Get rich quick: exact solutions reveal how unbalanced initializations promote rapid feature learning

Daniel Kunin* KUNIN@STANFORD.EDU
Stanford University

Allan Raventós* ARAVENTO@STANFORD.EDU
Stanford University

Clémentine Dominé CLEMENTINE.DOMINE98@GMAIL.COM
University College London

Feng Chen FENGC@STANFORD.EDU
Stanford University

David Klindt KLINDT@CSHL.EDU
Cold Spring Harbor Laboratory

Andrew Saxe A.SAXE@UCL.AC.UK
University College London

Surya Ganguli SGANGULI@STANFORD.EDU
Stanford University

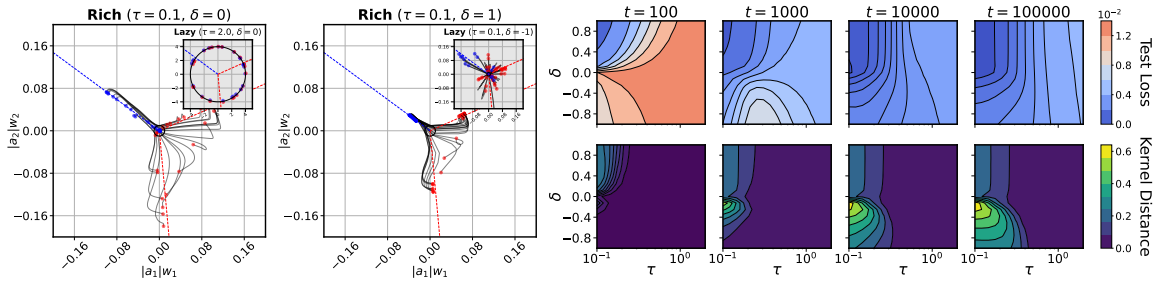
Abstract

While the impressive performance of modern neural networks is often attributed to their capacity to efficiently extract task-relevant features from data, the mechanisms underlying this *rich feature learning regime* remain elusive. In this work, we derive exact solutions to a minimal model that transitions between lazy and rich learning, precisely elucidating how unbalanced *layer-specific* initialization variances and learning rates conspire to influence the degree of feature learning through a set of conserved quantities that constrain and modify the geometry of learning trajectories. We extend our analysis to more complex linear models and to shallow nonlinear networks with piecewise linear activation functions. In linear networks, rapid feature learning only occurs with balanced initializations, while in nonlinear networks, unbalanced initializations that promote faster learning in earlier layers can accelerate rich learning. Through a series of experiments, we provide evidence that this unbalanced rich regime drives feature learning in deep finite-width networks, promotes interpretability of early layers in CNNs, reduces the sample complexity of learning hierarchical data, and decreases the time to grokking in modular arithmetic.

1. Introduction

It is widely believed that the impressive performance of deep learning models lies in their capacity to efficiently extract task-relevant features from data. However, understanding this feature acquisition requires unraveling a complex interplay between datasets, architectures, and optimization algorithms. Within this framework, two distinct regimes have emerged: lazy and rich.

* Equal contribution.



(*a*) Effect of scale (*b*) Effect of geometry (*c*) A complex phase portrait of feature learning

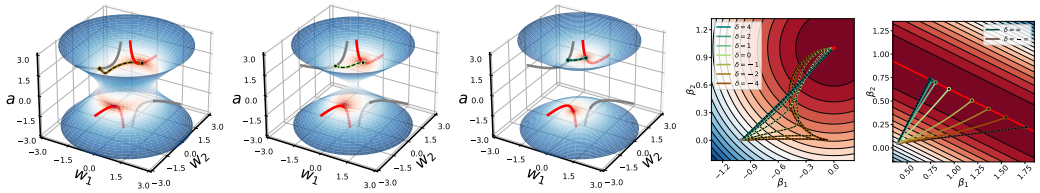
Figure 1: **Unbalanced initializations lead to rapid rich learning and generalization.** We follow the same setup used in Fig. 1 of Chizat et al. [14]: a wide two-layer ReLU student network $f(x;\theta) = \sum_{i=1}^{h} a_i \max\{0, w_i^\intercal x\}$ trained on labels generated by a narrow two-layer teacher. (a) and (b) show the training trajectories of $|a_i|w_i$: (a) small scale leads to rich and large scale to lazy, as in [14]; (b) even at small scale, initialization geometry can move the network between rich and lazy. (c) shows test loss and kernel distance from initialization computed through training over a sweep of τ and δ.

Lazy regime. Seminal work by Jacot et al. [30] demonstrated that in the infinite-width limit, the Neural Tangent Kernel (NTK) converges to a deterministic limit, making the learning dynamics akin to kernel regression. This *lazy* or *kernel* regime has been characterized by convex dynamics with minimal movement in parameter space, static hidden representations, exponential learning curves, and implicit biases aligned with a reproducing kernel Hilbert space norm [1, 2, 16, 18, 75]. Chizat et al. [14] showed that the lazy regime is contingent on the *scale* of the network at initialization, but may have worse generalization error. While the lazy regime offers insights into the network's convergence to a global minimum, it does not fully capture the generalization capabilities.

Rich regime. In contrast to the lazy regime, the *rich* or *feature-learning* or *active* regime is distinguished by a learned NTK that evolves through training, non-convex dynamics traversing between saddle points [31, 60, 61], sigmoidal learning curves, and simplicity biases such as low-rankness [40] or sparsity [68]. Recent analyses have shown that beyond scale, other aspects of the initialization can substantially impact the extent of feature learning, such as the effective rank [42], layer-specific initialization variances [43, 70, 71], and large learning rates [8, 15, 39, 73]. However, as shown in Fig. 1, for nonlinear networks, unbalanced initializations can induce both rich and lazy dynamics, creating a complex phase portrait of learning regimes influenced by both scale and geometry. Building on these observations, our study aims to precisely understand how layer-specific initialization variances and learning rates determine the transition between lazy and rich learning in finite-width networks, as well as the inductive biases of both regimes.

2. A Minimal Model of Lazy and Rich Learning with Exact Solutions

We explore an illustrative setting simple enough to admit exact gradient flow dynamics, yet complex enough to showcase lazy and rich learning regimes. We study a two-layer linear network with a single hidden neuron defined by the map $f(x;\theta) = aw^\intercal x$ where $a \in \mathbb{R}$, $w \in \mathbb{R}^d$ are the parameters. We examine how the parameter initializations a_0, w_0 and the layer-wise learning rates η_a, η_w influence the training trajectory in parameter space, function space ($\beta = aw$), and the evolution of the the NTK matrix $K = X\left(\eta_w a^2 I_d + \eta_a w w^\intercal\right) X^\intercal$. Even when the global minimum β_* is unique, the rescaling symmetry between a and w results in a manifold of minima in parameter space, which is a one-dimensional hyperbola and has two distinct branches for posi-



(*a*) $\delta = -2$ (*b*) $\delta = 0$ (*c*) $\delta = 2$ (*d*) Whitened (*e*) Low-Rank

Figure 2: Conservation of δ constrains the trajectory to (a) a one-sheeted hyperboloid for $\delta < 0$, (b) a double cone for $\delta = 0$, and (c) a two-sheeted hyperboloid for $\delta > 0$. Dashed lines indicate theory; minima are shown in red and equivalent β_0 initializations in gray. The surface is colored according to training loss (blue higher and red lower). When $X^\intercal X$ is whitened (d), we can solve for $\beta(t)$ (black dashed lines). When $X^\intercal X$ is low-rank (e), we can solve for the final solution (black dots) if the interpolating manifold is one-dimensional (Theorem 1) and $\beta(t)$ if $\delta = \pm\infty$.

tive and negative a. The symmetry also imposes a constraint on the trajectory, maintaining the difference $\delta = \eta_w a^2 - \eta_a \|w\|^2 \in \mathbb{R}$ throughout training, see Fig. 2 and Appendix B.1 for details. An upstream initialization occurs when $\delta > 0$, a balanced initialization when $\delta = 0$, and a downstream initialization when $\delta < 0$. For whitened input $X^\intercal X = \mathbf{I}_d$, the gradient flow dynamics, $\dot{a} = \eta_a \left(w^\intercal \beta_* - a\|w\|^2\right)$, $\dot{w} = \eta_w \left(a\beta_* - a^2 w\right)$, can be solved exactly, as discussed in Appendix B.2, and shown in Fig. 5.

Alternatively, we can study the influence of the initialization geometry by examining the dynamics in function space ($\beta = aw$), which is governed by the ODE,

$$\dot{\beta} = -\underbrace{\left(\eta_w a^2 I_d + \eta_a w w^\intercal\right)}_{M} \left(X^\intercal X \beta - X^\intercal y\right). \tag{1}$$

Notice that the matrix M also characterizes the NTK matrix, $K = XMX^\intercal$, so that understanding the evolution of M offers a method to discern between lazy and rich learning. At all $\beta \neq 0$, we can express $M = \frac{\kappa+\delta}{2} I_d + \frac{\kappa-\delta}{2} \frac{\beta\beta^\intercal}{\|\beta\|^2}$, where $\kappa = \sqrt{\delta^2 + 4\eta_a \eta_w \|\beta\|^2}$. (See Appendix B.3 for a derivation.) ***Upstream:*** When $\delta \gg 0$, $M \approx \delta \mathbf{I}_d$. Here the dynamics of β converge to the trajectory of linear regression. Along this trajectory, the NTK matrix remains constant, confirming the dynamics are lazy. ***Balanced:*** When $\delta = 0$, $M = \sqrt{\eta_a \eta_w}\|\beta\|(\mathbf{I}_d + \frac{\beta\beta^\intercal}{\|\beta\|^2})$. Here the dynamics balance between following the lazy trajectory and attempting to fit the task by only changing in norm. As a result, the NTK changes in both magnitude and direction through training, so the dynamics are rich. ***Downstream:*** When $\delta \ll 0$, $M \approx |\delta| \frac{\beta\beta^\intercal}{\|\beta\|^2}$ and β follows projected gradient descent. Along this trajectory, the NTK matrix doesn't evolve. However, if β_0 is not aligned to β_*, then at some point the dynamics of β will slowly align. In this second alignment phase, the NTK matrix will change, so the dynamics are initially lazy followed by a delayed rich phase.

Determining the implicit bias via mirror flow. So far we have considered that $X^\intercal X$ is whitened, ensuring the existence of a unique least squares solution β_*. Now we consider the case when $X^\intercal X$ is low-rank so that there exist infinitely many interpolating solutions in function space. By studying the structure of M, we can characterize how δ determines the interpolating solution the dynamics converge to. Extending a time-warped mirror flow analysis strategy pioneered by Azulay et al. [7] to the case of $\delta < 0$ (see Appendix B.4), we prove the following theorem, which shows a tradeoff between the minimum norm solution and preserving the direction of the initialization β_0.

Theorem 1 *For a single hidden neuron linear network, for all $\delta \in \mathbb{R}$, and initialization β_0 such that $\|\beta(t)\| > 0, \forall t \geq 0$, if the gradient flow solution $\beta(\infty)$ satisfies $X\beta(\infty) = y$, then,*

$$\beta(\infty) = \arg\min_{\beta \in \mathbb{R}^d} \Psi_\delta(\beta) - \psi_\delta \frac{\beta_0}{\|\beta_0\|}^\intercal \beta \quad \text{s.t.} \quad X\beta = y \tag{2}$$

where $\Psi_\delta(\beta) = \frac{1}{3}\left(\sqrt{\delta^2 + 4\|\beta\|^2} - 2\delta\right)\sqrt{\sqrt{\delta^2 + 4\|\beta\|^2} + \delta}$ *and* $\psi_\delta = \sqrt{\sqrt{\delta^2 + 4\|\beta_0\|^2} - \delta}$.

Generalization to wide linear networks. The advantage of studying the learning dynamics in function space is that the results can be generalized to wide linear networks with multiple outputs as shown in the following lemma.

Lemma 2 *We consider the dynamics of a two-layer linear network with h hidden neurons and c outputs, $f(x;\theta) = A^\intercal W x$, where $W \in \mathbb{R}^{h\times d}$ and $A \in \mathbb{R}^{h\times c}$. Denote $\beta_i = w_i a_i^\intercal \in \mathbb{R}^{d\times c}$ and assume $\|\beta_i\|_F \neq 0$ for all $i \in [h]$. Then, the gradient flow dynamics can be written as,*

$$\mathrm{vec}\left(\dot{\beta}\right) = -M\mathrm{vec}(X^\intercal X\beta - X^\intercal Y), \tag{3}$$

where $M = \sum_{i=1}^h M_i$ *and* $M_i = \left(\frac{\kappa_i+\delta_i}{2}\right)\frac{\beta_i^\intercal \beta_i}{\|\beta_i\|_F^2} \oplus \left(\frac{\kappa_i-\delta_i}{2}\right)\frac{\beta_i\beta_i^\intercal}{\|\beta_i\|_F^2}$, *with* $\kappa_i = \sqrt{\delta_i^2 + 4\eta_A\eta_W\|\beta_i\|_F^2}$.

With additional restrictions on the initialization, we can obtain a more general version of Theorem 1 for wide or deep linear networks (Theorem 10 and Theorem 20 in Appendix C).

3. Piecewise Linear Networks

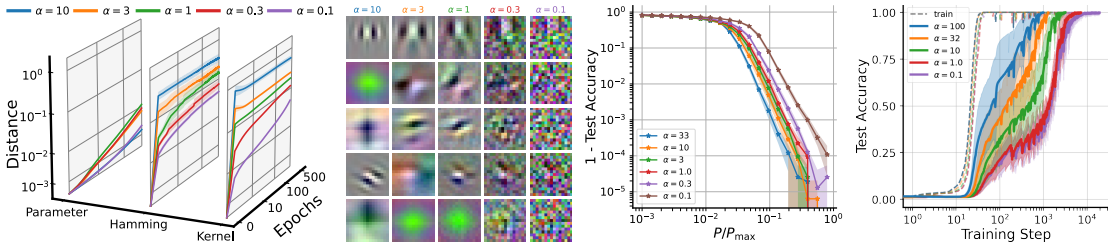
We now take a first step to extend our linear analysis to piecewise linear networks with activation functions of the form $\sigma(z) = \max\{z, \gamma z\}$. We consider the dynamics of a two-layer piecewise linear network without biases, $f(x;\theta) = a^\intercal\sigma(Wx)$, where $W \in \mathbb{R}^{h\times d}$ and $a \in \mathbb{R}^h$. As in the linear setting, each hidden neuron is associated with a conserved quantity, $\delta_i = \eta_w a_i^2 - \eta_a\|w_i\|^2$ [17]. However, unlike the linear setting, the neuron's contribution to the output $f(x_j;\theta)$ is regulated by whether the input x_j is in the neuron's active halfspace. Let $C \in \mathbb{R}^{h\times n}$ be the matrix of $c_{ij} = \sigma'(w_i^\intercal x_j)$ and let $\rho_j = f(x_j;\theta) - y_j$. Then, we can express the dynamics of β_i as,

$$\dot{\beta}_i = -\underbrace{\left(a_i^2 I_d + w_i w_i^\intercal\right)}_{M_i}\underbrace{\sum_{j=1}^n c_{ij}x_j\rho_j}_{\xi_i}. \tag{4}$$

Unlike in the linear setting, ξ_i is not shared for all neurons because of its dependence on c_{ij}. Additionally, the NTK matrix depends on M_i and C, with elements $K_{jk} = \sum_{i=1}^h c_{ij}x_j^\intercal M_i x_k c_{ik}$. We consider a *signed spherical coordinate* transformation separating the dynamics of β_i into its directional $\hat{\beta}_i = \mathrm{sgn}(a_i)\frac{\beta_i}{\|\beta_i\|}$ and radial $\mu_i = \mathrm{sgn}(a_i)\|\beta_i\|$ components, such that $\beta_i = \mu_i\hat{\beta}_i$. Here, $\hat{\beta}_i$ determines the direction and orientation of the halfspace where the i^{th} neuron is active, while μ_i determines the slope of the contribution in this halfspace. These coordinates evolve according to,

$$\dot{\mu}_i = -\sqrt{\delta_i^2 + 4\eta_a\eta_w\mu_i^2}\,\hat{\beta}_i^\intercal\xi_i, \qquad \dot{\hat{\beta}}_i = -\frac{\sqrt{\delta_i^2 + 4\eta_a\eta_w\mu_i^2} + \delta_i}{2\mu_i}\left(I_d - \hat{\beta}_i\hat{\beta}_i^\intercal\right)\xi_i. \tag{5}$$

Downstream. When $\delta_i \ll 0$, $M_i \approx |\delta_i|\hat{\beta}_i\hat{\beta}_i^\intercal$ and the dynamics are approximately $\partial_t\hat{\beta}_i = 0$ and $\partial_t\mu_i = -|\delta_i|\hat{\beta}_i^\intercal\xi_i$. Regardless of ξ_i, $\hat{\beta}_i(t) = \hat{\beta}_i(0)$, which implies the overall partition map cannot



(*a*) Kernel Distance (*b*) CNN Filters (*c*) Random Hierarchy (*d*) Grokking

Figure 3: **Impact of upstream initializations in practice.** Here we provide evidence that an upstream initialization (a) drives feature learning through changing activation patterns, (b) promotes interpretability of early layers in CNNs, (c) reduces the sample complexity of learning hierarchical data, and (d) decreases the time to grokking in modular arithmetic. Here, $\alpha \gg 1$ and $\alpha \ll 1$ correspond to upstream and downstream initializations, respectively. See Appendix E.3 for details.

change, nor the activation patterns C, nor M_i. Thus, the NTK remains constant, resulting in lazy learning. If there is an insufficient number of a hidden neurons to fit the data, then there will be a second, slow rich alignment phase. As in the linear case, the magnitude of δ_i will control the delay.
Balanced. When $\delta_i = 0$, $M_i = \sqrt{\eta_a \eta_w}|\mu_i|(\mathbf{I}_d + \hat{\beta}_i\hat{\beta}_i^\intercal)$ and the dynamics simplify to, $\partial_t \hat{\beta}_i = -\sqrt{\eta_a\eta_w}\text{sgn}(\mu_i)(\mathbf{I}_d - \hat{\beta}_i\hat{\beta}_i^\intercal)\xi_i$ and $\partial_t \mu_i = -2\sqrt{\eta_a\eta_w}|\mu_i|\hat{\beta}_i^\intercal\xi_i$. For vanishing initializations where $\|\beta_i(0)\| \to 0$ for all i, we can decouple the dynamics into two distinct phases of training. *Partition alignment.* At vanishing scale, the output $f(x;\theta_0) \approx 0$ for all input x, so that $\xi_i \approx -\sum_{j=1}^n c_{ij}x_j y_j$, independent of the other hidden neurons. Radial dynamics slow down relative to directional dynamics, and the function's output will remain small as each neuron aligns decoupled from the rest. *Data fitting.* Eventually, the magnitudes for the β_i will have grown such that $f(x;\theta) \not\approx 0$ and thus the residual will depend on all β_i. In this phase, the radial dynamics dominate the learning driving the network to fit the data. However, it is possible for directions to continue to change.
Upstream. When $\delta_i \gg 0$, the matrix $M_i \approx \delta_i \mathbf{I}_d$ and the dynamics are approximately $\partial_t \hat{\beta}_i = -\delta_i \mu_i^{-1}(\mathbf{I}_d - \hat{\beta}_i\hat{\beta}_i^\intercal)\xi_i$ and $\partial_t \mu_i = -\delta_i \hat{\beta}_i^\intercal \xi_i$. Unlike in the balanced setting, here M_i is independent of β_i and stays constant through training. Yet, as the β_i change in direction, so can C, and thus the NTK. This setting is unique, because it is rich due to a changing activation pattern, but the dynamics do not move far in parameter space. Furthermore, unlike in the balanced scenario where scale adjusts the speed of radial dynamics, here it regulates the speed of directional dynamics, with vanishing initializations prompting an extremely fast alignment phase, as observed in Fig. 1.

