On Monotonic Linear Interpolation of Neural Network Parameters

James Lucas JLUCAS@CS.TORONTO.EDU
Juhan Bae JBAE@CS.TORONTO.EDU
Michael R. Zhang MICHAEL@CS.TORONTO.EDU
Jimmy Ba JBA@CS.TORONTO.EDU
Richard Zemel ZEMEL@CS.TORONTO.EDU
Roger Grosse RGROSSE@CS.TORONTO.EDU
Vector Institute; University of Toronto

Abstract

Linearly interpolating between initial neural network parameters and converged parameters after training with SGD typically leads to a monotonic decrease in the training objective. This Monotonic Linear Interpolation (MLI) property, first observed by Goodfellow et al. [11], persists in spite of the non-convex objectives and highly non-linear training dynamics of neural networks. Extending on this work, we show that this property holds under varying network architectures, optimizers, and learning problems. We evaluate several possible hypotheses for this property that, to our knowledge, have not yet been explored. Additionally, we show that networks violating this property can be produced systematically, by forcing the weights to move far from initialization. The MLI property raises important questions about the loss landscape geometry of neural nets and highlights the need to further study its global properties.

1. Introduction

A simple and lightweight method to probe neural network loss landscapes is to linearly interpolate between the parameters at initialization and the parameters found after training. More formally, consider a neural network with parameters $\boldsymbol{\theta} \in \mathbb{R}^d$ trained with respect to loss function $\mathcal{L}\colon \mathbb{R}^d \to \mathbb{R}$ on a training dataset $\mathcal{D}$. Let the neural network be initialized with some parameters $\boldsymbol{\theta}_0$. Using a gradient descent optimizer, the network converges to some final parameters $\boldsymbol{\theta}_T$. A linear path is then constructed between these two parameters denoted as $\boldsymbol{\theta}_\alpha = (1-\alpha)\boldsymbol{\theta}_0 + \alpha\boldsymbol{\theta}_T$. A surprising phenomenon, first observed by Goodfellow et al. [11], is that the function $g(\alpha) = \mathcal{L}(\boldsymbol{\theta}_\alpha)$ monotonically decreases on the interval $\alpha \in [0, 1]$. We call this effect the *Monotonic Linear Interpolation (MLI) property* of neural networks.

Goodfellow et al. [11] conclude their study of the MLI property by stating that "the reason for the success of SGD on a wide variety of tasks is now clear: these tasks are relatively easy to optimize". Since the publication of their research, there has been significant developments both in terms of the neural network architectures that we train today [13, 32] and our theoretical understanding of them [1, 4, 5, 7, 17]. Therefore, with a wider lens that addresses these developments, we believe that a secondary investigation of this phenomenon is necessary.

In this work, we provide an expanded study of the MLI property. In particular, we extend the original evaluation of Goodfellow et al. [11] by testing the MLI property over a range of network architectures, optimizers, and training mechanisms (e.g. batch normalization [15]). We demonstrate

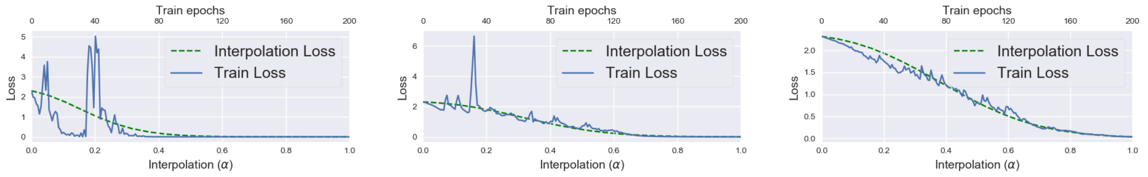


Figure 1: Linear interpolations (green) for neural networks trained on varying dataset sizes (30, 300, 3000 from left-to-right), with loss during training overlaid (blue). Even when the training dynamics are unstable and highly non-linear, the interpolation produces a smooth monotonic curve.

that neural networks retain the MLI property even when trained on data with arbitrary amounts of label corruption, for both under and over-parameterized networks, and in other challenging settings. However, despite the prevalence of the MLI property, we are able to systematically produce neural networks that violate it.

We propose one explanation for the absence of the MLI property that we found repeatedly through our experiments: neural networks that travel far in parameter space during optimization (in Euclidean distance) do not have the MLI property. Conversely, if the weights remain close to the initial weights after training, then in all settings that we explored, the MLI property is likely to hold.

Recent theoretical work has shown that training can succeed within a low dimensional affine subspace of the parameters about initialization [5, 12]. Moreover, the persistence of the MLI property suggests that in practice, a one-dimensional affine subspace suffices. However, this is not the only mechanism in which neural network training can succeed — the solutions that violate the MLI property can have good generalization capability and are found without significant training difficulty. This hints at a potential disconnect between our existing local theory and practical neural network training, and suggests the need to further study global geometry of the neural network loss landscape.

