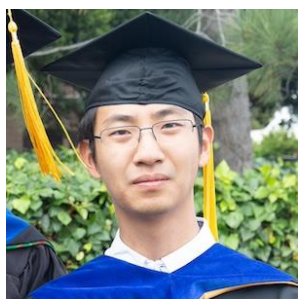


Generalization by Large Learning Rate and Weight Decay in overparameterized *neural networks*

Yu-Xiang Wang

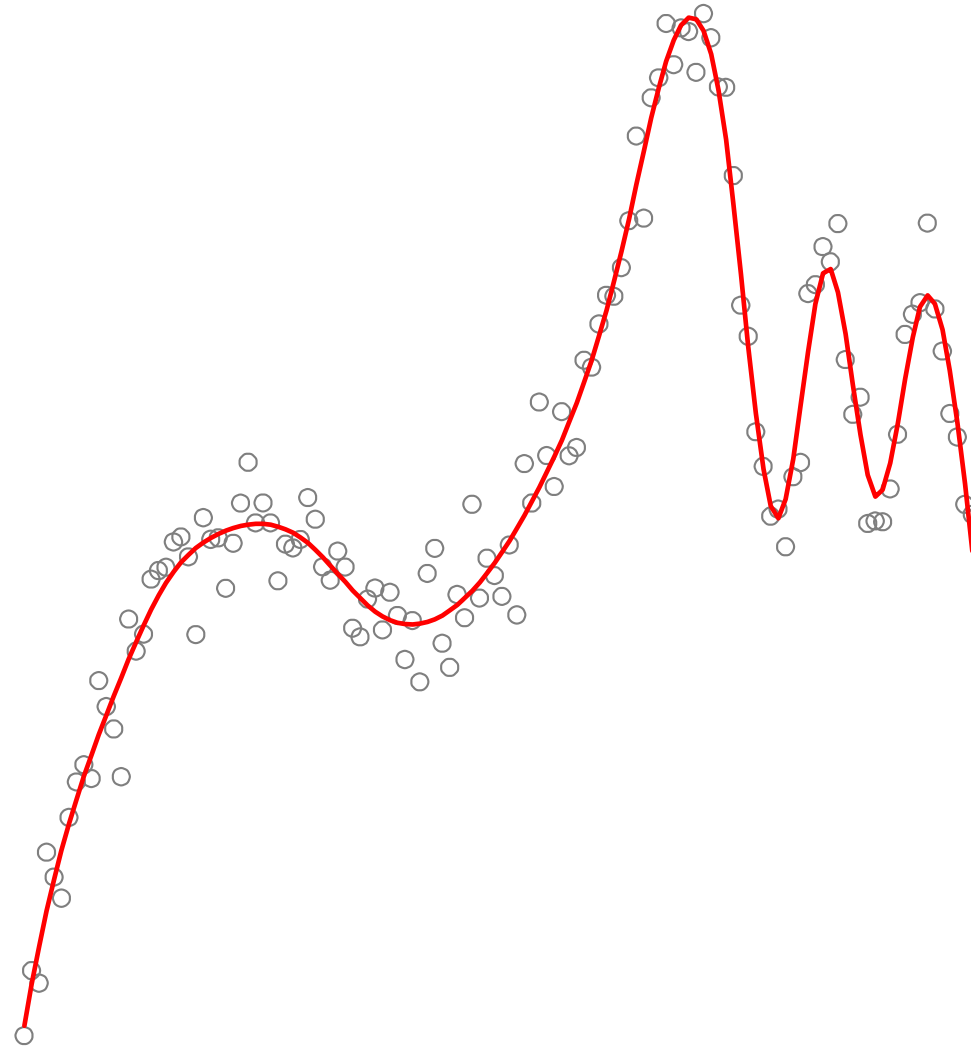
UCSD

Based on joint work with



Kaiqi Zhang, Dan Qiao, Tongtong Liang, Esha Singh, Zixuan Zhang, Minshuo Chen, Tuo Zhao, Daniel Soudry, Rahul Parhi

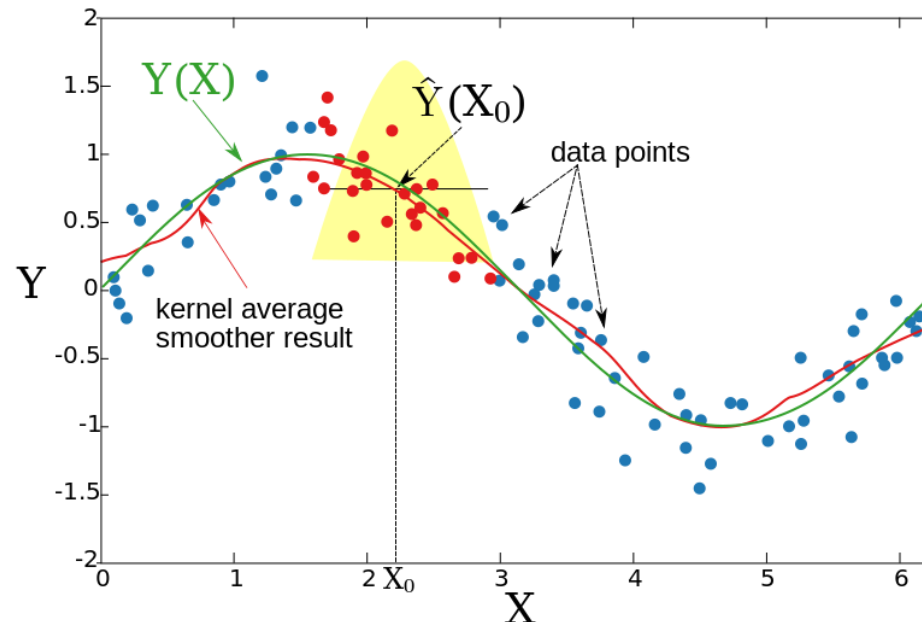
“Is deep learning is just curve fitting? ”



Judea Pearl
Turing Award Laureate
2011

Curve fitting isn't easy

- 60 years of associated literature
 - Since [Nadaraya, Watson, 1964] then Wahba, Donoho, Mammen, Van de Geer, Barron, Bartlett, Schölkopf etc...
 - **Methods:** Kernels, splines, local polynomials, Gaussian processes and RKHS, CART, MARS, neural networks



a.k.a., smoothing / filtering / denoising / signal processing / nonparametric regression.

Not whether you can “curve-fit” but *how well*
(*and how efficiently*) you can “curve-fit”

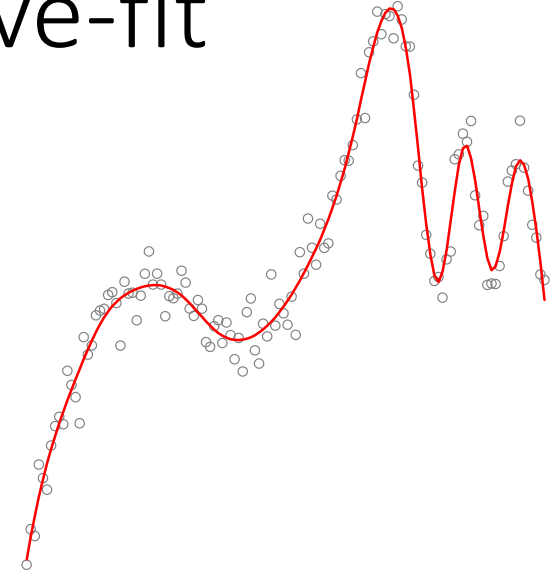
$$y_i = f(x_i) + \text{Noise for } i = 1, \dots, n.$$

Goal: Estimate the function using noisy data

$$(x_1, y_1), \dots, (x_n, y_n)$$

assume that $f \in \mathcal{F}$

$$\text{MSE}(\hat{f}) := \mathbb{E}_{\mathcal{D}_n} \frac{1}{n} \sum_{i=1}^n (\hat{f}(\mathbf{x}_i) - f_0(\mathbf{x}_i))^2.$$



- Approximation theory
- Estimation
(generalization) theory
- Optimality

Besides MSE, we will often consider
“Generalization” in a more agnostic setting

- iid samples from an unknown distribution

$$(x_1, y_1), \dots, (x_n, y_n) \sim \mathcal{D}^n$$

- (Prediction) loss of f , $\ell(f, (x, y)) = \frac{1}{2}(f(x) - y)^2$
- **Generalization gap** measures (expected) gap between training loss
test loss.

$$\left| \mathbb{E} \left[\frac{1}{n} \sum_{i=1}^n \ell(\hat{f}, (x_i, y_i)) \right] - \mathbb{E}_{(x,y) \sim \mathcal{D}} \left[\ell(\hat{f}, (x, y)) \right] \right|$$

Training loss

Test loss

“Curve fitting” is a perfect test-bed for understanding the fundamental properties of deep learning

- What function classes can NNs represent / estimate efficiently?
- Overparameterization
- Representation learning
- How does training algorithm / regularization affect generalization?

Why do Neural Networks work **better**?

- Universal function approximation ([Hornik et al, 1989](#))
 - But so are kernels and splines!
- Flexible representation and modelling language
 - So are graphical models / probabilistic programs
- Overparameterization
 - Neural Tangent Kernels ([Jacot et al., 2018](#); [Du et al. 2019](#); etc)
 - Interpolation regime / benign overfitting ([Belkin, 2019](#), [Bartlett et al. 2020](#), [Frei, 2022](#) and more)

The “adaptivity” hypothesis

- Neural networks aren’t stronger than classical methods in any specific problem
- But the standard practices of how people develop / train **DNNs enjoy strong adaptivity**
 - No need to carefully specify the problem
 - Automatically choose the right level of abstraction
 - Tune only standard hyperparameters (**e.g., Learning Rate, Weight Decay**).
 - They match the best classical methods on each problem

Remainder of the talk

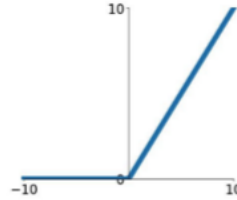
- Generalization by Large Learning Rate
- Generalization by weight decay
- Discussion

We consider two-Layer *overparameterized* *ReLU*-Neural Networks for “curve fitting”

$$\mathcal{F} = \left\{ f : \mathbb{R} \rightarrow \mathbb{R} \mid f(x) = \sum_{i=1}^k w_i^{(2)} \phi \left(w_i^{(1)} x + b_i^{(1)} \right) + b^{(2)} \right\}$$

- ReLU activation

ReLU
 $\max(0, x)$



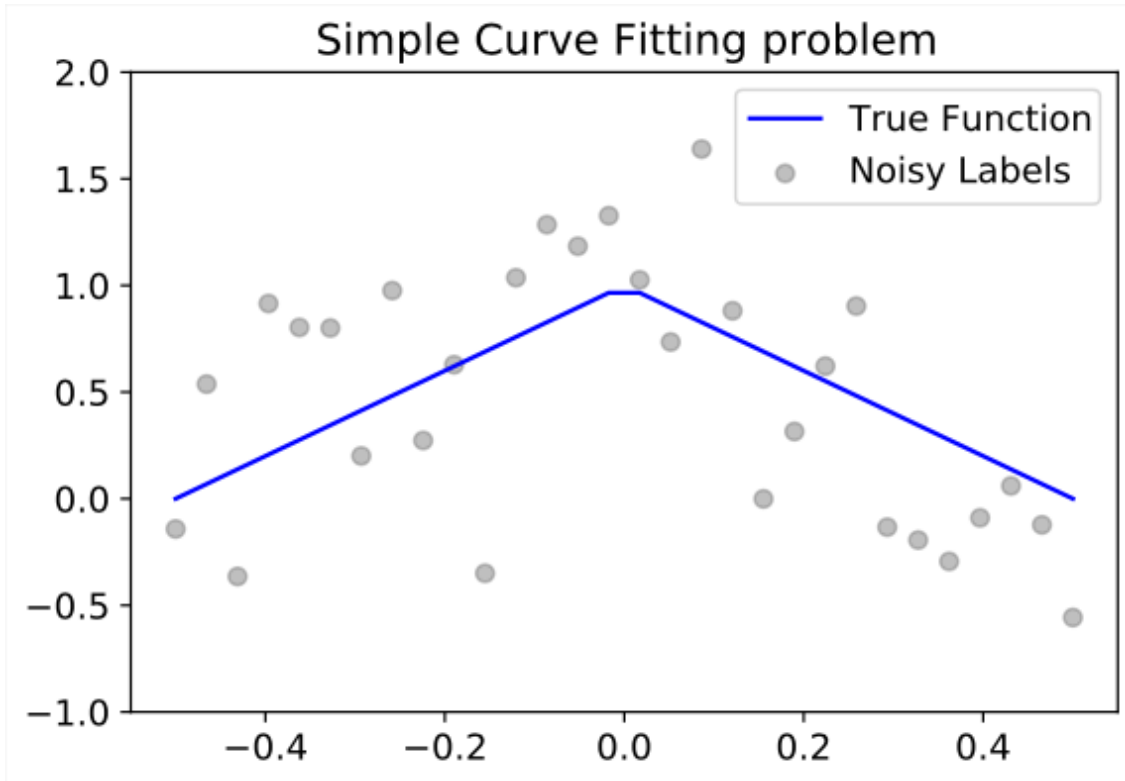
- Square loss $\mathcal{L}(\theta) = \frac{1}{2n} \sum_{i=1}^n (f_{\theta}(x_i) - y_i)^2$

- Let's train with gradient descent with no regularization.

$$\theta_{t+1} = \theta_t - \boxed{\eta} \nabla \mathcal{L}(\theta_t), \quad t \geq 0,$$

Step size (aka learning rate) parameter

Let's start with an example



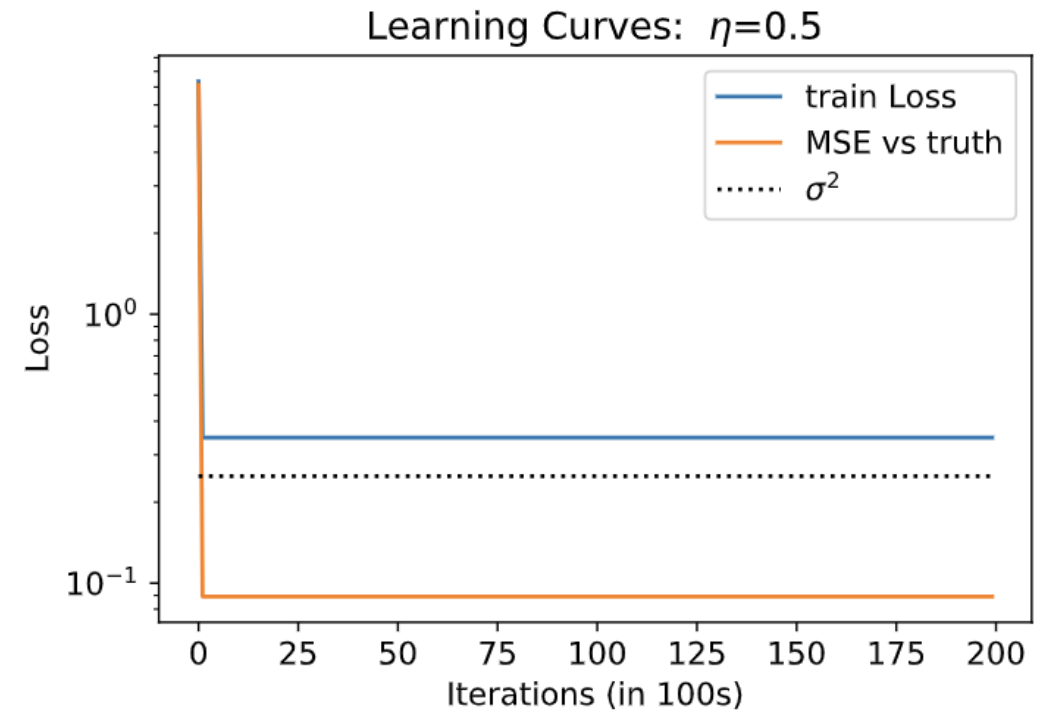
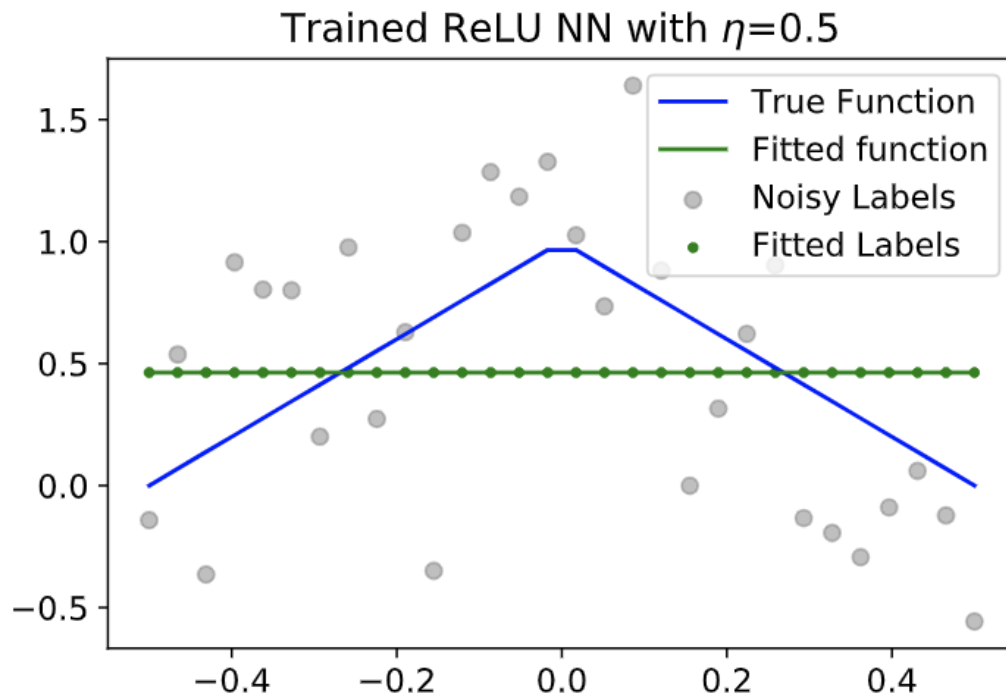
30 data points. Noisy labels.
2-Layer ReLU NN with 1000 neurons.
Minimizing **square loss**.
No regularization.

