Double-descent phenomenon in deep learning

Lu Lu

May 22, 2020 @ Crunch Seminar

Classical bias–variance trade-off

- Unknown target function: $y = f(x)$
- A training set: $D = \{(x_1, y_1), \ldots, (x_n, y_n)\}$
- Find a $\hat{f}(x; D)$ by minimizing $\frac{1}{n} \sum_i (y_i - \hat{f}(x_i; D))^2$

Expected error (over different choices of D) on an unseen sample x:

$$\mathbb{E}_D[(y - \hat{f}(x; D))^2] = (\mathrm{Bias}_D[\hat{f}(x; D)])^2 + \mathrm{Var}_D[\hat{f}(x; D)]$$

where

$$\mathrm{Bias}_D[\hat{f}(x; D)] = \mathbb{E}_D[\hat{f}(x; D)] - f(x)$$

$$\mathrm{Var}_D[\hat{f}(x; D)] = \mathbb{E}_D[\hat{f}(x; D)^2] - \mathbb{E}_D[\hat{f}(x; D)]^2$$

Classical bias–variance trade-off

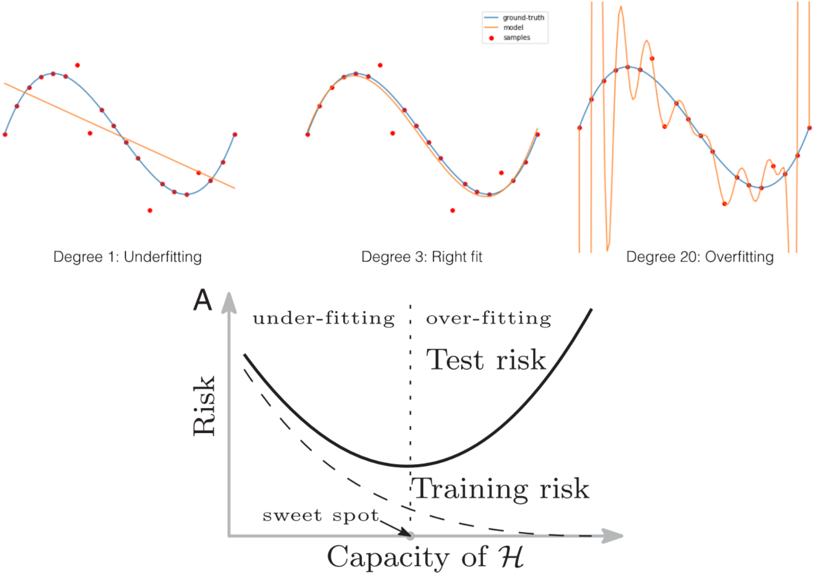
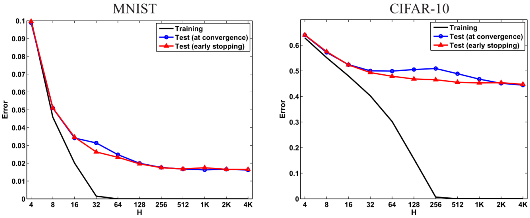


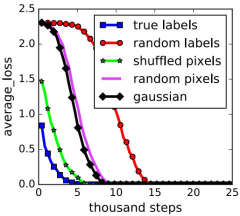
Figure 1: U-shaped curve.

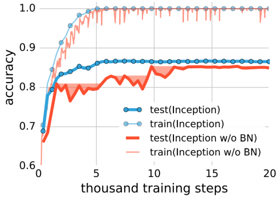
Generalization puzzle

In practice, neural networks learned by SGD:



Shallow NN. Neyshabur et al., *ICLR Workshop*, 2015.





(b) Inception on CIFAR10

No regularizers. Zhang et al., *ICLR*, 2017.

Generalization puzzle

Neural networks learned by SGD have

- far more model parameters than the number of samples
- sufficient capacity to memorize random labels
- **small generalization gap** between training and test

Bias–variance trade-off analysis failed!