1.1. The Monotonic Linear Interpolation Property

The Monotonic Linear Interpolation (MLI) property was first identified by Goodfellow et al. [11]. It states that when a network is randomly initialized and then trained to convergence, the linear path connecting the initialization and the solution is monotonically decreasing in the training loss. Specifically, we say that a network has the MLI property if, for all $\alpha_1, \alpha_2 \in [0, 1]$ with $\alpha_1 < \alpha_2$,

$$\mathcal{L}(\boldsymbol{\theta}_{\alpha_1}) \geq \mathcal{L}(\boldsymbol{\theta}_{\alpha_2}), \text{ where } \boldsymbol{\theta}_\alpha = \boldsymbol{\theta}_0 + \alpha(\boldsymbol{\theta}_T - \boldsymbol{\theta}_0). \tag{1}$$

Astonishingly, the MLI property holds for a majority of neural network architectures, optimizers and learning problems that we observe in practice.

In the remainder of this paper, we seek to answer the question: what causes the MLI property and when does it fail to hold?

2. Exploring & Explaining the MLI Property

In this section, we present several hypotheses that may provide explanations for monotonic parameter interpolation and then conduct a series of experiments to test their validity. In each experiment, we evaluate MLI only on the training set but note that the same results typically hold on the test set

Table 1: Proportion of networks trained with Adam [18] that satisfy the MLI property, over increasing learning rate. While networks trained with higher learning rates break the MLI property more frequently, but the relationship with test performance is less clear.

LEARNING RATE	PROPORTION MONOTONIC (%)	TEST ACCURACY (%)
	MNIST	
0.00001	96.00	95.44
0.0001	89.39	96.97
0.001	62.00	96.64
0.01	50.75	90.19
	CIFAR-10	
0.00001	100.00	41.29
0.0001	85.25	45.67
0.001	78.57	45.08
0.01	56.00	31.01

as well. Throughout, we discretize the interval $[0, 1]$ using 50 uniform steps to evaluate the MLI property. Some additional hypotheses and empirical results are presented in Appendix B.

2.1. Problem Difficulty

The hypothesis put forward by Goodfellow et al. [11] to explain monotonic interpolation is that the learning problem itself is sufficiently easy to optimize. Recent research has shown that over-parameterized networks learn faster and, in some cases, have more linear learning dynamics [22]. Intuitively, the least well-behaved network regime is the *critical parameterization* regime, when the network has just enough parameters to fit the data well.

Our experiments on parameter complexity explored this question on two fronts. First, we used a fixed network size and varied the number of data points in the dataset. Second, we used a fixed dataset size and varied the number of hidden units in a network of fixed depth. For the experiments using varying dataset size, we trained fully-connected networks on the FashionMNIST dataset [34] using SGD with a learning rate of 0.1. The networks had a single hidden layer with 1000 hidden units, and we varied the dataset size from 10 up to the full size 60000. Figure 1 shows the linear interpolation trained on varying dataset sizes. We observed that even when the training dynamics are unstable and highly non-linear, the interpolation is still monotonically decreasing. We did not find any trained networks that exhibited non-monotonic interpolations (on the training loss). Similarly, when varying the size of the hidden layer from 1 unit up to 10000, there were few runs that violated the MLI property. The non-monotonic interpolations were produced by the larger networks when using larger learning rates (≥ 1). Overall, we did not find significant evidence that network architecture or problem difficulty alone affect the MLI property.

2.2. Distance Moved During Optimization

Goodfellow et al. [11] only investigated the MLI property when using standard SGD. Here, we extend our evaluation to include adaptive optimizers such as RMSProp and Adam [18]. We evaluated

Table 2: Proportion of networks trained with SGD that satisfy the MLI property, over increasing learning rate.

LEARNING RATE	PROPORTION MONOTONIC (%)	TEST ACCURACY (%)
	MNIST	
0.0001	100.00	95.31
0.001	100.00	96.60
0.01	100.00	97.10
0.1	39.21	96.27
	CIFAR-10	
0.0001	100.00	45.52
0.001	100.00	44.22
0.01	100.00	45.72

the effects of using adaptive gradient optimizers under different initialization schemes, learning rates, and depth/width of fully connected networks. The results on the ratio of monotonicity breaks are presented in Table 1 and Table 2. Throughout the experiments, the MLI property is violated more frequently when using adaptive optimizers — and particularly at higher learning rates.

We believe the reason for this is that the parameters travel a shorter distance from the initialization when SGD is used in place of adaptive methods [1] or when a smaller learning rate is used. If the Hessian of the network at initialization is nearly positive semi-definite, an assumption that can be made rigorous and theoretically validated [5, 6], then the interpolation will be monotonically decreasing as the function is locally convex about the initialization. In Figure 2, we show that the distance travelled in parameter space has a strong positive correlation with the MLI property.

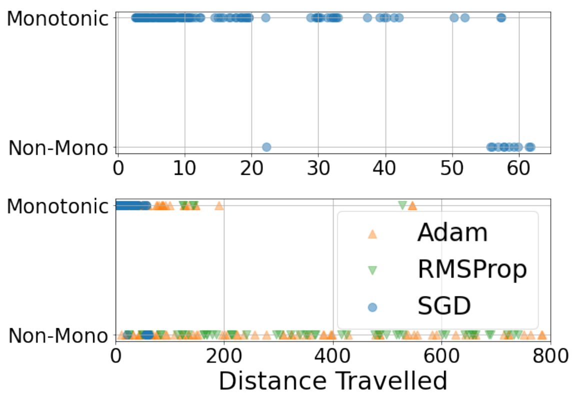


Figure 2: We plot distance traveled against monotonicity for SGD (top), and for all optimizers (bottom). The MLI property breaks more frequently when the parameters travel further from initialization.