Which ones of the following are true?

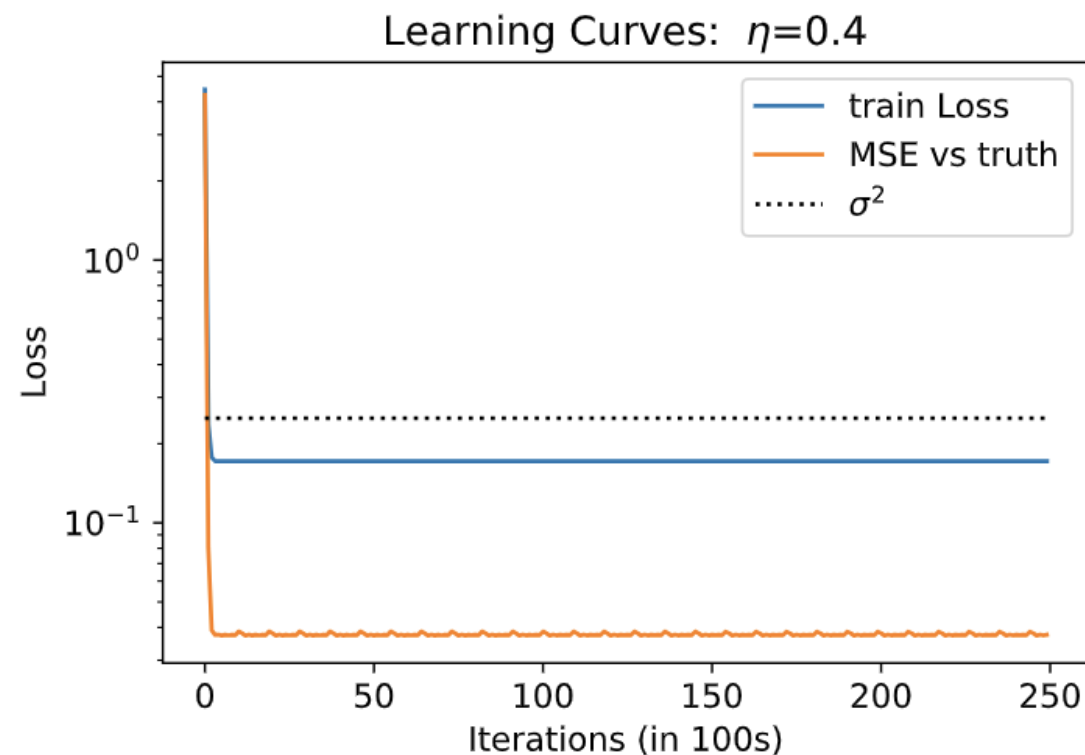
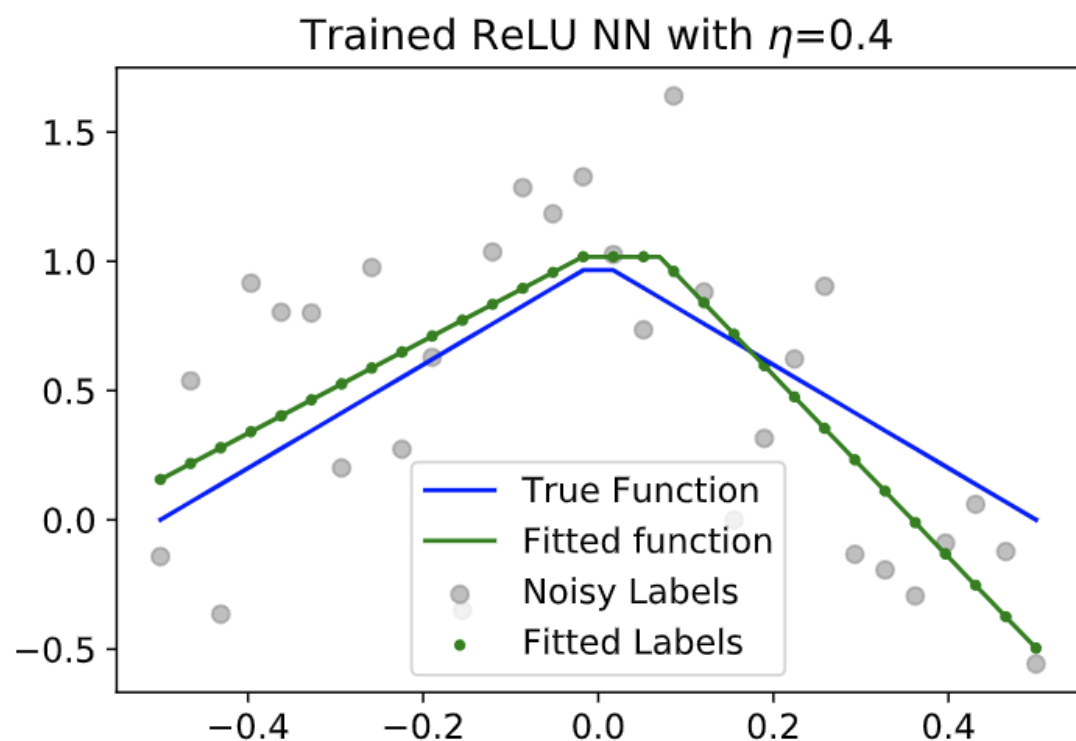
- A. Global optimal solution has 0-loss.
- B. Gradient descent finds global optimal (“interpolating”) solutions.
- C. GD solution satisfies “Benign overfitting”

Solution: (A) True. (B) It depends. (C) False.

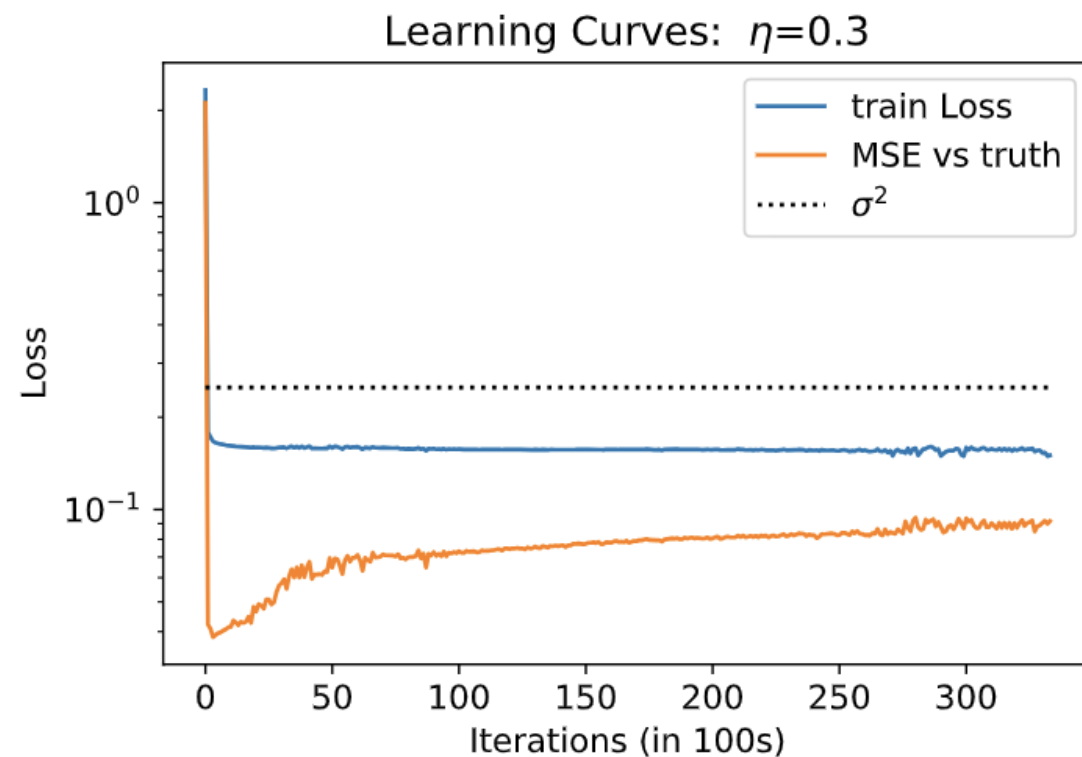
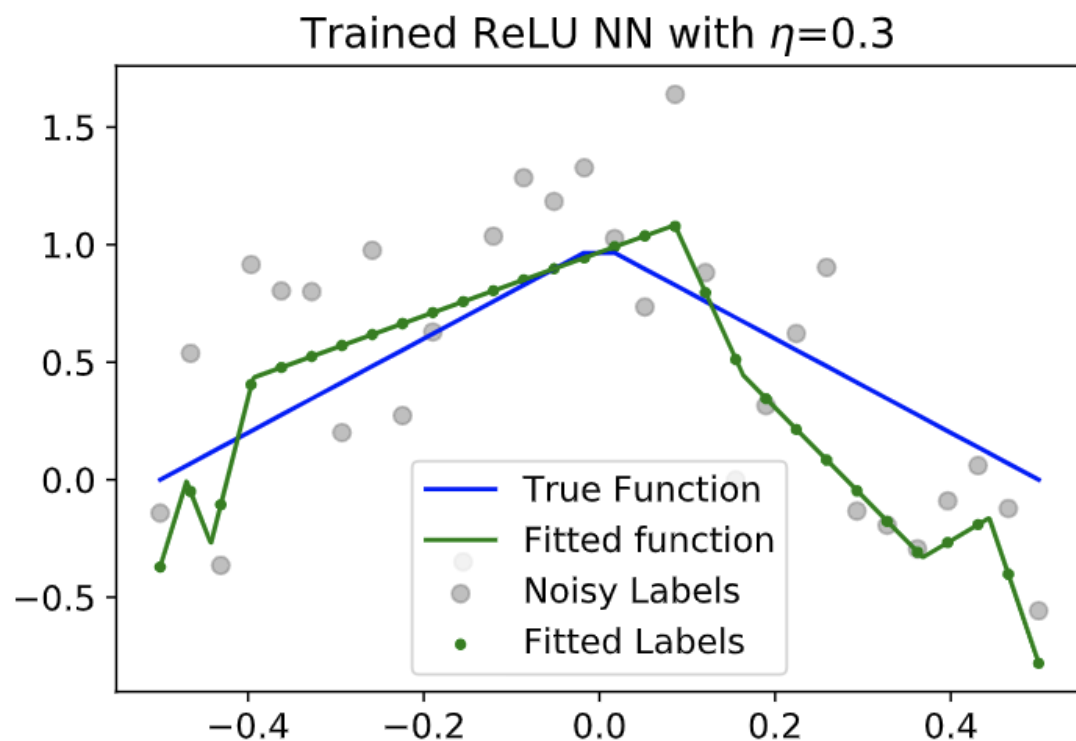
It's surprisingly difficult to find an interpolating solution with gradient descent.



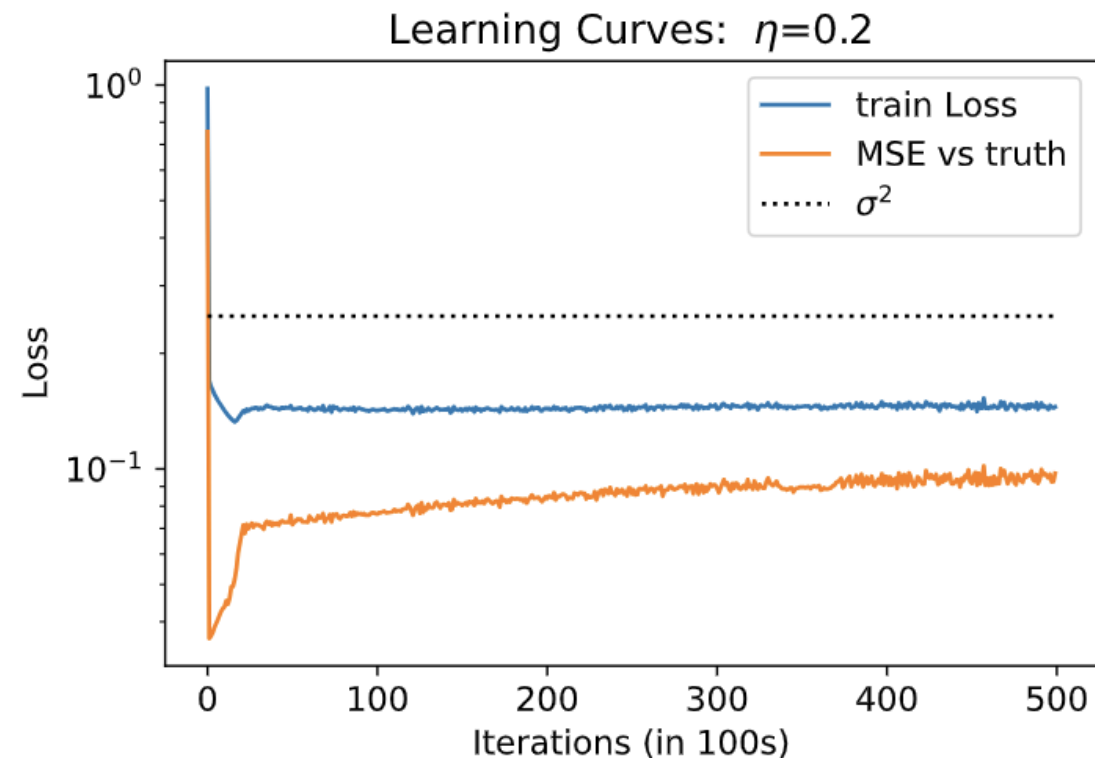
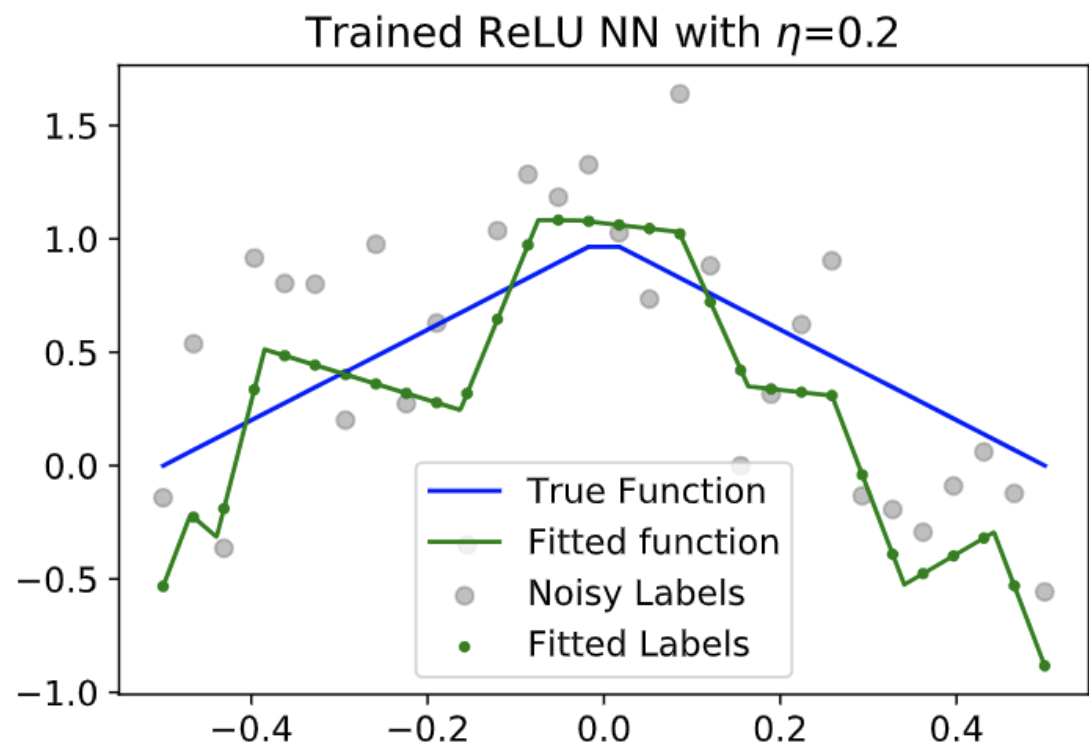
Learning Rate = 0.4



