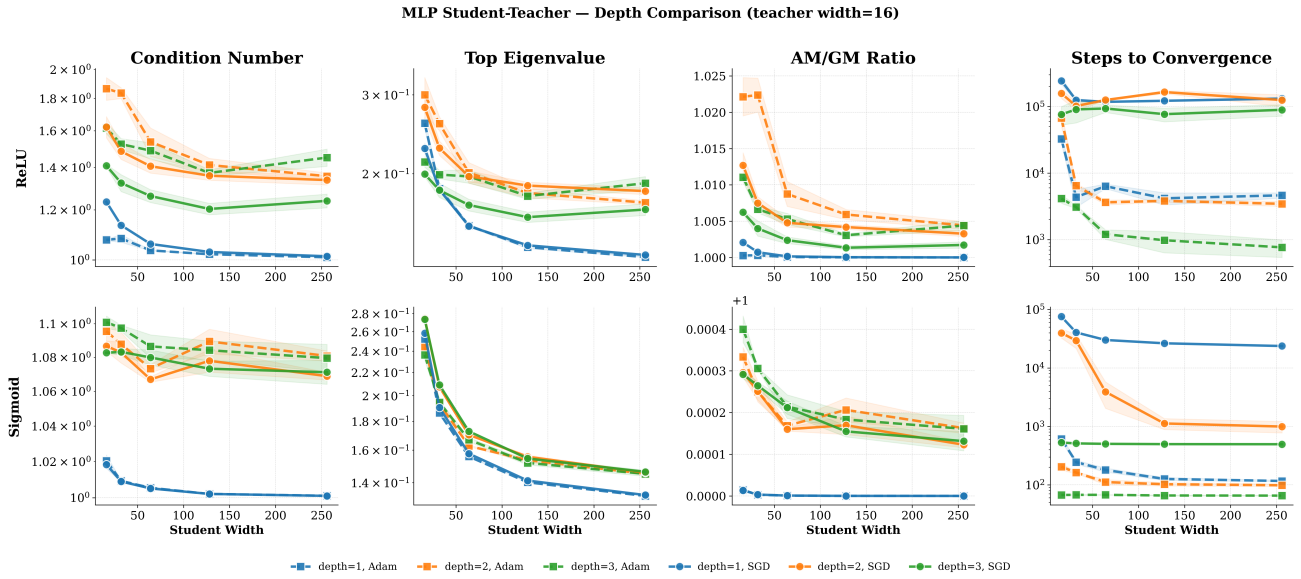

Figure 12. Effect of depth on MLP student-teacher conditioning (teacher width 16). Students at depth 1, 2, and 3 are trained with Adam and SGD (ReLU top, sigmoid bottom), with student width up to 256. Conditioning improves with width at all depths with no clear depth dependence, suggesting the symmetry-based mechanism is robust to depth.

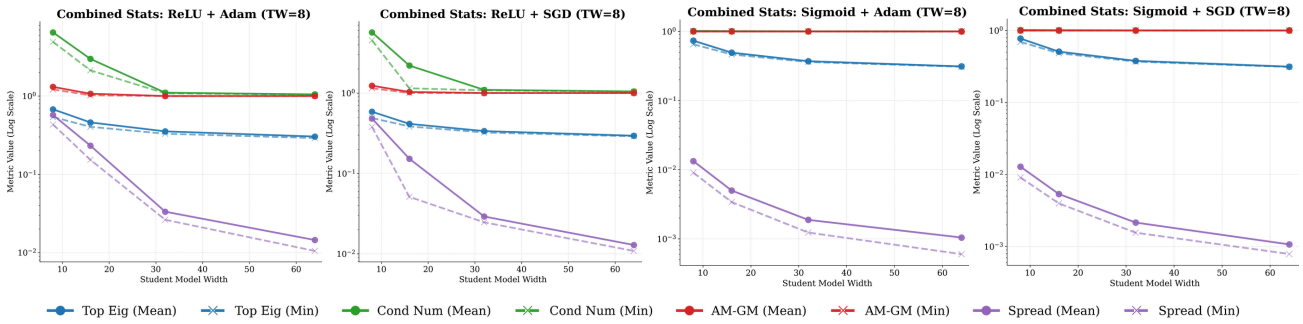

Figure 13. A comparison of conditioning across the average and best conditioned students. Training is done with 1 teacher and 50 student models across 5 seeds (an extended sweep, distinct from the main MLP setup in Table 1).

With the scaling factor $\det(K)$ explicitly defined, we now formally prove the invariance under even splitting and the volume expansion under uneven splitting.

Theorem C.1. *The pseudo-determinant of a full-rank Hessian H is invariant under even-splitting transformations, assuming an equal number of split and duplicated weights ($|S| = |D|$).*

Proof. For an even split, an original parameter is divided uniformly such that $\alpha_{d,i} = 1/m$ for all $i = 1, \ldots, m$. We evaluate the sum of the squared coefficients for any duplicated parameter $d \in D$:

$$\sum_{i=1}^{m} \alpha_{d,i}^2 = \sum_{i=1}^{m} \left(\frac{1}{m}\right)^2 = m\left(\frac{1}{m^2}\right) = \frac{1}{m}$$

Substituting this result into our expression for $\det(K)$ from the general framework, we find:

$$\det(K) = \prod_{d \in D} \left(m \cdot \frac{1}{m}\right) = \prod_{d \in D} (1) = 1$$

Because $\det(K) = 1$, the pseudo-determinant is strictly conserved:

$$\operatorname{pdet}(H') = \operatorname{pdet}(H)\det(K) = \operatorname{pdet}(H)$$

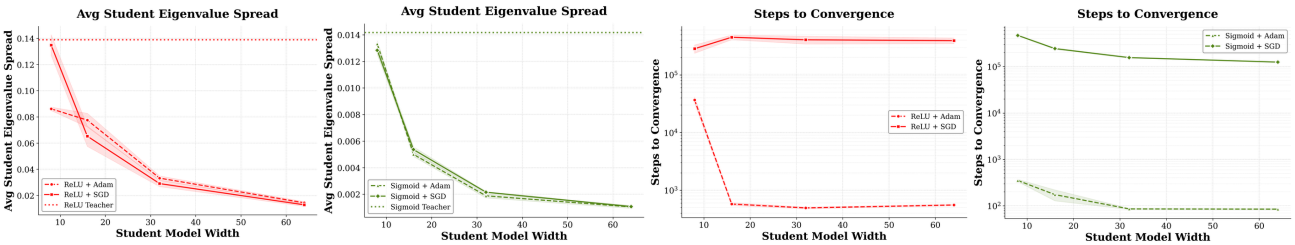

Figure 14. Additional Plots for Teacher-Student Network Training. Eigenvalue spread and steps to convergence.

Parameter	**MLP s–t**	**MLP Housing**	**CNN***	**LM s–t**
Architecture	MLP	MLP	ConvNet (4 conv)	CharTransformer
Dataset	synthetic	Cal. Housing	CIFAR-10	Alice in Wonderland
Teacher	random init	none (direct)	pretrained	pretrained
Teacher width n_f	$\{8, 16, 32, 64\}^\dagger$	16	32 channels	32 (d_{model})
Student widths n	$n_f \cdot \{1, 2, 4, 8, 16\}$	$\{16, 32, 64, 128\}$	$\{32, 64, 128, 256\}$	$\{32, 64, 128\}$
Depth	$\{1, 2, 3\}^\dagger$	2	4	4
Input / output dim.	n_f/n_f	8 / 1	3×32^2 / 10	vocab=73
Activation	ReLU, sigmoid	ReLU	ReLU	ReLU FFN, 4 heads
# students	5	25	5	5
Optimizer(s)	SGD, Adam	SGD, Adam	Adam	Adam
LR (SGD / Adam)	0.1 / 0.01	0.1 / 0.01	10^{-3}	10^{-3}
Max epochs (SGD/Adam)	50k / 5k	50k / 5k	5k	50k
Loss	MSE	MSE	MSE / CE*	KL on logits
Batch size	online	200	256	256
Early-stop loss	5×10^{-5}	5×10^{-5}	0.005/0.4*	—
Patience (epochs)	20k	20k	20k	20k
Param.	μP (base 8)	μP (base 8)	standard	standard
Top-k Hessian eigs.	100	100	100	50
Seeds	0, 1, 2	0, 1, 2	0, 1, 2	0, 1, 2

Table 1. Experimental hyperparameters. † For the synthetic MLP student-teacher, depth 1 is swept over $n_f \in \{8, 16, 32, 64\}$ and depths 2–3 over $n_f \in \{8, 16\}$. * CNN sweep covers two settings sharing all hyperparameters except loss/early-stop: *teacher–student* (MSE vs. pretrained ConvNet, stop 0.005) and *direct* (cross-entropy vs. CIFAR-10 labels, stop 0.4). MLP Housing uses 200 standardized real samples; CNN uses the full CIFAR-10 training split; LM uses the Alice corpus tokenised into 128-character non-overlapping windows.

This completes the proof. □

Theorem C.2. *Let H' be obtained after acting on the weights corresponding to a Hessian H with the pre-activation or post-activation splitting groupoid. For any general split where $|S| = |D|$, the pseudo-determinant of the Hessian satisfies* $\operatorname{pdet}(H') \geq \operatorname{pdet}(H)$.