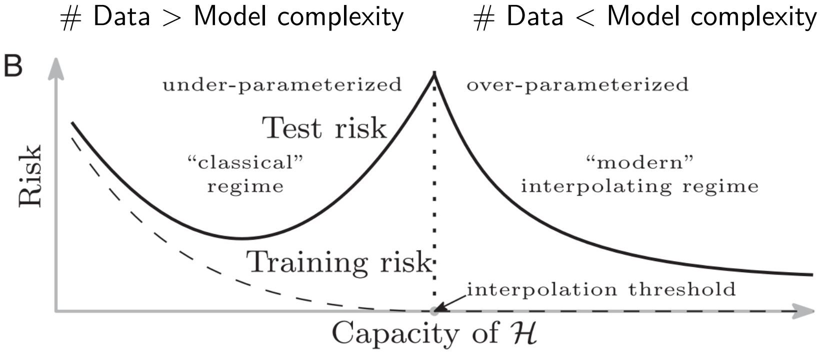
Possible explanations

Since 2017,

1. VC dimension/Rademacher complexity $\rightarrow$ low "complexity", e.g., path-norm, margin-based bounds, Fisher-Rao norm
2. good properties of SGD, e.g., stability, robustness, implicit biases/regularization
3. overparameterization
4. compression
5. Fourier analysis
6. "double descent" risk curve
7. PAC-Bayesian framework
8. information bottleneck
9. data-dependent analysis
10. properties of the trained neural networks

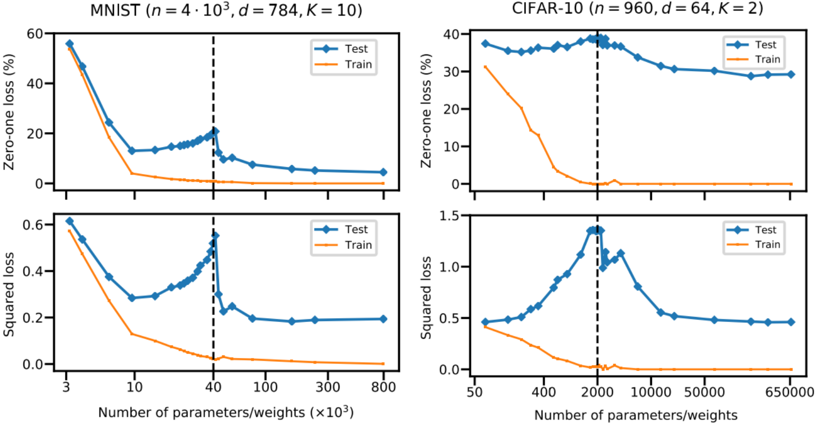
See [Jin et al., *arXiv:1905.11427*, 2019].

Double-descent phenomenon



Belkin et al., *PNAS*, 2019.

Experiment: A fully connected shallow neural network, classification



Belkin et al., *PNAS*, 2019.

More experiments on classification

Nakkiran et al., *ICLR*, 2019.

- Dataset: CIFAR 10, CIFAR 100, IWSLT'14 de-en, WMT'14 en-fr
- Architecture: CNN, ResNet, transformer
- Optimizers: SGD, SGD+Momentum, Adam

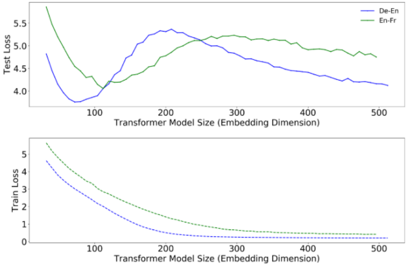
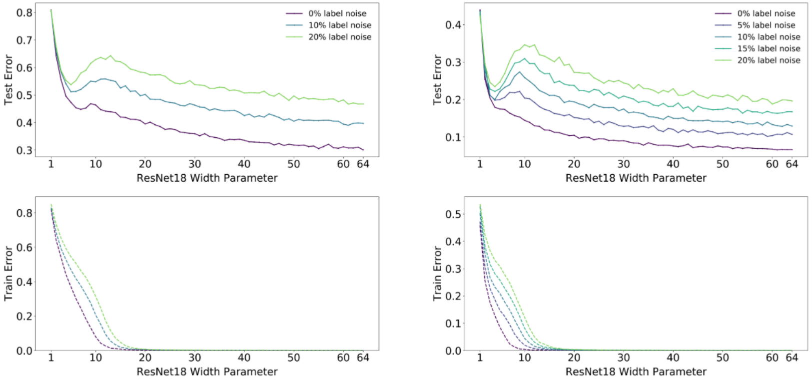


Figure 8: **Transformers on language translation tasks:** Multi-head-attention encoder-decoder Transformer model trained for 80k gradient steps with labeled smoothed cross-entropy loss on IWSLT'14 German-to-English (160K sentences) and WMT'14 English-to-French (subsampled to 200K sentences) dataset. Test loss is measured as per-token perplexity.