Unbalanced initializations in diverse domains. In our analysis, we find that upstream initializations can lead to rapid rich learning in nonlinear networks, explaining results shown in Fig. 1. Further experiments in Fig. 3 suggest that upstream initializations have an impact across various domains of deep learning: (a) Standard initializations see significant NTK evolution early in training [19]. We show the movement is linked to changes in activation patterns (Hamming distance) rather than large parameter shifts. Adjusting the initialization variance of the first and last layers can amplify or diminish this movement. (b) Filters in CNNs trained on image classification tasks often align with edge detectors [33]. We show that adjusting the learning speed of the first layer can enhance or degrade this alignment. (c) Deep learning models are believed to avoid the curse of dimensionality and learn with limited data by exploiting hierarchical structures in real-world tasks.

Using the Random Hierarchy Model [55] as a framework for synthetic hierarchical tasks, we show that modifying the initialization geometry can decrease or increase the sample complexity of learning. (d) Networks trained on simple modular arithmetic tasks will suddenly generalize long after memorizing their training data [57]. This behavior, termed grokking, is thought to result from a transition from lazy to rich learning [34, 46] and believed to be important towards understanding emergent phenomena [53]. We show that decreasing the variance of the embedding in a single-layer transformer ($< 6\%$ of all parameters) significantly reduces the time to grokking. Overall, our experiments suggest that upstream initializations may play a crucial role in neural network behaviors.

4. Conclusion

We derived exact solutions to a minimal model that can transition between lazy and rich learning to precisely elucidate how unbalanced layer-specific initialization variances and learning rates determine the degree of feature learning. We extended our analysis to wide and deep linear networks and to shallow piecewise linear networks. We find through theory and empirics that unbalanced initializations, which promote faster learning at earlier layers, can actually accelerate rich learning.

References

[1] Zeyuan Allen-Zhu, Yuanzhi Li, and Yingyu Liang. Learning and generalization in overparameterized neural networks, going beyond two layers. *Advances in neural information processing systems*, 32, 2019.

[2] Zeyuan Allen-Zhu, Yuanzhi Li, and Zhao Song. A convergence theory for deep learning via over-parameterization. In *International conference on machine learning*, pages 242–252. PMLR, 2019.

[3] Raman Arora, Amitabh Basu, Poorya Mianjy, and Anirbit Mukherjee. Understanding deep neural networks with rectified linear units. *arXiv preprint arXiv:1611.01491*, 2016.

[4] Sanjeev Arora, Nadav Cohen, and Elad Hazan. On the optimization of deep networks: Implicit acceleration by overparameterization. In *International conference on machine learning*, pages 244–253. PMLR, 2018.

[5] Sanjeev Arora, Nadav Cohen, Wei Hu, and Yuping Luo. Implicit regularization in deep matrix factorization. *Advances in Neural Information Processing Systems*, 32, 2019.

[6] Alexander Atanasov, Blake Bordelon, and Cengiz Pehlevan. Neural networks as kernel learners: The silent alignment effect. In *International Conference on Learning Representations*, 2021.

[7] Shahar Azulay, Edward Moroshko, Mor Shpigel Nacson, Blake E Woodworth, Nathan Srebro, Amir Globerson, and Daniel Soudry. On the implicit bias of initialization shape: Beyond infinitesimal mirror descent. In *International Conference on Machine Learning*, pages 468–477. PMLR, 2021.

[8] Jimmy Ba, Murat A Erdogdu, Taiji Suzuki, Zhichao Wang, Denny Wu, and Greg Yang. High-dimensional asymptotics of feature learning: How one gradient step improves the representation. *Advances in Neural Information Processing Systems*, 35:37932–37946, 2022.

[9] Blake Bordelon and Cengiz Pehlevan. Self-consistent dynamical field theory of kernel evolution in wide neural networks. *Advances in Neural Information Processing Systems*, 35: 32240–32256, 2022.

[10] Etienne Boursier, Loucas Pillaud-Vivien, and Nicolas Flammarion. Gradient flow dynamics of shallow relu networks for square loss and orthogonal inputs. *Advances in Neural Information Processing Systems*, 35:20105–20118, 2022.

[11] Lukas Braun, Clémentine Carla Juliette Dominé, James E Fitzgerald, and Andrew M Saxe. Exact learning dynamics of deep linear networks with prior knowledge. In *Advances in Neural Information Processing Systems*, 2022.

[12] Lenaic Chizat and Francis Bach. On the global convergence of gradient descent for over-parameterized models using optimal transport. *Advances in neural information processing systems*, 31, 2018.

[13] Lenaic Chizat and Francis Bach. Implicit bias of gradient descent for wide two-layer neural networks trained with the logistic loss. In *Conference on learning theory*, pages 1305–1338. PMLR, 2020.

[14] Lenaic Chizat, Edouard Oyallon, and Francis Bach. On lazy training in differentiable programming. *Advances in neural information processing systems*, 32, 2019.

[15] Hugo Cui, Luca Pesce, Yatin Dandi, Florent Krzakala, Yue M Lu, Lenka Zdeborová, and Bruno Loureiro. Asymptotics of feature learning in two-layer networks after one gradient-step. *arXiv preprint arXiv:2402.04980*, 2024.

[16] Simon Du, Jason Lee, Haochuan Li, Liwei Wang, and Xiyu Zhai. Gradient descent finds global minima of deep neural networks. In *International conference on machine learning*, pages 1675–1685. PMLR, 2019.

[17] Simon S Du, Wei Hu, and Jason D Lee. Algorithmic regularization in learning deep homogeneous models: Layers are automatically balanced. *Advances in Neural Information Processing Systems*, 31, 2018.

[18] Simon S Du, Xiyu Zhai, Barnabas Poczos, and Aarti Singh. Gradient descent provably optimizes over-parameterized neural networks. *arXiv preprint arXiv:1810.02054*, 2018.

[19] Stanislav Fort, Gintare Karolina Dziugaite, Mansheej Paul, Sepideh Kharaghani, Daniel M Roy, and Surya Ganguli. Deep learning versus kernel learning: an empirical study of loss landscape geometry and the time evolution of the neural tangent kernel. *Advances in Neural Information Processing Systems*, 33:5850–5861, 2020.

[20] Spencer Frei, Niladri S Chatterji, and Peter L Bartlett. Random feature amplification: Feature learning and generalization in neural networks. *Journal of Machine Learning Research*, 24 (303):1–49, 2023.

[21] Kenji Fukumizu. Effect of batch learning in multilayer neural networks. *Gen*, 1(04):1E–03, 1998.

[22] Gauthier Gidel, Francis Bach, and Simon Lacoste-Julien. Implicit regularization of discrete gradient dynamics in linear neural networks. *Advances in Neural Information Processing Systems*, 32, 2019.

[23] Daniel Gissin, Shai Shalev-Shwartz, and Amit Daniely. The implicit bias of depth: How incremental learning drives generalization. *arXiv preprint arXiv:1909.12051*, 2019.

[24] Suriya Gunasekar, Jason Lee, Daniel Soudry, and Nathan Srebro. Characterizing implicit bias in terms of optimization geometry. In *International Conference on Machine Learning*, pages 1832–1841. PMLR, 2018.

[25] Suriya Gunasekar, Jason D Lee, Daniel Soudry, and Nati Srebro. Implicit bias of gradient descent on linear convolutional networks. *Advances in neural information processing systems*, 31, 2018.

[26] Suriya Gunasekar, Blake Woodworth, and Nathan Srebro. Mirrorless mirror descent: A natural derivation of mirror descent. In *International Conference on Artificial Intelligence and Statistics*, pages 2305–2313. PMLR, 2021.

[27] Boris Hanin and David Rolnick. Complexity of linear regions in deep networks. In *International Conference on Machine Learning*, pages 2596–2604. PMLR, 2019.

[28] Boris Hanin and David Rolnick. Deep relu networks have surprisingly few activation patterns. *Advances in neural information processing systems*, 32, 2019.

[29] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034, 2015.

[30] Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. *Advances in neural information processing systems*, 31, 2018.

[31] Arthur Jacot, François Ged, Berfin Şimşek, Clément Hongler, and Franck Gabriel. Saddle-to-saddle dynamics in deep linear networks: Small initialization training, symmetry, and sparsity. *arXiv preprint arXiv:2106.15933*, 2021.

[32] Ziwei Ji and Matus Telgarsky. Gradient descent aligns the layers of deep linear networks. *arXiv preprint arXiv:1810.02032*, 2018.

[33] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90, 2017.

[34] Tanishq Kumar, Blake Bordelon, Samuel J Gershman, and Cengiz Pehlevan. Grokking as the transition from lazy to rich training dynamics. *arXiv preprint arXiv:2310.06110*, 2023.

[35] Daniel Kunin, Javier Sagastuy-Brena, Surya Ganguli, Daniel LK Yamins, and Hidenori Tanaka. Neural mechanics: Symmetry and broken conservation laws in deep learning dynamics. *arXiv preprint arXiv:2012.04728*, 2020.

[36] Daniel Kunin, Atsushi Yamamura, Chao Ma, and Surya Ganguli. The asymmetric maximum margin bias of quasi-homogeneous neural networks. *arXiv preprint arXiv:2210.03820*, 2022.

[37] Andrew K Lampinen and Surya Ganguli. An analytic theory of generalization dynamics and transfer learning in deep linear networks. *arXiv preprint arXiv:1809.10374*, 2018.

[38] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[39] Aitor Lewkowycz, Yasaman Bahri, Ethan Dyer, Jascha Sohl-Dickstein, and Guy Gur-Ari. The large learning rate phase of deep learning: the catapult mechanism. *arXiv preprint arXiv:2003.02218*, 2020.

[40] Zhiyuan Li, Yuping Luo, and Kaifeng Lyu. Towards resolving the implicit bias of gradient descent for matrix factorization: Greedy low-rank learning. *arXiv preprint arXiv:2012.09839*, 2020.

[41] Zhiyuan Li, Tianhao Wang, Jason D Lee, and Sanjeev Arora. Implicit bias of gradient descent on reparametrized models: On equivalence to mirror descent. *Advances in Neural Information Processing Systems*, 35:34626–34640, 2022.

[42] Yuhan Helena Liu, Aristide Baratin, Jonathan Cornford, Stefan Mihalas, Eric Shea-Brown, and Guillaume Lajoie. How connectivity structure shapes rich and lazy learning in neural circuits. *ArXiv*, 2023.

[43] Tao Luo, Zhi-Qin John Xu, Zheng Ma, and Yaoyu Zhang. Phase diagram for two-layer relu neural networks at infinite-width limit. *Journal of Machine Learning Research*, 22(71):1–47, 2021.

[44] Kaifeng Lyu and Jian Li. Gradient descent maximizes the margin of homogeneous neural networks. *arXiv preprint arXiv:1906.05890*, 2019.

[45] Kaifeng Lyu, Zhiyuan Li, Runzhe Wang, and Sanjeev Arora. Gradient descent on two-layer nets: Margin maximization and simplicity bias. *Advances in Neural Information Processing Systems*, 34, 2021.

[46] Kaifeng Lyu, Jikai Jin, Zhiyuan Li, Simon Shaolei Du, Jason D Lee, and Wei Hu. Dichotomy of early and late phase implicit biases can provably induce grokking. In *The Twelfth International Conference on Learning Representations*, 2023.

[47] Hartmut Maennel, Olivier Bousquet, and Sylvain Gelly. Gradient descent quantizes relu network features. *arXiv preprint arXiv:1803.08367*, 2018.

[48] Song Mei, Andrea Montanari, and Phan-Minh Nguyen. A mean field view of the landscape of two-layer neural networks. *Proceedings of the National Academy of Sciences*, 115(33): E7665–E7671, 2018.

[49] Hancheng Min, René Vidal, and Enrique Mallada. Early neuron alignment in two-layer relu networks with small initialization. *arXiv preprint arXiv:2307.12851*, 2023.

[50] Guido F Montufar, Razvan Pascanu, Kyunghyun Cho, and Yoshua Bengio. On the number of linear regions of deep neural networks. *Advances in neural information processing systems*, 27, 2014.

[51] Edward Moroshko, Blake E Woodworth, Suriya Gunasekar, Jason D Lee, Nati Srebro, and Daniel Soudry. Implicit bias in deep linear classification: Initialization scale vs training accuracy. *Advances in neural information processing systems*, 33:22182–22193, 2020.

[52] Mor Shpigel Nacson, Suriya Gunasekar, Jason Lee, Nathan Srebro, and Daniel Soudry. Lexicographic and depth-sensitive margins in homogeneous and non-homogeneous deep models. In *International Conference on Machine Learning*, pages 4683–4692. PMLR, 2019.

[53] Neel Nanda, Lawrence Chan, Tom Lieberum, Jess Smith, and Jacob Steinhardt. Progress measures for grokking via mechanistic interpretability. *arXiv preprint arXiv:2301.05217*, 2023.

[54] Razvan Pascanu, Guido Montufar, and Yoshua Bengio. On the number of response regions of deep feed forward networks with piece-wise linear activations. *arXiv preprint arXiv:1312.6098*, 2013.

[55] Leonardo Petrini, Francesco Cagnetta, Umberto M Tomasini, Alessandro Favero, and Matthieu Wyart. How deep neural networks learn compositional data: The random hierarchy model. *arXiv preprint arXiv:2307.02129*, 2023.

[56] Mary Phuong and Christoph H Lampert. The inductive bias of relu networks on orthogonally separable data. In *International Conference on Learning Representations*, 2020.

[57] Alethea Power, Yuri Burda, Harri Edwards, Igor Babuschkin, and Vedant Misra. Grokking: Generalization beyond overfitting on small algorithmic datasets. *arXiv preprint arXiv:2201.02177*, 2022.

[58] Maithra Raghu, Ben Poole, Jon Kleinberg, Surya Ganguli, and Jascha Sohl-Dickstein. On the expressive power of deep neural networks. In *international conference on machine learning*, pages 2847–2854. PMLR, 2017.

[59] Grant Rotskoff and Eric Vanden-Eijnden. Trainability and accuracy of artificial neural networks: An interacting particle system approach. *Communications on Pure and Applied Mathematics*, 75(9):1889–1935, 2022.

[60] Andrew M Saxe, James L McClelland, and Surya Ganguli. Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. *arXiv preprint arXiv:1312.6120*, 2013.

[61] Andrew M Saxe, James L McClelland, and Surya Ganguli. A mathematical theory of semantic development in deep neural networks. *Proceedings of the National Academy of Sciences*, 116 (23):11537–11546, 2019.

[62] Thiago Serra, Christian Tjandraatmadja, and Srikumar Ramalingam. Bounding and counting linear regions of deep neural networks. In *International Conference on Machine Learning*, pages 4558–4566. PMLR, 2018.

[63] Justin Sirignano and Konstantinos Spiliopoulos. Mean field analysis of neural networks: A law of large numbers. *SIAM Journal on Applied Mathematics*, 80(2):725–752, 2020.

[64] Daniel Soudry, Elad Hoffer, Mor Shpigel Nacson, Suriya Gunasekar, and Nathan Srebro. The implicit bias of gradient descent on separable data. *The Journal of Machine Learning Research*, 19(1):2822–2878, 2018.

[65] Salma Tarmoun, Guilherme Franca, Benjamin D Haeffele, and Rene Vidal. Understanding the dynamics of gradient flow in overparameterized linear models. In *International Conference on Machine Learning*, pages 10153–10161. PMLR, 2021.

[66] Matus Telgarsky. Representation benefits of deep feedforward networks. *arXiv preprint arXiv:1509.08101*, 2015.

[67] Gal Vardi and Ohad Shamir. Implicit regularization in relu networks with the square loss. In *Conference on Learning Theory*, pages 4224–4258. PMLR, 2021.

[68] Blake Woodworth, Suriya Gunasekar, Jason D Lee, Edward Moroshko, Pedro Savarese, Itay Golan, Daniel Soudry, and Nathan Srebro. Kernel and rich regimes in overparametrized models. In *Conference on Learning Theory*, pages 3635–3673. PMLR, 2020.

[69] Yizhou Xu and Liu Ziyin. When does feature learning happen? perspective from an analytically solvable model. *arXiv preprint arXiv:2401.07085*, 2024.

[70] Greg Yang and Edward J Hu. Feature learning in infinite-width neural networks. *arXiv preprint arXiv:2011.14522*, 2020.

[71] Greg Yang, Edward J Hu, Igor Babuschkin, Szymon Sidor, Xiaodong Liu, David Farhi, Nick Ryder, Jakub Pachocki, Weizhu Chen, and Jianfeng Gao. Tensor programs v: Tuning large neural networks via zero-shot hyperparameter transfer. *arXiv preprint arXiv:2203.03466*, 2022.

[72] Greg Yang, James B Simon, and Jeremy Bernstein. A spectral condition for feature learning. *arXiv preprint arXiv:2310.17813*, 2023.

[73] Libin Zhu, Chaoyue Liu, Adityanarayanan Radhakrishnan, and Mikhail Belkin. Catapults in sgd: spikes in the training loss and their impact on generalization through feature learning. *arXiv preprint arXiv:2306.04815*, 2023.

[74] Liu Ziyin, Botao Li, and Xiangming Meng. Exact solutions of a deep linear network. *Advances in Neural Information Processing Systems*, 35:24446–24458, 2022.

[75] Difan Zou, Yuan Cao, Dongruo Zhou, and Quanquan Gu. Gradient descent optimizes over-parameterized deep relu networks. *Machine learning*, 109:467–492, 2020.

Appendix A. Related Work

Linear networks. Significant progress in studying the rich regime has been achieved in the context of linear networks. In this setting, $f(x;\theta) = \beta(\theta)^\intercal x$ is linear in its input x, but can exhibit highly nonlinear dynamics in parameter θ and function $\beta(\theta)$ space. Foundational work by Saxe et al. [60] provided exact solutions to gradient flow dynamics in linear networks with task-aligned initializations. They achieved this by solving a system of Bernoulli differential equations that prioritize learning the most salient features first, which can be beneficial for generalization [37]. This analysis has been extended to wide [11, 21] and deep [4, 5, 74] linear networks with more flexible initialization schemes [22, 23, 65]. It has also been applied to study the evolution of the NTK [6] and the influence of the scale on the transition between lazy and rich learning [31, 69]. In this work, we present novel exact solutions for a minimal model utilizing a mix of Bernoulli and Riccati equations to showcase a complex phase portrait of lazy and rich learning with separate alignment and fitting phases.

