Mode Collapse Emerges from Low-Rank Biases in the Learning Dynamics of Generative Models

author names withheld

Under Review for the Workshop on High-dimensional Learning Dynamics, 2026

Abstract

Over the past decade, a growing body of work has established that implicit biases in learning dynamics fundamentally shape the solutions found by neural networks, governing their learned features and generalisation performance. However, despite these strides in deep learning theory, far less is known about representation learning in generative models based on dynamical systems, from normalising flows to diffusion. Importantly, continuous-time generative models can be prone to mode collapse, yet the learning dynamics underpinning such biases remain elusive. To address this, we use tools from dynamical systems and operator theory to show that bounds on the rank of the gradient of the model weights can explain how collapse occurs in generative models. We show that, in both variational inference and flow matching tasks, optimisation induces low-rank biases that encourage parsimonious representation learning but may also cause the learned distribution to collapse. Together these findings point to a trade-off between promoting feature learning while avoiding both memorisation and collapse in dynamical systems-based generative models.

1. Introduction

Neural networks exhibit learning phenomena that are not predicted by classical optimisation theory developed for convex problems [16, 38]. This observation has led to the development of new theoretical approaches aimed at studying the *learning dynamics* of neural networks [43]. This perspective makes it possible to apply tools from dynamical systems theory to analyse the rich behaviours observed during training [4, 42]. For example, gradient flow analyses on idealised settings provide information about stability of fixed points [46], and different forms of implicit regularisation [2, 48]. Despite this success, learning theory is currently under-developed for generative models based on differential equations. Yet, such dynamical systems are at the centre of flow and diffusion models, which are the state-of-the-art for generative models in vision [41, 47] biology [19, 45] and physics [26, 27].

The need for new mathematical tools is reinforced by the presence of *implicit biases* in the distributions learned by generative models. Implicit biases emerge through training, limiting which solutions are learned through optimisation. For example, even when a generative model is expressive enough to learn the full data distribution, training may favour solutions that capture only a subset of the distribution's modes [10, 32, 33, 39, 51, 52], a phenomenon known as "mode collapse". An immediate adverse consequence of mode collapse is a reduction in the diversity of generated samples, where regions of the data distribution may be ignored.

Despite these results, the relationship between implicit biases induced by training and mode collapse in generative models remains unclear. We hypothesise that while implicit biases drive

deep networks to learn low-dimensional data features, which is a beneficial form of representation learning, they are the same mechanism underlying mode collapse in generative dynamical system models. To support this hypothesis, we employ mathematical tools stemming from dynamical systems and operator theory to identify a fundamental trade-off between feature learning and mode collapse in generative modelling.

Related works

Learning theory and implicit biases. Prior work has shown that gradient flow tends to implicitly favour low-complexity solutions in overparametrised networks, including minimum-norm solutions [13, 22, 37], maximum-margin classifiers [48], algorithmically simple functions [44, 50], low-frequency modes [11, 40], and low-rank embeddings [5, 22, 28]. Low-rank biases in particular have been studied extensively in deep linear networks, with sequential feature learning producing plateaus in the loss referred to as "saddle-to-saddle" dynamics[43], where each saddle corresponds to an increase in the rank of the weights [7, 20, 29, 36, 42].

Operator view of learning dynamics. Recent work has suggested that low-rank biases also emerge in recurrent architectures by using adjoint dynamics to represent gradient flow as a composition of linear operators [24, 35]. In particular, one study derived bounds on the rank of weight updates in linear recurrent networks using this operator view [35]. Our work applies a similar operator perspective to the learning dynamics of continuous-time generative models. Notably, Koopman operator theory has been applied to diffusion to interpret the sampling process [6, 9], but not training dynamics.

Collapse in generative models. Generative models are known to exhibit mode collapse, in which the full diversity of modes in the data distribution are not captured [1, 8, 15, 25, 34]. Collapse can come from many sources: for example, distributional biases can cause mode collapse when there is class imbalance due to long-tailed distributions [3, 17, 39]. Mode collapse can also come from the objective, for example recently studied in variational inference [18, 23, 46].

2. Setup

An operator view of gradient flow. We are interested in the learning dynamics of generative models under gradient flow. Using adjoint dynamics, we show that, similarly to the neural tangent kernel of *deep networks*, the gradient of weight parameters of dynamical systems can be written as a composition of linear operators. In appendix A we provide a thorough derivation of these results along with illustrative examples. In summary:

Theorem 1 *The gradient of $\mathcal{L}$ with respect to a linear parameter $W \in \mathbb{R}^{m\times k}$ can be written as the composition of two linear operators:*

$$\nabla_W \mathcal{L} = A \circ X, \qquad A : L^2(\mathbb{R}) \to \mathbb{R}^m, \qquad X : \mathbb{R}^k \to L^2(\mathbb{R})$$

where the linear operators act via integration:

$$A(f) = \mathbb{E}_{\mathbf{x}_0,\mathbb{B}}\left[\int_0^T \mathbf{a}(t) f(t) dt\right], \qquad (X(\mathbf{v}))(T) = \mathbb{E}_{\mathbf{x}_0,\mathbb{B}}\left[\int_0^T \mathbf{x}(t)\cdot \mathbf{v} dt\right]$$

where $f \in L^2(\mathbb{R})$ and $\mathbf{v} \in \mathbb{R}^k$. [PROOF]

These operators are linear by the linearity of integration and expectation. Importantly, they compose as an integral over rank-1 terms. If either of these operators are low-rank, the gradient is also constrained to be low-rank (App. A).

Continuous normalizing flows. Let $p_\theta(\mathbf{x})$ denote a parametric family of distributions, whose parameters θ are optimized to fit a target distribution $p^*(\mathbf{x})$ by minimizing the 'reverse' Kullback Leibler divergence:

$$D_{KL}(p_\theta(\mathbf{x})\|p^*(\mathbf{x})) = \mathbb{E}_{p_\theta}\left[\log p_\theta(\mathbf{x})\right] - \mathbb{E}_{p_\theta}\left[\log p^*(\mathbf{x})\right].$$

Continuous normalising flows compute p_θ as the distribution of pushedforward samples from some base distribution p_0 under the parametrised velocity field $\frac{d\mathbf{x}(t)}{dt} = f(\mathbf{x}(t), t)$ (App. B). The reverse KL is commonly used to train neural samplers when only an unnormalized density function is available [25, 34]. Due to its asymmetric form, the reverse KL is intrinsically 'mode-seeking', as regions in the data distribution are ignored where $p_\theta(\mathbf{x}) = 0$. This provides an ideal case study to analytically study how the interplay between the state and adjoint operators leads to a low-rank bottleneck in the weight updates. As we will demonstrate, this results in collapse becoming an attractor of the gradient flow.

Flow matching. Unlike in CNFs, flow matching models are trained by directly regressing the optimal velocity, $\mathbf{u}_t(\mathbf{x})$ with a neural network model $\mathbf{u}_\theta(\mathbf{x}, t)$:

$$\mathcal{L}_{\text{FM}} = \mathbb{E}_{p_t(\mathbf{x})}\left[\int_0^1 ||\mathbf{u}_t(\mathbf{x}) - \mathbf{u}_\theta(\mathbf{x}, t)||^2 dt\right]$$

where $p_t(\mathbf{x})$ is a chosen probability path which interpolates between base distribution $p_0(\mathbf{x})$ and target distribution $p^*(\mathbf{x})$ over time $t \in [0, 1]$, and $\mathbf{u}_t(\mathbf{x})$ is the vector field that generates it (App. C). In practice, the optimal field is intractable, and so an objective conditioned on individual data samples is used, which yields the same gradient (up to a constant) as the unconditional objective defined above [31].

Model and task. For analytical tractability, we consider a linear model:

$$\tfrac{d}{dt}\mathbf{x}(t) = W\mathbf{x}(t), \qquad \mathbf{x}(0) \sim p_0$$

In this linear model, it is not possible for the flow to split an initial mode (i.e. from Gaussian p_0) to multiple target modes. Instead, we constrain both the initial and data distributions to be equally-weighted Gaussian mixtures, with p_0 and $p^\star$ containing an identical number of modes.

$$p_0(\mathbf{x}) = \sum_{i=1}^m \frac{1}{m}\mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_i, \sigma I_n), \quad p^*(\mathbf{x}) = \sum_{i=1}^m \frac{1}{m}\mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_i^*, \sigma I_n), \quad \boldsymbol{\mu}_i \cdot \boldsymbol{\mu}_j^* = 0, \quad \boldsymbol{\mu}_i \cdot \boldsymbol{\mu}_j = R^2\delta_{i,j}$$

Under these assumptions, there exists a linear transport between p_0 and $p^\star$ by rotating the initial subspace to the data subspace (Fig. 1a), meaning that any collapse is due to implicit bias rather than expressivity.

3. Results

3.1. Mode collapse as an attractor in learning dynamics under a variational objective

To understand how learning biases drive collapse, we first consider a variational inference task, for which generative models are known to have a mode-seeking behaviour. We found that training the

linear CNF with the variational objective made it converge to different values of D_{KL}, corresponding to different numbers of learned data modes (Fig. 1b). Moreover, we found that increasing the number of modes in the data made collapse more likely, with at least one mode typically collapsing as task complexity grew (Fig. 1c).

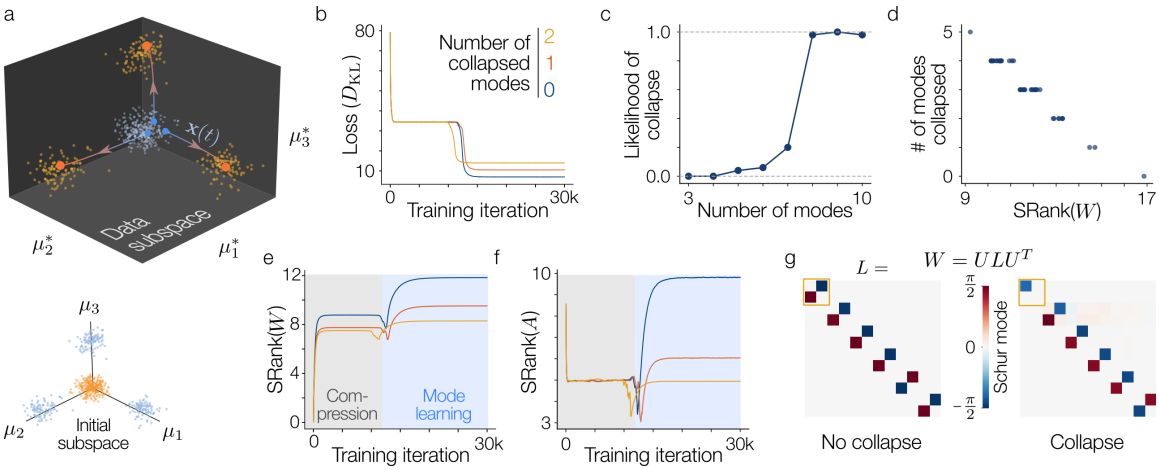

Figure 1: **Stable low-rank solutions in continuous normalising flows drive mode collapse. a.** The model is trained to transport a Gaussian mixture from orthogonal initial and data subspaces. **b.** Loss over iterations in models with different numbers of collapsed modes. **c.** Likelihood that at least one mode collapses as a function of the number of modes in $p_0(\cdot)$ and $p^*(\cdot)$. **d.** The number of collapsed modes scales linearly with the stable rank of the weights W at convergence. **e-f.** The stable rank of the weights and the adjoint over training in models with different numbers of modes. **g.** Top: Schur decomposition of the post-training weights in collapsed and non-collapsed models. Bottom: Non-collapsed models learn a rotational vector field. Collapsed models reveal attraction towards a stable slow manifold with rotational dynamics.

The rank of W directly determines collapse. Mechanistically, a mode collapses when it lies in the kernel of the right singular vectors of the weight matrix. Therefore we expect that the rank deficiency of W should lead to more collapsed modes. To quantify this effect, we computed the stable rank of the weights and observed that lower-rank solutions exhibited a greater number of collapsed modes (Fig. 1d-e). To further characterise these solutions, we performed a Schur decomposition $W = ULU^T$, where L is quasi-lower triangular. Non-collapsed models implemented the correct rotations between initial and target modes, reflected in anti-symmetric blocks of L, each 2×2 block representing a rotation in the subspace from one initial mode to one target mode (Fig. 1g). Instead, collapsed models did not learn all rotations, exhibiting compressive dynamics corresponding to negative real eigenvalues along the diagonal. Both collapsed and non-collapsed solutions were fixed points of the gradient flow (App. B.3).