More experiments on classification



(a) **CIFAR-100.** There is a peak in test error even with no label noise.

(b) **CIFAR-10.** There is a "plateau" in test error around the interpolation point with no label noise, which develops into a peak for added label noise.

Figure 4: **Model-wise double descent for ResNet18s.** Trained on CIFAR-100 and CIFAR-10, with varying label noise. Optimized using Adam with LR 0.0001 for 4K epochs, and data-augmentation.

Epoch-wise double descent

Conventional wisdom suggests that training is split into two phases:

1. learning phase: the network learns a function with a small generalization gap
2. overfitting phase: the network starts to over-fit the data leading to an increase in test error

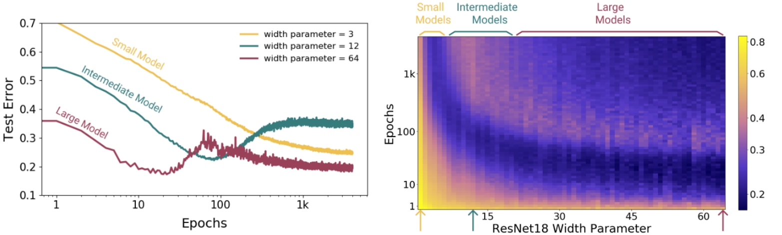


Figure 9: **Left:** Training dynamics for models in three regimes. Models are ResNet18s on CIFAR10 with 20% label noise, trained using Adam with learning rate 0.0001, and data augmentation. **Right:** Test error over (Model size $\times$ Epochs). Three slices of this plot are shown on the left.

Sample-wise non-monotonicity

"more data is always better"?

- increasing the number of samples has the effect of shifting this peak to the right
- more data actually hurts test performance

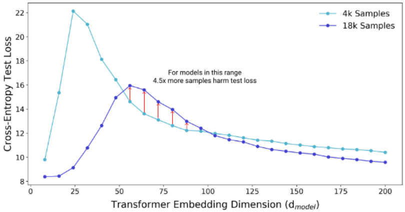


Figure 3: Test loss (per-token perplexity) as a function of Transformer model size (embedding dimension d_{model}) on language translation (IWSLT'14 German-to-English). The curve for 18k samples is generally lower than the one for 4k samples, but also shifted to the right, since fitting 18k samples requires a larger model. Thus, for some models, the performance for 18k samples is *worse* than for 4k samples.

Effective model complexity (EMC)

- Test error peaks around the point where EMC matches the number of samples

EMC of *a training procedure* $\mathcal{T}$, w.r.t. *distribution* $\mathcal{D}$ and *parameter* $\epsilon > 0$, is defined as:

$$\text{EMC}_{\mathcal{D},\epsilon}(\mathcal{T}) = \max\{n | \mathbb{E}_{S\sim\mathcal{D}^n}[\text{Error}_S(\mathcal{T}(S))] \leq \epsilon\}$$

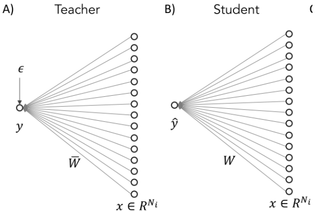
where $\text{Error}_S(M)$ is the mean error of *model* M on *train samples* S.

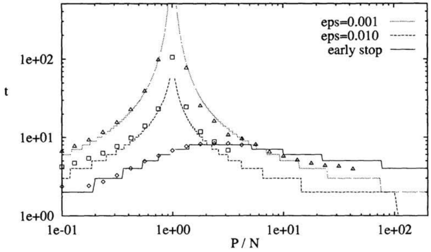
- model architecture
- training procedure: increasing training time will increase the EMC
- data size/distribution

Regression: Shallow linear neural network

A student-teacher scenario.

