A Modern Take on the Bias-Variance Tradeoff in Neural Networks

Anonymous Authors[1]

Abstract

Recent empirical results on over-parameterized deep networks are marked by a striking absence of the classic U-shaped test error curve: test error keeps decreasing in wider networks. Researchers are actively working on bridging this discrepancy by proposing better complexity measures. Instead, we directly measure prediction bias and variance for four classification and regression tasks on modern deep networks. We find that *both* bias *and* variance can decrease as the number of parameters grows. Qualitatively, the phenomenon persists over a number of gradient-based optimizers. To better understand the role of optimization, we decompose the total variance into variance due to training set sampling and variance due to initialization. Variance due to initialization is significant in the under-parameterized regime. In the over-parameterized regime, total variance is much lower and dominated by variance due to sampling. We provide theoretical analysis in a simplified setting that is consistent with our empirical findings.

1. Introduction

Despite a few notable exceptions, such as boosting (Schapire, 1990; Freund, 1995; Bühlmann & Yu, 2003), the dogma in machine learning has been: "the price to pay for achieving low bias is high variance" (Geman et al., 1992). This balance between underfitting (high bias) and overfitting (high variance) is commonly known as the *bias-variance tradeoff* (Fig. 1). Statistical learning theory (Vapnik, 1998) successfully predicts this U-shaped test error curve for a number of classic machine learning models by identifying a notion of model capacity, understood as the main parameter controlling this tradeoff. Complex (high capacity) models achieve low prediction bias at the expense of high variance. In their landmark work that highlighted this dilemma, Geman et al. (1992) suggest that bias decreases and variance increases with network size.

However, there is a growing amount of empirical evidence that *wider* networks generalize *better* than their smaller counterparts (Neyshabur et al., 2015; Zagoruyko & Komodakis, 2016; Novak et al., 2018; Lee et al., 2018; Belkin et al., 2018; Spigler et al., 2018; Liang et al., 2017; Canziani et al., 2016). In those cases the U-shaped test error curve is not observed. Researchers have identified classic measures of complexity as a culprit. The idea is that, once we have identified the right complexity measure, we will again be able to observe this fundamental tradeoff.

We bypass this important, ongoing discussion by measuring prediction bias and variance directly—something that has not been done in related literature since Geman et al. (1992), to the best of our knowledge. These measurements allow us to reason directly about the existence of a tradeoff with respect to network width. We find evidence that *both* bias *and* variance can decrease at the same time as network width increases in common classification and regression settings with deep networks.

We observe this qualitative behavior with a number of gradient-based optimizers. In order to get a closer look at the role of optimization and sampling, we propose a simple decomposition of total prediction variance. We use the law of total variance to get a term that corresponds to variance due to training set sampling and another that corresponds to variance due to initialization. Variance due to initialization is significant in the under-parameterized regime and monotonically decreases with width in the over-parameterized regime. There, total variance is much lower and dominated by variance due to sampling (Fig. 2).

We provide theoretical analysis, consistent with our empirical findings, in simplified analysis settings: i) prediction variance does not grow arbitrarily in linear models; ii) variance due to initialization diminishes in deep networks under strong assumptions.

1.1. Related Work

In concurrent work, Spigler et al. (2018); Belkin et al. (2018) point out that generalization error decreases with capacity in the over-parameterized setting, with a sharp transition between the under-parameterized and the over-parameterized settings. While this transition can also be seen as the early hump in variance we observe in some of

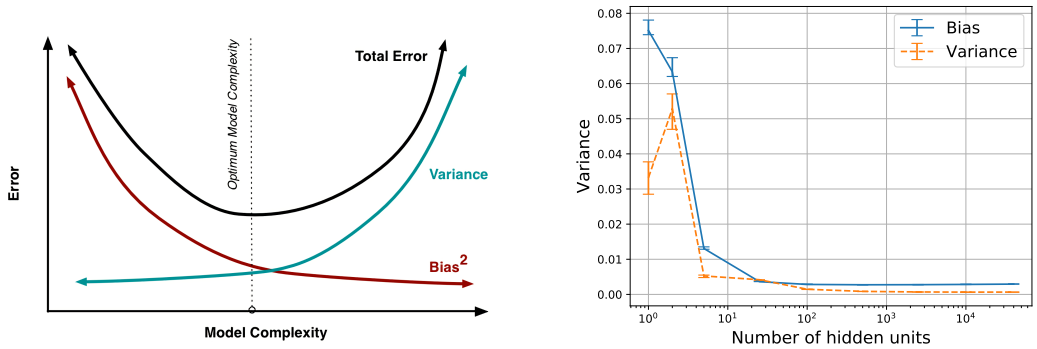


Figure 1: On the left is an illustration of the common intuition for the bias-variance tradeoff (Fortmann-Roe, 2012). We find that variance decreases along with bias when increasing network width (right). These results seem to contradict the traditional intuition.

our graphs, we mostly focus on the over-parameterized setting. Additionally, our work is unique in that we explicitly analyze and experimentally measure the quantities of bias and variance.

2. Preliminaries

2.1. Set-up

We consider the typical supervised learning task of predicting an output $y \in \mathcal{Y}$ from an input $x \in \mathcal{X}$, where the pairs (x, y) are drawn from some unknown joint distribution, $\mathcal{D}$. The learning problem consists of inferring a function $h_S : \mathcal{X} \to \mathcal{Y}$ from a finite training dataset S of m i.i.d. samples from $\mathcal{D}$. The quality of a predictor h can quantified by the expected error, $\mathcal{E}(h) = \mathbb{E}_{(x,y)\sim\mathcal{D}}\, \ell(h(x), y)$, for some loss function $\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$.

In this paper, predictors h_θ are parameterized by the weights $\theta \in \mathbb{R}^N$ of deep neural networks. We will consider the average performance over possible training sets (denoted by the random variable S) of size m. This is the same quantity Geman et al. (1992) consider. While S is the only random quantity focused on in traditional bias-variance decomposition, we also focus on randomness coming from optimization. We denote the random variable for optimization randomness (e.g. initialization) by I.[1]

Formally, given a fixed training set S and fixed optimization randomness I, the learning algorithm $\mathcal{A}$ produces $\theta = \mathcal{A}(S, I)$. Randomness in initialization translates to randomness in $\mathcal{A}(S, \cdot)$. Given a fixed training set, we encode the randomness due to I in a conditional distribution $p(\theta|S)$; marginalizing over the training set S of size m gives a marginal distribution $p(\theta) = \mathbb{E}_S p(\theta|S)$ on the weights learned by $\mathcal{A}$ from m samples. In this context, the average performance for the learning algorithm using training sets of size m can be expressed in the following ways:

$$\mathbb{E}_{\theta\sim p}\mathcal{E}(h_\theta) = \mathbb{E}_S\mathbb{E}_{\theta\sim p(\cdot|S)}\mathcal{E}(h_\theta) = \mathbb{E}_S\mathbb{E}_I\mathcal{E}(h_\theta) \quad (1)$$

2.2. Bias-Variance Decomposition

We briefly recall the standard bias-variance decomposition in the case of squared-loss. We work in the context of classification, where each class $k \in \{1 \cdots K\}$ is represented by a one-hot vector in $\mathbb{R}^K$. The predictor outputs a score or probability vector in $\mathbb{R}^K$. In this context, the average performance in Eq. (1) decomposes into three sources of error (Geman et al., 1992):

$$\mathcal{E}_{\text{noise}} + \mathcal{E}_{\text{bias}} + \mathcal{E}_{\text{variance}} \quad (2)$$

The first term is an intrinsic error term independent of the predictor; the second is a bias term:

$$\mathcal{E}_{\text{noise}} = \mathbb{E}_{(x,y)}\left[\|y - \bar{y}(x)\|^2\right],$$

$$\mathcal{E}_{\text{bias}} = \mathbb{E}_x\left[\|\mathbb{E}_\theta[h_\theta(x)] - \bar{y}(x)\|^2\right],$$

where $\bar{y}(x)$ denotes the expectation $\mathbb{E}[y|x]$ of y given x. The third term is the expected variance of the output predictions:

$$\mathcal{E}_{\text{variance}} = \mathbb{E}_x \text{Var}(h_\theta(x)),$$

$$\text{Var}(h_\theta(x)) = \mathbb{E}_\theta\left[\|(h_\theta(x) - \mathbb{E}_\theta[h_\theta(x)]\|^2\right],$$

where the expectation over θ can be done as in Eq. (1). Finally, in the set-up of Section 2.1, the sources of variance are the choice of training set S and the choice of initialization I (encoded into the conditional $p(\cdot|S)$). By the law of

[1] We focus on randomness from initialization and do not focus on randomness from stochastic mini-batching because we found the phenomenon of decreasing variance with width persists when using *batch* gradient descent (Section 3.1, Appendix B.6).

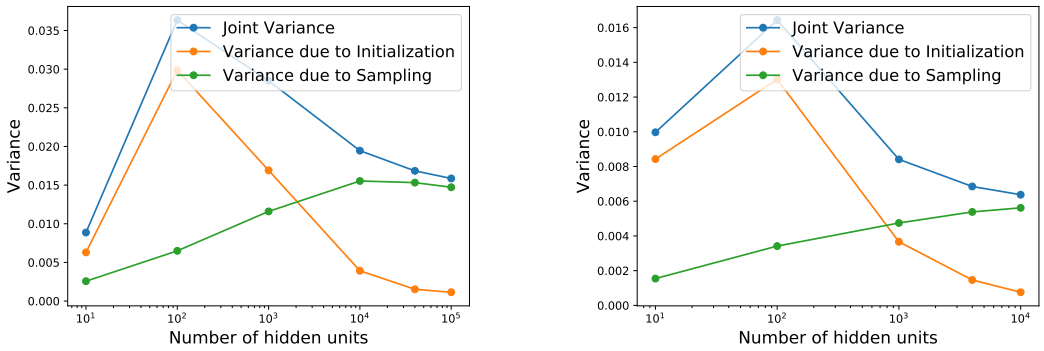


Figure 2: Trends of variance due to sampling and variance due to initialization with width on CIFAR10 (left) and on SVHN (right). Variance due to initializaiton decreases with width, once in the over-parameterized setting. Variance due to sampling plateaus and remains constant. This is in contrast with what the bias-variance tradeoff would suggest.

total variance, we then have the further decomposition:

$$\mathrm{Var}(h_\theta(x)) = \mathbb{E}_S\left[\mathrm{Var}_I\left(h_\theta(x)|S\right)\right] + \mathrm{Var}_S\left(\mathbb{E}_I\left[h_\theta(x)|S\right]\right) \tag{3}$$

We call the first term *variance due to initialization* and the second term *variance due to sampling* throughout the paper. Note that risks computed with classification losses (e.g cross-entropy or 0-1 loss) do not have such a clean bias-variance decomposition (Domingos, 2000; James, 2003). However, it is natural to expect that bias and variance are useful indicators of the performance of the models. In fact, we show the classification risk can be bounded as 4 times the regression risk in Appendix D.4.

