On Convexity and Linear Mode Connectivity in Neural Networks

David Yunis DYUNIS@TTIC.EDU
Kumar Kshitij Patel KKPATEL@TTIC.EDU
Pedro Savarese SAVARESE@TTIC.EDU
Gal Vardi GALVARDI@TTIC.EDU
Karen Livescu KLIVESCU@TTIC.EDU
Matthew Walter MWALTER@TTIC.EDU
Toyota Technological Institute at Chicago, Chicago, IL, USA

Jonathan Frankle JONATHAN@MOSAICML.COM
MosaicML, San Francisco, CA, USA
John A. Paulson School of Engineering and Applied Sciences, Harvard University, Cambridge, MA, USA

Michael Maire MMAIRE@UCHICAGO.EDU
Department of Computer Science, The University of Chicago, Chicago, IL, USA

Abstract

In many cases, neural networks trained with stochastic gradient descent (SGD) that share an early and often small portion of the training trajectory have solutions connected by a linear path of low loss. This phenomenon, called linear mode connectivity (LMC), has been leveraged for pruning and model averaging in large neural network models, but it is not well understood how broadly or why it occurs. LMC suggests that SGD trajectories somehow end up in a *"convex"* region of the loss landscape and stay there. In this work, we confirm that this eventually does happen by finding a high-dimensional convex hull of low loss between the endpoints of several SGD trajectories. But to our surprise, simple measures of convexity do not show any obvious transition at the point when SGD will converge into this region. To understand this convex hull better, we investigate the functional behaviors of its endpoints. We find that only a small number of correct predictions are shared between all endpoints of a hull, and an even smaller number of correct predictions are shared between the hulls, even when the final accuracy is high for every endpoint. Thus, we tie LMC more tightly to convexity, and raise several new questions about the source of this convexity in neural network optimization.

1. Introduction

Neural networks are omnipresent today, but we still do not understand many aspects of their optimization. It is well known that many, often infinite minima exist in the neural network loss landscape. Yet, only some solutions generalize well [14, 23], raising questions about how and why common optimization algorithms find the solutions they do and how one might realize algorithms that find better solutions.

Many researchers have studied the loss landscape of neural networks [4, 6, 13, 14, 17, 19] in the past decade. Linear mode connectivity (LMC), i.e., the existence of linear paths of low loss between solutions of optimization, was likely first observed by Goodfellow et al. [10] between two small networks with the same initialization but different data orders for stochastic gradient descent

(SGD). Frankle et al. [9] subsequently demonstrated that LMC occurs in larger vision models when branch trajectories with different data orders are split from a shared early portion of training with a common data order. They describe the length of the linear interpolation as being "large" in terms of the length of the whole trajectory traveled.

Such LMC behavior would be expected if the loss landscape were convex, as, by definition, any two points in a convex region could be connected by a linear path of lower loss. Even if the entire loss landscape were non-convex, but SGD converged to a convex region at some point in training, LMC would still hold due to local convexity. Given this simple explanation, we are motivated by several questions regarding LMC:

- **Is LMC really indicating that the loss surface is convex between the endpoints?** We find a convex hull of low loss defined by the endpoints of many SGD trajectories and show that its dimension is large. This shows that LMC is not a property of any two endpoints, but the entire loss-landscape around a '*'mode"* indeed looks convex. This gives more credence to the explanation for LMC through convexity.

- **Is the optimization problem approximately convex after the point at which we expect LMC?** We see that the Hessian nearly always has negative eigenvalues but their relative magnitude doesn't change much, and in between stochastic updates the training loss looks convex. Neither of these metrics change significantly throughout training, so even if they indicate optimization is nearly convex, they fail to identify the start of LMC.

- **How are the functions corresponding to linearly connected parameters related?** We probe deeper into the source of LMC. We see that the number of shared correct predictions between convex hulls is not close to the shared correct predictions among endpoints of a hull, even when the training accuracy is high, suggesting that the ability to interpolate between parameters is only loosely related to functional similarity.