$N = 200$. P: dataset size

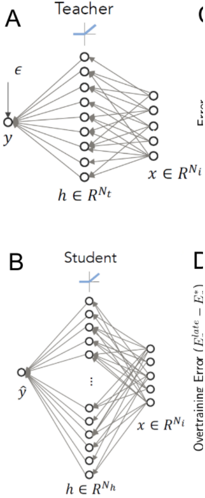


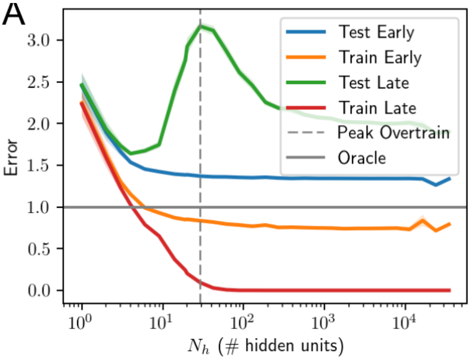


Bos & Opper, *NIPS*, 1997.

Regression: Shallow neural network

$N_i = 15$, $N_t = 30$, $P = 300$ training samples, $SNR = \sigma_w^2/\sigma_\epsilon^2 = 1$



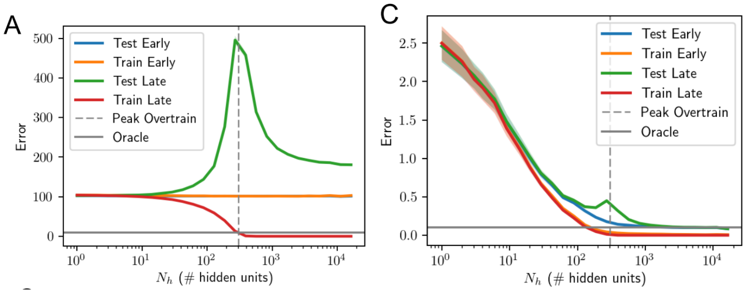


Advani & Saxe, *arXiv:1710.03667*, 2017.

Regression: Shallow neural network

Effects of SNR.

Left: SNR=0.01; Right: SNR=10.



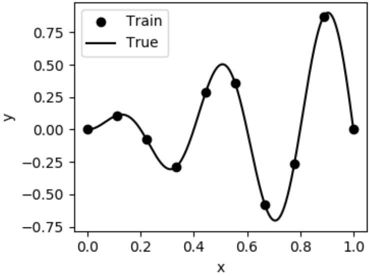
Advani & Saxe, *arXiv:1710.03667*, 2017.

Are you 100% convinced?

- Curves are too good to be true
- Error bars (due to random initialization/optimization) are usually missing
- Curve points are usually sparse
- Details are missing, e.g., achieve close to *zero* (?) training error
- Regression experiments are not realistic

My regression experiments

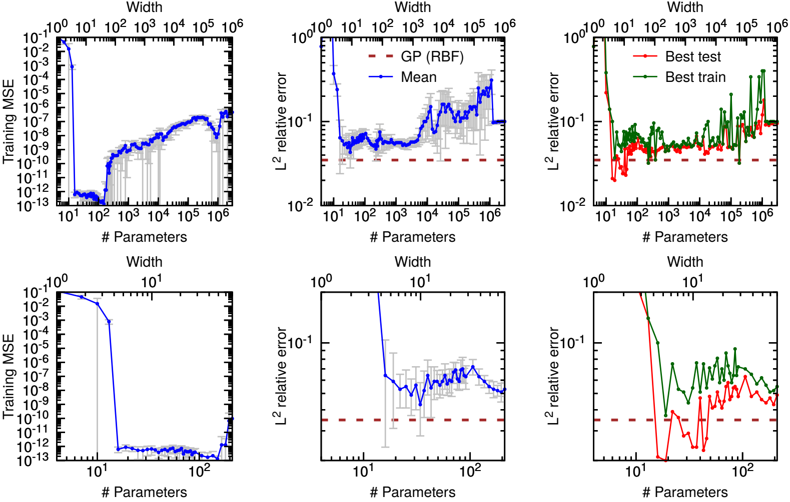
$f(x) = x \sin(5\pi x),\ x \in [0, 1]$



- shallow neural network, i.e., one hidden layer
- activation: tanh
- initialization: Glorot normal
- Adam, learning rate 5×10^{-5}, 3 million epochs