3. Experiments

In this section, we experimentally study how variance of fully connected single hidden layer networks varies with width. We provide evidence against Geman et al. (1992)'s claim that "bias falls and variance increases with the number of hidden units." Experimental details are specified in Appendix A.

3.1. Bias and Variance

In Fig. 1, we see that variance decreases (along with bias) as network width increases on MNIST. Similarly, we also observe this phenomenon in CIFAR10 and SVHN (Fig. 2 and Appendices B.1 and B.2). In addition to these classification tasks, we see this in a sinusoid regression task (Fig. 3c and Appendix B.7). In each of these tasks, the same hyperparameters are used across all widths.

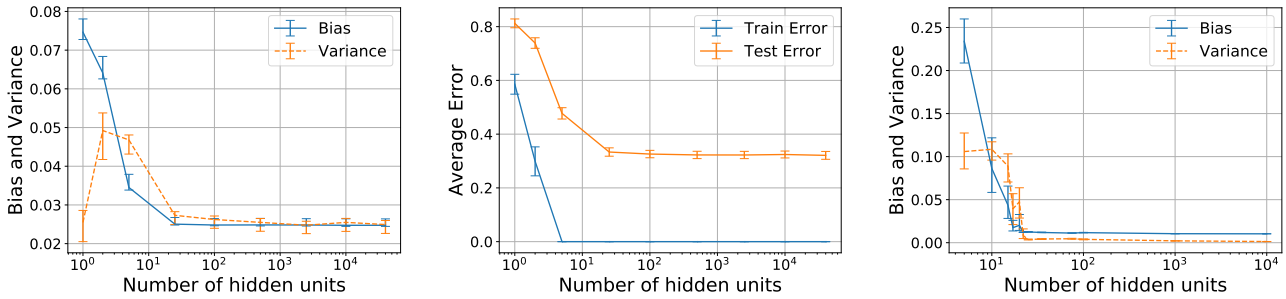
Decreasing the size of the dataset can only increase variance. To study the robustness of the above observation, we decrease the size of the MNIST training set to just 100 examples. In this small data setting, somewhat surprisingly, we still see that *both* bias *and* variance decrease with width (Figure 3a). The test error behaves similarly (Figure 3b). Test error trends for all of our experiments follow the bias-variance trends (Appendix B), as the bias-variance decomposition would suggest. Because performance is more sensitive to step size in the small data setting, the step size for each network size is tuned using a validation set (see Appendix B.4 for step sizes). This protocol allows the bias to decrease with width, indicating effective capacity is, indeed, increasing while variance is decreasing (see Appendix A.1 for more discussion).

To see how dependent this phenomenon is on SGD, we also run these experiments using batch gradient descent and PyTorch's version of LBFGS. Interestingly, we find a decreasing variance trend with those optimizers as well. These experiments are included in Appendix B.6.

3.2. Decoupling Variance due to Sampling from Variance due to Initialization

In order to better understand this variance phenomenon in neural networks, we separate the variance due to sampling from the variance due to initialization, according to the law of total variance (Equation 3). Contrary to what traditional bias-variance tradeoff intuition would suggest, we find variance due to sampling levels with increasingly large width (Fig. 2). Furthermore, we find that variance due to initialization decreases with width, causing the joint variance to decrease with width (Fig. 2).

A body of recent work has provided evidence that over-parameterization (in width) helps gradient descent optimize to global minima in neural networks (Du et al., 2019; Du & Lee, 2018; Soltanolkotabi et al., 2017; Livni et al., 2014; Zhang et al., 2018). Always reaching a global minimum implies low variance due to initialization on the *training set*. Our observation of decreasing variance on the *test set*



(a) Variance decreases with width, even in the small MNIST setting.

(b) Test error trend is same as bias-variance trend (small MNIST).

(c) Similar bias-variance trends on sinusoid regression task.

Figure 3: We see the same bias-variance trends in small data settings: small MNIST (left) and a regression setting (right).

shows that the over-parameterization (in width) effect on optimization seems to extend to generalization, on the data sets we consider.

4. Discussion and Theoretical Insights

Our empirical results demonstrate that in the practical setting, variance due to initialization decreases with network width while variance due to sampling levels off. Here, we take inspiration from linear models (Hastie et al., 2009, Section 7.3) to provide arguments for the behavior of variance in increasingly wide neural networks.

Remark 1: In *overparameterized* linear models, variance does *not* grow with the number of parameters. This is due to the fact that, all learning occurs in $\text{rowspace}(X)$ of the design matrix X (no learning in $\text{nullspace}(X)$), and the dimension of the solution space, $r = \text{rank}(X)$, is independent of N. For a complete walk-through of this, see Appendix C. We formalize this in Proposition 1 there.

We will illustrate our arguments in the following simplified setting, where $\mathcal{M}$, $\mathcal{M}^{\perp}$, and $d(N)$ are the more general analogs of $\text{rowspace}(X)$, $\text{nullspace}(X)$, and r (respectively):

Setting. Let N be the dimension of the parameter space. The prediction for a fixed example x, given by a trained network parameterized by θ depends on:

(i) a subspace of the parameter space, $\mathcal{M} \in \mathbb{R}^N$ with relatively small dimension, $d(N)$, which depends only on the learning task.

(ii) parameter components corresponding to directions orthogonal to $\mathcal{M}$. The orthogonal $\mathcal{M}^{\perp}$ of $\mathcal{M}$ has dimension, $N - d(N)$, and is essentially irrelevant to the learning task.

We can write the parameter vector as a sum of these two components $\theta = \theta_{\mathcal{M}} + \theta_{\mathcal{M}^{\perp}}$. We will further make the following assumptions:

Assumption 1 The optimization of the loss function is invariant with respect to $\theta_{\mathcal{M}\perp}$.

Assumption 2 Regardless of initialization, the optimization method consistently yields a solution with the same $\theta_{\mathcal{M}}$ component, (i.e. the same vector when projected onto $\mathcal{M}$).

We provide a short discussion on these assumptions in Appendix E. Given the above assumptions, the following result, proved in Appendix D.3, shows that the variance from initialization vanishes as we increase N.

Theorem 1 (Decay of variance due to initialization)**.** *For a fixed data set and parameters initialized as $\theta_0 \sim \mathcal{N}(0, \frac{1}{N}I)$, the variance of the prediction satisfies the inequality,*

$$Var_{\theta_0}(h_\theta(x)) \leq C\frac{2L^2}{N} \tag{4}$$

where L is the Lipschitz constant of the prediction with respect to θ, and for some universal constant $C > O$.

This result guarantees that the variance from initialization decreases to zero as N increases, provided the Lipschitz constant L grows more slowly than the square root of dimension, $L = o(\sqrt{N})$.

5. Conclusion

First, we provide evidence against Geman et al. (1992)'s claim that "the price to pay for achieving low bias is high variance," finding that *both* bias *and* variance decrease with width. Second, we find variance due to sampling (analog of regular variance in simple settings) does not appear to be dependent on width, once sufficiently over-parameterized. Third, variance due to initialization decreases with width. We see further theoretical treatment of variance as a fruitful direction for better understanding complexity and generalization abilities of neural networks.

References

Advani, M. S. and Saxe, A. M. High-dimensional dynamics of generalization error in neural networks. *CoRR*, abs/1710.03667, 2017.

Arpit, D., Jastrzebski, S., Ballas, N., Krueger, D., Bengio, E., Kanwal, M. S., Maharaj, T., Fischer, A., Courville, A., Bengio, Y., and Lacoste-Julien, S. A closer look at memorization in deep networks. *ICML 2017*, 70:233–242, 06–11 Aug 2017.

Belkin, M., Hsu, D., Ma, S., and Mandal, S. Reconciling modern machine learning and the bias-variance tradeoff. *arXiv e-prints*, art. arXiv:1812.11118, December 2018.

Bishop, C. M. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag, Berlin, Heidelberg, 2006.

Bousquet, O. and Elisseeff, A. Stability and generalization. *Journal of Machine Learning Research*, 2:499–526, March 2002.

Bühlmann, P. and Yu, B. Boosting with the l 2 loss: regression and classification. *Journal of the American Statistical Association*, 98(462):324–339, 2003.

Canziani, A., Paszke, A., and Culurciello, E. An analysis of deep neural network models for practical applications. *CoRR*, abs/1605.07678, 2016.

Domingos, P. A unified bias-variance decomposition and its applications. In *In Proc. 17th International Conf. on Machine Learning*, pp. 231–238. Morgan Kaufmann, 2000.

Du, S. and Lee, J. On the power of over-parametrization in neural networks with quadratic activation. In Dy, J. and Krause, A. (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 1329–1338, Stockholmsmässan, Stockholm Sweden, 10–15 Jul 2018. PMLR.

Du, S., Zhai, X., Poczos, B., and Singh, A. Gradient descent provably optimizes over-parameterized neural networks. volume abs/1810.02054, 2019.

Efron, B. Bootstrap methods: Another look at the jackknife. *Ann. Statist.*, 7(1):1–26, 01 1979.

EliteDataScience. Wtf is the bias-variance tradeoff? (infographic), May 2018.

Fortmann-Roe, S. Understanding the bias-variance tradeoff, June 2012.

Freund, Y. Boosting a weak learning algorithm by majority. *Information and computation*, 121(2):256–285, 1995.

Geman, S., Bienenstock, E., and Doursat, R. Neural networks and the bias/variance dilemma. *Neural Computation*, 4(1):1–58, 1992.

Gonzalez, J. E. Linear regression and the bias variance tradeoff, 2016. lecture notes.

Gur-Ari, G., Roberts, D. A., and Dyer, E. Gradient descent happens in a tiny subspace. *arXiv preprint arXiv: arXiv:1812.04754*, 2018.

Hastie, T., Tibshirani, R., and Friedman, J. *The elements of statistical learning: data mining, inference and prediction*. Springer, 2 edition, 2009.