Adjoint operators in collapsed models are low rank. To understand the underpinnings of this phenomenon, we looked at the stable rank of the adjoint operator over training (Fig. 1f). It revealed that in the mode learning phase, adjoints corresponding to collapsed solutions are lower rank than non-collapsed solutions. This mirrors the decrease in rank we observe in the weights.

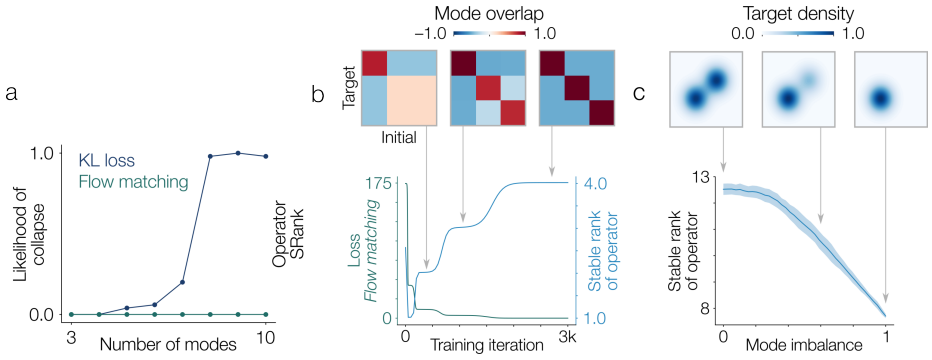

Figure 2: **Flow-matching can have saddle-to-saddle dynamics when the data has mode imbalance. a.** Likelihood of collapse for increasing number of modes for the KL and flow matching objectives. **b.** Flow-matching loss and stable rank of the model over training (here with weights $w = [0.95, 0.025, 0.025]$). Both display staircase-like pattern corresponding to the sequential matching of initial to target modes over learning. **c.** Rank of the state operator as a function of the imbalance of the weights of the mixture.

3.2. Saddle-to-saddle dynamics drive effective collapse during flow matching

In the previous section, we showed that collapse is a stable fixed point in the gradient flow, driven by the inherently mode-seeking KL objective. However, in flow matching it is possible to effectively get mode collapse from saddle-to-saddle dynamics, which lead to sequential feature learning, similar to what has been observed in the learning dynamics of deep linear networks [43].

Linear flow-matching is robust to stable collapse. We trained the same linear model as in the CNF using a flow-matching objective. As expected, the model was able to recover the optimal vector field, and did not exhibit mode collapse, even as the number of modes in the target distribution increased (Fig. 2a). This indicates that the gradient flow of the flow-matching objective does not induce collapse as a stable attractor, in contrast to the variational objective.

Saddle-to-saddle dynamics drive effective collapse. To investigate how data imbalance might drive the model towards biased solutions, we introduced varying degrees of class imbalance by modifying the mixture weights. In this setting, the singular value spectrum of the operator $\bar{X}$ can be explicitly derived as a function of these weights (App. C.2), and used to analytically calculate the rank of the data operator. Indeed, the stable rank of the $\bar{X}$ rapidly decreased as we increased the mode imbalance (Fig. 2c). For imbalanced target distributions, deep linear models exhibit saddle-to-saddle dynamics, corresponding to the sequential learning of modes (Fig. 2b). Moreover, the learning time of a mode evolves super-linearly with the corresponding mixture's weight π, as $O(\pi^{-2})$ (App. C.2). This suggests that although flow-matching prevents stable fixed-points of the gradient flow with collapsed modes, there can be an *effective collapse*, where modes with small weights take longer to be learned, driven by a low rank data operator. This transient effect can be problematic when considering a finite training budget. We also validated these results in a controlled non-linear model trained on a more practical setup, showing that modes are also learned sequentially (App. D).

References

[1] Yassine Abbahaddou and Amine M Aboussalah. A geometry-aware metric for mode collapse in time series generative models. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

[2] Alnur Ali, Edgar Dobriban, and Ryan Tibshirani. The implicit regularization of stochastic gradient flow for least squares. In *Proceedings of the 37th International Conference on Machine Learning*, pages 233–244. PMLR. URL https://proceedings.mlr.press/v119/ali20a.html.

[3] Anonymous. Debiasing diffusion models via score guidance. *Transactions on Machine Learning Research*, 2026.

[4] Luca Arnaboldi, Ludovic Stephan, Florent Krzakala, and Bruno Loureiro. From high-dimensional & mean-field dynamics to dimensionless ODEs: A unifying approach to SGD in two-layers networks. In *Proceedings of Thirty Sixth Conference on Learning Theory*, pages 1199–1227. PMLR. URL https://proceedings.mlr.press/v195/arnaboldi23a.html.

[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] Hanru Bai, Weiyang Ding, and Difan Zou. Hierarchical koopman diffusion: Fast generation with interpretable diffusion trajectory. *arXiv preprint arXiv:2510.12220*, 2025.

[7] Ioannis Bantzis, James B Simon, and Arthur Jacot. Saddle-to-saddle dynamics in deep relu networks: Low-rank bias in the first saddle escape. *arXiv preprint arXiv:2505.21722*, 2025.

[8] Roberto Barceló, Cristóbal Alcázar, and Felipe Tobar. Avoiding mode collapse in diffusion models fine-tuned with reinforcement learning. URL http://arxiv.org/abs/2410.08315.

[9] Nimrod Berman, Ilan Naiman, Moshe Eliasof, Hedi Zisling, and Omri Azencot. One-step offline distillation of diffusion-based models via koopman modeling. *arXiv preprint arXiv:2505.13358*, 2025.

[10] Denis Blessing, Xiaogang Jia, Johannes Esslinger, Francisco Vargas, and Gerhard Neumann. Beyond elbos: A large-scale evaluation of variational methods for sampling. *arXiv preprint arXiv:2406.07423*, 2024.

[11] Abdulkadir Canatar, Blake Bordelon, and Cengiz Pehlevan. Spectral bias and task-model alignment explain generalization in kernel regression and infinitely wide neural networks. 12 (1):2914. ISSN 2041-1723. doi: 10.1038/s41467-021-23103-1. URL https://www.nature.com/articles/s41467-021-23103-1.

[12] Ricky T. Q. Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc. URL https://papers.nips.cc/paper_files/paper/2018/hash/69386f6bb1dfed68692a24c8686939b9-Abstract.html.

[13] Lénaïc Chizat and Francis Bach. Implicit bias of gradient descent for wide two-layer neural networks trained with the logistic loss. In *Proceedings of Thirty Third Conference on Learning Theory*, pages 1305–1338. PMLR. URL https://proceedings.mlr.press/v125/chizat20a.html.

[14] John B Conway. *A course in functional analysis*. Springer, 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. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *ICML'24*, pages 9662–9695. JMLR.org.

[16] Alicia Curth, Alan Jeffares, and Mihaela van der Schaar. A u-turn on double descent: Rethinking parameter counting in statistical learning. *Advances in Neural Information Processing Systems*, 36:55932–55962, 2023.

[17] Jacob Deasy, Nikola Simidjievski, and Pietro Liò. Heavy-tailed denoising score matching. URL http://arxiv.org/abs/2112.09788.

[18] Luigi Fogliani, Bruno Loureiro, and Marylou Gabrié. Annealing in variational inference mitigates mode collapse: A theoretical study on gaussian mixtures. *arXiv preprint arXiv:2602.12923*, 2026.

[19] Tomas Geffner, Kieran Didi, Zuobai Zhang, Danny Reidenbach, Zhonglin Cao, Jason Yim, Mario Geiger, Christian Dallago, Emine Kucukbenli, Arash Vahdat, et al. Proteina: Scaling flow-based protein structure generative models. *arXiv preprint arXiv:2503.00710*, 2025.

[20] Sebastian Goldt, Madhu S. Advani, Andrew M. Saxe, Florent Krzakala, and Lenka Zdeborová. Dynamics of stochastic gradient descent for two-layer neural networks in the teacher-student setup. 2020(12):124010. ISSN 1742-5468. doi: 10.1088/1742-5468/abc61e.

[21] Will Grathwohl, Ricky T. Q. Chen, Jesse Bettencourt, and David Duvenaud. Scalable reversible generative models with free-form continuous dynamics. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=rJxgknCcK7.

[22] Suriya Gunasekar, Blake E Woodworth, Srinadh Bhojanapalli, Behnam Neyshabur, and Nati Srebro. Implicit regularization in matrix factorization. *Advances in neural information processing systems*, 30, 2017.

[23] Anthony GX-Chen, Jatin Prakash, Jeff Guo, Rob Fergus, and Rajesh Ranganath. Kl-regularized reinforcement learning is designed to mode collapse. *arXiv preprint arXiv:2510.20817*, 2025.

[24] James Hazelden, Laura Driscoll, Eli Shlizerman, and Eric Shea-Brown. Kpflow: An operator perspective on dynamic collapse under gradient descent training of recurrent networks. *arXiv preprint arXiv:2507.06381*, 2025.

[25] Jiajun He, Wenlin Chen, Mingtian Zhang, David Barber, and José Miguel Hernández-Lobato. Training neural samplers with reverse diffusive kl divergence. *arXiv preprint arXiv:2410.12456*, 2024.

[26] Benjamin Holzschuh, Simona Vegetti, and Nils Thuerey. Solving inverse physics problems with score matching. *Advances in Neural Information Processing Systems*, 36:61888–61922, 2023.

[27] Jiahe Huang, Guandao Yang, Zichen Wang, and Jeong Joon Park. Diffusionpde: Generative pde-solving under partial observation. *Advances in Neural Information Processing Systems*, 37:130291–130323, 2024.

[28] Minyoung Huh, Hossein Mobahi, Richard Zhang, Brian Cheung, Pulkit Agrawal, and Phillip Isola. The low-rank simplicity bias in deep networks. URL http://arxiv.org/abs/2103.10427.

[29] 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. URL https://ui.adsabs.harvard.edu/abs/2021arXiv210615933J. ADS Bibcode: 2021arXiv210615933J.

[30] Patrick Kidger, James Foster, Xuechen Chen Li, and Terry Lyons. Efficient and accurate gradients for neural sdes. *Advances in Neural Information Processing Systems*, 34:18747–18761, 2021.

[31] Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=PqvMRDCJT9t.

[32] Laurence Illing Midgley, Vincent Stimper, Gregor N. C. Simm, Bernhard Schölkopf, and José Miguel Hernández-Lobato. Flow annealed importance sampling bootstrap. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=XCTVFJwS9LJ.

[33] Tom Minka et al. Divergence measures and message passing. 2005.

[34] Kim A Nicoli, Christopher J Anders, Tobias Hartung, Karl Jansen, Pan Kessel, and Shinichi Nakajima. Detecting and mitigating mode-collapse for flow-based sampling of lattice field theories. *Physical Review D*, 108(11):114501, 2023.

[35] Arthur Pellegrino, N Alex Cayco Gajic, and Angus Chadwick. Low tensor rank learning of neural dynamics. *Advances in Neural Information Processing Systems*, 36:11674–11702, 2023.

[36] Scott Pesme and Nicolas Flammarion. Saddle-to-saddle dynamics in diagonal linear networks. URL https://openreview.net/forum?id=iuqCXg1Gng.

[37] Scott Pesme, Loucas Pillaud-Vivien, and Nicolas Flammarion. Implicit bias of sgd for diagonal linear networks: a provable benefit of stochasticity. *Advances in Neural Information Processing Systems*, 34:29218–29230, 2021.

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

[39] Yiming Qin, Huangjie Zheng, Jiangchao Yao, Mingyuan Zhou, and Ya Zhang. Class-balancing diffusion models. URL http://arxiv.org/abs/2305.00562.

[40] Nasim Rahaman, Aristide Baratin, Devansh Arpit, Felix Draxler, Min Lin, Fred Hamprecht, Yoshua Bengio, and Aaron Courville. On the spectral bias of neural networks. In *Proceedings of the 36th International Conference on Machine Learning*, pages 5301–5310. PMLR. URL https://proceedings.mlr.press/v97/rahaman19a.html.