Proof. Consider a general split where the proportions $\alpha_{d,1}, \ldots, \alpha_{d,m}$ are not necessarily uniform, but inherently satisfy $\sum_{i=1}^{m} \alpha_{d,i} = 1$. By the Root-Mean-Square/Arithmetic Mean (RMS-AM) inequality, the sum of the squares of these coefficients is strictly bounded below:

$$\sqrt{\frac{1}{m}\sum_{i=1}^{m}\alpha_{d,i}^2} \geq \frac{1}{m}\sum_{i=1}^{m}\alpha_{d,i} = \frac{1}{m}(1)$$

Squaring both sides and multiplying by m, we obtain:

$$\sum_{i=1}^{m}\alpha_{d,i}^2 \geq \frac{1}{m} \implies m\sum_{i=1}^{m}\alpha_{d,i}^2 \geq 1$$

Applying this inequality to our established expression for $\det(K)$, we find:

$$\det(K) = \prod_{d\in D}\left(m\sum_{i=1}^{m}\alpha_{d,i}^2\right) \geq \prod_{d\in D}(1) = 1$$

Consequently, the scaling factor applied to the pseudo-determinant is always greater than or equal to 1, yielding:

$$\operatorname{pdet}(H') = \operatorname{pdet}(H)\det(K) \geq \operatorname{pdet}(H)$$

By the exact equality conditions of the RMS-AM inequality, $\det(K) = 1$ if and only if $\alpha_{d,1} = \cdots = \alpha_{d,m}$ for all $d \in D$. Thus, the pseudo-determinant strictly increases under any uneven split, recovering equality only in the case of a perfectly even split as proven in Theorem C.1. □

C.4 Proof of Theorem 7.1 (Trace Formula)

Direct calculation from the Hessian block structure. After an order-m split, the diagonal entries transform as:

- Entries in H_{OO}: unchanged
- Entries in H_{SS}: each entry H_{ss} is replicated m times, contributing $m \cdot H_{ss}$
- Entries in H_{DD}: each entry H_{dd} is replicated m times but scaled by $1/m^2$, contributing $(1/m) \cdot H_{dd}$

Summing: $\operatorname{Tr}(H') = \operatorname{Tr}(H_{OO}) + m\sum_s H_{ss} + (1/m)\sum_d H_{dd}$.

Since originally $\operatorname{Tr}(H) = \operatorname{Tr}(H_{OO}) + \sum_s H_{ss} + \sum_d H_{dd}$, we get the stated formula.

Extension to non-even splits. The same calculation (or directly via $\operatorname{Tr}(H') = \operatorname{Tr}(B^\top HB) = \operatorname{Tr}(KH)$, using the diagonal entries of K from Theorem 6.1's proof) extends to splits with arbitrary coefficients $\alpha_1, \ldots, \alpha_m$ summing to 1:

$$\operatorname{Tr}(H') = \operatorname{Tr}(H_{\setminus S,D}) + m\sum_{s\in S} H_{ss} + \left(\sum_{i=1}^{m} \alpha_i^2\right)\sum_{d\in D} H_{dd}, \tag{6}$$

recovering Theorem 7.1 when $\alpha_i = 1/m$. Since RMS-AM gives $\sum_i \alpha_i^2 \geq 1/m$ with equality iff $\alpha_i = 1/m$, even splits also minimize the trace at fixed m.

Optimal Splitting Degree. Treating the splitting degree m as continuous, the trace formula is convex in m; we call its minimizer (the value of m that yields the smallest trace under the fixed split structure S, D) the *optimal splitting degree*. Setting $\partial \operatorname{Tr}(H')/\partial m = 0$,

$$m^* = \sqrt{\frac{\sum_{d\in D} H_{dd}}{\sum_{s\in S} H_{ss}}}. \tag{7}$$

Substituting m^* back into the trace formula (or applying AM-GM) yields the minimum trace

$$\min \operatorname{Tr}(H') = \operatorname{Tr}(H_{\setminus S,D}) + 2\sqrt{\sum_{d\in D} H_{dd} \sum_{s\in S} H_{ss}}. \tag{8}$$

D Zero-type Neurons

The main text describes two function-preserving operations for arbitrary activations. The first (neuron duplication) is characterized in Section 3 (and Appendix O) using duplication matrices M and pseudoinverses M^+. Here we give an analogous characterization of the second: adding *0-type neurons*, whose total contribution to the network output cancels.

Setup. Consider a single hidden layer with n neurons, input weight matrix $W_l \in \mathbb{R}^{n\times d_{\text{in}}}$ and output weight matrix $W_{l+1} \in \mathbb{R}^{d_{\text{out}}\times n}$. We expand to $n+k$ neurons via expansion matrix $M_0 \in \mathbb{R}^{(n+k)\times n}$ and contraction matrix $M_0^+ \in \mathbb{R}^{n\times(n+k)}$:

$$M_0 = \begin{bmatrix} I_n \\ A \end{bmatrix}, \qquad M_0^+ = \begin{bmatrix} I_n & Z \end{bmatrix}, \tag{9}$$

where $A \in \mathbb{R}^{k\times n}$ and $Z \in \mathbb{R}^{n\times k}$. The transformation $W_l \to M_0 W_l$ and $W_{l+1} \to W_{l+1}M_0^+$ adds k new neurons with input weights AW_l and effective output weights $W_{l+1}Z$.

Function preservation. Verifying $f_\theta(x)$ is preserved:

$$\begin{aligned} W_{l+1} M_0^+ \sigma(M_0 W_l x) &= \begin{bmatrix} W_{l+1} & W_{l+1} Z \end{bmatrix} \begin{bmatrix} \sigma(W_l x) \\ \sigma(A W_l x) \end{bmatrix} \\ &= W_{l+1}\, \sigma(W_l x) + W_{l+1} Z\, \sigma(A W_l x). \end{aligned} \tag{10}$$

The new neurons' total contribution $W_{l+1} Z\, \sigma(A W_l x)$ must vanish for all x. Partition the new neurons into groups by equality of their input weights (rows of AW_l). For arbitrary σ, this constraint is equivalent to: for each group G_g, the corresponding columns of $W_{l+1} Z$ sum to the zero vector. Writing $\mathbf{1}_g \in \mathbb{R}^k$ for the indicator vector of group G_g:

$$W_{l+1} Z\, \mathbf{1}_g = 0 \quad \text{for every group } g. \tag{11}$$

Two special cases.

- **Identically zero outputs:** when no two new neurons share an input weight (rows of AW_l are distinct), each group has size 1 and the constraint reduces to $W_{l+1} Z = \mathbf{0}$, i.e., every column of Z lies in the right null space of W_{l+1}.
- **Cancellation:** when new neurons come in pairs (or larger groups) with shared input weights, each group's columns of $W_{l+1} Z$ must sum to zero, but individual columns may be nonzero.

Relation to duplication. The 0-type expansion is complementary to neuron duplication. Duplication copies existing neurons and redistributes their output weights (the new neurons are functionally redundant copies of existing ones). The 0-type expansion adds genuinely new neurons whose total output cancels. Together, these two operations generate the full set of overparameterization symmetries for arbitrary activations described in Section 3 (and Appendix O).

Example: Adding 0-type Neurons

For a network with $n = 2$ hidden neurons and $d_{\text{out}} = 2$, adding two 0-type neurons sharing their input weight (rows 3 and 4 of M_0 are equal):

$$M_0 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ a_1 & a_2 \\ a_1 & a_2 \end{bmatrix}, \qquad M_0^+ = \begin{bmatrix} 1 & 0 & z_1 & -z_1 \\ 0 & 1 & z_2 & -z_2 \end{bmatrix}.$$

Both new neurons share the pre-activation $(a_1 W_l[1,:] + a_2 W_l[2,:])x$, and their output contributions $W_{l+1} z\, \sigma(\cdot)$ and $-W_{l+1} z\, \sigma(\cdot)$ cancel for any $z = (z_1, z_2)^\top$. The parameters a_1, a_2, z_1, z_2 are all free.

E Per-Weight Splitting and the Inverse Transformation

E.1 Per-Weight Coefficient Matrices

The formulation in Simsek et al. (2021) requires scaling the entire outgoing weight vector together. However, there exists a strictly richer symmetry: the splitting coefficients need not be shared across the entire outgoing weight vector.

Let $W_{l+1} \in \mathbb{R}^{d \times n}$ have columns $\vec{a}_1, \vec{a}_2, \ldots, \vec{a}_n$. If column i is duplicated m_i times, we can choose a corresponding coefficient matrix $C_i \in \mathbb{R}^{d \times m_i}$ with normalized rows: $\sum_{j=1}^{m_i} (C_i)_{kj} = 1$ for all rows k.

The transformed output weights are:

$$W_{l+1} \to \begin{bmatrix} C_1 \odot \vec{a}_1 & C_2 \odot \vec{a}_2 & \ldots & C_n \odot \vec{a}_n \end{bmatrix}, \tag{12}$$

where $\odot$ denotes element-wise multiplication and broadcasting applies column-wise (equivalently, the i-th block is given by $\operatorname{diag}(\vec{a}_i) C_i$).