Moreover, a smaller step size decreases the variance of the final iterates, an argument we detail in Appendix D.1. Specifically, we show that, within a noisy quadratic model, non-monotonicity occurs half the time with fixed learning rates. Moreover, the severity (and thus ease of detection) of the non-monotonicty increases with increasing learning rate.

2.3. Modern Architectures and Training Techniques

Finally, we present experiments on larger-scale neural network architectures. We trained VGG19 [31] (without batch normalization) and ResNet-32 [13] (with batch normalization) on SGD with initial learning rate of 0.1 and Adam with learning rate of 0.01 followed by the decay with factor of 2 every 60 epochs.

The initialization-solution interpolations are shown in Figure 3. The results show that the MLI property may not persist on modern architectures. We hypothesize that these violations may be due to the fact that (1) parameters may travel far in larger-scale architectures, especially those trained with adaptive optimizers, and (2) ambiguity in how modern training techniques affect the interpolation between initial and final checkpoints (see next paragraph). However, even when the MLI property breaks, the networks performs well on both training and testing datasets, and it is unclear how the monotonic interpolated path relates to the ease of training. These results suggest that the original conclusion in Goodfellow et al. [11], that the MLI property is due to ease of training, may not be sufficient. Further study is necessary to understand the loss landscape geometry globally.

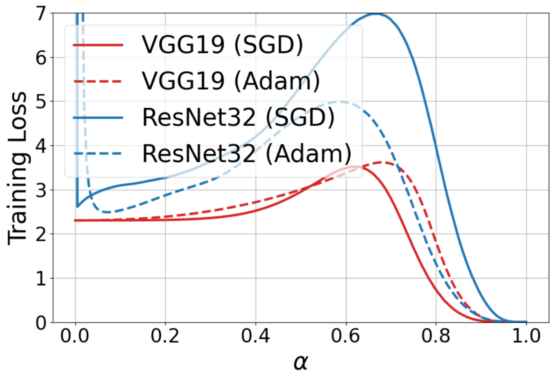


Figure 3: A linearly interpolated path between initialization and final checkpoints with larger scale architectures and different optimizers.

Batch normalization [15] standardizes the activation of the layer, making the network invariant to the scale of the weights [2]. Batch normalization is now considered a standard technique in training modern neural networks, but was invented after the work of Goodfellow et al. [11]. We empirically analyzed the effect of batch normalization under different initialization schemes, learning rates, and depth/width of the fully connected networks on the MNIST [21] and CIFAR-10 [19] datasets. We found that the MLI property breaks more frequently when batch normalization is used; 83% and 58% of the time with and without batch normalization (respectively), averaged over all other settings. We show the comparison of the interpolations on a network with 8 hidden layers in Figure 4 with differing learning rates.

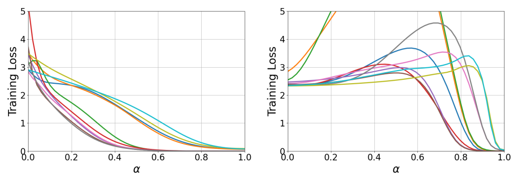


Figure 4: A comparison of interpolations **(left)** without batch normalization and **(right)** with batch normalization.

3. Conclusion

The MLI property, when the linear path connecting initial and final parameters after training decreases monotonically, holds surprisingly often. In this paper, we extended previous studies of MLI by exploring variations in datasets, architecture, and optimization. While the MLI property generally persisted across these settings, we observed that final parameters that are far from initialization systematically lead to violations of the MLI property. This observation points to exciting connections with analysis of the loss landscape geometry of neural networks.

References

[1] Shun-ichi Amari, Jimmy Ba, Roger Grosse, Xuechen Li, Atsushi Nitanda, Taiji Suzuki, Denny Wu, and Ji Xu. When does preconditioning help or hurt generalization? *arXiv preprint*

arXiv:2006.10732, 2020.

[2] Sanjeev Arora, Zhiyuan Li, and Kaifeng Lyu. Theoretical analysis of auto rate-tuning by batch normalization. *arXiv preprint arXiv:1812.03981*, 2018.

[3] Laurent Dinh, Razvan Pascanu, Samy Bengio, and Yoshua Bengio. Sharp minima can generalize for deep nets. *arXiv preprint arXiv:1703.04933*, 2017.

[4] Felix Draxler, Kambis Veschgini, Manfred Salmhofer, and Fred A Hamprecht. Essentially no barriers in neural network energy landscape. *arXiv preprint arXiv:1803.00885*, 2018.

[5] Stanislav Fort and Surya Ganguli. Emergent properties of the local geometry of neural loss landscapes. *arXiv preprint arXiv:1910.05929*, 2019.

[6] Stanislav Fort and Adam Scherlis. The goldilocks zone: Towards better understanding of neural network loss landscapes. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 3574–3581, 2019.

[7] Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. *arXiv preprint arXiv:1803.03635*, 2018.

[8] Jonathan Frankle, Gintare Karolina Dziugaite, Daniel M Roy, and Michael Carbin. Linear mode connectivity and the lottery ticket hypothesis. *arXiv preprint arXiv:1912.05671*, 2019.

[9] Jonathan Frankle, Gintare Karolina Dziugaite, Daniel M Roy, and Michael Carbin. Stabilizing the lottery ticket hypothesis. *arXiv preprint arXiv:1903.01611*, 2019.

