PREDICTING GROKKING LONG BEFORE IT HAPPENS: A LOOK INTO THE LOSS LANDSCAPE OF MODELS WHICH GROK

Pascal Jr. Tikeng Notsawo* **& Hattie Zhou**
Université de Montréal, Montréal, Quebec, Canada
Mila, Montréal, Quebec, Canada

Mohammad Pezeshki
Meta AI Research

Irina Rish
Université de Montréal, Quebec, Canada
Mila, Montréal, Quebec, Canada

Guillaume Dumas
Université de Montréal, Quebec, Canada
CHU Sainte Justine Research Center, Quebec, Canada.

ABSTRACT

This paper presents a cost-effective method for predicting grokking in neural networks—delayed perfect generalization following overfitting or memorization. By analyzing the learning curve of the first few epochs, we show that certain oscillations forecast grokking in extended training. Our approach, using the Fourier transform's *spectral signature*, efficiently detects these oscillations. Additional experiments explore their origins and characterize the loss landscape.

1 INTRODUCTION

Despite the recent growth of theoretical studies and empirical successes of neural networks (Graves et al., 2013; He et al., 2015; Krizhevsky et al., 2012; Silver et al., 2016), understanding why such networks find generalizable solutions in over-parameterized regimes, remains an open question. Indeed, one of the major challenges of deep learning is that, in practice, neural networks are highly overparameterized (Allen-Zhu et al., 2019; Zhang et al., 2017), and in some cases, it is observed that overparameterization empirically improves optimization and generalization, which seems to contradict traditional learning theory. For example, Livni et al. (2014) observed that in synthetic data generated from a target network, the learned network converges faster when it has more parameters than the target network. Arora et al. (2018) also found that, in practice, trained overparameterized networks can often be compressed into simpler networks with far fewer parameters without affecting their generalizability. Nakkiran et al. (2020) experiment a double-descent phenomenon where performance first gets worse as the model size increases and then gets better. Recently, Power et al. (2022) have shown through a phenomenon they named *grokking* that long after severe overfitting, validation accuracy sometimes suddenly begins to increase from chance level to perfect generalization.

The grokking phenomenon opens the way to new studies concerning the structure of the minimum found by Stochastic Gradient Descent (SGD) and how networks behave in the neighborhood of SGD training convergence. Indeed, neural activity is often characterized by an exploratory early phase of rapid learning with a rapid and sometimes abrupt decrease in the loss function (Nakkiran et al., 2020; Feng & Tu, 2021). This phase is followed by a second phase, often called the diffusion phase (Shwartz-Ziv & Tishby, 2017), when the learning error reaches its minimum value and the global loss decreases again, but much more slowly and gradually. This second phase, often chaotic (Herrmann et al., 2022; Cohen et al., 2021; Thilak et al., 2022), is characteristic of the structure of the minimum found by the optimization algorithm (Goodfellow & Vinyals, 2015; Im et al., 2016; Smith & Topin, 2017; Keskar et al., 2017; Li et al., 2018; Jastrzebski et al., 2018; Feng & Tu, 2021). Optimization hyper-parameters and initialization strongly affect training dynamics during the above phases, the convergence to a specific region in parameter space, and the geometries and generalization properties of solutions found by SGD (Keskar et al., 2017; Jastrzebski et al., 2018). Recent work has shown that grokking is observed only with a certain range of hyperparameters (Power et al., 2022; Liu et al.,

*Correspondence to: Pascal Tikeng Notsawo (pascal.tikeng@mila.quebec)

2022). Others have studied microscopic phenomena that coincide or come in tandem with delayed generalization, such as the emergence of structure in embedding space (Liu et al., 2022) and the slingshot effect (Thilak et al., 2022). However, observing grokking often requires training the model for a very long time, making it difficult to construct a phase diagram of generalization covering all the hyperparameters. In this work, we propose a low-cost method that can predict grokking long before it occurs. Our main contributions are: 1) We study the learning curves of a transformer network (Vaswani et al., 2017) trained on arithmetic data in settings with and without grokking (section 2). 2) We propose *spectral signature* to quantify the oscillations of the loss in the early phases of training and show that empirically it can be used to predict a potential generalization (section 3). 3) To understand the origin of the oscillations and the shape of the minimizer, we analyze the model's loss landscape along the training trajectory and present evidence linking the shape of this landscape to the different phenomena related to grokking such as the slingshot mechanism (Thilak et al., 2022) (section 4).

2 PRELIMINARIES

Task definition and training scheme Let $\circ$ be a binary mathematical operator and $p, q \in \mathbb{N}^*$. The task is to predict $(a \circ b) \ mod \ q$ given $a, b \in [p] = \{0, \dots, p-1\}$. The dataset $\mathcal{D}$ that we can thus constitute has a size of $p(p+1)/2$ if $\circ$ is symmetric and p^2 otherwise. $\mathcal{D}$ is randomly partitioned into two disjoint and non-empty sets $\mathcal{D}_{train}$ and $\mathcal{D}_{val}$, the training and the validation dataset respectively. The training data fraction $r = |\mathcal{D}_{train}|/|\mathcal{D}|$ is a hyperparameter. This problem can be solved with an auto-regressive approach. For $a, b \in [p]$, let $s = s_1 \dots s_5 = \langle a \rangle \langle \circ \rangle \langle b \rangle \langle = \rangle \langle (a \circ b) \ mod \ q \rangle$ where $s_i = \langle x \rangle$ stands for the token corresponding to the element x. The training is performed by maximizing the likelihood under the direct autoregressive factorization, and the loss (as well as the accuracy) is calculated only on the answer part s_5 of the equation. We focus on the multiplication in the permutation group S_5 and modular addition, and we use a transformer (Vaswani et al., 2017) as a model (more details in the appendix D).

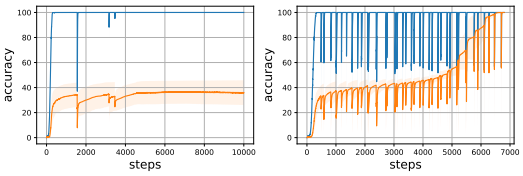


Figure 1: Oscillation in training and validation accuracies for on modular addition with $r = 0.5$. The left curve shows a case where the model did not grok after 10k steps of training. The right curve shows the generalization after overfitting.

Grokking and Non-grokking Let t_1 denote the step when training accuracy first exceeds 0%, t_2 when training accuracy reaches 100% (memorization), t_3 when validation accuracy first exceeds 0%, and t_4 when validation accuracy reaches 100% (generalization). The delay between memorization and generalization, $t_4 - t_2$, characterizes grokking (figure 1). On the other hand, formally defining non-grokking is challenging within finite training steps. We proceeded empirically by training the models for a large range of hyperparameters, and we fit a function that predicts t_4 for each training data fraction r. That is, if a given value for each hyper-parameters (including r), we train a model for $t_4(r) + \epsilon$ (we used $\epsilon = 1k$) steps and have no generalization, we can stop the training. In general, more data leads to faster grokking.Empirically, t_4 follows a power law of the form $t_4(r) = ar^{-\gamma} + b$ (more details in the appendix E). This was first predicted by Žunkovič & Ilievski (2022).

3 PREDICTING GROKKING

A starting observation is that the learning curves of models that grok exhibit oscillatory behaviors (Figure 1). A related phenomenon, the slingshot effect, was identified recently by Thilak et al. (2022), who observed that slingshots and grokking tend to come in tandem. Based on this observation, we conjecture that *the spectral signature of training loss in early epochs can hint us about the existence of an upcoming grokking*. We first try to quantify the oscillations in the training loss comparatively when the model groks and when it does not grok, the idea being to stop the training if the model does not seem to be able to grok in order to save computational resources. Let $\mathcal{F}(L)$ denote the Fourier transform of $L(t)$, the training loss as a function of training step t, and $m_n(L) = \int \omega^n |\mathcal{F}(L)(\omega)|^2 d\omega$,

the n^{th} moment of $\mathcal{F}^2(L)$, with $|\mathcal{F}(L)(\omega)|^2$ the energy spectral density present in the pulse ω. The Hjorth activity represents the signal power, the surface of the power spectrum in the frequency domain. It is given by $m_0(L)$, which is equal to $\int L^2(t)dt$ by the parseval's theorem.

Figure 2 shows a similarity between the oscillation in the training loss in the early phases of training and the validation accuracy for $r = 0.5$ (more results in the appendix, E), suggesting that the spectral signature can serve as a proxy to upcoming grokking phenomenon. The generalization is most observed for small learning rates and small weight decay. Although large learning rates have the effect of increasing the oscillations, this does not directly result in grokking and is not necessarily visible in the early steps but more near the basin of attraction of the minimum. Importantly, the spectral signature of the loss is not an explicit capacity measure, so either a positive or negative correlation with generalization could be informative. Our observations are related to the empirical findings of Jiang et al. (2019). They investigate more than 40 complexity measures derived from both theoretical and empirical studies and train a variety of models by systematically varying commonly used hyperparameters. Their results suggest that the difficulty of optimization during the initial phase of the optimization benefits the final generalization, but the evolution of the loss when it reaches a certain value is not correlated to the generalization of the final solution.

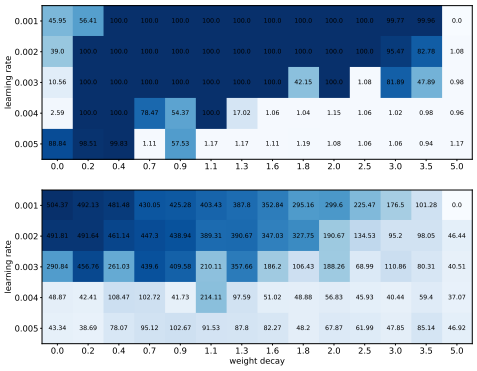


Figure 2: Final validation accuracy (top) and spectral energy $m_0(L)$ for the first 400 training steps (bottom), all this as a function of the weight decay strength (x-axis) and the learning rate (y-axis). The similarity between $m_0(L)$ and the validation accuracy suggests that $m_0(L)$ can be used as an indicator for the upcoming grokking.

4 GROKKING LOSS LANDSCAPE

What can be a possible explanation for the oscillations observed in the early phases and the slingshot phenomenon? Many hypotheses can, in fact, be put forward to reconcile oscillations with delayed generalization. *Does the model, during the confusion, memorization, and comprehension phases, oscillate around a local minimum, cross a very flat region, or circumvent a large obstacle?* One initial intuitive explanation for delayed generalization could be that the model gets stuck in local solutions during the memorization phase. The ease of escaping these local solutions depends on factors such as initialization or hyperparameters like the amount of training data, and the model achieves grokking when it successfully breaks free from the basin of attraction of such solutions. This explanation is similar to the working hypothesis of Dziugaite & Roy (2017) that SGD finds good solutions only if they are surrounded by a relatively large volume of solutions that are nearly as good. A second attempt to explain grokking landscape is that the model crosses an ill-conditioned surface, potentially a valley with almost no curvature in the majority of directions and very high curvature in some directions. This results in a weak progression in the directions of low curvature and a lot of back and forth in the directions of high curvature. A third attempt is to consider the first two hypotheses together. That is, the model goes through several ill-conditioned local minima during training. In the following paragraphs, we present the loss landscape of grokking and discuss how they are linked to our hypothesis.

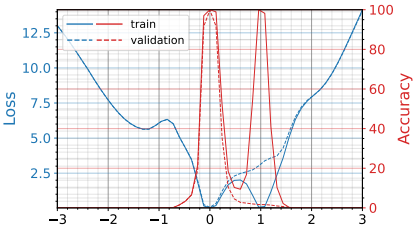


Figure 3: 1D projection of the landscape. The x-axis is $\alpha \in [-3, 3]$, and the y-axis are the loss and accuracy at $\theta^* + \alpha\delta$ with $\delta \propto \theta_0 - \theta^*$, where θ_0 is the initial parameter and θ^* the parameter after the model has grokked. We have two minimizers of the training loss but only one minimizes the validation loss.

For $r = 0.3$, figure 3 shows the 1D projection of the grokking loss surface just after the grokking step, while figures 4 shows it for different training epochs (more experiments in the Appendix G). We can see that the 1-D subspace from initial to final parameters and from one minimizer to another contains many difficult and exotic structures. We can clearly see two minimizers of the training loss, but only one minimizes the validation loss: during memorization, the model is in this local minimum, and it achieves grokking when it successfully breaks free from this local solution. When r increases, the time t_4 it takes for the model to reach the global minimum decreases as $\Theta\left(1/r^{\gamma}\right)$ with $\gamma > 0$ (section E)

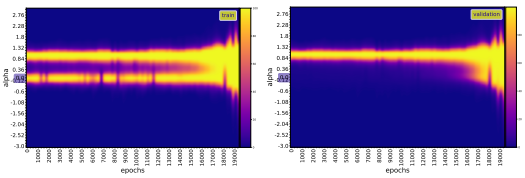


Figure 4: Accuracy surface $f_t(\alpha) = Accuracy(\theta_t + \alpha\vec{\delta}_t)$ for each epoch t. This corresponds to figure 3, but for several training epochs. The direction $\vec{\delta}_t$ used for each training epoch t is the unit vector of $\theta^* - \theta_t$, the direction from the parameter at epoch t to the minimum. Here, the structure is more exotic.

To measure the level of curvature of the loss function, we compute the maximum (λ_{max}) and minimum (λ_{min}) eigenvalue of its Hessian. We observe that there is no significant negative curvature in the trajectory. The curvature remains generally positive and is greatly disturbed at the slingshot points (λ_{min} remains in general close to 0 while λ_{max} is large, but during slingshot λ_{min} becomes negative). Our loss is λ_{max}-smooth, i.e. its gradient is λ_{max}-Lipschitz. It is known from optimization literature that a function with bounded Hessian eigenvalues has a gradient that tends to decay when the parameter gets closer to the minimum, in contrast to a non-smooth one that generally has abrupt bends at the minimum, which causes significant oscillations for gradient descent (Bubeck et al., 2015).