[41] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022.

[42] David Saad, Sara A Solla, et al. On-line learning in soft committee machines. *Physical Review E*, 52(4):4225–4243, 1995.

[43] Andrew M. Saxe, James L. McClelland, and Surya Ganguli. Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. URL http://arxiv.org/abs/1312.6120.

[44] Harshay Shah, Kaustav Tamuly, Aditi Raghunathan, Prateek Jain, and Praneeth Netrapalli. The pitfalls of simplicity bias in neural networks. *Advances in Neural Information Processing Systems*, 33:9573–9585, 2020.

[45] Ergan Shang, Yuting Wei, and Kathryn Roeder. Predicting the unseen: a diffusion-based debiasing framework for transcriptional response prediction at single-cell resolution. *Proceedings of the National Academy of Sciences*, 122(52):e2525268122, 2025.

[46] Roman Soletskyi, Marylou Gabrié, and Bruno Loureiro. A theoretical perspective on mode collapse in variational inference. URL http://arxiv.org/abs/2410.13300.

[47] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020.

[48] Daniel Soudry, Elad Hoffer, Mor Shpigel Nacson, Suriya Gunasekar, and Nathan Srebro. The implicit bias of gradient descent on separable data. *Journal of Machine Learning Research*, 19(70):1–57, 2018.

[49] Vincent Thibeault, Antoine Allard, and Patrick Desrosiers. The low-rank hypothesis of complex systems. *Nature Physics*, 20(2):294–302, 2024.

[50] Guillermo Valle-Perez, Chico Q Camargo, and Ard A Louis. Deep learning generalizes because the parameter-function map is biased towards simple functions. *arXiv preprint arXiv:1805.08522*, 2018.

[51] Jiancong Xiao, Ziniu Li, Xingyu Xie, Emily Getzen, Cong Fang, Qi Long, and Weijie J Su. On the algorithmic bias of aligning large language models with rlhf: Preference collapse and matching regularization. *Journal of the American Statistical Association*, 120(552):2154–2164, 2025.

[52] Jiayi Zhang, Simon Yu, Derek Chong, Anthony Sicilia, Michael R Tomz, Christopher D Manning, and Weiyan Shi. Verbalized sampling: How to mitigate mode collapse and unlock llm diversity. *arXiv preprint arXiv:2510.01171*, 2025.

Appendix A. Theory

Problem setup. We study a general class of dynamical systems-based generative models:

$$\tfrac{d}{dt}\mathbf{x} = \mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x}, t)dt + G_{\boldsymbol{\theta}}(\mathbf{x}, t)\mathbf{dB}, \quad \mathbf{x}_0 \sim p_0$$

where $\mathbf{x}(t)$ is the state of the system, $\mathbf{dB}$ increments of a Brownian motion, and $\mathbf{f}_{\boldsymbol{\theta}} : \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}^n$ and $G_{\boldsymbol{\theta}} : \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}^{n\times n}$ the drift and diffusion components of the stochastic differential equation with parameters $\boldsymbol{\theta} \in \Theta$. We assume a general loss function:

$$\mathcal{L}(\boldsymbol{\theta}) = \mathbb{E}_{\mathbf{x}_0,\mathbf{B}}\left[\int_0^1 l(\mathbf{x}(t), t)dt\right],$$

where $l : \mathbb{R}^n \times [0, 1] \to \mathbb{R}$ is the instantaneous loss. Depending on the form of $l(\cdot)$, this setting includes D_{KL}, flow-matching and score-matching objectives as special cases.

Here, we are interested in the case where $\boldsymbol{\theta}$ contains weights $W \in \mathbb{R}^{m\times k}$ parametrising the drift and diffusion terms $\mathbf{f}_{\boldsymbol{\theta}}$ and $G_{\boldsymbol{\theta}}$. In particular, the rank of W can constrain the trajectories of the dynamical system and, therefore, the probability distribution learned by the generative model. As we argue below, implicit low-rank biases can arise in the gradient flow learning dynamics, which can in turn bias the model towards collapsing certain modes of the learned distribution. To analyse this phenomenon, we study gradient flow by decomposing it into two linear operators.

Low-rank biases. Key results from functional analysis[14] say that A and X are compact, and admit singular value decompositions:

$$A = U^a \circ S^a \circ V^a, \qquad X = U^x \circ S^x \circ V^x$$

Furthermore, because they are maps between or from a finite-dimensional vector space, they have finite rank (i.e. finitely many non-zero s_i^a and s_i^x). It immediately follows that the singular values of their composition (i.e. of the gradient) are directly determined by i) the overlap between the right singular vectors V^a and the left singular vectors U^x and ii) the singular values S^a, S^x.

For example, usual bounds on the rank of matrix products also hold for linear operators, and the rank of the gradient of the weights can be bounded as: $\mathrm{Rank}(\nabla_W \mathcal{L}) \leq \min\{\mathrm{Rank}(A), \mathrm{Rank}(X)\}$. In practical settings, however, these operators will be full rank, so we may be more interested in a numerically stable definition of the *effective* rank.

Definition 2 *The stable rank of a compact bounded linear operator A is defined as:*

$$\mathrm{SRank}(A) := \frac{\|A\|_*}{\|A\|_2} = \sum_{i=1}^{m} \frac{s_i^a}{s_1^a}$$

where $\|\cdot\|_*$ denotes the nuclear norm and $\|\cdot\|_2$ the spectral norm and s_i the singular values of A.

In particular, if all nonzero singular values are equal, the stable rank coincides with the rank. Furthermore, because it is only normalised by the top singular value, the stable rank is less sensitive to the tails of the singular value distribution than other smooth ranks, such as the participation ratio.

Theorem 3 *The stable rank of the gradient can be bounded as:*

$$\mathrm{SRank}(\nabla_W \mathcal{L}) \leq \frac{\|A\|_2\|X\|_2}{\|A \circ X\|_2} \min\left\{\mathrm{SRank}(A), \mathrm{SRank}(X)\right\}.$$

where A and X are the operators defined above. [PROOF]

Proof of theorem 1

In this section, we derive the exact gradient of a generalised loss function in a nonlinear flow model, with respect to a linear weight parameter W. We consider a system governed by a Stratonovich stochastic differential equation (SDE), where W affects the instantaneous loss, the drift, and the diffusion terms through potentially distinct feature inputs. This derivation encompasses the three types of losses commonly encountered in generative modelling with dynamical systems: variational objectives, flow and score matching.

Let $\mathbf{x}(t) \in \mathbb{R}^d$ denote the state of the system at time $t \in [0, T]$, evolving according to the Stratonovich SDE:

$$\mathbf{dx}(t) = \mathbf{f}_W(\mathbf{x}(t), t)dt + G_W(\mathbf{x}(t), t) \circ \mathbf{dB}, \qquad \mathbf{x}_0 \sim p_0$$

where $\mathbf{B}(t)$ is a standard Brownian motion, $\mathbf{f}$ is the drift vector field, and G is the diffusion matrix field. We use the Stratonovich integral ($\circ$) over the Itô integral, but we note that one and the other can be interchanged.

We evaluate the system via a continuous-time expected loss functional:

$$\mathcal{L}(W) = \mathbb{E}_{\mathbf{x}_0, \mathbb{B}} \left[\int_0^T l(\mathbf{x}(t), t)dt \right]$$

The core architectural assumption is that the parameter matrix $W \in \mathbb{R}^{k \times q}$ acts linearly. For the sake of generality, we assume that the weight parameter could potentially be part of any of the loss, drift and diffusion functions. In several cases the drift and diffusion models are parametrised by different networks, in which case any particular weight would only be part of one of them. In specific scenarios, such as physics-informed neural networks, and some generative architecture, a network and its derivative can be parametrising the drift and diffusion terms at once, in which case W appears in both terms.

To account for this, we assume W may receive distinct post-activation feature vectors $\mathbf{h}_l(t), \mathbf{h}_\mathbf{f}(t), \mathbf{h}_G(t) \in \mathbb{R}^q$ for the loss, drift, and diffusion terms respectively. The pre-activations feature vectors are thus given by:

$$\mathbf{z}_l(t) = W\mathbf{h}_l(t), \quad \mathbf{z}_\mathbf{f}(t) = W\mathbf{h}_\mathbf{f}(t), \quad \mathbf{z}_G(t) = W\mathbf{h}_G(t) \in \mathbb{R}^k$$

Our goal is to derive $\nabla_W \mathcal{L}$ in terms of two operators, one which will be directly related to the post-activation vector, and one adjoint operator.

Adjoints. Because the state $\mathbf{x}(t)$ depends on W through the integration of the SDE, we cannot differentiate $\mathcal{L}(W)$ by simply moving the gradient operator inside the expectation. Instead, we define the adjoint $\mathbf{a}(t) \in \mathbb{R}^d$, representing how small changes in the trajectory of the dynamical system relate to small changes in the loss function:

$$\mathbf{a}(t) = \frac{\partial L}{\partial \mathbf{x}(t)}$$

Classic results state that this adjoint is the solution to the following dynamical system:[30]

$$\mathbf{da}(t) = -\left(\nabla_\mathbf{x} l^\top + (\nabla_\mathbf{x} \mathbf{f})^\top \mathbf{a}(t)\right) dt - (\nabla_\mathbf{x} G)^\top \mathbf{a}(t) \circ \mathbf{dB}(t), \quad \mathbf{a}(T) = \mathbf{a}_T$$

The gradient of model parameters can be written in terms of this adjoint (where we have dropped the dependence on $\mathbf{x}(t)$ for notational clarity):

$$\nabla_\theta \mathcal{L} = \mathbb{E}\left[\int_0^T \left(\nabla_\theta l + (\nabla_\theta \mathbf{f})^\top \mathbf{a}(t)\right) dt + \int_0^T (\nabla_\theta G)^\top \mathbf{a}(t) \circ d\mathbf{B}(t)\right]$$

In our setting, the parameter of interest is the matrix W, which acts linearly, thus endowing the SDE of the gradient with a particular structure.

We evaluate the parameter Jacobians $\nabla_W l$, $\nabla_W \mathbf{f}$, and $\nabla_W G$ by applying the chain rule through the linear projections $\mathbf{z}_l, \mathbf{z}_\mathbf{f}$, and $\mathbf{z}_G$. First, the terms of the loss which do not depend on $\mathbf{x}$ have gradients independent of the adjoint. Since $\mathbf{z}_l(t) = W\mathbf{h}_l(t)$, differentiating with respect to the matrix W gives an outer product:

$$\nabla_W l = \mathbf{a}_l(t) \otimes \mathbf{h}_l(t)$$

where $\mathbf{a}_l = \partial_{\mathbf{z}_l} l(t)$. The sensitivity of the drift and diffusion terms are mediated by the adjoint state $\mathbf{a}(t)$. We define the drift error vector $\mathbf{a}_\mathbf{f}(t) \in \mathbb{R}^k$ by pulling the adjoint back through the drift function:

$$\mathbf{a}_\mathbf{f}(t) = \left(\nabla_{\mathbf{z}_\mathbf{f}} \mathbf{f}\right)^\top \mathbf{a}(t)$$

Applying the chain rule through $\mathbf{z}_\mathbf{f}(t) = W\mathbf{h}_\mathbf{f}(t)$ gives:

$$(\nabla_W \mathbf{f})^\top \mathbf{a}(t) = \mathbf{a}_\mathbf{f}(t) \otimes \mathbf{h}_\mathbf{f}(t)$$

Similarly:

$$\mathbf{a}_G(t) = \left(\nabla_{\mathbf{z}_G} G(\mathbf{x}(t), \mathbf{z}_G(t), t)\right)^\top \mathbf{a}(t) \qquad (\nabla_W G)^\top \mathbf{a}(t) = \mathbf{a}_G(t) \otimes \mathbf{h}_G(t)$$

Substituting these rank-1 matrices back into the gradient equation, we obtain the exact gradient of the loss with respect to the weight matrix:

$$\nabla_W \mathcal{L} = \mathbb{E}\left[\int_0^T \Big(\mathbf{a}_l(t) \otimes \mathbf{h}_l(t) + \mathbf{a}_\mathbf{f}(t) \otimes \mathbf{h}_\mathbf{f}(t)\Big) dt + \int_0^T (\mathbf{a}_G(t) \otimes \mathbf{h}_G(t)) \circ d\mathbf{B}(t)\right]$$

In particular, the instantaneous gradient is rank 3.