[10] Timur Garipov, Pavel Izmailov, Dmitrii Podoprikhin, Dmitry P Vetrov, and Andrew G Wilson. Loss surfaces, mode connectivity, and fast ensembling of dnns. In *Advances in Neural Information Processing Systems*, pages 8789–8798, 2018.

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

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

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[14] Sepp Hochreiter and Jürgen Schmidhuber. Flat minima. *Neural Computation*, 9(1):1–42, 1997.

[15] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

[16] Pavel Izmailov, Dmitrii Podoprikhin, Timur Garipov, Dmitry Vetrov, and Andrew Gordon Wilson. Averaging weights leads to wider optima and better generalization. *arXiv preprint arXiv:1803.05407*, 2018.

[17] Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. In *Advances in neural information processing systems*, pages 8571–8580, 2018.

[18] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[19] Alex Krizhevsky et al. Learning multiple layers of features from tiny images. 2009.

[20] Rohith Kuditipudi, Xiang Wang, Holden Lee, Yi Zhang, Zhiyuan Li, Wei Hu, Rong Ge, and Sanjeev Arora. Explaining landscape connectivity of low-cost solutions for multilayer nets. In *Advances in Neural Information Processing Systems*, pages 14574–14583, 2019.

[21] Yann LeCun, Corinna Cortes, and CJ Burges. Mnist handwritten digit database. *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist*, 2, 2010.

[22] Jaehoon Lee, Lechao Xiao, Samuel S Schoenholz, Yasaman Bahri, Jascha Sohl-Dickstein, and Jeffrey Pennington. Wide neural networks of any depth evolve as linear models under gradient descent. *arXiv preprint arXiv:1902.06720*, 2019.

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

[24] Chunyuan Li, Heerad Farkhoor, Rosanne Liu, and Jason Yosinski. Measuring the intrinsic dimension of objective landscapes. *arXiv preprint arXiv:1804.08838*, 2018.

[25] Eran Malach, Gilad Yehudai, Shai Shalev-Shwartz, and Ohad Shamir. Proving the lottery ticket hypothesis: Pruning is all you need. *arXiv preprint arXiv:2002.00585*, 2020.

[26] Quynh Nguyen. On connected sublevel sets in deep learning. *arXiv preprint arXiv:1901.07417*, 2019.

[27] Boris T Polyak and Anatoli B Juditsky. Acceleration of stochastic approximation by averaging. *SIAM journal on control and optimization*, 30(4):838–855, 1992.

[28] David Ruppert. Efficient estimations from a slowly convergent robbins-monro process. Technical report, Cornell University Operations Research and Industrial Engineering, 1988.

[29] Tom Schaul, Sixin Zhang, and Yann LeCun. No more pesky learning rates. In *International Conference on Machine Learning*, pages 343–351, 2013.

[30] Alexander Shevchenko and Marco Mondelli. Landscape connectivity and dropout stability of sgd solutions for over-parameterized neural networks. *arXiv preprint arXiv:1912.10095*, 2019.

[31] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[32] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

[33] Yuhuai Wu, Mengye Ren, Renjie Liao, and Roger Grosse. Understanding short-horizon bias in stochastic meta-optimization. *arXiv preprint arXiv:1803.02021*, 2018.

[34] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

[35] Guodong Zhang, Lala Li, Zachary Nado, James Martens, Sushant Sachdeva, George Dahl, Chris Shallue, and Roger B Grosse. Which algorithmic choices matter at which batch sizes? insights from a noisy quadratic model. In *Advances in Neural Information Processing Systems*, pages 8196–8207, 2019.

[36] Michael Zhang, James Lucas, Geoffrey E Hinton, and Jimmy Ba. Lookahead optimizer: k steps forward, 1 step back. In *Advances in Neural Information Processing Systems*, pages 9593–9604, 2019.

Appendix A. Related Work

Monotonic linear interpolation. Our work follows up on the initial observation made by Goodfellow et al. [11]. In their original study, a variety of architectures, activation functions, and training objectives were evaluated. In addition to analyzing the phenomena empirically, Goodfellow et al. [11] provide a qualitative analysis of MLI in a 1D deep linear model. They show that MLI property will typically hold, despite negative curvature about initialization and disconnected manifolds of global optima.

Linear connectivity. This work is inspired by empirical and theoretical advancements in understanding the loss landscape of neural networks. Much of this recent work has involved characterizing mode connectivity of neural networks. In general, linear paths between modes cross regions of high loss [11]. However, Draxler et al. [4], Garipov et al. [10] show that local minima found by stochastic gradient descent (SGD) can be connected via piecewise linear paths. Frankle et al. [8] further show that linearly connected solutions may be found if networks share the same initialization. Kuditipudi et al. [20] posit *dropout stability* as one possible explanation for this phenomena. A solution is dropout stable if the network can maintain a low loss, in spite of half the neurons in each layer being removed (intuitively, using half its capacity). They show that local minima that are dropout stable are piecewise linearly connected. They also construct an example where there is *no* mode connectivity, demonstrating that the behavior is not a general property of neural networks. Shevchenko and Mondelli [30] extend this result and show the loss landscape becomes increasingly connected and more dropout stable with the increased depth of the network. Finally, Nguyen [26] shows that every sublevel set of an overparameterized network is connected, implying that all global minima are connected.