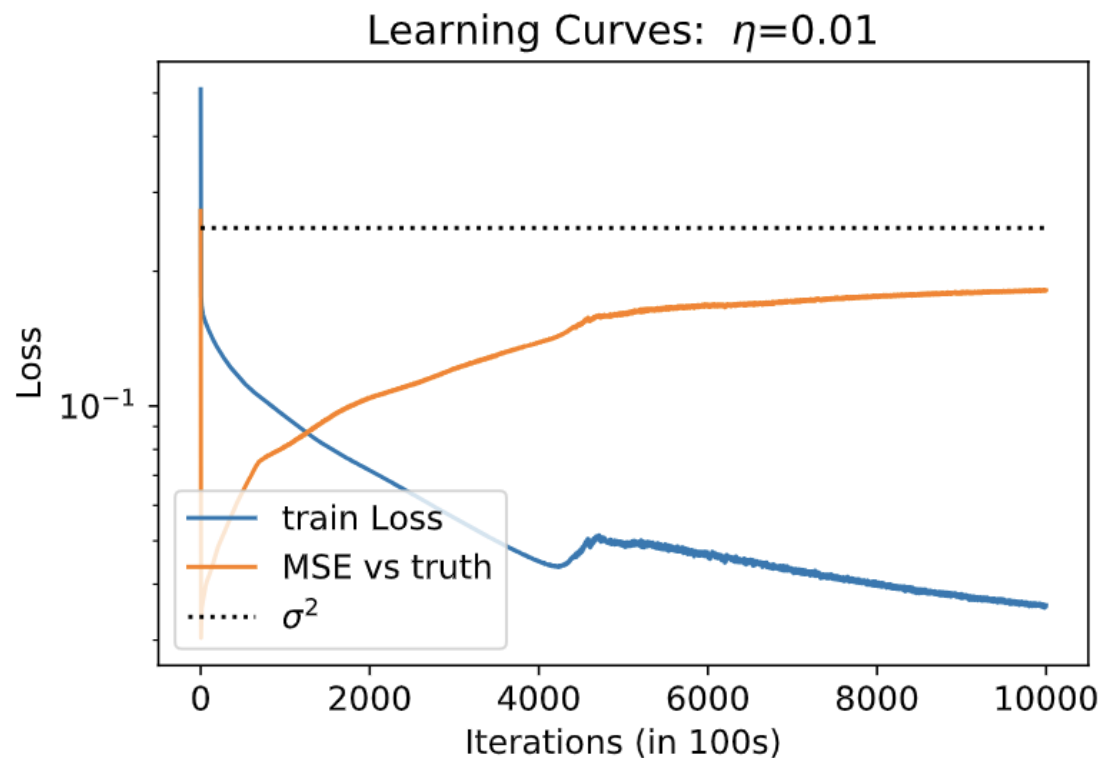
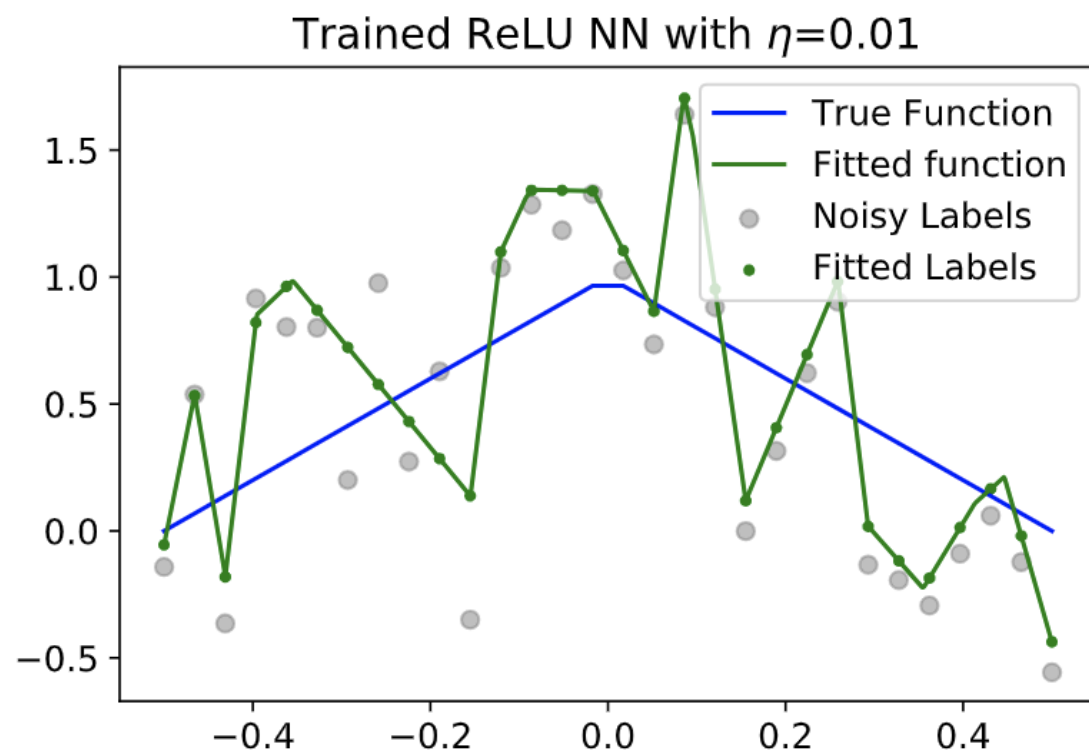
Learning Rate = 0.3



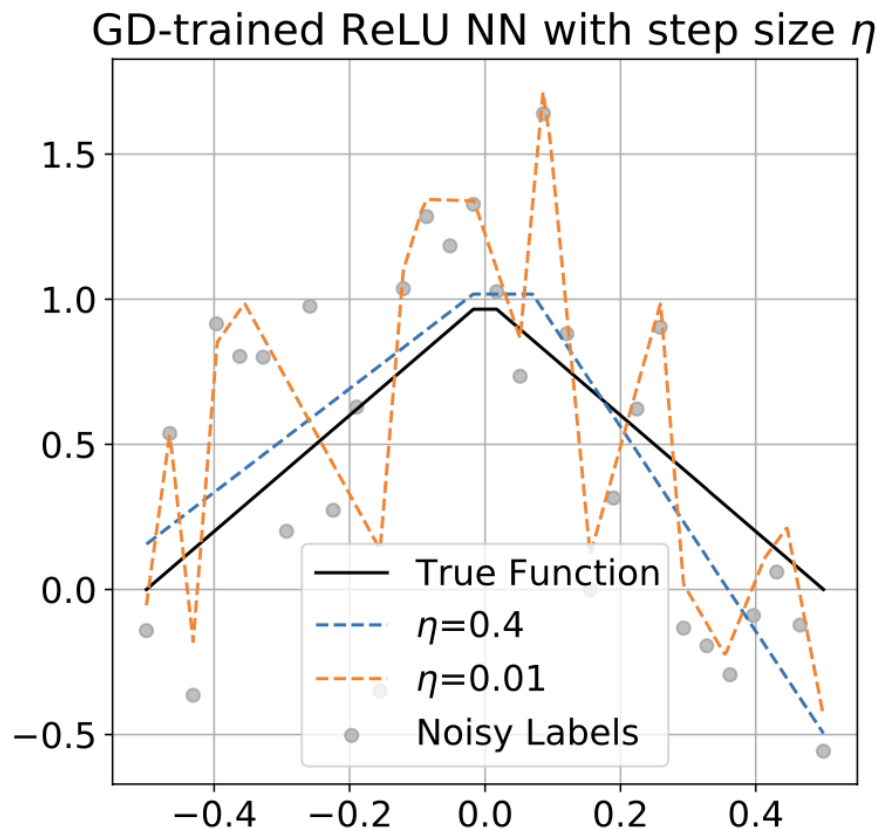
Learning Rate = 0.2



Learning Rate = 0.01



Observation: By tuning learning rates, we are effectively tuning the number of “active knots”. GD with larger learning rates learns simpler functions.

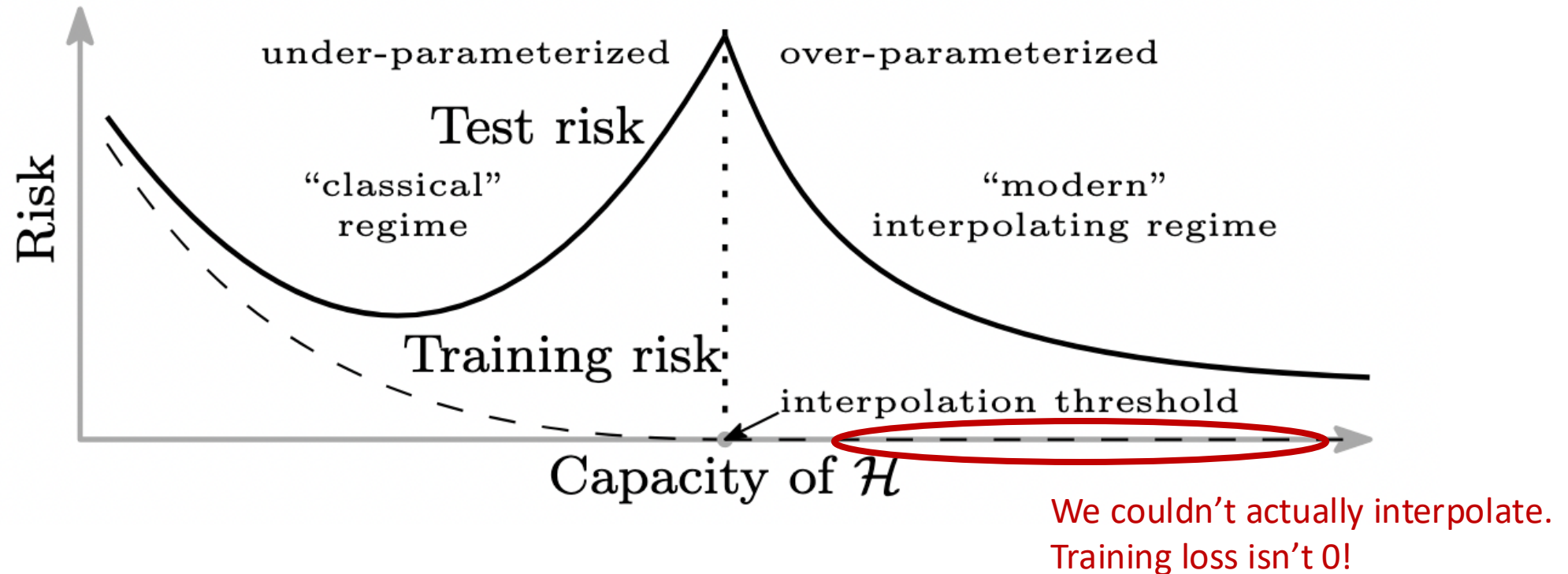


But how did “sparsity” emerge?

Is this a general phenomenon? Did we get lucky?

Can we prove anything about this phenomenon rigorously?

This is very different from existing theory of overparameterization: Double-Descent and NTK



(Figure from Belkin et al. (2018) "Double Descent")

Did we see Benign Overfitting? No!

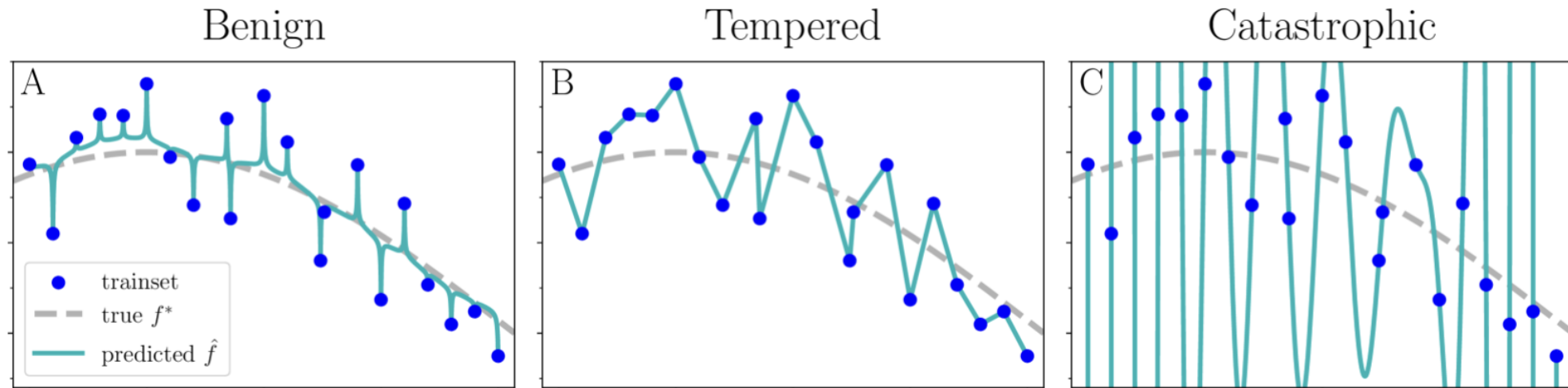
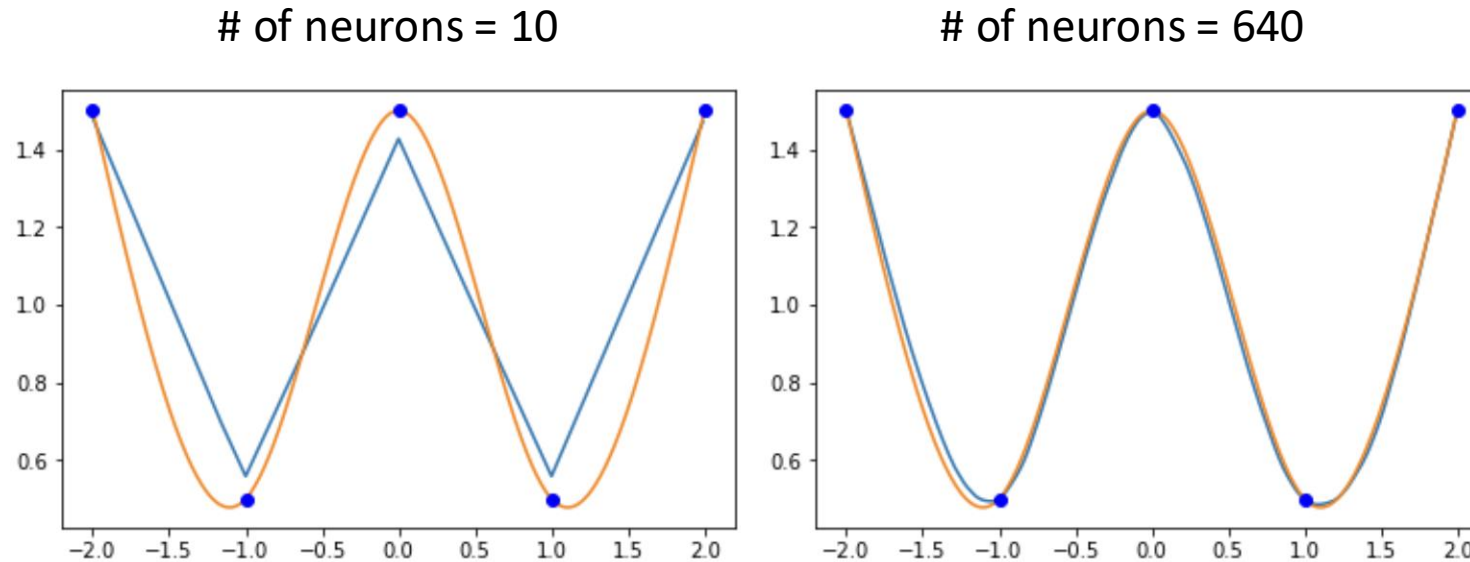


Figure 1: **As $n \rightarrow \infty$, interpolating methods can exhibit three types of overfitting.** (A) In *benign overfitting*, the predictor asymptotically approaches the ground-truth, Bayes-optimal function. Nadaraya-Watson kernel smoothing with a singular kernel, shown here, is asymptotically benign. (B) In *tempered overfitting*, the regime studied in this work, the predictor approaches a constant test risk greater than the Bayes-optimal risk. Piecewise-linear interpolation is asymptotically tempered. (C) In *catastrophic overfitting*, the predictor generalizes arbitrarily poorly. Rank- n polynomial interpolation is asymptotically catastrophic.

(Figure from Mallinar, Neil, et al. "Benign, tempered, or catastrophic: A taxonomy of overfitting." *arXiv preprint arXiv:2207.06569* (2022).)