James, G. M. Variance and bias for general loss functions. In *Machine Learning*, pp. 115–135, 2003.

Keskar, N. S., Mudigere, D., Nocedal, J., Smelyanskiy, M., and Tang, P. T. P. On large-batch training for deep learning: Generalization gap and sharp minima. In *International Conference on Learning Representations*, 2017.

Kohavi, R. and Wolpert, D. Bias plus variance decomposition for zero-one loss functions. In *Proceedings of the Thirteenth International Conference on International Conference on Machine Learning*, ICML'96, pp. 275–283, San Francisco, CA, USA, 1996. Morgan Kaufmann Publishers Inc.

LeCun, Y., Kanter, I., and Solla, S. Eigenvalues of covariance matrices: Application to neural-network learning. *Physical Review Letters*, 66:2396–2399, 05 1991.

Ledoux, M. *The Concentration of Measure Phenomenon*. Mathematical surveys and monographs. American Mathematical Society, 2001.

Lee, J., Sohl-dickstein, J., Pennington, J., Novak, R., Schoenholz, S., and Bahri, Y. Deep neural networks as gaussian processes. In *International Conference on Learning Representations*, 2018.

Li, C., Farkhoor, H., Liu, R., and Yosinski, J. Measuring the intrinsic dimension of objective landscapes. *ICLR 2018*, 2018.

Liang, T., Poggio, T. A., Rakhlin, A., and Stokes, J. Fisher-rao metric, geometry, and complexity of neural networks. *CoRR*, abs/1711.01530, 2017.

Livni, R., Shalev-Shwartz, S., and Shamir, O. On the computational efficiency of training neural networks. In Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N. D., and Weinberger, K. Q. (eds.), *Advances in Neural Information Processing Systems 27*, pp. 855–863. Curran Associates, Inc., 2014.

Neyshabur, B., Tomioka, R., and Srebro, N. In search of the real inductive bias: On the role of implicit regularization in deep learning. *International Conference on Learning Representations workshop track*, 2015.

Novak, R., Bahri, Y., Abolafia, D. A., Pennington, J., and Sohl-Dickstein, J. Sensitivity and generalization in neural networks: an empirical study. In *International Conference on Learning Representations*, 2018.

Sagun, L., Evci, U., Guney, V. U., Dauphin, Y., and Bottou, L. Empirical analysis of the hessian of over-parametrized neural networks. 2017.

Schapire, R. E. The strength of weak learnability. *Machine learning*, 5(2):197–227, 1990.

Smith, S. L., Kindermans, P.-J., and Le, Q. V. Don't decay the learning rate, increase the batch size. In *International Conference on Learning Representations*, 2018.

Soltanolkotabi, M., Javanmard, A., and Lee, J. D. Theoretical insights into the optimization landscape of over-parameterized shallow neural networks. *CoRR*, abs/1707.04926, 2017.

Spigler, S., Geiger, M., d'Ascoli, S., Sagun, L., Biroli, G., and Wyart, M. A jamming transition from under- to over-parametrization affects loss landscape and generalization. *CoRR*, abs/1810.09665, 2018.

Vapnik, V. N. *Statistical learning theory*. Adaptive and learning systems for signal processing, communications and control series. John Wiley & Sons, New York. A Wiley-Interscience Publication, 1998.

Vapnik, V. N. An overview of statistical learning theory. *Trans. Neur. Netw.*, 10(5):988–999, September 1999.

Zagoruyko, S. and Komodakis, N. Wide residual networks. In Richard C. Wilson, E. R. H. and Smith, W. A. P. (eds.), *Proceedings of the British Machine Vision Conference (BMVC)*, pp. 87.1–87.12. BMVA Press, September 2016.

Zhang, C., Bengio, S., Hardt, M., Recht, B., and Vinyals, O. Understanding deep learning requires rethinking generalization. *ICLR 2017*, 2017.

Zhang, C., Liao, Q., Rakhlin, A., Miranda, B., Golowich, N., and Poggio, T. A. Theory of deep learning iib: Optimization properties of SGD. *CoRR*, abs/1801.02254, 2018.

Appendices

A. Experimental details

We run experiments on different datasets: MNIST, CIFAR10, SVHN, small MNIST, and a sinusoid regression task. Averages over data samples are performed by taking the training set S and creating 50 bootstrap replicate training sets S' by sampling with replacement from S. We train 50 different neural networks for each hidden layer size using these different training sets. Then, we estimate $\mathcal{E}_{\text{bias}}$[2] and $\mathcal{E}_{\text{variance}}$ as in Section 2.2, where the population expectation $\mathbb{E}_x$ is estimated with an average over the test set. To estimate the two terms from the law of total variance (Equation 3), we use 10 random seeds for the outer expectation and 10 for the inner expectation, resulting in a total of 100 neural networks for each hidden layer size. Furthermore, we compute 99% confidence intervals for our bias and variance estimates using the bootstrap (Efron, 1979).

The networks are trained using SGD with momentum and generally run for long after 100% training set accuracy is reached (e.g. 500 epochs for full data MNIST and 10000 epochs for small data MNIST). The overall trends we find are robust to how long the networks are trained after the training error converges. To make our study as general as possible, we consider networks without regularization bells and whistles such as weight decay, dropout, or data augmentation, which Zhang et al. (2017) found to not be necessary for good generalization.

Hyperparameters: In the full data experiments (all but small MNIST), the same step size is used for all networks for a given dataset (0.1 for MNIST, 0.005 for CIFAR10, and 0.005 for SVHN). The momentum hyperparameter is always set to 0.9. In the small data MNIST experiment, the is tuned, using a validation set, for each width. The training for tuning is stopped after 1000 epochs, whereas the training for the final models is stopped after 10000 epochs. The chosen step sizes can be found in Appendix B.4.

A.1. Justification for tuning the step size on small MNIST

Because performance is more sensitive to step size in the small data setting, the step size for each network size is tuned using a validation set (see Appendix B.4 for step sizes).

Note that because we see decreasing bias with width, effective capacity is, indeed, increasing while variance is decreasing.

One control that motivates the experimental design choice of optimal step size is that it leads to the conventional decreasing bias trend (Fig. 3a) that indicates increasing effective capacity. In fact, in the corresponding experiment where step size is the same 0.01 for all network sizes, we do not see monotonically decreasing bias (Appendix B.5).

This sensitivity to step size in the small data setting is evidence that we are testing the limits of our hypothesis. By looking at the small data setting, we are able to test our hypothesis when the ratio of size of network to dataset size is quite large, and we still find this decreasing trend in variance (Fig. 3a).

[2] Because we do not have access to $\bar{y}$, we use the labels y to estimate $\mathcal{E}_{\text{bias}}$. This is equivalent to assuming noiseless labels and is standard procedure for estimating bias (Kohavi & Wolpert, 1996; Domingos, 2000).

B. Additional empirical results and discussion

B.1. CIFAR10

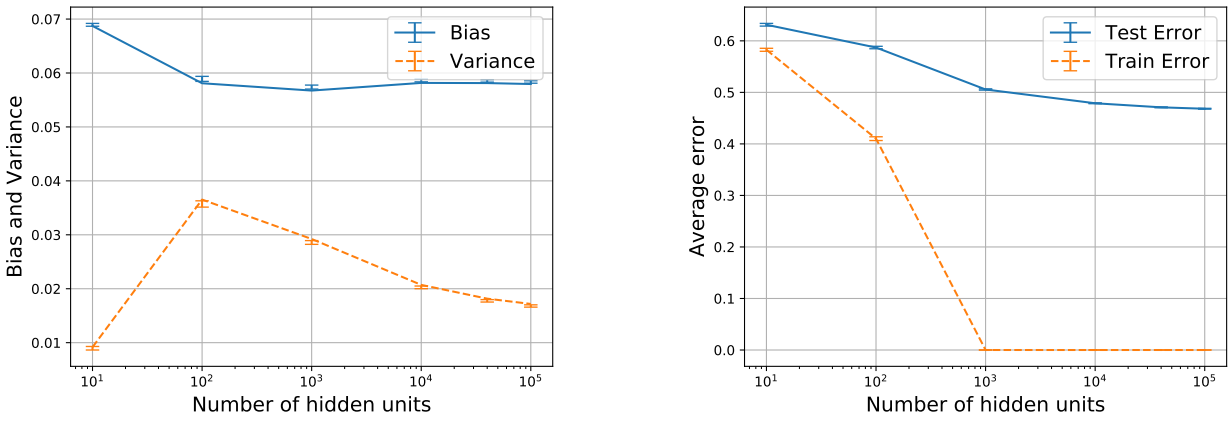


Figure 4: Bias-variance plot (left) and corresponding train and test error (right) for CIFAR10 after training for 150 epochs with step size 0.005 for all networks.

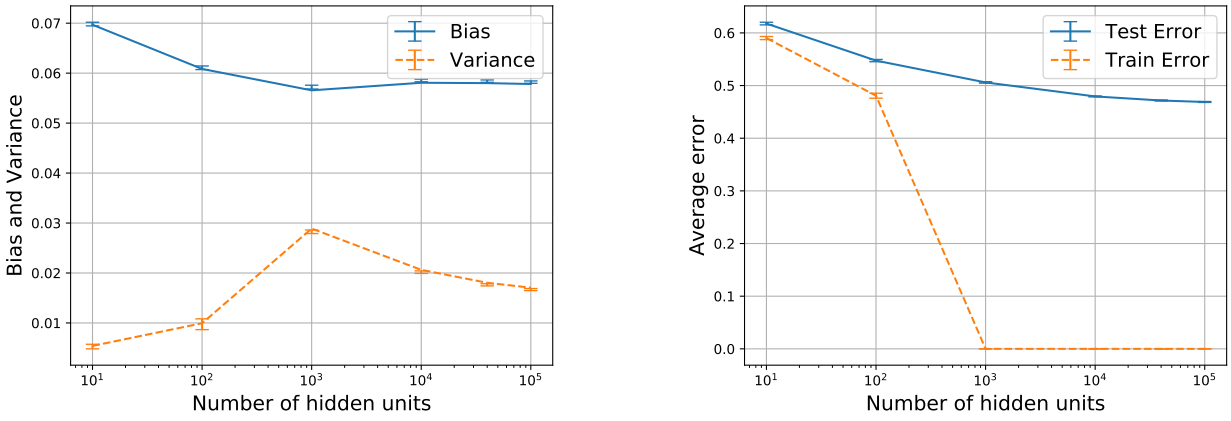


Figure 5: Bias-variance plot (left) and corresponding train and test error (right) for CIFAR10 after training for using *early stopping* with step size 0.005 for all networks.