Implicit bias. An effective analysis approach to understanding the rich regime studies how the initialization influences the inductive bias at interpolation. The aim is to identify a function $Q(\theta)$ such that the network converges to a first-order KKT point minimizing $Q(\theta)$ among all possible interpolating solutions. Foundational work by Soudry et al. [64] pioneered this approach for a linear classifier trained with gradient descent, revealing a max margin bias. These findings have been extended to deep linear networks [25, 32, 51], homogeneous networks [13, 44, 52], and quasi-homogeneous networks [36]. A similar line of research expresses the learning dynamics of networks trained with mean squared error as a *mirror flow* for some potential $\Phi(\beta)$, such that the inductive bias can be expressed as a *Bregman divergence* [24]. This approach has been applied to diagonal linear networks, revealing an inductive bias that interpolates between ℓ^1 and ℓ^2 norms in the rich and lazy regimes respectively [68]. However, finding the potential $\Phi(\beta)$ is problem-specific and requires solving a second-order differential equation, which may not be solvable even in simple settings [26, 41]. Azulay et al. [7] extended this analysis to a time-warped mirror flow, enabling the study of a broader class of architectures. In this work we derive exact expressions for the inductive bias of our minimal model and extend the results in Azulay et al. [7] to wide and deep linear networks.

Two-layer networks. Two-layer, or single-hidden layer, piecewise linear networks have emerged as a key setting for advancing our understanding of the rich regime. Maennel et al. [47] observed that in training two-layer ReLU networks from small initializations, the first-layer weights concentrate along fixed directions determined by the training data, irrespective of network width. This phenomenon, termed *quantization*, has been proposed as a *simplicity bias* inherent to the rich regime, driving the network towards low-rank solutions when feasible. Subsequent studies have aimed to further elucidate this effect by introducing structural constraints on the training data [10, 20, 45, 49, 56]. Across these analyses, a consistent observation is that the learning dynamics involve distinct phases: an initial alignment phase characterized by quantization, followed by a fitting phase where the task is learned. All of these studies assumed a balanced initialization between the first and second layer. In this study, we explore how unbalanced initializations influence the phases of learning, demonstrating that it can eliminate or augment the quantization effect.

Infinite-width networks. Many recent advancements in understanding the rich regime have come from studying how the initialization variance and layer-wise learning rates should scale in the infinite-width limit to ensure constant movement in the activations, gradients, and outputs. In

this limit, analyzing dynamics becomes simpler in several respects: random variables concentrate, nonlinearities act linearly, and quantities will either vanish to zero, remain constant, or diverge to infinity [43]. A set of works used tools from statistical mechanics to provide analytic solutions for the rich population dynamics of two-layer nonlinear neural networks initialized according to the *mean field* parameterization [12, 48, 59, 63]. These ideas were extended to deeper networks through a *tensor program* framework, leading to the derivation of *maximal update parametrization* (μP) [70, 71]. The μP parameterization has also been derived through a self-consistent dynamical mean field theory [9] and a spectral scaling analysis [72]. In this study, we focus on finite-width neural networks.

Appendix B. Single-Neuron Linear Network

In this section, we provide a detailed analysis of the two-layer linear network with a single hidden neuron discussed in Section 2. The network is defined by the function $f(x;\theta) = aw^\intercal x$, where $a \in \mathbb{R}$ and $w \in \mathbb{R}^d$ are the parameters. We aim to understand the impact of the initializations a_0, w_0 and the layer-wise learning rates η_a, η_w on the training trajectory in parameter space, function space (defined by the product $\beta = aw$), and the evolution of the Neural Tangent Kernel (NTK) matrix K:

$$K = X\left(\eta_w a^2 I_d + \eta_a w w^\intercal\right) X^\intercal. \tag{6}$$

The gradient flow dynamics are governed by the following coupled ODEs:

$$\dot{a} = -\eta_a w^\intercal \left(X^\intercal X a w - X^\intercal y\right), \qquad a(0) = a_0, \tag{7}$$

$$\dot{w} = -\eta_w a \left(X^\intercal X a w - X^\intercal y\right), \qquad w(0) = w_0. \tag{8}$$

The global minima of this problem are determined by the normal equations $X^\intercal X a w = X^\intercal y$. Even when $X^\intercal X$ is invertible, yielding a unique global minimum in the function space $\beta_* = (X^\intercal X)^{-1} X^\intercal y$, the symmetry between a and w, permitting scaling transformations, $a \to a\alpha$ and $w \to w/\alpha$ for any $\alpha \neq 0$ without changing the product aw, results in a manifold of minima in parameter space. This minima manifold is a one-dimensional hyperbola where $aw = \beta_*$, with two distinct branches for positive and negative a. The set of saddle-points $\{(a, w)\}$ forms a $(d-1)$-dimensional subspace satisfying $a = 0$ and $w^\intercal X^\intercal y = 0$. Except for a measure zero set of initializations that converge to the saddle points, all gradient flow trajectories will converge to a global minimum. In Appendix B.2.3, we detail the basin of attraction for each branch of the minima manifold and the d-dimensional surface of initializations that converge to saddle points, separating the two basins.

B.1. Conserved quantity

The symmetry between a and w results in a conserved quantity $\delta \in \mathbb{R}$ throughout training, as noted in many prior works [17, 35, 60], where

$$\delta = \eta_w a^2 - \eta_a \|w\|^2 \tag{9}$$

This can be directly checked as one writes out the dynamics of δ. Define $\rho = (X^\intercal X a w - X^\intercal y)$ for succinct notation, such that

$$\begin{aligned}
\dot{\delta} &= 2\eta_w a \dot{a} - 2\eta_a w^\intercal \dot{w} \\
&= 2\eta_w a \left(-\eta_a w^\intercal \rho\right) - 2\eta_a w^\intercal \left(-\eta_w a \rho\right) \\
&= 0
\end{aligned}$$

The conserved quantity confines the parameter dynamics to the surface of a hyperboloid where the magnitude and sign of the conserved quantity determines the geometry, as shown in Fig. 2. A hyperboloid of the form $\sum_{i=1}^{k} x_i^2 - \sum_{i=k+1}^{n} x_i^2 = \alpha$, with $\alpha \geq 0$, exhibits varied topology and geometry based on k and α. It has two sheets when $k = 1$ and one sheet otherwise. Its geometry is primarily dictated by α: as α tends to infinity, curvature decreases, while at $\alpha = 0$, a singularity occurs at the origin.

B.2. Exact solutions

To derive exact dynamics we assume the input data is whitened such that $X^\intercal X = I_d$ and $\beta_* = X^\intercal y$. The dynamics of a and w can then be simplified as

$$\dot{a} = \eta_a \left(w^\intercal \beta_* - a\|w\|^2 \right), \qquad a(0) = a_0 \tag{10}$$

$$\dot{w} = \eta_w \left(a\beta_* - a^2 w \right), \qquad w(0) = w_0. \tag{11}$$

B.2.1. Deriving the dynamics for μ and ϕ

As discussed in Section 2 we study the variables $\mu = a\|w\|$, an invariant under the rescale symmetry, and $\phi = \frac{w^\intercal \beta_*}{\|w\|\|\beta_*\|}$, the cosine of the angle between w and β_*. This change of variables can also be understood as a signed spherical decomposition of β: μ is the signed magnitude of β and ϕ is the cosine angle between β and β_*. Through chain rule, we obtain the dynamics for μ and ϕ, which can be expressed as

$$\dot{\mu} = \sqrt{\delta^2 + 4\eta_a\eta_w\mu^2}\left(\phi\|\beta_*\| - \mu\right), \qquad \mu(0) = a_0\|w_0\|, \tag{12}$$

$$\dot{\phi} = \frac{\eta_a\eta_w 2\mu\|\beta_*\|}{\sqrt{\delta^2 + 4\eta_a\eta_w\mu^2} - \delta}\left(1 - \phi^2\right), \qquad \phi(0) = \frac{w_0^\intercal \beta_*}{\|w_0\|\|\beta_*\|}. \tag{13}$$

We leave the derivation to the reader, but emphasize that a key simplification used is to express the sum $\eta_w a^2 + \eta_a\|w\|^2$ in terms of δ,

$$\eta_w a^2 + \eta_a\|w\|^2 = \sqrt{\delta^2 + 4\eta_a\eta_w\mu^2}. \tag{14}$$

Additionally, notice that η_a and η_w only appear in the dynamics for μ and ϕ as the product $\eta_a\eta_w$ or in the expression for δ. If we were to define $\mu' = \sqrt{\eta_a\eta_w}\mu$ and $\beta'_* = \sqrt{\eta_a\eta_w}\beta_*$, then it is not hard to show that the product $\eta_a\eta_w$ is absorbed into the dynamics. Thus, without loss of generality we can assume the product $\eta_a\eta_w = 1$, resulting in the following coupled system of nonlinear ODEs,

$$\dot{\mu} = \sqrt{\delta^2 + 4\mu^2}\left(\phi\|\beta_*\| - \mu\right), \qquad \mu(0) = a_0\|w_0\| \tag{15}$$

$$\dot{\phi} = \frac{2\mu\|\beta_*\|}{\sqrt{\delta^2 + 4\mu^2} - \delta}\left(1 - \phi^2\right), \qquad \phi(0) = \frac{w_0^\intercal \beta_*}{\|w_0\|\|\beta_*\|} \tag{16}$$

We will now show how to solve this system of equations for μ and ϕ. We will solve this system when $\delta = 0$, $\delta > 0$, and $\delta < 0$ separately. We will then in Appendix B.2.4 show a general treatment on how to obtain the individual coordinates of a and w from the dynamics of μ and ϕ.

B.2.2. Balanced $\delta = 0$

When $\delta = 0$, the dynamics for μ, ϕ become,

$$\dot{\mu} = \operatorname{sgn}(\mu)2\mu(\phi\|\beta_*\| - \mu), \qquad \dot{\phi} = \operatorname{sgn}(\mu)\|\beta_*\|(1 - \phi^2). \tag{17}$$

First, we show that the sign of μ cannot change through training and $\operatorname{sgn}(\mu) = \operatorname{sgn}(a)$. Because $\delta = 0$, the dynamics of a and w are constrained to a double cone with a singularity at the origin $(a = 0, w = 0)$. This point is a saddle point of the dynamics, so the trajectory cannot pass through this point to move from one cone to the other. In other words, the cone where the dynamics are

initialized on is the cone they remain on. Without loss of generality, we assume $a_0 > 0$, and solve the dynamics. The dynamics of μ is a Bernoulli differential equation driven by a time-dependent signal $\phi\|\beta_*\|$. The dynamics of ϕ is decoupled from μ and is in the form of a Riccati equation evolving from an initial value ϕ_0 to 1, as we have assumed an initialization with positive a_0. This ODE is separable with the solution,

$$\phi(t) = \tanh\left(c_\phi + \|\beta_*\| t\right), \tag{18}$$

where $c_\phi = \tanh^{-1}(\phi_0)$. Plugging this solution into the dynamics for μ gives a Bernoulli differential equation,

$$\dot{\mu} = 2\|\beta_*\| \tanh\left(c_\phi + \|\beta_*\| t\right) \mu - 2\mu^2, \tag{19}$$

with the solution,

$$\mu(t) = \frac{2\cosh^2\left(c_\phi + \|\beta_*\| t\right)}{2\left(c_\phi + \|\beta_*\| t\right) + \sinh\left(2(c_\phi + \|\beta_*\| t)\right) + c_\mu}, \tag{20}$$

where $c_\mu = 2\mu_0^{-1}\cosh^2(c_\phi) - (2c_\phi + \sinh(2c_\phi))$. Note, if $\phi_0 = -1$, then $\dot{\phi} = 0$, and the dynamics of μ will be driven to 0, which is a saddle point.

B.2.3. UNBALANCED $\delta \neq 0$

In this setting, the dynamics live on a hyperboloid. We assume $a_0 > 0$ without loss of generality. The dynamics of μ and ϕ do not decouple. Assuming $a(t) \neq 0, \forall t \geq 0$, we consider $\nu = \frac{w^\intercal \beta_*}{a}$, which leads to the decoupled equation:

$$\dot{\nu} = \|\beta_*\|^2 - \delta\nu - \nu^2, \qquad \nu(0) = \tfrac{w_0^\intercal \beta_*}{a_0} \tag{21}$$

Assuming that $\|\beta_*\| \neq 0$, the solution is given by:

$$\nu(t) = \frac{2R\nu_0 \cosh\left(Rt\right) + \left(2\|\beta_*\|^2 - \delta\nu_0\right)\sinh\left(Rt\right)}{2R\cosh\left(Rt\right) + \left(2\nu_0 + \delta\right)\sinh\left(Rt\right)} \tag{22}$$

where $R = \frac{1}{2}\sqrt{\delta^2 + 4\|\beta_*\|^2}$. With $\nu(t)$, $a(t)$ can be analytically solved from the Bernoulli equation,

$$\dot{a} = a(\nu(t) + \delta - a^2), \qquad a(0) = a_0 \tag{23}$$

We omit the solution due to its complexity, but provide a notebook used to generate our figures encoding the solution. Note that in the upstream case where $\delta > 0$, $a(t) > 0$ is guaranteed by the fact that $a^2 = \delta + \|w\|^2 > 0$. Therefore, ν is always well defined. However, in the downstream case where $\delta < 0$, a can cross 0, which leads to the problem defining ν. The following lemma shows that a can only cross 0 at most once.

Lemma 3 *Assuming the existence and uniqueness of the solution and that $a(0) \neq 0$ or $w(0)^\intercal \beta_* \neq 0$, $a(t)w(t)^\intercal \beta_* = 0$ has at most one solution for $t \geq 0$.*

Proof Denote $w_\parallel(t) = w(t)^\intercal \beta_*$ The dynamics of $a(t)$ and $w_\parallel(t)$ is given by,

$$\dot{w}_\parallel = a\|\beta_*\|^2 - a^2 w_\parallel \tag{24}$$

$$\dot{a} = w_\parallel - a(a^2 - \delta) \tag{25}$$

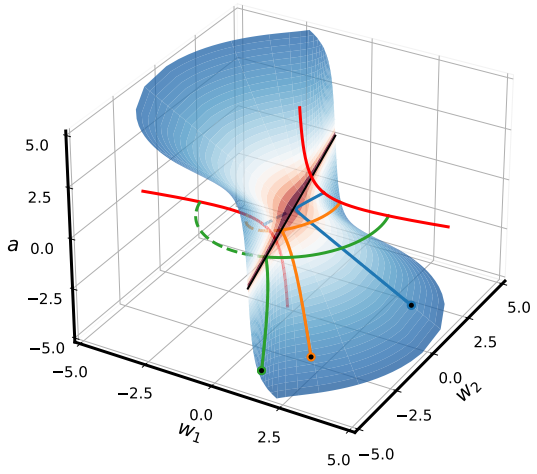
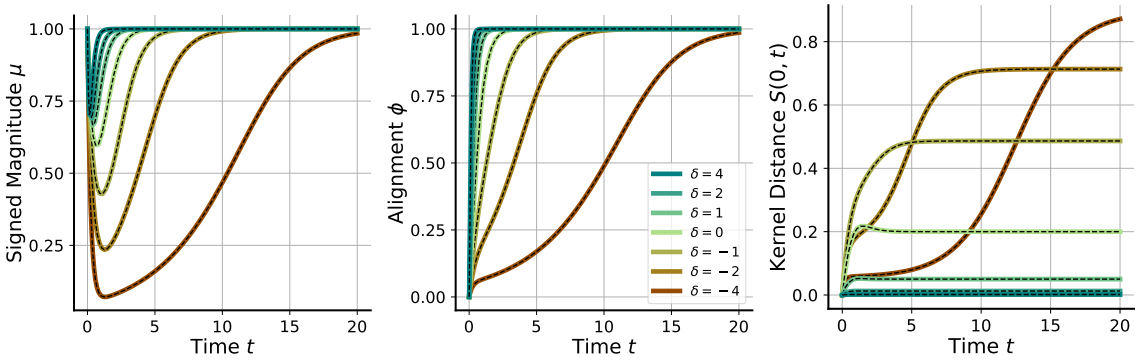


Figure 4: **Two basins of attraction.** For this model, parameter space is partitioned into two basins of attraction, one for the positive and negative branch of the minima manifold. The surface separating the basins of attraction is determined by the equation $w_0^\intercal \beta_* + \frac{a_0}{2}\left(\delta + \sqrt{\delta^2 + 4\|\beta_*\|^2}\right) = 0$. For a given δ, this equation describes a hyperplane through the origin. However, a given δ can only be achieved on the surface of some hyperboloid. Thus, the separating surface is the union of the intersections of a hyperplane and a hyperboloid, both parameterized by δ. This intersection is empty if $\delta > 0$. Initializations exactly on the separating surface will travel along the surface to a saddle point where $w^\intercal \beta_* = a = 0$.

Consider $S_+ = \{(a, w_\|)|a > 0, w_\| > 0\}$. At the boundary $\{(a, w_\|)|a = 0, w_\| \geq 0\}$, $\dot{a} \geq 0$; at the boundary $\{(a, w_\|)|a \geq 0, w_\| = 0\}$, $\dot{w}_\| \geq 0$. Therefore, S_+ is a positively invariant set. Similarly, $S_- = \{(a, w_\|)|a < 0, w_\| < 0\}$ is a positively invariant set. On the boundary $\partial S_+ \cup \partial S_- = \{(a, w_\|)|aw_\| = 0\}$, the flow is contained in the boundary only at the origin $a = 0, w_\| = 0$, which is a saddle point of the system. By assumption, the system does not start at the origin, and thus the origin in not reachable for all $t \geq 0$ by uniqueness. As a result, the trajectory $(a(t), w_\|(t))$ will at most intersect the boundary $\partial S_+ \cup \partial S_-$ once. ■

From Lemma 3, we know that a sign change can happen for at most one value of t. In the case that a does not change sign, ν is well-behaved along its entire trajectory, and our derivation still holds, leading to solutions for w and a. Conversely, suppose a changes sign at some $t^* > 0$. Assuming that the existence and uniqueness of the solutions in Eq. (10), we proceed by simply following the same approach and then verifying that the solution we obtain for a and w in fact solves Eq. (10).