Gradient as an operator. The gradient above can be rewritten as:

$$\nabla_W \mathcal{L} = A_l \circ H_l + A_\mathbf{f} \circ H_\mathbf{f} + A_G \circ H_G$$

where $A_{(\cdot)}$ and $H_{(\cdot)}$ are linear operators:

$$A_{(\cdot)} : L^2(\mathbb{R}) \to \mathbb{R}^k, \qquad A_{(\cdot)}(f) = \mathbb{E}_{\mathbf{x}_0, \mathbb{B}}\left[\int_0^T \mathbf{a}_{(\cdot)}(t) f(t) dt\right]$$

First, we briefly describe how more specific losses commonly used in generative modelling fall within this framework, before describing general spectral properties of the gradient in terms of these operators.

Proof of theorem 3

We've demonstrated above that, even with highly nonlinear models with general losses, the gradient of weight parameters can be written as a sum of compositions of linear operators.

$$\nabla_W \mathcal{L} = A_l \circ H_l + A_{\mathbf{f}} \circ H_{\mathbf{f}} + A_G \circ H_G$$

where $A_{(\cdot)}$ are *adjoint* operators and $H_{(\cdot)}$ *state* operators. In case of linear models as used in the main text, the state operator is the state of the system itself, hence we call it X. The adjoint operator directly dependent on the residual. As these operators are linear, and because they are maps from or to a finite dimensional space (the state-space) they are compact. Thus they admit singular value decompositions:

$$A_{(\cdot)} = U^a \circ S^a \circ V^a \qquad H_{(\cdot)} = U^h \circ S^h \circ V^h$$

where:

$$U^a \in \mathbb{R}^{k\times k}, \quad S^a \in \mathbb{R}^k, \quad V^a : L^2(\mathbb{R}) \to \mathbb{R}^k$$

and:

$$U^h : \mathbb{R}^q \to L^2(\mathbb{R}), \quad S^h \in \mathbb{R}^q, \quad V^h \in \mathbb{R}^{q\times q}$$

such that their composition is a linear map $\nabla_W \mathcal{L} \in \mathbb{R}^{k\times q}$:

$$\nabla_W \mathcal{L} : \mathbb{R}^q \xrightarrow{V^h} \mathbb{R}^q \xrightarrow{S^h} \mathbb{R}^q \xrightarrow{U^h} L^2(\mathbb{R}) \xrightarrow{V^a} \mathbb{R}^k \xrightarrow{S^a} \mathbb{R}^k \xrightarrow{U^a} \mathbb{R}^k$$

Stable rank. A natural consequence of this structure in the gradient is that if either operator is low rank then the gradient will be low rank as it is generally true that:

$$\mathrm{Rank}(X \circ Y) \leq \min(\mathrm{Rank}(X), \mathrm{Rank}(Y))$$

This can for example occur when the model weights are already low rank. In both examples above, we had that the adjoint the W_1 matrix had the form:

$$\mathbf{a}_{(\cdot)}(t) = \mathrm{diag}(\phi'(W_1\mathbf{x}(t)))W_2^\top(\,\cdot\,)$$

Thus the rank of the operator is determined by the W_2 weights. For linear networks, as considered in the main text, a consequence of this is that the rank of the gradient of model weights bound each other:

$$\mathrm{Rank}(A_{(\cdot)}) \leq \mathrm{Rank}(W_2) \quad \Longrightarrow \quad \nabla_{W_1}\mathcal{L} \leq \mathrm{Rank}(W_2)$$

this can cause an entrainment of the weight learning dynamics towards lower-rank solutions.

In practice weights are never exactly low-rank. However, we can characterise their *effective* rank via the stable rank:

$$\mathrm{SRank}(W) = \frac{\|W\|_*}{\|W\|_2} = \sum_{i=1}^{\min(k,q)} \frac{s_i}{s_1}$$

Other notions of effective rank include the participation ratio or the energy rank[49]. In general, the stable rank of the product of two matrices can be expressed in terms of the stable rank of each individual matrices. Indeed, define the Schatten norm of a linear operator:

$$\|X\|_{Sc(r)} = \mathrm{Tr}(S^r)^{1/r}$$

where S is that of the SVD of X. For all $r^{-1} \leq p^{-1} + q^{-1}$ the Schatten norm of the composition of linear operators is satisfies:

$$\|X \circ Y\|_{Sc(r)} \leq \|X\|_{Sc(q)} \|Y\|_{Sc(p)}$$

In particular, taking $r = 1$ and $p = \infty$ (resp. 1) and $q = 1$ (resp. ∞), yields the bound:

$$\|X \circ Y\|_* \leq \min\{\|Y\|_2 \|X\|_*, \|X\|_2 \|Y\|_*\}$$

multiplying and dividing the first and second argument of the $\min$ by the spectral norm of Y and X respectively gives:

$$\|X \circ Y\|_* \leq \|Y\|_2 \|X\|_2 \min\{\mathrm{SRank}(X), \mathrm{SRank}(Y)\}$$

or:

$$\mathrm{SRank}(X \circ Y) \leq \frac{\|X\|_2 \|Y\|_2}{\|X \circ Y\|_2} \min\{\mathrm{SRank}(X), \mathrm{SRank}(Y)\}$$

Or more specifically for our weight gradient:

$$\mathrm{SRank}(\nabla_W \mathcal{L}) \leq \frac{\|A\|_2 \|H\|_2}{\|A \circ H\|_2} \min\left\{\mathrm{SRank}(A), \mathrm{SRank}(H)\right\}.$$

Thus, in the examples above, the rank of the gradient of the hidden layer weights are bounded by the rank of the decoder:

$$\mathrm{SRank}(\nabla_{W_1} \mathcal{L}) \leq \gamma \, \mathrm{SRank}(W_2)$$

where γ depends on the spectral norms of the operators.

Appendix B. Collapse in continuous normalising flow

B.1. Linear continuous normalizing flows

In section 3.1 we used a linear continuous normalising flow model. The model is trained to minimize the reverse-KL divergence:

$$D_{\mathrm{KL}}(p_1(\mathbf{x}) || p^*(\mathbf{x})) = \mathbb{E}_{\mathbf{x} \sim p_1} \left[\log p_1(\mathbf{x}) - \log p^*(\mathbf{x})\right]$$

Following previous work[21], we define a parametrisation of the velocity field, $\frac{d\mathbf{z}}{dt} = f(\mathbf{z}(t), t)$. We can rewrite this using the learned distribution at time t, integrated from $t \in [0, 1]$:

$$\begin{aligned}
&\mathbb{E}_{\mathbf{z}(0) \sim p_0} \left[\log p_0(\mathbf{z}(0)) + \int_0^1 \log p_t(\mathbf{z}(t)) dt - \log p^*(\mathbf{z}(1))\right] \\
&\overset{(1)}{=} \mathbb{E}_{\mathbf{z}(0) \sim p_0} \left[\log p_0(\mathbf{z}(0)) - \int_0^1 \nabla \cdot f(\mathbf{z}(t), t) dt - \log p^*(\mathbf{z}(1))\right]
\end{aligned}$$

where p_t and $\mathbf{z}(t)$ are the learned distribution and pushed forward initial sample at time t, and in (1) we used the instantaneous change of variables formula[12]. To compute gradients with respect to the loss, we define an augmented ODE:

$$\frac{d\mathbf{x}(t)}{dt} = \begin{bmatrix} f(\mathbf{z}(t)) \\ -\nabla \cdot f(\mathbf{z}(t), t) \end{bmatrix} \qquad \mathbf{x}(0) = \begin{bmatrix} \mathbf{z}(0) \\ 0 \end{bmatrix}$$

and integrate forward in time using a differentiable ODE solver to obtain $\mathbf{z}(1)$ and the accumulated log-density of the learned distribution. Backpropagating through the ODE solver obtains the gradients.

Definition 4 *For a dynamical system $\dot{\mathbf{x}} = \mathbf{f}_\theta(\mathbf{x}_t, t)$ defined on $t \in [0, 1]$, with loss functional $\mathcal{L} : \mathbb{R}^n \to \mathbb{R}$, the state adjoint is defined as $\mathbf{a}(t) = \frac{d\mathcal{L}(\mathbf{x}(1))}{d\mathbf{x}(t)}$. Its solution is given by the following dynamical system and terminal condition:*

$$\frac{d}{dt}\mathbf{a} = -J_{\mathbf{f}}^\top \mathbf{a}, \qquad \mathbf{a}(1) = \frac{d\mathcal{L}}{d\mathbf{x}(1)}$$

where $J_{\mathbf{f}}$ is the Jacobian of $\mathbf{f}_\theta$.

For the linear model and variational loss above, the adjoint dynamical system is therefore given by:

$$\frac{d}{dt}\mathbf{a} = -W^\top \mathbf{a}, \qquad \mathbf{a}(1) = -\nabla_{\mathbf{x}} \log p^\star(\mathbf{x}(1)) := \mathbf{a}_1$$

We can then analytically derive the gradient as a Fréchet derivative of the matrix exponential of $W^\top$.

Theorem 5 *The gradient of the weights is given by:*

$$\nabla_W \mathcal{L} = A \circ X = \int_0^1 \mathbf{a}(t) \otimes \mathbf{x}(t) dt$$

which simplifies to:

$$A \circ X = \mathbb{E}_{\mathbf{x}_0 \sim p_0}\left[\int_0^1 \exp\left(W^\top(1-t)\right)(\mathbf{a}(1) \otimes \mathbf{x}(0)) \exp\left(W^\top t\right) dt\right]$$

[PROOF]

We initialised weights $w_{ij} \sim \mathcal{N}(0, \sigma_{\text{init}}/\sqrt{n})$ where the dimension $n = 2m$ and m is the number of initial and target modes. For simulations we used a differentiable ODE solver with the Tsit5 method and adaptive step size. To optimize the system we used SGD in Optax. The hyperparameters are as follows:

m	σ_{init}	lr	Iterations	Batch size	Optimizer	ODE solver	PID; rtol, atol
3-10	$1e^{-4}$	$1e^{-3}$	30k	20k	SGD	Tsit5	$1e^{-3}$, $1e^{-6}$

Identifying collapsed modes Given a trained model with weights W, we can compute the pushforward of some initial mode $\boldsymbol{\mu}_i$ analytically as $e^W \boldsymbol{\mu}_i$. Thus, we can compute the overlap between each pushed forward initial mode and each target mode $\boldsymbol{\mu}_j^*$ (normalized by the mode radius) as an $m \times m$ matrix: $s_{ij} = e^W \boldsymbol{\mu}_i \cdot \boldsymbol{\mu}_j^*$. We can identify which target mode each initial mode has been assigned to by taking an argmax over the rows of $\mathbf{s}$, since each initial is assigned to exactly one target. If one or more target modes have no initial modes assigned to it, then collapse has occurred.

Computing the operator rank In figure **??**c we plot the stable rank of the data operator over training. From theorem 1 and theorem 5, we can write the gradient as a composition of two linear operators, $\nabla_W \mathcal{L} = A \circ X$, where:

$$A = -\mathbb{E}_{\mathbf{x}_0}\left[\int_0^T e^{(1-t)W^\top} \nabla_{\mathbf{x}_1} \log p_{\text{data}}(\mathbf{x}_1) dt\right], \qquad X = \mathbb{E}_{\mathbf{x}_0}\left[\int_0^1 e^{tW} \mathbf{x}_0 dt\right]$$

where $\mathbf{x}_1 = e^W \mathbf{x}_0$. There is a fundamental theorem in functional analysis that says that these operators admit a singular value decomposition. Since $rank(A) \leq min(dim((R^n)^*) = n$ and $dim(L^2) = \infty$, it has at most n non-zero singular values. By discretising the integral with q initial samples and T time points we can write $A \in (R^n)^* \otimes R^{T \times q}$, on which we can perform singular value decomposition and compute the stable rank for each training time.