B.2. SVHN

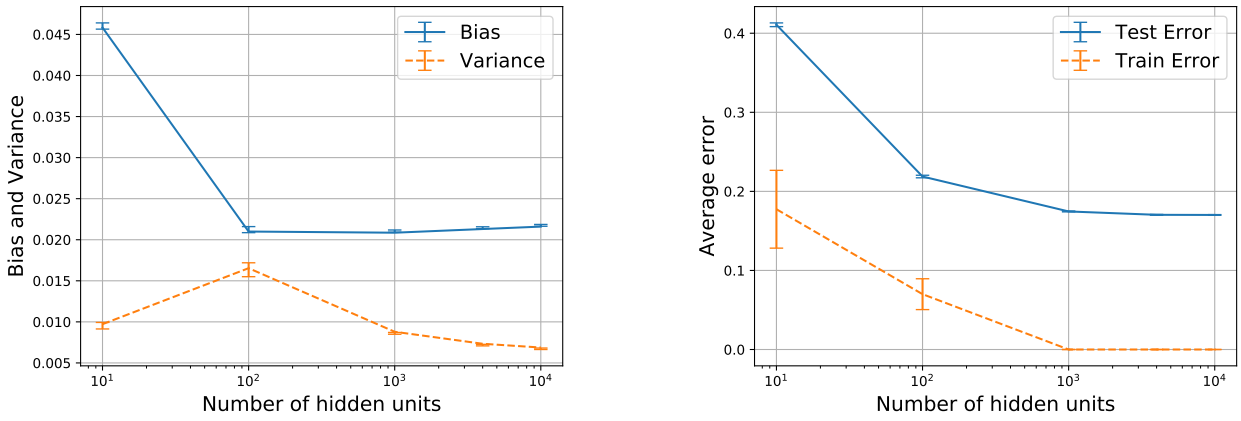


Figure 6: Bias-variance plot (left) and corresponding train and test error (right) for SVHN after training for 150 epochs with step size 0.005 for all networks.

B.3. MNIST

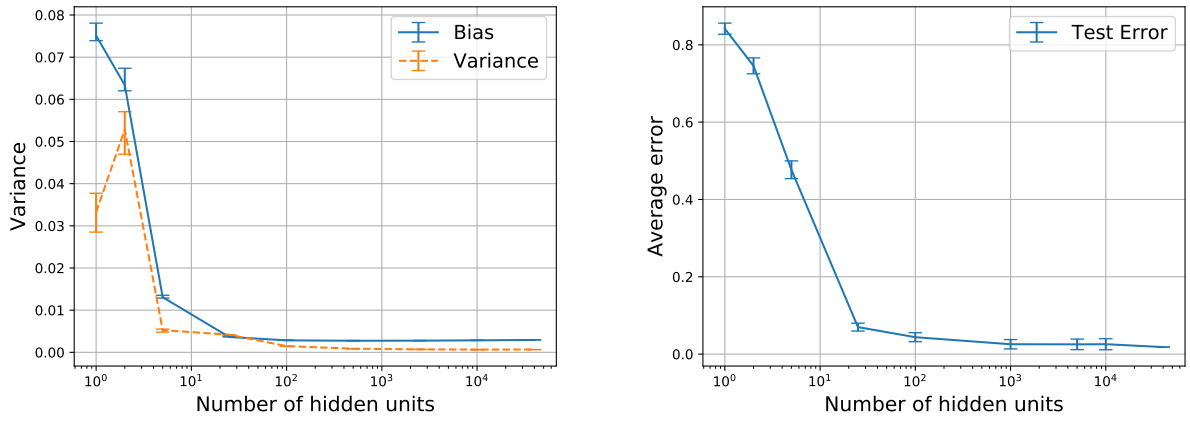


Figure 7: MNIST bias-variance plot from main paper (left) next to the corresponding test error (right)

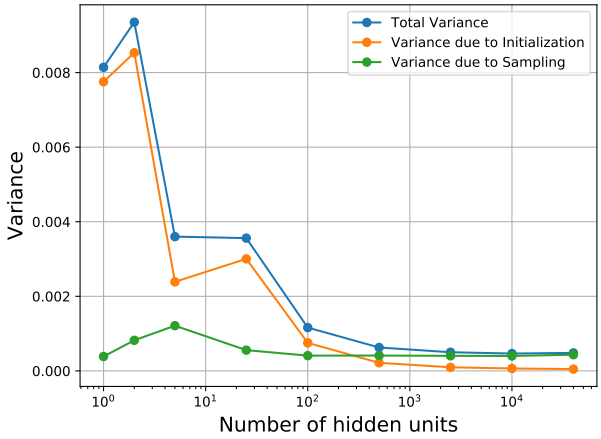
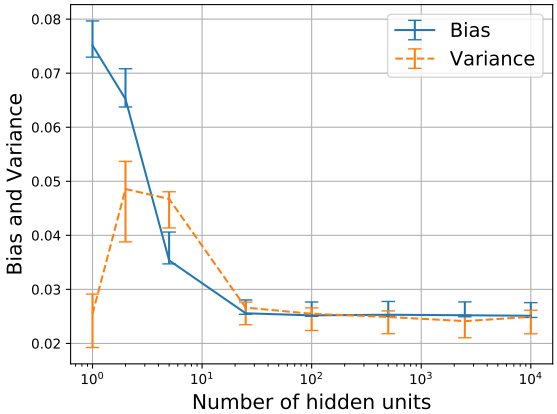
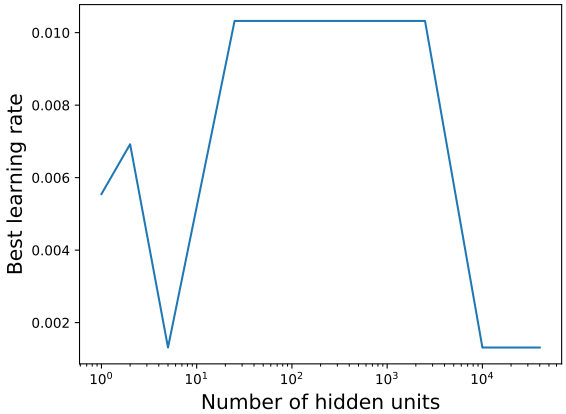


Figure 8: Decomposed variance on MNIST

B.4. Tuned learning rates for SGD



(a) Variance decreases with width, even in the small data MNIST setting (SGD). This figure is in the main paper, but we include it here to compare with the corresponding step sizes used.



(b) Corresponding optimal learning rates found, by random search, and used.

B.5. Fixed learning rate results for small data MNIST

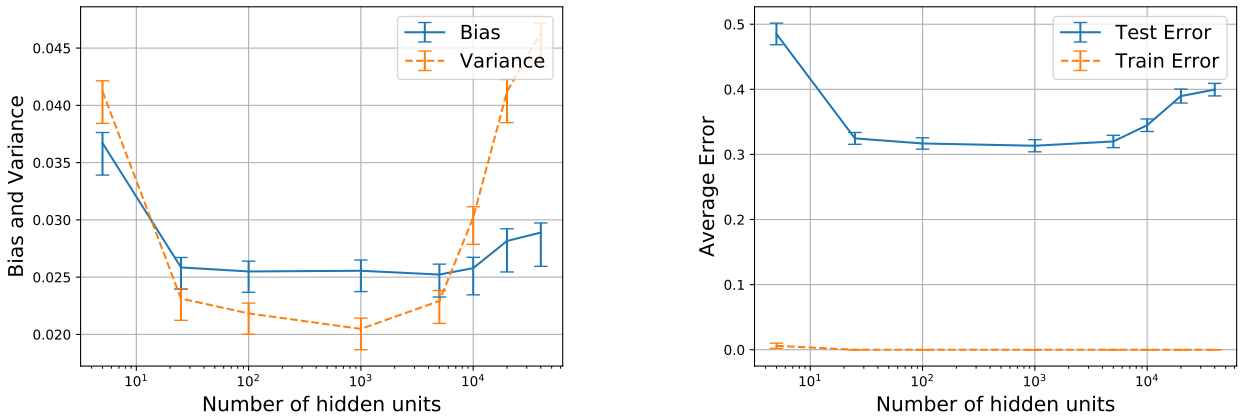


Figure 10: Variance on small data with a fixed learning rate of 0.01 for all networks.

Note that the U curve shown in Fig. 10 when we do not tune the step size is explained by the fact that the constant step chosen is a "good" step size for some networks and "bad" for others. Results from Keskar et al. (2017) and Smith et al. (2018) show that a step size that corresponds well to the noise structure in SGD is important for achieving good test set accuracy. Because our networks are different sizes, their stochastic optimization process will have a different landscape and noise structure. By tuning the step size, we are making the experimental design choice to keep *optimality of step size* constant across networks, rather than keeping step size constant across networks. To us, choosing this control makes much more sense than choosing to control for step size.

B.6. Other optimizers for width experiment on small data mnist

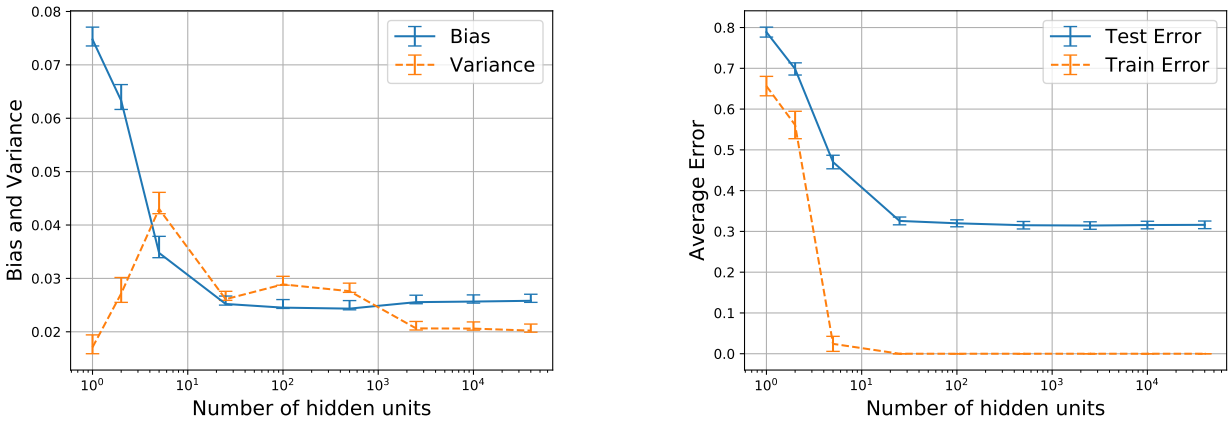


Figure 11: Variance decreases with width in the small data setting, even when using batch gradient descent.