Example: Per-Weight Splitting Coefficients

For $M = \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}$, we can transform a 2×2 matrix:

$$W_{l+1} = \begin{bmatrix} a_1 & a_2 \\ a_3 & a_4 \end{bmatrix} \to \begin{bmatrix} \mu_1 a_1 & (1-\mu_1)a_1 & a_2 \\ \mu_2 a_3 & (1-\mu_2)a_3 & a_4 \end{bmatrix}$$

for constants $\mu_1, \mu_2 \in \mathbb{R}$. Crucially, μ_1 and μ_2 can be chosen independently, giving per-weight control rather than per-neuron control.

E.2 Row-wise Hessian Scaling

The per-weight splitting above forces us to scale rows/columns in the Hessian as neuron-wise blocks. Here we discuss the gradient behavior when splits use individual weight-wise coefficients c_{jk}.

Theorem E.1. *Using the notation from Appendix P, if our splits use individual coefficients per weight c_{jk}, we get:*

$$\frac{\partial L}{\partial w_{ij}'^{(l)}} = h_i^{(l-1)} (\sigma^l)' \sum_k c_{jk} w_{jk}^l \delta_k^{(l+1)} \tag{13}$$

$$= \frac{\sum_k c_{jk} w_{jk}^l \delta_k^{(l+1)}}{\sum_k w_{jk}^l \delta_k^{(l+1)}} \frac{\partial L}{\partial w_{ij}^{(l)}} \tag{14}$$

$$:= d_j \frac{\partial L}{\partial w_{ij}^{(l)}}. \tag{15}$$

The coefficient d_j only depends on index j, so we cannot control the gradient of individual weights independently.

E.3 Inverse Transformation

Given an expanded network with duplication matrix $M \in \mathbb{R}^{r \times n}$ and contraction M^+, the inverse transformation recovers the original width-n parameterization. For post-activation splitting, where neuron i was duplicated m_i times, the inverse groups each set of m_i copies back into a single neuron: the input weights are averaged across copies (recovering the original, since all copies are identical by construction), and the output weights are summed.

Concretely, the inverse uses the even-split pseudoinverse $M^+_{\text{even}} \in \mathbb{R}^{n \times r}$ for the input side and the original expansion matrix $M \in \mathbb{R}^{r \times n}$ for the output side:

$$W_l' \to M^+_{\text{even}} W_l', \quad W_{l+1}' \to W_{l+1}' M, \tag{16}$$

where $(M^+_{\text{even}})_{i,j} = 1/m_i$ if column j is a copy of neuron i and 0 otherwise. The input averaging recovers the shared weights since $M^+_{\text{even}} M = I_n$. Right-multiplying by M naturally sums the split output weights: because the expanded weights are $W_{l+1}' = W_{l+1} M^+$, the inverse operation yields $W_{l+1}' M = W_{l+1} M^+ M = W_{l+1} I_n = W_{l+1}$.

For pre-activation splitting (ReLU), the roles reverse: output weights are averaged and input weights are summed to undo the splitting coefficients D_{m_i}.

Note that the inverse is only well-defined on the image of the forward transformation i.e., on networks whose weights exhibit the tied structure produced by expansion. This is why the symmetries form a groupoid rather than a group (Section 4).

Example: Inverse of Post-Activation Splitting

For $M = \left[\begin{smallmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 1 \end{smallmatrix}\right]$ (duplicating neuron 1), suppose the expanded network has:

$$W_l' = \begin{bmatrix} \vec{w}_1 \\ \vec{w}_1 \\ \vec{w}_2 \end{bmatrix}, \quad W_{l+1}' = \begin{bmatrix} \mu_1 a_1 & (1-\mu_1)a_1 & a_2 \\ \mu_2 a_3 & (1-\mu_2)a_3 & a_4 \end{bmatrix}.$$

The inverse uses $M_{\text{even}}^+ = \left[\begin{smallmatrix} 1/2 & 1/2 & 0 \\ 0 & 0 & 1 \end{smallmatrix}\right]$ and the original matrix M:

$$M_{\text{even}}^+ W_l' = \begin{bmatrix} \vec{w}_1 \\ \vec{w}_2 \end{bmatrix},$$

$$W_{l+1}' M = \begin{bmatrix} \mu_1 a_1 + (1-\mu_1)a_1 & a_2 \\ \mu_2 a_3 + (1-\mu_2)a_3 & a_4 \end{bmatrix} = \begin{bmatrix} a_1 & a_2 \\ a_3 & a_4 \end{bmatrix},$$

recovering the original parameterization.

F Transformation of Weight Space, Gradients and Hessian under Splitting

For an L-layer neural network, assume we insert a transformation pair $M_l^+ M_l = I$ between every adjacent layer l and $l+1$. The forward parameters transform as follows:

- First layer: $W_1 \to M_1 W_1$
- Middle layers ($1 < l < L$): $W_l \to M_l W_l M_{l-1}^+$
- Final layer: $W_L \to W_L M_{L-1}^+$

By vectorizing the weights using the identity $\text{vec}(ABC) = (C^T \otimes A)\text{vec}(B)$, the parameter transformation matrix $\mathbf{P}$ across all layers is the block-diagonal matrix:

$$\mathbf{P} = \text{diag}\left(I \otimes M_1, \ldots, (M_{l-1}^+)^T \otimes M_l, \ldots, (M_{L-1}^+)^T \otimes I\right) \tag{17}$$

Because parameters are contravariant while gradients are covariant, the gradient vector ∇L_θ transforms via the transpose of the pseudo-inverse of the parameter transformation: $\nabla L_{\theta'} = (\mathbf{P}^+)^T \nabla L_\theta$.

Letting $B^T = (\mathbf{P}^+)^T$, the correct transformation for the expanded gradient vector across the entire network is $\nabla L_{\theta'} = B^T \nabla L_\theta$, where B^T is the block-diagonal matrix:

$$B^T = \text{diag}\left(I \otimes (M_1^+)^T, \ldots, M_{l-1} \otimes (M_l^+)^T, \ldots, M_{L-1} \otimes I\right) \tag{18}$$

By the multivariate chain rule, the Jacobian of this transformation is simply the constant matrix B:

$$J = \frac{\partial \theta}{\partial \theta'} = B \tag{19}$$

The first derivative (the gradient) transforms via the transposed Jacobian:

$$\nabla_{\theta'} L = J^T \nabla_\theta L = B^T \nabla_\theta L \tag{20}$$

To find the transformed Hessian $H_{\theta'} = \nabla_{\theta'}^2 L$, we differentiate the transformed gradient with respect to the new parameters θ':

$$H_{\theta'} = \frac{\partial}{\partial \theta'}\left(B^T \nabla_\theta L\right) \tag{21}$$

Because B is a matrix of constants, its derivative with respect to θ' is zero. Thus, the product rule yields no higher-order derivative tensor terms. Applying the chain rule, we extract the original Hessian $H_\theta = \frac{\partial}{\partial\theta}(\nabla_\theta L)$ and multiply by the Jacobian $J = B$:

$$\begin{aligned} H_{\theta'} &= B^T \left(\frac{\partial(\nabla_\theta L)}{\partial\theta} \right) \frac{\partial\theta}{\partial\theta'} \\ &= B^T H_\theta B \end{aligned} \tag{22}$$

The gradient identity $\nabla_{\theta'} L = B^\top \nabla_\theta L$ holds on the symmetry orbit. The corresponding Hessian relation picks up an extra term proportional to ∇L off critical points, so $H_{\theta'} = B^\top H_\theta B$ holds exactly at the zero-loss minima we study. More generally, the relation holds exactly whenever the Gauss-Newton approximation to the Hessian is exact.

In overparameterized networks, almost all local minima are globally optimal (Nguyen & Hein, 2017). The residual $e\nabla^2 f$ vanishes at these zero-loss minima, so $H_{\theta'} = B^\top H_\theta B$ holds exactly throughout the minimum landscape.

Block form for even splits. Partitioning the weights into O (other), S (split), and D (duplicated) with $|S| = |D|$, an order-m even split transforms the Hessian as

$$H = \begin{bmatrix} H_{OO} & H_{OS} & H_{OD} \\ H_{OS}^\top & H_{SS} & H_{SD} \\ H_{OD}^\top & H_{SD}^\top & H_{DD} \end{bmatrix} \rightarrow H' = \begin{bmatrix} H_{OO} & H'_{OS} & \frac{1}{m} H'_{OD} \\ H'^\top_{OS} & H'_{SS} & \frac{1}{m} H'_{SD} \\ \frac{1}{m} H'^\top_{OD} & \frac{1}{m} H'^\top_{SD} & \frac{1}{m^2} H'_{DD} \end{bmatrix}, \tag{23}$$

where $H'_{XX} = H_{XX} \otimes \mathbf{1}_{m\times m}$. Every duplicated row/column appears m times in its block; the $1/m$ and $1/m^2$ factors come from the $\alpha_i = 1/m$ row scalings of B for duplicated weights.