Obtaining the basins of attraction. From Lemma 3 we know that a can cross 0 at most once in its trajectory. As a result, we can find the basin of attraction by deriving the conditions under which a changes sign. From Eq. (22) we can immediately see that a will change sign when the denominator vanishes. This can happen if $\sqrt{\delta^2 + 4\|\beta_*\|^2} < -2\nu_0 - \delta$. For $\delta < 0$, this is satisfied if $\nu_0 < \frac{1}{2}\left(-\delta - \sqrt{\delta^2 + 4s^2}\right)$, which gives the hyperplane $w_0^\intercal \beta_* + \frac{a_0}{2}\left(\delta + \sqrt{\delta^2 + 4\|\beta_*\|^2}\right) = 0$ that



(a) $\mu(t)$ (b) $\phi(t)$ (c) $S(0,t)$

Figure 5: **Exact solutions for the single hidden neuron model.** Our theoretical predictions (black dashed lines) agree with gradient flow simulations (solid lines, color-coded based on δ values), shown here for three key metrics: μ (left), ϕ (middle), and $S(0,t)$ (right). Each metric starts at the same value for all δ, but varying δ has a pronounced effect on the metric's dynamics. For upstream initializations ($\delta \gg 0$), μ changes only slightly, ϕ exponentially aligns, and S remains near zero, indicative of the lazy regime. For balanced initializations ($\delta = 0$), both μ and ϕ change significantly and S quickly moves away from zero, indicative of the rich regime. For downstream initializations ($\delta \ll 0$), μ quickly drops to zero, then μ and ϕ slowly climb back to one. Similarly, S remains small before a sudden transition towards one, indicative of a delayed rich regime. For further details on the solutions see Appendix B.2.

separates between initializations for which a changes sign and initializations for which it does not (Fig. 4). Consequently, letting S^+ be the set of initializations attracted to the minimum manifold with $a > 0$, we have that:

$$S^+ = \left\{ (w_0, a_0) \;\middle|\; \begin{array}{ll} a_0 > 0 & \text{if } \delta \geq 0 \\ w_0^\intercal \beta_* > -\frac{a_0}{2}\left(\delta + \sqrt{\delta^2 + 4\|\beta_*\|^2}\right) & \text{if } \delta < 0 \end{array} \right\} \tag{26}$$

where the bottom inequality means that β_0 is sufficiently aligned to β_* in the case of $a_0 \geq 0$ or sufficiently misaligned in the case of $a_0 \leq 0$. We can similarly define the analogous S^-. An initialization on the separating hyperplane will converge to a saddle point where $w^\intercal \beta_* = a = 0$.

B.2.4. RECOVERING PARAMETERS (a, w) FROM (μ, ϕ)

We can recover a and $\|w\|$ from μ. Using Eq. (14) discussed previously, we can show

$$a = \text{sgn}(\mu)\sqrt{\frac{\sqrt{\delta^2 + 4\mu^2} + \delta}{2}}, \qquad \|w\| = \sqrt{\frac{\sqrt{\delta^2 + 4\mu^2} - \delta}{2}}. \tag{27}$$

We now consider how to obtain the vector w from ϕ. The key observation, as discussed in Section 2, is that w only moves in the span of w_0 and β_*. This means we can express $w(t)$ as

$$w(t) = c_1(t)\left(\frac{\beta_*}{\|\beta_*\|}\right) + c_2(t)\left(\frac{\left(\mathbf{I}_d - \frac{\beta_*\beta_*^\intercal}{\|\beta_*\|^2}\right)w_0}{\sqrt{\|w_0\|^2 - \left(\frac{\beta_*^\intercal w_0}{\|\beta_*\|}\right)^2}}\right) \tag{28}$$

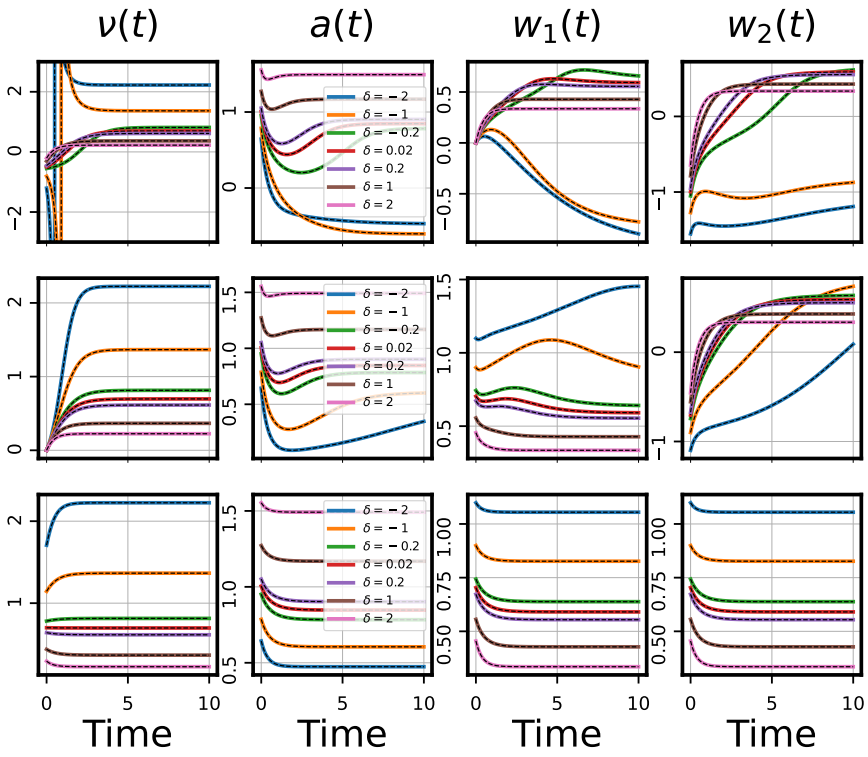


Figure 6: **Exact temporal dynamics of relevant variables in single-hidden neuron model.** Our theory recovers the time evolution under gradient flow of the quantities considered in this section, specifically ν, φ, and ζ, as well as the resulting dynamics of the model parameters $\{a, w_1, w_2\}$. The true $\beta*$ is a unit vector pointing in $\pi/4$ direction; $\beta(0)$ is a unit vector pointing towards $3\pi/2$, $-\pi/4$, and $\pi/4$ directions, respectively, for each of the three rows. δ then defines how $a(0)$ and $\|w(0)\|$ are chosen for a particular $\beta(0)$ where by convention we choose $a(0) > 0$.

where $c_1(t)$ is the coefficient in the direction of β_* and $c_2(t)$ is the coefficient in the direction orthogonal to β_* on the two-dimensional plane defined by w_0. From the definition of ϕ we can easily obtain the coefficients $c_1 = \|w\|\phi$ and $c_2 = \sqrt{\|w\|^2 - c_1^2}$. We always choose the positive square root for c_2, as $c_2(t) \geq 0$ by definition for all t. This is because c_2 starts non-negative and cannot switch signs. If c_2 were to become zero, then $\phi = \pm 1$, which is a fixed point of its dynamics.

B.3. Function space dynamics of β

The network's function is determined by the product $\beta = aw$ and governed by the ODE,

$$\dot{\beta} = a\dot{w} + \dot{a}w = -\underbrace{\left(\eta_w a^2 I_d + \eta_a w w^\intercal\right)}_{M}\underbrace{\left(X^\intercal X\beta - X^\intercal y\right)}_{X^\intercal \rho}. \tag{29}$$

Notice, that the vector $X^\intercal\rho$ driving the dynamics of β is the gradient of the loss with respect to β, $X^\intercal\rho = \nabla_\beta \mathcal{L}$. Thus, these dynamics can be interpreted as preconditioned gradient flow on the loss in β space where the preconditioning matrix M depends on time through its dependence on a^2 and $ww^\intercal$. The matrix M also characterizes the NTK matrix, $K = XMX^\intercal$. As discussed in Section 2,

our goal is to understand the evolution of M along a trajectory $\{\beta(t) \in \mathbb{R}^d \ : \ t \geq 0\}$ solving Eq. (29).

First, notice that by expanding $\|\beta\|^2 = a^2\|w\|^2$ in terms of the conservation law, we can show

$$a^2 = \frac{\sqrt{\delta^2 + 4\eta_a\eta_w\|\beta\|^2} + \delta}{2\eta_w}, \tag{30}$$

which is the unique positive solution of the quadratic expression $\eta_w a^4 - \delta a^2 - \eta_a\|\beta\|^2 = 0$. When $a^2 > 0$ we can use this solution and the outer product $\beta\beta^\intercal = a^2 ww^\intercal$ to solve for $ww^\intercal$ in terms of β,

$$ww^\intercal = \frac{\sqrt{\delta^2 + 4\eta_a\eta_w\|\beta\|^2} - \delta}{2\eta_a}\frac{\beta\beta^\intercal}{\|\beta\|^2}. \tag{31}$$

Plugging these expressions into M gives

$$M = \frac{\sqrt{\delta^2 + 4\eta_a\eta_w\|\beta\|^2} + \delta}{2} I_d + \frac{\sqrt{\delta^2 + 4\eta_a\eta_w\|\beta\|^2} - \delta}{2}\frac{\beta\beta^\intercal}{\|\beta\|^2}. \tag{32}$$

Thus, given any initialization a_0, w_0 such that $a(t)^2 > 0$ for all $t \geq 0$, we can express the dynamics of β entirely in terms of β. This is true for all initialization with $\delta \geq 0$, except if initialized on the saddle point at the origin. It is also true for all initializations with $\delta < 0$ where the sign of a does not switch signs. In the next section we will show how to interpret these trajectories as time-warped mirror flows for a potential that depends on δ. As a means of keeping the analysis entirely in β space, we will make the slightly more restrictive assumption to only study trajectories given any initialization β_0 such that $\|\beta(t)\| > 0$ for all $t \geq 0$.

Notice, that η_a and η_w only appear in the dynamics for β as the product $\eta_a\eta_w$ or in the expression for δ. By defining $\beta' = \sqrt{\eta_a\eta_w}\beta$ and $y' = \sqrt{\eta_a\eta_w}y$ and studying the dynamics of β', we can absorb $\eta_a\eta_w$ into the β terms in M and the additional factor $\sqrt{\eta_a\eta_w}$ into the β and y terms in ρ. This transformation of β and y merely rescales β space without changing the loss landscape or location of critical points. As a result, from here on we will, without loss of generality, study the dynamics of β assuming $\eta_a\eta_w = 1$.

B.4. Proof of Theorem 1

Until now, we have primarily considered that $X^\intercal X$ is either whitened or full rank, ensuring the existence of a unique least squares solution β_*. In this setting, δ influences the trajectory the model takes from initialization to convergence, but all models eventually converge to the same point, as shown in Fig. 2. Now we consider the over-parameterized setting where we have more features d than observations n such that $X^\intercal X$ is low-rank and there exists infinitely many interpolating solutions in function space. By studying the structure of M we can characterize or even predict how δ determines which interpolating solution the dynamics converge to among all possible interpolating solutions. To do this we will extend a time-warped mirror flow analysis strategy pioneered by Azulay et al. [7].

B.4.1. OVERVIEW OF TIME-WARPED MIRROR FLOW ANALYSIS

Here we recap the standard analysis for determining the implicit bias of a linear network through mirror flow. As first introduced in Gunasekar et al. [24], if the learning dynamics of the predictor β

can be expressed as a *mirror flow* for some strictly convex potential $\Phi_\alpha(\beta)$,

$$\dot{\beta} = -\left(\nabla^2 \Phi_\alpha(\beta)\right)^{-1} X^\intercal \rho, \tag{33}$$

where $\rho = (X\beta - y)$ is the residual, then the limiting solution of the dynamics is determined by the constrained optimization problem,

$$\beta(\infty) = \underset{\beta \in \mathbb{R}^d}{\arg\min}\, D_{\Phi_\alpha}(\beta, \beta(0)) \quad \text{s.t.} \quad X\beta = y, \tag{34}$$

where $D_{\Phi_\alpha}(p, q) = \Phi_\alpha(p) - \Phi_\alpha(q) - \langle \nabla \Phi_\alpha(q), p - q \rangle$ is the Bregman divergence defined with Φ_α. To understand the relationship between mirror flow Eq. (33) and the optimization problem Eq. (34), we consider an equivalent constrained optimization problem

$$\beta(\infty) = \underset{\beta \in \mathbb{R}^d}{\arg\min}\, Q(\beta) \quad \text{s.t.} \quad X\beta = y, \tag{35}$$

where $Q(\beta) = \Phi_\alpha(\beta) - \nabla \Phi_\alpha(\beta(0))^\intercal \beta$, which is often referred to as the *implicit bias*. $Q(\beta)$ is strictly convex, and thus it is sufficient to show that $\beta(\infty)$ is a first order KKT point of the constrained optimization (35). This is true iff there exists $\nu \in \mathbb{R}^n$ such that $\nabla Q(\beta(\infty)) = X^\intercal \nu$. The goal is to derive ν from the mirror flow Eq. (33). Notice, we can rewrite Eq. (33) as, $(\dot{\nabla \Phi_\alpha(\beta)}) = -X^\intercal \rho$, which integrated over time gives

$$\nabla \Phi_\alpha(\beta(\infty)) - \nabla \Phi_\alpha(\beta(0)) = -X^\intercal \int_0^\infty \rho(t) dt. \tag{36}$$

The LHS is $\nabla Q(\beta(\infty))$. Thus, by defining $\nu = \int_0^\infty \rho(t) dt$, which assumes the residual decays fast enough such that this is well defined, then we have shown the desired KKT condition. Crucial to this analysis is that there exists a solution to the second-order differential equation

$$\nabla^2 \Phi_\alpha(\beta) = \left(\nabla_\theta \beta \nabla_\theta \beta^\intercal\right)^{-1}, \tag{37}$$

which even for extremely simple Jacobian maps may not be true [26]. Azulay et al. [7] showed that if there exists a smooth scalar function $g(\beta) : \mathbb{R}^d \to \mathbb{R}$ such that the ODE,

$$\nabla^2 \Phi_\alpha(\beta) = g(\beta) \left(\nabla_\theta \beta \nabla_\theta \beta^\intercal\right)^{-1}, \tag{38}$$

has a solution, then the previous interpretation holds for $\Phi_\alpha(\beta)$ with $\nu = \int_0^\infty g(\beta(t'))\rho(t') dt$. As before, it is crucial that this integral exists and is finite. Azulay et al. [7] further explained that this scalar function $g(\beta)$ can be considered as warping time $\tau(t) = \int_0^t g(\beta(t')) dt'$ on the trajectory taken in predictor space $\beta(\tau(t))$. So long as this warped time doesn't "stall out", that is we require that $\tau(\infty) = \infty$, then this will not change the interpolating solution.

B.4.2. APPLYING TIME-WARPED MIRROR FLOW ANALYSIS

Here show how to apply the time-warped mirror flow analysis to the dynamics of β derived in Appendix B.3 where $\nabla_\theta \beta \nabla_\theta \beta^\intercal = M$. We will only consider initializations β_0 such that $\|\beta(t)\| > 0$ for all $t \geq 0$, such that M can be expressed as

$$M = \frac{\sqrt{\delta^2 + 4\|\beta\|^2} + \delta}{2} I_d + \frac{\sqrt{\delta^2 + 4\|\beta\|^2} - \delta}{2} \frac{\beta\beta^\intercal}{\|\beta\|^2}. \tag{39}$$

Computing M^{-1}. Whenever $\|\beta\| > 0$, then M is a positive definite matrix with a unique inverse that can be derived using the Sherman–Morrison formula, $(A+uv^\intercal)^{-1} = A^{-1} - \frac{A^{-1}uv^\intercal A^{-1}}{1+u^\intercal A^{-1}v}$. Here we can define A, u, and v as

$$A = \left(\frac{\sqrt{\delta^2 + 4\|\beta\|^2} + \delta}{2}\right) I_d,\ u = \left(\frac{\sqrt{\delta^2 + 4\|\beta\|^2} - \delta}{2\|\beta\|^2}\right)\beta,\ v = \beta \tag{40}$$

First notice the following simplification, $u^\intercal A^{-1}v = \frac{\sqrt{\delta^2+4\|\beta\|^2}-\delta}{\sqrt{\delta^2+4\|\beta\|^2}+\delta}$. After some algebra, M^{-1} is

$$M^{-1} = \left(\frac{2}{\sqrt{\delta^2 + 4\|\beta\|^2} + \delta}\right) I_d - \left(\frac{\frac{\sqrt{\delta^2+4\|\beta\|^2}-\delta}{\sqrt{\delta^2+4\|\beta\|^2}+\delta}}{\|\beta\|^2\sqrt{\delta^2 + 4\|\beta\|^2}}\right)\beta\beta^\intercal \tag{41}$$

To make notation simpler we will define the following two scalar functions,

$$f_\delta(x) = \frac{2}{\sqrt{\delta^2 + 4x} + \delta}, \qquad h_\delta(x) = \frac{\sqrt{\delta^2 + 4x} - \delta}{x\sqrt{\delta^2 + 4x}\left(\sqrt{\delta^2 + 4x} + \delta\right)}, \tag{42}$$

such that we can express $M^{-1} = f_\delta\left(\|\beta\|^2\right) I_d - h_\delta\left(\|\beta\|^2\right)\beta\beta^\intercal$.

Proving M^{-1} is not a Hessian map. If M^{-1} is the Hessian of some potential, then we can show that the dynamics of β are a mirror flow. However, from our expression for M^{-1} we can actually prove that it is *not* a Hessian map. As discussed in Gunasekar et al. [26], a symmetric matrix $H(\beta)$ is the Hessian of some potential $\Phi(\beta)$ if and only if it satisfies the condition,

$$\forall \beta \in \mathbb{R}^m, \quad \forall i, j, k \in [m] \quad \frac{\partial H_{ij}(\beta)}{\partial \beta_k} = \frac{\partial H_{ik}(\beta)}{\partial \beta_j}. \tag{43}$$

We will use this property to show M^{-1} is not a Hessian map. First, notice this condition is trivially true when $i = j = k$. Second, notice that for all $i \neq j \neq k$,

$$\frac{\partial M^{-1}_{ij}}{\partial \beta_k} = \frac{\partial M^{-1}_{ik}}{\partial \beta_j} = -2\nabla h_\delta\left(\|\beta\|^2\right)\beta_i\beta_j\beta_k \tag{44}$$

Thus, M^{-1} is a Hessian map if and only if for all $i \neq j$, $\frac{\partial M^{-1}_{ii}}{\partial \beta_j} = \frac{\partial M^{-1}_{ij}}{\partial \beta_i}$. Using our expression for M^{-1}, the LHS is

$$\frac{\partial M^{-1}_{ii}}{\partial \beta_j} = 2\nabla f_\delta\left(\|\beta\|^2\right)\beta_j - 2\nabla h_\delta\left(\|\beta\|^2\right)\beta_j\beta_i^2 \tag{45}$$

while the RHS is

$$\frac{\partial M^{-1}_{ij}}{\partial \beta_i} = -h_\delta\left(\|\beta\|^2\right)\beta_j - 2\nabla h_\delta\left(\|\beta\|^2\right)\beta_j\beta_i^2 \tag{46}$$

Thus, M^{-1} is a Hessian map if and only if $2\nabla f_\delta(x) + h_\delta(x) = 0$. Plugging in our definitions of $f_\delta(x)$ and $h_\delta(x)$ we find

$$2\nabla f_\delta(x) + h_\delta(x) = \frac{-4}{\sqrt{\delta^2 + 4x}(\sqrt{\delta^2 + 4x} + \delta)^2}, \tag{47}$$

which does not equal zero and thus M^{-1} is not a Hessian map.