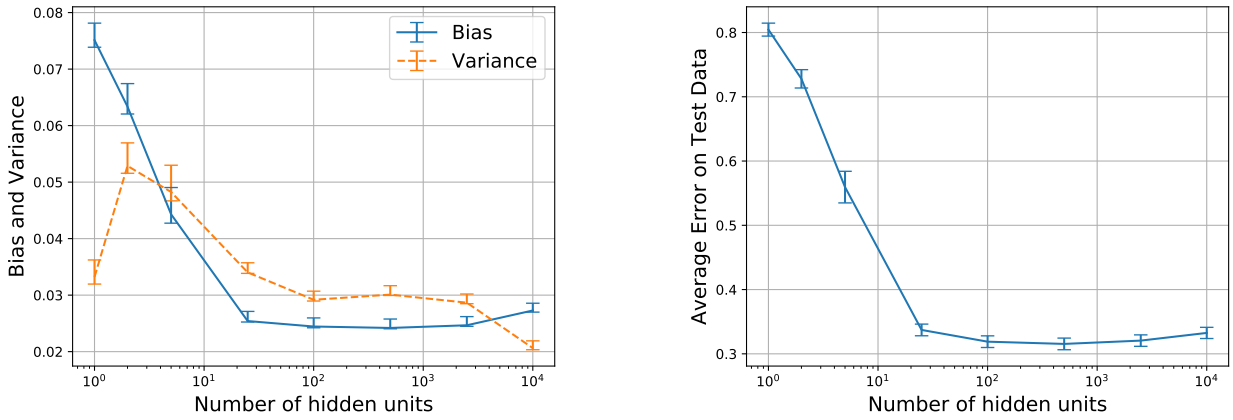
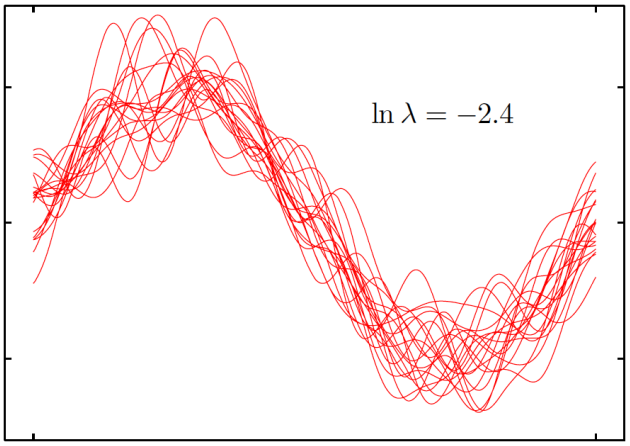
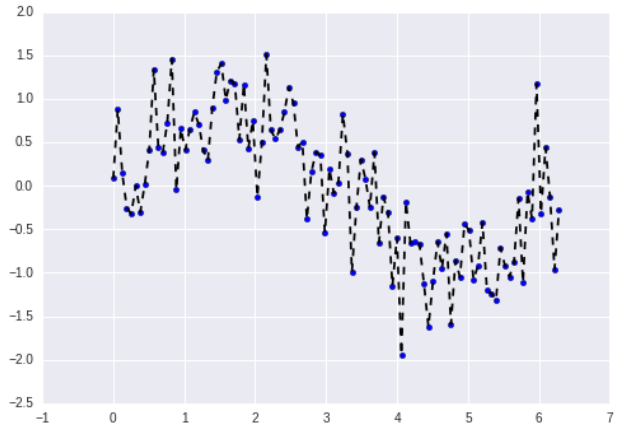


Figure 12: Variance decreases with width in the small data setting, even when using a strong optimizer, such as PyTorch's LBFGS, as the optimizer.

B.7. Sinusoid regression experiments



(a) Example of the many different functions learned by a high variance learner (Bishop, 2006, Section 3.2)



(b) Caricature of a single function learned by a high variance learner (EliteDataScience, 2018)

Figure 13: Caricature examples of high variance learners on sinusoid task. Below, we find that this does not happen with increasingly wide neural networks (Fig. 15 and Fig. 16).

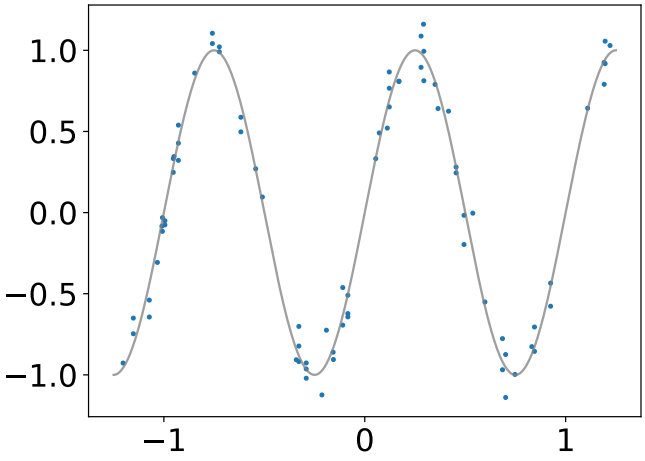


Figure 14: Target function of the noisy sinusoid regression task (in gray) and an example of a training set (80 data points) sampled from the noisy distribution.

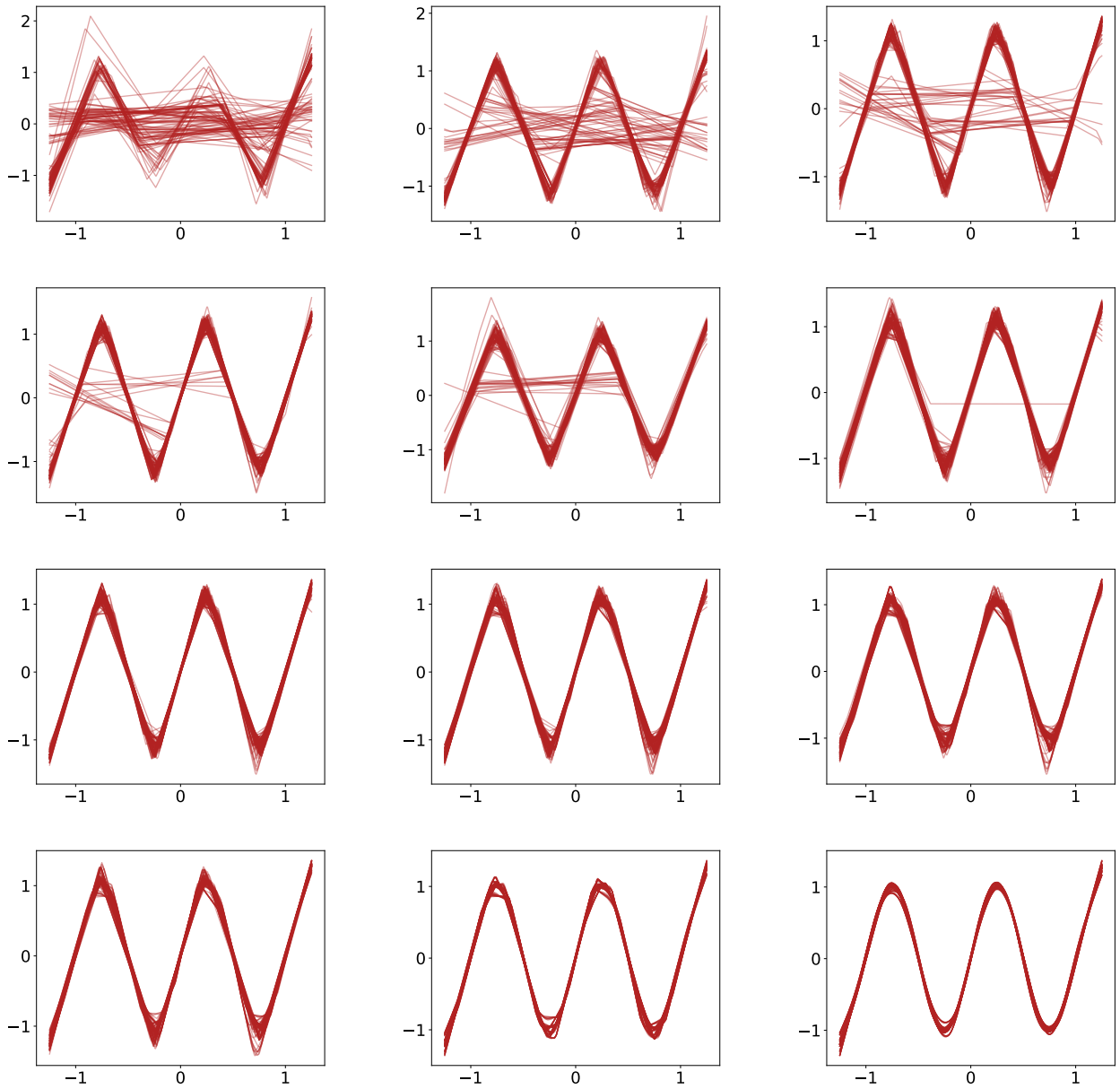


Figure 15: Visualization of 100 different functions learned by the different width neural networks. Darker color indicates higher density of different functions. Widths in increasing order from left to right and top to bottom: 5, 10, 15, 17, 20, 22, 25, 35, 75, 100, 1000, 10000. We do *not* observe the caricature from Fig. 13 as width is increased.