2. Convex Mode Connectivity

Linear mode connectivity was defined for a pair of branch trajectories. Instead of a pair, we split into P branches and sample convex combinations of the endpoints to better understand the loss inside the convex hull of these endpoints. Algorithm 1 presents this convex mode connectivity (CMC) procedure, which is a slight modification of the original LMC procedure [9].

In Figure 2, we plot the probability that convex combinations of hull parameters have better training loss hull endpoints, and likewise for random perturbations around hull endpoints. Notably, like LMC, after some point in training, random convex combinations become better than endpoint averages, while random perturbations are almost always worse—strong evidence that LMC describes a convex region in the loss surface. We call the time after this first point in training the LMC regime. We verify these observations in a regression task (SIREN) and a few different networks trained to classify CIFAR-10. For more details on the task selection, see Appendix A.

In order to understand the space that this mode takes up in parameter space, we stack all the parameters $W_1, \dots, W_n$ into a matrix $A = [\text{flatten}(W_1)^\top \cdots \text{flatten}(W_n)^\top]^\top$ and take its SVD. We then plot the singular values of this matrix as a measure of the volume spanned by the endpoints for the different tasks. We see in Figure 3, that for all tasks for which we train sufficient branches, there does not appear to be a sharp cutoff in the magnitude plot corresponding to the number of classes

Algorithm 1: Convex mode procedure

```
Initialize weights with $W_0$, and seed for data order with $z_0$;
Train $k$ steps with algorithm $\mathcal{A}$ to yield $W_k = \mathcal{A}_k(W_0, z_0)$;
Sample $P$ new data order seeds $z_1, \ldots, z_P$;
Train for $T$ steps: $W_T^1 = \mathcal{A}_T(W_k, z_1), \ldots, W_T^P = \mathcal{A}_T(W_k, z_P)$;
for $j = 1, \ldots, J$ do
    Sample $a^j \sim \text{Unif}(\Delta_n)$ and compute $W_{\text{conv}}^j = \sum_{i=1}^P a_i^j W_T^i$;
    Find closest endpoint $W_{\text{end}}^j = \arg\min_{W_T^i} \|W_{\text{conv}}^j - W_T^i\|$;
    Sample random perturbation $\epsilon^j \sim \mathcal{N}(0, I)$;
    Compute perturbed parameter $W_{\text{rand}}^j = W_{\text{end}}^j + \frac{\|W_{\text{end}}^j - W_{\text{conv}}^j\|}{\|\epsilon^j\|}\epsilon^j$;
    Measure $I_{\text{conv}}^j = \mathbb{1}[\mathcal{L}(W_{\text{conv}}^j) < \mathcal{L}(W_{\text{end}}^j)]$ and
      $I_{\text{rand}}^j = \mathbb{1}[\mathcal{L}(W_{\text{rand}}^j) < \mathcal{L}(W_{\text{end}}^j)]$;
end
Compute $P_{\text{conv}} = \frac{1}{J}\sum_j I_{\text{conv}}^j$ and $P_{\text{rand}} = \frac{1}{J}\sum_j I_{\text{rand}}^j$;
```

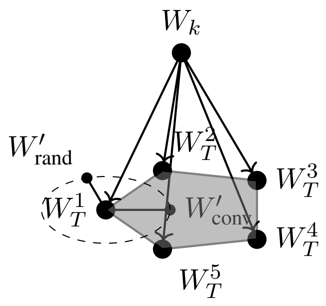


Figure 1: The algorithmic procedure and graphic for measuring convex mode connectivity.