Finding a scalar function $g_\delta(x)$ such that $g_\delta(\|\beta\|^2)M^{-1}$ is a Hessian map. While we have shown that M^{-1} is not a Hessian map, it is very close to a Hessian map. Here we will show that there exists a scalar function $g_\delta(x)$ such that $g_\delta\left(\|\beta\|^2\right)M^{-1}$ is a Hessian map. For any $g_\delta(x)$ can define $g_\delta\left(\|\beta\|^2\right)M^{-1}$ in terms of two new functions $\tilde{f}_\delta(x)$ and $\tilde{h}_\delta(x)$ evaluated at $x = \|\beta\|^2$,

$$g_\delta\left(\|\beta\|^2\right)M^{-1} = \underbrace{g_\delta\left(\|\beta\|^2\right)f_\delta\left(\|\beta\|^2\right)}_{\tilde{f}_\delta(\|\beta\|^2)} I_d - \underbrace{g_\delta\left(\|\beta\|^2\right)h_\delta\left(\|\beta\|^2\right)}_{\tilde{h}_\delta(\|\beta\|^2)}\beta\beta^\intercal. \tag{48}$$

Thus, as derived in the previous section, we get the analogous condition on $\tilde{f}_\delta(x)$ and $\tilde{h}_\delta(x)$ for $g_\delta\left(\|\beta\|^2\right)M^{-1}$ to be a Hessian map,

$$2\underbrace{\left(\nabla g_\delta(x)f_\delta(x) + g(x)\nabla f_\delta(x)\right)}_{\nabla\tilde{f}_\delta(x)} + \underbrace{g_\delta(x)h_\delta(x)}_{\tilde{h}_\delta(x)} = 0 \tag{49}$$

Rearranging terms we find that $g_\delta(x)$ must solve the ODE

$$\nabla g_\delta(x) = -\left(2f_\delta(x)\right)^{-1}\left(2\nabla f_\delta(x) + h_\delta(x)\right)g_\delta(x). \tag{50}$$

Using our previous expressions (Eq. (42) and Eq. (47)) we find

$$-\left(2f_\delta(x)\right)^{-1}\left(2\nabla f_\delta(x) + h_\delta(x)\right) = \frac{1}{\sqrt{\delta^2+4x}(\sqrt{\delta^2+4x}+\delta)}, \tag{51}$$

which implies $g_\delta(x)$ solves the differential equation, $\nabla g_\delta(x) = \frac{g_\delta(x)}{\sqrt{\delta^2+4x}(\sqrt{\delta^2+4x}+\delta)}$. The solution is $g_\delta(x) = c\sqrt{\sqrt{\delta^2+4x}+\delta}$, where $c \in \mathbb{R}$ is a constant. Let $c = 1$. Plugging in our expressions for $g_\delta\left(\|\beta\|^2\right)$, $f_\delta\left(\|\beta\|^2\right)$, $h_\delta\left(\|\beta\|^2\right)$, we get that

$$g_\delta\left(\|\beta\|^2\right)M^{-1} = \left(\frac{2}{\sqrt{\sqrt{\delta^2+4\|\beta\|^2}+\delta}}\right)I_d - \left(\frac{\frac{\sqrt{\delta^2+4\|\beta\|^2}-\delta}{\sqrt{\sqrt{\delta^2+4\|\beta\|^2}+\delta}}}{\|\beta\|^2\sqrt{\delta^2+4\|\beta\|^2}}\right)\beta\beta^\intercal \tag{52}$$

is a Hessian map for some unknown potential $\Phi_\delta(\beta)$.

Solving for the potential $\Phi_\delta(\beta)$. Take the ansatz that there exists some function scalar $q(x)$ such that $\Phi_\delta(\beta) = q_\delta(\|\beta\|) + c_\delta$ where c_δ is a constant such that $\Phi_\delta(\beta) > 0$ for all $\beta \neq 0$ and $\Phi_\delta(0) = 0$. The Hessian of this ansatz takes the form,

$$\nabla^2\Phi_\delta(\beta) = \left(\frac{\nabla q(\|\beta\|)}{\|\beta\|}\right)I_d - \left(\frac{\nabla q(\|\beta\|)}{\|\beta\|^3} - \frac{\nabla^2 q(\|\beta\|)}{\|\beta\|^2}\right)\beta\beta^\intercal. \tag{53}$$

Equating terms from our expression for $g_\delta\left(\|\beta\|^2\right)M^{-1}$ (equation 52) we get the expression for $\nabla q(\|\beta\|)$

$$\nabla q(\|\beta\|) = \frac{2\|\beta\|}{\sqrt{\sqrt{\delta^2+4\|\beta\|^2}+\delta}}, \tag{54}$$

which plugged into the second term gives the expression for $\nabla^2 q(\|\beta\|)$,

$$\nabla^2 q(\|\beta\|) = \frac{2}{\sqrt{\sqrt{\delta^2+4\|\beta\|^2}+\delta}} - \left(\frac{\frac{\sqrt{\delta^2+4\|\beta\|^2}-\delta}{\sqrt{\sqrt{\delta^2+4\|\beta\|^2}+\delta}}}{\sqrt{\delta^2+4\|\beta\|^2}}\right) = \frac{\sqrt{\sqrt{\delta^2+4\|\beta\|^2}+\delta}}{\sqrt{\delta^2+4\|\beta\|^2}}. \tag{55}$$

We now look for a function $q(x)$ such that both these conditions (Eq. (54) and Eq. (55)) are true. Consider the following function and its derivatives,

$$q(x) = \frac{1}{3}\left(\sqrt{\delta^2+4x^2}-2\delta\right)\sqrt{\sqrt{\delta^2+4x^2}+\delta} \tag{56}$$

$$\nabla q(x) = \frac{2x}{\sqrt{\sqrt{\delta^2+4x^2}+\delta}} \tag{57}$$

$$\nabla^2 q(x) = \frac{\sqrt{\sqrt{\delta^2+4x^2}+\delta}}{\sqrt{\delta^2+4x^2}} \tag{58}$$

Letting $x = \|\beta\|$ notice $\nabla q(\|\beta\|)$ and $\nabla^2 q(\|\beta\|)$ satisfies the previous conditions. Furthermore, $\nabla^2 q(x) > 0$ for all δ as long as $x \neq 0$ and thus $q(x)$ is a convex function which achieves its minimum at $x = 0$. Thus, the constant $c_\delta = -q(0)$ is

$$c_\delta = \begin{cases} 0 & \text{if } \delta \leq 0 \\ \frac{\sqrt{2}|\delta|^{\frac{3}{2}}}{3} & \text{if } \delta > 0 \end{cases} = \max\left\{0, \operatorname{sgn}(\delta)\frac{\sqrt{2}|\delta|^{\frac{3}{2}}}{3}\right\}, \tag{59}$$

and the potential $\Phi_\delta(\beta)$ is

$$\Phi_\delta(\beta) = \frac{1}{3}\left(\sqrt{\delta^2+4\|\beta\|^2}-2\delta\right)\sqrt{\sqrt{\delta^2+4\|\beta\|^2}+\delta} + \max\left\{0, \operatorname{sgn}(\delta)\frac{\sqrt{2}|\delta|^{\frac{3}{2}}}{3}\right\}. \tag{60}$$

Finally, putting it all together, we proved Theorem 1.

B.4.3. CONNECTION TO THEOREM 2 IN AZULAY ET AL. [7]

We discuss how Theorem 1 connects to Theorem 2 in Azulay et al. [7], which we rewrite:

Theorem 4 (Theorem 2 from Azulay et al. [7]) *For a depth 2 fully connected network with a single hidden neuron ($h = 1$), any $\delta \geq 0$, and initialization β_0 such that $\|\beta_0\| > 0$, if the gradient flow solution $\beta(\infty)$ satisfies $X\beta(\infty) = y$, then,*

$$\beta(\infty) = \arg\min_{\beta\in\mathbb{R}^d} q_\delta(\|\beta\|) + z^\intercal\beta \quad \text{s.t.} \quad X\beta = y \tag{61}$$

where $q_\delta(x) = \frac{\left(x^2 - \frac{\delta}{2}\left(\frac{\delta}{2}+\sqrt{x^2+\frac{\delta^2}{4}}\right)\right)\sqrt{\sqrt{x^2+\frac{\delta^2}{4}}-\frac{\delta}{2}}}{x}$ *and* $z = -\frac{3}{2}\sqrt{\sqrt{\|\beta_0\|^2+\frac{\delta^2}{4}}-\frac{\delta}{2}}\frac{\beta_0}{\|\beta_0\|}$.

The most striking difference is in the expressions for the inductive bias. Azulay et al. [7] take an alternative route towards deriving the inductive bias by inverting M in terms of the original parameters a and w and then simplifying M^{-1} in terms of β, which results in quite a different expression for their inductive bias. However, they are actually functionally equivalent. It requires a bit of algebra, but one can show that

$$\Phi_\delta(\beta) = \frac{2\sqrt{2}}{3} q_\delta(\|\beta\|) + c_\delta. \tag{62}$$

Another important distinction between our two theorems lies in the assumptions we make. Azulay et al. [7] consider only initializations such that $\delta \geq 0$ and $\|\beta_0\| > 0$. We make a less restrictive assumption by considering initializations β_0 such that $\|\beta(t)\| > 0$ for all $t \geq 0$, which allows for both positive and negative δ. Except for a measure zero set of initializations, all initializations considered by Azulay et al. [7] also satisfy our assumptions. In both cases, our assumptions ensure that M is invertible for the entire trajectory from initialization to interpolating solution. However, it is worth considering whether the theorems would hold even when there exists a point on the trajectory where M is low-rank. As discussed in Appendix B.3, this can only happen for an initialization with $\delta < 0$ and where the sign of a changes. Only at the point where $a(t) = 0$ does M become low-rank. A similar challenge arose in this setting when deriving the exact solutions presented in Appendix B.2.3. We were able to circumvent the issue in part by using a lemma showing that this sign change could only happen at most once given any initialization. This lemma was based on the setting with whitened input, but a similar statement likely holds for the general setting. If this were the case, we could define M at this unique point on the trajectory in terms of the limit of M as it approached this point. This could potentially allow us to extend the time-warped mirror flow analysis to all initializations such that $\|\beta_0\| > 0$.

B.4.4. EXACT SOLUTION WHEN INTERPOLATING MANIFOLD IS ONE-DIMENSIONAL

When the null space of $X^\intercal X$ is one-dimensional, the constrained optimization problems in Theorem 1 and Theorem 4 have an exact analytic solution. In this case we can parameterize all interpolating solutions β with a single scalar $\alpha \in \mathbb{R}$ such that $\beta = \beta_* + \alpha v$ where $X^\intercal X v = 0$ and $\|v\| = 1$. Using this description of β, we can then differentiate the inductive bias with respect to α, set to zero, and solve for α. We will use the following expressions,

$$\nabla_x q(x) = \frac{3}{2}\text{sign}(x)\sqrt{\sqrt{x^2 + \frac{\delta^2}{4}} - \frac{\delta}{2}}, \quad \nabla_\alpha \|\beta\| = \frac{\alpha}{\|\beta\|}, \quad \nabla_\alpha z^\intercal \beta = z^\intercal v. \tag{63}$$

We will also use the expression, $\|\beta\|^2 = \|\beta_*\|^2 + \alpha^2$. Pulling these expressions together we get the following equation for α,

$$\sqrt{\sqrt{\|\beta_*\|^2 + \alpha^2 + \frac{\delta^2}{4}} - \frac{\delta}{2}} \frac{\alpha}{\sqrt{\|\beta_*\|^2 + \alpha^2}} = -\frac{2z^\intercal v}{3}. \tag{64}$$

If we let $k = -\frac{2z^\intercal v}{3}$, the solution for α is

$$\alpha = k\sqrt{\frac{k^2 + \delta}{2} + \sqrt{\left(\frac{k^2 + \delta}{2}\right)^2 + \|\beta_*\|^2}}. \tag{65}$$

This solution always works for the initializations we considered in Theorem 1. Interestingly, it appears that $\beta = \beta_* - \alpha v$ also works for initializations not previously considered. This includes trajectories that pass through the origin, resulting in a change in the sign of a.

Appendix C. Wide and Deep Linear Networks

In the previous section we demonstrated how the balancedness δ influences the regime of learning in a single-neuron linear network by studying the dynamics in parameter space, function space, and the implicit bias. Throughout our analysis, we identified three learning regimes – lazy, rich, and delayed rich – that correspond to different values of δ. The driving cause of this distinction is the change in the geometry and topology of the conserved surface. Here we discuss how our analysis techniques can be extended to linear networks with multiple neurons, layers, and outputs. As we move towards more complex networks, the number of conserved quantities will grow, one for each hidden-neuron. As a result, the analysis in this section will get more complex, but overall the main points identified in the single-neuron setting still hold.

C.1. Two layer function space dynamics.

We consider the dynamics of a two-layer linear network with h hidden neurons and c outputs, $f(x;\theta) = A^\intercal W x$, where $W \in \mathbb{R}^{h\times d}$ and $A \in \mathbb{R}^{h\times c}$. We assume that $\min\{d,c\} \leq h \leq \max\{d,c\}$, such that this parameterization can represent all linear maps from $\mathbb{R}^d \to \mathbb{R}^c$. As in the single-neuron setting, the rescaling symmetry in this model between the first and second layer implies the $h \times h$ matrix $\Delta = A_0 A_0^\intercal - W_0 W_0^\intercal$ determined at initialization remains conserved throughout gradient flow [17]. The NTK matrix can be expressed as $K = (\mathbf{I}_c \otimes X)\,(A^\intercal A \oplus W^\intercal W)\,(\mathbf{I}_c \otimes X^\intercal)$, where $\otimes$ and $\oplus$ denote the Kronecker product and sum[1] respectively. We consider the dynamics of $\beta = W^\intercal A \in \mathbb{R}^{d\times c}$ in function space. The network function β is governed by the ODE,

$$\dot{\beta} = \dot{W}^\intercal A + W^\intercal \dot{A}. \tag{66}$$

Respectively W and A follow the temporal dynamics given by the ODE

$$\dot{W}^\intercal = -\eta_W X^\intercal (X\beta - Y) A^\intercal, \tag{67}$$

and

$$\dot{A} = -\eta_A W X^\intercal (X\beta - Y). \tag{68}$$

Replacing equations 68 and 67 in equation 66 we get

$$\dot{\beta} = -\left(\eta_W X^\intercal (X\beta - Y) A^\intercal A + \eta_A W^\intercal W X^\intercal (X\beta - Y)\right). \tag{69}$$

Vectorising using the identity $vec(\mathbf{ABC}) = (\mathbf{C}^\intercal \otimes \mathbf{A}) vec(\mathbf{B})$ equation 69 becomes

$$vec(\dot{\beta}) = -vec\left(\eta_W I_d X^\intercal \underbrace{(X\beta - Y)}_{P} A^\intercal A + \eta_A W^\intercal W X^\intercal \underbrace{(X\beta - Y)}_{P} I_c\right), \tag{70}$$

$$= -(\eta_W A^\intercal A \otimes I_c + \eta_A I_d \otimes W^\intercal W) vec(X^\intercal P), \tag{71}$$

$$= -\underbrace{(\eta_W A^\intercal A \oplus \eta_A W^\intercal W)}_{M} \mathrm{vec}(X^\intercal P). \tag{72}$$

The vectorised form of the network function is given by

$$\mathrm{vec}\left(\dot{\beta}\right) = -\underbrace{(\eta_W A^\intercal A \oplus \eta_A W^\intercal W)}_{M} \mathrm{vec}(X^\intercal X\beta - X^\intercal Y). \tag{73}$$

1. The Kronecker sum is defined for square matrices $A \in \mathbb{R}^{c\times c}$ and $B \in \mathbb{R}^{d\times d}$ as $A \oplus B = A \otimes \mathbf{I}_d + \mathbf{I}_c \otimes B$.

Interpreting $M(\beta)$ in different limit and architectures

As in the single-neuron setting, we find that the dynamics of β can be expressed as gradient flow preconditioned by a matrix M that depends on quadratics of A and W.

Consider a single hidden neuron $i \in [h]$ of the multi-output model defined by the parameters $w_i \in \mathbb{R}^d$ and $a_i \in \mathbb{R}^c$. Let $\beta_i = w_i a_i^\intercal$ be the $\mathbb{R}^{d\times c}$ matrix representing the contribution of this hidden neuron to the input-output map of the network. As in the previous section, we will consider the two gram matrices $\beta_i^\intercal \beta_i \in \mathbb{R}^{c\times c}$ and $\beta_i \beta_i^\intercal \in \mathbb{R}^{d\times d}$,

$$\beta_i^\intercal \beta_i = \|w_i\|^2 a_i a_i^\intercal, \qquad \beta_i \beta_i^\intercal = \|a_i\|^2 w_i w_i^\intercal. \tag{74}$$

Notice that we can express $\|\beta_i\|_F^2$ as

$$\|\beta_i\|_F^2 = \mathrm{Tr}(\beta_i^\intercal \beta_i) = \mathrm{Tr}(\beta_i \beta_i^\intercal) = \|a_i\|^2 \|w_i\|^2 \tag{75}$$

At each hidden neuron we have the conserved quantity[2] $\eta_W \|a_i\|^2 - \eta_A \|w_i\|^2 = \delta_i$ where $\delta_i \in \mathbb{R}$. Using this quantity we can invert the expression for $\|\beta_i\|_F^2$ to get

$$\|a_i\|^2 = \frac{\sqrt{\delta_i^2 + \eta_A \eta_W 4 \|\beta_i\|_F^2} + \delta_i}{2}, \tag{76}$$

$$\|w_i\|^2 = \frac{\sqrt{\delta_i^2 + \eta_A \eta_W 4 \|\beta_i\|_F^2} - \delta_i}{2}. \tag{77}$$

When $\|\beta_i\|_F^2 > 0$, we can use these expressions to solve for the outer products $a_i a_i^\intercal$ and $w_i w_i^\intercal$ entirely in terms of β_i,

$$a_i a_i^\intercal = \frac{\sqrt{\delta_i^2 + \eta_A \eta_W 4 \|\beta_i\|_F^2} + \delta_i}{2} \frac{\beta_i^\intercal \beta_i}{\|\beta_i\|_F^2}, \tag{78}$$

$$w_i w_i^\intercal = \frac{\sqrt{\delta_i^2 + \eta_A \eta_W 4 \|\beta_i\|_F^2} - \delta_i}{2} \frac{\beta_i \beta_i^\intercal}{\|\beta_i\|_F^2}. \tag{79}$$

Without making any assumptions on the initialization (such as the isotropic initialization) we can express the NTK in terms of the β_i and consider the effect the vector of conserved quantities $\delta \in \mathbb{R}^h$ has on the dynamics.

Lemma 5 *Assuming $\|\beta_i\|_F \neq 0$ for all $i \in [h]$ and let $\kappa_i = \sqrt{\delta_i^2 + 4\eta_A \eta_W \|\beta_i\|_F^2}$, then the matrix M can be expressed as the sum $M = \sum_{i=1}^h M_i$ over hidden neurons where M_i is defined as,*

$$M_i = \left(\frac{\kappa_i + \delta_i}{2}\right) \frac{\beta_i^\intercal \beta_i}{\|\beta_i\|_F^2} \oplus \left(\frac{\kappa_i - \delta_i}{2}\right) \frac{\beta_i \beta_i^\intercal}{\|\beta_i\|_F^2}. \tag{80}$$

We show how to express M in terms of the matrices $\beta_i = w_i a_i^\intercal \in \mathbb{R}^{d\times c}$, which represent the contribution to the input-output map of a single hidden neuron $i \in [h]$ of the network with parameters $w_i \in \mathbb{R}^d$, $a_i \in \mathbb{R}^c$, and conserved quantity $\delta_i = \Delta_{ii}$.

2. As long as $c > 1$, then the surface of this $d + c$ hyperboloid is always connected, however its topology will depend on the relationship between d and c.