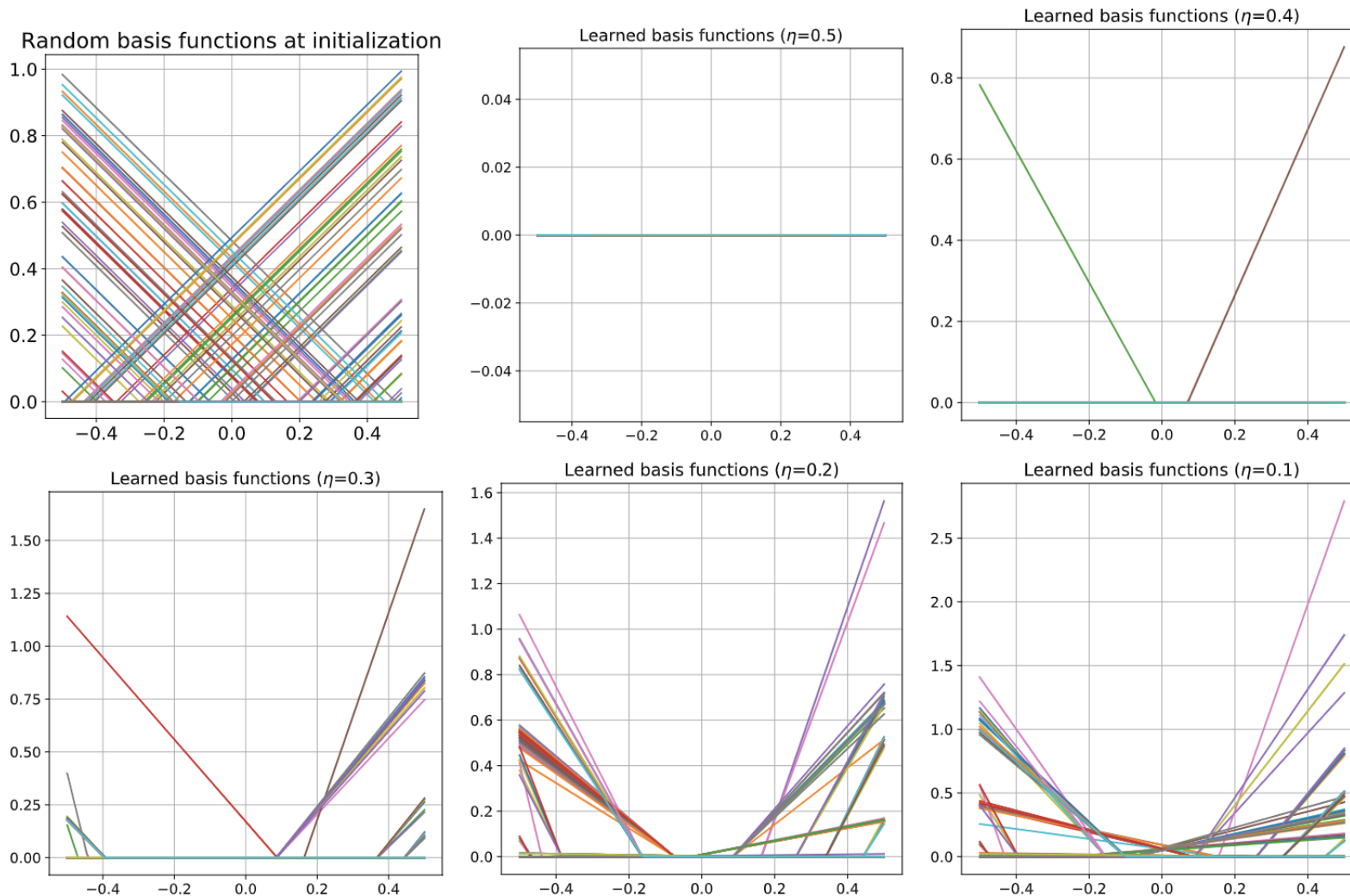
When we take the $\eta \rightarrow 0$, i.e., Gradient Flow, it does eventually interpolate as “cubic smoothing splines”



- Jin, H., & Montúfar, G. (2023). Implicit bias of gradient descent for mean squared error regression with two-layer wide neural networks. *Journal of Machine Learning Research*, 24(137), 1-97.

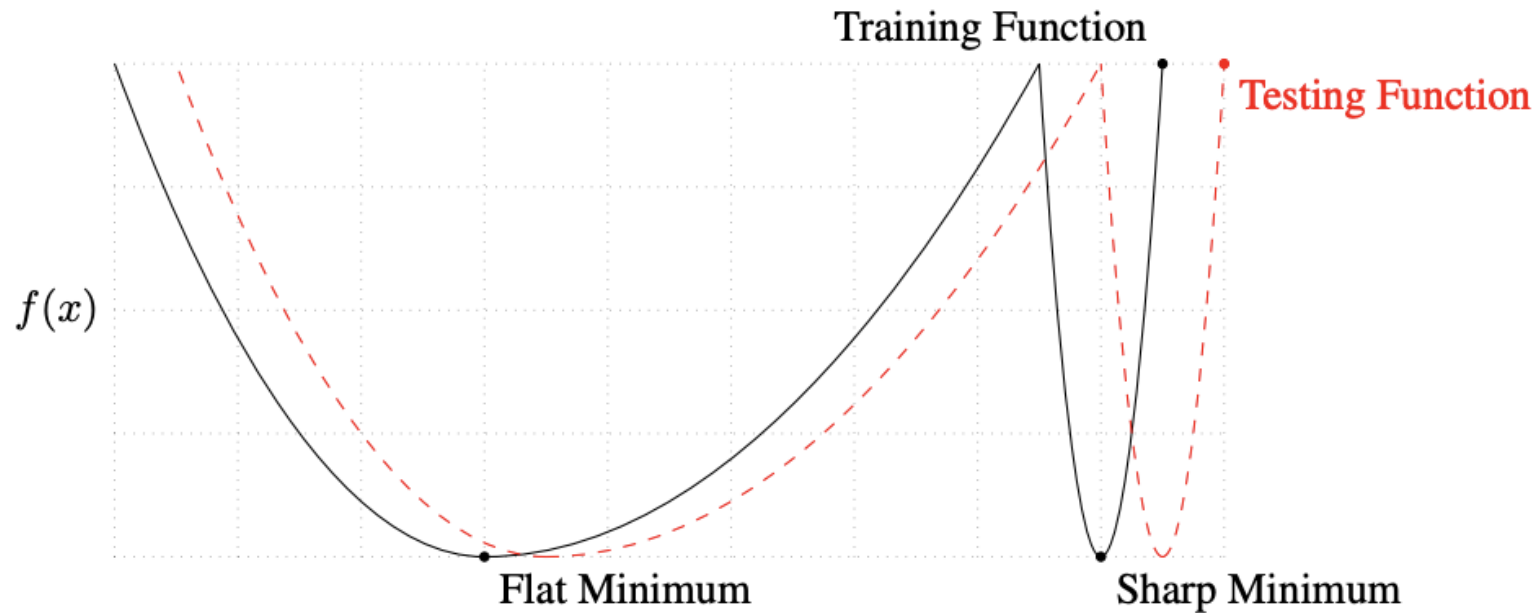
(See also new work for the same setting under “**Mirror Descent**” with $\eta \rightarrow 0$ by Liang and Montufar.)

Bigger issue: Existing theory does not capture any representation learning --- a prominent phenomenon in our example.



We need new theory to understand
the large learning rate regime!

Minima stability of Gradient Descent and SGD

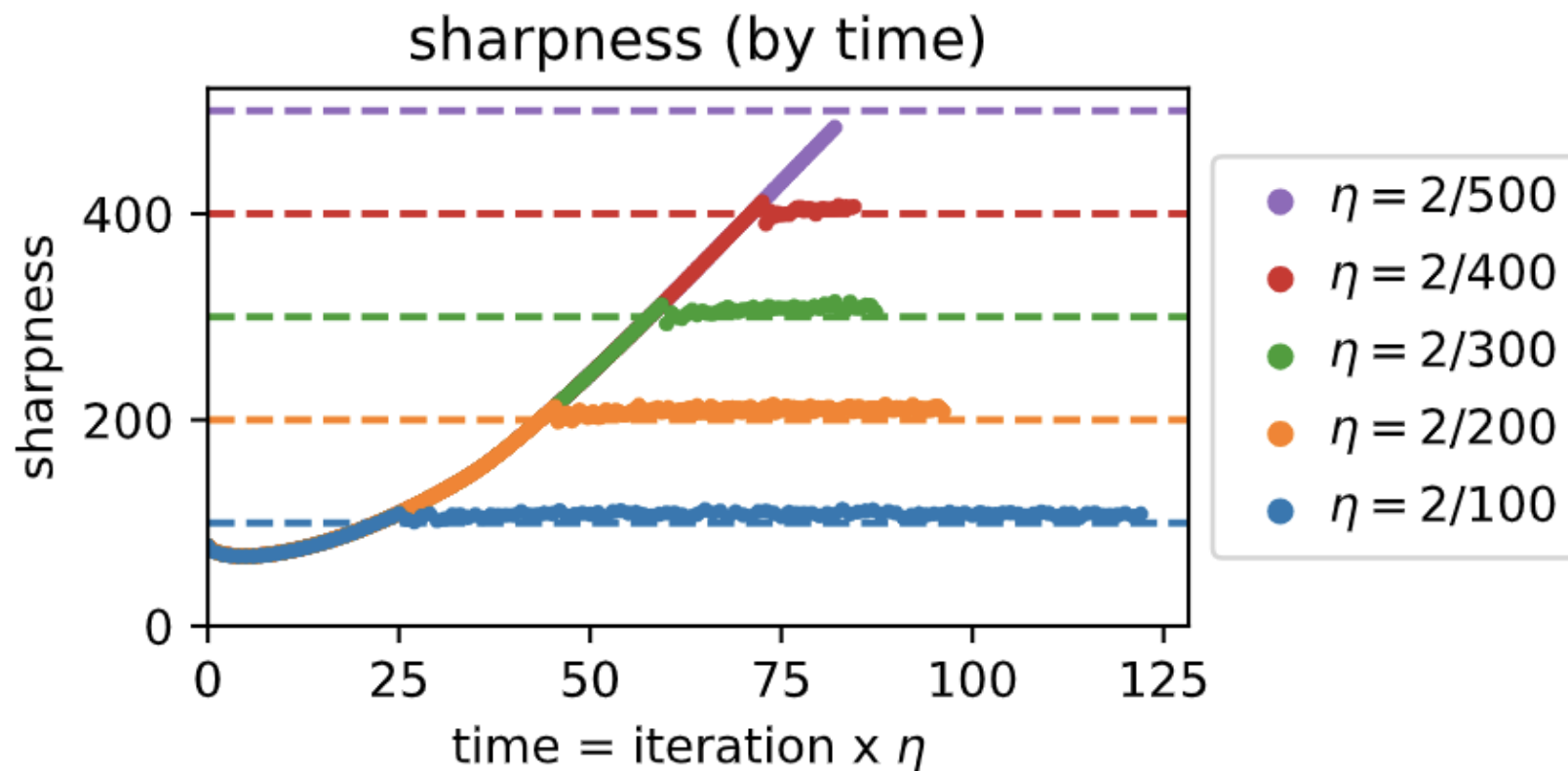


$$x_{t+1} = x_t - \eta \nabla \text{Loss}(x_t)$$

- Need learning rate $\eta \leq 2/\beta$ to converge for **β -smooth** loss function. **Stable minima** are those that GD can converge to.

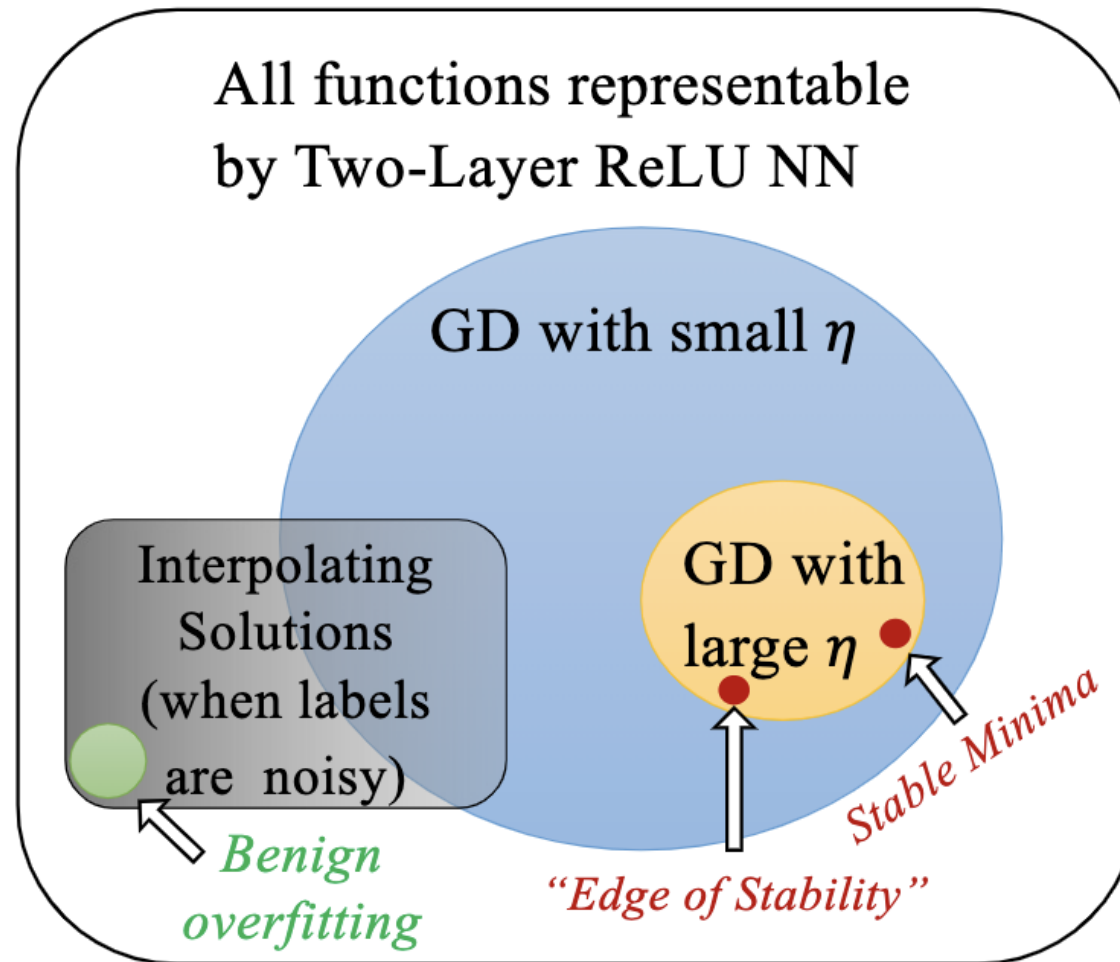
(Wu et al. 2018, Mulayoff et al. 2021)

“Edge of Stability” regime



Cohen et al (2021) Gradient Descent on Neural Networks Typically Occurs at the Edge of Stability

Gradient Descent selects a set of points it can converge to (or stabilize around).



What does this set of functions look like? **A Weighted TV1** class.

$$\begin{aligned} & \{f_\theta \mid \lambda_{\max}(\nabla^2 \mathcal{L}(\theta)) \leq 2/\eta\} \\ & \quad \subseteq \\ & \left\{f \mid \int |f''(x)|g(x)dx \leq \frac{2}{\eta}\right\} =: \text{TV}_g^{(1)}(C) \\ & \quad \text{where } C = 1/\eta + \tilde{O}(1) \end{aligned}$$

Flatness of Loss (in parameter space) implies a TV-type constraint (in function space)

Theorem (Qiao, Zhang, Singh, Soudry and W., 2024): Consider a ReLU activated two-layer NN / square loss. Let f be any function represented by the neural network parameterized by θ . Assume f_θ is twice differentiable at θ .

$$\int_{-x_{\max}}^{x_{\max}} |f''(x)|g(x)dx \leq \frac{\lambda_{\max}(\nabla_{\theta}^2 \mathcal{L}(\theta))}{2} - \frac{1}{2} + x_{\max} \sqrt{2\mathcal{L}(\theta)},$$

Assume data is coming from $y_i = f_0(x_i) + \text{noise}$, then w.h.p.

$$\int_{-x_{\max}}^{x_{\max}} |f''(x)|g(x)dx \leq \frac{\lambda_{\max}(\nabla_{\theta}^2 \mathcal{L}(\theta))}{2} - \frac{1}{2} + \tilde{O}\left(\sigma x_{\max} \cdot \min\left\{1, \sqrt{\frac{k}{n}}\right\}\right) + x_{\max} \sqrt{\text{MSE}(f)}.$$

- Tune learning rate => select smoothness of f
- Smoothness of f => Generalization bounds

Interpolating solutions have high curvature

- Theorem from the previous slide

$$\int_{-x_{\max}}^{x_{\max}} |f''(x)|g(x)dx \leq \frac{\lambda_{\max}(\nabla_{\theta}^2 \mathcal{L}(\theta))}{2} - \frac{1}{2} + x_{\max} \sqrt{2\mathcal{L}(\theta)},$$

- We prove that for any interpolating solution (noise level):

$$\int_{-x_{\max}}^{x_{\max}} |f''(x)|g(x)dx = \Omega \left(\sigma n \left[n - 24 \log \left(\frac{1}{\delta} \right) \right] \right),$$

- Implies that learning rate needs to be extremely small $O\left(\frac{1}{n^2\sigma}\right)$ to stably learn interpolating solutions.