Note that the MLI property is distinct from these mode connectivity properties, where paths are drawn between different final solutions instead of initialization-solution pairs. So far as we are aware, no prior work has explored connections between the MLI property and mode connectivity.

Lottery tickets. The Lottery Ticket Hypothesis [7] states that a dense randomly initialized network contains a sub-network that can match the test error of the full network when trained in isolation. Frankle et al. [8] showed connections between the lottery ticket hypothesis and linear mode connectivity used to measure stability to SGD noise. Follow-up works have improved stability of the lottery ticket hypothesis for large networks [9] and proved that the lottery ticket hypothesis holds for large networks even without training the subnetwork [25].

Loss landscape geometry. Recent analysis argues that there exists a small subspace at initialization in which the network converges [5, 12]. Li et al. [24] showed that some of these spaces can be identified by learning in a random affine subspace. Fort and Scherlis [6] showed that the success of these random spaces related to the *Goldilocks zone* that depends on the Hessian at initialization. In a loose sense, the MLI can be considered a special case of these results, wherein a 1D space is sufficient for training to succeed.

It has long been argued that flatter minima lead to better generalization [14] with some caveats [3]. Recent work has shown that (full-batch) gradient descent with large learning rates is able to find flatter minima, by overcoming regions of initial high curvature [23]. Intuitively, gradient descent breaks out of one locally convex region of the space and into another — suggesting that a barrier in the loss landscape should have been surpassed. However, in this paper, we show that such barriers do not typically appear when interpolating between initialization and the final solution, unless the new parameters are sufficiently far from initialization.

Optimization algorithms. There has been empirical success in applying weights averaging and linear interpolation during training. While Polyak and Juditsky [27], Ruppert [28] investigated averaging weights in convex optimization, these techniques have empirically proven useful in the training of deep neural networks [16, 36]. Stochastic Weight Averaging [16] performs tail averaging at the end of the final epochs. Similarly, the Lookahead optimizer [36] applies interpolation in weight space throughout training. In the case of Lookahead optimizer, it was shown that the linear interpolation stabilizes the optimization process and the interpolated checkpoints are typically linearly connected throughout optimization. Amari et al. [1] recently showed that for linear regression, natural gradient descent travels more in parameter space, as measured by Euclidian distance, compared to gradient descent. This agrees with our hypothesis that networks which move far violate MLI.

Appendix B. Additional Explanations and Experiments

In our exploration of the MLI property we performed many additional experiments. Generally, we found that turning common knobs of neural network training did not have a significant effect on the likelihood of a network satisfying the MLI property. For example, varying activation functions, loss functions, batch size, regularization, and different forms of initialization had no significant effect on the MLI property. In this section, we present a few of the more interesting additional experiments that we performed.

B.1. Δ-Monotonicity

Goodfellow et al. [11] provided primarily qualitative evidence of MLI, by plotting $\mathcal{L}(\boldsymbol{\theta}_\alpha)$ with discretizations of $[0, 1]$ using varying resolutions. While we define MLI as a continuous property, we would like quantitatively define the notion of monotonicity. We propose a simple metric by which to measure monotonicity of initialization-solution interpolations.

Definition 1 *(Δ-monotonicity) Consider a linear parameter interpolation parameterized by $\alpha, \boldsymbol{\theta}_1, \boldsymbol{\theta}_2$, with corresponding (differentiable) loss function $\mathcal{L}$. The path is Δ-monotonic if for all pairs of points $\alpha_1 < \alpha_2 \in (0, 1)$ with $\frac{\partial \mathcal{L}}{\partial \alpha_1} > 0$ and $\mathcal{L}(\alpha_2) = \mathcal{L}(\alpha_1)$ and $\mathcal{L}(\alpha') > \mathcal{L}(\alpha_1)$ for all $\alpha' \in (\alpha_1, \alpha_2)$, then $\max_{\alpha' \in (\alpha_1, \alpha_2)} \mathcal{L}(\alpha') < \Delta$.*

Intuitively, the above definition states that any concave bump in the loss over the interpolation path should have a height upper-bounded by Δ. This metric is useful as we can approximate it well numerically by stepping along the interpolation path in fixed intervals to find α_1 and α_2. We are typically interested in the largest $\Delta \geq 0$ for which the initialization-solution interpolation is Δ-monotonic.

In Figure 5 we display the distance from initialization against the largest Δ achieving Δ-monotonicity. Each point represents a neural network that achieved a minimum training loss of at most 0.01, with varying architectures, optimizers, datasize, and learning rate. All points with $\max \Delta > 0$ are coloured in orange, and $\max \Delta = 0$ are blue.