C.1.1. FUNNEL NETWORKS

We consider *funnel networks*, which narrow from input to output ($d > h \geq c$), and *inverted-funnel networks*, which expand from input to output ($d \leq h < c$).

As $\delta_i \to \infty$, $M \to \sum_{i=1}^{h} |\delta_i| \frac{\beta_i^\intercal \beta_i}{\|\beta_i\|_F^2} \otimes \mathbf{I}_d$.

Lemma 6 *Consider the rank-one matrices $\frac{\beta_i^\intercal \beta_i}{\|\beta_i\|_F^2}$ in the space $\mathbb{R}^{c\times c}$. The rank of M is bounded by*

$$\operatorname{rank}(M) \leq \begin{cases} d \times h & \text{if } h < c, \text{ Low-rank} \\ d \times c & \text{if } h \geq c. \end{cases} \tag{81}$$

Proof In this limit, the rank of M is given by

$$d \times \operatorname{rank}(\sum_{i=1}^{h} \frac{\beta_i^\intercal \beta_i}{\|\beta_i\|_F^2}) \leq d \times \sum_{i=1}^{h} \operatorname{rank}(\frac{\beta_i^\intercal \beta_i}{\|\beta_i\|_F^2}). \tag{82}$$

It follows that

$$\sum_{i=1}^{h} \operatorname{rank}\left(\frac{\beta_i^\intercal \beta_i}{\|\beta_i\|_F^2}\right) \leq \begin{cases} h & \text{if } h < c. \\ c & \text{if } h \geq c. \end{cases} \tag{83}$$

as the rank of the sum $\sum_{i=1}^{h} \frac{\beta_i^\intercal \beta_i}{\|\beta_i\|_F^2}$ is also at most $\operatorname{rank} c$. Therefore,

$$\operatorname{rank}(M) \leq \begin{cases} d \times h & \text{if } h < c. \\ d \times c & \text{if } h \geq c. \end{cases} \tag{84}$$

■

According to Lemma 6

- If $h < c$, $\operatorname{rank}(M)$ is bounded by $d \times h$ and remains below $d \times c$, categorizing it as a low-rank matrix. As a result, the solution $\beta_*^\intercal \beta_*$ might be in the null space of M. The network may enter either the lazy or a lazy followed by rich regime, depending on the relationship between β_0 and β_*. If $\beta_0 \propto \beta_*$ the network will enter the lazy regime.
- Assuming the $\frac{\beta_i^\intercal \beta_i}{\|\beta_i\|_F^2}$ terms are linearly independent, the matrix $\operatorname{rank}(M)$ achieves full rank and spans the solution space. The network can learn the task by only changing their norm while keeping their direction and the NTK matrix fixed. Thus, *funnel networks* defined by ($d > h \geq c$) will transition into the lazy regime in this limit.

A similar assertion applies as $\delta_i \to -\infty$. In this limit, $M \to \mathbf{I}_c \otimes \sum_{i=1}^{h} |\delta_i| \frac{\beta_i \beta_i^\intercal}{\|\beta_i\|_F^2}$ with the $\operatorname{rank}(M)$ being constrained by the relationship between the number of hidden layers h and the input layer dimensions d.

- If $h < d$, the matrix M is low rank and bounded by $c \times h$. These networks may enter either the lazy or a lazy followed by rich regime, depending on the relationship between β_0 and β_*.

- When $h \geq d$, M is full rank if $\frac{\beta_i^\intercal \beta_i}{\|\beta_i\|_F^2}$ are linearly independent. Except for a measure zero set of initializations, *inverted funnel* networks always enter the lazy regime in this limit.

When $\delta_i = 0$, $M = \sqrt{\eta_A \eta_W}|\beta_i| \sum_{i=1}^h \frac{\beta_i^\intercal \beta_i}{\|\beta_i\|_F^2} \oplus \sum_{i=1}^h \frac{\beta_i \beta_i^\intercal}{\|\beta_i\|_F^2}$ all networks transition into the rich regime. Employing a similar rationale as before, assuming that all terms $\frac{\beta_i^\intercal \beta_i}{|\beta_i|_F^2}$ and $\frac{\beta_i \beta_i^\intercal}{|\beta_i|_F^2}$ are respectively linearly separable, then one term of M will be low-rank, while the other will be full-rank, contingent on the relationship between c, d, and h. Consequently, the dynamics of the network balance between low-rank and full-rank elements, leading to changes in both the magnitude and direction of the Neural Tangent Kernel (NTK) during training, a hallmark of the rich regime.

C.1.2. SINGLE-NEURON

$$M = \frac{\kappa + \delta}{2} I_d + \frac{\kappa - \delta}{2} \frac{\beta \beta^\intercal}{\|\beta\|^2} \tag{85}$$

For *funnel network* with a single hidden neuron ($h = c = 1$), we recover equation 85 from equation 80.We extend this analysis to *inverted-funnel network* with a single hidden neuron ($h = d = 1$). Assuming $h = c = 1$, the rank one matrix $\frac{\beta_i^\intercal \beta_i}{\|\beta_i\|_F^2} = 1$. Therefore, equation 80 becomes

$$M = \left(\frac{\kappa + \delta}{2}\right) \oplus \left(\frac{\kappa - \delta}{2}\right) \frac{\beta \beta^\intercal}{\|\beta\|_F^2}. \tag{86}$$

$$= \frac{\kappa + \delta}{2} I_d + \frac{\kappa - \delta}{2} \frac{\beta \beta^\intercal}{\|\beta\|^2}. \tag{87}$$

where $\kappa = \sqrt{\delta^2 + \eta_A \eta_W 4\|\beta\|^2}$. We recover the *funnel network* equation 85 for a single-neuron. In the main text, we analyze how the expression for $M(\beta)$ simplifies when δ approaches $-\infty$, 0, and ∞. This analysis helps us develop a deeper understanding of $M_\delta(\beta)$.

We now turn to single neuron *inverted funnel network* where $h = d = 1$, the rank one matrices $\frac{\beta_i \beta_i^\intercal}{\|\beta_i\|_F^2} = 1$. Therefore, equation 80 becomes

$$M = \left(\frac{\kappa + \delta}{2}\right) \frac{\beta^\intercal \beta}{\|\beta\|_F^2} \oplus \left(\frac{\kappa - \delta}{2}\right) \tag{88}$$

$$= \frac{\kappa + \delta}{2} \frac{\beta^\intercal \beta}{\|\beta\|_F^2} + \frac{\kappa - \delta}{2} I_c. \tag{89}$$

From our expression for $M(\beta)$ we will consider how it simplifies when $\delta \to -\infty, 0, \infty$.

$$M \to \begin{cases} \delta I_c & \delta \to -\infty. \\ \sqrt{\eta_A \eta_W}\|\beta\| \left(\frac{\beta^\intercal \beta}{\|\beta\|^2} + I_c\right) & \delta = 0. \\ |\delta| \frac{\beta^\intercal \beta}{\|\beta\|^2} & \delta \to \infty. \end{cases} \tag{90}$$

As anticipated, the single-neuron *inverted funnel network* also transitions into the ***Rich*** regime when $\delta = 0$. Under this condition, $M = \sqrt{\eta_A \eta_W}\|\beta\|(\frac{\beta^\intercal \beta}{\|\beta\|^2} + I_c)$, where the initial term denotes a projection matrix. Here the dynamics balance between following the lazy trajectory and attempting

to fit the task by only changing in norm. As a result the NTK changes in both magnitude and direction through training, confirming the dynamics are rich.

Additionally, the single-neuron *inverted funnel network* follows analogous regime transitions to the single-neuron *funnel network*, albeit in opposite directions. As $\delta \to -\infty$, $M \to \delta I_c$, the single-neuron *inverted funnel network* network enters the ***Lazy*** regime. In this regime, the dynamics of β converge to the trajectory of linear regression trained by gradient flow and along this trajectory the NTK matrix remains constant.

Conversely, as $\delta \to +\infty$, $M \to |\delta| \frac{\beta^\intercal \beta}{\|\beta\|^2}$, transitions into the ***Lazy-to-Rich*** regime for the network. Here the dynamics of β are constrained to learn the task by only changing their norm while keeping the direction and the NTK matrix fixed – an initial lazy phase. However, if β must change direction to fit the task, and assuming finite δ or that $t \to \infty$ faster than $\delta \to -\infty$, then at some point there will be a slow alignment of β to β_*. In this second phase the NTK matrix will change, confirming the dynamics are lazy-to-rich.

In summary, networks with scalar outputs enter the lazy regime as $\delta \to \infty$, while networks with scalar inputs enter the lazy regime as $\delta \to -\infty$. Conversely, both types of networks enter the "active" regime in opposite directions. Finally, both cases of scalar output and scalar input enter the rich regime when $\delta \to 0$

C.1.3. MULTI-OUTPUT

A fruitful setting for analysis is found in *square networks*, where the dimensions of the input, hidden, and output layers coincide ($d = h = c$). By studying the dependence of M on the conserved quantity $\mathrm{diag}(\Delta)$ and the shape of the network, defined by the the dimensions d, h and c, we can identify the lazy, rich, and lazy-to-rich regimes. We establish that as δ tends towards $\pm\infty$, the network symmetrically transitions into the lazy regime, while approaching zero, it converges into a rich regime. Furthermore, in this setting, we can precisely identify the influence δ_i has on the inductive bias.

We consider the isotropic initialization defined as $\Delta = \delta I_h$ in this section. The conserved quantity becomes $AA^\intercal - WW^\intercal = \delta I_h$.

Lemma 7 $W^\intercal W = \frac{1}{\eta_A}\left(-\frac{\delta}{2}I + \sqrt{\eta_A \eta_W \beta\beta^\intercal + \frac{\delta^2}{4}I}\right)$.

Proof Multiplying on the left and write by $W^\intercal$ and rearranging equation the conservation law we get

$$\eta_A (W^\intercal W)^2 + \delta W^\intercal W = \eta_W \beta\beta^\intercal. \tag{91}$$

Completing the square

$$\eta_A (W^\intercal W)^2 + \delta W^\intercal W + \frac{\delta^2}{4\eta_A} I = \eta_W \beta\beta^\intercal + \frac{\delta^2}{4\eta_A} I. \tag{92}$$

Solving for $W^\intercal W$

$$\left(\sqrt{\eta_A}W^\intercal W + \frac{\delta}{2\sqrt{\eta_A}}I\right)^2 = \eta_W \beta\beta^\intercal + \frac{\delta^2}{4\eta_A}I \tag{93}$$

$$\sqrt{\eta_A}W^\intercal W + \frac{\delta}{2\sqrt{\eta_A}}I = +\sqrt{\eta_W \beta\beta^\intercal + \frac{\delta^2}{4\eta_A}I} \tag{94}$$

$$\sqrt{\eta_A}W^\intercal W = -\frac{\delta}{2\sqrt{\eta_A}}I + \sqrt{\eta_W \beta\beta^\intercal + \frac{\delta^2}{4\eta_A}I}. \tag{95}$$

$$W^\intercal W = \frac{1}{\eta_A}\left(-\frac{\delta}{2}I + \sqrt{\eta_A\eta_W \beta\beta^\intercal + \frac{\delta^2}{4}I}\right)). \tag{96}$$

■

Lemma 8 $A(t)^\intercal A(t) = \frac{1}{\eta_W}\left(\frac{\delta}{2}I + \sqrt{\eta_A\eta_W\beta^\intercal\beta + \frac{\delta^2}{4}I}\right)$.

Proof *For A multiplying on the left and write by $A^\intercal$ and A the conservation equation we get*

$$-\delta A(t)^\intercal A(t) + \eta_W (A(t)^\intercal A(t))^2 = \eta_A \beta\beta^\intercal. \tag{97}$$

Completing the square

$$\eta_W (A(t)^\intercal A(t))^2 - \delta A(t)^\intercal A(t) + \frac{\delta^2}{4\eta_W}I = \eta_A \beta^\intercal\beta + \frac{\delta^2}{4\eta_W}I \tag{98}$$

Solving for $A(t)^\intercal A(t)$

$$\left(\sqrt{\eta_W}A(t)^\intercal A(t) - \frac{\delta}{2\sqrt{\eta_W}}I\right)^2 = \eta_A \beta^\intercal\beta + \frac{\delta^2}{4\eta_W}I \tag{99}$$

$$\sqrt{\eta_W}A(t)^\intercal A(t) - \frac{\delta}{2\sqrt{\eta_W}}I = +\sqrt{\eta_A \beta^\intercal\beta + \frac{\delta^2}{4\eta_W}I} \tag{100}$$

$$\sqrt{\eta_W}A(t)^\intercal A(t) = +\frac{\delta}{2\sqrt{\eta_W}}I + \sqrt{\eta_A \beta^\intercal\beta + \frac{\delta^2}{4\eta_W}I} \tag{101}$$

$$A(t)^\intercal A(t) = +\frac{1}{\eta_W}\left(\frac{\delta}{2}I + \sqrt{\eta_A\eta_W \beta^\intercal\beta + \frac{\delta^2}{4}I}\right). \tag{102}$$

■

Lemma 9 *Assuming an initialization where $\|\beta(0)\|_F > 0$ and $\Delta = \delta I_h$, then the dynamics of the network can be expressed as $vec(\beta) = -Mvec(X^\intercal P)$, where M is defined as*

$$M = \left(\sqrt{\eta_A\eta_W \beta^\intercal\beta + \frac{\delta^2}{4}I} \otimes I\right) + \left(I \otimes \sqrt{\eta_A\eta_W \beta\beta^\intercal + \frac{\delta^2}{4}I}\right). \tag{103}$$

Proof

We start from

$$\text{vec}\left(\dot{\beta}\right) = -\underbrace{(\eta_W A^\intercal A \oplus \eta_A W^\intercal W)}_{M} \text{vec}(X^\intercal X\beta - X^\intercal Y), \tag{104}$$

where$P = (X\beta - Y)$. Replacing the expressions for $W^\intercal W$ (equation 7) and $A^\intercal A$ (equation 8); the vectorised network function dynamics (equation 73) reads

$$vec(\dot{\beta}) = -\left[\frac{\eta_W}{\eta_W}\left(\frac{\delta}{2}I + \sqrt{\eta_A\eta_W\beta^\intercal\beta + \frac{\delta^2}{4}I}\right) \otimes I + I \otimes \frac{\eta_A}{\eta_A}\left(-\frac{\delta}{2}I + \sqrt{\eta_A\eta_W\beta\beta^\intercal + \frac{\delta^2}{4}I}\right)\right] vec(X^\intercal P) \tag{105}$$

$$= -\left[\left(\frac{\delta}{2}I \otimes I\right) + \left(\sqrt{\eta_A\eta_W\beta^\intercal\beta + \frac{\delta^2}{4}I} \otimes I\right) + \left(I \otimes -\frac{\delta}{2}I\right) + \left(I \otimes \sqrt{\eta_A\eta_W\beta\beta^\intercal + \frac{\delta^2}{4}I}\right)\right] vec(X^\intercal P) \tag{106}$$

$$= -\left[\underbrace{\left(\sqrt{\eta_A\eta_W\beta^\intercal\beta + \frac{\delta^2}{4}I} \otimes I\right) + \left(I \otimes \sqrt{\eta_A\eta_W\beta\beta^\intercal + \frac{\delta^2}{4}I}\right)}_{M}\right] vec(X^\intercal P). \tag{107}$$

■

From our expression for $M(\beta)$ we will consider how it simplifies in settings of $\delta \to -\infty, 0, \infty$ allowing us to gain intuition for $M_\delta(\beta)$.

$$M \to \begin{cases} \delta I & \delta \to -\infty. \\ \beta\beta^\intercal \otimes I + I \otimes \beta^\intercal\beta & \delta = 0. \\ \delta I & \delta \to \infty. \end{cases} \tag{108}$$

Lazy. As $\delta \to \pm\infty$ $M \to \delta I_d$, the network transitions into the lazy regime, the dynamics of β converge to the trajectory of linear regression trained by gradient flow and along this trajectory the NTK matrix remains constant, confirming the dynamics are lazy. ***Rich.*** When $\delta = 0$, $M = \beta\beta^\intercal \otimes I + I \otimes \beta^\intercal\beta$. As a result the NTK changes in both magnitude and direction through training, confirming the dynamics are rich.

Computing $M^{-1} = \nabla^2 q_\delta(\beta)$. Consistent with prior analyses, the natural next step of this derivation would be to compute the inverse of M in order to find the potential q_δ indicative of the implicit bias. However, it is not straight forward to take the inverse of a Kronecker sum. Therefore, we turn to a simplification of our setting where we can precisely identify the influence δ has on the inductive bias .

Theorem 10 *For a depth 2 fully connected square network ($d = h = c$), initialized such that $\Delta = \delta I$ for some $\delta \in \mathbb{R}$ and task-aligned such that $\beta(0) = U\Lambda(0)V^\intercal$, where $\Lambda(0)$ is a diagonal matrix of strictly positive singular values, U, V are the singular vectors of $X^\intercal Y$, and $X^\intercal X$ is diagonalizable by V.*

If the gradient flow solution $\beta(\infty)$ satisfies $X^\intercal\beta(\infty) = Y$, then $\beta(\infty) = U\Lambda_* V^\intercal$ where Λ_* solves the constrained optimization,

$$\Lambda_* = \underset{\substack{\Lambda\in \mathrm{Diag}(\mathbb{R}^d) \\ \Lambda_{ii}>0 \quad \textit{for } i=1,\dots,i}}{\arg\min} \quad Q_\delta(\Lambda) \quad \text{s.t.} \quad XU\Lambda V^\intercal = Y \tag{109}$$

where $Q_\delta(\Lambda) = \sum_i^h q_\delta(\Lambda_i) - \nabla q_\delta(\Lambda_i(0))\Lambda_i$ and
$q_\delta(x) = \frac{1}{4}\left(2x\sinh^{-1}\left(\frac{\sqrt{\eta_A\eta_W}2x}{\delta}\right) - \sqrt{\eta_A\eta_W 4x^2+\delta^2} + \delta\right)$.

Proof

Assumption 11 *The singular value decomposition of the network function at initialization is $\mathrm{SVD}(\beta(0)) = V\Lambda(0)U^\intercal$ where $\Lambda(0)$ is a diagonal matrix of strictly positive singular values and U, V are the singular vectors of $X^\intercal Y$, ($\mathrm{SVD}(X^\intercal Y) = VSU^\intercal$),*

Assumption 12 *The input data $X^\intercal X$ is diagonalizable by V.*

Under assumptions 11 and 12, the network is said to be the task-aligned (Saxe et al. [60]), the eigenvectors of the network function β are constant throughout training and equal to U and V respectivaly. We can therefore describe the dynamics of $\Lambda \in \mathrm{Diag}(\mathbb{R}^d)$ with $\Lambda_{ii} > 0 \quad$ for $i = 1,\dots,d$ where $\Lambda = V^\intercal\beta U$.

Rearranging the expressions for $W^\intercal W$ (equation 7) and $A^\intercal A$ (equation 8) in terms of the singular vectors of the task we get

Lemma 13 $A(t)^\intercal A(t) = \frac{1}{\eta_W} U\left(\frac{\delta}{2}I + \sqrt{\eta_A\eta_W\Lambda^2 + \left(\frac{\delta}{2}I\right)^2}\right)U^\intercal$.