Key proof idea: Hessian Decomposition

Interpolation helped (Mulayoff et al. 2021)

$$\nabla_{\theta}^2 (f_{\theta}(x) - y)^2 = \nabla_{\theta} f_{\theta}(x) \nabla_{\theta} f_{\theta}(x)^T + (f_{\theta}(x) - y) \nabla_{\theta}^2 f_{\theta}(x)$$

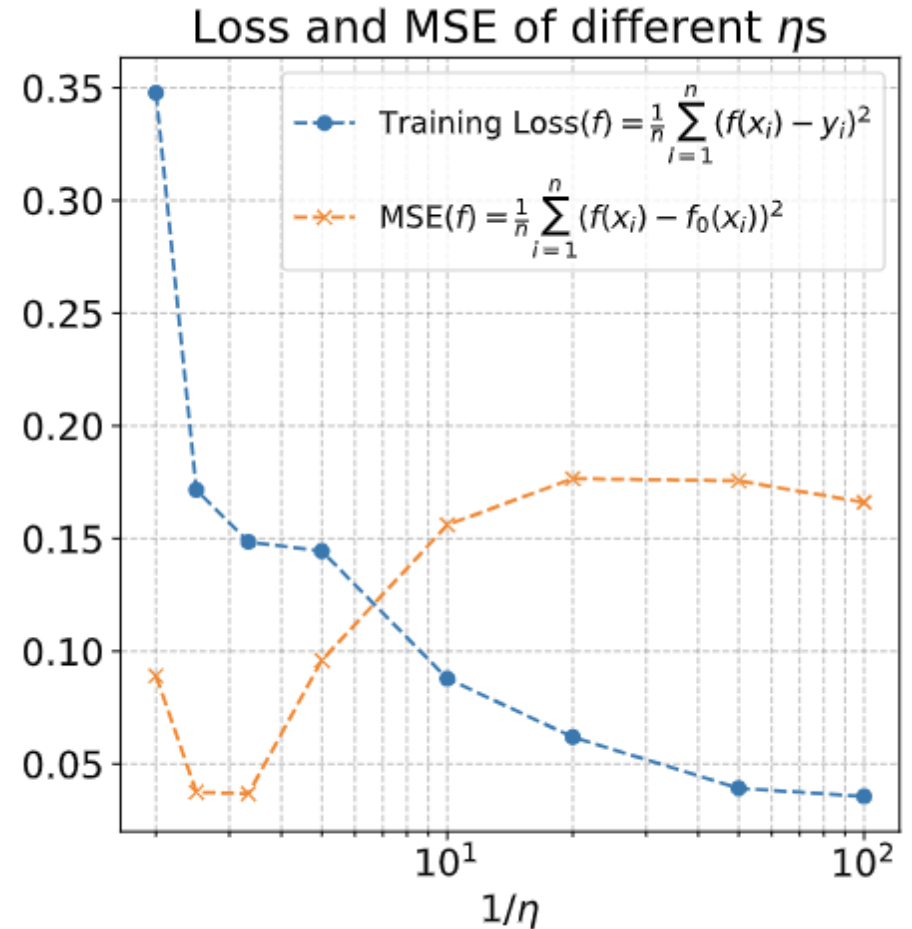
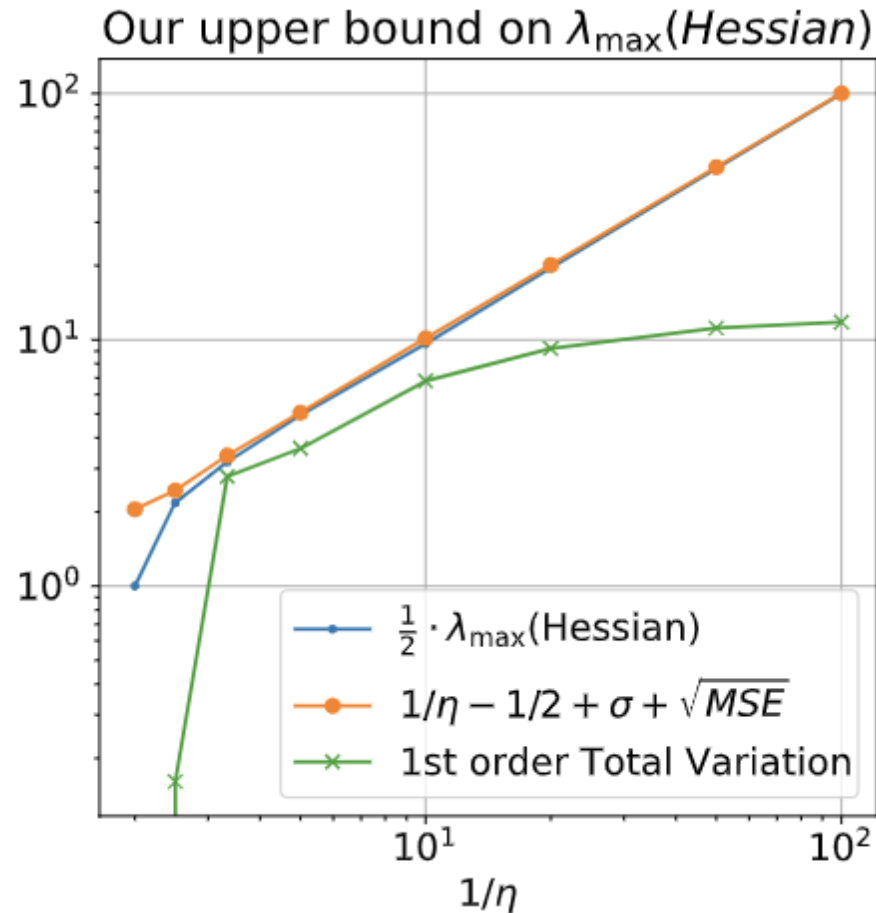
(Mulayoff et al. 2021) showed that:

$$\lambda_{\max} \left(\frac{1}{n} \sum_i \nabla_{\theta} f_{\theta}(x_i) \nabla_{\theta} f_{\theta}(x_i)^T \right)$$

is connected to the TV1 of f_{θ} .

Main challenge: Handling the second term without “interpolation”.

$1/\eta$ very precisely predicts the sharpness, and gives a classical U-shape risk curve.



Generalization bounds that stem from these function space characterization

Theorem (informal): We proved that in the **strict interior of the data support**:

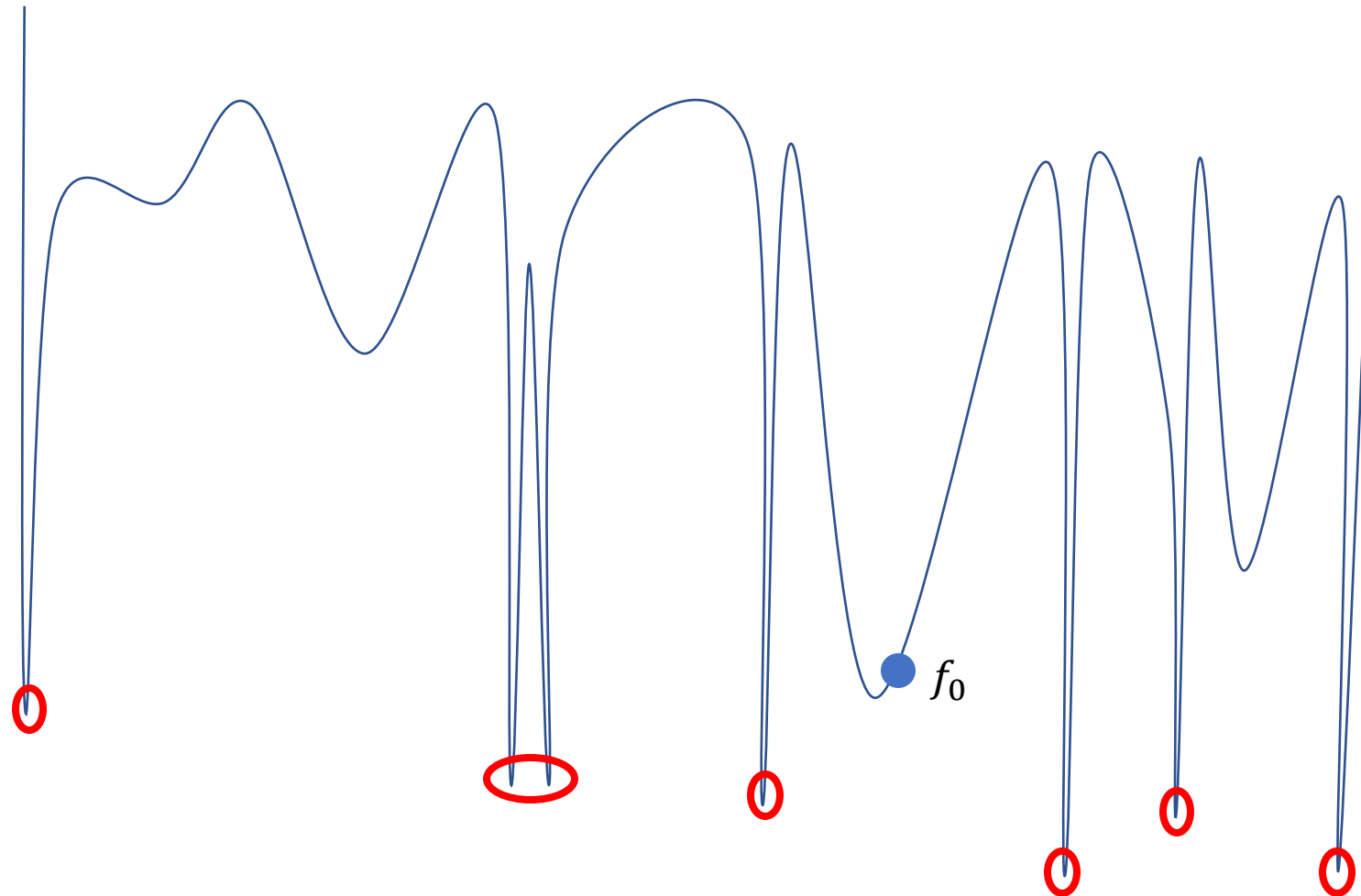
1. Agnostic case: generalization gap = $O(n^{-2/5})$
2. In the non-parametric regression setting, if training loss smaller than σ^2 then w.h.p., get an MSE

$$\text{MSE}_{\mathcal{I}}(f) = \frac{1}{n_{\mathcal{I}}} \sum_{x_i \in \mathcal{I}} (f(x_i) - f_0(x_i))^2 \leq \tilde{O} \left(\left(\frac{\sigma^2}{n_{\mathcal{I}}} \right)^{\frac{4}{5}} \left(\frac{x_{\max}}{\eta} + \sigma x_{\max}^2 \right)^{\frac{2}{5}} \right)$$

* near minimax optimal.

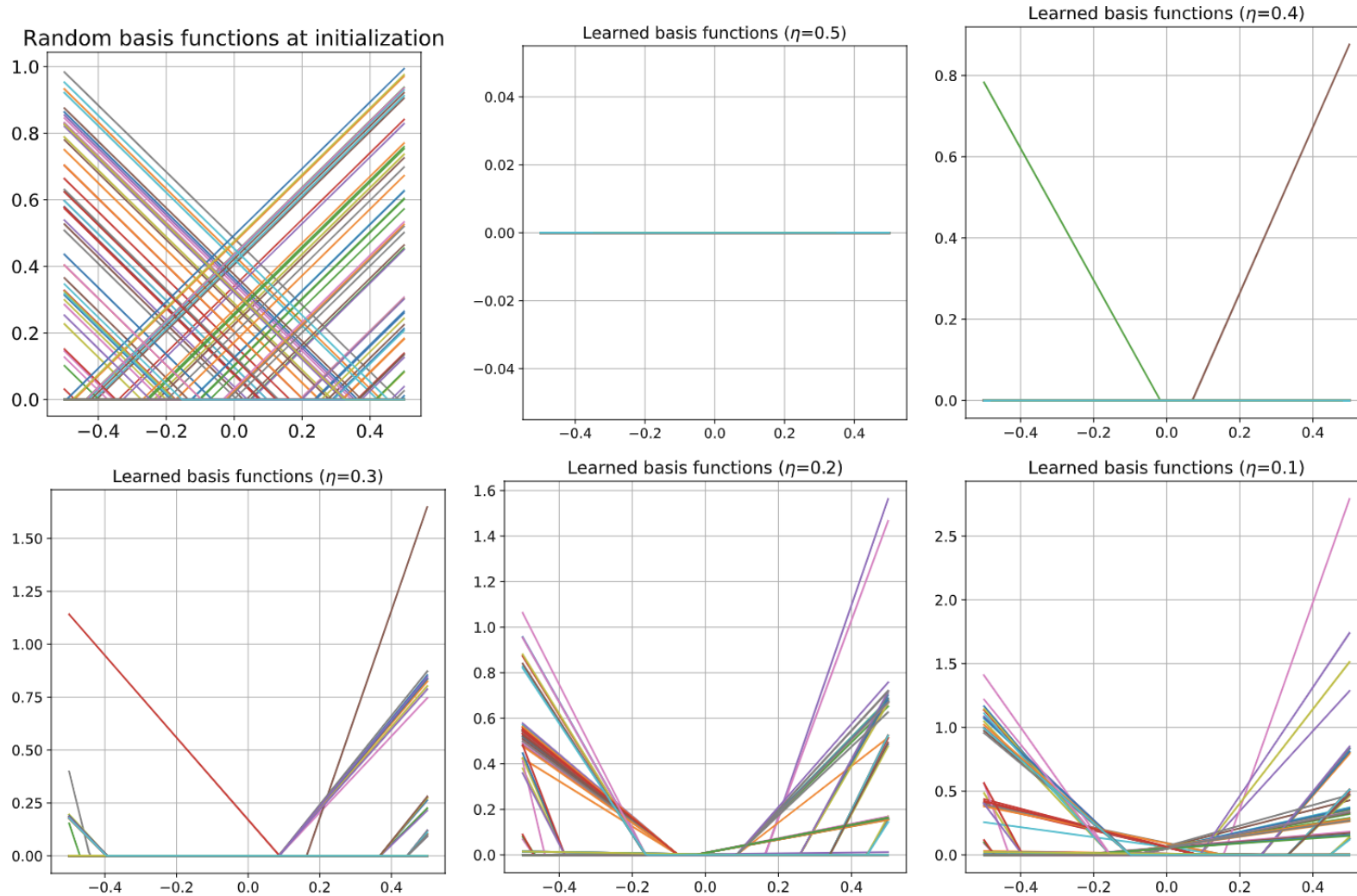
* Kernel ridge regression with any RKHS including Neural Tangent Kernel are suboptimal.

It tells us something new about the energy landscape of overparameterized NN training on noisy problems



Training with GD automatically avoids these sharp solutions

“Representation learning” becomes learning knots, and knot “sparsity” but are they really zero?



Checkpoint

- In simple “curve fitting” problem, two-layer ReLU NN does not overfit if trained with GD (regardless how overparameterized it is)
- Tuning learning rate choice is connected to an L1-type smoothness that we can quantify.
- Two-layer ReLU NN is provably stronger than NTK, as long as we tune the hyperparameter appropriately.

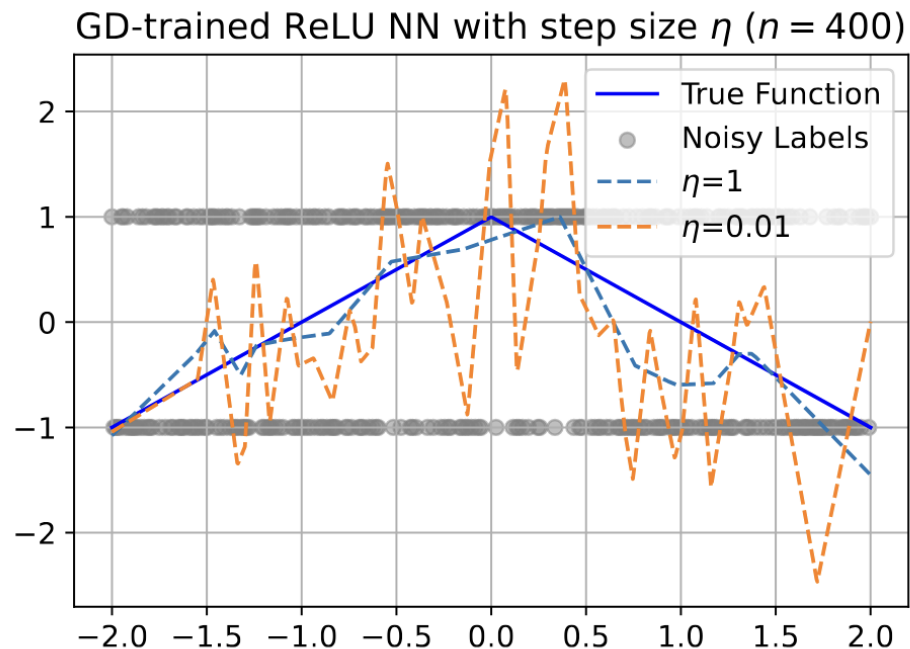
“What happens to multivariate input? What happens to Logistic Loss?” – Peter Bartlett



- Why? For multivariate Gaussian input, it often interpolates in practice (in the kernel regime).
- For *linear logistic regression* (with margin-separable data), GD can converge “stably” to the max-margin separator at infinity with arbitrarily large step size. ([Jingfeng Wu et al. 2025](#))

Does Flatness imply Generalization for Logistic Loss in Univariate Two-Layer ReLU Network?