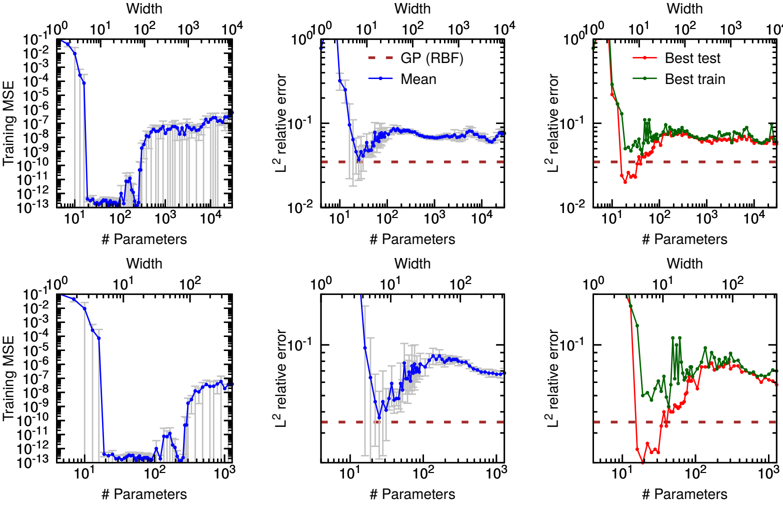
My regression experiments

Do we have double descent?



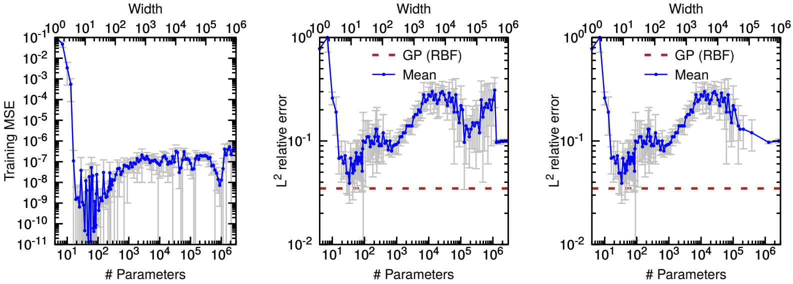
My regression experiments

learning rate 1×10^{-5}: consistent peak location with a more clear trend



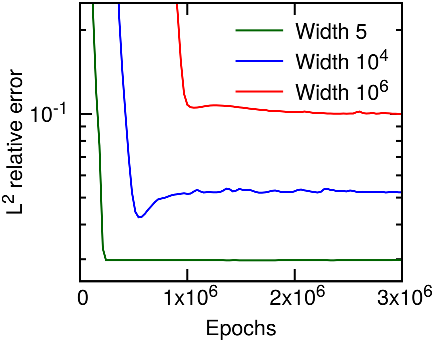
My regression experiments

Warning: What if learning rate and epochs are not fixed?



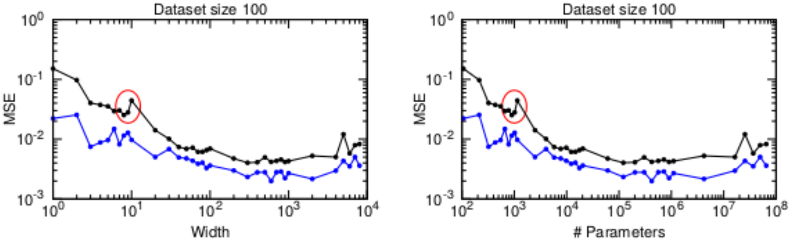
My regression experiments

Epoch-wise double descent? Yes.