B.2. Problem Difficulty

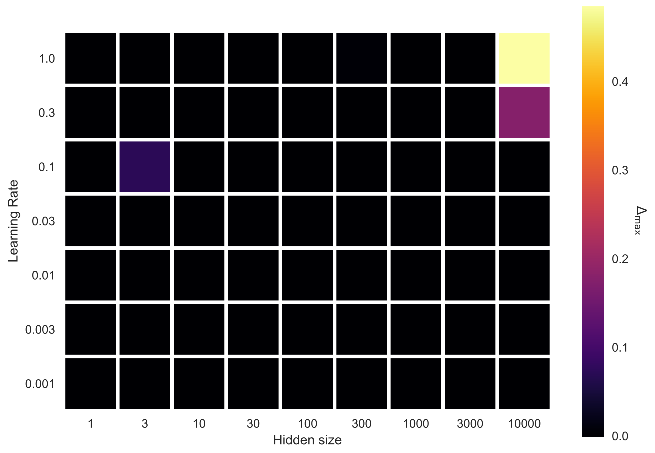
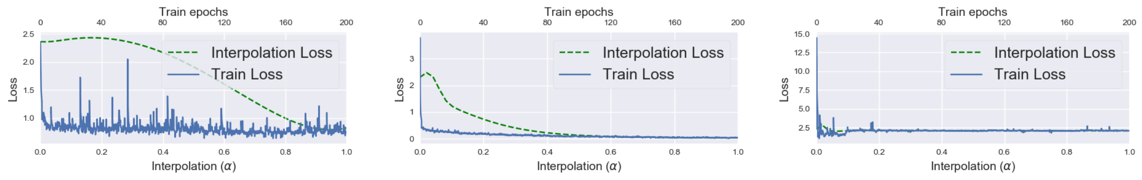


Figure 6: A heatmap of $\max \Delta$ as a function of the learning rate and hidden size.



(a) Learning rate: 0.1, Hidden size: 3

(b) Learning rate: 0.3, Hidden size: 10000

(b) Learning rate: 1.0, Hidden size: 10000

Figure 7: The three models which violate the MLI property in the varying data size experiments. Two of the models (a and c) fail to converge to a local minima. The final model (b) has a large learning rate with 10000 hidden units.

Additional results on varying data size Here we include some additional figures summarizing the results on varying data size. In Figure 6 we display a heatmap of $\max \Delta$ as a function of the learning rate and hidden size of fully-connected neural networks trained on FashionMNIST. There are only three models which violate the MLI property, these are detailed further in Figure 7.

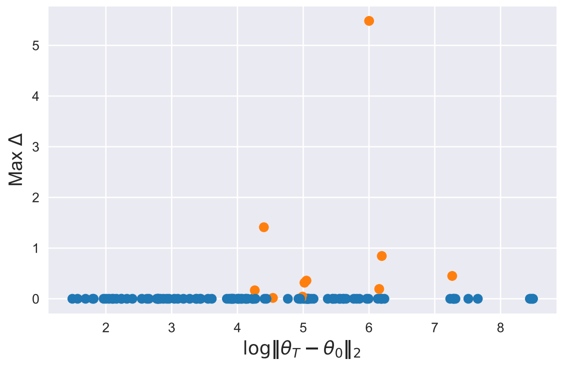


Figure 5: The maximum Δ-monotonicity achieved against distance from initialization.

MLI vs. label corruption. When the dataset is sufficiently simple, the learning problem is easy and SGD consistently finds solutions with the MLI property. To explore this hypothesis, we trained neural networks with label corruption. We trained a neural network with two hidden layers each with 1024 units. We used SGD with a learning rate of 0.1 and momentum of 0.9. The labels were corrupted by uniformly sampling labels for some proportion of the data points. We varied the label corruption from 0% to 100% in 2.5% intervals. We varied the proportion of label corruption from 0% up to 100%. At all levels of label corruption, the MLI property persisted. One possible explanation for this result, follows from the fact that logit gradients cluster together by logit index — even for inputs belonging to different true classes [5]. This provides an explanation for gradient descent exploring a low dimensional subspace relative to the parameter space. Therefore, corrupting the label will not disrupt this clustering at initialization and, as empirically verified, is unlikely to prevent the MLI property from holding.

B.3. Learning Dynamics

Breakouts break MLI property. Lewkowycz et al. [23] observed a region of critical large learning rates wherein gradient descent breaks out of high-curvature regions at initialization, and explores regions of high-loss before settling in a low-loss region with lower curvature. We might expect that such trajectories lead to initialization-solution pairs that do not satisfy the MLI property. However, in Figure 1, we observed several runs where SGD overcame large barriers but the MLI property holds. We consider our findings here to be inconclusive in general, and hope to explore this further in future work.

Appendix C. Experiment Details

Throughout, we evaluated interpolations using 50 equally spaced steps along the line connecting initialization and final solution.

Problem difficulty experiments. For the experiments evaluating problem difficulty (parameter complexity and label corruption), we trained fully-connected networks on the FashionMNIST dataset. In all cases, the networks used ReLU activations and were trained with batch sizes of at most 512 (depending on dataset size), and for 200 epochs. Learning rates were fixed throughout training.

When varying the dataset size, we trained models on random subsets of FashionMNIST with sizes in the set $\{10, 30, 100, 300, 1000, 3000, 10000, 30000, 60000\}$. We also evaluated over learning rates in the set $\{0.03, 0.1, 0.3, 1.0\}$. When varying the number of hidden units, we evaluated networks with sizes in $\{10000, 3000, 1000, 300, 100, 30, 10, 3, 1\}$ and learning rates in the set $\{0.03, 0.1, 0.3, 1.0, 3.0\}$. For experiments with varying levels of label corruption. We trained a fully-connected network with 2 hidden-layers each of width 1024.

Distance moved experiments. We conducted our MNIST & CIFAR-10 experiments with fully connected networks that had $\{1, 4, 8, 16\}$ hidden layers with $\{128, 1024, 4096\}$ hidden units in each layer. We used the activation function of ReLU, tanh, and sigmoid. We further tried both with batch normalization (with batch sizes of 128 datapoints) and without batch normalization. We used Xavier and Kaiming initializations with 3 random seeds. We trained all neural networks for 100 epochs.