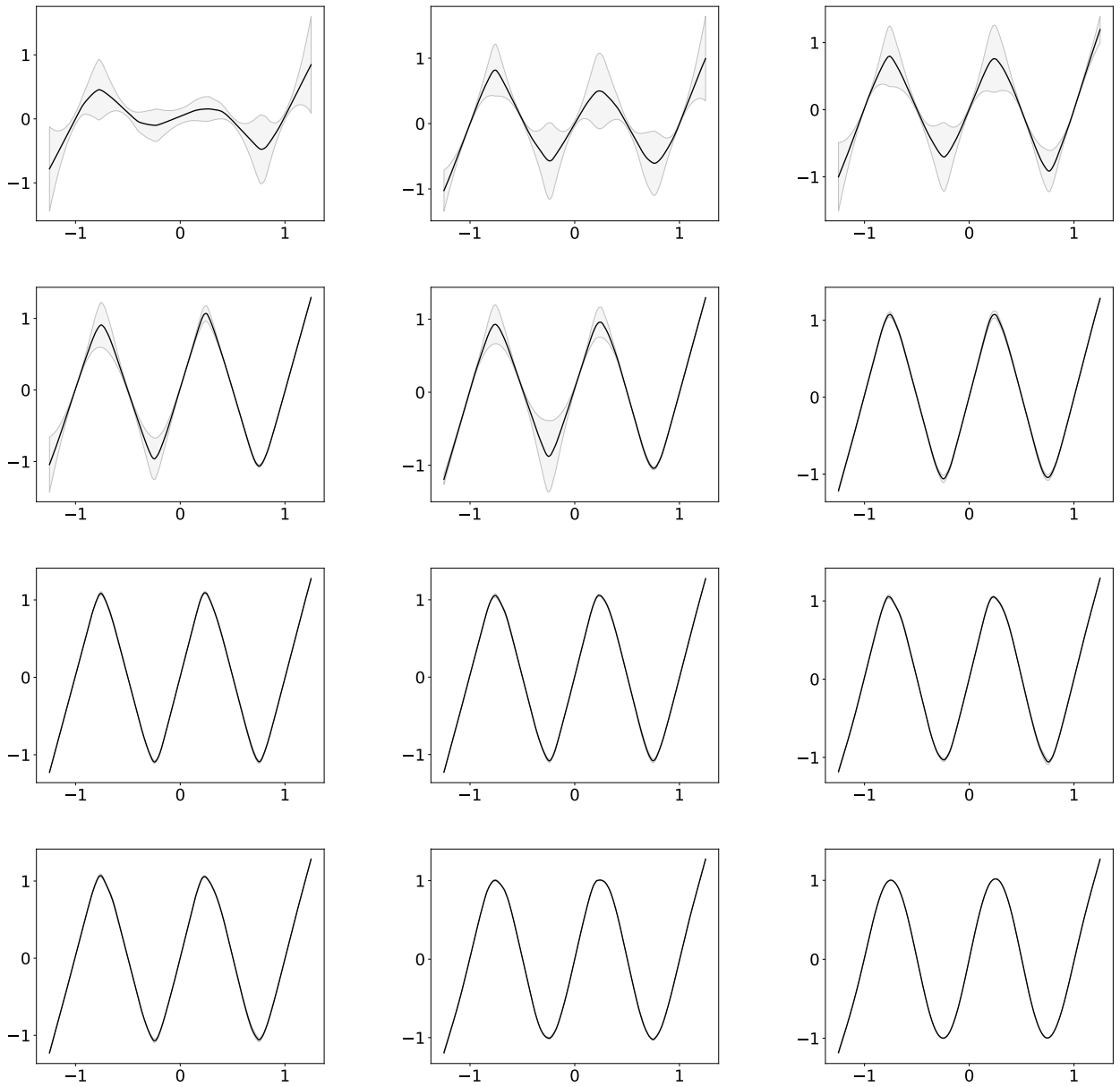
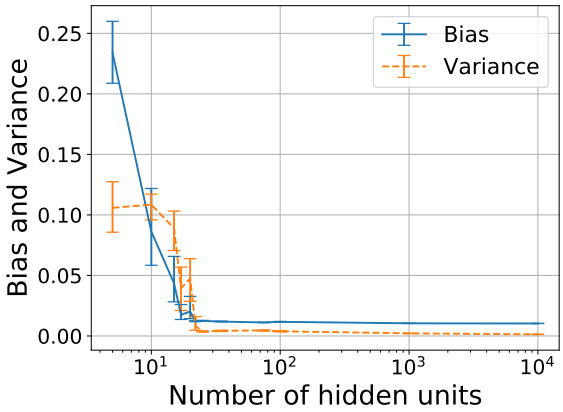


Figure 16: Visualization of the mean prediction and variance of the different width neural networks. Widths in increasing order from left to right and top to bottom: 5, 10, 15, 17, 20, 22, 25, 35, 75, 100, 1000, 10000.



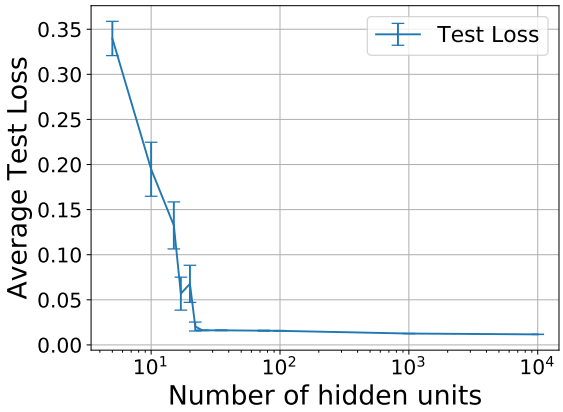


Figure 17: We observe the same trends of bias and total variance in the sinusoid regression setting. The figure on the left is in the main paper, while the figure on the right is support.

C. Insights from Linear Models

In this section, we review the classic result that the variance of a linear model grows with the number of parameters (Hastie et al., 2009, Section 7.3) and point out that variance behaves differently in the over-parameterized setting.

We consider least-squares linear regression in a standard setting which assumes a noisy linear mapping $y = \theta^T x + \epsilon$ between input feature vectors $x \in \mathbb{R}^N$ and real outputs, where ϵ denotes the noise random variable with $\mathbb{E}[\epsilon] = 0$ and $\text{Var}(\epsilon) = \sigma_\epsilon^2$. In this context, the over-parameterized setting is when the dimension N of the input space is larger than the number m of examples.

Let X denote the $m \times N$ design matrix whose i^{th} row is the training point x_i^T, let Y denote the corresponding labels, and let $\Sigma = X^T X$ denote the empirical covariance matrix. We consider the "fixed design" setting where X is fixed, so all of the randomness due to data sampling comes solely from ϵ. $\mathcal{A}$ learns weights $\hat{\theta}$ from (X, Y), either by a closed-form solution or by gradient descent, using a standard initialization $\theta_0 \sim \mathcal{N}(0, \frac{1}{N} I)$. The predictor makes a prediction on $x \sim \mathcal{D}$: $h(x) = \hat{\theta}^T x$. Then, the quantity we care about is $\mathbb{E}_x \text{Var}(h(x))$.

C.1. Under-parameterized Setting

The case where $N \leq m$ is standard: if X has maximal rank, Σ is invertible; the solution is independent of the initialization and given by $\hat{\theta} = \Sigma^{-1} X^T Y$. All of the variance is a result of randomness in the noise ϵ. For a fixed x,

$$\text{Var}(h(x)) = \sigma_\epsilon^2 \text{Tr}(x x^T \Sigma^{-1}) \,. \tag{5}$$

This grows with the number of parameters N. For example, taking the expected value over the empirical distribution, $\hat{p}$, of the sample, we recover that the variance grows with N:

$$\mathbb{E}_{x \sim \hat{p}}[\text{Var}(h(x))] = \frac{N}{m} \sigma_\epsilon^2 \,. \tag{6}$$

We provide a reproduction of the proofs in Appendix D.1.

C.2. Over-parameterized Setting

The over-parameterized case where $N > m$ is more interesting: even if X has maximal rank, Σ is not invertible. This leads to a subspace of solutions, but gradient descent yields a unique solution from updates that belong to the span of the training points x_i (row space of X) (LeCun et al., 1991), which is of dimension $r = \text{rank}(X) = \text{rank}(\Sigma)$. Correspondingly, no learning occurs in the null space of X, which is of dimension $N - r$. Therefore, gradient descent yields the solution that is closest to initialization: $\hat{\theta} = P_\perp(\theta_0) + \Sigma^+ X^T Y$, where $P_\perp$ projects onto the null space of X and $+$ denotes the Moore-Penrose inverse.

The variance has two contributions: one due to initialization and one due to sampling (here, the noise ϵ), as in Eq. (3). These are made explicit in Proposition 1.

Proposition 1 (Variance in over-parameterized linear models)**.** *Consider the over-parameterized setting where $N > m$. For a fixed x, the variance decomposition of Eq. (3) yields*

$$Var(h(x)) = \frac{1}{N}\|P_{\perp}(x)\|^2 + \sigma_\epsilon^2 Tr(xx^T\Sigma^+)\,. \tag{7}$$

This does not grow with the number of parameters N. In fact, because Σ^{-1} is replaced with Σ^+, the variance scales as the dimension of the data *(i.e the rank of X), as opposed to the number of parameters. For example, taking the expected value over the empirical distribution, $\hat{p}$, of the sample, we obtain*

$$\mathbb{E}_{x\sim\hat{p}}[Var(h(x))] = \frac{r}{m}\sigma_\epsilon^2\,, \tag{8}$$

where $r = \mathrm{rank}(X)$. We provide the proofs for over-parameterized linear models in Appendix D.2.

D. Some Proofs

D.1. Proof of Classic Result for Variance of Linear Model

Here, we reproduce the classic result that variance grows with the number of parameters in a linear model. This result can be found in Hastie et al. (2009)'s book, and a similar proof can be found in Gonzalez (2016)'s lecture slides.