In fact, Let $\{\lambda_t(i)\}_i$ be the eigenvalues of $\mathcal{H}_t$ and $\{v_t(i)\}_i$ the associated eigenvectors. For a very small step size ϵ_t of SGD, $L_{t+1} - L_t \approx -\epsilon_t\|G_t\|^2 + \frac{1}{2}\epsilon_t^2 G_t^T \mathcal{H}_t G_t - o(\epsilon_t^2\|G_t\|^2)$. Futher, if $\lambda_t(i) > 2/\epsilon_t$, we get $-\epsilon_t\|G_t\|^2 + \frac{1}{2}\epsilon_t^2 G_t^T \mathcal{H}_t G_t > 0$ [1]. When $\mathcal{H}_t$ has some large positive eigenvalues (i.e., high-curvature directions) and some eigenvalues close to 0 (i.e., low-curvature directions), gradient descent bounces back and forth in high-curvature directions and makes slow progress in low-curvature directions. In this case, the optimization problem has an ill-conditioned curvature. Furthermore, if the loss function near θ_t has a high condition number, that is, very small steps cause an increase in the cost function (for example, if θ_t is a very sharp minimum surrounded by high loss regions), the optimization problem becomes also ill-conditioned. During training, if the gradient norm does not shrink but $G_t^T \mathcal{H}_t G_t$ increases in order of magnitude, learning can become very slow despite a strong gradient. The above observation ($\lambda_t(i) > 2\epsilon_t^{-1}$) is similar to what Herrmann et al. (2022) [Theorem 2.1] defines for the eigenvalues of a positive-definite $\mathcal{H}_t$, as a locally chaotic training behavior of the Local Lyapunov Exponents. They show evidence that neural network training is intrinsically locally chaotic due to the negative eigenspectrum of the Hessian and that network training with SGD exhibits globally edge-chaotic behavior. This observation is also linked to the *progressive sharpening* phenomenon (Cohen et al., 2021) in which $max_i \lambda_t(i)$ increases and reaches a value that is equal to or slightly larger than $2\epsilon_t^{-1}$, leading the model to enter an Edge of Stability regime where loss shows non-monotonic training behavior over short time spans (Thilak et al., 2022).

We also observed that more than 98% of the total variance in the parameter space occurs in the first 2 PCA modes, much smaller than the total number of weights, suggesting that the optimization dynamics are embedded in a low-dimensional space (Li et al., 2018; Feng & Tu, 2021). Moreover, the model remains in a lazy training regime (Chizat et al., 2019; Berner et al., 2021) most of the time, as the measure of cosine distance between the model weights from one training step to the next remains almost constant, except at the slingshot location (more details on these observations in the Appendix G.2).

These observations seem to support the hypothesis that the model crosses a perturbed valley of bad solutions and experiences disturbances along the direction of the valley. The valley here is just a surface with a large curvature in most directions and no (low) curvature in the rest. Hence, the learning activity of SGD is governed by these directions of high curvature. When the iterates fall in

[1] This comes come from the fact that near θ_t, $L(\theta) \approx L_t + (\theta - \theta_t)^T G_t + \frac{1}{2}(\theta - \theta_t)^T \mathcal{H}_t (\theta - \theta_t) + o(\|\theta - \theta_t\|^2)$. Taking $\theta = \theta_t - \epsilon_t G_t$ gives the first approximation. The last inequality is obtained with $G_t^T \mathcal{H}_t G_t = \sum_i \lambda_t(i) \langle G_t, v_t(i)\rangle^2$ and $\sum_i \langle G_t, v_t(i)\rangle^2 = \|G_t\|^2$.

the valley, we are at the minimum for the training objective (figure 4), so that the model can memorize the training data. This local minimum in the valley directions is flat, but in the high curvature directions, it is sharp. The pathological curvature is no longer a big problem with accelerated or adaptative gradient methods, but the local minimum and saddle points are. This interpretation of grokking was the working hypothesis of Dziugaite & Roy (2017) that SGD finds good solutions only if they are surrounded by a relatively large volume of solutions that are nearly as good, as we saw above with grokking optimum that is surrounded by many local minima along the principal directions of curvature. Moreover, the loss function here does not seem to satisfy the strict saddle property ($\lambda_{min} < 0$), which guarantees that gradient descent does not converge to saddle points on continuously differentiable functions when using a step size smaller than $1/\lambda_{max}$ (Lee et al., 2016). The dynamic, therefore, has components in the unstable subspace of the Hessian (a sub-space spanned by the dimensions of non-negative eigenvalues), and if θ_0 is chosen out of this space (which happens with zero probability for a random initialization), there is hope that we will quickly converge to a global solution. The spectral signature reflects the dynamics in and out of this unstable sub-space. The strict saddle property is indeed verified only when there are strong oscillations.

5 Summary and Discussion

In this work, we observe that in the context of grokking, the memorization phase is characterized by a perturbed landscape, and it is separated from comprehension by a perturbed valley of bad solutions. Small data results in the slow progression of SGD in this region, causing a delay in generalization. During the comprehension phase, the loss and accuracy of training and validation show a periodic perturbation. We found that these perturbation points are characterized at the level of loss (resp. accuracy) by a sudden increase-decrease (resp. decrease-increase), at the level of the model weights by a sudden variation of the relative cosine similarity, and at the level of the loss landscape by obstacles. This last point goes against what Goodfellow & Vinyals (2015) observed, namely that a variety of state-of-the-art neural networks never encounter any significant obstacles from initialization to solution. The slingshot mechanism also contradicts the idea that SGD spends most of its time exploring the flat region at the bottom of the valley surrounding a flat minimizer (Goodfellow & Vinyals, 2015) since it goes with the model from confusion to the terminal phase of training, even after the model generalized. The Hessian of the grokking loss function is characterized by larger condition numbers, leading to a slower convergence of gradient descent. Also, more than 98% of the total variance in the parameter space occurs in the first 2 PCA modes, much smaller than the total number of weights, suggesting that the optimization dynamics are embedded in a low-dimensional space (Li et al., 2018; Feng & Tu, 2021). Moreover, the model remains in a lazy training regime (Chizat et al., 2019; Berner et al., 2021) most of the time.

From the point of view of the landscape, grokking seems a bit clearer: landscape geometry affects generalization and can allow in the early stages of training to know if the model will generalize or not by just looking at a microscopic quantity characteristic of that landscape, like the empirical risk.

References

Zeyuan Allen-Zhu, Yuanzhi Li, and Yingyu Liang. Learning and generalization in overparameterized neural networks, going beyond two layers. *ArXiv*, abs/1811.04918, 2019.

Sanjeev Arora, Rong Ge, Behnam Neyshabur, and Yi Zhang. Stronger generalization bounds for deep nets via a compression approach. *CoRR*, abs/1802.05296, 2018. URL `http://arxiv.org/abs/1802.05296`.

Mikhail Belkin and Partha Niyogi. Laplacian eigenmaps and spectral techniques for embedding and clustering. In T. Dietterich, S. Becker, and Z. Ghahramani (eds.), *Advances in Neural Information Processing Systems*, volume 14. MIT Press, 2001. URL `https://proceedings.neurips.cc/paper/2001/file/f106b7f99d2cb30c3db1c3cc0fde9ccb-Paper.pdf`.

Julius Berner, Philipp Grohs, Gitta Kutyniok, and Philipp Petersen. The modern mathematics of deep learning. *arXiv preprint arXiv: Arxiv-2105.04026*, 2021.

Sébastien Bubeck et al. Convex optimization: Algorithms and complexity. *Foundations and Trends® in Machine Learning*, 8(3-4):231–357, 2015.

Lin Chen, Yifei Min, Mikhail Belkin, and Amin Karbasi. Multiple descent: Design your own generalization curve. *Advances in Neural Information Processing Systems*, 34:8898–8912, 2021.

Lenaic Chizat, Edouard Oyallon, and Francis Bach. On lazy training in differentiable programming. *Advances in Neural Information Processing Systems*, 32, 2019.

Jeremy M. Cohen, Simran Kaur, Yuanzhi Li, J. Z. Kolter, and Ameet S. Talwalkar. Gradient descent on neural networks typically occurs at the edge of stability. *International Conference On Learning Representations*, 2021.

MacKay David J.C. and Ghahramani Zoubin. Comments on 'maximum likelihood estimation of intrinsic dimension' by elizaveta levina and peter bickel (2004). 2005. URL http://www.inference.org.uk/mackay/dimension/.

Laurent Dinh, Razvan Pascanu, Samy Bengio, and Yoshua Bengio. Sharp minima can generalize for deep nets. In Doina Precup and Yee Whye Teh (eds.), *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pp. 1019–1028. PMLR, 2017. URL http://proceedings.mlr.press/v70/dinh17b.html.

David L. Donoho and Carrie Grimes. Hessian eigenmaps: Locally linear embedding techniques for high-dimensional data. *Proceedings of the National Academy of Sciences*, 100(10):5591–5596, 2003. doi: 10.1073/pnas.1031596100. URL https://www.pnas.org/doi/abs/10.1073/pnas.1031596100.

Gintare Karolina Dziugaite and Daniel M. Roy. Computing nonvacuous generalization bounds for deep (stochastic) neural networks with many more parameters than training data. In Gal Elidan, Kristian Kersting, and Alexander T. Ihler (eds.), *Proceedings of the Thirty-Third Conference on Uncertainty in Artificial Intelligence, UAI 2017, Sydney, Australia, August 11-15, 2017*. AUAI Press, 2017. URL http://auai.org/uai2017/proceedings/papers/173.pdf.

Elena Facco, Maria d'Errico, Alex Rodriguez, and Alessandro Laio. Estimating the intrinsic dimension of datasets by a minimal neighborhood information. *Scientific Reports*, 7 (1), sep 2017. doi: 10.1038/s41598-017-11873-y. URL https://doi.org/10.1038%2Fs41598-017-11873-y.

Yu Feng and Yuhai Tu. The inverse variance–flatness relation in stochastic gradient descent is critical for finding flat minima. *Proceedings of the National Academy of Sciences*, 118(9): e2015617118, 2021. doi: 10.1073/pnas.2015617118. URL https://www.pnas.org/doi/abs/10.1073/pnas.2015617118.

Ian J. Goodfellow and Oriol Vinyals. Qualitatively characterizing neural network optimization problems. In Yoshua Bengio and Yann LeCun (eds.), *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL http://arxiv.org/abs/1412.6544.

Alex Graves, Abdel-rahman Mohamed, and Geoffrey E. Hinton. Speech recognition with deep recurrent neural networks. *CoRR*, abs/1303.5778, 2013. URL http://arxiv.org/abs/1303.5778.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015. URL http://arxiv.org/abs/1512.03385.

Luis Herrmann, Maximilian Granz, and Tim Landgraf. Chaotic dynamics are intrinsic to neural network training with SGD. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho (eds.), *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=ffy-h0GKZbK.

G. Hinton. Neural networks for machine learning. coursera, video lectures, 2012.

Bo Hjorth. Eeg analysis based on time domain properties. *Electroencephalography and Clinical Neurophysiology*, 29(3):306–310, 1970. ISSN 0013-4694. doi: https://doi.org/10.1016/0013-4694(70)90143-4. URL https://www.sciencedirect.com/science/article/pii/0013469470901434.

Sepp Hochreiter and Jürgen Schmidhuber. Flat Minima. *Neural Computation*, 9(1):1–42, 01 1997. ISSN 0899-7667. doi: 10.1162/neco.1997.9.1.1. URL https://doi.org/10.1162/neco.1997.9.1.1.

Daniel Jiwoong Im, Michael Tao, and Kristin Branson. An empirical analysis of the optimization of deep network loss surfaces. *arXiv preprint arXiv: Arxiv-1612.04010*, 2016.

Stanislaw Jastrzebski, Zachary Kenton, Devansh Arpit, Nicolas Ballas, Asja Fischer, Yoshua Bengio, and Amos Storkey. Three factors influencing minima in sgd. *ICLR*, 2018.

Yiding Jiang, Behnam Neyshabur, H. Mobahi, Dilip Krishnan, and Samy Bengio. Fantastic generalization measures and where to find them. *International Conference On Learning Representations*, 2019.

Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. URL https://openreview.net/forum?id=H1oyRlYgg.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *International Conference On Learning Representations*, 2014.

Schalk Kok and Carl Sandrock. Locating and characterizing the stationary points of the extended rosenbrock function. *Evol. Comput.*, 17(3):437–453, sep 2009. ISSN 1063-6560. doi: 10.1162/evco.2009.17.3.437. URL https://doi.org/10.1162/evco.2009.17.3.437.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C.J. Burges, L. Bottou, and K.Q. Weinberger (eds.), *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012. URL https://proceedings.neurips.cc/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf.

Jason D. Lee, Max Simchowitz, Michael I. Jordan, and Benjamin Recht. Gradient descent converges to minimizers. *arXiv preprint arXiv: Arxiv-1602.04915*, 2016.

Elizaveta Levina and Peter Bickel. Maximum likelihood estimation of intrinsic dimension. In L. Saul, Y. Weiss, and L. Bottou (eds.), *Advances in Neural Information Processing Systems*, volume 17. MIT Press, 2004. URL https://proceedings.neurips.cc/paper/2004/file/74934548253bcab8490ebd74afed7031-Paper.pdf.

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: Arxiv-2003.02218*, 2020.

Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein. Visualizing the loss landscape of neural nets. In Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett (eds.), *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pp. 6391–6401, 2018. URL https://proceedings.neurips.cc/paper/2018/hash/a41b3bb3e6b050b6c9067c67f663b915-Abstract.html.

Ziming Liu, Ouail Kitouni, Niklas Stefan Nolte, Eric J. Michaud, Max Tegmark, and Mike Williams. Towards understanding grokking: An effective theory of representation learning. *ArXiv*, abs/2205.10343, 2022.

Roi Livni, Shai Shalev-Shwartz, and Ohad Shamir. On the computational efficiency of training neural networks. *CoRR*, abs/1410.1141, 2014. URL http://arxiv.org/abs/1410.1141.

Seyed Iman Mirzadeh, Mehrdad Farajtabar, Razvan Pascanu, and Hassan Ghasemzadeh. Understanding the role of training regimes in continual learning. *Advances in Neural Information Processing Systems*, 33:7308–7320, 2020.

Seyed Iman Mirzadeh, Arslan Chaudhry, Huiyi Hu, Razvan Pascanu, Dilan Gorur, and Mehrdad Farajtabar. Wide neural networks forget less catastrophically. *ICML*, 2021.

Preetum Nakkiran, Gal Kaplun, Yamini Bansal, Tristan Yang, Boaz Barak, and Ilya Sutskever. Deep double descent: Where bigger models and more data hurt. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. URL https://openreview.net/forum?id=B1g5sA4twr.

Neel Nanda and Tom Lieberum. A mechanistic interpretability analysis of grokking. *Alignment Forum*, Aug 2022.

Behnam Neyshabur, Srinadh Bhojanapalli, David McAllester, and Nati Srebro. Exploring generalization in deep learning. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett (eds.), *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pp. 5947–5956, 2017. URL https://proceedings.neurips.cc/paper/2017/hash/10ce03a1ed01077e3e289f3e53c72813-Abstract.html.

M. Pezeshki, Amartya Mitra, Yoshua Bengio, and Guillaume Lajoie. Multi-scale feature learning dynamics: Insights for double descent. *icml*, 2021a.

Mohammad Pezeshki, Oumar Kaba, Yoshua Bengio, Aaron C Courville, Doina Precup, and Guillaume Lajoie. Gradient starvation: A learning proclivity in neural networks. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 1256–1272. Curran Associates, Inc., 2021b. URL https://proceedings.neurips.cc/paper/2021/file/0987b8b338d6c90bbedd8631bc499221-Paper.pdf.

B.T. Polyak. Some methods of speeding up the convergence of iteration methods. *USSR Computational Mathematics and Mathematical Physics*, 4(5):1–17, 1964. ISSN 0041-5553. doi: https://doi.org/10.1016/0041-5553(64)90137-5. URL https://www.sciencedirect.com/science/article/pii/0041555364901375.