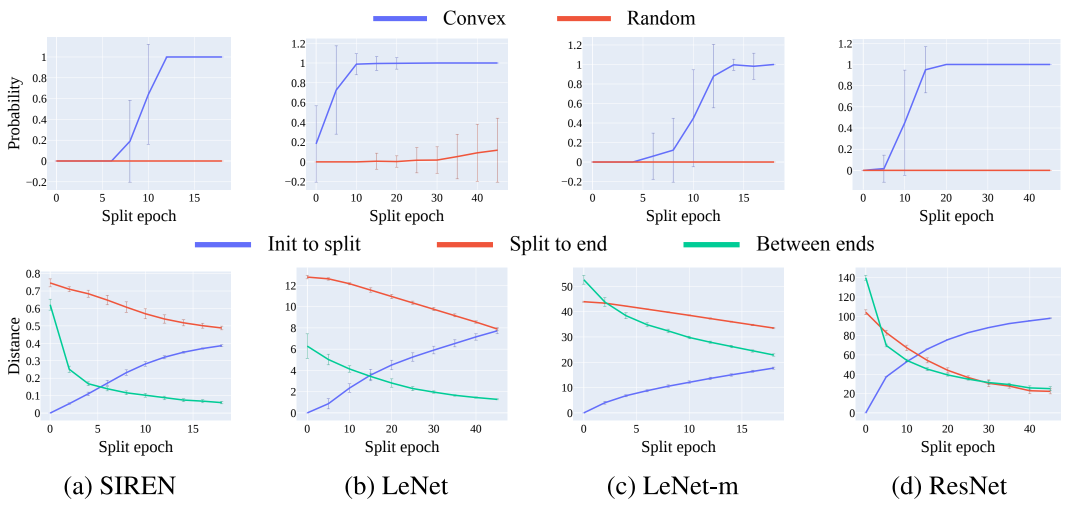
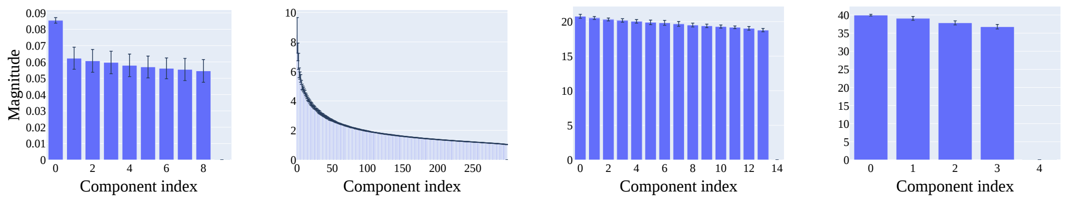


Figure 2: **Top row:** The empirical probability P_{conv} that a sample inside the convex hull (Convex) and a random perturbation P_{rand} around the endpoint (Random) has lower loss than its closest endpoint vs. the split epoch (k in 1) for the many different trajectories. After an initial phase of training, the probability of lower loss inside the hull quickly converges to one, the loss inside the hull is always better than at the endpoints. **Bottom row:** Parameter distances between pairs of points in the CMC procedure. The distance between pairs of endpoints is of the same order of magnitude as other distances for all tasks. In other words, the apparent scale of convexity is similar to that of the total trajectory. Error bars for probabilities are computed by empirical standard deviations $\text{std}(I_{\text{conv}}^j)$ and $\text{std}(I_{\text{rand}}^j)$, hence are not meaningful when values stretch greater than 1 or less than 0.

or the hidden dimension. The only cutoffs occur when there are not enough endpoints to see any higher dimension (P points define an object of dimension at most $P-1$). This tells us that we are



(a) SIREN (b) LeNet (c) LeNet-m (d) ResNet

Figure 3: Singular values for the matrix of stacked endpoints of the mode. We examine endpoints split after the point when the hull center performs better than the endpoints (see Figure 2). For P points in space, the shape they define can have at most $P-1$ dimensions. We see in all cases that the magnitudes do not decay to zero until component $P-1$. In the case that there are K outputs and $P<K$ (ResNet), we see a roughly uniform mode. In cases with $P>K$ (SIREN, LeNet, and LeNet-m), it does not appear that there is any special fragmenting at K. In particular, the LeNet plot is over 300 endpoints, and the tail does not look as if it will decay to zero any time soon. The computational expense of these experiments limits how many endpoints we train.

seeing a non-degenerate convex mode in the loss landscape, and that the particular region does not seem related to the Hessian subspace identified by Gur-Ari et al. [11], or to other simplices of low loss achieved via additional optimization [2, 7].