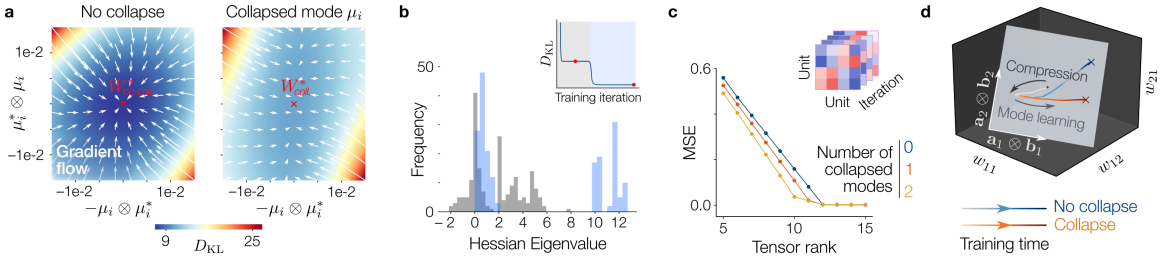

Figure 3: **Low-rank learning dynamics of CNF model. a.** Visualisation of gradient flow near collapsed and non-collapsed fixed points. **b.** Empirical distributions of the eigenvalues of the Hessian, evaluated at the compression slow-point (grey) and at the end of training (blue) for a non-collapsed model. **c.** MSE of a tensor rank-k decomposition of the weight tensor formed by stacking W over training. **d.** Low-tensor-rank subspace of the weight space highlighting how the trajectory of the collapsed and non-collapsed models split during the mode learning phase.

B.2. Linear CNF adjoint

Proof of theorem 5

Following from theorem 1, the gradient flow can be written as

$$\nabla_W \mathcal{L} = A \circ X = \int_0^1 \mathbf{a}(t) \otimes \mathbf{x}(t) dt$$

As previously shown, for a linear model the adjoint has dynamics

$$\frac{d}{dt}\mathbf{a}(t) = -W^\top \mathbf{a}(t)$$

This admits the solution

$$\mathbf{a}(t) = \exp(W^\top (1-t))\mathbf{a}(1)$$

where the initial condition is given by $\mathbf{a}(1) = -\nabla_{\mathbf{x}} \log p_{\text{data}}(\mathbf{x}(1))$. The state of the system at time t for the linear model is

$$\mathbf{x}(t) = \exp(Wt)\mathbf{x}_0$$

Putting these together, we obtain

$$A \circ X = \int_0^1 \exp(W^\top (1-t))\mathbf{a}(1) \otimes \mathbf{x}(0) \exp(W^\top t) dt.$$

B.3. Fixed points of the gradient flow

First, note that:

$$\nabla_W \mathcal{L} = \nabla_{\mathbf{x}} D_{\mathrm{KL}}(p_1, p_{\mathrm{data}}) = \nabla_{\mathbf{x}} \mathbb{E}\left[\log p_{\mathrm{data}}(\mathbf{x}(1)) - \log p_1(\mathbf{x}(1))\right]$$

By the continuity equations:

$$\log p(\mathbf{x}(1)) = \log p(\mathbf{x}(0)) - \int_0^1 \mathrm{div}(W\mathbf{x}(t))dt = \log p(\mathbf{x}(0)) - \mathrm{Tr}(W)$$

Hence, because $\mathbf{x}(0)$ is independent of W:

$$\nabla_W \mathcal{L} = \nabla_W \mathbb{E}\left[\log p_{\mathrm{data}}(\mathbf{x}_1)\right] - I$$

That is, the gradient is given by:

$$\nabla_W \mathcal{L} = \mathbb{E}\left[\int_0^1 e^{-W^\top(1-t)}\mathbf{a}(1) \otimes \mathbf{x}(0) e^{W^\top t} dt\right] - I$$

where $\mathbf{a}(1) = \nabla_{\mathbf{x}} \log p_{\mathrm{data}}(\mathbf{x}(1))$ is the score of the data distribution. We are interested in the fixed points $\nabla_W \mathcal{L} = 0$, that is:

$$I = \mathbb{E}\left[\int_0^1 e^{-W^\top(1-t)}\mathbf{a}(1) \otimes \mathbf{x}(0) e^{W^\top t} dt\right]$$

Data assumptions. For a Gaussian Mixture, the score can be derived explicitly:

$$\nabla_{\mathbf{x}} \log p_{\mathrm{data}}(\mathbf{x}(1)) = \nabla_{\mathbf{x}} \log \sum_{i=1}^m w_i \mathcal{N}(\mathbf{x}(1); \boldsymbol{\mu}_i, \Sigma_i)$$

We make the additional assumptions that $w_i = 1/m$ and $\boldsymbol{\mu}_i^* \cdot \boldsymbol{\mu}_j^* = \delta_{i,j}$ and $\Sigma_i = \sigma^2 I$. Then:

$$\nabla_{\mathbf{x}} \log p_{\mathrm{data}}(\mathbf{x}(1)) = \sum_{i=1}^m \gamma_i(\mathbf{x}(1))(\boldsymbol{\mu}_i^* - \mathbf{x}(1))$$

where:

$$\gamma_i(\mathbf{x}(1)) = \frac{\mathcal{N}(\mathbf{x}(1); \boldsymbol{\mu}_i^*, \sigma^2 I)}{\sum_{j=1}^m \mathcal{N}(\mathbf{x}(1); \boldsymbol{\mu}_j^*, \sigma^2 I)}$$

We can further assume, that, since we are interested in fixed points of the gradient flow, $\mathbf{x}(1)$ should be near one of the data modes (otherwise the D_{KL} will be large), and if $\sigma \ll 1$ the density of all other Gaussians will be vanishingly small:

$$\gamma_i \approx \delta_{i,j}$$

for that corresponding mode. Thus we focus on the case where:

$$\nabla_{\mathbf{x}} \log p_{\mathrm{data}}(\mathbf{x}(1)) \approx \frac{1}{\sigma^2}(\boldsymbol{\mu}_i^* - \mathbf{x}(1))$$

for some i. Thus, the gradient can be written as:

$$\nabla_W \mathcal{L} = \frac{1}{\sigma^2}\mathbb{E}\left[\int_0^1 e^{W^\top(1-t)}(\boldsymbol{\mu}_i^* - \mathbf{x}(1)) \otimes \mathbf{x}(0) e^{W^\top t} dt\right] - I$$

Or:

$$\nabla_W \mathcal{L} = \frac{1}{\sigma^2}\mathbb{E}\left[\int_0^1 e^{W^\top(1-t)}(\boldsymbol{\mu}_i^* - e^W\mathbf{x}(0)) \otimes \mathbf{x}(0) e^{W^\top t} dt\right] - I$$

The fixed points are thus given by:

$$-\sigma^2 e^W = \mathbb{E}\left[\int_0^1 e^{-W^\top t}(\boldsymbol{\mu}_i^* - e^W\mathbf{x}(0)) \otimes \mathbf{x}(0) e^{W^\top t} dt\right]$$

Optimal solution. An easy guess as to a solution is the rotation of the initial to target means. Such a solution can be written as:

$$W = ULU^T$$

where $U = [\boldsymbol{\mu}_1, \boldsymbol{\mu}_1^*, ..., \boldsymbol{\mu}_m, \boldsymbol{\mu}_m^*, \mathbf{u}_{2m+1}, ..., \mathbf{u}_n]$ where $\mathbf{u}_i$ are arbitrary (since they have the same zero eigenvalue), and L is given by:

$$\begin{pmatrix} B_1 & & & \\ & \ddots & & \\ & & B_m & \\ & & & D \end{pmatrix}, \quad B_i = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}, \quad D = \begin{pmatrix} 0 & & \\ & \ddots & \\ & & 0 \end{pmatrix}$$

Then,

$$e^{Wt} = Ue^{Lt}U^T$$

Writing the fixed-point equation in this basis and noticing that $-L = L^\top$ we get:

$$-\sigma^2 U e^L U^\top = \mathbb{E}\left[\int_0^1 U e^{Lt} U^\top (\boldsymbol{\mu}_i^* - U e^L U^\top \mathbf{x}(0)) \otimes \mathbf{x}(0) U e^{-Lt} U^\top dt\right]$$

Now noting that $\mathbf{x}(0) = \boldsymbol{\mu}_i + \sigma\xi$ and cancelling the U matrices on both sides:

$$\sigma^2 e^L = \mathbb{E}\left[\int_0^1 e^{Lt}(\mathbf{e}_{2i} - e^L(\mathbf{e}_{2i-1} + \sigma U\xi)) \otimes (\mathbf{e}_{2i-1} + \sigma U\xi) e^{-Lt} dt\right]$$

We can expand:

$$-\sigma^2 e^L = \mathbb{E}\left[\int_0^1 e^{Lt}\mathbf{e}_{2i} \otimes (\mathbf{e}_{2i-1} + \sigma U\xi) e^{-Lt}\, dt\right] - e^L\mathbb{E}\left[\int_0^1 e^{Lt}(\mathbf{e}_{2i-1} + \sigma U\xi) \otimes (\mathbf{e}_{2i-1} + \sigma U\xi) e^{-Lt}\, dt\right]$$

Now bringing the expectation over the Gaussian inside:

$$\begin{aligned} -\sigma^2 e^L = {} & \mathbb{E}\left[\int_0^1 e^{Lt}\mathbf{e}_{2i} \otimes \mathbf{e}_{2i-1} e^{-Lt}\, dt\right] - e^L\mathbb{E}\left[\int_0^1 e^{Lt}\mathbf{e}_{2i-1} \otimes \mathbf{e}_{2i-1}\, e^{-Lt}\, dt\right] \\ & + \sigma\mathbb{E}\left[\int_0^1 e^{Lt}\mathbf{e}_{2i-1} \otimes U\xi\, e^{-Lt}\, dt\right] + \sigma\mathbb{E}\left[\int_0^1 e^{Lt} U\xi \otimes \mathbf{e}_{2i-1}\, e^{-Lt}\, dt\right] \\ & - \sigma^2 e^L\mathbb{E}\left[\int_0^1 e^{Lt} U\xi \otimes U\xi\, e^{-Lt}\, dt\right] \end{aligned}$$

Now the middle two terms vanish, and the last term contains the covariance of an isotropic Gaussian. Moreover, the outer product of standard basis vectors contain a single scalar. Thus, the exponentials cancel out (noting that they trivially commute):

$$-\sigma^2 e^L = \mathbb{E}\left[\mathbf{e}_{2i} \otimes \mathbf{e}_{2i-1}\right] - e^L \mathbb{E}\left[\mathbf{e}_{2i-1} \otimes \mathbf{e}_{2i-1}\right] - \sigma^2 e^L$$

or:

$$0 = A - e^L B$$

where:

$$A = \begin{bmatrix} C & & & \\ & \ddots & & \\ & & C & \\ & & & D \end{bmatrix}, \qquad C = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}$$

and:

$$B = \begin{bmatrix} E & & & \\ & \ddots & & \\ & & E & \\ & & & D \end{bmatrix}, \qquad E = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}$$

now note that:

$$e^L B = \begin{bmatrix} C^\top & & & \\ & \ddots & & \\ & & C^\top & \\ & & & D \end{bmatrix} = A^\top + I^D$$

where I^D is the matrix containing ones along the diagonal matching the D matrix and zero everywhere else. Hence:

$$e^L = A - A^\top + I^D = A - A^\top + I^D$$

Collapsed solution. The collapsed solution maps two initial modes to the same target mode. We consider a n mode model collapsing one mode. The solution can be written as:

$$W = ULU^T$$

with:

$$U = [\frac{1}{\sqrt{2}}(\boldsymbol{\mu}_2 - \boldsymbol{\mu}_1), -\boldsymbol{\mu}_1^*, \frac{1}{\sqrt{2}}(\boldsymbol{\mu}_1 + \boldsymbol{\mu}_2), \mathbf{u}_4, ..., \mathbf{u}_n]$$

and

$$L = \begin{bmatrix} -\lambda & 0 & 0 & \\ 0 & 0 & -1 & \\ 0 & 1 & 0 & \\ & & & \ddots \end{bmatrix}$$

in particular, because of the block structure, the dynamics in all other initial-target mode pairs than modes 1 and 2 will be identical to the optimal solution, and so we focus on these modes.