Phillip Pope, Chen Zhu, Ahmed Abdelkader, Micah Goldblum, and Tom Goldstein. The intrinsic dimension of images and its impact on learning. *arXiv preprint arXiv:2104.08894*, 2021.

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.

M. Riedmiller and H. Braun. A direct adaptive method for faster backpropagation learning: the rprop algorithm. In *IEEE International Conference on Neural Networks*, pp. 586–591 vol.1, 1993. doi: 10.1109/ICNN.1993.298623.

Sam T. Roweis and Lawrence K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, 2000. doi: 10.1126/science.290.5500.2323. URL https://www.science.org/doi/abs/10.1126/science.290.5500.2323.

Ravid Shwartz-Ziv and Naftali Tishby. Opening the black box of deep neural networks via information. *arXiv preprint arXiv: Arxiv-1703.00810*, 2017.

David Silver, Aja Huang, Christopher J. Maddison, Arthur Guez, Laurent Sifre, George van den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, Sander Dieleman, Dominik Grewe, John Nham, Nal Kalchbrenner, Ilya Sutskever, Timothy Lillicrap, Madeleine Leach, Koray Kavukcuoglu, Thore Graepel, and Demis Hassabis. Mastering the game of go with deep neural networks and tree search. *Nature*, 529:484–503, 2016. URL http://www.nature.com/nature/journal/v529/n7587/full/nature16961.html.

Leslie N. Smith and Nicholay Topin. Exploring loss function topology with cyclical learning rates. *arXiv preprint arXiv: Arxiv-1702.04283*, 2017.

Joshua B. Tenenbaum, Vin de Silva, and John C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000. doi: 10.1126/science.290.5500.2319. URL https://www.science.org/doi/abs/10.1126/science.290.5500.2319.

Vimal Thilak, Etai Littwin, Shuangfei Zhai, Omid Saremi, Roni Paiss, and Joshua M. Susskind. The slingshot mechanism: An empirical study of adaptive optimizers and the {Grokking Phenomenon}. 2022.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *CoRR*, abs/1706.03762, 2017. URL http://arxiv.org/abs/1706.03762.

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. *ArXiv*, abs/1611.03530, 2017.

Bojan Žunkovič and Enej Ilievski. Grokking phase transitions in learning local rules with gradient descent. *arXiv preprint arXiv: Arxiv-2210.15435*, 2022.

A Related works

Grokking Power et al. (2022) are the first to have studied and named the grokking phenomenon. They train a decoder-only causal transformer (Vaswani et al., 2017) to predict the response c of a binary operation of the form $a \circ b = c$, where a, b, c are discrete symbols without internal structure, and $\circ$ a binary operation (addition, composition of permutations, bivariate polynomials, etc). Liu et al. (2022) studied grokking in simple toy models: an embedding layer followed by a multilayer perceptron for classification and regression. The authors empirically show that generalization on algorithmic datasets coincides with the emergence of structure in embeddings, that the quality of representation predicts generalization, and they defined the notion of representation quality in the toy setting and showed that it predicts generalization by developing an effective theory to describe the learning dynamics of the representations in the same toy setting. They also illustrate phase diagrams, including memorization, comprehension, confusion, and grokking (similar to comprehension, but with a very large number of training steps between memorization and generalization) as a function of hyperparameters. The taxonomy used throughout this paper is identical to theirs, with the difference that we refer in our work to the phases along the training trajectory. We examined the loss landscape in more detail during these training phases but did not analyze the embedding structure. Thilak et al. (2022)'s slingshot mechanism correlates with our oscillation hypotheses. Their work focuses on characterizing periodic oscillations observed in the different phases of model training, and they were the first to empirically observe that grokking almost exclusively happens at the onset of slingshots and is absent without it. Our work reveals that these oscillations are caused at the landscape level by the escape of sharp mountains surrounding local optima. Moreover, the model weights change abruptly during the slingshot and slightly outside the slingshot. Other works have also been interested in grokking recently, such as the mechanistic interpretability of grokking (Nanda & Lieberum, 2022) and grokking as a phase transition (Žunkovič & Ilievski, 2022).

Neural networks loss landscape The visualization methods used in this paper were first introduced by Goodfellow & Vinyals (2015), and are still one the most used method in date to study the effect of loss landscapes on generalization (Goodfellow & Vinyals, 2015; Im et al., 2016; Smith & Topin, 2017; Keskar et al., 2017; Li et al., 2018; Feng & Tu, 2021). Li et al. (2018) observe that, when networks become sufficiently deep, neural loss landscapes quickly transition from being nearly convex to being highly chaotic, and that this transition coincides with a dramatic drop in generalization error, and ultimately to a lack of trainability. Feng & Tu (2021) investigate the connection between the loss function landscape and the stochastic gradient descent (SGD) learning dynamics. They show that around a solution found by SGD, the loss function landscape can be characterized by its flatness in each PCA direction. As Li et al. (2018), they show that optimization trajectories lie in an extremely low dimensional space.

Flatness, Sharpness Hochreiter & Schmidhuber (1997) defined a flat minimizer as a (large) region in weight space with the property that each weight vector from that region leads to a similar small error, and proposed an algorithm to search such a minimizer, called flat minimum search. Similarly, we can say that a minimizer is sharp when the loss function has a (very) high condition number near it, at any point in its basin of attraction. Keskar et al. (2017) characterize flatness using eigenvalues of the Hessian and empirically show that large-batch methods make the minimizers of training and testing functions sharp while small-batch methods make them flat. Hochreiter & Schmidhuber (1997) and Keskar et al. (2017) methods to quantify sharpness are not invariant to symmetries in the network, so, are not sufficient to determine generalization ability, as shown by Dinh et al. (2017); Neyshabur et al. (2017).

Related phenomena Grokking can be seen as an extreme gradient starvation phenomenon (Pezeshki et al., 2021b) or a very slow double descent (Nakkiran et al., 2020), for which the model only manages to learn the features necessary for generalization in the terminal phases of training (Pezeshki et al., 2021a). In fact, we observe that the validation loss exhibits a double descent behavior with an initial decrease, then growth, and a rapid decrease to zero when the model grows. Grokking can also be seen as a phase transition (see (Nanda & Lieberum, 2022; Žunkovič & Ilievski, 2022)). Regarding catastrophic forgetting, there is a link between it when learning many tasks and the sharpness of the optimum for each task; so that the slightest update to one task makes the optimum escape from its basin of attraction for the other tasks. Mirzadeh et al. (2020; 2021) formalize this.

B Limitations and Perspectives

Limitations Some limitations that would require further scrutiny include:

1. We view the loss surface with a dramatic reduction in dimensionality, and we need to be careful in interpreting these graphs.

2. We are only optimizing an unbiased estimate of the total loss function (added up across all training examples), whose structure may be different from the global loss function, so that the delay in generalization is an effect of each individual term in the loss function, or of the noise induced by the sampling of these terms, or of the stochasticity in the initialization of weights, generation of dropout masks, etc (Goodfellow & Vinyals, 2015). More research needs to be done to find out if it is indeed the fact that the loss surface is perturbed that causes the generalization in the long run.

Perspectives The future direction is to study what a feature learning-based explanation of double descent Pezeshki et al. (2021a) has in common with grokking and to see if it is possible to design a specific learning curve as Chen et al. (2021) derived for the generalization curve. Also, this work focused on detecting grokking; however, it ultimately paved the way to induce grokking. Another path of development is to move from arithmetic tasks with deterministic answers to language models and computer vision with more probabilistic answers.

C Notations

The index t is used to characterize the training steps (θ_t, etc.). $L_t = L(\theta_t)$ denotes the loss at θ_t (the parameter update at time t given the optimization algorithm of choice), $G_t = \nabla L(\theta_t)$ and $\mathcal{H}_t = \nabla^2 L(\theta_t)$ the gradient and the local Hessian matrix of the loss function at θ_t.

D Training settings

We focus on the following binary operations, for a prime number $p = 97$, and $q = p$: modular addition ($+$), $a \circ b = a + b \ (mod\ p)$ for $a, b \in [p]$; and multiplication in the permutation group S_5, $a \circ b = a \cdot b$ for $a, b \in S_5$.

For all experiments, we used a transformer with 2 layers, 128 embeddings, and 4 attention heads, with a total of roughly $4 \cdot 10^5$ non-embedding parameters. We use the following hyperparameters unless stated otherwise: AdamW optimizer with a learning rate of 10^{-4}, weight decay of 1 and

$(\beta_1, \beta_2) = (0.9, 0.98)$, linear learning rate warm up over the first 10 updates, and minibatch size of $|\mathcal{D}_{train}|$.

E GROKKING

E.1 TRAINING PHASES

Previous work (Power et al., 2022; Liu et al., 2022) used the terms confusion, memorization and comprehension in the phase diagram based on different hyperparameters, but in this paper they also refer to the phases along the training trajectory. Conventional deep neural networks training is split into two phases (Shwartz-Ziv & Tishby, 2017; Nakkiran et al., 2020; Feng & Tu, 2021). A first phase during which the network learns a function with a small generalization gap and a second phase during which the network starts to overfit the data, leading to an increase in the test error. However, Nakkiran et al. (2020) show that in some regimes, the test error decreases again and can reach a lower value at the end of the training compared to the first minimum, suggesting potential training phases to exploit. Feng & Tu (2021) distinguish two phases, an initial fast learning phase where the loss function decreases quickly and sometimes abruptly, followed by an exploration phase when the training error reaches its minimum value and the overall loss still decreases, but much more slowly and gradually. However, considering just two phases does not allow to study grokking properly, as the whole ingredient of grokking is in the memorization phase and in how the model gets in this phase and out.

E.2 NON-GROKKING

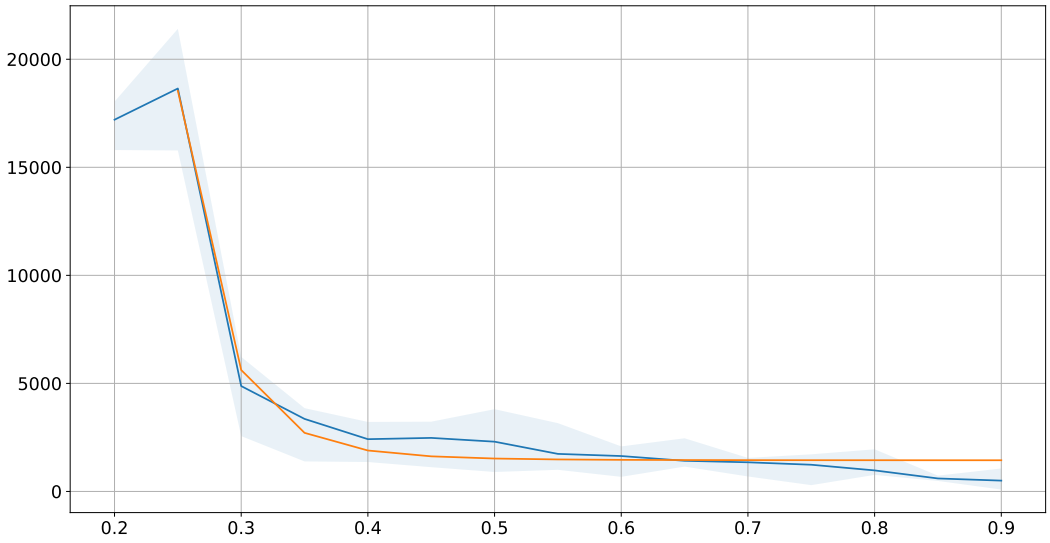


Figure 5: $t_4(r)$ is a transition step from memorization to generalization for each training data fraction $r >= 0.2$. The blue curve shows the empirically obtained values of $t_4(r)$ (y-axis) as a function of r (x-axis), and the orange curve represents an estimate of the form $t_4(r) = ar^{-\gamma} + b$ fitted on the obtained data, with $(\gamma, a, b) = (7.73, 1.09 \times 10^{15}, 1442.63)$ in this case.

Although it is easy to identify grokking, it is very difficult to give a formal definition of its opposite since nothing contradicts the fact that if the model is allowed to train indefinitely, it will not eventually generalize. Knowing the hyperparameters (learning rate, weight decays strength, ...) that allow to have grokking (let's note by H_g the set of values of such hyperparameters); we use them to train our model for different data sizes. Then, we fit a function that predicts t_4, the generalization step, for each training data fraction r. We then used the expression of $t_4(r)$ estimated as a proxy of the number of training steps needed to get the generalization for each r. That is, if for other hyper-parameters that are not in H_g, and for a given training data fraction r, we train a model for $t_4(r) + \epsilon$ (we used

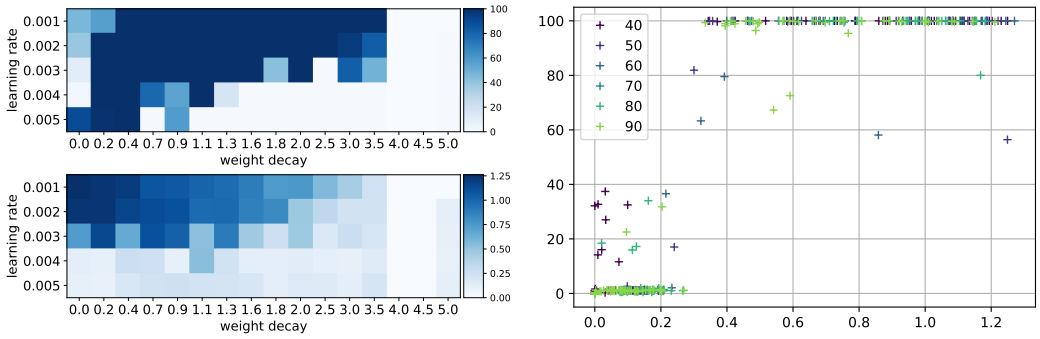


Figure 6: Left) The first figure (top) represents the validation accuracy (%) at the end of the training ($10k$ steps), and the second figure (bottom) represents the spectral energy (activity) in the training loss for the first 400 training steps ($r = 0.5$). On the x-axis we have the weight decay strength, and on the y-axis we have the learning rate. Right) On the x-axis we have the (normalize) activity (400 steps), and on the y-axis we have the validation accuracy (%), for different value of r. On the x-axis we have the weight decay strength, and on the y-axis we have the learning rate. A similarity is observed between the oscillation patterns in the training loss during the initial stages of training and the validation accuracy. This suggests that the spectral signature can be used as an indicator or proxy for the upcoming grokking phenomenon. The highest degree of generalization is typically observed when using small learning rates and small weight decay. While large learning rates may increase oscillations, this does not directly lead to grokking and is not necessarily evident in the early stages of training. Instead, such effects become more noticeable near the basin of attraction of the minimum.