G The Overparameterization Groupoid

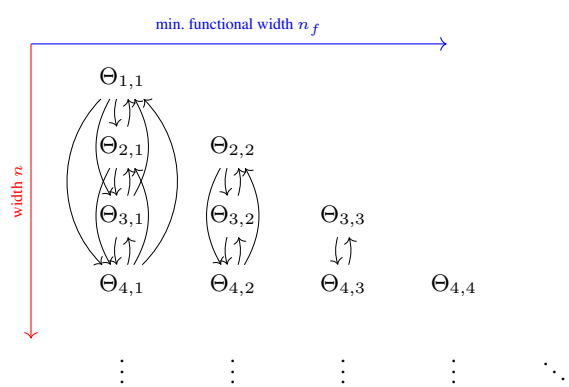

Figure 15. The overparameterization groupoid. Θ_{n,n_f} denotes weights of width n and minimum functional width n_f. Arrows indicate symmetry transformations that connect equivalent parameters across the widths.

Let $\Gamma^{(L,N)}$ be the set of functions exactly representable by a neural network with L hidden layers and N neurons per layer. Notice that $\Gamma^{(L,N)} \subset \Gamma^{(L+p,N+q)}$ for any $p, q \in \mathbb{N}$. Let Θ_N^L be the corresponding parameter space of weights for such a network. For brevity, we denote the single-layer parameter space as $\Theta_N = \Theta_N^1$, and the union of parameter spaces with width at least a as $\Theta_{\geq a}^L = \bigcup_{N=a}^\infty \Theta_N^L$.

To formalize overparameterization (i.e., post-activation splitting) that preserves the network's function, we define operations mapping between these parameter spaces. For $b \geq a$, let $T_{a\to b}$ be the set of single-layer expansion operations:

$$T_{a\to b} = \{(M_{a\to b}, C_1, C_2 \dots C_a) \mid M_{a\to b} \text{ and } C_i \text{ are valid expansions from } a \text{ to } b\}, \tag{24}$$

where $M_{a\to b}$ and C_i are the expansion and coefficient matrices from Section 3. Any transformation $t \in T_{a\to b}$ maps $\Theta_a \to \Theta_b$.

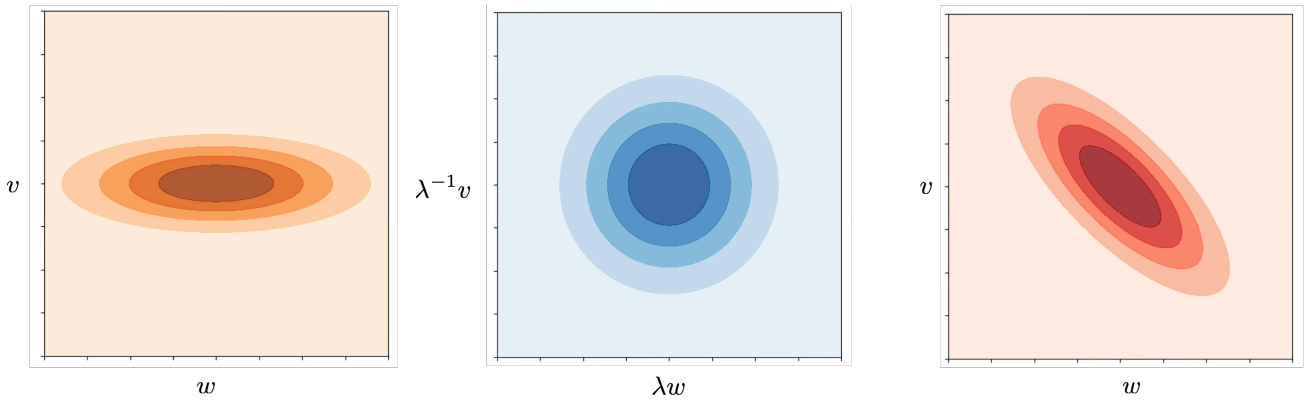

Figure 16. Diagonal preconditioning via symmetry transformations. Left: a poorly conditioned minimum (w^*, v^*) with elongated level sets. Center: rescaling to $(\lambda w^*, v^*/\lambda)$ compresses the w-axis and stretches the v-axis, producing rounder level sets that are easier to optimize. Right: when the principal curvature directions are not axis-aligned, diagonal rescaling cannot improve conditioning.

Conversely, we define a set of inverse operations, $T_{b\to a}$, which are strictly defined on the image of the expansions, $\mathrm{im}(T_{a\to b}) \subset \Theta_b$. Specifically, any $t \in T_{b\to a}$ takes $b-a$ duplicated pre-activation weights and collapses them into a single weight, while summing the corresponding output weights. These duplicated weights are guaranteed to exist for any $\theta \in \mathrm{im}(T_{a\to b})$ (see Appendix E.3).

For an entire L-layer network, the transformations are applied layer-wise. We denote the set of full-network expansions as the Cartesian product $T^L_{a\to b} = (T_{a\to b})^L$, with inverses defined similarly.

We now construct an algebraic structure to capture all valid sequences of these transformations. We must use a **groupoid** rather than a group for two reasons:

1. **Partial composition:** A transformation mapping $\tau_{a\to b}$ can only be composed with a subsequent transformation $\tau_{b\to c}$. The binary operation is only defined when input and output widths match.
2. **Conditional invertibility:** We can only invert an expansion (apply a contraction) if the weights are perfectly tied as a result of a previous expansion.

Let n_f be the minimum width such that the network's underlying function $f_\theta \in \Gamma^{(L,n_f)}$. We define the overparameterization groupoid $\mathcal{O}^L_{n_f}$ over the parameter space $\Theta^L_{\geq n_f}$. $\mathcal{O}^L_{n_f}$ is formally generated by the collection of all valid layer-wise expansions, their inverses, and the identity operations I_N:

$$\mathcal{O}^L_{n_f} = \langle T^L_{a\to b} \cup T^L_{b\to a} \cup \{I_N\} \mid \forall a, b, N \geq n_f \rangle. \tag{25}$$

The partial binary operation $*$ is standard function composition, which is inherently associative. Because $\mathcal{O}^L_{n_f}$ contains all identities, inverses on valid images, and associative partial composition, it satisfies the groupoid axioms. A key property of $\mathcal{O}^L_{n_f}$ is that it describes the exact orbit of parameters that leave the underlying function invariant; for any $\theta \in \Theta^L_{\geq n_f}$ and valid $g \in \mathcal{O}^L_{n_f}$, $f_\theta = f_{g\cdot\theta}$.

We remark that even some weight-space symmetries at a fixed width can be viewed as "local" or groupoid symmetries; Grigsby et al. (2023) provide a detailed analysis of such hidden symmetries in ReLU networks.

H Optimization Intuition via Rescale Symmetry

Before analyzing the overparameterization symmetries from the main text, we build intuition with the simpler rescale symmetry, which is present even in exactly-parameterized networks. The core argument carries over to the overparameterization case. It is well documented that rescale symmetries impact the curvature of minima and can be exploited to improve optimization (Huang et al., 2017; Salimans & Kingma, 2016; Neyshabur et al., 2015; Mishkin et al., 2025).

Consider a two-layer ReLU network $f(x; W, U) = U \cdot \mathrm{ReLU}(Wx)$. Since ReLU is equivariant to positive rescaling, the transformation $g_\lambda(W, U) = (\lambda W, \lambda^{-1} U)$ for $\lambda \in \mathbb{R}^+$ preserves the function. More generally, each hidden neuron can be rescaled independently.

Gradients under rescaling. At a rescaled point $(\lambda W, \lambda^{-1}U)$, the gradients transform as:

$$\frac{\partial L}{\partial W'} = \lambda^{-1}\frac{\partial L}{\partial W}, \qquad \frac{\partial L}{\partial U'} = \lambda\frac{\partial L}{\partial U}. \tag{26}$$

This follows from the chain rule: $W' = \lambda W$ so $\partial L/\partial W'_{ij} = (\partial W_{ij}/\partial W'_{ij}) \cdot \partial L/\partial W_{ij} = \lambda^{-1}\partial L/\partial W_{ij}$, and similarly for U. Rescaling thus redistributes gradient magnitudes between the input and output weights. When $\lambda > 1$, input weight gradients shrink while output weight gradients grow, and vice versa. This is analogous to the gradient scaling under splitting (Theorem P.1), where the splitting coefficients play the role of λ.

Hessian under rescaling. If (W^*, U^*) is a global minimum, rescaling gives a class of equivalent minima $[(W^*, U^*)]_{\mathcal{T}}$. The minimum (W^*, U^*) can be arbitrarily poorly conditioned, but the symmetry orbit contains solutions with different conditioning. The Hessian at (W^*, U^*) has the block structure:

$$H = \begin{bmatrix} H_{WW} & H_{WU} \\ H_{WU}^\top & H_{UU} \end{bmatrix}. \tag{27}$$

At the rescaled point $(\lambda W^*, \lambda^{-1}U^*)$, the Hessian becomes:

$$H(\lambda) = \begin{bmatrix} \lambda^{-2}H_{WW} & H_{WU} \\ H_{WU}^\top & \lambda^2 H_{UU} \end{bmatrix}. \tag{28}$$

This is diagonal preconditioning with $K = \mathrm{diag}(\lambda^{-2}I, \lambda^2 I)$, matching the structure from Section 5.1. Choosing λ lets us trade off curvature between the W and U blocks.