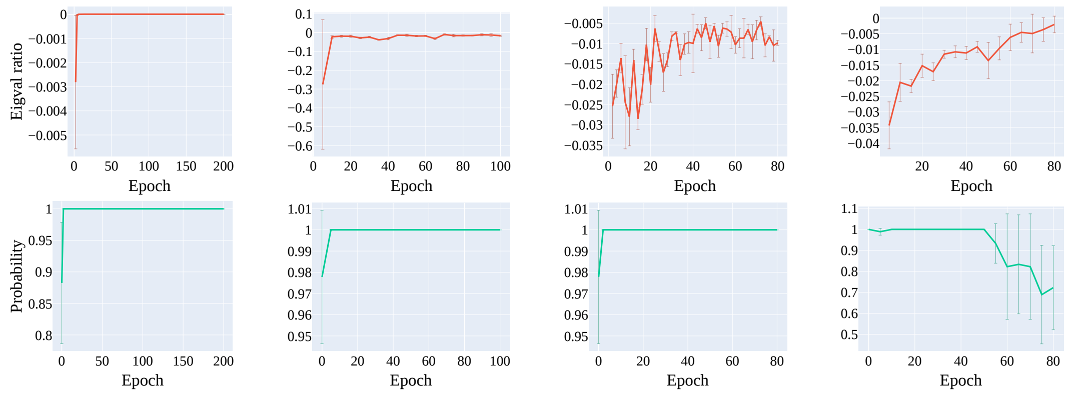
3. Measures of Convexity

One natural explanation for the previously demonstrated convex hulls is that, after some point in training, the trajectory enters a convex "basin" and the rest of the time is spent inside this region. However, previous work has shown that the linear interpolation between points early on the training trajectory and at the end can exhibit large increases in training loss [8, 22], making such an explanation tenuous.

Still, the loss could locally be behaving more *"convexly"* after some early point in training. In order to test this, we examine the bottom part of the Hessian spectrum and the convexity of the training loss between two possible stochastic updates at a point (see Appendix B for details). We see in Figure 4 that the Hessian shows only a small amount of local non-convexity throughout training and that the measurement of convexity between pairs of updates makes training always appear locally convex, making neural network optimization look quite well-behaved. None of the curves show distinguishing features at the LMC point (see Figure 2). Rather than believing that there is a particular point in training at which a difference in SGD updates causes the trajectories to diverge, it seems the eventual convergence to different convex hulls happens much more gradually.

4. Diversity of Endpoints

Figure 2 and prior work has shown that endpoints that exhibit LMC are far apart in parameter space with respect to the total trajectory length [9, 18]. A natural question is whether these distances are deceptive and the endpoints are actually *"functionally similar"*. It is well known that with clever scaling, one can transform a set of parameters into another set, and the distance between the two will be large, while the functions they describe will remain the same. Such scaling was employed



(a) SIREN (b) LeNet (c) LeNet-m (d) ResNet

Figure 4: **Top row:** The ratio of smallest to largest eigenvalue of the Hessian throughout training. This ratio is almost always negative, but quite small, making any local non-convexity appear unimportant. **Bottom row:** Probability of a convex line of loss between a pair of updates. Notice that this probability is close to 1 for the entire optimization, typically before the LMC regime in Figure 2.

by Dinh et al. [4] to make the case that two equivalent functions could have very different sharpness. Another way to preserve functionality is through permutations of rows of weight matrices. Entezari et al. [5] hypothesize that such permutations can map unconnected endpoints into the same mode and Ainsworth et al. [1] show this is true for wide networks, but not the case for standard ResNets, which [3] concurrently confirm. Still, these works do not characterize the behavior within a mode.