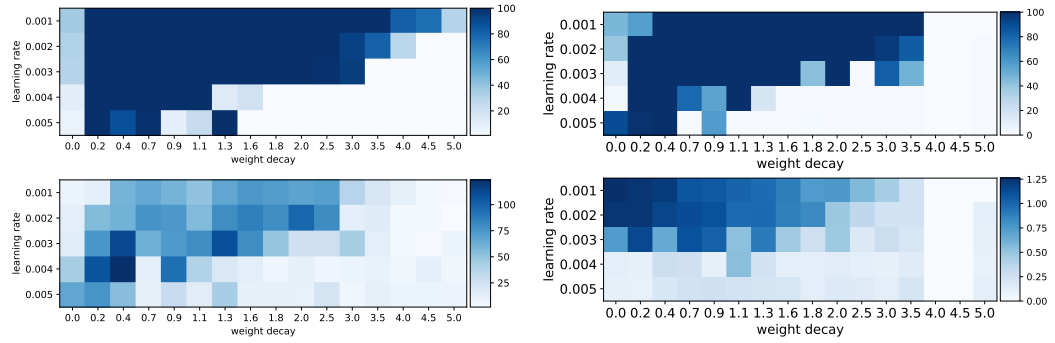
$\epsilon = 1k$) steps and have no generalization, we can stop the training. This does not necessarily imply that there will be no grokking if the model is left in training for longer. The limit of this approach is that $t_4(r)$ is just an empirical law estimated for $r \geq r_{min} > 0$, which collapses when $r \to 0$.

E.3 The over-parameterization ratio and the effect of the optimizer parameters

In general, more data leads to faster grokking (figure 5-b), i.e. $t_2(r)$ and $t_4(r)$ are a decreasing functions of r. Empirically, t_4 follows a power law of the form $t_4(r) = ar^{-\gamma} + b$. This was first predicted by Žunkovič & Ilievski (2022). For $r \geq r_{min}$, we found $(\gamma, a, b) = (7.73, 1.09 \times 10^{15}, 1442.63)$ for modular addition (figure 5.b), and $(\gamma, a, b) = (1.18, 1.85 \times 10^{6}, 0.0)$ for multiplication in S_5, in H_g. Smaller learning rates require more training steps for convergence, whereas larger learning rates result in rapid changes and require fewer training epochs. However, a learning rate that is too large can cause the model to converge too quickly to a suboptimal solution, whereas a learning rate that is too small can cause the process to get stuck. This is verified with grokking because the more we increase the learning step, the faster we observe grokking, up to a threshold that depends on the value of the weight decay strength used.

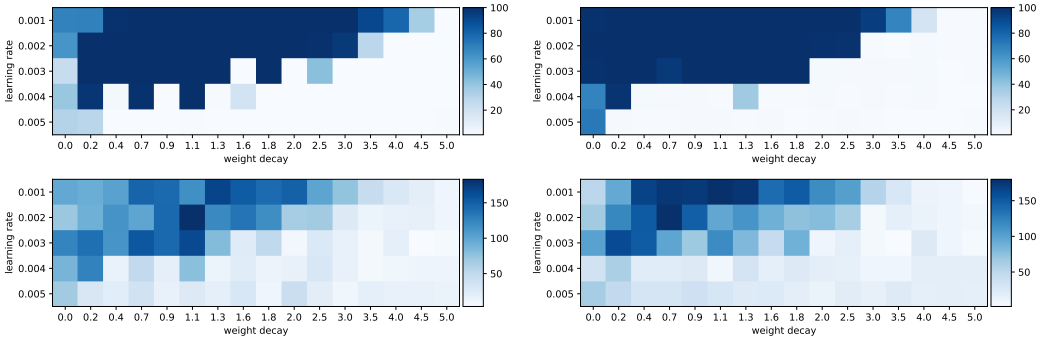
F Spectral signature

A starting observation is that the learning curves of models that grok exhibit oscillatory behaviors. A related phenomenon, the slingshot effect, was observed and named recently by Thilak et al. (2022). Cyclic transitions between a stable and unstable training regime characterize this phenomenon. Thilak et al. (2022) characterized this phenomenon as the complete cycle starting with the norm growth phase and ending with the norm plateau phase and found that it is ubiquitous and can be easily replicated in multiple scenarios, encompassing a variety of models and data sets. They observe that slingshots and grokking tend to come in tandem, that is grokking almost exclusively happens at the onset of slingshots, and is absent without it. As pointed out by Thilak et al. (2022), this type of transition is reminiscent of the *catapult phenomenon* (Lewkowycz et al., 2020), where the loss initially increases and begins to decrease once the model catapults to a region of lower curvature early in the training, for a sufficiently large training step.



(a) $r = 0.4$

(b) $r = 0.5$



(c) $r = 0.7$

(d) $r = 0.9$

Figure 7: For (a)-(d), the first figure (top) represents the validation accuracy (%) at the end of the training (10k steps), and the second figure (bottom) represents the spectral energy (activity) in the training loss for the first 400 training steps. On the x-axis, we have the weight decay strength, and on the y-axis, we have the learning rate. A similarity is observed between the oscillation patterns in the training loss during the initial stages of training and the validation accuracy.

We first try to comparatively quantify the oscillations in the training loss when the model groks and when it does not grok, the idea being to stop the training if the model does not seem to be able to grok in order to save computational resources. From the gradient flow equation $\dot{\theta} = -G(t)$, it holds that $\dot{L} \approx -\|G(t)\|^2$ and $\ddot{L} \approx 2G(t)^T H(t) G(t) = 2\sum_i \lambda_i(t) \langle G(t), v_i(t) \rangle^2$ with $\{\lambda_i(t)\}_i$ the spectrum of $H(t)$, and $\{v_i(t)\}_i$ the associated eigenvectors. From this, it becomes clear that the evolution of L over time depends on the norm of the gradient, and how fast it changes depends on the curvature of its landscape. Any signal that can be represented as a variable that varies in time has a corresponding frequency spectrum. We considered the training loss L over the training steps (and in the early stages of training) as signals and analyzed its spectral signature. By spectral signature of the loss, we mean any measure or set of measures that can quantify the oscillations in the loss, such as the spectral energy or the Hjorth parameters (Hjorth, 1970) - activity, mobility, and complexity. The Hjorth activity, which is the variance of the signal in the time domain, is equal to the spectral energy if the signal L has zero mean. The latter condition is obtained by passing L through a sufficiently low-pass filter in the frequency domain, which removes its non-oscillatory components [2] that are not necessary for the quantification of the oscillations. In this case, the Hjorth parameters are directly related to the loss landscape, as illustrated below.

The Hjorth activity represents the signal power, the surface of the power spectrum in the frequency domain. It is given by $m_0(L)$, which is equal to $\int L^2(t)dt$ by the parseval's theoremThe Hjorth mobility is the mean frequency or the proportion of standard deviation of the power spectrum

[2] A non-oscillatory component of a signal is any component that does not vary rapidly with time, such as a constant value, a linear trend, or a smooth curve. These components are generally considered to be low-frequency signals and can be removed by passing the signal through a low-pass filter.

and is given by $\sqrt{m_2(L)/m_0(L)}$ with $m_2(L) = \int |\omega \mathcal{F}(L)(\omega)|^2 d\omega = \int \dot{L}^2(t)dt \approx m_0(\dot{L})$ the activity of the gradient norm $\|G(t)\|^2$. In a similar way, The Hjorth complexity, which indicates how the shape of a signal is similar to a pure sine wave, is given by $\sqrt{m_4(L)/m_2(L)}$ with $m_4(L) = \int |\omega^2 \mathcal{F}(L)(\omega)|^2 d\omega = \int \ddot{L}^2(t)dt \approx m_0(\ddot{L})$ the activity of the hessian spectrum.

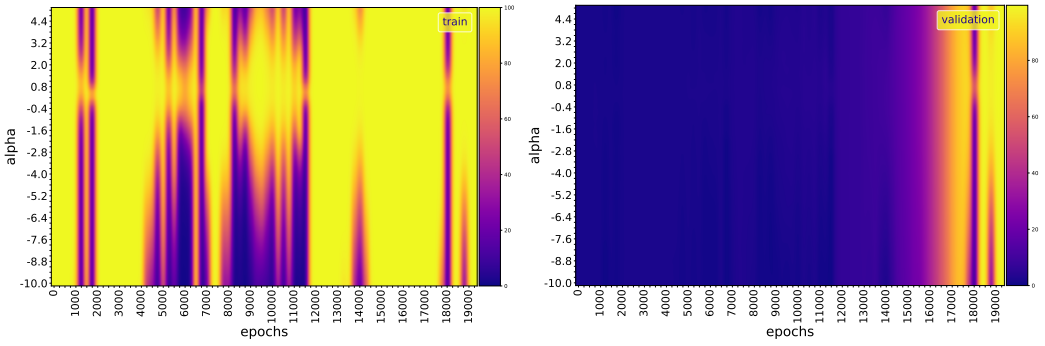
Figures and 6 and 7 show a similarity between the oscillation in the training loss in the early phases of training and the validation accuracy for many values of r.

G LOSS LANDSCAPE

G.1 1D/2D PROJECTION

Since neural loss functions live in a very high-dimensional space, visualizations are only possible using low-dimensional 1D (line) or 2D (surface) plots. In this work, we consider the approach of Li et al. (2018). Let θ be the point near which we want to observe the loss landscape. We plot the loss and the accuracy as a function of $A \subseteq \mathbb{R}$, $f(\alpha) = L(\theta + \alpha\vec{\delta})$, where $\vec{\delta}$ is a direction vector carefully chosen in $\vec{\Theta}$ [3]. Due to scale invariance in network weights, this approach may fail to capture the intrinsic geometry of loss surfaces. To remove this scaling effect, we plot the loss functions using an adaptation of the filter-wise normalized directions (Li et al., 2018), that is $w_k(\vec{\delta}) \leftarrow \frac{w_k(\vec{\delta})}{\|w_k(\vec{\delta})\|}\|w_k(\theta)\|$ and $b_k(\vec{\delta}) \leftarrow \frac{b_k(\vec{\delta})}{|b_k(\vec{\delta})|}|b_k(\theta)|$ for each weight $W = [\dots, w_k, \dots]^T$ (w_k is a vector) and bias $b = [\dots, b_k, \dots]$ (b_k is a scalar) in each layer of θ and $\vec{\delta}$ [4]. This loss-landscape visualization approach, although simple, has the advantage of allowing to visualize the potential local convexity of the loss in the chosen direction [5].

For $r = 0.3$, figure 3 shows the 1D projection of the grokking accuracy surface for a single epoch of training (just after the grokking step), while figures 4 ($\vec{\delta_t} \propto \theta^* - \theta_t$), 8 ($\vec{\delta_t} \propto \theta_{t+1} - \theta_t$), 10 ($\vec{\delta_t} \propto \theta_0 - \theta_t$) and 9 (random $\vec{\delta_t}$) show it for different training epochs.



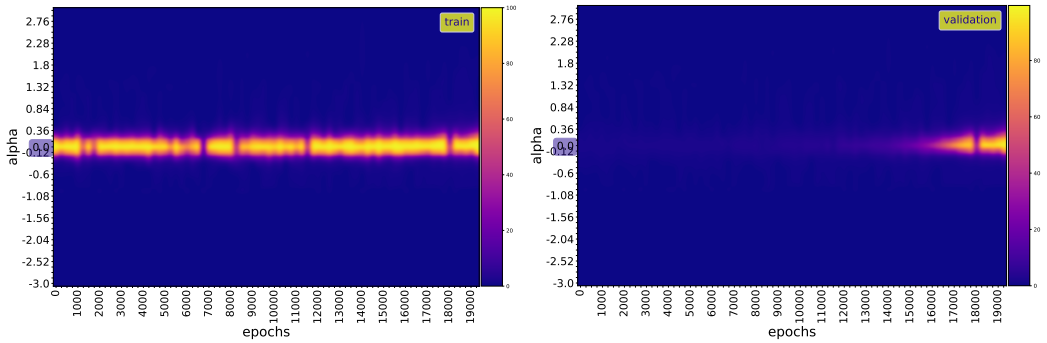
(a) Accuracy surface : $f_t(\alpha) = Acc(\theta_t + \alpha\vec{\delta_t})$ for each epoch t

Figure 8: 1D projection of the grokking loss surface for epoch to epoch, $\vec{\delta_t} \propto \theta_{t+1} - \theta_t \propto -\nabla L(\theta_t)$. This allows a closer look at the behavior of the landscape in the (local) direction of the gradient at each training step t. For some t, the loss, although disturbed, remains locally flat (up to a resolution used for visualization). These are the instants during which the weight progresses along the directions of least curvature, resulting in a stagnation of the loss. These points are usually followed by a slingshot.

[3] In 2D, the loss is plot as a function of $A \times B \subseteq \mathbb{R}^2$, $f(\alpha, \beta) = L(\theta + \alpha\vec{\delta} + \beta\vec{\eta})$, where $\vec{\delta}$ and $\vec{\eta}$ are two carefully chosen direction vectors in $\vec{\Theta}$. $\vec{\delta}$ and $\vec{\eta}$ can be randomly chosen or defined by $\vec{\delta} = \theta^{'} - \theta$ and $\vec{\eta} = \theta^{''} - \theta$, with $\theta^{'}$ and $\theta^{''}$ another points whose choice will be specified.

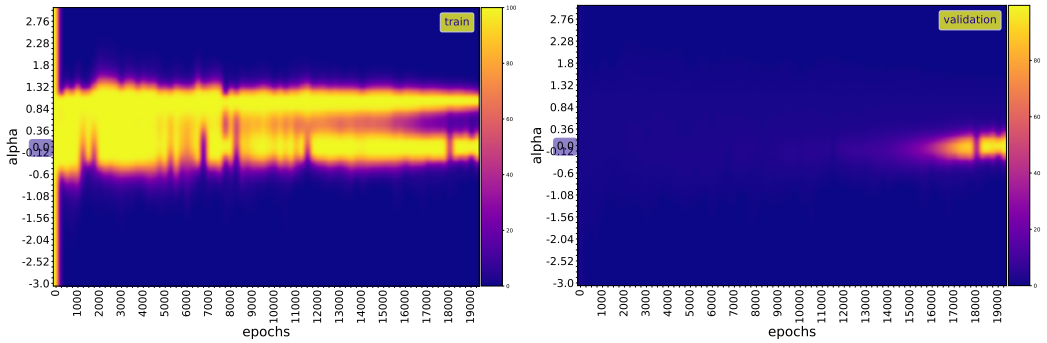
[4] The same as for $\vec{\eta}$ when applied

[5] A function $f : \mathbb{R}^n \to \mathbb{R}$ is convex if and only if $g_{(x,y)} : \alpha \in [0,1] \mapsto g_{(x,y)}(\alpha) = f\big(x + \alpha(y - x)\big)$ is convex for all $x, y \in \mathbb{R}^n$.



(a) Accuracy surface : $f_t(\alpha) = Acc(\theta_t + \alpha\vec{\delta}_t)$ for each epoch t

Figure 9: 1D projection of the grokking loss surface in a random direction $\vec{\delta}$. In this direction, no obstacles are visible along the trajectory, and the surface is rather well-conditioned.