Just as for the optimal solution, given that U is an orthogonal matrix, the fixed points satisfy:

$$-\sigma^2 e^L = \mathbb{E}\left[\int_0^t e^{Lt}(U^\top \boldsymbol{\mu}_i^* - e^L U^\top \mathbf{x}_0) \otimes \mathbf{x}_0 U e^{-Lt} dt\right]$$

We can split expectation into each possible initial mode. Moreover, by the same argument as before, the integral of exponentials cancel out, so that the fixed point is.

$$\begin{aligned}-\sigma^2 e^L =& -\frac{1}{2}\mathbf{e}_2 \otimes (\mathbf{e}_3 - \mathbf{e}_1) - \frac{1}{2}\mathbf{e}_2 \otimes (\mathbf{e}_3 + \mathbf{e}_1) \\ & - e^L\left(\frac{1}{2}(\mathbf{e}_3 - \mathbf{e}_1) \otimes (\mathbf{e}_3 - \mathbf{e}_1) + \frac{1}{2}(\mathbf{e}_3 + \mathbf{e}_1) \otimes (\mathbf{e}_3 + \mathbf{e}_1) - \sigma^2 I\right)\end{aligned}$$

Which simplifies to:

$$0 = -\mathbf{e}_2 \otimes \mathbf{e}_3 - e^L(\mathbf{e}_1 \otimes \mathbf{e}_1 + \mathbf{e}_3 \otimes \mathbf{e}_3)$$

Now noting that $e^L \mathbf{e}_1 = e^{-\lambda}\mathbf{e}_1$ and $e^L \mathbf{e}_3 = -\mathbf{e}_2$:

$$0 = -\mathbf{e}_2 \otimes \mathbf{e}_3 - e^{-\lambda}\mathbf{e}_1 \otimes \mathbf{e}_1 + \mathbf{e}_2 \otimes \mathbf{e}_3$$

which is satisfied in the limit of large λ.

Saddle/slow compression point Empirically, we observe training go through a regime we call the "compression" phase. This involves the weight structure:

$$W = ULU^\top$$

where $U = [\boldsymbol{\mu}_1^*, \ldots, \boldsymbol{\mu}_m^*, \boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_m, \mathbf{u}_{2m+1}, ..., \mathbf{u}_n]$ and

$$L = \begin{pmatrix} \lambda_+ I_m & & \\ & \lambda_- I_m & \\ & & D \end{pmatrix}, \qquad D = \begin{pmatrix} 0 & & \\ & \ddots & \\ & & 0 \end{pmatrix}$$

where $I_m \in \mathbb{R}^{m\times m}$, $\lambda_+ > 0$ and $\lambda_- < 0$. Since L is diagonal, e^L is also diagonal with entries of $e^{\text{diag}(L)_i}$. Similarly to before, we can write the fixed point equation in the basis:

$$-\sigma^2 I_n = \mathbb{E}\left[\int_0^1 e^{L(1-t)}U^\top(\boldsymbol{\mu}_i^* - U e^L U^\top \mathbf{x}(0)) \otimes \mathbf{x}(0) U e^{Lt} dt\right]$$

Evaluating the modes in the rotated basis we get

$$U^\top \boldsymbol{\mu}_i = \mathbf{e}_{i+m}, \qquad U^\top \boldsymbol{\mu}_j^* = \mathbf{e}_j$$

Also writing $\mathbf{x}(0) = \boldsymbol{\mu}_i + \sigma\xi$ and noting that $U^\top U = UU^\top = I$, we can rewrite the fixed point equation as:

$$-\sigma^2 I_n = \mathbb{E}\left[\int_0^1 e^{L(1-t)}(\mathbf{e}_i - e^L(\mathbf{e}_{i+m} + \sigma\tilde{\xi})) \otimes (\mathbf{e}_{j+m} + \sigma\tilde{\xi}) e^{Lt} dt\right]$$

where $\tilde{\xi} = U^\top \xi \sim \mathcal{N}(0, I)$ since U is orthogonal. Since e^{Lt} is diagonal and acts in each subspace independently, we can rewrite the RHS for each subspace independently:

$$-\sigma^2 I_{ab} = \int_0^1 e^{(1-t)\lambda_a} e^{t\lambda_b} dt\, \mathbb{E}\left[(\mathbf{e}_i - e^L(\mathbf{e}_{i+m} + \sigma\tilde{\xi}))\,|_{V_a} \otimes (\mathbf{e}_{j+m} + \sigma\tilde{\xi})\,|_{V_b}\right]$$

where a is the column subspace with corresponding eigenvalue λ_a, b is the row subspace with eigenvalue λ_b and $\cdot\,|_{V_c}$ indicates projection into the corresponding c subspace. The integral can be evaluated as

$$\int_0^1 e^{(1-t)\lambda_a} e^{t\lambda_b} dt = \begin{cases} \frac{e^{\lambda_a} - e^{\lambda_b}}{\lambda_a - \lambda_b} & \lambda_a \neq \lambda_b \\ e^{\lambda_a} & \lambda_a = \lambda_b \end{cases}$$

Thus there are four blocks which we evaluate separately, $V_+ \times V_+$, $V_- \times V_-$, $V_+ \times V_-$ and $V_- \times V_+$. For the first block we have

$$-\sigma^2 I_d = e^{\lambda_+} \mathbb{E}\left[(\mathbf{e}_i - e^{\lambda_+}\sigma\tilde{\xi}_+) \otimes \sigma\tilde{\xi}_+\right]$$

The second block gives

$$\begin{aligned} -\sigma^2 I_d &= e^{\lambda_-} \mathbb{E}\left[-e^{\lambda_-}(\mathbf{e}_{i+m} + \sigma\tilde{\xi}_-) \otimes (\mathbf{e}_{j+m} + \sigma\tilde{\xi}_-)\right] \\ &= -e^{2\lambda_-}\left[\mathbb{E}\left[\mathbf{e}_{i+m} \otimes \mathbf{e}_{j+m}\right] + \sigma^2 \mathbb{E}\left[\tilde{\xi}_- \otimes \tilde{\xi}_-\right]\right] \\ &= -e^{2\lambda_-}\left[\frac{R^2}{M} + \sigma^2\right] I_- \end{aligned}$$

and so we obtain

$$\lambda_- = -\frac{1}{2}\ln\left(\frac{R^2/M + \sigma^2}{\sigma^2}\right).$$

Since the different subspaces are non-overlapping, the cross blocks are trivially zero.

Appendix C. Effective collapse in flow matching models

The flow matching objective can be written generally as

$$\mathcal{L} = \mathbb{E}_{\mathbf{x}_0, \mathbf{z}}\left[\int_0^1 ||\mathbf{u}(\mathbf{x}_t) - \mathbf{u}_\theta(\mathbf{x}_t)||^2 dt\right]$$

where the expectation is over initial samples $\mathbf{x}_0$ and data samples $\mathbf{z}$, $\mathbf{u}(\mathbf{x})$ is the true field and $\mathbf{x}_t = \gamma(\mathbf{x}_0, \mathbf{z}, t)$ interpolates smoothly between initial samples and data samples for $t \in [0, 1]$.

C.1. Linear flow matching

To compare the FM objective to the linear CNF model, we choose $\mathbf{u}_\theta(\mathbf{x}) = W\mathbf{x}$. For the mixture of Gaussians set up described in section 3.1, we can then derive the optimal weights, W^*, which transport each initial mode to a respective target mode (appendix B.3). Thus, we can define an interpolation path which follows this known optimal trajectory: $\mathbf{x}_t = e^{tW^*}\mathbf{x}_0$, where W^* is constructed such that initial modes are assigned to different targets. The optimal velocity is then $\dot{\mathbf{x}}_t = W^* e^{tW^*}\mathbf{x}_0 = W^*\mathbf{x}_t$, and so we can write the objective as a linear regression integrated over time:

$$\mathcal{L} = \mathbb{E}_{\mathbf{x}_0}\left[\int_0^1 ||W^*\mathbf{x}_t - W\mathbf{x}_t||^2 dt\right]$$

To train the model, we use full-batch gradient descent with a fixed set of initial samples and time points. We initialise weights as $w_{ij} \sim \mathcal{N}(0, \sigma_{\text{init}}/\sqrt{n})$. The hyperparameters are:

n	σ_{init}	lr	Iterations	Batch size	Optimizer
3-10	$1e^{-4}$	$1e^{-3}$	3000	10k	SGD

In figure 2 we parametrise the linear flow as $\mathbf{u}_\theta(\mathbf{x}) = W_2 W_1 \mathbf{x}$. Classic work in learning theory[43] has shown that this parametrisation leads to non-linear training dynamics, inducing saddle-to-saddle dynamics.

C.2. Rank of the data operator

Covariance operator of the flow matching objective. To decompose the gradient of the FM objective into two linear operators, we note that the gradient (up to a scalar constant) is:

$$\begin{aligned}\nabla_W \mathcal{L} &= \mathbb{E}_{\mathbf{x}_0}\left[\int_0^1 \mathbf{x}_t \left(W^* \mathbf{x}_t - W \mathbf{x}_t\right)^\top dt\right] \\ &= \mathbb{E}_{\mathbf{x}_0}\left[\int_0^1 \mathbf{x}_t \mathbf{x}_t^\top (W^* - W) dt\right]\end{aligned}$$

which we can rewrite as the composition of two linear operators:

$$X \circ R \equiv \mathbb{E}_{\mathbf{x}_0}\left[\int_0^1 \mathbf{x}_t \otimes \mathbf{x}_t dt\right] \circ (W^* - W)$$

where X depends only on the data distribution and R is simply the residual between the weights and the optimal weights. We can compute the rank of X analytically. For this, we will compute the eigenvalues of the matrix:

$$C = X \circ \text{ad}(X) = \mathbb{E}_{\mathbf{x}_0, \mathbf{x}_1}\left[\int_0^t \mathbf{x}(t) \otimes \mathbf{x}(t) dt\right] \in R^{n \times n}$$

where $\mathbf{x}(t)$ follows the linear interpolation path. By commutating the integral and expectation, while also noting that since both $\mathbb{E}[\mathbf{x}_0] = \mathbb{E}[\mathbf{x}_1] = 0$, we have that $\mathbb{E}[\mathbf{x}_0(t)] = 0$, we obtain:

$$C = \int_0^t \mathbb{E}_{\mathbf{x}_0, \mathbf{x}_1}\left[\mathbf{x}(t) \otimes \mathbf{x}(t)\right] dt = \int_0^t \text{Cov}(\mathbf{x}(t)) dt$$

The covariance at a particular time point is given by:

$$\text{Cov}(\mathbf{x}(t)) = t^2 \text{Cov}(\mathbf{x}_0) + (1-t)^2 \text{Cov}(\mathbf{x}_1)$$

and so integrating the time terms, we obtain that:

$$C = \frac{1}{3}(\text{Cov}(\mathbf{x}_1) + \text{Cov}(\mathbf{x}_0))$$

substituting the Gaussian Mixture's covariance $\Sigma = \sigma^2 I + \sum_i \pi_i \boldsymbol{\mu}_i \otimes \boldsymbol{\mu}_i - \bar{\boldsymbol{\mu}} \otimes \bar{\boldsymbol{\mu}}$, where $\bar{\boldsymbol{\mu}}$ is the centre of the means, we obtain:

$$C = \frac{1}{3}\left(\sum_{i=1}^m \pi_i \boldsymbol{\mu}_i \otimes \boldsymbol{\mu}_i\right) - \frac{1}{3}\bar{\boldsymbol{\mu}} \otimes \bar{\boldsymbol{\mu}} + \frac{\sigma^2 + \sigma_0^2}{3} I$$

This is the covariance of a spiked matrix, whose eigenvalues are well studied.