Limitations. While $\min_\lambda \kappa(H(\lambda)) \leq \kappa(H)$ always holds, rescaling cannot improve conditioning universally. To see this, consider the Rayleigh quotient for $x = (u, v)$, which is a weighted average of eigenvalues:

$$R(H(\lambda), x) = \frac{\lambda^{-2}u^\top H_{WW}u + 2u^\top H_{WU}v + \lambda^2 v^\top H_{UU}v}{\|u\|^2 + \|v\|^2}. \tag{29}$$

The off-diagonal term $u^\top H_{WU}v$ is unaffected by λ, so if the extreme eigenvalues are dominated by the cross-block coupling, rescaling has little effect. Similarly, if both extreme eigenvalues come from a single block (say H_{WW}), rescaling scales them together and cannot change their ratio. Conditioning improves when the largest and smallest eigenvalues are split across the H_{WW} and H_{UU} blocks, which corresponds to the eigenvectors being approximately axis-aligned in the (W, U) decomposition. This is depicted in Figure 16.

I Which Activations Allow Pre-Activation Splitting?

Understanding general pre-activation splitting requires solving the functional equation representing a single split:

$$\forall x \in \mathbb{R}, \quad \sigma(\alpha x) + \sigma(\beta x) = \sigma(x), \tag{30}$$

for some expansion constants α, β. A direct observation is that setting $x = 0$ yields $2\sigma(0) = \sigma(0)$, implying $\sigma(0) = 0$. This immediately rules out activations with non-zero origins, such as the standard sigmoid.

To solve this systematically, we first restrict our attention to the domain $x > 0$ and assume $\alpha, \beta > 0$. We can map this to a continuous difference equation by substituting $x = e^t$ and defining $y(t) = \sigma(e^t)$. Letting $\alpha = e^a$ and $\beta = e^b$, the equation becomes:

$$y(t + a) + y(t + b) = y(t). \tag{31}$$

Applying the Laplace transform (assuming mild growth conditions on y) yields $(e^{as} + e^{bs} - 1)Y(s) = 0$. For non-trivial solutions ($Y(s) \neq 0$), s must be a root of the characteristic equation, meaning s lies in the set:

$$S_{a,b} = \{s \in \mathbb{C} \mid e^{as} + e^{bs} = 1\}. \tag{32}$$

If the ratio a/b is rational (i.e., $a/b = p/q$ with $\gcd(p, q) = 1$), the substitution $u = e^{s/q}$ reduces this to a polynomial equation $u^p + u^q = 1$, which has exactly $\max(p, q)$ roots for u by the Fundamental Theorem of Algebra. Taking the inverse Laplace transform for a given principal root yields base solutions of the form $y(t) = Ce^{st}$, which translates back to $\sigma(x) = Cx^s$, subject to the algebraic constraint $\alpha^s + \beta^s = 1$. When s is strictly real, these correspond to standard monomial functions.

However, we must also consider complex roots. When $s = \lambda + iw$, the solutions exhibit oscillatory behavior. For a purely imaginary root ($s = iw$, so $\lambda = 0$), the real-valued solutions take the form $\sigma(x) = \cos(w \ln x)$. Returning to the difference equation, this requires $\cos(wt + aw) + \cos(wt + bw) = \cos(wt)$. By angle addition, this holds when the phases form an equilateral triangle in the complex plane, yielding $(aw, bw) = (\pm\frac{\pi}{3} + 2\pi m, \mp\frac{\pi}{3} + 2\pi n)$ for $m, n \in \mathbb{Z}$. Because the limit $\lim_{x\to 0^+} \cos(w \ln x)$ is undefined, these purely oscillatory solutions are pathological and impractical for neural networks.

A careful treatment of signs maps our real roots ($s = 1, 2, \dots$) directly to practical piecewise-polynomial activations. This framework naturally recovers both the standard ReLU ($s = 1$) and ReLU^2 ($s = 2$), which has recently emerged as a performant choice in sparse LLMs (Zhang et al., 2024).

J Axis-Algined Eigenvectors and Transformer Rotation Symmetries

We empirically check the axis alignment of the Hessian eigenvectors of pretrained teacher networks across architectures.

In 17, we compute the top-1000 Hessian eigenpairs via Lanczos at the trained parameters, and for each eigenvector v measure the participation ratio $\mathrm{PR}(v) = 1/\sum_i v_i^4$ on the parameter basis. The fourth power simply makes PR an interpretable count: if v has equal mass on k coordinates and zero on the rest, then $\sum_i v_i^4 = 1/k$, so $\mathrm{PR}(v) = k$. A lower power would not give this property: $\sum_i v_i^2 = 1$ for any unit v, so it cannot distinguish a one-hot eigenvector from a fully spread one. We report $\mathrm{PR}(v)/n_{\text{params}}$, the fraction of parameters the eigenvector effectively occupies; a uniformly random unit vector in $\mathbb{R}^{n_{\text{params}}}$ has expected fraction $\approx 1/3$.

Across the three architectures we tested (a CNN trained on CIFAR-10, an absolute-position character Transformer pretrained on Alice in Wonderland, and the same Transformer with RoPE positional encoding) the Hessian eigenvectors lie below this random baseline (Figure 17), indicating a degree of axis alignment relative to random directions. The localization is strongest at the largest eigenvalues and decays towards the random baseline as one moves down the spectrum.

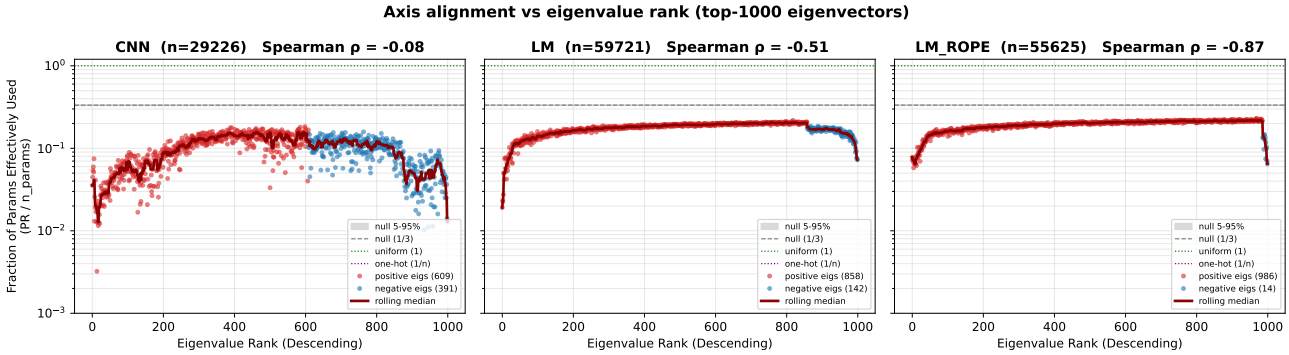

Figure 17. Fraction of parameters effectively used by each of the top-1000 Hessian eigenvectors of three pretrained teachers, against a random-unit-vector null ($\approx 1/3$, gray band). Red points correspond to positive eigenvalues, blue to negative; the dark red line is a rolling median.

To see where this localization sits in parameter space, we decompose each top eigenvector v into per-tensor energy fractions $\sum_{i\in T} v_i^2$ for each parameter tensor T, and divide by the parameter-count fraction $|T|/n_{\text{params}}$ that a uniform random direction would assign to T. The resulting quantity (one per (eigenvector, tensor) pair) takes value 1 for a uniform direction and grows when the eigenvector concentrates more energy on a tensor than its size alone would predict. Figure 18 shows this decomposition for the top-20 eigenvectors of the RoPE Transformer teacher. The brightest columns correspond to the per-block additive biases on the residual stream (the attention output biases `attn.out.b` and the FFN output biases `ff.2.b` of every layer), each of which contains only 32 of the ∼56k model parameters but absorbs a disproportionate share of each top eigenvector's energy.