Proof. For a fixed x, we have $h(x) = x^T\hat{\theta}$. Taking $\hat{\theta} = \Sigma^{-1}X^TY$ to be the gradient descent solution, and using $Y = X\theta + \epsilon$, we obtain:

$$h(x) = x^T\Sigma^{-1}X^T(X\theta+\epsilon) = x^T\theta + x^T\Sigma^{-1}X^T\epsilon$$

Hence $\mathbb{E}_\epsilon[h(x)] = x^T\theta$, and the variance is,

$$\begin{aligned}
\mathrm{Var}_\epsilon(h(x)) &= \mathbb{E}_\epsilon[(h(x) - \mathbb{E}_\epsilon[h(x)])^2] \\
&= \mathbb{E}_\epsilon[(x^T\theta + x^T\Sigma^{-1}X^T\epsilon - x^T\theta)^2] \\
&= \mathbb{E}_\epsilon[(x^T\Sigma^{-1}X^T\epsilon)^2] \\
&= \mathbb{E}_\epsilon[(x^T\Sigma^{-1}X^T\epsilon)(x^T\Sigma^{-1}X^T\epsilon)^T] \\
&= \mathbb{E}_\epsilon[x^T\Sigma^{-1}X^T\epsilon\epsilon^T(x^T\Sigma^{-1}X^T)^T] \\
&= \sigma_\epsilon^2 x^T\Sigma^{-1}\Sigma\Sigma^{-1}x \\
&= \sigma_\epsilon^2 x^T\Sigma^{-1}\Sigma\Sigma^{-1}x \\
&= \sigma_\epsilon^2 x^T\Sigma^{-1}x \\
&= \sigma_\epsilon^2 \mathrm{Tr}(x^T\Sigma^{-1}x) \\
&= \sigma_\epsilon^2 \mathrm{Tr}(xx^T\Sigma^{-1})
\end{aligned}$$

Taking the expected value over the empirical distribution, $\hat{p}$, of the sample, we find an explicit increasing dependence on N:

$$\begin{aligned}
\mathbb{E}_{x\sim\hat{p}}[\mathrm{Var}_\epsilon(h(x))] &= \mathbb{E}_{x\sim\hat{p}}[\sigma_\epsilon^2\mathrm{Tr}(xx^T\Sigma^{-1})] \\
&= \sigma_\epsilon^2\mathrm{Tr}(\mathbb{E}_{x\sim\hat{p}}[xx^T]\Sigma^{-1}) \\
&= \sigma_\epsilon^2\mathrm{Tr}\left(\frac{1}{m}\Sigma\Sigma^{-1}\right) \\
&= \sigma_\epsilon^2\frac{1}{m}\mathrm{Tr}(I_N) \\
&= \sigma_\epsilon^2\frac{N}{m}
\end{aligned}$$

□

D.2. Proof of Result for Variance of Over-parameterized Linear Models

Here, we produce a variation on what was done in Appendix D.1 to show that variance does not grow with the number of parameters in over-parameterized linear models. Recall that we are considering the setting where $N > m$, where N is the number of parameters and m is the number of training examples.

Proof. By the law of total variance,

$$\text{Var}(h(x)) = \mathbb{E}_\epsilon \text{Var}_{\theta_0}(h(x)) + \text{Var}_\epsilon(\mathbb{E}_{\theta_0}[h(x)])$$

Here have $h(x) = x^T\hat{\theta}$, where $\hat{\theta}$ the gradient descent solution $\hat{\theta} = P_\perp(\theta_0) + \Sigma^+ X^T Y$, and $\theta_0 \sim \mathcal{N}(0, \frac{1}{N}I)$. Then,

$$\begin{aligned}
\text{Var}_{\theta_0}(h(x)) &= \mathbb{E}_{\theta_0}[(h(x) - \mathbb{E}_{\theta_0}[h(x)])^2] \\
&= \mathbb{E}_{\theta_0}[x^T(P_\perp(\theta_0) - \mathbb{E}_{\theta_0}[P_\perp(\theta_0)])^2] \\
&= \text{Var}_{\theta_0}(x^T P_\perp(\theta_0)) \\
&= \text{Var}_{\theta_0}(P_\perp(x)^T P_\perp(\theta_0)) \\
&= \frac{1}{N}\|P_\perp(x)\|^2
\end{aligned}$$

Since $\mathbb{E}_{\theta_0}(h(x)) = x^T\Sigma^+X^TY$, the calculation of $\text{Var}_\epsilon(\mathbb{E}_{\theta_0})h(x))$ is similar as in D.1, where Σ^{-1} is replaced by Σ^+. Thus,

$$\text{Var}_\epsilon(\mathbb{E}_{\theta_0}h(x)) = \sigma_\epsilon^2 \text{Tr}(xx^T\Sigma^+)$$

Taking the expected value over the empirical distribution, $\hat{p}$, of the sample, we find an explicit dependence on $r = \text{rank}(X)$, not N:

$$\begin{aligned}
\mathbb{E}_{x\sim\hat{p}}[\text{Var}(h(x))] &= 0 + \mathbb{E}_{x\sim\hat{p}}[\sigma_\epsilon^2 \text{Tr}(xx^T\Sigma^+)] \\
&= \sigma_\epsilon^2 \text{Tr}(\mathbb{E}_{x\sim\hat{p}}[xx^T]\Sigma^+) \\
&= \sigma_\epsilon^2 \text{Tr}\left(\frac{1}{m}\Sigma\Sigma^+\right) \\
&= \sigma_\epsilon^2 \frac{1}{m}\text{Tr}(I_r^+) \\
&= \sigma_\epsilon^2 \frac{r}{m}
\end{aligned}$$

where I_r^+ denotes the diagonal matrix with 1 for the first r diagonal elements and 0 for the remaining $N - r$ elements. □

D.3. Proof of Theorem 1

First we state some known concentration results (Ledoux, 2001) that we will use in the proof.

Lemma 1 (Levy). *Let $h : S_R^n \to \mathbb{R}$ be a function on the n-dimensional Euclidean sphere of radius R, with Lipschitz constant L; and $\theta \in S_R^n$ chosen uniformly at random for the normalized measure. Then*

$$\mathbb{P}(|h(\theta) - \mathbb{E}[h]| > \epsilon) \leq 2\exp\left(-C\frac{n\epsilon^2}{L^2R^2}\right) \tag{9}$$

for some universal constant $C > 0$.

Uniform measures on high dimensional spheres approximate Gaussian distributions (Ledoux, 2001). Using this, Levy's lemma yields an analogous concentration inequality for functions of Gaussian variables:

Lemma 2 (Gaussian concentration). *Let $h : \mathbb{R}^n \to \mathbb{R}$ be a function on the Euclidean space $\mathbb{R}^n$, with Lipschitz constant L; and $\theta \sim \mathcal{N}(0, \sigma\mathbb{I}_n)$ sampled from an isotropic n-dimensional Gaussian. Then:*

$$\mathbb{P}(|h(\theta) - \mathbb{E}[h]| > \epsilon) \leq 2\exp\left(-C\frac{\epsilon^2}{L^2\sigma^2}\right) \tag{10}$$

for some universal constant $C > 0$.

Note that in the Gaussian case, the bound is dimension free.

In turn, concentration inequalities give variance bounds for functions of random variables.

Corollary 1. *Let h be a function satisfying the conditions of Theorem 2, and Var$(h) = \mathbb{E}[(h - \mathbb{E}[h])^2]$. Then*

$$Var(h) \leq \frac{2L^2\sigma^2}{C} \tag{11}$$

Proof. Let $g = h - \mathbb{E}[h]$. Then $\text{Var}(h) = \text{Var}(g)$ and

$$\text{Var}(g) = \mathbb{E}[|g|^2] = 2\mathbb{E}\int_0^{|g|} t dt = 2\mathbb{E}\int_0^{\infty} t\mathbb{1}_{|g|>t}\, dt \tag{12}$$

Now swapping expectation and integral (by Fubini theorem), and by using the identity $\mathbb{E}\mathbb{1}_{|g|>t} = \mathbb{P}(|g| > t)$, we obtain

$$\begin{aligned}\text{Var}(g) &= 2\int_0^{\infty} t\,\mathbb{P}_R(|g| > t)\, dt \\ &\leq 2\int_0^{\infty} 2t\exp\left(-C\frac{t^2}{L^2\sigma^2}\right) dt \\ &= 2\left[-\frac{L^2\sigma^2}{C}\exp\left(-C\frac{t^2}{L^2\sigma^2}\right)\right]_0^{\infty} = \frac{2L^2\sigma^2}{C}\end{aligned}$$

□

We are now ready to prove Theorem 1. We first recall our assumptions:

Assumption 1. *The optimization of the loss function is invariant with respect to $\theta_{\mathcal{M}\perp}$.*

Assumption 2. *Along $\mathcal{M}$, optimization yields solutions independently of the initialization θ_0.*

We add the following assumptions.

Assumption 3. *The prediction $h_\theta(x)$ is L-Lipschitz with respect to $\theta_{\mathcal{M}\perp}$.*

Assumption 4. *The network parameters are initialized as*

$$\theta_0 \sim \mathcal{N}(0, \frac{1}{N}\cdot I_{N\times N}). \tag{13}$$

We first prove that the Gaussian concentration theorem translates into concentration of predictions in the setting of **??**.

Theorem 2 (Concentration of predictions)**.** *Consider the setting of* **??** *and Assumptions 1 and 4. Let θ denote the parameters at the end of the learning process. Then, for a fixed data set, S we get concentration of the prediction, under initialization randomness,*

$$\mathbb{P}(|h_\theta(x) - \mathbb{E}[h_\theta(x)]| > \epsilon) \leq 2\exp\left(-C\frac{N\epsilon^2}{L^2}\right) \tag{14}$$

for some universal constant $C > 0$.

Proof. In our setting, the parameters at the end of learning can be expressed as

$$\theta = \theta^*_{\mathcal{M}} + \theta_{\mathcal{M}^\perp} \tag{15}$$

where $\theta^*_{\mathcal{M}}$ is independent of the initialization θ_0. To simplify notation, we will assume that, at least locally around $\theta^*_{\mathcal{M}}$, $\mathcal{M}$ is spanned by the first $d(N)$ standard basis vectors, and $\mathcal{M}^\perp$ by the remaining $N - d(N)$. This will allow us, from now on, to use the same variable names for $\theta_{\mathcal{M}}$ and $\theta_{\mathcal{M}^\perp}$ to denote their lower-dimensional representations of dimension $d(N)$ and $N - d(N)$ respectively. More generally, we can assume that there is a mapping from $\theta_{\mathcal{M}}$ and $\theta_{\mathcal{M}^\perp}$ to those lower-dimensional representations.

From Assumptions 1 and 4 we get

$$\theta_{\mathcal{M}^\perp} \sim \mathcal{N}\left(0, \frac{1}{N} I_{(N-d(N))\times(N-d(N))}\right). \tag{16}$$

Let $g(\theta_{\mathcal{M}^\perp}) \triangleq h_{\theta^*_{\mathcal{M}}+\theta_{\mathcal{M}^\perp}}(x)$. By Assumption 3, $g(\cdot)$ is L-Lipschitz. Then, by the Gaussian concentration theorem we get,

$$\mathbb{P}(|g(\theta_{\mathcal{M}^\perp}) - \mathbb{E}[g(\theta_{\mathcal{M}^\perp})]| > \epsilon) \leq 2\exp\left(-C\frac{N\epsilon^2}{L^2}\right). \tag{17}$$

□

The result of Theorem 1 immediately follows from Theorem 2 and Corollary 1, with $\sigma^2 = 1/N$:

$$\text{Var}_{\theta_0}(h_\theta(x)) \leq C\frac{2L^2}{N} \tag{18}$$

Provided the Lipschitz constant L of the prediction grows more slowly than the square of dimension, $L = o(\sqrt{N})$, we conclude that the variance vanishes to zero as N grows.