Appendix D. Infinitely Wide Networks

Limit of infinite width We can lean on prior analysis from Lee et al. [22] to show that infinitely wide fully-connected networks have the MLI property. In this setting, we assume that the fully-connected network has the following layer sizes $d \to n \to \ldots n \to k$, with $n \to \infty$. We also assume that the loss function is given by $\mathcal{L}(\mathbf{x}, \mathbf{y}; \boldsymbol{\theta}) = 0.5\|f_{\boldsymbol{\theta}}(\mathbf{x}) - \mathbf{y}\|^2$.

First, we have the following result, first proved in Lee et al. [22], [some additional assumptions are needed which are excluded here for now],

Lemma 2 *(INFORMAL - Theorem G.1 [22]) For all $\delta > 0$, there exists $T \in \mathbb{N}$ (and some other constants, $C(\delta)$), such that with probability at least $1 - \delta$,*

$$\|\boldsymbol{\theta}_0 - \boldsymbol{\theta}_t\|_2 \leq C(\delta) n^{-1/2},$$

for all $t > T$, and learning rate $O(1/n)$.

From this, we can show that in the limit of infinite width, gradient descent with a suitably small learning rate finds a solution which is linearly connected to the initialization.

Intuitively, this result holds as in a region near a minimum the objective is locally convex. As the width of the network grows, the minimum found by gradient descent becomes arbitrarily close to initialization and thus the linear interpolation is acting over a convex function.

More formally, writing $\Delta\boldsymbol{\theta} = \boldsymbol{\theta}_t - \boldsymbol{\theta}_0$, we have,

$$\begin{aligned}
\frac{d}{d\alpha}\mathcal{L}(\mathbf{x}, \mathbf{y}; \theta_0 + \alpha(\theta_t - \theta_0)) &= 0.5\frac{d}{d\alpha}[\|f(\mathbf{x};\theta_0) + \alpha J(\mathbf{x};\boldsymbol{\theta}_0)\Delta\boldsymbol{\theta} - \mathbf{y} + O(\|\Delta\boldsymbol{\theta}\|^2)\|], \\
&= \alpha\Delta\boldsymbol{\theta}^T J(\mathbf{x};\boldsymbol{\theta}_0)(f(\mathbf{x};\theta_0) - \mathbf{y}) + O(\|\Delta\boldsymbol{\theta}\|^2), \\
&= \alpha\Delta\boldsymbol{\theta}^T [J(\mathbf{x};\boldsymbol{\theta}_0)(f(\mathbf{x};\theta_0) - \mathbf{y}) + O(\|\Delta\boldsymbol{\theta}\|^2), \\
&= \alpha\Delta\boldsymbol{\theta}^T \frac{\partial}{\partial\theta}\mathcal{L}(\mathbf{x}, \mathbf{y}; \boldsymbol{\theta}_0) + O(\|\Delta\boldsymbol{\theta}\|^2),
\end{aligned}$$

where $J(\boldsymbol{\theta})$ denotes the Jacobian matrix of the network (with respect to the weights). If $\Delta\boldsymbol{\theta}$ is anti-aligned with initial gradient of the loss, then this term is negative for all α and sufficiently small $\Delta\boldsymbol{\theta}$ (equivalently, large n).

Now, consider the Taylor expansion,

$$\mathcal{L}(\mathbf{x}, \mathbf{y}; \boldsymbol{\theta}_0) = \mathcal{L}(\mathbf{x}, \mathbf{y}; \boldsymbol{\theta}^*) + 0.5\Delta\boldsymbol{\theta}^\top \nabla^2\mathcal{L}\Delta\boldsymbol{\theta} + O(\|\Delta\boldsymbol{\theta}\|^3)$$

As $n \to \infty$, $\boldsymbol{\theta}^* \to \boldsymbol{\theta}_0$ by Lemma 2, and thus as $\boldsymbol{\theta}^*$ is a minimum the function is convex locally. Therefore, $\frac{d}{d\alpha}\mathcal{L}(\mathbf{x}, \mathbf{y}; \theta_0 + \alpha(\theta_t - \theta_0)) < 0$, using the convexity definition.

D.1. A Noisy Quadratic Model

The noisy quadratic model (NQM) [29, 33, 35] serves as a useful guide for understanding the effects of stochasticity in asymptotic neural network training. Indeed, Zhang et al. [35] demonstrate that the NQM makes predictions that are aligned with experimental results on deep neural networks. Using this model, we can provide an explanation for one possible cause of non-monotonicity: an inflection point of the interpolation curve with positive second derivative close to $\alpha = 1$. Let our loss function be as follows:

$$\mathcal{L}(\boldsymbol{\theta}) = \frac{1}{2}\boldsymbol{\theta}^\top \mathbf{K}\boldsymbol{\theta}, \tag{2}$$

where $\boldsymbol{\theta} \in \mathbb{R}^d$ and $\mathbf{K} \in \mathbb{R}^{d\times d}$. The optimization algorithm receives stochastic gradients $\mathbf{K}\boldsymbol{\theta} + \mathbf{c}$, where $\mathbf{c} \sim \mathcal{N}(\mathbf{0}, \mathbf{K})$. Consider the iterates $\{\boldsymbol{\theta}_i\}_{i=0}^T$ from the gradient descent. With a sufficiently small learning rate, the expected value of the iterate converges i.e. $\lim_{t\to\infty} \mathbb{E}[\mathcal{L}(\boldsymbol{\theta}_t)] = 0$.