Let C_i^j be the set of correct predictions for endpoint W_T^i of a convex hull, where j indexes over different hulls. The set of correct predictions on the training set common to all endpoints is $S_j = \bigcap_i C_i^j$. The set of correct predictions common between hulls is $S^* = \bigcap_j S_j$. Comparing $|S^*|$ to $\mathbb{E}[|S_j|]$ provides a proxy measurement of the similarity between the functions for two different hulls. Figure 5 reveals that there is a substantial gap between $|S^*|$, $\mathbb{E}[|S_j|]$, and $\mathbb{E}[|C_i^j|]$ for functions that do not interpolate the data perfectly, suggesting only a fraction of the predictions within a mode are shared, and only a fraction of those are shared across hulls. We also see that accuracy improves inside the convex hull, so it is not the case that the loss improvement we saw previously is driven by better performance on a common set of points shared across the whole mode. Rather, it seems like there is a large collection of points in the data for which it is possible to flip predictions easily without incurring increases in the loss. Additionally, the poor agreement between hulls suggests that permutations either are not a sufficient explanation for the similarity between hulls, or that they only preserve the function on a relatively small set of predictions.

5. Discussion

In this paper, we presented some dichotomous evidence for local convexity in the neural network loss landscape. We show that trajectories that split after a point early in training define a convex hull of low loss in the loss landscape. On one hand, this might be taken as evidence for the loss landscape being (weakly) convex after this early point in training. But on the other, we cannot detect such a point with simple proxies for convexity. In particular, our measures indicate no increase in the

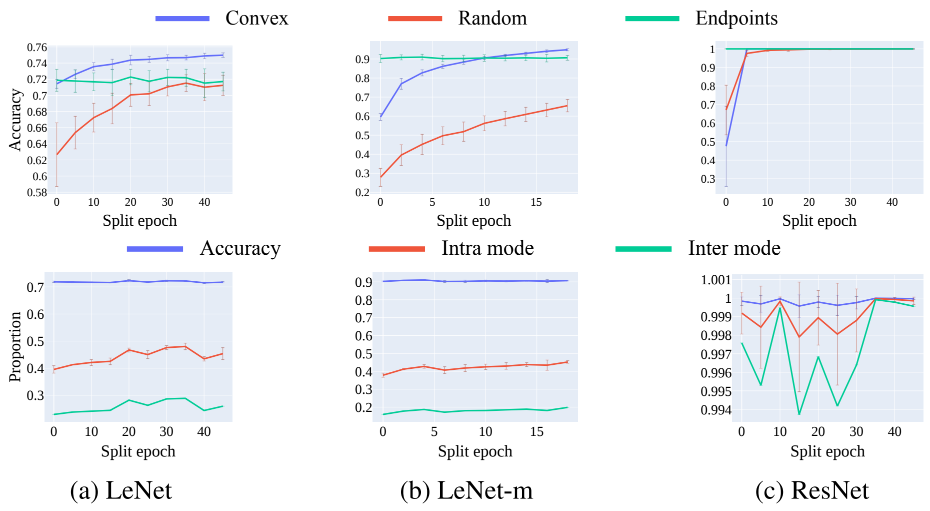


Figure 5: **Top row:** Average accuracies at the endpoints of training (Endpoint), inside the convex mode defined by the endpoints (Convex), and at random directions around endpoints (Random). Notice that after a late enough split, samples in the hull are better than the endpoints. **Bottom row:** Average training accuracy over runs (Accuracy) compared to common points predicted correctly within a mode (Intra mode) and common points predicted correctly between hulls (Inter mode). We see that for models with an imperfect training accuracy, intra mode agreement is substantially lower than accuracy, and inter mode agreement is even lower. This suggests that endpoints of a mode are not similar functions.

convexity of the function before or after the split point which gives rise to LMC. Failure to measure a difference could be due to sparse sampling of our metrics, but it might also indicate that neural network optimization is much more convex than we might expect. Probing the mode, it seems the convexity in training loss and error is not due to a large subset of shared predictions on the training set, but rather a diverse set of functions that are amenable to interpolation.