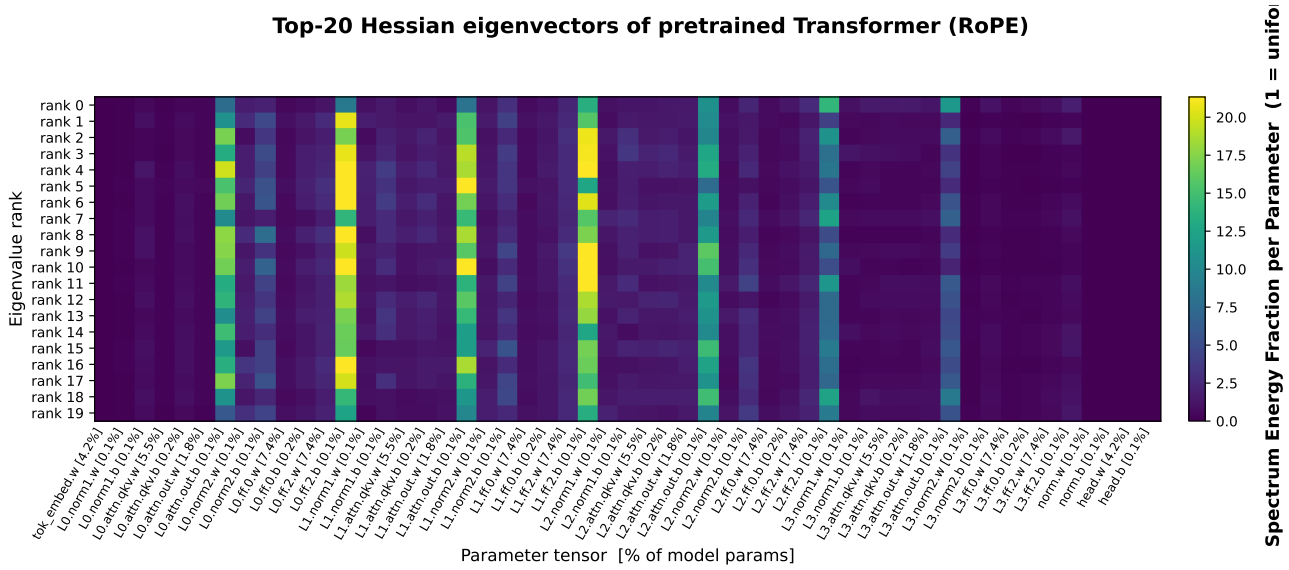

Figure 18. Spectrum energy fraction per parameter for the top-20 Hessian eigenvectors of the RoPE Transformer teacher, broken down by parameter tensor. Each column is a parameter tensor (annotated with its share of total parameters); each row is an eigenvector ranked by descending eigenvalue. Brighter cells indicate more eigenvector energy per parameter than a uniform random direction would assign.

Transformers have an additional rotation symmetry in their MLP weights, partially described in Zhang et al. (2025a) for networks without skip connections. In modern pre-norm Transformers with skip connections, the symmetry is more restricted. In general, the symmetry performs a rotation in the MLP weight matrices $W \to R_{\text{left}} W R_{\text{right}}^{\top}$ for some special orthogonal matrices $R_{\text{left}}, R_{\text{right}} \in O(n)$, acting jointly across adjacent blocks to preserve the residual stream.

To see how this rotates weight space, we apply the Kronecker product identity:

$$\text{vec}(R_{\text{left}} W R_{\text{right}}^{\top}) = (R_{\text{left}} \otimes R_{\text{right}})\text{vec}(W). \tag{33}$$

Intuitively, this rotates in unison the column weights by R_{left} and the row weights by R_{right}.

This rotation symmetry can aid optimization algorithms with built-in diagonal preconditioners, such as Adam. Recent work has noted the underperformance of Adam when rotating the preconditioner in Transformers but not ResNets (which lack this rotation symmetry), suggesting the minima Adam converges to typically have axis-aligned eigenvectors (Zhang et al., 2025b).

K Notes on Plasticity Loss

A direct application of our framework concerns plasticity loss in continual learning. If optimization converges to a simple generalizing function, the solution lies on a low functional width column of the overparameterization groupoid (Figure 15). Such solutions are typically realized through dormant capacity such as duplicated or dead neurons, configurations that are fixed points of subgroups of the network's symmetry group. It is a well-known result that gradient descent preserves these fixed-point subspaces under any loss: cloned neurons stay cloned, dead neurons stay dead, throughout training. Once the objective changes, the optimizer cannot recruit this dormant capacity to fit the new task.

Ziyin (2023) previously analyzed this trapping mechanism for permutation, rescaling, and rotation symmetries under weight decay or noise, and explicitly linked it to plasticity loss. A more detailed and concurrent dynamical systems treatment appears in Joudaki et al. (2025).

L Local Volume of Minima

We compute the volume of the local ϵ-sublevel set used in Section 6. Taylor-expanding the loss near a minimum θ^*, the local ellipsoidal region $\mathcal{E}_\epsilon(\theta^*)$ where the loss increases by at most ϵ satisfies $\frac{1}{2}(\theta - \theta^*)^{\top} H(\theta - \theta^*) \leq \epsilon$.

Assuming the parameter space has dimension n, the affine change of variables $u = \frac{1}{\sqrt{2\epsilon}} H^{1/2}(\theta - \theta^*)$ maps the region to the unit n-ball $u^\top u \leq 1$, with Jacobian determinant $(2\epsilon)^{n/2}(\det(H))^{-1/2}$. Integrating over the unit ball gives

$$\mathrm{Vol}(\mathcal{E}_\epsilon(\theta^*)) = \frac{(2\epsilon)^{n/2}\ \mathrm{Vol}(B_n)}{\sqrt{\det(H)}}, \tag{34}$$

where $\mathrm{Vol}(B_n) = \pi^{n/2}/\Gamma(n/2+1)$. When H is singular (as for H' obtained by splitting), $\det(H)$ is replaced by the pseudo-determinant $\mathrm{pdet}(H)$, giving the volume modulo flat symmetry directions used in Theorem 6.1.

M Derivation of the Volume Ratio

We derive the probability ratio when expanding from n to $n+1$ neurons (i.e. $k = 1$). Throughout we fix $d_{\text{in}} = d_{\text{out}} = 1$, so the base network has $2n$ parameters and the expanded network has $2(n+1)$ parameters. Both networks use the same constant $c > 0$, with the m-neuron network initialized uniformly on $[-b_m, b_m]^{2m}$ where $b_m = 1/\sqrt{m}$.

Domain assumption. We restrict attention to base minima θ^* whose weights all satisfy $|w_j^{\text{in}}|, |w_j^{\text{out}}| \leq b_{n+1}$ for every j. Under this condition, for any $\alpha \in [0,1]$ the split minimum $\theta^*_{\alpha,j}$ lies entirely inside $[-b_{n+1}, b_{n+1}]^{2(n+1)}$: the shared incoming weights are bounded by b_{n+1} by assumption, and $|\alpha w_j^{\text{out}}|, |(1-\alpha) w_j^{\text{out}}| \leq |w_j^{\text{out}}| \leq b_{n+1}$. The entire interval $\alpha \in [0,1]$ is therefore admissible.

M.1 Base network: P_n

Volume of the ϵ-sublevel ellipsoid. Fix a global minimum θ^*. The quadratic approximation of the loss defines an ellipsoid $\{\theta : \frac{1}{2}(\theta - \theta^*)^\top H(\theta - \theta^*) \leq \epsilon\}$ in $\mathbb{R}^{2n}$. Assuming H is full rank (no flat directions for the base network), the ellipsoid has volume

$$\mathrm{Vol}(\mathcal{E}_\epsilon(\theta^*)) \ = \ \frac{(2\epsilon)^n\ \mathrm{Vol}(B_{2n})}{\sqrt{\det(H)}}, \tag{35}$$

where $\mathrm{Vol}(B_{2n}) = \pi^n/\Gamma(n+1)$.

Summing over permutation copies. Permuting the n neurons gives $n!$ equivalent minima. Assuming the ϵ-sublevel sets around distinct permutation copies are disjoint (which holds when ϵ is small relative to the separation between minima), and that each ellipsoid lies inside $[-b_n, b_n]^{2n}$ (which holds when ϵ is small relative to the distance from θ^* to the boundary), the probability of initializing within loss ϵ of any global minimum of this type is

$$\boxed{P_n \ = \ \frac{n!\ (2\epsilon)^n\ \mathrm{Vol}(B_{2n})}{(2b_n)^{2n}\ \sqrt{\det(H)}}.} \tag{36}$$

M.2 Expanded network: P_{n+1}

Structure of minima. For the $k = 1$ expansion, a global minimum of the expanded network arises by splitting exactly one neuron $j \in \{1, \ldots, n\}$ into two copies with outgoing weights αw_j^{out} and $(1-\alpha) w_j^{\text{out}}$ for some $\alpha \in [0,1]$, while both copies share the incoming weight w_j^{in}. The corresponding parameter vector is

$$\theta^*_{\alpha,j} \ = \ \big(\ldots,\ w_j^{\text{in}},\ \alpha w_j^{\text{out}},\ w_j^{\text{in}},\ (1-\alpha) w_j^{\text{out}},\ \ldots\big) \ \in\ \mathbb{R}^{2(n+1)}.$$

Pseudo-determinant of the expanded Hessian. By Theorem 6.1 applied to the single split, the expanded Hessian H' has $2n$ nonzero eigenvalues and 2 zero eigenvalues. Its pseudo-determinant satisfies

$$\mathrm{pdet}(H') \ = \ \det(H)\ \cdot\ 2\big(\alpha^2 + (1-\alpha)^2\big). \tag{37}$$

Flat directions. The 2 zero eigenvalues correspond to two distinct directions.

- *Globally flat direction*: shifting $\alpha \mapsto \alpha + t$ (redistributing the outgoing weight) leaves the network function unchanged. We integrate over this direction explicitly below.