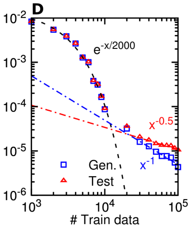
DeepONet [Lu et al., *arXiv:1910.03193*, 2019]

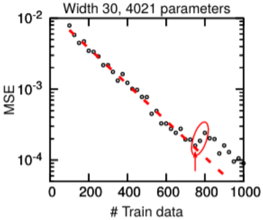
Model-wise double descent



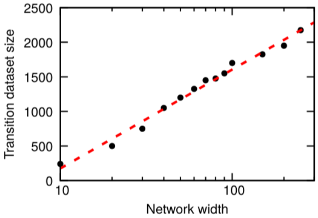
DeepONet [Lu et al., *arXiv:1910.03193*, 2019]

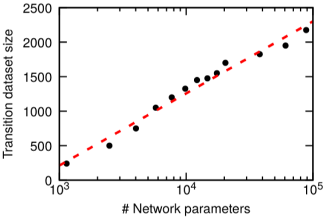
Sample-wise non-monotonicity





Transition size $\sim$ log (# network parameters)



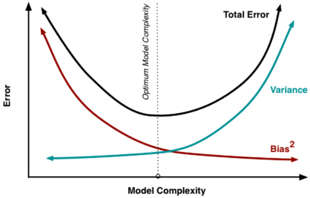


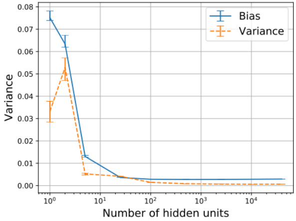
Double-descent analysis

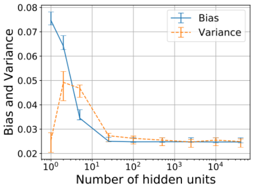
- Linear regression models (with random features)
  - Bos & Opper, *NIPS*, 1997.
  - Advani & Saxe, *arXiv:1710.03667*, 2017.
  - Belkin et al., *arXiv:1903.07571*, 2019.
  - Bibas et al., *IEEE ISIT*, 2019.
  - Hastie et al., *arXiv:1903.08560*, 2019.
  - Mei & Montanari, *arXiv:1908.05355*, 2019.
  - Mitra, *arXiv:1906.03667*, 2019.
  - Bartlett et al., *PNAS*, 2020.
  - Muthukumar et al., *IEEE JSAIT*, 2020.
- Jamming transition
  - Spigler et al., *J. Phys. A*, 2019.
  - Geiger et al., *Phys. Rev. E*, 2019.
  - Geiger et al., *J. Stat. Mech.*, 2020.

Yet another possibility [Neal et al., *arXiv:1810.08591*, 2018]

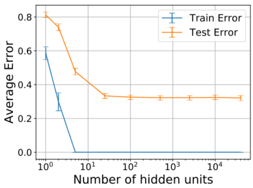
"Bias-variance trade-off" is still "true", but variance can decrease!



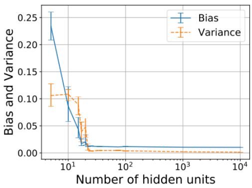




(a) Variance decreases with width, even in the small MNIST setting.



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



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

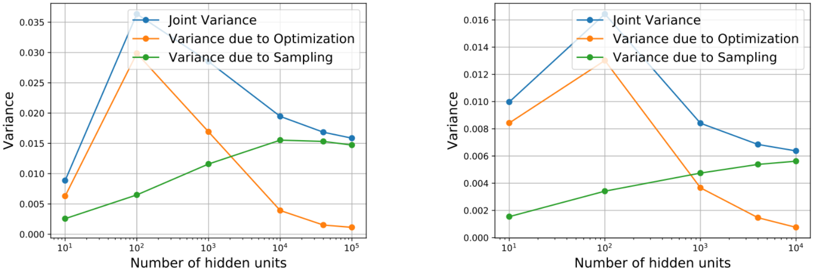
Variance decomposition

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

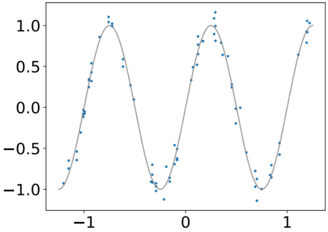
A new decomposition of the variance

- Variance due to sampling: $\mathrm{Var}_S(\mathbb{E}_O[h_\theta(x)|S])$
- Variance due to optimization: $\mathbb{E}_S[\mathrm{Var}_O(h_\theta(x)|S)]$