Eigenvalues of flow-matching state operator. For small standard deviations relative to the means, this matrix has eigenvalue eigenvector $(\lambda_i, \mathbf{v}_i)$ pairs within and outside the plane of the means. First within:

$$1 \leq i \leq m: \quad \lambda_i = \frac{\sigma_0^2 + \sigma^2 + \rho_i}{3} \quad \mathbf{v}_i = \sqrt{1 - \frac{1}{m}}(\boldsymbol{\mu}_i - \bar{\boldsymbol{\mu}})$$

where ρ_i are the roots of $\sum_{j=1}^{m} \frac{\pi_j}{\pi_j - \rho_i} = 0$. Then outside:

$$m < i \leq n: \quad \lambda_i = \frac{\sigma^2 + \sigma_0^2}{3}, \quad \mathbf{v}_i \in \ker(\{\boldsymbol{\mu}_1, ..., \boldsymbol{\mu}_m\})$$

In particular, the spike eigenvalues are bounded by:

$$\frac{\sigma_0^2 + \sigma^2 + \pi_i}{3} \geq \lambda_i \geq \frac{\sigma_0^2 + \sigma^2 + \pi_{i+1}}{3}$$

where $\pi_i \geq \pi_{i+1}$ are the ordered weights. In the special case of two modes the roots are more explicitly given by:

$$\frac{\pi_1}{\pi_1 - \rho} + \frac{\pi_2}{\pi_2 - \rho} = 0$$

That is:

$$2\pi_1\pi_2 - \rho(\pi_1 + \pi_2) = 0$$

and since the π's sum to one:

$$\rho = 2\pi_1(1 - \pi_1), \qquad \lambda_1 = \frac{\sigma_0^2 + \sigma^2 + 2\pi_1(1 - \pi_1)}{3}$$

In particular the eigenvalue depends quadratically on π_1. A similar derivation for three modes yields:

$$\frac{\pi_1}{\pi_1 - \rho} + \frac{\pi_2}{\pi_2 - \rho} + \frac{\pi_3}{\pi_3 - \rho} = 0$$

that is:

$$\rho^2 - 2\rho(\pi_1\pi_2 + \pi_2\pi_3 + \pi_1\pi_3) + 3\pi_1\pi_2\pi_3 = 0$$

noting that since the π's sum to one this is a quadratic equation. Calling $b = (\pi_1\pi_2 + \pi_2\pi_3 + \pi_1\pi_3)$ and $c = 3\pi_1\pi_2\pi_3$ this yields:

$$\rho = b \pm \sqrt{b^2 - 3c}$$

For a high imbalance $\pi_1 \ll \pi_2 \ll \pi_3$, the $\pi_2\pi_3$ terms will be small, and the eigenvalues approximately equal to:

$$\lambda_1 \approx \frac{\sigma_0^2 + \sigma^2 + 2\pi_2}{3}, \quad \lambda_2 \approx \frac{\sigma_0^2 + \sigma^2}{3} + \pi_3$$

which corresponds to one eigenvector through the means of the two larger weights modes and one eigenvector through the smallest weight mode. In two-layer deep linear networks, the learning time of a particular singular value s grows as $O(s^{-1})$.[43] Thus, in our case, the learning time of the smallest mode grows as $O(\pi_{\min}^{-2})$

Appendix D. Non-linear flow matching model

Nonlinear models, unlike linear flows, can create new modes. We therefore asked whether similar sequential learning dynamics emerge in nonlinear models, and whether this could contribute to a form of effective collapse. As in the linear case, we assume a Gaussian mixture target distribution with equal, isotropic covariances $\sigma_*^2 I_d$ and unequal mixture weights. However here, unlike in the linear settings, the initial distribution is a single-mode Gaussian.

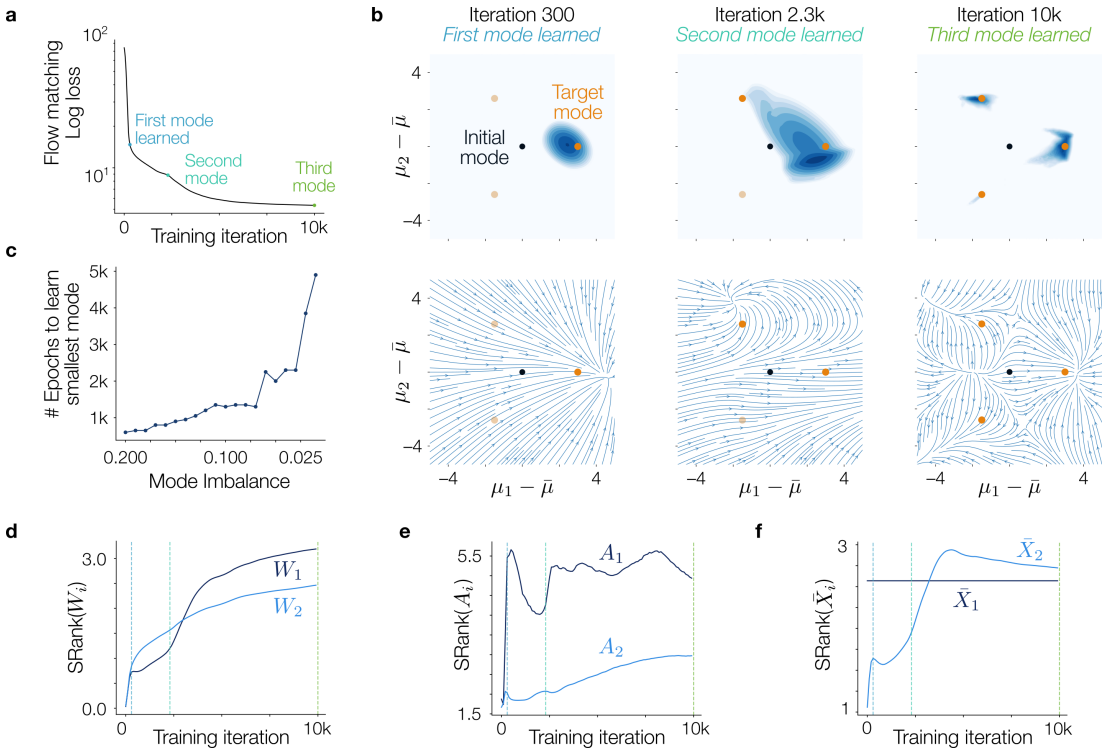

Figure 4: **Nonlinear flow-matching models exhibit effective collapse due to low-rank biases.** **a.** Flow matching loss over training displaying progressively longer learning time for each mode. **b.** (*Top*) Generative log-density at three time points of training evaluated via the continuity equation (App. D.1). The modes are progressively learned (*Bottom*) Corresponding vector field, where additional fixed points are created with each learned mode. **c.** Learning time as a function of the mode imbalance $\pi(1-\pi)$. **d.** Stable rank of the weights over training. The hidden layer weights W_1 show progressive increase in their rank with each learned mode. **e.** Adjoint (residual) operator rank for the two weight matrices. **f.** State operator rank for the two weight matrices.

Nonlinear model operators. We trained a one hidden-layer ReLU network parameterised to match the form of the optimal velocity (App. D.3):

$$\mathbf{u}_\theta(\mathbf{x}, t) = c(t)\mathbf{x} + (1 - tc(t))W_2\text{ReLU}(W_1\mathbf{x} + \mathbf{b}), \qquad c(t) = \frac{t\sigma_*^2 - (1-t)\sigma^2}{t^2\sigma_*^2 + (1-t)^2\sigma^2}$$

where $c(t)$ is a scalar given analytically and $W_2 \in \mathbb{R}^{n\times m}$, $W_1 \in \mathbb{R}^{m\times n}$, $\mathbf{b} \in \mathbb{R}^m$ are trained parameters. By using such a parametrisation, the time-dependency is handled purely by the scalar prefactors, and the model only needs to learn the spatial component of the field. For such a model trained under the MSE loss with conditional velocity field target (App. D.1), we can write the gradients with respect to the two weight matrices as a composition of two operators (App. D.4):

$$\nabla_{W_1}\mathcal{L} = \mathbb{E}_{\mathbf{x}_0,\mathbf{x}_1}\left[\int_0^1 \underbrace{\text{diag}\left(\text{ReLU}'(\mathbf{z}_1)\right) W_2^\top R(\bar{\mathbf{x}}(t))}_{A_1} \otimes \underbrace{\bar{\mathbf{x}}(t)}_{\bar{X}_1} dt\right]$$

$$\nabla_{W_2}\mathcal{L} = \mathbb{R}_{\mathbf{x}_0,\mathbf{x}_1}\left[\int_0^1 \underbrace{R(\bar{\mathbf{x}}(t))}_{A_2} \otimes \underbrace{\mathbf{h}_1(\bar{\mathbf{x}}(t))}_{\bar{X}_2} dt\right]$$

where $\bar{\mathbf{x}}(t)$ is a point along the interpolant trajectory, $R(\bar{\mathbf{x}}(t))$ is the residual error per sample, $\mathbf{z} = W_1\mathbf{x} + \mathbf{b}$ is the network pre-activations, $\mathbf{h}$ is the network post-activations and ReLU$'$ denotes the derivative of the activation. Similarly to the linear flow matching case, these are both of the form $A \circ \bar{X}$, where the adjoint operator A propagates errors backwards and the state operator $\bar{X}$ represents the state of the network. Also like before, the W_1 gradient has an operator which is dependent purely on the data distribution.

Numerical results. Similar to the linear model, we observed sequential learning dynamics corresponding to each mode being learned one-by-one (Fig. 4a). The first inflection point in the loss corresponded to the probability density being transported to the dominant mode. After a few thousand training iterations a second inflection point occurred when the density expanded to cover the second strongest mode. By the point the loss saturated, the transported density covered all modes, with attracting dynamics surrounding each mode in the learned vector field (Fig. 4b). The ranks of the weights correlated with the learning of each modes, with an increase in W_1 at each time a mode was learned (Fig. 4d). The first layer adjoint rank and second layer state rank also reflect mode learning, with sharp changes at each inflection point (Fig. 4e-f). By systematically varying the imbalance between the strongest and weakest mode weightings, we observed the time to learn increasing super-linearly with increasing imbalance (Fig. 4c).

D.1. Non-linear flow matching

In section 4, we consider a task of learning a mixture of Gaussians target distribution with isotropic covariances:

$$\mathbf{z} \sim p_{\text{data}}(\mathbf{z}) = \sum_{i=1}^{M} \pi_i \mathcal{N}(\mathbf{z}|\boldsymbol{\mu}_i, \sigma_*^2 I_d) \in \mathbb{R}^n$$

from a unimodal initial distribution

$$\mathbf{x}_0 \sim p_0(\mathbf{x}) = \mathcal{N}(\mathbf{x}|\mathbf{0}, \sigma^2 I_d) \in \mathbb{R}^n.$$

with $\sigma = 1.0$ and $\sigma_* = 0.1$. The mixture weights are distributed according to a power law: $\pi_i = \frac{i^{-\alpha}}{\sum_i i^{-\alpha}}$, $\alpha = 3$. We choose the linear interpolation probability path, $\mathbf{x}_t = t\mathbf{z} + (1-t)\mathbf{x}_0$. Motivated by the form of the analytic field derived in section D.3, we parameterise our model as:

$$\mathbf{u}_\theta(\mathbf{x}, t) = c(t)\mathbf{x} + (1 - tc(t))W_2\text{ReLU}(W_1\mathbf{x} + \mathbf{b})$$

where $c(t)$ is given explicitly to the model and $W_2 \in \mathbb{R}^{n\times m}$, $W_1 \in \mathbb{R}^{m\times n}$, $\mathbf{b} \in \mathbb{R}^m$ are trained parameters. We train using the conditional velocity objective:

$$\mathcal{L} = \mathbb{E}_{\mathbf{x}_0,\mathbf{z}}\left[\int_0^1 ||\mathbf{u}_\theta(\mathbf{x},t) - (\mathbf{z}-\mathbf{x}_0)||^2 dt\right]$$

There is a fundamental theorem in flow matching which says that in loss has the same gradients (up to a constant) to training on the unconditional velocity field[31]. To train the model, we use full-batch gradient descent with a fixed set of initial samples and time points. We initialise weights as $w_{ij} \sim \mathcal{N}(0, \sigma_{\text{init}}/\sqrt{n})$ and $\mathbf{b} = \mathbf{0}$. The hyperparameters are:

n	m	σ_{init}	lr	Iterations	Batch size	Optimizer
5	64	$1e^{-4}$	$5e^{-4}$	10k	1000	Adam

Learned density To compute the log-density of the learned model, $\log p_{\theta,t}(\mathbf{x})$, we use the instantaneous change of variables for probability flow, derived from the continuity equation[12]:

$$\log p_{\theta,t}(\mathbf{x}) = \log p_0(\mathbf{x}) - \int_0^t \nabla \cdot u_\theta(\mathbf{x},t)dt$$

By solving the same augmented ODE as in appendix B.1, we can compute the accumulated log-density at time t at a grid of initial points.

Time to learning vs mode imbalance In figure 4c, we plot the time taken to learn the smallest mode as a function of the imbalance between the weightings of the largest and smallest modes. We used a symmetric 3-mode setup in 2-d, with weightings $\boldsymbol{\pi} = [\alpha, \frac{1-\alpha}{2}, \frac{1-\alpha}{2}]$, and with $\alpha \in [0.01, 0.20]$. We define the time to learn as the first training iteration (except initialisation) where any generated sample is within a threshold distance of the smallest weighted mode.