D.4. Bound on classification error in terms of regression error

In this section we give a bound on classification risk $\mathcal{R}_{\text{classif}}$ in terms of the regression risk $\mathcal{R}_{\text{reg}}$.

Notation. Our classifier defines a map $h : \mathcal{X} \to \mathbb{R}^k$, which outputs probability vectors $h(x) \in \mathbb{R}^k$, with $\sum_{y=1}^k h(x)_y = 1$. The classification loss is defined by

$$\begin{aligned} L(h) &= \text{Prob}_{x,y}\{h(x)_y < \max_{y'} h(x)_{y'}\} \\ &= \mathbb{E}_{(x,y)} I(h(x)_y < \max_{y'} h(x)_{y'}) \end{aligned} \tag{19}$$

where $I(a) = 1$ if predicate a is true and 0 otherwise. Given trained predictors h_S indexed by training dataset S, the classification and regression risks are given by,

$$\mathcal{R}_{\text{classif}} = \mathbb{E}_S L(h_S), \qquad \mathcal{R}_{\text{reg}} = \mathbb{E}_S \mathbb{E}_{(x,y)} ||h_S(x) - Y||_2^2 \tag{20}$$

where Y denotes the one-hot vector representation of the class y.

Proposition 2. *The classification risk is bounded by four times the regression risk, $\mathcal{R}_{classif} \leq 4\mathcal{R}_{reg}$.*

Proof. First note that, if $h(x) \in \mathbb{R}^k$ is a probability vector, then

$$h(x)_y < \max_{y'} h(x)_{y'} \implies h(x)_y < \frac{1}{2}$$

By taking the expectation over x, y, we obtain the inequality $L(h) \leq \widetilde{L}(h)$ where

$$\widetilde{L}(h) = \text{Prob}_{x,y}\{h(x)_y < \frac{1}{2}\} \tag{21}$$

We then have,

$$\begin{aligned} \mathcal{R}_{\text{classif}} := \mathbb{E}_S L(h_S) &\leq \mathbb{E}_S \tilde{L}(h_S) \\ &= \text{Prob}_{S;\,x,y}\{h_S(x)_y < \frac{1}{2}\} \\ &= \text{Prob}_{S;\,x,y}\{|h_S(x)_y - Y_y| > \frac{1}{2}\} \\ &\leq \text{Prob}_{S;\,x,y}\{||h_S(x) - Y||_2 > \frac{1}{2}\} \\ &= \text{Prob}_{S;\,x,y}\{||h_S(x) - Y||_2^2 > \frac{1}{4}\} \leq 4\mathcal{R}_{\text{reg}} \end{aligned}$$

where the last inequality follows from Markov's inequality.

□

E. Discussion on Assumptions

We made strong assumptions, but there is some support for them in the literature. The existence of a subspace $\mathcal{M}_\perp$ in which no learning occurs was also conjectured by Advani & Saxe (2017) and shown to hold in linear neural networks under a simplifying assumption that decouples the dynamics of the weights in different layers. Li et al. (2018) empirically showed the existence of a critical number $d(N) = d$ of relevant parameters for a given learning task, independent of the size of the model. Sagun et al. (2017) showed that the spectrum of the Hessian for over-parameterized networks splits into (i) a bulk centered near zero and (ii) a small number of large eigenvalues; and Gur-Ari et al. (2018) recently gave evidence that the small subspace spanned by the Hessian's top eigenvectors is preserved over long periods of training. These results suggest that learning occurs mainly in a small number of directions.

F. Probabilistic notion of effective capacity

The problem with classical complexity measures is that they do not take into account optimization and have no notion of what will actually be learned. Arpit et al. (2017, Section 1) define a notion of an *effective* hypothesis class to take into account what functions are possible to be learned by the learning algorithm.

However, this still has the problem of not taking into account what hypotheses are *likely* to be learned. To take into account the probabilistic nature of learning, we define the ϵ-*hypothesis class* for a data distribution $\mathcal{D}$ and learning algorithm $\mathcal{A}$, that contains the hypotheses which are at least ϵ-likely for some $\epsilon > 0$:

$$\mathcal{H}_{\mathcal{D}}(\mathcal{A}) = \{h : p(h(\mathcal{A}, S)) \geq \epsilon\}, \tag{22}$$

where S is a training set drawn from $\mathcal{D}^m$, $h(\mathcal{A}, S)$ is a random variable drawn from the distribution over learned functions induced by $\mathcal{D}$ and the randomness in $\mathcal{A}$; p is the corresponding density. Thinking about a model's ϵ-hypothesis class can lead to drastically different intuitions for the complexity of a model and its variance (Fig. 18). This is at the core of the intuition for why the traditional view of bias-variance as a tradeoff does not hold in all cases.

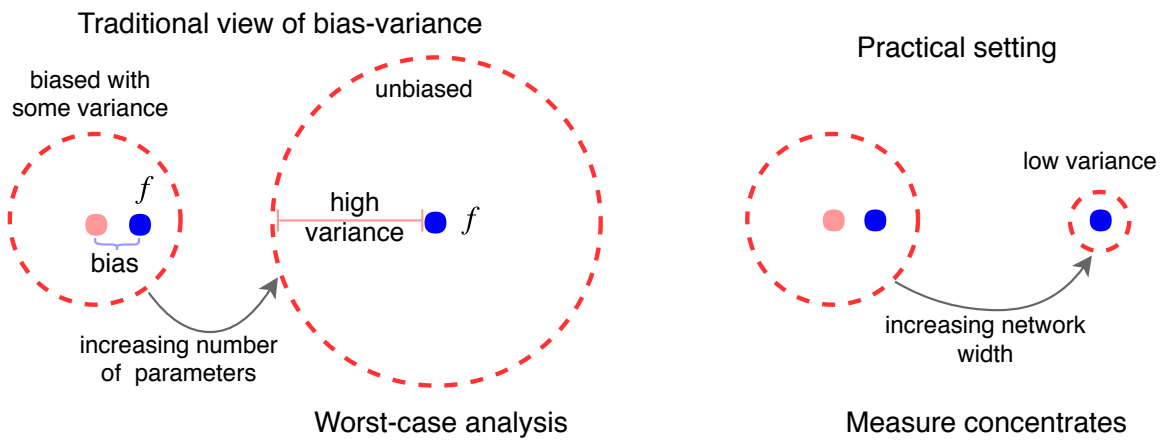


Figure 18: The dotted red circle depicts a cartoon version of the ϵ-hypothesis class of the learner. The left side reflects common intuition, as informed by the bias-variance tradeoff and worst-case analysis from statistical learning theory. The right side reflects our view that variance can decrease with network width.

G. Common intuitions from impactful works

"Neural Networks and the Bias/Variance Dilemma" from (Geman et al., 1992): "How big a network should we employ? A small network, with say one hidden unit, is likely to be biased, since the repertoire of available functions spanned by

$f(x; w)$ over allowable weights will in this case be quite limited. If the true regression is poorly approximated within this class, there will necessarily be a substantial bias. On the other hand, if we overparameterize, via a large number of hidden units and associated weights, then the bias will be reduced (indeed, with enough weights and hidden units, the network will interpolate the data), but there is then the danger of a significant variance contribution to the mean-squared error. (This may actually be mitigated by incomplete convergence of the minimization algorithm, as we shall see in Section 3.5.5.)"

"An Overview of Statistical Learning Theory" from (Vapnik, 1999): "To avoid over fitting (to get a small confidence interval) one has to construct networks with small VC-dimension."

"Stability and Generalization" from Bousquet & Elisseeff (2002): "It has long been known that when trying to estimate an unknown function from data, one needs to find a tradeoff between bias and variance. Indeed, on one hand, it is natural to use the largest model in order to be able to approximate any function, while on the other hand, if the model is too large, then the estimation of the best function in the model will be harder given a restricted amount of data." Footnote: "We deliberately do not provide a precise definition of bias and variance and resort to common intuition about these notions."

Pattern Recognition and Machine Learning from Bishop (2006): "Our goal is to minimize the expected loss, which we have decomposed into the sum of a (squared) bias, a variance, and a constant noise term. As we shall see, there is a tradeoff between bias and variance, with very flexible models having low bias and high variance, and relatively rigid models having high bias and low variance."

"Understanding the Bias-Variance Tradeoff" from Fortmann-Roe (2012): "At its root, dealing with bias and variance is really about dealing with over- and under-fitting. Bias is reduced and variance is increased in relation to model complexity. As more and more parameters are added to a model, the complexity of the model rises and variance becomes our primary concern while bias steadily falls. For example, as more polynomial terms are added to a linear regression, the greater the resulting model's complexity will be."

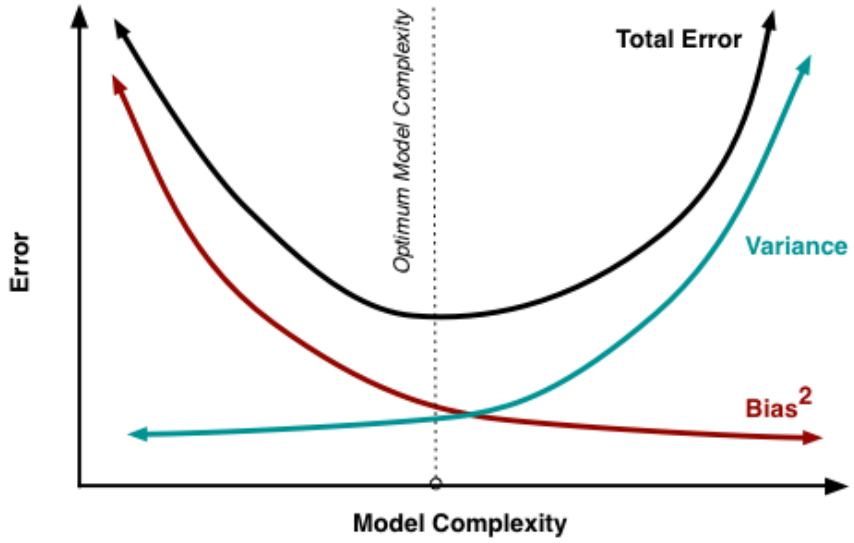


Figure 19: Illustration of common intuition for bias-variance tradeoff (Fortmann-Roe, 2012)