Prior work has shown that, on the scale of the optimization trajectory, the neural network loss landscape does not appear convex [8, 17, 22], yet a large convex hull appears at the end of training. Given that hulls only agree on a small subset of predictions, yet interpolation is possible, studying the behavior of the convex hull while training on the subset of the data corresponding to predictions that are and are not shared may provide more insight.

There may also be a relationship between LMC and neural collapse [20], a recently identified tendency in classifiers to collapse the output feature space to a set of class means over the course of training. It could be the case that interpolating within a given convex hull corresponds to minimal disturbances to class means, but interpolating between convex hulls corresponds to changing the location of class means in feature space. If such a relationship existed, neural collapse might provide an early metric for detecting the first iteration after which a convex hull will exist, which would allow for parallelization of training without the extra computational cost needed to identify the LMC regime.

Acknowledgments

We thank Nati Srebro, Mikhail Belkin, Suriya Gunasekar, David McAllester, Behnam Neyshabur, Tengyu Ma, Robert Nowak, and Shane Settle for helpful discussions. This material is based upon work supported by the National Science Foundation Graduate Research Fellowship Program under Grant No. 1754881. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

[1] Samuel K. Ainsworth, Jonathan Hayase, and Siddhartha Srinivasa. Git re-basin: Merging models modulo permutation symmetries. *arXiv preprint arXiv:2209.04836*, 2022.

[2] Gregory Benton, Wesley Maddox, Sanae Lotfi, and Andrew Gordon Wilson. Loss surface simplexes for mode connecting volumes and fast ensembling. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 769–779, 2021.

[3] Frederik Benzing, Simon Schug, Robert Meier, Johannes von Oswald, Yassir Akram, Nicolas Zucchet, Laurence Aitchison, and Angelika Steger. Random initialisations performing above chance and how to find them. *arXiv preprint arXiv:2209.07509*, 2022.

[4] Laurent Dinh, Razvan Pascanu, Samy Bengio, and Yoshua Bengio. Sharp minima can generalize for deep nets. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 1019–1028, 2017.

[5] Rahim Entezari, Hanie Sedghi, Olga Saukh, and Behnam Neyshabur. The role of permutation invariance in linear mode connectivity of neural networks. *arXiv preprint arXiv:2110.06296*, 2021.

[6] Stanislav Fort and Stanislaw Jastrzebski. Large scale structure of neural network loss landscapes. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

[7] Stanislav Fort, Huiyi Hu, and Balaji Lakshminarayanan. Deep ensembles: A loss landscape perspective. *arXiv preprint arXiv:1912.02757*, 2019.

[8] Jonathan Frankle. Revisiting "Qualitatively characterizing neural network optimization problems". *arXiv preprint arXiv:2012.06898*, 2020.

[9] Jonathan Frankle, Gintare Karolina Dziugaite, Daniel Roy, and Michael Carbin. Linear mode connectivity and the lottery ticket hypothesis. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 3259–3269, 2020.

[10] Ian J Goodfellow, Oriol Vinyals, and Andrew M Saxe. Qualitatively characterizing neural network optimization problems. *arXiv preprint arXiv:1412.6544*, 2014.

[11] Guy Gur-Ari, Daniel A Roberts, and Ethan Dyer. Gradient descent happens in a tiny subspace. *arXiv preprint arXiv:1812.04754*, 2018.

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[13] Yiding Jiang, Behnam Neyshabur, Hossein Mobahi, Dilip Krishnan, and Samy Bengio. Fantastic generalization measures and where to find them. *arXiv preprint arXiv:1912.02178*, 2019.

[14] 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. *arXiv preprint arXiv:1609.04836*, 2016.

[15] Alex Krizhevsky. Learning multiple layers of features from tiny images. Master's thesis, University of Toronto, 2009.

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

[17] Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein. Visualizing the loss landscape of neural nets. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

[18] Vaishnavh Nagarajan and J Zico Kolter. Uniform convergence may be unable to explain generalization in deep learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

[19] Behnam Neyshabur, Hanie Sedghi, and Chiyuan Zhang. What is being transferred in transfer learning? In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 512–523, 2020.