- Empirically, kinda yes.
- Data: $y \sim \text{Bernoulli}(\text{Sigmoid}(f_0(x)))$



But we can no longer talk about the set of all flat solutions.

$$\{f_\theta \mid \lambda_{\max}(\nabla^2 \mathcal{L}(\theta)) \leq 2/\eta\}$$

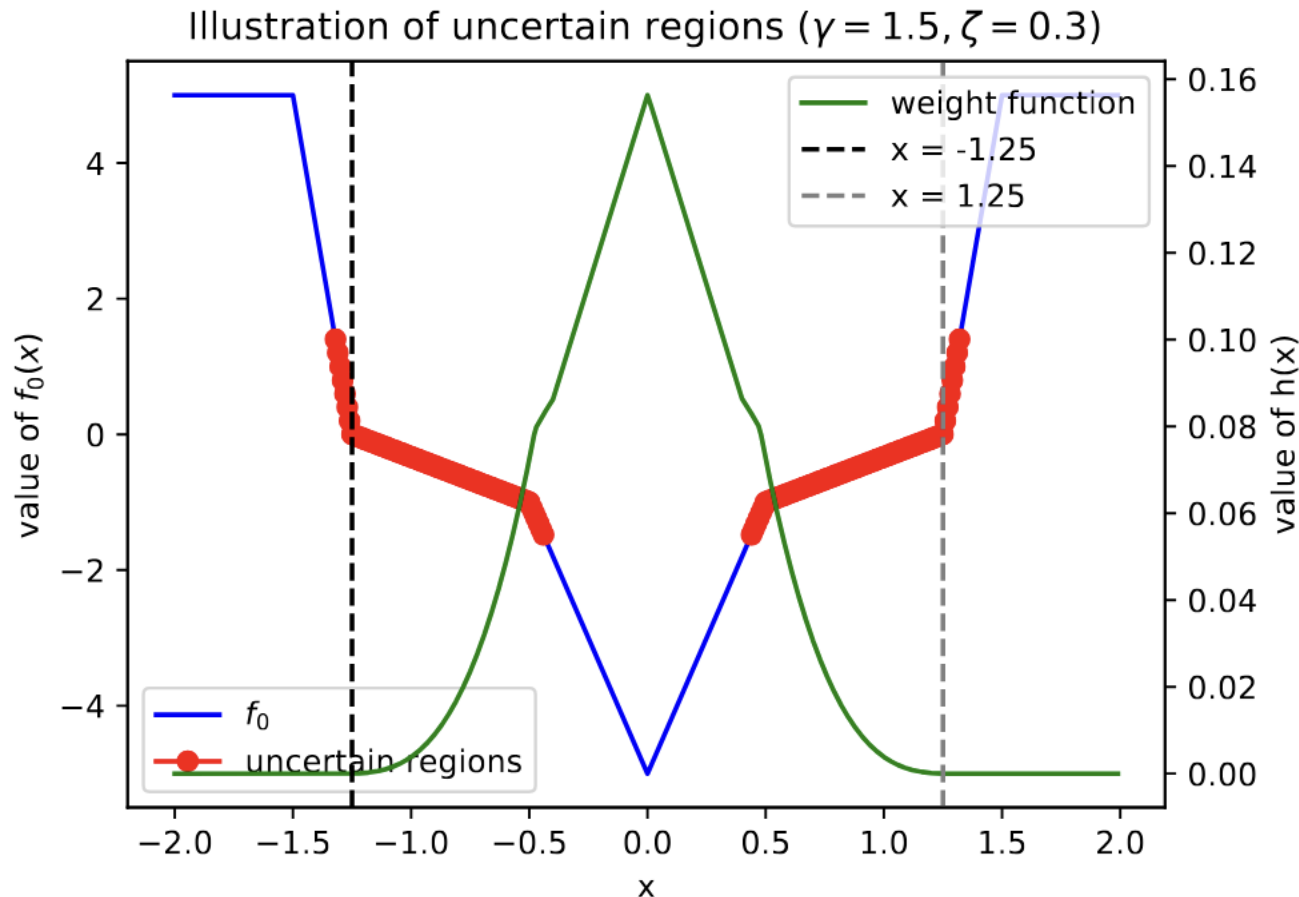
$\subset$

$$\left\{ f \mid \int |f''(x)| \boxed{g(x)} dx \leq \frac{2}{\eta} \right\}$$

But the weighting function g now depends on f !

The weighting function now depends on the uncertainty region of the current NN configuration.

$$\left\{ f \left| \int |f''(x)|g(x)dx \leq \frac{2}{\eta} \right. \right\}$$

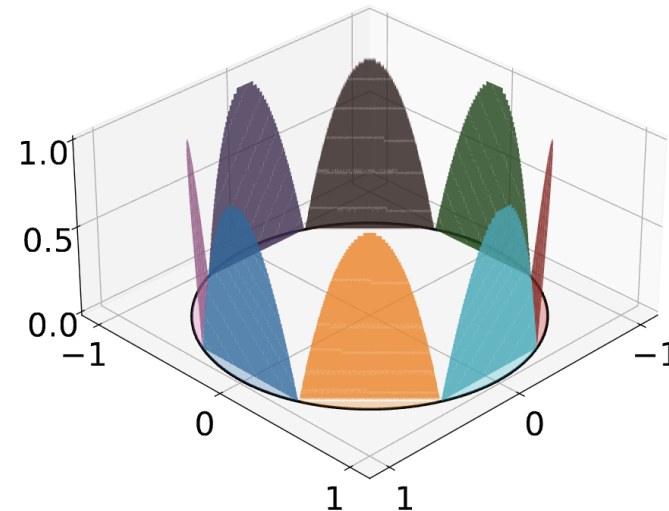
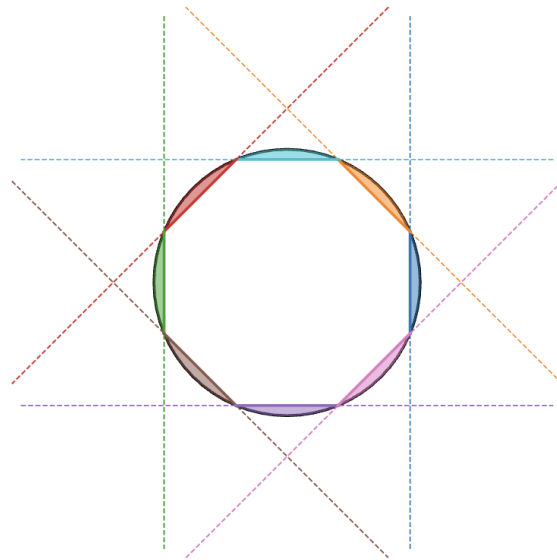


What's worse, we can construct a solution that is

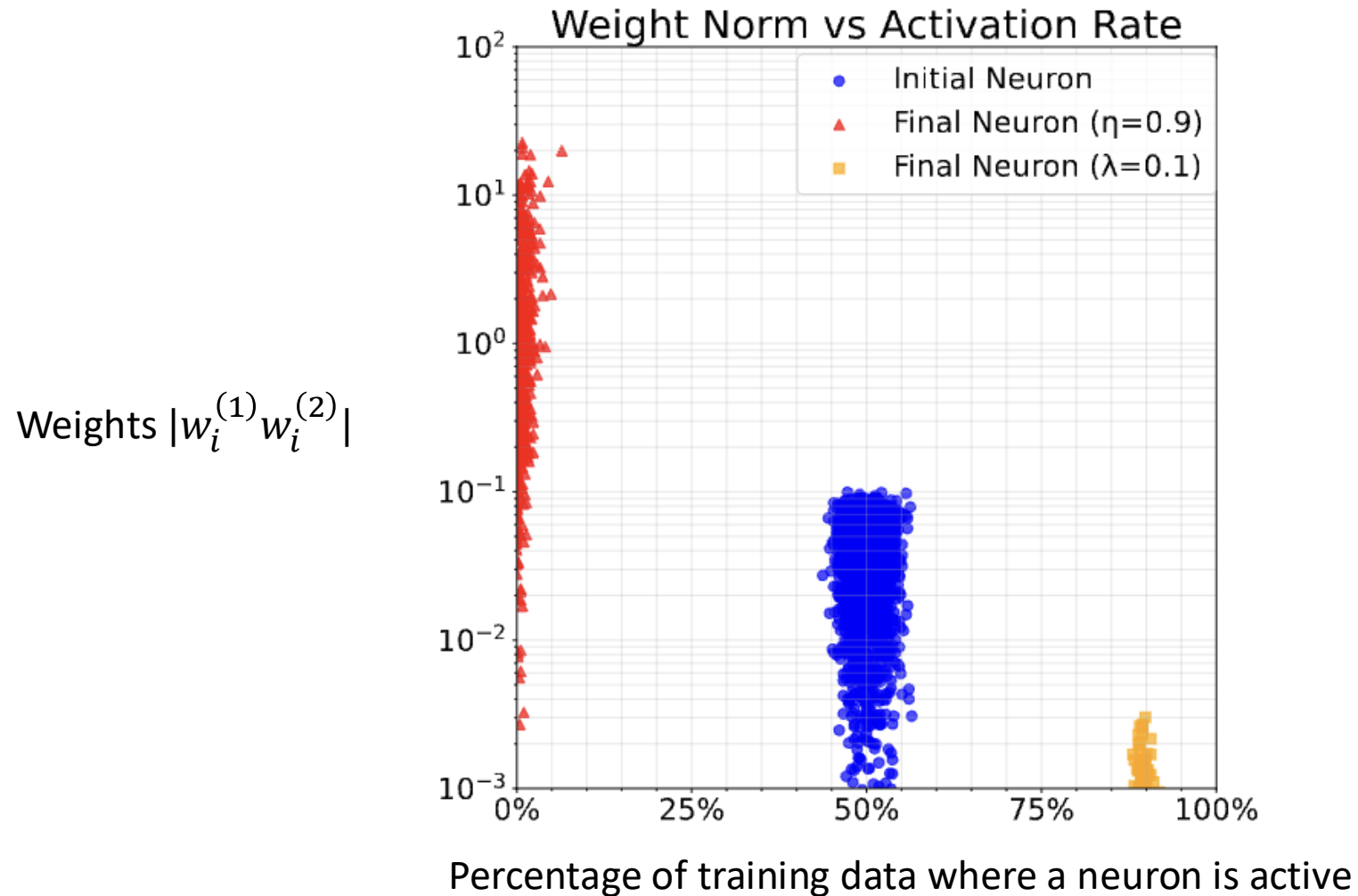
1. interpolating
2. arbitrarily flat loss

How about the multivariate case? It works, but suffers from the curse of dimensionality.

- Lower bound from our new paper reveals a **neural Shattering Phenomenon**: *It's very easy for each neuron to single out one data point at boundary.*



Neural Shattering does not happen if there is weight decay



Remainder of the talk

- Generalization by Large Learning Rate
- Generalization by weight decay
- Discussion

Background: L2-Regularization and Gradient Descent with Weight Decays

- “Weight decay” == L2 Regularization

$$\nabla_{\theta} \left(\mathcal{L}(\theta) + \frac{\lambda}{2} \|\theta\|^2 \right) = \nabla \mathcal{L}(\theta) + \lambda \theta$$

- Gradient Descent:

$$\theta_{t+1} = \theta_t - \eta(\nabla \mathcal{L}(\theta_t) + \lambda \theta_t) = \overset{\text{“Weight decay”}}{(1 - \eta\lambda)} \theta_t - \eta \nabla \mathcal{L}(\theta_t)$$

Explicit Bias through Small Weights

- Again, we consider two-layer NNs with ReLU activations

$$\mathcal{F} = \left\{ f : \mathbb{R} \rightarrow \mathbb{R} \mid f(x) = \sum_{i=1}^k w_i^{(2)} \phi \left(w_i^{(1)} x + b_i^{(1)} \right) + b^{(2)} \right\}$$

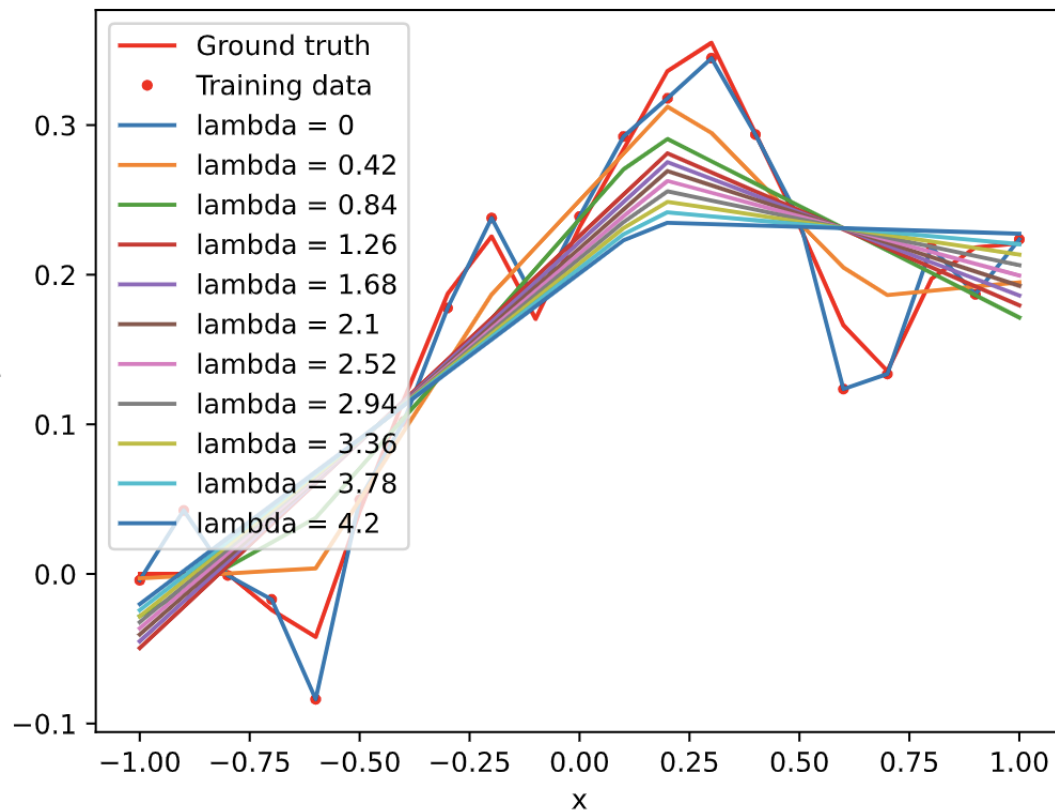
- The right family of solutions

$$\{ f_{\theta} \mid \|\theta\|^2 \leq C \}$$

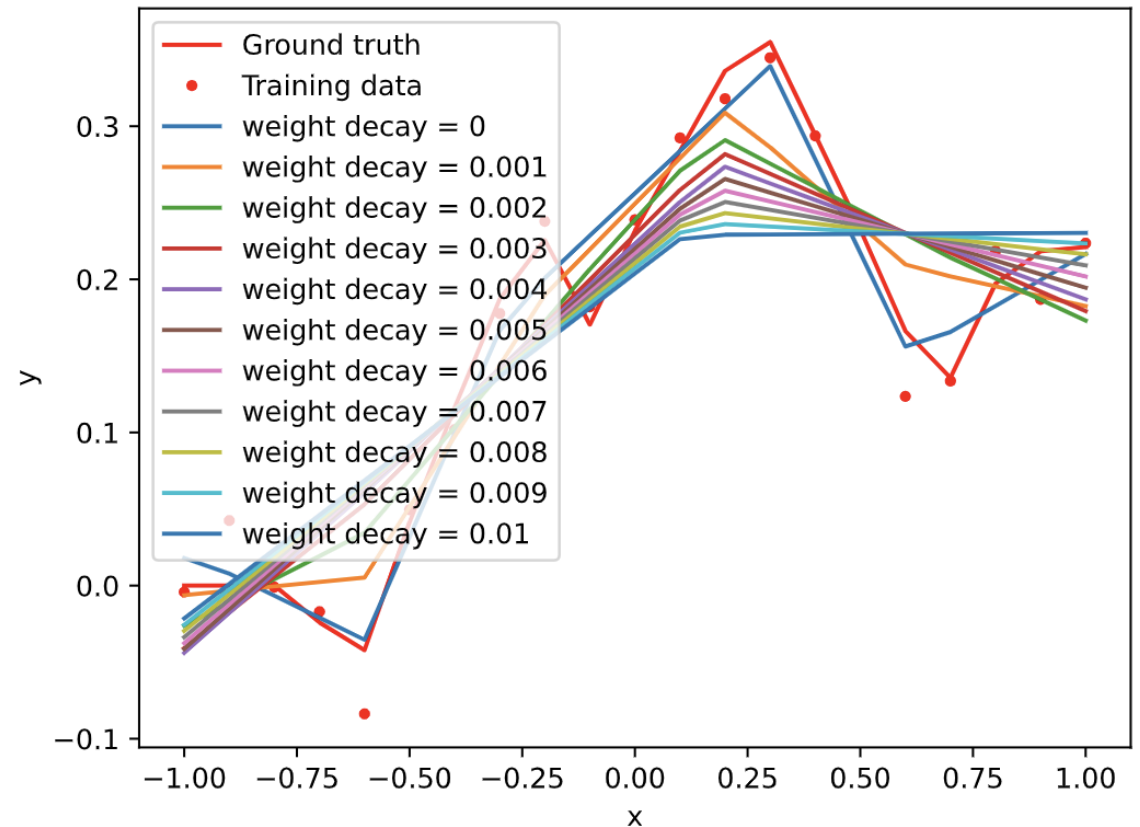
What do these small weights Neural Networks look like in function space?