- *Locally flat direction*: the null direction of the Hessian block for the two copies' incoming weights is $v_\alpha = (1-\alpha, -\alpha)$, since linearizing the network output in incoming-weight perturbations $(\delta w_{\text{in}}^{(1)}, \delta w_{\text{in}}^{(2)})$ gives a change proportional to $\alpha\, \delta w_{\text{in}}^{(1)} + (1-\alpha)\, \delta w_{\text{in}}^{(2)}$. Consider a perturbation size s along v_α giving a set of parameters θ_s. Along this direction the first order change in f_{θ_s} is 0, and since the loss is quadratic in f_{θ_s} it grows at most quartically (as s^4), so the sublevel set $\{L \le \epsilon\}$ has width $O(\epsilon^{1/4})$, contributing a factor $(\epsilon/C)^{1/4}$ for a constant $C > 0$ taken as a uniform upper bound on the fourth-order coefficient over $\alpha \in [0,1]$ and admissible teacher minima.

Volume in the nonflat subspace. In the $2n$-dimensional subspace orthogonal to both flat directions:

$$\mathrm{Vol}\big(\mathcal{E}_\epsilon(\theta^*_{\alpha,j})^\perp\big) \;=\; \frac{(2\epsilon)^n\ \mathrm{Vol}(B_{2n})}{\sqrt{\mathrm{pdet}(H')}} \;=\; \frac{(2\epsilon)^n\ \mathrm{Vol}(B_{2n})}{\sqrt{\det(H)}\cdot\sqrt{2(\alpha^2+(1-\alpha)^2)}}. \tag{38}$$

Integrating over the globally flat direction. The orbit is the line segment $\theta^*_\alpha = (\ldots, w_j^{\text{in}}, \alpha w_j^{\text{out}}, w_j^{\text{in}}, (1-\alpha) w_j^{\text{out}}, \ldots)$ with tangent vector $\partial_\alpha \theta^*_\alpha$ of norm $\sqrt{2}\,|w_j^{\text{out}}|$, so the arc-length element is $d\ell = \sqrt{2}\,|w_j^{\text{out}}|\, d\alpha$. By the domain assumption, the full interval $\alpha \in [0,1]$ is admissible. Integrating the per-α cross-section with respect to arc length,

$$I(1) \;:=\; \int_0^1 \frac{\sqrt{2}\,|w_j^{\text{out}}|\, d\alpha}{\sqrt{2\big(\alpha^2+(1-\alpha)^2\big)}} \;=\; \sqrt{2}\,|w_j^{\text{out}}|\,\ln(1+\sqrt{2}), \tag{39}$$

where the closed form follows by the substitution $u = \alpha - \frac{1}{2}$ and the standard arsinh integral. The two permutation copies $(\alpha, 1-\alpha)$ and $(1-\alpha, \alpha)$ are already counted in the global $(n+1)!$ permutation factor, so we divide by $2! = 2$.

Summing over choices of split neuron. There are n choices of which neuron j to split, each contributing $I(1) = \sqrt{2}\,|w_j^{\text{out}}|\,\ln(1+\sqrt{2})$ to the orbit integral. Summing over j gives a total of $\sqrt{2}\,\ln(1+\sqrt{2})\sum_{j=1}^n |w_j^{\text{out}}| = \sqrt{2}\,\ln(1+\sqrt{2})\,\|w^{\text{out}}\|_1$. Collecting all factors:

$$P_{n+1} \ge \underbrace{(2b_{n+1})^{-2(n+1)}}_{\text{init density}} \cdot \underbrace{(n+1)!}_{\text{perm. copies}} \cdot \underbrace{(\epsilon/C)^{1/4}}_{\text{locally flat direction}} \cdot \underbrace{\frac{(2\epsilon)^n\ \mathrm{Vol}(B_{2n})}{\sqrt{\det(H)}}}_{\text{ellipsoid vol.}} \cdot \underbrace{\frac{\sqrt{2}\,\ln(1+\sqrt{2})\,\|w^{\text{out}}\|_1}{2}}_{\text{split neuron choices}}. \tag{40}$$

M.3 The ratio P_{n+1}/P_n

Dividing (40) by (36), and substituting Xavier initialization $b_m = 1/\sqrt{m}$:

$$\begin{aligned}\frac{P_{n+1}}{P_n} &\ge \underbrace{\frac{(2b_n)^{2n}}{(2b_{n+1})^{2(n+1)}}}_{=\frac{(n+1)^{n+1}}{4\,n^n}} \cdot \underbrace{\frac{(n+1)!}{n!}}_{=n+1} \cdot (\epsilon/C)^{1/4} \cdot \frac{\sqrt{2}\,\ln(1+\sqrt{2})\,\|w^{\text{out}}\|_1}{2} \\ &= \frac{\sqrt{2}\,\ln(1+\sqrt{2})}{8} \cdot \|w^{\text{out}}\|_1\,(n+1)^2\left(\frac{n+1}{n}\right)^n (\epsilon/C)^{1/4}.\end{aligned} \tag{41}$$

Remark M.1 (Growth threshold). All factors in (41) are positive, so $P_{n+1}/P_n > 0$ always. The ratio exceeds 1 if and only if

$$\|w^{\text{out}}\|_1\,(n+1)^2\left(\frac{n+1}{n}\right)^n \;>\; \frac{8C^{1/4}}{\sqrt{2}\,\ln(1+\sqrt{2})\,\epsilon^{1/4}}. \tag{42}$$

Since $((n+1)/n)^n$ is increasing in n with $((n+1)/n)^n \ge 2$ for all $n \ge 1$, a sufficient condition is

$$\|w^{\text{out}}\|_1\,(n+1)^2 > \frac{8C^{1/4}}{\sqrt{2}\,\ln(1+\sqrt{2})\,\epsilon^{1/4}}. \tag{43}$$

Remark M.2 (Lower bound interpretation). The formula (41) counts only global minima arising from single-neuron splitting (the $k = 1$ duplication symmetry). The true P_{n+1} also receives contributions from other global minima not of this form, so (41) provides a lower bound on the true ratio.

N Related Work

Geometry of overparameterization. Sussmann (1992) showed early on that for minimal-width networks, weights are determined up to permutation and sign-flip symmetries. Fukumizu & Amari (2000) later demonstrated that as width increases, the hierarchical structure of MLPs allows local minima to transform into saddle points, smoothing the landscape. Simsek et al. (2021) rigorously extended this to the overparameterized setting, identifying the continuous symmetries (neuron splitting and merging) that arise when width exceeds the functional minimum; their work provides the foundational groupoid structure for our analysis. Recently, Martinelli et al. (2024) leveraged these symmetries for parameter recovery, showing that overparameterized student networks naturally "cluster" their weights around the underlying minimal representation. We build upon these results by showing that these symmetries act as diagonal preconditioning on the Hessian and scale the probability mass of global minima.

Loss landscape structure. Cooper (2021) proves that for networks with d parameters and n data points ($d > n$), the global minima generically form a $(d - n)$-dimensional submanifold rather than isolated points. Nguyen (2019) shows that all sublevel sets of the loss are connected under mild overparameterization, ruling out "bad local valleys." Empirically, Garipov et al. (2018) and Draxler et al. (2018) independently found that minima from different training runs can be connected by paths of near-constant loss. Entezari et al. (2022) conjectured that after accounting for permutation symmetry, SGD solutions from different initializations lie in a single basin; Ainsworth et al. (2023) gave practical algorithms for aligning models modulo these symmetries. Our work adds to this picture by showing that overparameterization symmetries not only connect minima but reshape the curvature at those minima. Safran & Shamir (2015) bounded the probability of initializing in a basin with monotonic descent to a global minimum, and Soudry & Hoffer (2017) proved that sub-optimal local minima vanish exponentially in volume as width grows. Our Section 8 provides a specific symmetry-based mechanism for these basin-quality improvements.

Convergence theory. Jacot et al. (2018) introduced the neural tangent kernel (NTK), showing that in the infinite-width limit, training reduces to kernel regression with a fixed kernel. Building on this, several works proved that gradient descent on sufficiently wide networks converges to global minima at a linear rate: Du et al. (2019) for deep residual networks, Allen-Zhu et al. (2019) for general deep networks, and Zou et al. (2019) for SGD on deep ReLU networks. These results typically require the width to be polynomial in the number of samples and the depth. Chizat et al. (2019) showed that this favorable convergence arises from a "lazy training" regime where parameters stay near initialization and the model behaves as its linearization, which can underperform feature learning in practice. Our approach is different: rather than analyzing dynamics in a linearized regime, we characterize the geometric properties (conditioning, volume) of minima at finite width.

Implicit acceleration and deep linear networks. Deep linear networks, where $f(x) = W_L \cdots W_1 x$, are a standard testbed for optimization theory. Saxe et al. (2014) derived exact learning dynamics, showing that depth creates saddle-to-saddle transitions that progressively learn singular value components. Arora et al. (2018) showed that the resulting dynamics acts as implicit acceleration: the factored parameterization preconditions the optimization of the end-to-end matrix, with the effect depending on the singular values of the current solution. Ghosh et al. (2025) extend this analysis beyond the edge of stability, characterizing oscillatory dynamics at large learning rates. We extend the preconditioning perspective in a different direction, from depth to width and from linear to general activations, showing that overparameterization symmetries provide diagonal preconditioning on the Hessian.