[20] Vardan Papyan, XY Han, and David L Donoho. Prevalence of neural collapse during the terminal phase of deep learning training. *Proceedings of the National Academy of Sciences*, 117(40):24652–24663, 2020.

[21] Vincent Sitzmann, Julien Martel, Alexander Bergman, David Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 7462–7473, 2020.

[22] Tiffany J Vlaar and Jonathan Frankle. What can linear interpolation of neural network loss landscapes tell us? In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 22325–22341, 2022.

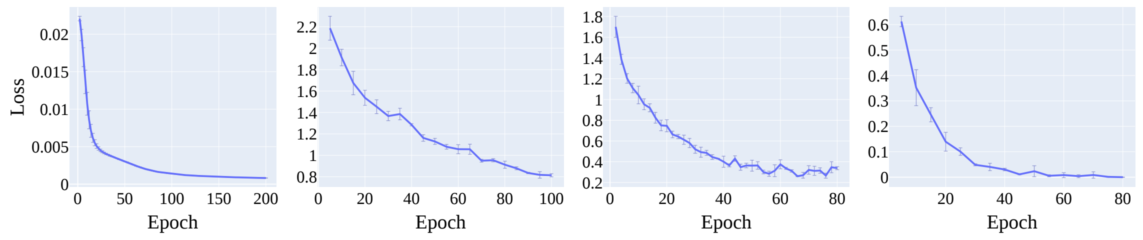
[23] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning (still) requires rethinking generalization. *Communications of the ACM*, 64(3):107–115, 2021.

Appendix A. Task Details

SIREN [21] is a fully-connected network with sine activations. It was originally designed to overcome the low-frequency bias in radiance fields. Here it is tuned to reconstruct an audio waveform, where the input is the timestep, and the output is the waveform value. We choose to study it because all prior experiments on LMC were conducted with ReLU activations, and we wished to observe a system that was not piece-wise linear. For our other models, we use LeNet-5 [16], and ResNet-20 [12] trained on CIFAR-10 [15]. These combinations are some of the simplest systems for which we do not see LMC at initialization [1, 9, 18]. For the sake of simplicity, all learning rates are constant in our experiments. Because the SIREN task does not include generalization, we stop training when the training loss has converged. In the case of our other models, we tune hyper-parameters and stop training when the validation error has converged, which is typically the desired criterion in practice. For hyperparameters, see Table 1. For full training loss curves, see Figure 6.

Table 1: Summary of tasks studied in this paper.

Abbrev.	**Task**	**Dataset**	**Model**	**Optimizer**	**LR**	**Batch size**
SIREN	Regress.	Bach Waveform	5-layer SIREN	Adam	1e-5	8192
LeNet	Classif.	CIFAR-10	LeNet-5	SGD	1e-2	128
LeNet-m	Classif.	CIFAR-10	LeNet-5	SGD+mom. (0.9)	1e-2	128
ResNet	Classif.	CIFAR-10	ResNet-20	SGD+mom. (0.9)	0.1	128



(a) SIREN (b) LeNet (c) LeNet-m (d) ResNet

Figure 6: Training loss curves for all tasks. When cross-referencing with Figure 2, we see that the first split point that leads to a convex hull is quite early, long before the loss has converged, and often before it has even halved.

Appendix B. Metric Calculation Details

In order to calculate Hessian eigenvalues, we use a power iteration method with Hessian vector products, like in Li et al. [17]. To measure local convexity between a pair of stochastic updates from parameters W, we take compute two stochastic updates ΔW_1 and ΔW_2, and measure the gap

$$g = \mathcal{L}\left(\frac{1}{2}(W + \Delta W_1) + \frac{1}{2}(W + \Delta W_2)\right) - \left(\frac{1}{2}\mathcal{L}(W + \Delta W_1) + \frac{1}{2}\mathcal{L}(W + \Delta W_2)\right).$$

We then compute the probability that this gap is negative over many samples. This gives us a metric for the non-convexity that a single update induces, which we can measure at different points throughout training.