Let's first look at an experiment: compare ReLU NN with L1 trend filtering

L1-Trend Filtering

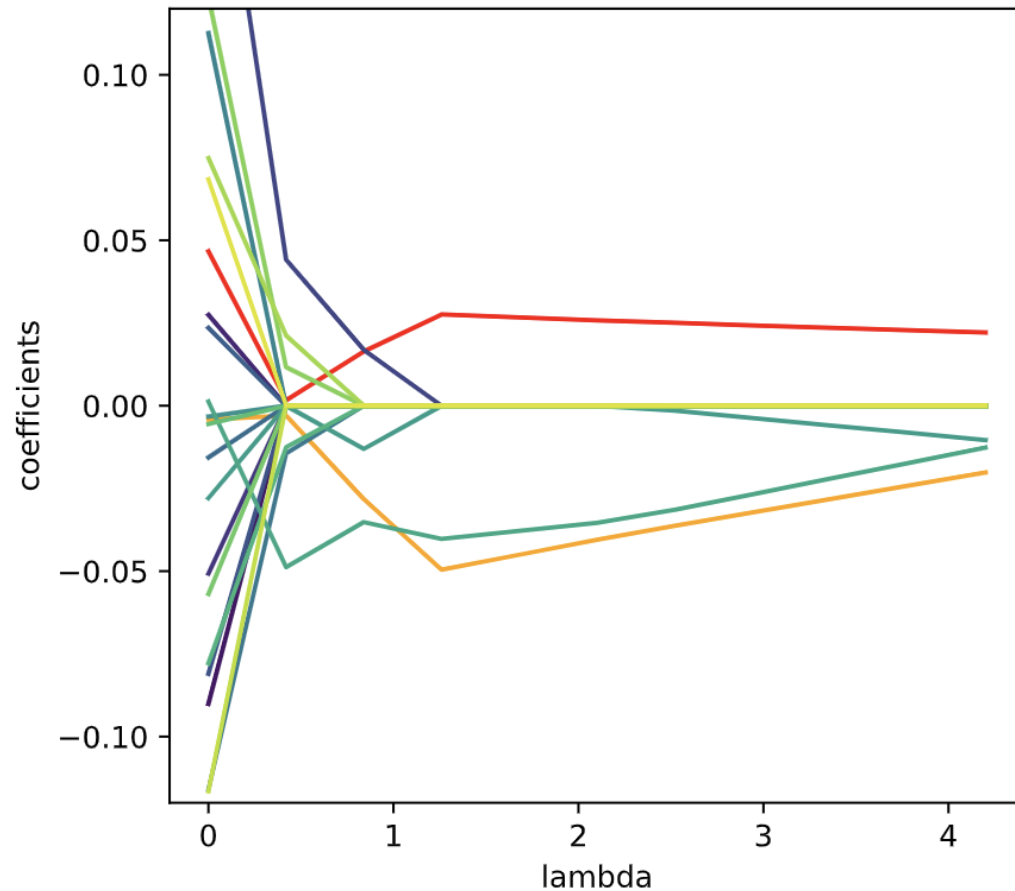


Two-layer ReLU NN with weight decay

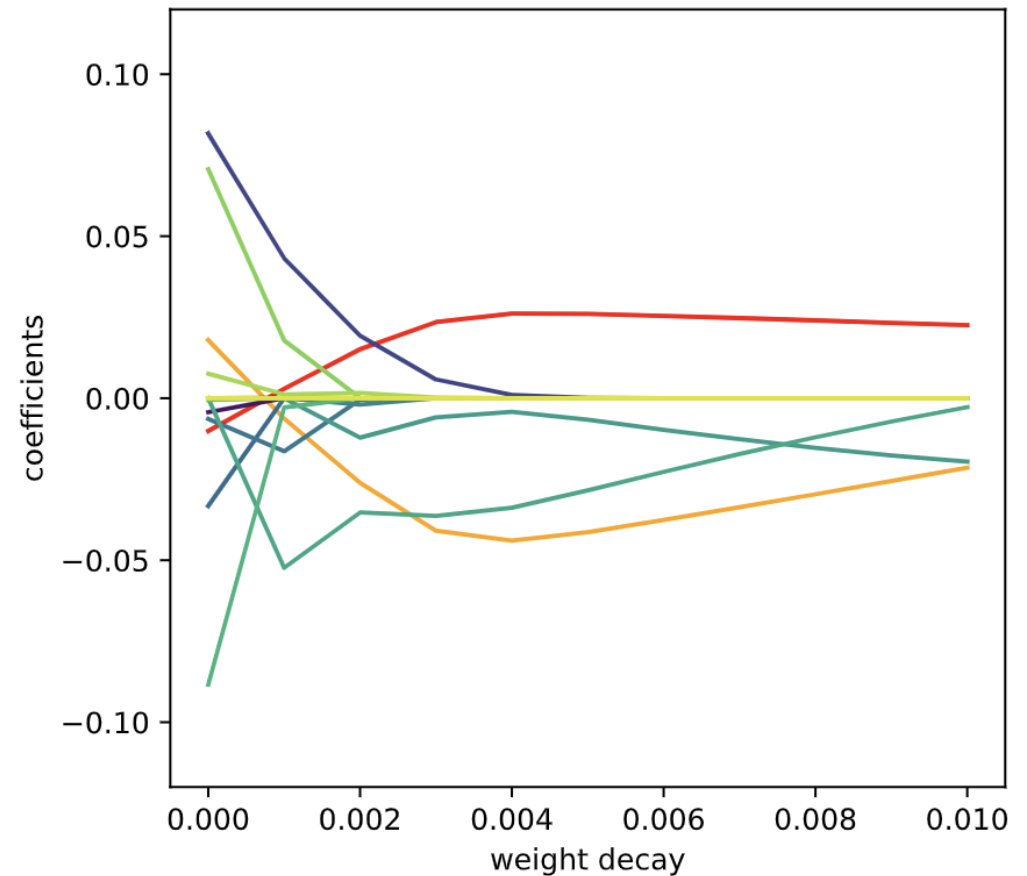


Let's inspect the regularization paths

L1-Trend Filtering



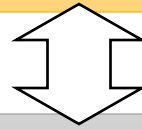
Two-layer ReLU NN with weight decay



It turns out that they are equivalent!

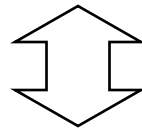
L1-Trend Filtering

(Kim, Koh and Boyd, 2009; Tibshirani, 2024)



LAR Splines

(Mammen and Van De Geer, 1997)



Reveals the effect of weight decay

(Sufficiently wide) Two-layer ReLU
Neural Networks with Weight Decay

Two-layer Weight-Decayed NN is equivalent to LAR Splines (Parhi and Nowak, 2021) when mildly overparameterized

(*for this talk, consider only $m = 1$)

- When the number of knots $M > n - m$
 - Banach space representer Thm (Theorem 8 of Parhi and Nowak, 2021)

$$\min_{w,v} \hat{L}(f) + \frac{\lambda}{2} \sum_{j=1}^M (|v_j|^2 + |w_j|^{2m}) \iff \min_f \hat{L}(f) + \lambda TV(f^{(m)}(x)),$$

↑
over all functions!

- By Mammen and Van De Geer (1997)

$$\text{MSE}(\hat{f}) = O(n^{-(2m+2)(2m+3)}). \quad \text{(Minimax rate)}$$

In fact, we don't have to solve argmin. A generalization gap bound works for all small-norm solutions.

$$\{f_\theta \mid \|\theta\|^2 \leq C\} \subset \left\{f \mid \int_{-x_{\max}}^{x_{\max}} |f''(x)| dx \leq C\right\}$$

Theorem 3.2. *For any constant $B > 0$, with probability $1 - \delta$, for any two-layer ReLU NN $f = f_\theta$ supported on $[-x_{\max}, x_{\max}]$ such that $\|f\|_\infty \leq B$, it holds that*

$$|\mathcal{L}(f) - \bar{\mathcal{L}}(f)| \leq O \left(\left[\frac{\log(1 + e^B)^4 \cdot \max\{\|\theta\|_2^2, 1\} \cdot x_{\max} \log \left(\frac{2 \max\{\|\theta\|_2^2, 1\}}{\delta} \right)^2}{n^2} \right]^{\frac{1}{5}} \right), \quad (6)$$

where the randomness is over the data generation process.

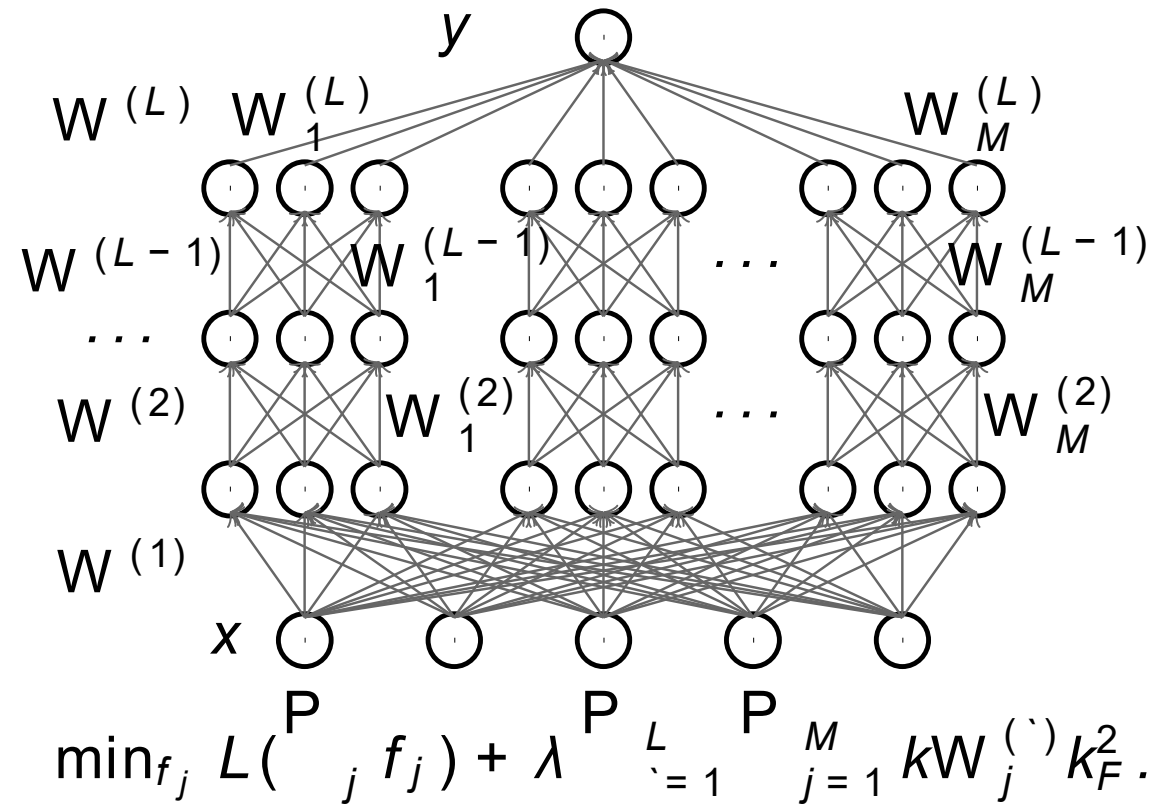
Much cleaner than “Large Step Size”: No restriction on data domain.

Checkpoint: Overparameterize + Weight Decay in 2-layer NN is an optimal estimator for TV1 functions.

- Tuning weight decay is equivalent to tuning the TV1 radius.
- Explains what kernels cannot explain! are provably suboptimal.
- Better and cleaner than relying on “implicit bias” from large-step size (and flatness).

Can we generalize the result to deeper networks?

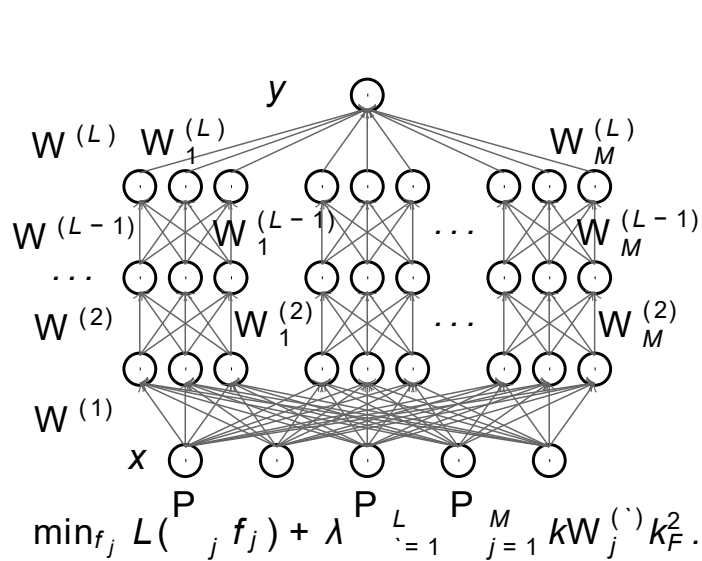
L-Layer *Parallel* Neural Networks



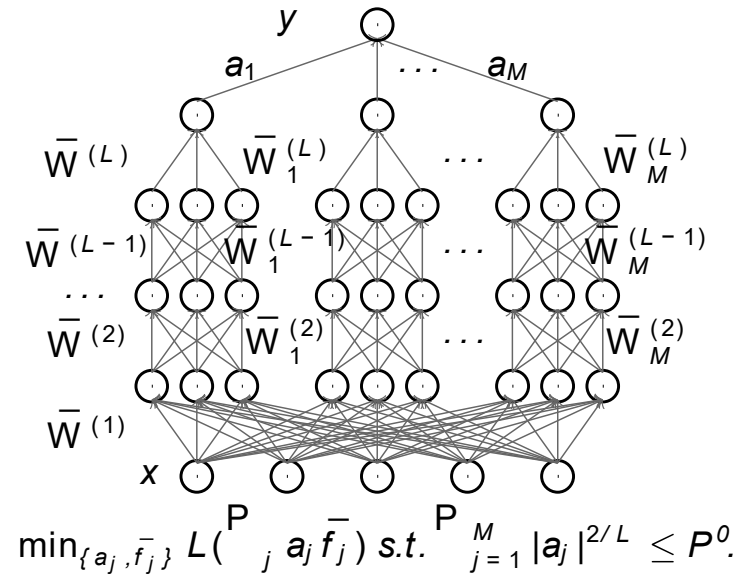
(a) Parallel NN with Weight Decay

(Ergen&Pilanci, 2021; Haeffele & Vidal, 2017). Also, SqueezeNet, ResNeXT etc.