CIFAR10 (left) and on SVHN (right)



Variance decomposition: Regression



80 data points

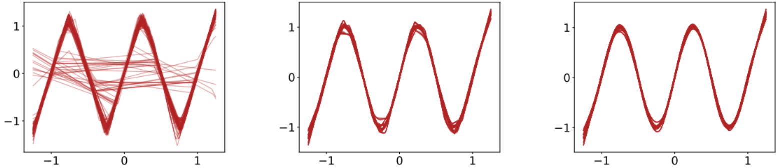
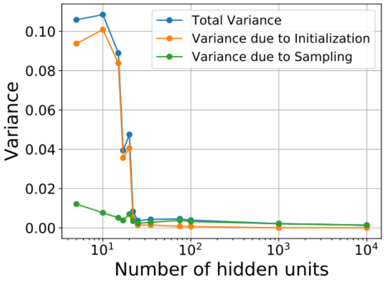
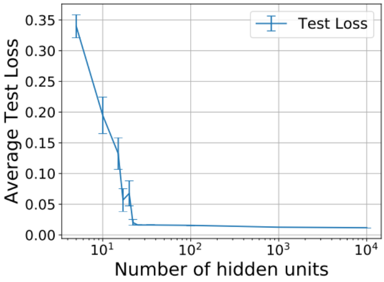


Figure 4: Visualization of the 100 different learned functions of single hidden layer neural networks of widths 15, 1000, and 10000 (from left to right) on the task of learning a sinusoid. The learned functions are increasingly similar with width, suggesting decreasing variance. More in Appendix B.7.

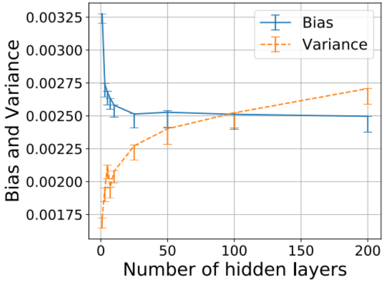
Variance decomposition: Regression

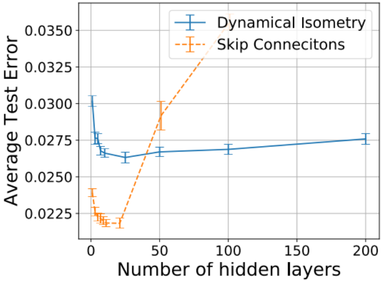
- Variance due to sampling should increase, but not here.
- Variance due to optimization should increase and then decrease, but not clear here.
- Double descent of test loss? Not clear.
- Points are too sparse.





Variance decomposition: Not for depth





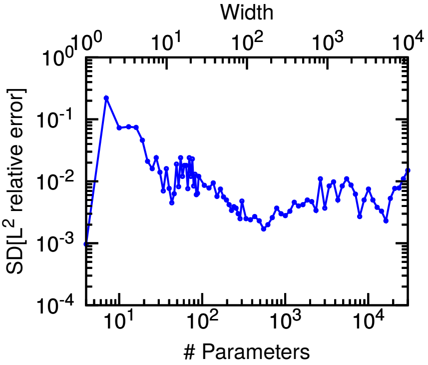
Variance decomposition?

Experiments are not convincing enough.

- Experiments are limited
- Hyperparameters (learning rate) changes between experiments

My regression experiments

- Variance due to sampling: $\mathrm{Var}_S(\mathbb{E}_O[h_\theta(x)|S])$
- Variance due to optimization:
  $\mathbb{E}_S[\mathrm{Var}_O(h_\theta(x)|S)] = \mathrm{Var}_O(h_\theta(x)|S)$



Increase and then decrease? Yes and no.

Double-descent?

Promising, but still at an early stage.

In practice, don't rely on the double descent too much.

- No guarantee the second U is better than the first U
- To optimize large networks: much smaller learning rate, and much more epochs

Keep tuning your hyperparameters manually or automatically.