(a) Accuracy surface : $f_t(\alpha) = Acc(\theta_t + \alpha\vec{\delta}_t)$ for each epoch t

Figure 10: 1D projection of the grokking loss surface from the initialization point to any step t, $\vec{\delta}_t \propto \theta_0 - \theta_t$. In this direction, each point considered behaves locally as a minimum, and as one approaches the optimum, local minima become preponderant.

G.2 Loss surface curvature

The approach described in the previous section allows us to visualize the loss surface under a dramatic dimensionality reduction, and we need to be careful how we interpret the plots. One way to measure the level of convexity in a loss function is to compute the principal curvatures, which are simply eigenvalues of the Hessian. Since we are in a very high dimension, it would be impractical to compute all the eigenvalues of the Hessian or simply to compute the Hessian itself. To avoid this dimensional problem, we estimated the curvature through the condition number of the Hessian of $L(\theta)$ at each step t, which is simply the ratio $\lambda_{min}(t)/\lambda_{max}(t)$, where $\lambda_{min}(t)$ and $\lambda_{max}(t)$ are, respectively, the minimum and maximum eigenvalues of the Hessian of L_t. The condition number is a direct measure of pathological curvature. Larger condition numbers imply slower convergence of gradient descent. Li et al. (2018) compute the above ratio at each point of the loss surface and observe that the convex-looking regions in the surface plots correspond to regions with insignificant negative eigenvalues, while chaotic regions contain large negative curvatures, and that for convex-looking surfaces, the negative eigenvalues remain extremely small.

To measure the level of curvature of the loss function, we compute the maximum (λ_{max}) and minimum (λ_{min}) eigenvalue of its Hessian (figure 11). We observe that there is no significant negative curvature in the trajectory. The curvature remains generally positive and is greatly disturbed at the slingshot points (λ_{min} remains in general close to 0 while λ_{max} is large, but during slingshot λ_{min} becomes negative). Our loss is λ_{max}-smooth, i.e. its gradient is λ_{max}-Lipschitz. It is known from optimization literature that a function with bounded Hessian eigenvalues has a gradient that

tends to decay when the parameter gets closer to the minimum, in contrast to a non-smooth one that generally has abrupt bends at the minimum, which causes significant oscillations for gradient descent (Bubeck et al., 2015).

In fact, Let $\{\lambda_t(i)\}_i$ be the eigenvalues of $\mathcal{H}_t$ and $\{v_t(i)\}_i$ the associated eigenvectors. For a very small step size ϵ_t of SGD, $L_{t+1} - L_t \approx -\epsilon_t \|G_t\|^2 + \frac{1}{2}\epsilon_t^2 G_t^T \mathcal{H}_t G_t - o(\epsilon_t^2 \|G_t\|^2)$. Futher, if $\lambda_t(i) > 2/\epsilon_t$, we get $-\epsilon_t \|G_t\|^2 + \frac{1}{2}\epsilon_t^2 G_t^T \mathcal{H}_t G_t > 0$ [6]. When $\mathcal{H}_t$ has some large positive eigenvalues (i.e., high-curvature directions) and some eigenvalues close to 0 (i.e., low-curvature directions), gradient descent bounces back and forth in high-curvature directions and makes slow progress in low-curvature directions. In this case, the optimization problem has an ill-conditioned curvature. Furthermore, if the loss function near θ_t has a high condition number, that is, very small steps cause an increase in the cost function (for example, if θ_t is a very sharp minimum surrounded by high loss regions), the optimization problem becomes also ill-conditioned. During training, if the gradient norm does not shrink but $G_t^T \mathcal{H}_t G_t$ increases in order of magnitude, learning can become very slow despite a strong gradient. The above observation ($\lambda_t(i) > 2\epsilon_t^{-1}$) is similar to what Herrmann et al. (2022) [Theorem 2.1] defines for the eigenvalues of a positive-definite $\mathcal{H}_t$, as a locally chaotic training behavior of the Local Lyapunov Exponents. They show evidence that neural network training is intrinsically locally chaotic due to the negative eigenspectrum of the Hessian and that network training with SGD exhibits globally edge-chaotic behavior. This observation is also linked to the *progressive sharpening* phenomenon (Cohen et al., 2021) in which $max_i \lambda_t(i)$ increases and reaches a value that is equal to or slightly larger than $2\epsilon_t^{-1}$, leading the model to enter an Edge of Stability regime where loss shows non-monotonic training behavior over short time spans (Thilak et al., 2022).

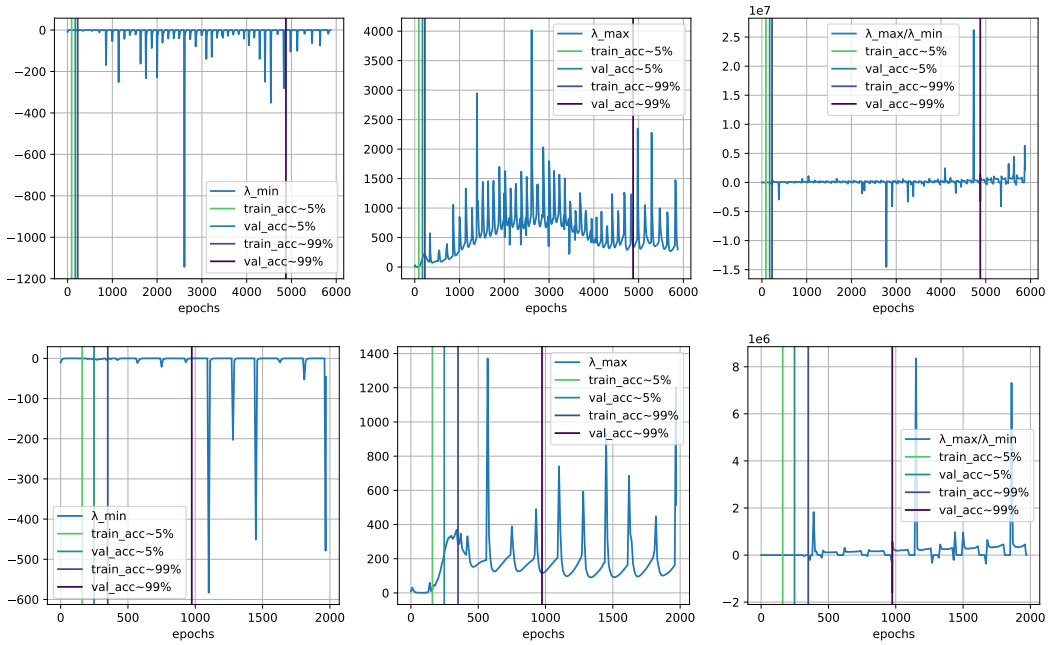
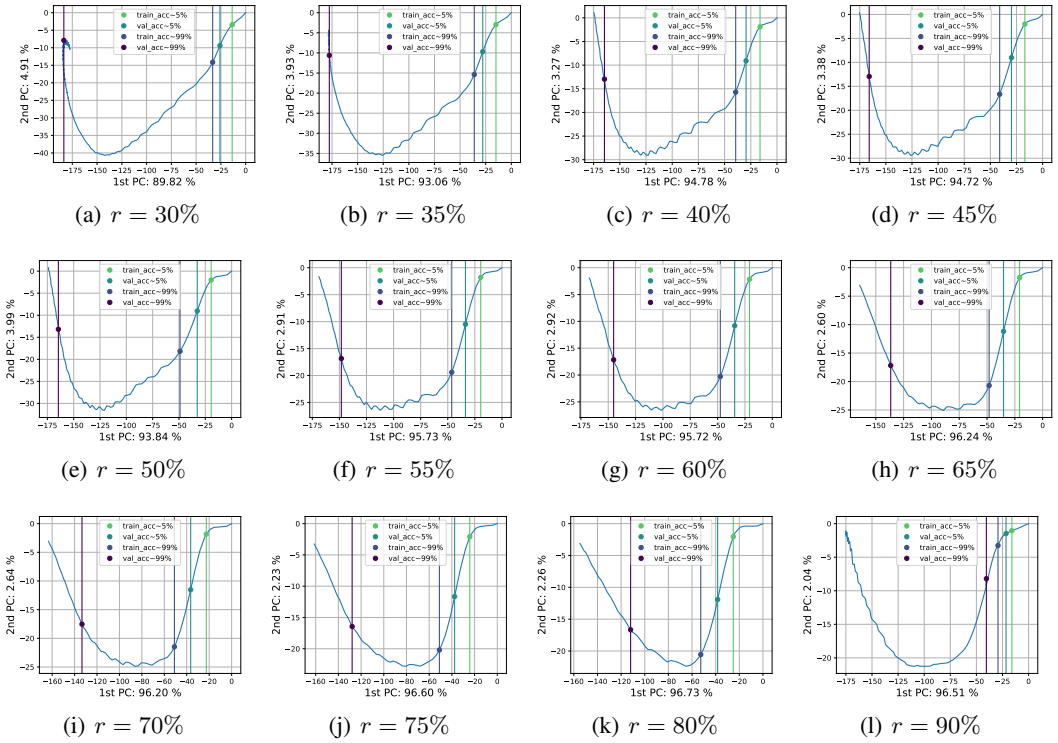


Figure 11: Condition number of the Hessian over training epoch for modular addition (top, $r = 0.3$, bottom, $r = 0.8$). We can see that there is no significant negative curvature in the trajectory (figure 11). The curvature remains in general positive and is greatly disturbed at the slingshot points.

[6] This comes come from the fact that near θ_t, $L(\theta) \approx L_t + (\theta - \theta_t)^T G_t + \frac{1}{2}(\theta - \theta_t)^T \mathcal{H}_t (\theta - \theta_t) + o(\|\theta - \theta_t\|^2)$. Taking $\theta = \theta_t - \epsilon_t G_t$ gives the first approximation. The last inequality is obtained with $G_t^T \mathcal{H}_t G_t = \sum_i \lambda_t(i) \langle G_t, v_t(i) \rangle^2$ and $\sum_i \langle G_t, v_t(i) \rangle^2 = \|G_t\|^2$.

G.3 OTHER LANDSCAPE MEASURES

To see how far optimization strays from the primary linear subspace, we plot the norm of the residual of the parameter value after projecting the parameters at each epoch into the 1-D subspace using the following two approaches. Let $M = [\theta_t - \theta_T]_{1 \leq t \leq T-1} \in \mathbb{R}^{(T-1) \times d}$ where T is the total number of training steps. We applied PCA to the matrix M, select the 2 most explanatory directions, then project each parameter θ_t on these two directions to have $\alpha(t)$ and $\beta(t)$. In figure 12, we plot the projection along the two first PCA axes from initialization to solution, $\alpha(t)$ and $\beta(t)$. More than 98% of the total variance in the parameter space occurs in the first 2 PCA modes, much smaller than the total number of weights, suggesting that the optimization dynamics are embedded in a low-dimensional space (Li et al., 2018; Feng & Tu, 2021).



(a) $r = 30\%$ (b) $r = 35\%$ (c) $r = 40\%$ (d) $r = 45\%$

(e) $r = 50\%$ (f) $r = 55\%$ (g) $r = 60\%$ (h) $r = 65\%$

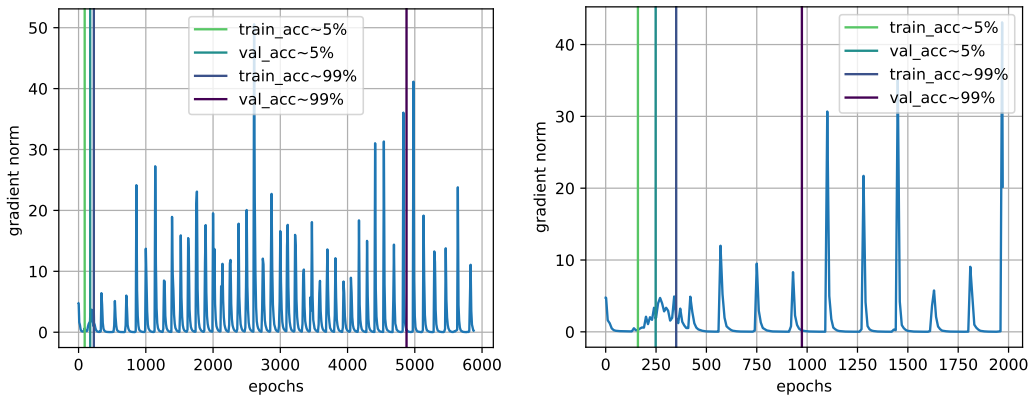
(i) $r = 70\%$ (j) $r = 75\%$ (k) $r = 80\%$ (l) $r = 90\%$

Figure 12: Projection along the axis from initialization to solution, $\alpha(t)$ and $\beta(t)$. We can see that more than 98% of the total variance in the parameter space occurs in the first 2 PCA modes, much smaller than the total number of weights, suggesting that the optimization dynamics are embedded in a low-dimensional space.

Also, the cosine similarity measure between the model weights from one epoch to the next remains almost constant, except at the slingshot location (figure 14). This allows us to see that our model passes through the abovementioned anomalies. It has been shown that with high probability over the initialization, the iterates of the gradient descent algorithm even stay in a small fixed neighborhood of the initialization during training. Because the parameters only move very little, this type of training has also been coined lazy training (Chizat et al., 2019; **?**). These steps, where the distances change abruptly, are caused by spikes in the training gradient (figure 13).

G.4 THE SLINGSHOT MECHANISM

Above, we derive $\dot{L}(t) = -\|G(t)\|^2$ for a small enough step size. This relaxation property of gradient descent reflects the fact that the loss function cannot increase. However, we lose this property with certain accelerated or adaptative methods like Adam or on ill-conditioned problems, as we saw above with $L_{t+1} > L_t$ for $\epsilon_t \lambda_t(i) > 2$. In the case of grokking, the loss, even when it becomes zero, presents a sudden growth, followed by decay, and this is in a periodic way. Thilak et al. (2022) have



(a) $r = 0.3$ (b) $r = 0.8$

Figure 13: Gradient Norm $\|G_t\|^2$. Most of the time, there is no significant gradient, which results in a weak progression of gradient descent.