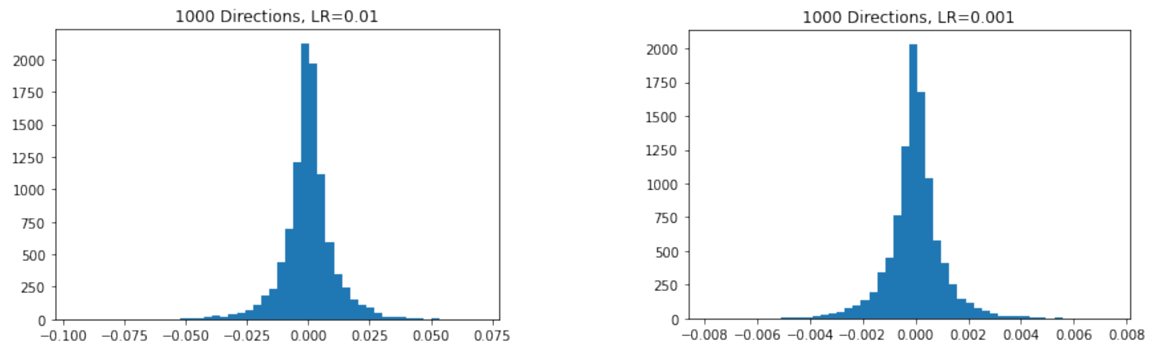


Figure 8: For smaller learning rates, the standard deviation of the distribution goes down. Hence the probability that $P((\mathbf{c}_2 - \mathbf{c}_1)^\top \mathbf{K}\mathbf{c}_2) < \epsilon$ for some small ϵ goes up (indicating non-monotonicity that is hard to detect.

Also consider interpolating between arbitrary $\mathbf{c}_1$ and $\mathbf{c}_2$. The loss along the interpolation direction is $\mathcal{L}(\mathbf{c}_1 + \alpha(\mathbf{c}_2 - \mathbf{c}_1))$. We compute the derivative with respect to α:

$$\frac{\partial \mathcal{L}}{\partial \alpha}(\mathbf{c}_1 + \alpha(\mathbf{c}_2 - \mathbf{c}_1)) = \frac{\partial}{\partial \alpha}\left[\frac{1}{2}(\mathbf{c}_1 + \alpha(\mathbf{c}_2 - \mathbf{c}_1))^\top \mathbf{K}(\mathbf{c}_1 + \alpha(\mathbf{c}_2 - \mathbf{c}_1))\right] \tag{3}$$

$$= (\mathbf{c}_2 - \mathbf{c}_1)^\top \mathbf{K}(\mathbf{c}_1 + \alpha(\mathbf{c}_2 - \mathbf{c}_1)) \tag{4}$$

Hence, the loss is monotonically decreasing if, for all $\alpha \in [0, 1]$,

$$(\mathbf{c}_2 - \mathbf{c}_1)^\top \mathbf{K}(\mathbf{c}_1 + \alpha(\mathbf{c}_2 - \mathbf{c}_1)) < 0 \tag{5}$$

In the one dimension case, this equation is saying that interpolation is non-monotonic when $\mathbf{c}_1$ and $\mathbf{c}_2$ are on the opposite side of the minima. More generally, note that because $\frac{\partial \mathcal{L}}{\partial \alpha}$ is linear in α, the interpolation is monotonically decreasing if and only if both of these conditions at the endpoints are satisfied:

$$(\mathbf{c}_2 - \mathbf{c}_1)^\top \mathbf{K}\mathbf{c}_1 < 0 \tag{6}$$

$$(\mathbf{c}_2 - \mathbf{c}_1)^\top \mathbf{K}\mathbf{c}_2 < 0 \tag{7}$$

These two conditions correspond to a negative derivative with respect to α at $\mathbf{c}_1$ and $\mathbf{c}_2$. Since we choose a learning rate so that the loss decreases in expectation (and hence the derivative is anti-aligned with $\mathbf{c}_2 - \mathbf{c}_1$ at initialization), it suffices to check just the second condition:

$$(\mathbf{c}_2 - \mathbf{c}_1)^\top \mathbf{K}\mathbf{c}_2 < 0 \tag{8}$$

As in Zhang et al. [35], we use $\mathbf{c}_1 = \boldsymbol{\theta}_0 \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. As $t \to \infty$, the point $c_2 \sim \mathcal{N}(\mathbf{0}, \eta\mathbf{K})$, where η is the final learning rate and the random variable comes from the noise in the gradient. This is *almost* a symmetric distribution about 0, so the probability we have monotonic interpolation is roughly $\frac{1}{2}$. This is empirically verified in Figure 8. A smaller learning rate means that the distribution of $(\mathbf{c}_2 - \mathbf{c}_1)^\top \mathbf{K}\mathbf{c}_2$ has less variance. Because we discretize α when we check for MLI, we have $P((\mathbf{c}_2 - \mathbf{c}_1)^\top \mathbf{K}\mathbf{c}_2) < \epsilon$ increase as the learning rate decreases for some small ϵ.