D.2. Marginal distribution for MoG targets

At time t, the linear interpolation path yields the conditional (on a data sample $\mathbf{z}$) probability distribution

$$p_t(\mathbf{x}|\mathbf{z}) = \mathcal{N}(\mathbf{x}|t\mathbf{z}, (1-t)^2\sigma^2 I_d).$$

Marginalising over the data distribution:

$$\begin{aligned}
p_t(\mathbf{x}) &= \int p_t(\mathbf{x}|\mathbf{z}) p_{\text{data}}(\mathbf{z}) d\mathbf{z} \\
&= \int \left(\mathcal{N}(\mathbf{x}|t\mathbf{z}, (1-t)^2\sigma^2 I_d)\right)\left(\sum_{i=1} \pi_i \mathcal{N}(\mathbf{z}|\mu_i, \sigma_*^2 I_d)\right) d\mathbf{z} \\
&= \sum_{i=1}^M \pi_i \int \mathcal{N}(\mathbf{x}|t\mathbf{z}, (1-t)^2\sigma^2 I_d)\mathcal{N}(\mathbf{z}|\mu_i, \sigma_*^2 I_d) d\mathbf{z} \\
&= \sum_{i=1}^M \pi_i p_t^{(i)}(\mathbf{x})
\end{aligned}$$

where $p_t^{(i)}(\mathbf{x}) = \mathcal{N}(\mathbf{x}|t\mu_i, \Sigma_t)$ is the probability density conditioned on target mode i and $\Sigma_t = (t^2\sigma_*^2 + (1-t)^2\sigma^2)I_d$.

D.3. Marginal velocity field for MoG targets

We start by writing the conditional velocity field for the linear interpolation path:

$$\mathbf{u}_t(\mathbf{x}|\mathbf{z}) = \dot{\mathbf{x}}_t = \mathbf{z} - \mathbf{x}_0$$

Rearranging the probability path equation we get

$$\mathbf{x}_0 = \frac{\mathbf{x}_t - t\mathbf{z}}{1-t}$$

which we can substitute into the conditional velocity to remove dependency on $\mathbf{x}_0$:

$$\mathbf{u}_t(\mathbf{x}|\mathbf{z}) = \frac{1}{1-t}(\mathbf{z} - \mathbf{x})$$

To obtain the marginal field (for the i^{th} target mode) we marginalise over $\mathbf{z}$ by taking the expectation with respect to the posterior distribution $p_t(\mathbf{z}|\mathbf{x})$:

$$\begin{aligned} u_t^{(i)}(\mathbf{x}) &= \mathbb{E}_{\mathbf{z}\sim p_t(\mathbf{z}|\mathbf{x})}[u_t(\mathbf{x}|\mathbf{z})] \\ &= \frac{1}{1-t}(\mathbb{E}_{\mathbf{z}\sim p_t(\mathbf{z}|\mathbf{x})}[\mathbf{z}] - \mathbf{x}) \end{aligned}$$

To evaluate the expectation, we use the following result:

Theorem 6 *Let* $\mathbf{u} \sim \mathcal{N}(\mathbf{a}_\mathbf{u}, B_\mathbf{u})$ *and* $\mathbf{v} \sim \mathcal{N}(\mathbf{a}_\mathbf{v}, B_\mathbf{v})$.

$$p(\mathbf{u}|\mathbf{v}+\mathbf{u}) = \mathcal{N}\left(\mathbf{a}_\mathbf{u} + B_\mathbf{u}(B_\mathbf{u}+B_\mathbf{v})^{-1}\left[(\mathbf{u}+\mathbf{v}) - (\mathbf{a}_\mathbf{u}+\mathbf{a}_\mathbf{v})\right], \quad B_\mathbf{u} - B_\mathbf{u}(B_\mathbf{u}+B_\mathbf{v})^{-1}B_\mathbf{u}\right)$$

I.e. the conditional distribution of the sum of two Gaussians is itself Gaussian.

Applying this, we obtain $\mathbb{E}[\mathbf{z}|\mathbf{x}] = \boldsymbol{\mu}_i + t\sigma_*^2 I_d \Sigma_t^{-1}(\mathbf{x} - t\boldsymbol{\mu}_i)$ where $\Sigma_t = t^2\sigma_*^2 + (1-t)^2\sigma^2$. Thus the field becomes:

$$\begin{aligned} \mathbf{u}_t^{(i)}(\mathbf{x}) &= \frac{1}{1-t}\left(\boldsymbol{\mu}_i + t\sigma_*^2 I_d \Sigma_t^{-1}(\mathbf{x} - t\boldsymbol{\mu}_i)\right) - \frac{1}{1-t}\mathbf{x} \\ &\overset{(1)}{=} \frac{1}{1-t}\boldsymbol{\mu}_i - \frac{1}{1-t}(\mathbf{x} - t\boldsymbol{\mu}_i) - \frac{t}{1-t}\boldsymbol{\mu}_i + \frac{t}{1-t}\sigma_*^2 I \Sigma_t^{-1}(\mathbf{x} - t\boldsymbol{\mu}) \\ &\overset{(2)}{=} \boldsymbol{\mu}_i + \left[\frac{t}{1-t}\sigma_*^2 I - \frac{1}{1-t}\Sigma_t\right]\Sigma_t^{-1}(\mathbf{x} - t\boldsymbol{\mu}_i) \\ &\overset{(3)}{=} \boldsymbol{\mu}_i + \frac{t\sigma_*^2 - (1-t)\sigma^2}{t^2\sigma_*^2 + (1-t)^2\sigma^2}(\mathbf{x} - t\boldsymbol{\mu}_i) \end{aligned}$$

where in (1) we have added and subtracted by $\frac{t}{1-t}\boldsymbol{\mu}_i$, in (2) we took out a factor of $\Sigma_t^{-1}(\mathbf{x} - t\boldsymbol{\mu}_i)$ on the right, and in (3) we have substituted in the expression for Σ_t and simplified. Defining $c(t) = \frac{t\sigma_*^2 - (1-t)\sigma^2}{t^2\sigma_*^2 + (1-t)^2\sigma^2}$ we can write the field as

$$\mathbf{u}_t^{(i)}(\mathbf{x}) = c(t)\mathbf{x} + (1 - tc(t))\boldsymbol{\mu}_i$$

Note that this is still conditional on a specific target mode $\boldsymbol{\mu}_i$. To get the full unconditional field, we use the continuity equation:

Theorem 7 (Continuity Equation) *For a flow model with vector field u_t and $\mathbf{x}_0 \sim p_{init}$, then $\mathbf{x}_t \sim p_t$ for all $0 \leq t \leq 1$ if and only if*

$$\partial_t p_t(\mathbf{x}) = -\nabla(p_t u_t)(\mathbf{x})$$

for all $\mathbf{x} \in \mathbb{R}^d$, $0 \leq t \leq 1$.

Then, using the form of the marginal probability path in section D.2, we can write the LHS as

$$\begin{aligned}
\partial_t p_t(\mathbf{x}) &= \partial_t \left(\sum_{i=1}^{M} \pi_i p_t^{(i)}(\mathbf{x}) \right) \\
&\overset{(1)}{=} \sum_{i=1}^{M} \pi_i \partial_t p_t^{(i)}(\mathbf{x}) \\
&\overset{(2)}{=} -\sum_{i=1}^{M} \pi_i \nabla(p_t^{(i)} u_t^{(i)})(\mathbf{x}) \\
&\overset{(3)}{=} -\nabla \left(\sum_{i=1}^{M} \pi_i p_t^{(i)}(\mathbf{x}) u_t^{(i)}(\mathbf{x}) \right)
\end{aligned}$$

where in (1) we used the linearity of derivatives, in (2) we used the continuity equation for the i^{th} target mode and in (3) we use the linearity of the divergence operator. Comparing this to the RHS of the continuity equation, we get that

$$p_t(\mathbf{x}) u_t(\mathbf{x}) = \sum_{i=1}^{M} \pi_i p_t^{(i)}(\mathbf{x}) u_t^{(i)}(\mathbf{x})$$

which we can rearrange to get

$$u_t(\mathbf{x}) = \sum_{i=1}^{M} \gamma_t^{(i)}(\mathbf{x}) u_t^{(i)}(\mathbf{x})$$

where $\gamma_t^{(i)}(\mathbf{x}) = \frac{\pi_i p_t^{(i)}(\mathbf{x})}{p_t(\mathbf{x})}$ acts as a weighting between different modes. Substituting the expression for the field conditioned on target mode i, we obtain the final form of the marginal field:

$$\mathbf{u}_t(\mathbf{x}) = c(t)\mathbf{x} + (1 - tc(t)) \sum_{i=1}^{M} \gamma_t^{(i)}(\mathbf{x}) \boldsymbol{\mu}_i$$

where we used the fact that $\sum_i \gamma_t^{(i)}(\mathbf{x}) = 1$.

D.4. Operators of the non-linear model

Consider the loss functional:

$$\mathcal{L} = \mathbb{E}_{\mathbf{x}_0, \mathbf{x}_1} \left[\int_0^1 \ell(\bar{\mathbf{x}}(t)) dt \right]$$

where $\ell(\bar{\mathbf{x}}(t)) = ||\Phi(\bar{\mathbf{x}}(t)) - \mathbf{u}(\bar{\mathbf{x}}(t))||^2$ is the MSE objective for model Φ and target velocity $\mathbf{u}$, and $\bar{\mathbf{x}}(t)$ is the interpolated trajectory. Also consider a deep neural network architecture:

$$\Phi = W_k \circ \phi \circ W_{k-1} \circ \phi \circ \cdots \circ W_1$$

which has i^{th} layer pre-activations $\mathbf{z}_i = W_i \circ \phi \dots$ and post-activations $\mathbf{h}_i = \phi(\mathbf{z}_i)$ for some non-linearity ϕ. The gradients with respect to the i^{th} layer weights are:

$$\begin{aligned}\nabla_{W_i}\mathcal{L} &= \mathbb{E}_{\mathbf{x}_0,\mathbf{x}_1}\left[\int_0^1 \partial_{\mathbf{z}_i}\ell(\bar{\mathbf{x}}(t)) \otimes \mathbf{h}_{i-1}(\bar{\mathbf{x}}(t))dt\right] \\ &= \mathbb{E}_{\mathbf{x}_0,\mathbf{x}_1}\left[\int_0^1 R(\bar{\mathbf{x}}(t))\frac{\partial\Phi}{\partial \mathbf{z}i} \otimes \mathbf{h}_{i-1}(\bar{\mathbf{x}}(t))dt\right]\end{aligned}$$

where $R(\bar{\mathbf{x}}(t)) = \Phi(\bar{\mathbf{x}}(t)) - \mathbf{u}(\bar{\mathbf{x}}(t))$ is the residual per sample.

Now we define our network as in section **??** (omitting the skip connection which contributes zero to the gradient, and the time-dependent pre-factor which just scales the gradient):

$$\Phi(\mathbf{x}) = W_2\text{ReLU}(W_1\mathbf{x} + \mathbf{b})$$

We can then compute the gradients as:

$$\nabla_{W_1}\mathcal{L} = \mathbb{E}_{\mathbf{x}_0,\mathbf{x}_1}\left[\int_0^1 \underbrace{\text{diag}\left(\text{ReLU}'(\mathbf{z}_1)\right) W_2^\top R(\bar{\mathbf{x}}(t))}_{A_1} \otimes \underbrace{\bar{\mathbf{x}}(t)}_{\bar{X}_1} dt\right]$$

$$\nabla_{W_2}\mathcal{L} = \mathbb{R}_{\mathbf{x}_0,\mathbf{x}_1}\left[\int_0^1 \underbrace{R(\bar{\mathbf{x}}(t))}_{A_2} \otimes \underbrace{\mathbf{h}_1(\bar{\mathbf{x}}(t))}_{\bar{X}_2} dt\right]$$

where we define A_1, A_2 as the adjoint (backward) operators and $\bar{X}_1$, $\bar{X}_2$ as the state (forward) operators for layer 1 and layer 2 respectively.

[heading=subbibliography,title=Supplementary References]