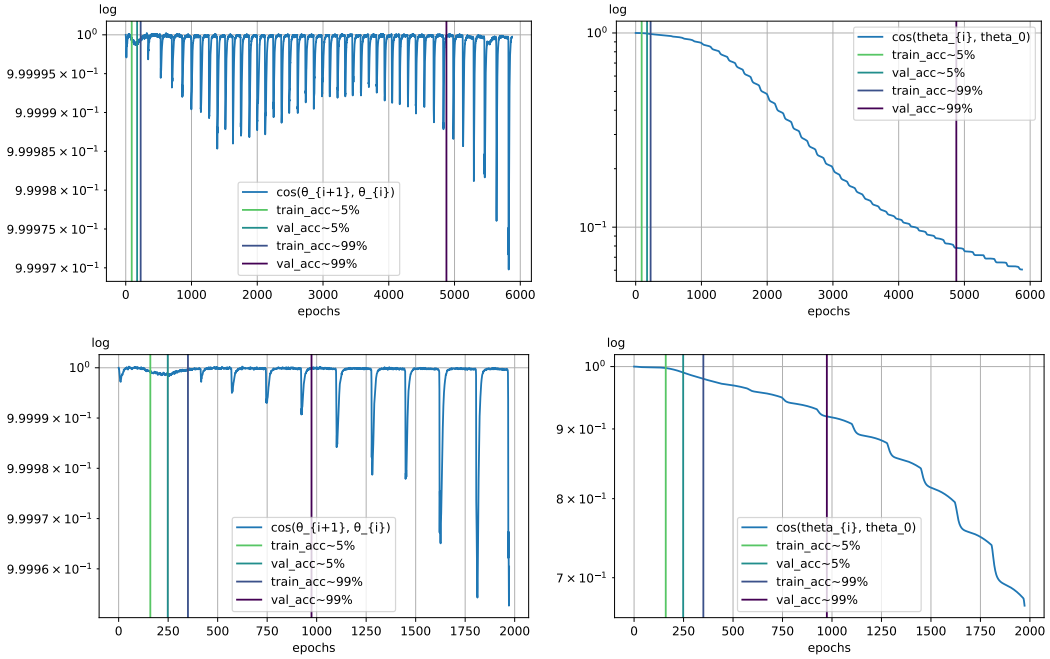


Figure 14: Relative consine similarity ($r = 0.3$ and 0.8). Left) $cos(\theta_t, \theta_{t+1})$ Right) $cos(\theta_t, \theta_0)$. Most of the time the weights progress very little, jumping when the gradient explodes, with a slingshot effect.

recently shown that this phenomenon is general to the optimization of deep neural networks. Between two slingshots, the gradient is almost zero, all eigenvalues of the Hessian are nonnegative, and there is one direction of very large curvature that dominates the others and the weights update, resulting in a small change in weight from one training step to another. The model thus seems to traverse a flat valley.

To reduce this phenomenon (i.e., the amplitude of the spikes), we clipped the gradient norm during training (using a threshold $\eta > 0$). This had the effect of slowing down generalization but not preventing it.

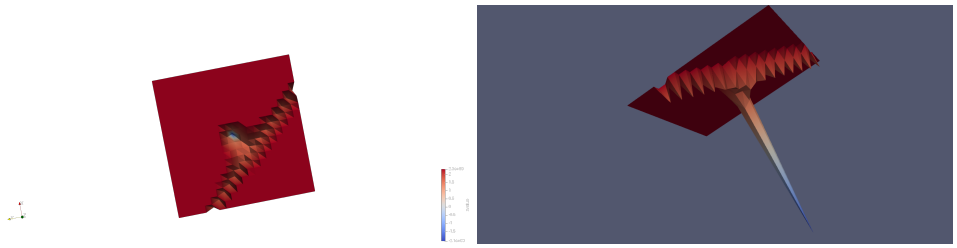


Figure 15: 3D ($r = 0.5$)

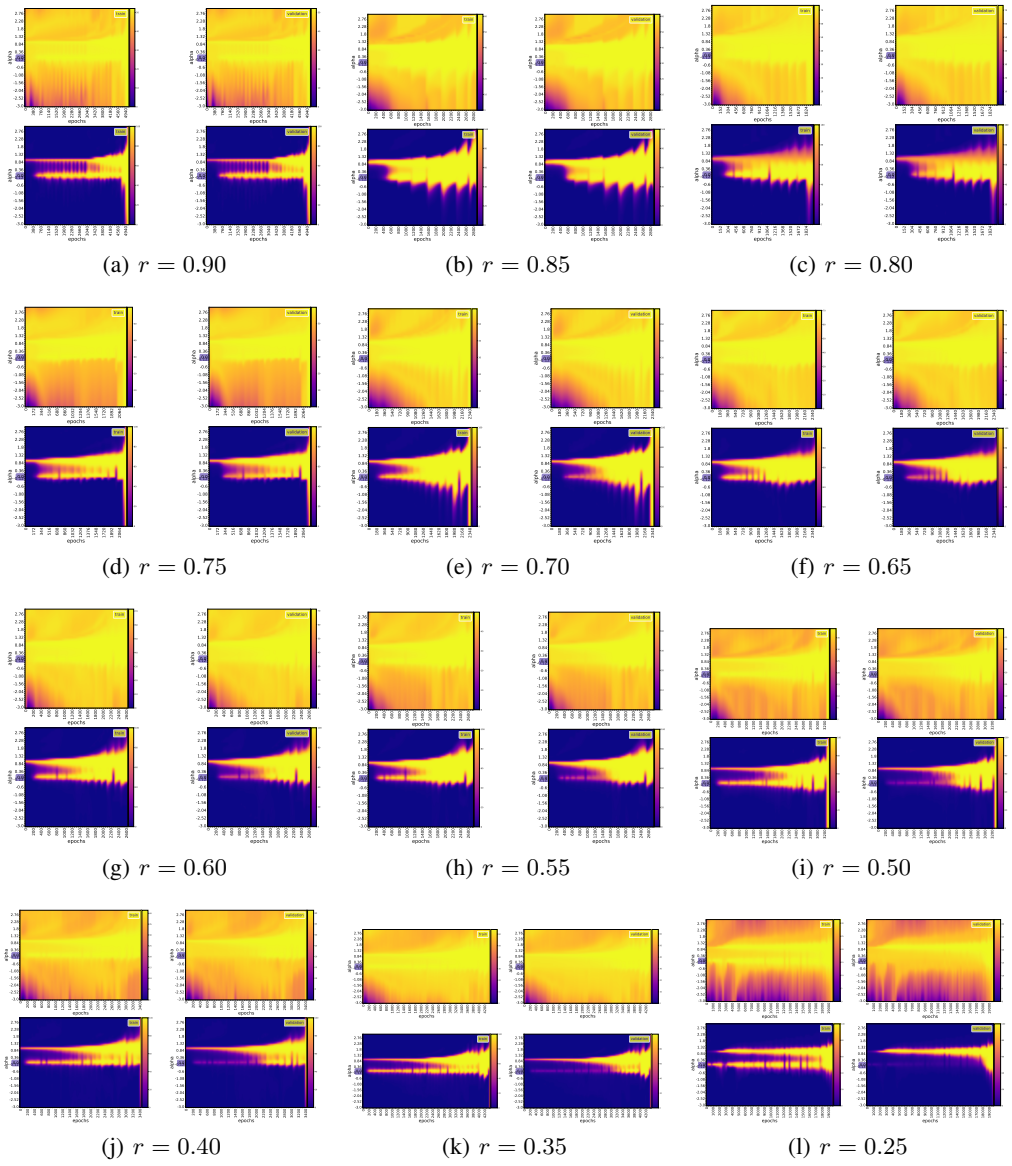
$$\dot{\theta} = \begin{cases} -\frac{\eta}{\|G(t)\|} G(t) & \text{if } \|G(t)\| \geq \eta \\ -G(t) & \text{otherwise.} \end{cases} \quad (\eta > 0)$$

$$\implies \dot{L} = \begin{cases} -\eta \|G(t)\| & \text{if } \|G(t)\| \geq \eta \\ -\|G(t)\|^2 & \text{otherwise.} \end{cases} \quad (\eta > 0)$$

The spikes are reduced as $\eta \to 0$, but remain visible as $\theta \to \theta^*$. By reducing the learning rate extremely, the spikes also become less visible, but we also pay the same cost because the model takes more steps to generalize.

G.5 Three-dimensional visualizations

We can accomplish this by viewing a heatmap of the cost function in 2D (figure 15).



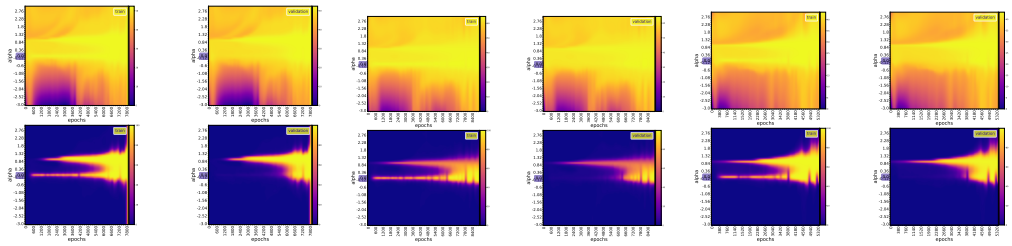
(a) $r = 0.90$ (b) $r = 0.85$ (c) $r = 0.80$

(d) $r = 0.75$ (e) $r = 0.70$ (f) $r = 0.65$

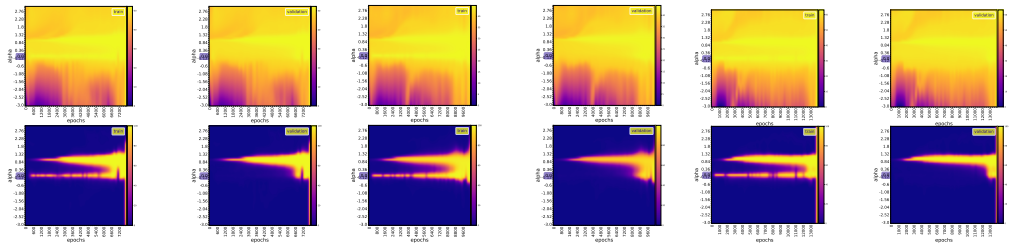
(g) $r = 0.60$ (h) $r = 0.55$ (i) $r = 0.50$

(j) $r = 0.40$ (k) $r = 0.35$ (l) $r = 0.25$

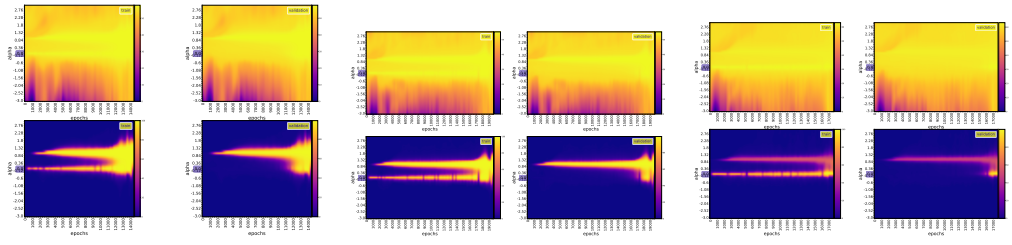
Figure 16: 1D projection of the grokking loss (top) and accuracy (bottom) surface for different values of the training data fraction r, for **modular addition**. The direction used for each training epoch t is $\vec{\delta}_t = \theta^* - \theta_t$.



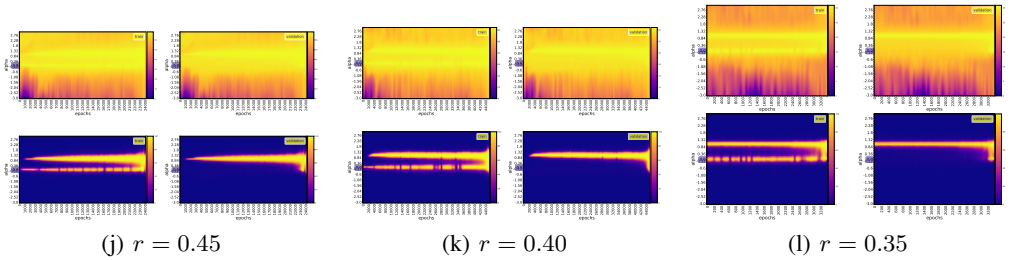
(a) $r = 0.90$ (b) $r = 0.85$ (c) $r = 0.80$



(d) $r = 0.75$ (e) $r = 0.70$ (f) $r = 0.65$

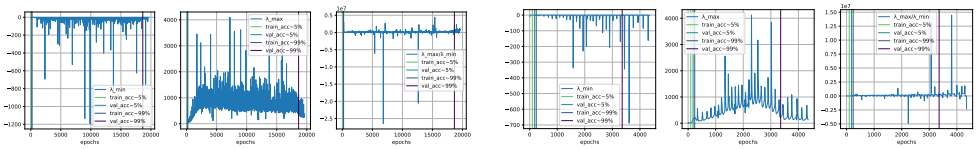


(g) $r = 0.60$ (h) $r = 0.55$ (i) $r = 0.50$



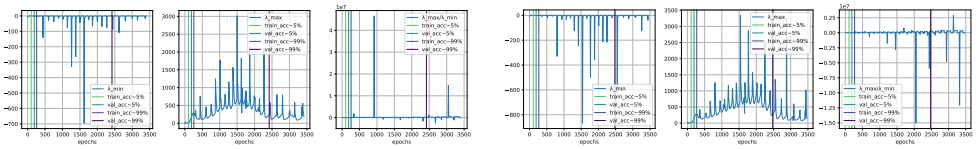
(j) $r = 0.45$ (k) $r = 0.40$ (l) $r = 0.35$

Figure 17: 1D projection of the grokking loss (top) and accuracy (bottom) surface for different values of the training data fraction r, for **multiplication in** S_5. The direction used for each training epoch t is $\vec{\delta}_t = \theta^* - \theta_t$.



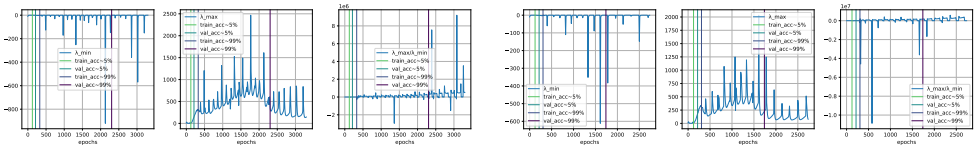
(a) $r = 0.25$

(b) $r = 0.35$



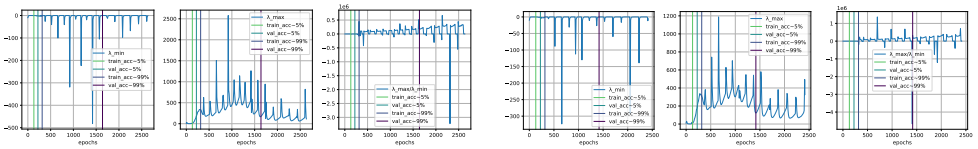
(c) $r = 0.40$

(d) $r = 0.45$



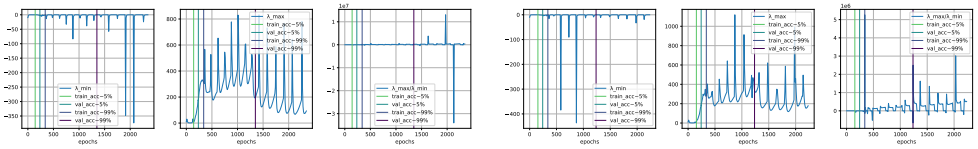
(e) $r = 0.50$

(f) $r = 0.55$



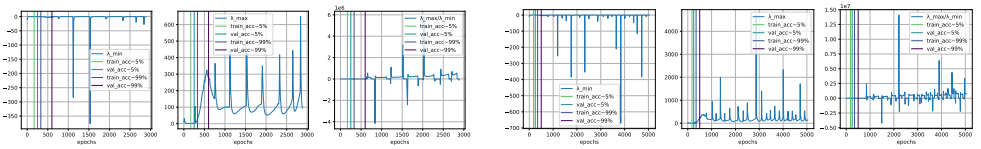
(g) $r = 0.60$

(h) $r = 0.65$



(i) $r = 0.70$

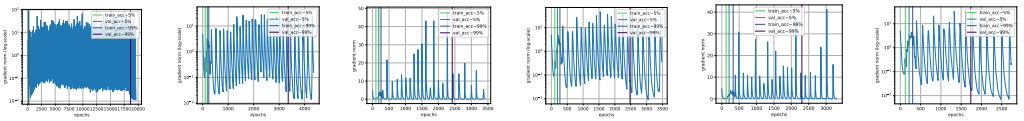
(j) $r = 0.75$



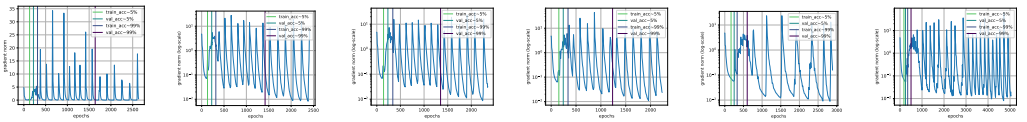
(k) $r = 0.85$

(l) $r = 0.90$

Figure 18: Condition number of the Hessian over training epoch for modular addition.