Proof

$$A(t)^\intercal A(t) = -\frac{1}{\eta_W}\left(\frac{\delta}{2}I + \sqrt{\eta_A\eta_W\beta^\intercal\beta + \left(\frac{\delta}{2}I\right)^2}\right) \tag{110}$$

$$= \frac{1}{\eta_W}\left(\frac{\delta}{2}UU^\intercal + \sqrt{\eta_A\eta_W U\Lambda^2U^\intercal + \left(\frac{\delta}{2}UU^\intercal\right)^2}\right) \tag{111}$$

$$= \frac{1}{\eta_W}U\underbrace{\left(\frac{\delta}{2}I + \sqrt{\eta_A\eta_W\Lambda^2 + \left(\frac{\delta}{2}I\right)^2}\right)}_{\Lambda_{AA}}U^\intercal. \tag{112}$$

■

Lemma 14 $W^\intercal W = \frac{1}{\eta_A} V\left(-\frac{\delta}{2}I + \sqrt{\eta_A\eta_W\Lambda^2 + \left(\frac{\delta}{2}I\right)^2}\right)V^\intercal$.

Proof

$$W^\intercal W = \frac{1}{\eta_A}\left(\frac{\delta}{2}I + \sqrt{\eta_A\eta_W\beta\beta^\intercal + \left(\frac{\delta}{2}I\right)^2}\right) \tag{113}$$

$$= \frac{1}{\eta_A} V \underbrace{\left(-\frac{\delta}{2}I + \sqrt{\eta_A\eta_W\Lambda^2 + \left(\frac{\delta}{2}I\right)^2}\right)}_{\Lambda_{WW}} V^\intercal. \tag{114}$$

■

Lemma 15 $\dot{\Lambda} = -\sqrt{\eta_A\eta_W 4\Lambda^2 + \delta^2 I}(\Lambda - S)$

Proof

Beggining with the network function dynamic β, rewriting the expression in terms of the eigenvectors U and V

$$\dot{\beta} = -\left(\eta_W(X^\intercal X\beta - X^\intercal Y)A^\intercal A + \eta_A W^\intercal W(X^\intercal X\beta - X^\intercal Y)\right) \tag{115}$$

$$= -\left(\eta_W(X^\intercal X\beta - X^\intercal Y)\frac{1}{\eta_W}U\Lambda_{AA}U^\intercal + \frac{\eta_A}{\eta_A}V\Lambda_{WW}V^\intercal(X^\intercal X\beta - X^\intercal Y)\right) \tag{116}$$

$$= -\left((X^\intercal XV\Lambda U^\intercal - VSU^\intercal)U\Lambda_{AA}U^\intercal + V\Lambda_{WW}V^\intercal(X^\intercal XV\Lambda U^\intercal - VSU^\intercal)\right) \tag{117}$$

$$= -\left(V(\Lambda - S)\Lambda_{AA}U^\intercal + V\Lambda_{WW}(\Lambda - S)U^\intercal\right) \tag{118}$$

$$= -\left(V((\Lambda - S)\Lambda_{AA} + \Lambda_{WW}(\Lambda - S))U^\intercal\right) \tag{119}$$

Projecting $\dot{\beta}$ onto the singular vectors $V^\intercal$ and U, we now consider the dynamics of $\dot{\Lambda}$

$$\dot{\Lambda} = -\left((\Lambda - S)\Lambda_{AA} + \Lambda_{WW}(\Lambda - S)\right) \tag{120}$$

$$= -\left((\Lambda - S)\left(+\frac{\delta}{2}I + \sqrt{\eta_A\eta_W\Lambda^2 + \left(\frac{\delta}{2}I\right)^2}\right) + \left(-\frac{\delta}{2}I + \sqrt{\eta_A\eta_W\Lambda^2 + \left(\frac{\delta}{2}I\right)^2}\right)(\Lambda - S)\right) \tag{121}$$

$$= -\left((\Lambda - S)\left(\sqrt{\eta_A\eta_W\Lambda^2 + \left(\frac{\delta}{2}I\right)^2}\right) + \left(\sqrt{\eta_A\eta_W\Lambda^2 + \left(\frac{\delta}{2}I\right)^2}\right)(\Lambda - S)\right) \tag{122}$$

Given that $\sqrt{\eta_A\eta_W\Lambda^2 + \left(\frac{\delta}{2}I\right)^2}$ is a diagonal matrix and $(\Lambda - S)$ is also a diagonal matrix, the expression for $\eta_A\dot{\eta_W}\Lambda$ simplifies accordingly to

$$\dot{\Lambda} = -M_\Lambda(\Lambda - S) \tag{123}$$

with $M_\Lambda = \sqrt{\eta_A\eta_W 4\Lambda^2 + \delta^2 I}$. ■

As layed out in Appendix B.4.1 the standard analysis for determining the implicit bias of a linear network through mirror flow, if the learning dynamics of the task aligned predictor Λ can be expressed as a mirror flow *for some strictly convex potential $\Phi_\delta(\beta)$,*

$$\dot{\Lambda} = -\left(\nabla^2\Phi_\delta(\Lambda)\right)^{-1}(\Lambda - S) \tag{124}$$

then if the gradient flow solution $\beta(\infty)$ satisfies $X^\intercal\beta(\infty) = Y$, then $\beta(\infty) = U\Lambda_* V^\intercal$ where Λ_* solves the constrained optimization

$$\Lambda_* = \underset{\substack{\Lambda\in \mathrm{Diag}(\mathbb{R}^d)\\ \Lambda_{ii}>0 \quad \text{for } i=1,\dots,h}}{\arg\min} \quad Q_\delta(\Lambda) \quad \text{s.t.} \quad XU\Lambda V^\intercal = Y \tag{125}$$

where $Q_\delta(\Lambda) = \Phi_\delta(\Lambda) - \nabla\Phi_\delta(\Lambda(0))^\intercal\Lambda$

The natural next step of this derivation is to compute the inverse of M_Λ in order to find the potential Φ_δ indicative of the implicit bias. $M_\Lambda = \sqrt{\eta_A\eta_W 4\Lambda^2 + \delta^2 I}$ is a diagonal matrix therefore has a known inverse and

$$(M_\Lambda)^{-1} = \nabla^2\Phi_\alpha(\Lambda) = \left(\sqrt{\eta_A\eta_W 4\Lambda^2 + \delta^2 I}\right)^{-1} \tag{126}$$

Solving for the potential $\Phi_\delta(\Lambda)$**.** *Consider the hypothesis that a scalar function $q(x)$ exists, allowing us to express $\Phi_\delta(\Lambda) = \sum_{i=1}^h q_\delta(\Lambda_i) + c_\delta$. Here, c_δ is a constant chosen to ensure that $\Phi_\delta(\Lambda_{ii}) > 0$ for all $\Lambda_{ii} > 0$ for $i = 1,\dots,h$, and that $\Phi_\delta(0) = 0$. The Hessian of this ansatz takes the form,*

$$\nabla^2\Phi_\delta(\Lambda) = \left(\sqrt{\eta_A\eta_W 4\Lambda^2 + \delta^2 I}\right)^{-1} \tag{127}$$

We now look for a function $q_\delta(x)$ and its derivatives,

$$q_\delta(\Lambda_i) = \frac{1}{4}\left(\sqrt{\eta_A\eta_W}2\Lambda_i \sinh^{-1}\left(\frac{\sqrt{\eta_A\eta_W}2\Lambda_i}{\delta}\right) - \sqrt{\eta_A\eta_W 4\Lambda_i^2 + \delta^2} + \delta\right) \tag{128}$$

The hessian of $q_\delta(x)$ is

$$\nabla^2\left(\sum_i^h q_\delta(\Lambda_i) + c_\delta\right) = \left(\sqrt{\eta_A\eta_W 4\Lambda^2 + \delta^2}\right)^{-1} \tag{129}$$

We find that the inductive bias is given by a hyperbolic entropy potential evaluated at the singular values of β. In the scenario of aligned networks, the distinction between ℓ^1 and ℓ^2 norms loses significance as the network consistently converges to the same solution. Nevertheless, the learning dynamics of β will differ due to the potential q_δ smooth transition between an ℓ^1 and ℓ^2 penalty, distinguishing the rich and lazy regimes. This potential was initially identified as the inductive bias for diagonal linear networks by Woodworth et al. [68]. ■

C.2. Multi-Layer

We now consider the influence of depth by studying a depth-$(l+1)$ linear network, $f(x;\theta) = a^\intercal\prod_{i=1}^l W_i x$, where $W_1 \in \mathbb{R}^{h\times d}$, $W_i \in \mathbb{R}^{h\times h}$ for $1 < i \le l$, and $a \in \mathbb{R}^h$. We assume that the dimensions $d = h$ and that all parameters share the same learning rate $\eta = 1$. For this model the predictor coefficients are computed by the product $\beta = \prod_{i=1}^l W_i^\intercal a \in \mathbb{R}^d$. Similar to our analysis of a two-layer setting, we assume an isotropic initializations of the parameters.

Definition 16 *There exists a $\delta \in \mathbb{R}$ such that $aa^\intercal - W_l W_l^\intercal = \delta\mathbf{I}_h$ and for all $i \in [l-1]$ $W_{i+1}^\intercal W_{i+1} = W_i W_i^\intercal$.*

This assumption can easily be achieved by setting $a = 0$ and $W_i = \alpha O_i$ for all $i \in [l]$, where $O_i \in \mathbb{R}^{d\times d}$ is an random orthogonal matrix and $\alpha \geq 0$. In this case $\delta = -\alpha^2$. Further, notice this parameterization is naturally achieved in the high-dimensional limit as $d \to \infty$ under a standard Gaussian initialization with a variance inversely proportional with width. As in the two-layer setting, this structure of the initialization will remain conserved throughout gradient flow. We now show how two natural quantities of β, its squared norm $\|\beta\|^2$ and its outer product $\beta\beta^\intercal$, can always be expressed as polynomials of $\|a\|^2$ and $W_1^\intercal W_1$ respectively.

Lemma 17 $\|\beta\|^2 = \|a\|^2 \left(\|a\|^2 - \delta\right)^l$.

Proof The norm of the regression coefficients is the product $\|\beta\|^2 = a^\intercal \left(\prod_{i=1}^l W_i\right) \left(\prod_{i=1}^l W_i\right)^\intercal a$. Using the conservation of the initial conditions between consecutive weight matrices, $W_{i+1}^\intercal W_{i+1} = W_i W_i^\intercal$, we can express this telescoped product as $\|\beta\|^2 = a^\intercal \left(W_l W_l^\intercal\right)^d a$. When plugging in the conservation between last two layers, this implies $\|\beta\|^2 = a^\intercal \left(aa^\intercal - \delta I_d\right)^d a$, which expanded gives the desired result. ■

Lemma 18 $\beta\beta^\intercal = \left(W_1^\intercal W_1\right)^{l+1} + \delta \left(W_1^\intercal W_1\right)^l$.

Proof The outer product of the regression coefficients is $\beta\beta^\intercal = \left(\prod_{i=1}^l W_i\right)^\intercal aa^\intercal \left(\prod_{i=1}^l W_i\right)$. Using the conserved initial conditions of the last weights we can factor the outer product as the sum, $\beta\beta^\intercal = \left(\prod_{i=1}^l W_i\right)^\intercal W_l W_l^\intercal \left(\prod_{i=1}^l W_i\right) + \delta \left(\prod_{i=1}^l W_i\right)^\intercal \left(\prod_{i=1}^l W_i\right)$. Both these telescoping products factor using the conservation of the initial conditions between consecutive weight matrices giving the desired result. ■

We now demonstrate how the quadratic terms $|a|^2$ and $W_1^\intercal W_1$ significantly influence the dynamics of β, similar to our analysis in the two-layer setting.

Lemma 19 *The dynamics of β are given by a differential equation $\dot{\beta} = -MX^\intercal \rho$ where M is a positive semi-definite matrix that solely depends on $\|a\|^2$, $W_1^\intercal W_1$, and δ,*

$$M = \left(W_1^\intercal W_1\right)^l + \|a\|^2 \left(\sum_{i=0}^{l-1} (\|a\|^2 - \delta)^i \left(W_1^\intercal W_1\right)^{l-1-i}\right). \tag{130}$$

Proof Using a similar telescoping strategy used in the previous proofs we can derive the form of M, which we leave to the reader. ■

Finally, we consider how the expression for M simplifies in the limit as $\delta \to 0$ allowing us to be precise about the inductive bias in this setting.

Theorem 20 *For a depth-$(l+1)$ linear network with square width ($d = h$) and isotropic initialization β_0 such that $\|\beta(t)\| > 0$ for all $t \geq 0$, then in the limit as $\delta \to 0$, if the gradient flow solution $\beta(\infty)$ satisfies $X\beta(\infty) = y$, then,*

$$\beta(\infty) = \underset{\beta\in\mathbb{R}^d}{\arg\min} \left(\frac{l+1}{l+2}\right) \|\beta\|^{\frac{l+2}{l+1}} - \left(\frac{\beta(0)}{\|\beta(0)\|^{\frac{l}{l+1}}}\right)^\intercal \beta \quad \text{s.t.} \quad X\beta = y. \tag{131}$$

Proof Whenever $\|\beta\| > 0$ and in the limit as $\delta \to 0$, then we can find a unique expression for $\|a\|^2$ and $W_1^\intercal W_1$ in terms of $\|\beta\|^2$ and $\beta\beta^\intercal$,

$$\|a\|^2 = \|\beta\|^{\frac{2}{l+1}}, \qquad W_1^\intercal W_1 = \|\beta\|^{-\frac{2l}{l+1}}\beta\beta^\intercal. \tag{132}$$

Plugged into the previous expression for M results in a positive definite rank-one perturbation to the identity,

$$M = \|\beta\|^{\frac{2l}{l+1}}\mathbf{I}_d + l\|\beta\|^{-\frac{2}{l+1}}\beta\beta^\intercal. \tag{133}$$

Using the Sherman-Morrison formula we find that M^{-1} is

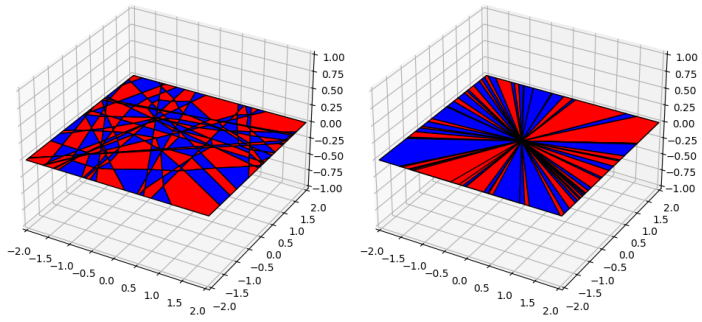
$$M^{-1} = \|\beta\|^{-\frac{2l}{l+1}}\mathbf{I}_d + \left(\frac{l}{l+1}\right)\|\beta\|^{-\frac{4l+2}{l+1}}\beta\beta^\intercal \tag{134}$$

We can now apply a time-warped mirror flow analysis similar to the analysis presented in Appendix B.4. Consider the time-warping function $g_\delta(\|\beta\|) = \|\beta\|^{-\frac{l}{l+1}}$ and the potential $\Phi(\beta) = \left(\frac{l+1}{l+2}\right)\|\beta\|^{\frac{l+2}{l+1}}$, then its not hard to show $M^{-1} = g_\delta(\|\beta\|)\nabla^2\Phi(\beta)$. This gives the desired result. ■

This theorem is a generalization of Proposition 1 derived in [7] for two-layer linear networks in the rich limit to deep linear networks in the rich limit. We find that the inductive bias, $Q(\beta) = (\frac{l+1}{l+2})\|\beta\|^{\frac{l+2}{l+1}} - \|\beta_0\|^{-\frac{l}{l+1}}\beta_0^\intercal\beta$, strikes a balance between attaining the minimum norm solution and preserving the initialization direction, which with increased depth emphasizes the latter.

Appendix D. Piecewise Linear Networks

Here, we elaborate on the theoretical results presented in Section 3. Our goal is to extend the tools developed in our analysis of linear networks to piecewise linear networks and understand their limitations. We focus on the dynamics of the input-output map, rather than on the inductive bias of the interpolating solutions. As discussed in Azulay et al. [7], Vardi and Shamir [67], extending a mirror flow style analysis directly to non-trivial piecewise linear networks is very difficult or provably impossible. In this section, we first describe the properties of the input-output map of a piecewise linear function, then describe the dynamics of a two-layer network, and finally discuss the challenges in extending this analysis to deeper networks and potential directions for future work.



(*a*) With Biases (*b*) Without Biases

Figure 7: **Surface of a ReLU network.** Here we depict the surface of a three-layer ReLU network $f(x;\theta) : \mathbb{R}^2 \to \mathbb{R}$ with twenty hidden units per layer at initialization, comparing configurations with biases (left) and without biases (right). The network with biases partitions input space into convex polytopes that tile input space. The network without biases partitions input space into convex conic sections emanating from the origin. Each region exhibits a distinct activation pattern, allowing the partition to be colored with two colors based on the parity of active neurons. The network operates linearly within each region and maintains continuity across boundaries.

D.1. Surface of a piecewise linear network

The input-output map of a piecewise linear network $f(x;\theta)$, with l hidden layers and h hidden neurons per layer, is comprised of potentially $O(h^{dl})$ connected linear regions, each with their own vector of predictor coefficients [58]. The exploration of this complex surface has been the focus of numerous prior works, the vast majority of them focused on counting and bounding the number of linear regions as a function of the width and depth [3, 27, 28, 50, 54, 58, 62, 66]. The central object in all of these studies is the *activation region*,

Definition 21 *For a piecewise linear network $f(x;\theta)$, comprising m hidden neurons with pre-activation $z_i(x;\theta)$ for $i \in [m]$, let the* activation pattern $\mathcal{A}$ *represent an assignment of signs $a_i \in \{-1, 1\}$ to each hidden neuron. The* activation region $\mathcal{R}(\mathcal{A};\theta)$ *is the subset of input space that generates $\mathcal{A}$,*

$$\mathcal{R}(\mathcal{A};\theta) = \{x \in \mathbb{R}^d \mid \forall i \; a_i z_i(x;\theta) > 0\}. \tag{135}$$

The input-output map is linear within each non-empty activation region and continuous at the boundary between regions. Linearity implies that every non-empty[3] activation region is associated with a *linear predictor* vector $\beta_{\mathcal{R}} \in \mathbb{R}^d$ such that for all $x \in R(\mathcal{A}; \theta)$, $\beta_{\mathcal{R}} = \nabla_x f(x; \theta)$. Continuity implies that the boundary between regions is formed by a hyperplane determined by where the pre-activation for a neuron is exactly zero, $\{x : z_i(x; \theta) = 0\}$. When the neighboring regions have different linear predictors[4], then this hyperplane is orthogonal to their difference, which is a vector in the span of the first-layer weights. Taken together, this implies that the union of all activation regions forms a convex partition of input space, as shown in Fig. 7. We now present a surprisingly simple, yet to the best of our knowledge not previously understood property of this partition:

Proposition 22 (2-colorable) *If $f(x; \theta)$ lacks redundant neurons, implying that every neuron influences an activation region, then the partition of input space can be colored with two distinct colors such that neighboring regions do not share the same color.*

The proof of this proposition is straightforward. There is one color for regions with an even number of active neurons and another for regions with an odd number of active neurons. Because $f(x; \theta)$ lacks redundant neurons, there does not exist a boundary between activation regions where two neurons activations change simultaneously. In this work, we solely utilize this proposition for visualization purposes, as shown in Fig. 7. Nonetheless, we believe it may be of independent interest as it strengthens the connection between the surface of piecewise linear networks and the *mathematics of paper folding*, a connection previously alluded to in the literature [50].