Conditioning and curvature. Sagun et al. (2017) showed empirically that overparameterized networks have Hessian spectra with a large bulk of near-zero eigenvalues and a few data-dependent outliers; the near-zero eigenvalues correspond to flat directions from overparameterization, while the outliers capture the curvature relevant to learning. Ghorbani et al. (2019) gave a more detailed eigenvalue density analysis, connecting spectral structure to properties of the data and architecture. Guille-Escuret et al. (2023) found that optimization paths in neural networks have surprisingly simple geometry despite the non-convexity of the loss. Yang et al. (2022) showed through μP that wider networks are more robust to learning rate choices, suggesting that width improves conditioning in a way that reduces hyperparameter sensitivity. Classical results on diagonal preconditioning (Forsythe & Straus, 1955; Collobert, 2004) show that coordinate rescaling can improve the condition number when eigenvectors are not uniformly distributed across coordinates; this underpins our analysis in Section 5.1.

Normalization and symmetry exploitation. Several practical methods explicitly navigate weight-space symmetries to find better-conditioned parameterizations. Weight normalization (Salimans & Kingma, 2016) decouples weight magnitude from direction, moving along the rescaling orbit. Path-SGD (Neyshabur et al., 2015) uses a path-regularized objective invariant to rescaling, so the trajectory is not distorted by the parameterization choice within an equivalence class. Mishkin et al. (2025) introduced level-set teleportation, which solves for a better-conditioned point on the same level set by optimizing over the

symmetry group. All of these can be viewed as searching within a symmetry orbit for favorable parameterizations.

O Additional Symmetry Constructions

O.1 Fixed-width Symmetries in Nonlinear Networks

A number of well-studied fixed-width parameter symmetries can be seen as matrices. For a network with activation σ to have a symmetry via matrix M, we require $M^{-1}\sigma(Mx) = \sigma(x)$ for all x. For example, permuting hidden units in a width-n layer corresponds to M a permutation matrix; for ReLU networks, neuron-wise rescaling corresponds to $M = \mathrm{diag}(\lambda_1, \ldots, \lambda_n)$ with $\lambda_i > 0$.

O.2 Overparameterization in Deep Linear Networks

Deep linear networks $f(x) = W_L \cdots W_1 x$ have a matrix factorization symmetry that impacts optimization (Saxe et al., 2014; Ghosh et al., 2025): between any pair $W_{k+1}W_k$ at fixed width n we can insert $M^{-1}M$ for any $M \in \mathrm{GL}_n(\mathbb{R})$. This is the same symmetry present between W_K and W_Q in self-attention. To generalize to overparameterization, $M \in \mathbb{R}^{r\times n}$ becomes an expansion ($r > n$) and M^+ a corresponding pseudoinverse with $M^+M = I_n$; the freedom in M^+ corresponds to redistributing weights. The transformation $W_k \to MW_k$ and $W_{k+1} \to W_{k+1}M^+$ preserves the function while changing the parameterization.

O.3 ReLU Pre-Activation Splitting

ReLU and its variants are linear in half their domain, equivariant to positive rescaling: $\sigma(\lambda x) = \lambda\sigma(x)$ for $\lambda > 0$. This enables a stricter *pre-activation splitting* symmetry. To split activations evenly, multiply the pre-activation weights by positive coefficients $[d_1, \ldots, d_m]$ with $\sum_i d_i = 1$. If the pre-activation is positive, ReLU remains linear and outputs are preserved; if negative, the activation stays negative. In matrix form, the expansion is $M' = [D_{m_1} \otimes \vec{e}_1^\top; \ldots; D_{m_n} \otimes \vec{e}_n^\top]$ with $D_{m_i} = [d_1, \ldots, d_{m_i}]^\top$, and the contraction M'^+ copies the columns. This is stricter than post-activation splitting because it forces the entire incoming weight vector to share the same coefficient; activations that satisfy this property are characterized in Appendix I.

O.4 Per-Weight Splitting Coefficients

The post-activation splitting in the main text uses neuron-wise coefficients α_i, but the construction admits a strictly richer per-weight refinement (generalizing Simsek et al., 2021). Details and the inverse transformation are in Appendix E.

P Gradients Under Splitting

Theorem P.1 (Gradient Scaling)**.** *Consider a splitting operation with coefficients $\alpha_1, \alpha_2, \ldots, \alpha_m$ summing to 1, where each coefficient multiplies the full vector of incoming or outgoing weights. Let D denote the set of duplicated weights. Then for the duplicated weight w'_i associated with coefficient α_i,*

$$\frac{\partial L}{\partial w'_i} = \alpha_i \frac{\partial L}{\partial w}.$$

The gradients of all other weights, including the split weights, are unchanged.

Corollary P.2 (Even Split)**.** *For an order-m even split (all $\alpha_i = 1/m$), the gradient of each duplicated weight is $\frac{1}{m}\frac{\partial L}{\partial w}$.*

For a set of transformations $\{M_l, M_l^+\}_{l=1\ldots L}$ applied at width-n layers $\{W_l\}_{l=1\ldots L}$, the gradient vector $\theta = \mathrm{vec}(\{W_l\})$ transforms as $\nabla L_{\theta'} = B^\top \nabla L_\theta$ where $B^\top = \mathrm{diag}(I \otimes (M_1^+)^\top, \ldots, M_{L-1} \otimes I)$ is a rectangular block-diagonal matrix obtained by vectorizing the expanded weights and differentiating. Full backpropagation derivations and the corresponding pre-activation argument are in Appendix C.

Q Hessians Under Splitting

Let $H \in \mathbb{R}^{a\times a}$ be the full-rank Hessian of the original parameterization and let $H' \in \mathbb{R}^{b\times b}$ be the expanded Hessian after an expansion from $a \to b$ weights. Differentiating the gradient at a global minimum gives $H' = B^\top HB$ where B is the

gradient-mapping matrix from Appendix P. Writing S for the split set, D for the duplicated set, and O for the other weights, an order-m even split has the explicit block decomposition

$$H' = \begin{bmatrix} H_{OO} & H'_{OS} & \frac{1}{m}H'_{OD} \\ {H'_{OS}}^{\top} & H'_{SS} & \frac{1}{m}H'_{SD} \\ \frac{1}{m}{H'_{OD}}^{\top} & \frac{1}{m}{H'_{SD}}^{\top} & \frac{1}{m^2}H'_{DD} \end{bmatrix},$$

where $H'_{XX} = H_{XX} \otimes \mathbf{1}_{m\times m}$ duplicates the rows and columns of the corresponding block. The full m-way derivation, including uneven splits, is in Appendix F.

Example: Hessian Transformation

For a 1-hidden-layer network with 1 hidden neuron, 1 input weight w and 1 output weight a, duplicating a and splitting w with coefficients $(\alpha, 1-\alpha)$ gives (writing $\bar{\alpha} := 1-\alpha$)

$$H = \begin{bmatrix} h_{ww} & h_{wa} \\ h_{wa} & h_{aa} \end{bmatrix} \rightarrow H' = \begin{bmatrix} h_{ww} & h_{ww} & \alpha h_{wa} & \bar{\alpha} h_{wa} \\ h_{ww} & h_{ww} & \alpha h_{wa} & \bar{\alpha} h_{wa} \\ \alpha h_{wa} & \alpha h_{wa} & \alpha^2 h_{aa} & \alpha\bar{\alpha} h_{aa} \\ \bar{\alpha} h_{wa} & \bar{\alpha} h_{wa} & \alpha\bar{\alpha} h_{aa} & \bar{\alpha}^2 h_{aa} \end{bmatrix}.$$

The two duplicated rows/columns are everywhere scaled by α and $\bar{\alpha}$ respectively, tracing back to the gradient scaling in Theorem P.1.

R Extended Discussion

Empirical corroboration of the existence picture. Figure 13 shows that the small gap between the mean and the best student conditioning suggests the existence of well-conditioned minima improves practical convergence: optimization algorithms naturally bypass poorly-conditioned minima with small volume, even though such minima do exist.

Limitations and future work. Our analysis is restricted to width-wise symmetries; extending the framework to depth-wise symmetries is a natural next direction. The relation $H' = B^{\top}HB$ also assumes zero-loss minima, so a perturbative analysis of the Hessian around small non-zero loss would extend our conditioning and volume results to the mildly underparameterized regime probed in Figure 5. Finally, our analysis is primarily about the existence of good minima; we hope to connect the theory to practical training dynamics in future work.

Wideness of minima. There is an old idea in machine learning that generalizing minima are wide (Hochreiter & Schmidhuber, 1997). Considering parameter-space symmetries it is clear this picture is incomplete (Dinh et al., 2017): using symmetry transformations we can construct narrow minima that generalize. A more plausible reading is that the implication runs in the other direction: wide minima represent simpler functions, and simpler functions generalize. The wonderful coincidence of deep learning is that these generalizing minima also happen to be favored by the optimization process.

Underparameterized regime. Even when a network is not overparameterized, low-width subnetworks of the loss landscape correspond to regions of weight space representing simpler functions (Figure 15). Figure 5 (Bottom) provides a first look at this regime, where the predicted trends continue to hold. An exciting direction for future work is a perturbative analysis around H' at global minima, since prior work has shown even underparameterized networks have a proliferation of near-zero loss local minima.