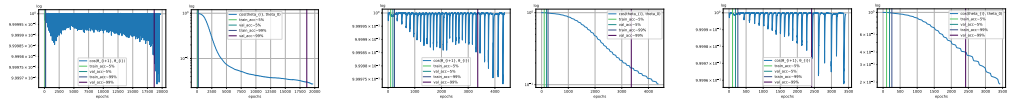


(a) $r = 0.25$ (b) $r = 0.35$ (c) $r = 0.40$ (d) $r = 0.45$ (e) $r = 0.50$ (f) $r = 0.55$

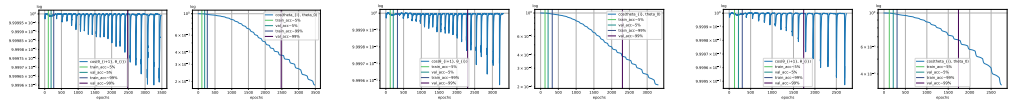


(g) $r = 0.60$ (h) $r = 0.65$ (i) $r = 0.70$ (j) $r = 0.75$ (k) $r = 0.85$ (l) $r = 0.90$

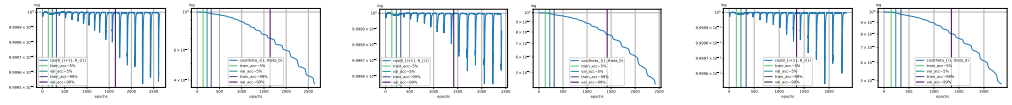
Figure 19: Gradient Norm for modular addition.



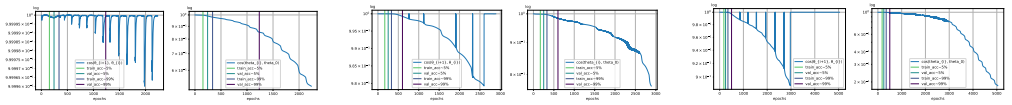
(a) $r = 0.25$ (b) $r = 0.35$ (c) $r = 0.40$



(d) $r = 0.45$ (e) $r = 0.50$ (f) $r = 0.55$



(g) $r = 0.60$ (h) $r = 0.65$ (i) $r = 0.70$



(j) $r = 0.75$ (k) $r = 0.85$ (l) $r = 0.90$

Figure 20: Relative cosine distance for modular addition. Left) $cos(\theta_t, \theta_{t+1})$ Right) $cos(\theta_t, \theta_0)$

H INTRINSIC DIMENSION ESTIMATION

We also empirically investigate the relationship between the evolution of the objective function of a neural network during training and the evolution of the intrinsic dimension (ID) of the activation manifolds of its layers. We found that the dimensionality of the network layers (mainly the last layer) correlates with oscillations in training and validation performances (loss and accuracy) both in toy models (Liu et al., 2022) and transformer (Power et al., 2022). The intrinsic dimensionality grows at the very beginning of the training, then decreases (oscillating very strongly) until it stabilizes at a certain level of the training.

H.1 METHODS

Data are usually represented by high-dimensional feature vectors, but in many cases, they could be, in principle, embedded in lower-dimensional spaces without any loss of information (Facco et al., 2017). In other words, any set of points of a high-dimensional ambient space $\mathbb{R}^D$ usually actually lies on (or close to) another space of dimension d smaller than D : d is then called the intrinsic dimension of the ambient space $\mathbb{R}^D$. Several methods have long been developed to estimate this intrinsic dimension (Roweis & Saul, 2000; Tenenbaum et al., 2000; Belkin & Niyogi, 2001; Donoho & Grimes, 2003; Levina & Bickel, 2004; David J.C. & Zoubin, 2005), among which Two Nearest Neighbors (TWONN) (Facco et al., 2017) and Maximum Likelihood Estimation (MLE) approaches (Levina & Bickel, 2004; David J.C. & Zoubin, 2005) that examine a neighborhood around each point and compute the Euclidean distance to the nearest neighbor k^{th} (Pope et al., 2021).

Assuming that the density is constant within small neighborhoods, the MLE (Levina & Bickel, 2004) uses a Poisson process to model the number of points found by random sampling within a given radius around each sample point (Pope et al., 2021). By relating the rate of this process to the surface area of the sphere, the likelihood equations yield an estimate of the intrinsic dimension at a given point x as

$$\hat{m}_k(x) = \left[\frac{1}{k-1} \sum_{j=1}^{k-1} \log \frac{T_k(x)}{T_j(x)} \right]^{-1}$$

where $T_j(x)$ is the Euclidean distance from x to its j^{th} nearest neighbor. Levina & Bickel (2004) propose to average the local estimates at each point to obtain a global estimate (n is the number of samples) :

$$\hat{m}_k = \frac{1}{n} \sum_{i=1}^{n} \hat{m}_k(x_i)$$

David J.C. & Zoubin (2005) suggest a correction based on the averaging of the inverses:

$$\hat{m}_k = \left[\frac{1}{n} \sum_{i=1}^{n} \hat{m}_k(x_i)^{-1} \right]^{-1}$$

TWONN (Facco et al., 2017) uses just the two nearest neighbors of each point to make the estimate. The distribution of $R = \frac{\Delta v_2}{\Delta v_1}$ has as probability density function $g(R) = \frac{1}{(1+R)^2}$ (Facco et al., 2017), where $\Delta v_l = \omega_d (r_l^d - r_{l-1}^d)$ is the volume of the hyperspherical shell enclosed between two successive neighbors $l-1$ and l of a given point i of the dataset (r_l being the distance between i and its l^{th} nearest neighbor), d the dimensionality of the space in which the points are embedded and $\omega_d = \frac{\pi^{d/2}}{\Gamma(d/2+1)}$ the volume of the d-sphere with unitary radius (Γ is the Euler gamma function). Let $\mu = \frac{r_2}{r_1} \geq 1$, then $R = \mu^d - 1$ since $r_0 = 0$, which allows finding an explicit formula for the distribution of μ, $f(\mu) = d\mu^{-d-1} \mathbb{1}_{[1,+\infty]}(\mu)$, and thus its cumulative distribution $F(\mu) = (1-\mu^{-d}) \mathbb{1}_{[1,+\infty]}(\mu) \Rightarrow d = -\frac{\log(1-F(\mu))}{\log(\mu)}, \mu \geq 1$. This yields the following algorithm.

1. Compute the pairwise distances for each point in the dataset $i = 1, \ldots, n$
2. For each point i find the two shortest distances $r_i(i)$ and $r_i(2)$.
3. For each point i compute $\mu_i = \frac{r_i(2)}{r_i(1)}$

4. Compute the empirical cumulate $F(\mu_i)$ by sorting the values of $\{\mu_i\}_{i=1}^n$ in an ascending order through a permutation σ, then define $F(\mu_{\sigma(i)}) = \frac{i}{n}$
5. Fit the points of the plane given by coordinates $\{(\log(\mu_i), -\log(1-F(\mu_i))) \mid i = 1, \ldots, n\}$ with a straight line passing through the origin.

Note that the last step of this algorithm is to solve the linear equation $-\log(1 - F(\mu_i)) = d\ \log(\mu_i)$ for any point i of the data set, which by the least squares reduces to $d^* = argmin_{d\in\mathbb{R}} \sum_{i=1}^n \left(\log(1 - F(\mu_i)) + d\ \log(\mu_i)\right)^2$, that is $d^* = -\frac{\sum_{i=1}^n \log(\mu_i)\log(1-F(\mu_i))}{\sum_{i=1}^n \log(\mu_i)^2} = -\frac{\sum_{i=1}^n \log(1-i/n)\log(\mu_{\sigma(i)})}{\sum_{i=1}^n \log(\mu_i)^2}$. For $k = 2$, we have $\hat{m}_{ki} = \log(\mu_i)^{-1}$ for any point i in the data set. This implies, if $-\log(1 - F(\mu_i)) = d^* \log(\mu_i)\ \forall i$, $\hat{m}_k^{-1} d^* = \frac{1}{n}\sum_{i=1}^n \hat{m}_{ki}^{-1} d^* = -\frac{1}{n}\sum_{i=1}^n \log(1 - i/n)$, that is, $\hat{m}_k \propto d^*$ up to the least squares error $\sum_{i=1}^n \left(\log(1 - F(\mu_i))\right)^2 - (d^*)^2 \sum_{i=1}^n \log(\mu_i)^2$. We empirically obtained a Pearson correlation coefficient of $\sim 99.60\%$ between MLE (with $k = 2$) and TWONN (with a *p-value* in the order of $\sim 10^{-7}$). Since we will only be interested in the evolution of the intrinsic dimension (and not its value itself), we could use any of the two methods in our work.

H.2 Results

We use David J.C. & Zoubin (2005)'s Maximum Likelihood Estimation approach with $k = 2$ neighbors to estimate the intrinsic dimensionality of every layer of our model during training, training data, and validation data. For modular addition, we varied the percentage of training data from 0.35 to 0.9 in steps of 0.5, that is, in $\{0.35, 0.40, \ldots, 0.85, 0.90\}$. We have repeated each experiment for each dataset size with 2/3/5 random seeds. We fixed the learning rate to 10^{-4}. See figures 21 and 22 below.

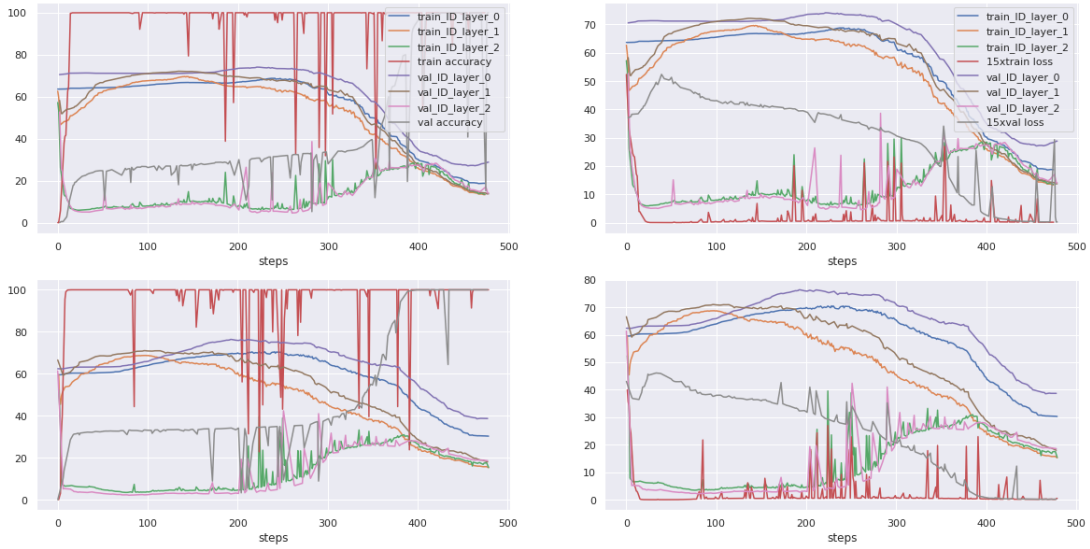
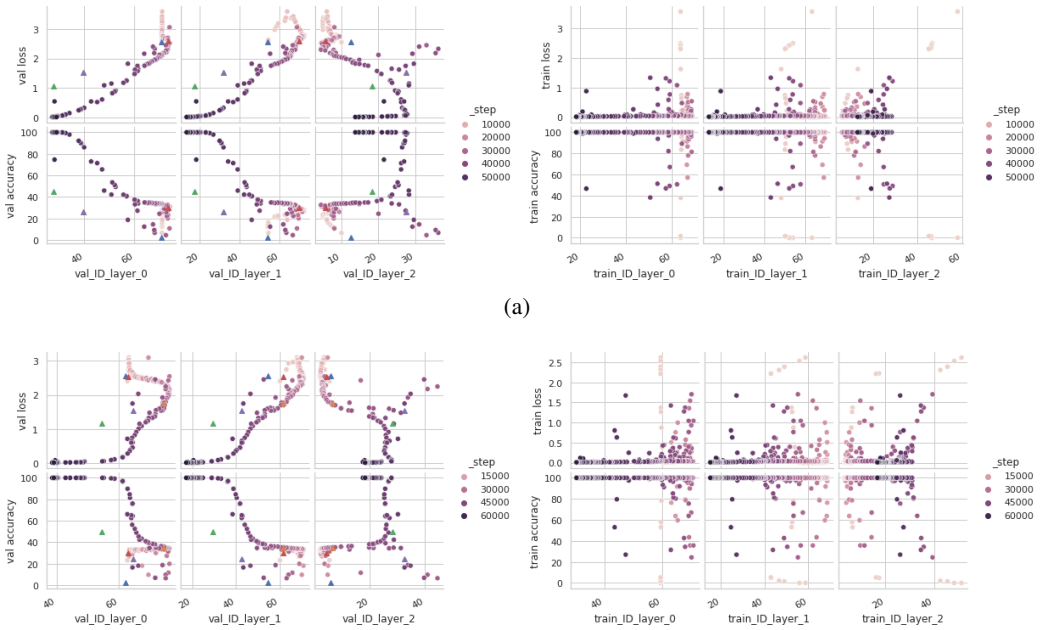


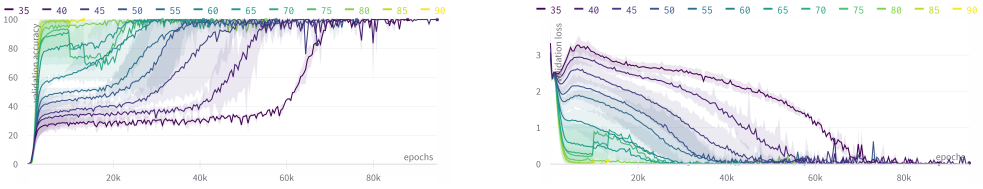
Figure 21: Modular addition, 2 different initial conditions (one per row). Steps is training steps $\times 100$.