D.2. Dynamics of a two-layer piecewise linear network

We consider the dynamics of a two-layer piecewise linear network without biases, $f(x; \theta) = a^\intercal \sigma(Wx)$, where $W \in \mathbb{R}^{h \times d}$ and $a \in \mathbb{R}^h$. The activation function is $\sigma(z) = \max\{z, \gamma z\}$ for $\gamma \in [0, 1)$, which includes ReLU $\gamma = 0$ and Leaky ReLU $\gamma \in (0, 1)$. We permit $h > d$, which in the limit as $h \to \infty$, ensures the network possesses the functional expressivity to represent any continuous nonlinear function from $\mathbb{R}^d$ to $\mathbb{R}$ passing through the origin. We consider the contribution to the input-output map from a single hidden neuron $i \in [h]$ with parameters $w_i \in \mathbb{R}^d$ and $a_i \in \mathbb{R}$. As in the linear setting, each hidden neuron is associated with a conserved quantity, $\delta_i = \eta_w a_i^2 - \eta_a \|w_i\|^2$. Unlike in the linear setting, this neuron's contribution to the output $f(x_j; \theta)$ is regulated by whether the input x_j is in the neuron's *active halfspace*, $\{x \in \mathbb{R}^d : w_i^\intercal x > 0\}$. Let $C \in \mathbb{R}^{h \times n}$ be the matrix with elements $c_{ij} = \sigma'(w_i^\intercal x_j)$, which determines the activation of the i^{th} neuron for the j^{th} training data point. The subgradient $\sigma'(z) = 1$ if $z > 0$, $\sigma'(z) \in [\gamma, 1]$ if $z = 0$, and $\sigma'(z) = \gamma$ if $z < 0$. These activation functions exhibit positive homogeneity, implying $\sigma(z) = \sigma'(z) z$. Thus, we can express $\sigma(w_i^\intercal x_j) = c_{ij} w_i^\intercal x_j$, allowing us to express the gradient flow dynamics for w_i and a_i as

$$\dot{a}_i = -\eta_a w_i^\intercal \left(\sum_{j=1}^{n} c_{ij} x_j \rho_j \right), \qquad \dot{w}_i = -\eta_w a_i \left(\sum_{j=1}^{n} c_{ij} x_j \rho_j \right), \tag{136}$$

3. While it is trivial to see that for a network $f(x; \theta)$ with m hidden neurons there are 2^m distinct activation patterns, not all activation patterns are attainable. See Raghu et al. [58] for a discussion.
4. It is possible for neighboring regions to have the same linear predictor. Some works define linear regions as maximally connected component of input space with the same linear predictor [28].

where $\rho_j = f(x_j;\theta) - y_j$ is the residual associated with the j^{th} training data point. If we let $\beta_i = a_i w_i$, which determines the contribution of each hidden neuron to the output $f(x_j;\theta)$, then its not hard to see that the gradient flow dynamics of β_i are

$$\dot{\beta}_i = -\underbrace{\left(\eta_w a_i^2 I_d + \eta_a w_i w_i^\intercal\right)}_{M_i}\underbrace{\left(\sum_{j=1}^n c_{ij} x_j \rho_j\right)}_{\xi_i}. \tag{137}$$

As in the linear setting, the matrix $M_i \in \mathbb{R}^{d\times d}$ appears as a preconditioning matrix on the dynamics Using the exact same derivation presented in Appendix B.3, whenever $a_i^2 > 0$, we can express M_i entirely in terms of β_i and δ_i,

$$M_i = \frac{\sqrt{\delta_i^2 + 4\eta_a\eta_w\|\beta_i\|^2} + \delta_i}{2} I_d + \frac{\sqrt{\delta_i^2 + 4\eta_a\eta_w\|\beta_i\|^2} - \delta_i}{2}\frac{\beta_i\beta_i^\intercal}{\|\beta_i\|^2}. \tag{138}$$

However, unlike in the linear setting, the vector $\xi_i \in \mathbb{R}^d$ driving the dynamics is not shared for all neurons because of its dependence on c_{ij}. Additionally, the NTK matrix in this setting depends on M_i and C, with elements $K_{jk} = \sum_{i=1}^h c_{ij} x_j^\intercal \left(\eta_w a_i^2 I_d + \eta_a w_i w_i^\intercal\right) x_k c_{ik}$. Thus, in order to asses the temporal dynamics of the NTK matrix, we must understand the dynamics of M_i and C. We consider a *signed spherical coordinate* transformation separating the dynamics of β_i into its directional $\hat{\beta}_i = \operatorname{sgn}(a_i)\frac{\beta_i}{\|\beta_i\|}$ and radial $\mu_i = \operatorname{sgn}(a_i)\|\beta_i\|$ components, such that $\beta_i = \mu_i\hat{\beta}_i$. Here, $\hat{\beta}_i$ determines the orientation and direction of the halfspace where the i^{th} neuron is active, while μ_i determines the slope of the linear region in this halfspace. These coordinates evolve according to,

$$\dot{\mu}_i = -\sqrt{\delta_i^2 + 4\eta_a\eta_w\mu_i^2}\,\hat{\beta}_i^\intercal\xi_i, \qquad \dot{\hat{\beta}}_i = -\frac{\sqrt{\delta_i^2 + 4\eta_a\eta_w\mu_i^2} + \delta_i}{2\mu_i}\left(I_d - \hat{\beta}_i\hat{\beta}_i^\intercal\right)\xi_i. \tag{139}$$

These equations can be derived directly from Eq. (136) through chain rule similar to Appendix B.2.1. In fact its worth noting that the this change of coordinates is very similar to the change of coordinates used in the single-neuron analysis. Expressed in terms of the parameters, $\hat{\beta}_i = \frac{w_i}{\|w_i\|}$ and $\mu_i = a_i\|w_i\|$.

Appendix E. Experimental Details

E.1. Figure 1: Teacher-Student with Two-layer ReLU Networks

For Fig. 1 we consider a student-teacher setup similar to that in [14], with one-hidden layer ReLU networks of the form $f(x;\theta) = \sum_{i=1}^{m} a_i \sigma(w_i^\intercal x)$, where $f : \mathbb{R}^d \to \mathbb{R}$ and σ is the ReLU activation function. The teacher model, f^{teacher}, has $m = k$ hidden neurons initialized as $w_i^{\text{teacher}} \overset{\text{i.i.d.}}{\sim} \text{Unif}(S^{d-1})$ and $a_i \overset{\text{i.i.d.}}{\sim} \text{Unif}(\{\pm 1\})$ for $i \le k$.

The student, f^{student}, in turn, has h hidden neurons. We use a symmetrized initialization, as considered in [14], where for $i \le h/2$, we sample $w_i \overset{\text{i.i.d.}}{\sim} S^{d-1}$ and $a_i \overset{\text{i.i.d.}}{\sim} \text{Unif}(\{\pm 1\})$, and then for $i \ge \frac{h}{2} + 1$ we symmetrize by setting $w_i = w_{i-h/2}$ and $a_i = -a_{i-h/2}$. This ensures that f^{student} predicts 0 on any input at initialization.

Note that the *base* student initialization described thus far is perfectly balanced at each neuron, that is $\delta_i = 0$ for $i \in [m]$; we also define this to be our setting where the scale τ is 1. In order to transform the base initialization into a particular setting of τ and δ, we first solve for the relative layer scaling α in $\delta^2 = \tau^2(\alpha^2 - \alpha^{-2})$ and then scale each w_i by τ/α and each a_i by $\tau\alpha$.

We obtain a training dataset $\{x^{(i)}, y^{(i)}\}_{i=1}^n$ by sampling $x^{(i)} \overset{\text{i.i.d.}}{\sim} S^{d-1}$ and computing noiseless labels as $y^{(i)} = f^{\text{teacher}}(x^{(i)}; \theta^{\text{teacher}})$. The student is then trained with full-batch gradient descent on a mean square loss objective.

Figure 1 (a).

Here the setting is: $d = 2$, $h = 50$, $k = 3$, and $n = 20$. We sample a single teacher and then train four students with the same base initialization but different configurations of τ and δ: $(\tau = 0.1, \delta = 0)$ and $(\tau = 2, \delta = 0)$ for the left subfigure, and $(\tau = 0.1, \delta = 1)$ and $(\tau = 0.1, \delta = -1)$ for the right subfigure. Training is for 1 million steps at a learning rate of 1e-4.

Figure 1 (b).

Here the setting is: $d = 100$, $m = 50$, $k = 3$, and $n = 1000$, as in Fig. 1c of [14]. Training is performed with learning rate of 5e-3/τ^2. Test error is computed as mean square error over a held-out set of 10,000 datapoints.

We sweep over τ over a logarithmic scale in the range $[0.1, 2]$ and δ over a linear scale in the range $[-1, 1]$. We average over 16 random seeds, where the seed controls the sampling of: the teacher weights θ^{teacher}, the base initialization of θ^{student}, and the training data $\{x^{(i)}\}_{i=1}^n$. In this way, each random seed is used for a sweep over all combinations of τ and δ in the sweep; we simply apply the scaling described above to get to each point on the (τ, δ) grid.

The kernel distance computed is as defined in [19], where here we compute it at time t relative to the kernel at initialization, i.e. $S(t) = 1 - \langle K_0, K_t \rangle / (\|K_0\|_F \|K_t\|_F)$.

E.2. Figures 2, 3, 4: Single-Neuron Linear Network

Figures 2, 3, and 4 were generated by simulating gradient flow using `scipy.integrate.solve_ivp` function with the RK45 method for solving the ODEs, with a relative tolerance of 1×10^{-6} and time span of $(0, 20)$. In the experiments with full-rank data, we used $X^\intercal X = \mathbf{I}_2$, $\beta_* = \left[\begin{smallmatrix} 0 \\ 1 \end{smallmatrix}\right]$, and $\beta_0 = \left[\begin{smallmatrix} -1 \\ 0 \end{smallmatrix}\right]$. For the experiment with low-rank data, we used $X^\intercal X = \left[\begin{smallmatrix} 0.25 & 0.5 \\ 0.5 & 1 \end{smallmatrix}\right]$, $\beta_* = \left[\begin{smallmatrix} 0.44 \\ 0.88 \end{smallmatrix}\right]$, and $\beta_0 = \left[\begin{smallmatrix} 0.4 \\ 0.05 \end{smallmatrix}\right]$. See the discussion in Appendix B.2 for details on how we determined our theoretical predictions. A notebook generating all the figures is provided.

E.3. Figure 5:

Kernel Distance

We trained LeNet-5 [38] (with ReLU nonlinearity and Max Pooling) on MNIST [38]. We use He initialization [29] and divide the first layer weights by α and multiply the last layer weights by α at initialization, which keeps the network functionally the same at initialization. We trained the model for 500 epochs with a learning rate of 1e-4 and a batch size of 512. The parameter distance is defined as the L_2 distance between all the parameters. To quantify the distance between the activations, we binarize the hidden activation with 1 representing an active neuron. We evaluate Hamming distance over all the binarized hidden activations normalized by the the total number of the activations. We use kernel distance [19], defined as $S(t_1, t_2) = 1 - \langle K_{t_1}, K_{t_2} \rangle / \left(\|K_{t_1}\|_F \|K_{t_2}\|_F \right)$, which is a scale invariant measure of similarity between the NTK at two points in time. We subsample 10% of MNIST to evaluate the Hamming distance and kernel distance. All curves in the figure are averaged over 8 runs.

Gabor Filters

We are training a small ResNet based on the CIFAR10 script provided in the DAWN benchmark.[5] The only modification to the provided code base is that we set the weight decay parameter to 0, since this might confound our results. Moreover, we are dividing the convolutional filters weights by a parameter α (after standard initialization) which controls the balancedness of the network. To quantify the smoothness of the filters, we compute the normalized Laplacian of each filter $w_{ij} \in \mathbb{R}^{15\times 15}$, over input $i = (1, 2, 3)$ and output $j = (1, ..., 64)$ channels

$$\text{smoothness}(w_{ij}) := \left\| \frac{w_{ij}}{\|w_{ij}\|_2} * \Delta \right\|_2^2 \tag{140}$$

where the Laplacian kernel is defined as

$$\Delta := \begin{pmatrix} -0.25 & -0.5 & -0.25 \\ -0.5 & 2 & -0.5 \\ -0.25 & -0.5 & -0.25 \end{pmatrix}. \tag{141}$$

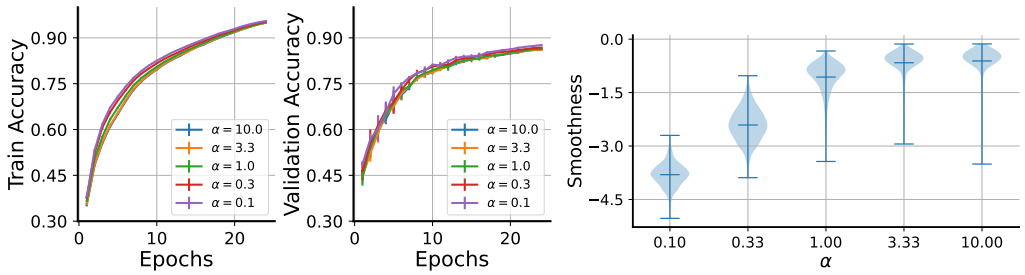
Random Hierarchy Model

We refer to [55], who originally proposed the random hierarchy model (RHM) as a tool for studying how deep networks learn compositional data, for a more in-depth treatment. Here we briefly recap the setup following the notation used in [55].

An RHM essentially lets us build a random classification task with a clear hierarchical structure. The top level of the RHM specifies m equivalent high-level features for *each* class label in $\{1, \ldots, n_c\}$, where each feature has length s and n_c is the number of classes. For example, suppose the vocabulary at the top level is $\mathcal{V}_L = \{a, b, c\}$, $n_c = 2$, $m = 3$, and $s = 2$. Then in a particular instantiation of this RHM, we might have that Class 1 has ab, aa, and ca as equivalent high-level features (this is precisely the example used in Fig.1 of [55]). Class 2 will then have three random high-level features, with the constraint that they are **not** features for Class 1, for example, bb, bc, ac.

Each successive level specifies m equivalent lower-level features for each "token" in the vocabulary at the previous level. For example, if $\mathcal{V}_{L-1} = \{d, e, f\}$, we might have that a can be equivalently represented as de, df, or ff; b and c will each have m equivalent representations of

5. Code available here.



(*a*) Accuracy (*b*) Gabor Smoothness

Figure 8: **Interpreting convolutional filters.** CNN experiments on CIFAR10. We can see in **A)** that all networks achieve comparable training and test accuracy, despite the modification in initialization. However, in **B)** we see that networks with a small initialization ($\alpha < 1$) learn much smoother filters, giving quantiative support to results in Fig. 3. The smoothness is defined as the normalized Laplacian of the filters (see text, eq. 140).

their own. We assume that the vocabulary size, v, is the same at all levels. Therefore, sampling an RHM with hyperparameters n_c, m, s, v requires sampling $mn_c + (L-1)mv$ rules.

In order to sample a datapoint from an RHM, we first sample a class label (e.g. Class 1), then uniformly sample one of the highest level features, (e.g. ab), then for each "token" in this feature we sample lower level features (e.g. $a \to de$, $b \to ee$), and so on recursively. The generated sample will therefore have length s^L and a class label. For training a neural network to perform this classification task, each input is converted into a one-hot representation, which will be of shape (s^L, v), and is then flattened.

We use the code released by [55] to train an MLP of width 64 with three hidden layers to learn an RHM with $L = 3$, $n_c = 8, m = 4, s = 2, v = 8$. The main change we make is allowing for scaling the initialization of the first layer by $1/\alpha$ and the initialization the readout layer by α. We then sweep over $\alpha \in \{0.03, 0.1, 0.3, 1, 3, 10\}$ and over the number of datapoints in the training set, which is specified as a fraction of the total number of datapoints the RHM can generate. We average test accuracy, which is by default computed on a held-out set of 20,000 samples, over six random seed configurations, where each configuration seeds the RHM, the neural network, and the data generation.

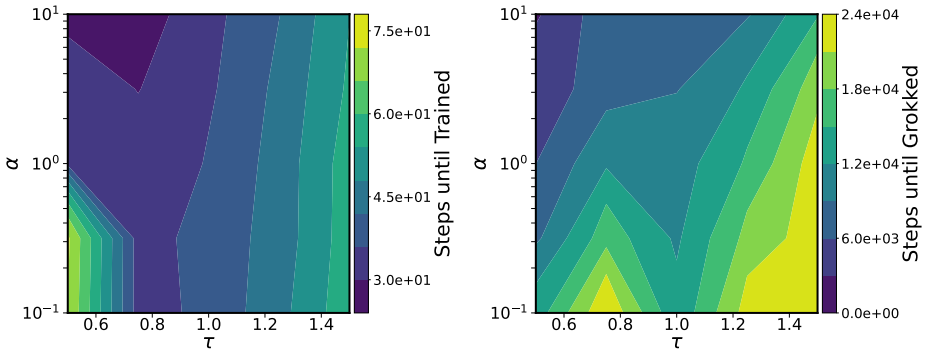
We train with the default settings used in [55], that is stochastic gradient descent with momentum of 0.9, run for 250 epochs with a learning rate initialized at 6.4 (0.1 times width) and decayed with a cosine schedule down to 80% of epochs. The batch size of 128; we do not use biases or weight decay.

Grokking

We are training a one layer transformer model on the modular arithmetic task in Power et al. [57]. Our experimental code is based on an existing Pytorch implementation.[6] The only modifications to the provided code base is that we use a single transformer layer (instead of the default 2-layer model). Prior analysis in Nanda et al. [53] has shown that this model can learn a minimal (attention-based) circuit that solves the task.

6. Code available here.

We study the effects on grokking time (defined as ≥ 0.99 accuracy on the validation data) of two manipulations. Firstly, we divide the embedding weights of the positional and token embeddings by the same balancedness parameter α as in the CNN gabor experiments. Secondly, like in Kumar et al. [34], we multiply the output of the model (i.e., the logits) by a factor τ and divide the learning rate by τ^2.



(*a*) Steps until training (*b*) Steps until grokked

Figure 9: **Transformer Grokking in Modular Arithmetic Task. A)** Shows the number of training steps required until the training accuracy passes a predefined threshold of 99%; we sample scaling $\tau \in \{0.5, 0.75, 1.0, 1.25, 1.5\}$ [34] and balance $\alpha \in \{0.1, 0.3, 1.0, 3.0, 10\}$ on a regular grid over $n = 5$ random initializations with a maximal computational budget of $m = 30,000$ training steps. **B)** Same as **A)**, but reporting the number of training steps required until the test performance passes the predefined threshold of 99%. We clearly see the fastest grokking in an unbalanced rich setting.