(a)

(b)

Figure 22: Training & validation accuracy & loss versus estimated intrinsic dimension in the case of modular addition, 2 different initial conditions (one per column)



(a) Accuracy

(b) Loss

Figure 23: (Grokking) Validation accuracy and loss for modular addition, with different data size. For other operators, the validation accuracy stays at 0 before starting to grow. For addition, it goes up to the percentage of training data at the beginning of training, stays there, before grokking later.

I THEORETICAL MODELS OF GROKKING

We will use the following acronyms to designate our algorithms : sgd for vanilla SGD, momentum for SGD with momentum (Polyak, 1964), rmsprop for the RMSProp algorithm (Hinton, 2012), rprop for the resilient backpropagation algorithm (Riedmiller & Braun, 1993), adam for Adam (Kingma & Ba, 2014) and adamax for Adamax (Kingma & Ba, 2014).

I.1 ROSENBROCK FUNCTION

The vanilla rosenbrok function is given by $g_n(x) = \sum_{i=1}^{n/2} \left[100(x_{2i} - x_{2i-1}^2)^2 + (x_{2i-1} - 1)^2\right]$, with the gradient $\nabla_i g_n(x) = 200(x_i - x_{i-1}^2) \cdot \mathbb{1}_{i \in 2\mathbb{N}} - \left[400x_i(x_{i+1} - x_i^2) - 2(x_i - 1)\right] \cdot \mathbb{1}_{i \in 2\mathbb{N}-1}$, and $x^* \in \{(1, \dots, 1), (-1, 1, \dots, 1)\} \subset \{x, \nabla g_n(x) = 0\}$ [7]. A more involved variant is given by $g_n(x) = \sum_{i=1}^{n-1} \left[100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2\right]$, with the gradient $\nabla_i g_n(x) = 200(x_i - x_{i-1}^2) \cdot \mathbb{1}_{i>1} - \left[400x_i(x_{i+1} - x_i^2) - 2(x_i - 1)\right] \cdot \mathbb{1}_{i<n}$, and $x^* = \{1, \dots, 1)\} \subset \{x, \nabla g_n(x) = 0\}$ [8]. The number of stationary points of this function grows exponentially with dimensionality n, most of which are unstable saddle points (Kok & Sandrock, 2009).

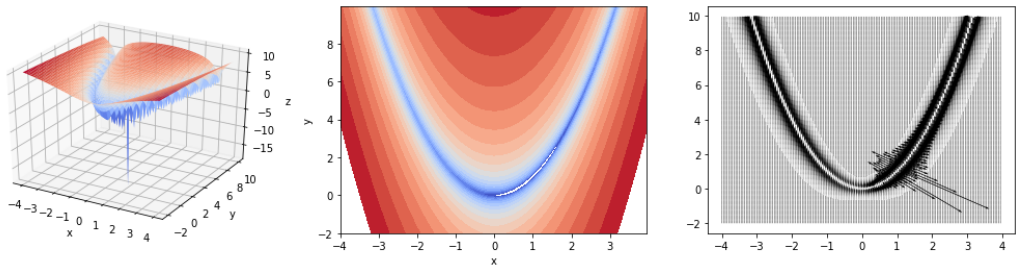


Figure 24: Left) Rosenbrock function in log scale for $n = 2$, Center) Contours Right) Gradient field: note how these vectors are pronounced in norm near the global minimum; that is important to understand how even near this global optimum many optimizers can fail to reach it.

We optimized the Rosenbrock function in a logarithmic scale (to create a ravine, figure 24), for $n = 2$. The function is unimodal, and the global minimum is very sharp and surrounded in the direction of the ravine by many local minima. We fall very quickly into the ravine at the beginning of optimization because the surface is well-conditioned. Then, depending on the learning rate and the optimizer used (as well as the associated hyperparameters), we go down the ravine very slowly. Indeed, without momentum, we do not go directly down to the minimum since the gradient is almost zero along the ravine direction but very large in the perpendicular directions: we go from left to right (perpendicular to the ravine), while going down a little, but very slowly. Moreover, once we are near the minimum, we turn there almost indefinitely. With adaptive gradient, we go down to the minimum very quickly because this direction problem is corrected (due to momentum, left-right ravine perpendicular directions cancel out): if the learning rate is too small, we will also go down very slowly (small gradient in the flat ravine direction). Unlike SGD, here, we always reach the minimum (and stay there). Also, for some learning rates and initializations, there is a double descent (Nakkiran et al., 2020) in error (euclidean distance between the global minimum and the current position at a given time) when landing in the ravine. The methods that succeed in reaching the minimum are rmsprop, rprop, adam, adamax (figures 25 and 26). The method that comes close to it without reaching it is momentum.

[7] When the coordinates range from 0 to $n-1$, $g_n(x) = \sum_{i=0}^{n/2-1} \left[100(x_{2i+1} - x_{2i}^2)^2 + (x_{2i} - 1)^2\right]$ and $\nabla_i g_n(x) = 200(x_i - x_{i-1}^2) \cdot \mathbb{1}_{i \in 2\mathbb{N}+1} - \left[400x_i(x_{i+1} - x_i^2) - 2(x_i - 1)\right] \cdot \mathbb{1}_{i \in 2\mathbb{N}}$.

[8] When the coordinates range from 0 to $n-1$, $g_n(x) = \sum_{i=0}^{n-2} \left[100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2\right]$ and $\nabla_i g_n(x) = 200(x_i - x_{i-1}^2) \cdot \mathbb{1}_{i>0} - \left[400x_i(x_{i+1} - x_i^2) - 2(x_i - 1)\right] \cdot \mathbb{1}_{i<n-1}$.

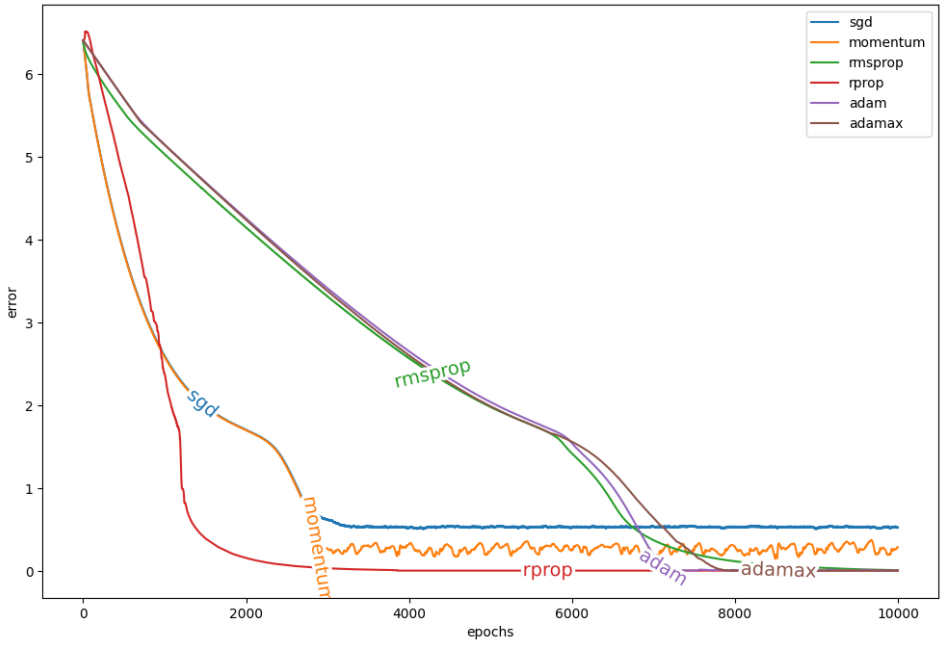


Figure 25: Comparative visualization of the progression error of each algorithm on the rosenbrock function

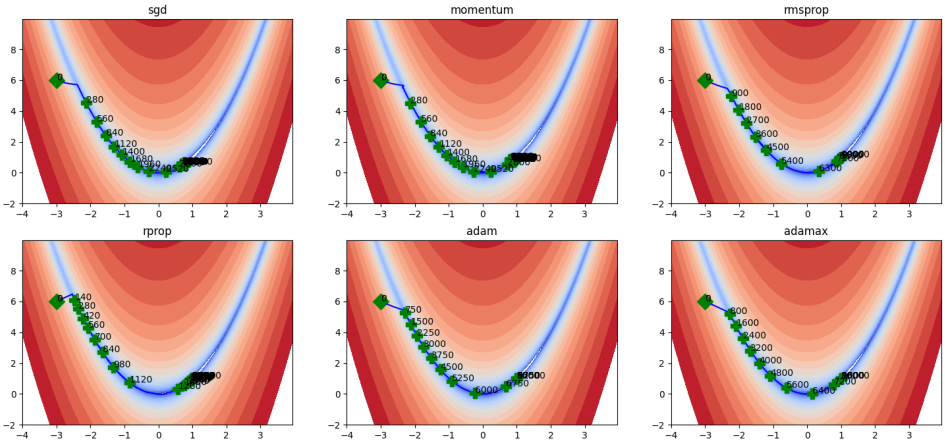


Figure 26: Comparative visualization of the progression of each algorithm on the rosenbrock function

I.2 RASTRIGIN FUNCTION

The rastrigin function is given by $g_n(x) = na + \sum_{i=1}^{n} \left[x_i^2 - a\cos(2\pi x_i)\right] = na + x^T x - a1_n^T \cos(2\pi x)$ with $a \in \mathbb{R}$. Its gradient is $\nabla g_n(x) = 2x + 2\pi a \sin(2\pi x)$, and $x^* = \{0, \ldots, 0)\} \subset \{x, \nabla g_n(x) = 0\}$.

We also optimized the Rastrigin function in a logarithmic scale (to create many local minimums and make the global minimum sharp, figure 27). The function is unimodal, and the global minimum is sharp and surrounded symmetrically by many local minima. At the beginning of optimization, we fall very quickly into the one local minimum. Then, depending on the learning rate and the optimizer used (and the associated hyperparameters), we can move successively from one minimum to another until we reach the global minimum. No method has succeeded in reaching the global minimum (figures 28 and 29).

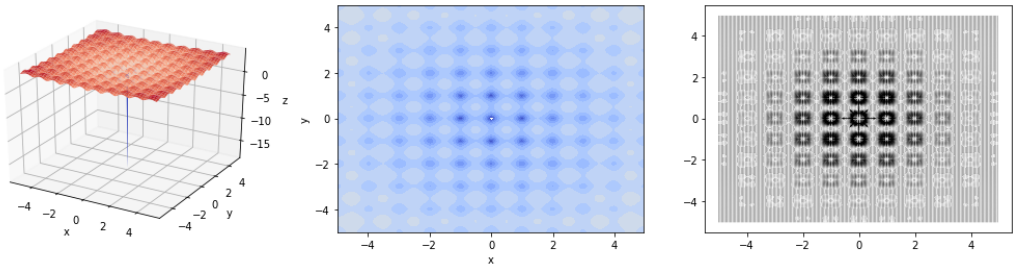


Figure 27: Left) Rastrigin function in log scale ($a = 10, n = 2$), Center) Contours, Right) Gradient field

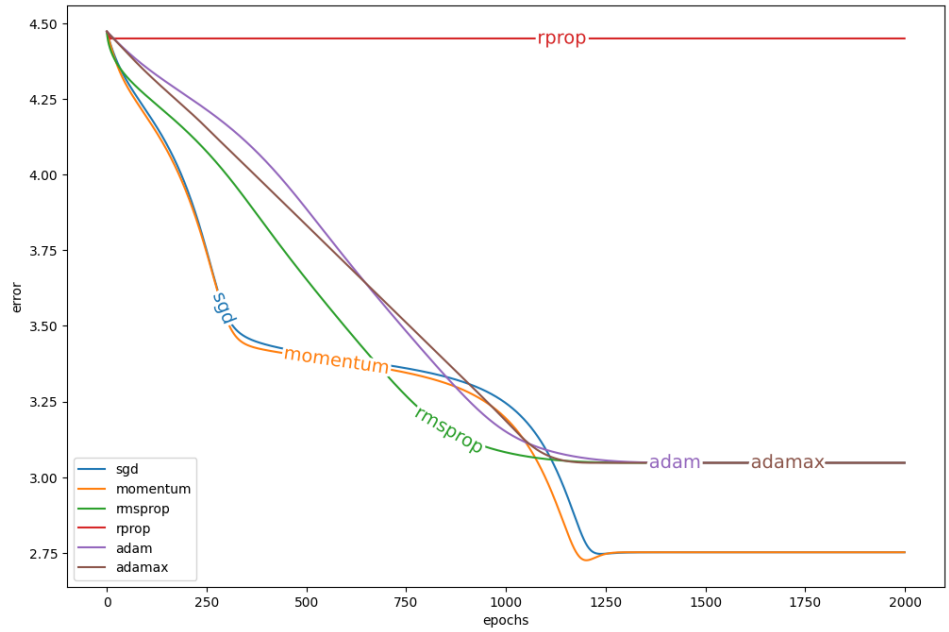


Figure 28: Comparative visualization of the progression error of each algorithm on the Rastrigin function

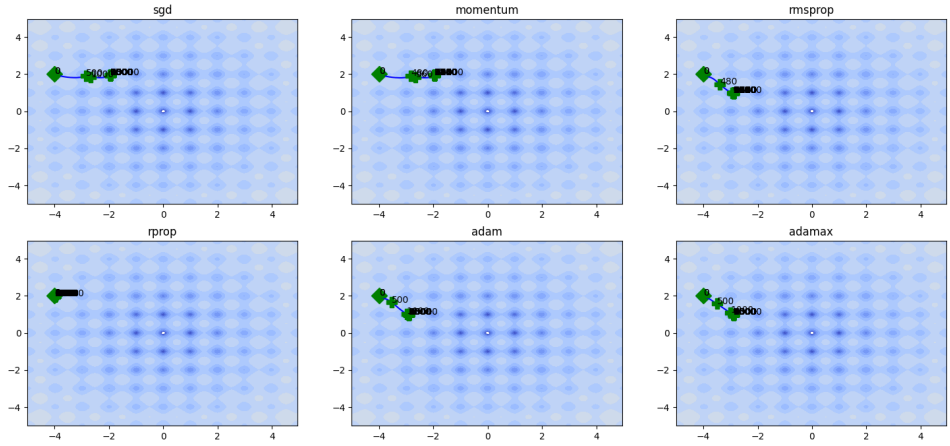


Figure 29: Comparative visualization of the progression of each algorithm on the Rastrigin function