Weight decayed L-Layer PNN is equivalent to Sparse Linear Regression with learned basis functions



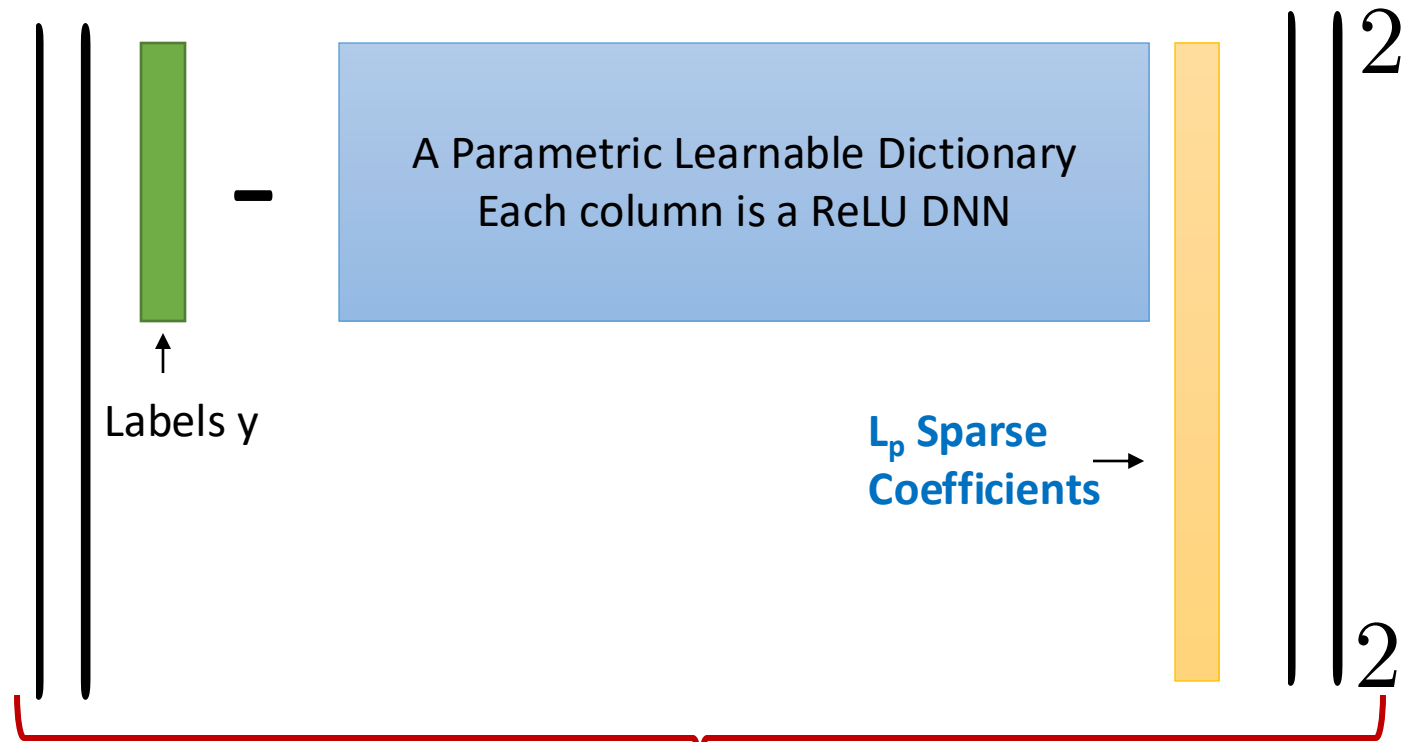
(a) Parallel NN with Weight Decay



(b) Sparse Regression with Learned Representation

$$\begin{aligned} \arg \min_{\{\bar{\mathbf{W}}_j^{(\ell)}, \bar{\mathbf{b}}_j^{(\ell)}, a_j\}} \hat{L}\left(\sum_{j=1}^M a_j \bar{f}_j\right) &= \frac{1}{n} \sum_i (y_i - \bar{f}_{1:M}(\mathbf{x}_i)^T \mathbf{a})^2 \\ s.t. \quad &\|\bar{\mathbf{W}}_j^{(1)}\|_F \leq c_1 \sqrt{d}, \forall j \in [M], \\ &\|\bar{\mathbf{W}}_j^{(\ell)}\|_F \leq c_1 \sqrt{w}, \forall j \in [M], 2 \leq \ell \leq L, \quad \|\{a_j\}\|_{2/L}^{2/L} \leq P' \end{aligned}$$

Weight decayed L-Layer PNN is equivalent to Sparse Linear Regression with learned basis functions



$$\arg \min_{\{\bar{\mathbf{W}}_j^{(\ell)}, \bar{\mathbf{b}}_j^{(\ell)}, a_j\}} \hat{L} \left(\sum_{j=1}^M a_j \bar{f}_j \right) = \frac{1}{n} \sum_i (y_i - \bar{f}_{1:M}(\mathbf{x}_i)^T \mathbf{a})^2$$

$$s.t. \quad \|\bar{\mathbf{W}}_j^{(1)}\|_F \leq c_1 \sqrt{d}, \forall j \in [M],$$

$$\|\bar{\mathbf{W}}_j^{(\ell)}\|_F \leq c_1 \sqrt{w}, \forall j \in [M], 2 \leq \ell \leq L, \quad \|\{a_j\}\|_{2/L}^{2/L} \leq P'$$

Formal setup / notations

- Function classes

- Bounded Variation class: $BV(m) := \left\{ f \in L_\infty : TV(f^{(m)}) < \infty \right\}$
- Besov class $B_{p,q}^\alpha$ d-dimensional
- Connections: $B_{1,1}^{m+1} \subset BV(m) \subset B_{1,\infty}^{m+1}$

- Metric
$$\text{MSE}(\hat{f}) := \mathbb{E}_{\mathcal{D}_n} \frac{1}{n} \sum_{i=1}^n (\hat{f}(\mathbf{x}_i) - f_0(\mathbf{x}_i))^2.$$

- Problem setting:

- Fixed design, subgaussian noise

(Zhang and W. 2023): Parallel ReLU DNN approaches the minimax rates as it gets deeper.



	Minimax Rate	Minimax Linear Rate
Besov Space	$n^{-\frac{2\alpha}{2\alpha+d}}$	$n^{-\frac{2\alpha-1}{2\alpha+d-1}}$
Bounded Variation	$n^{-\frac{2m+2}{2m+3}}$	$n^{-\frac{2m+1}{2m+2}}$

- **Theorem 2:** Besov space $B_{p,q}^\alpha$

$$\text{MSE}(\hat{f}) = \tilde{O}\left(n^{-\frac{2\alpha/d(1-2/L)}{2\alpha/d+1-2/(pL)}}\right) + O(e^{-c_6 L})$$

- **Corollary 3** for $BV(m)$ class:

$$\text{MSE}(\hat{f}) = \tilde{O}\left(n^{-\frac{(2m+2)(1-2/L)}{2m+3-2/L}}\right) + O(e^{-c_6 L}),$$

Arbitrarily close to the minimax rates when we choose $L = C \log n$.

Many interesting insights we can read off from the theorem

1. Formal separation from kernels (NTK or other kernel ridge regressions)
 - Our upper bound + [Donoho, Liu, MacGibbon \(1990\)](#)'s linear smoother lower bound.
2. Deep NNs achieve smaller error than shallow NNs
3. Overparameterization does not cause overfitting
 - Due to explicit regularization (i.e., weight decay) => sparsity

Comparing to classical nonparametric regression methods

$$\hat{f}(x) = \sum_{i=1}^M g_i(x) c_i$$

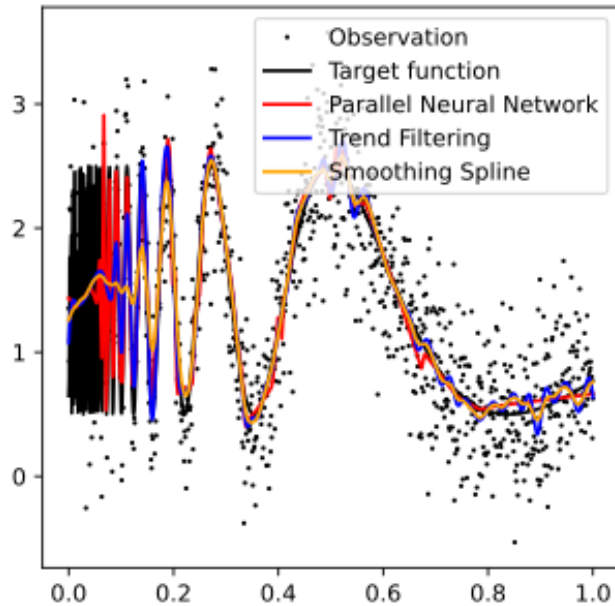
$$[g_1, \dots, g_M]$$

$$c_{1:M} \in \mathbb{R}^M$$

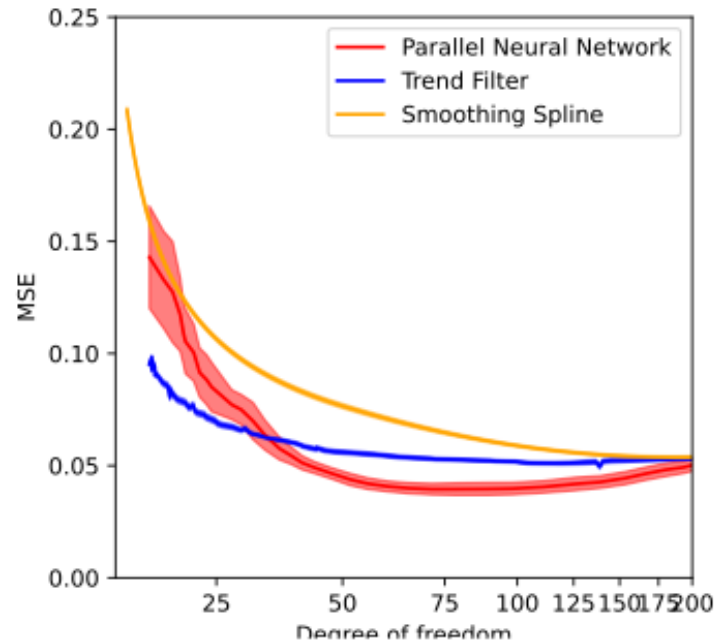
	LAR Splines / Trend filtering	Wavelet smoothing	Parallel DNN
Basis functions	Hard-coded for each order of smoothness	Hard-coded to the chosen wavelets	Parametric and learned from data.
Coefficient vector	L1-sparsity	L1 or L0- sparsity	Lp sparsity (p=2/L)

- DNNs adapt to different function classes
 - By overparameterizing / learning representation and tuning regularization weight via cross-validation (implicitly selecting a few basis functions!)
 - Paying almost no statistical price!

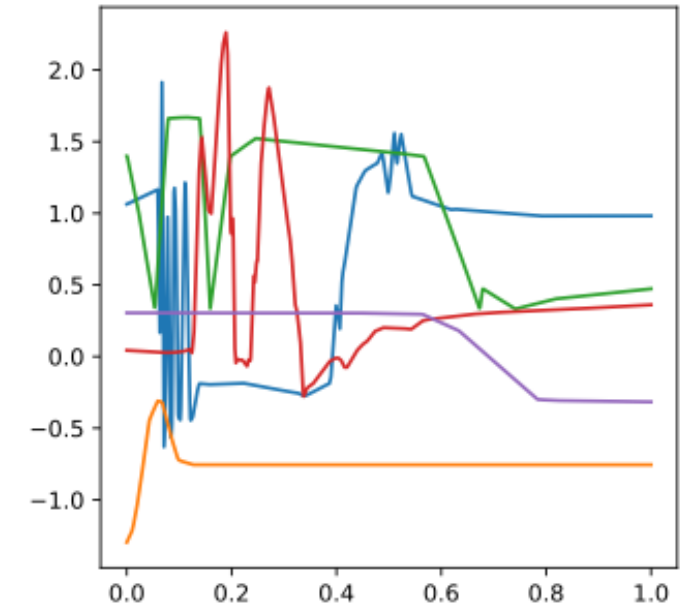
Examples of Functions with Heterogeneous Smoothness



Fitted functions with optimally tuned parameter



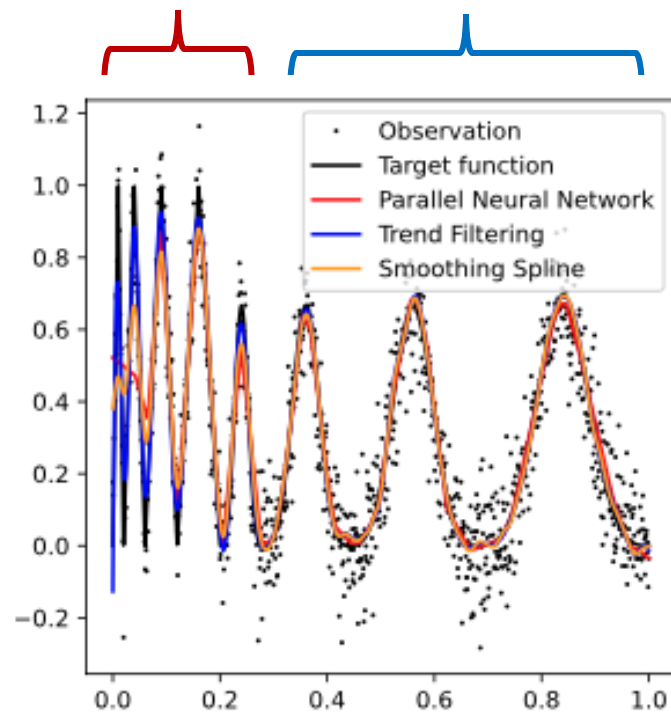
MSE comparison over effective degree of freedom



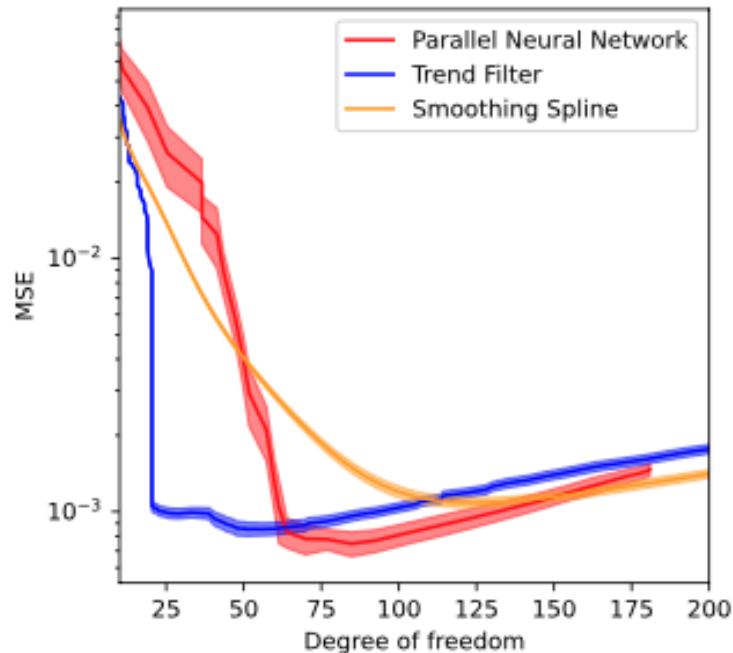
Learned basis functions. Only a handful that are active, i.e. sparsity. Lottery ticket?

Examples of Functions with **even more** Heterogeneous Smoothness

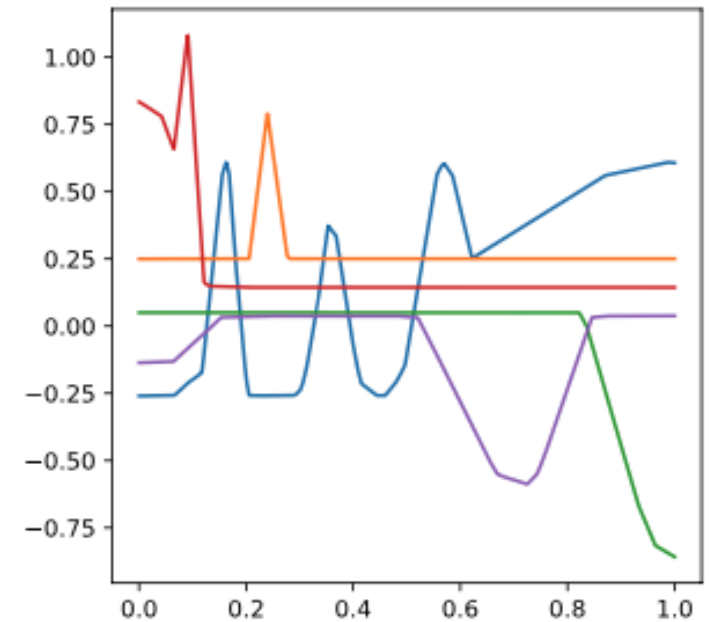
Piecewise Linear Piecewise Cubic



Fitted functions with optimally tuned parameter



MSE comparison over effective degree of freedom



Learned basis functions. Only a handful that are active, i.e. sparsity. Lottery ticket?

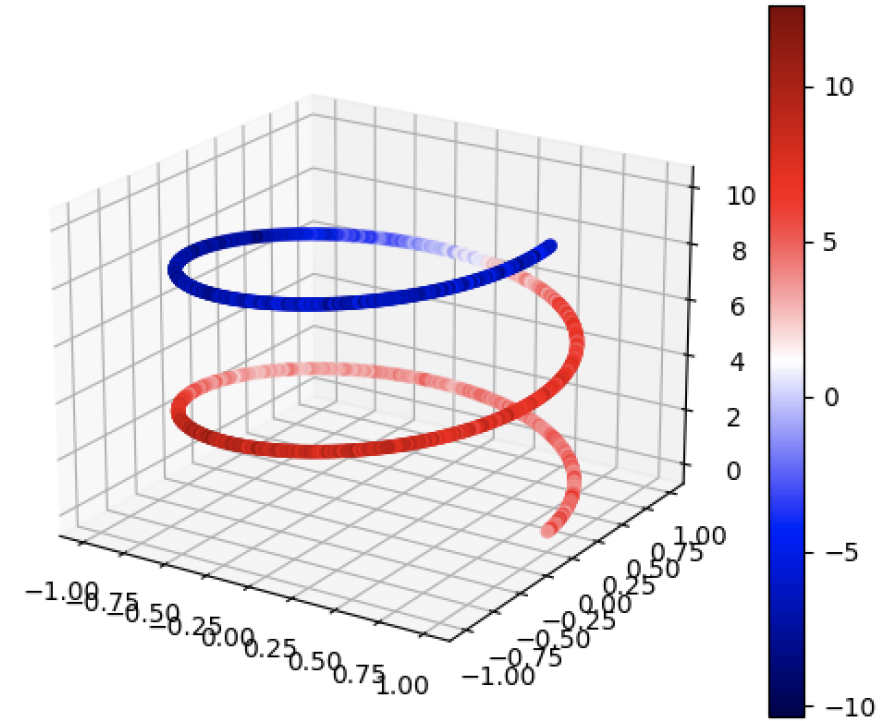
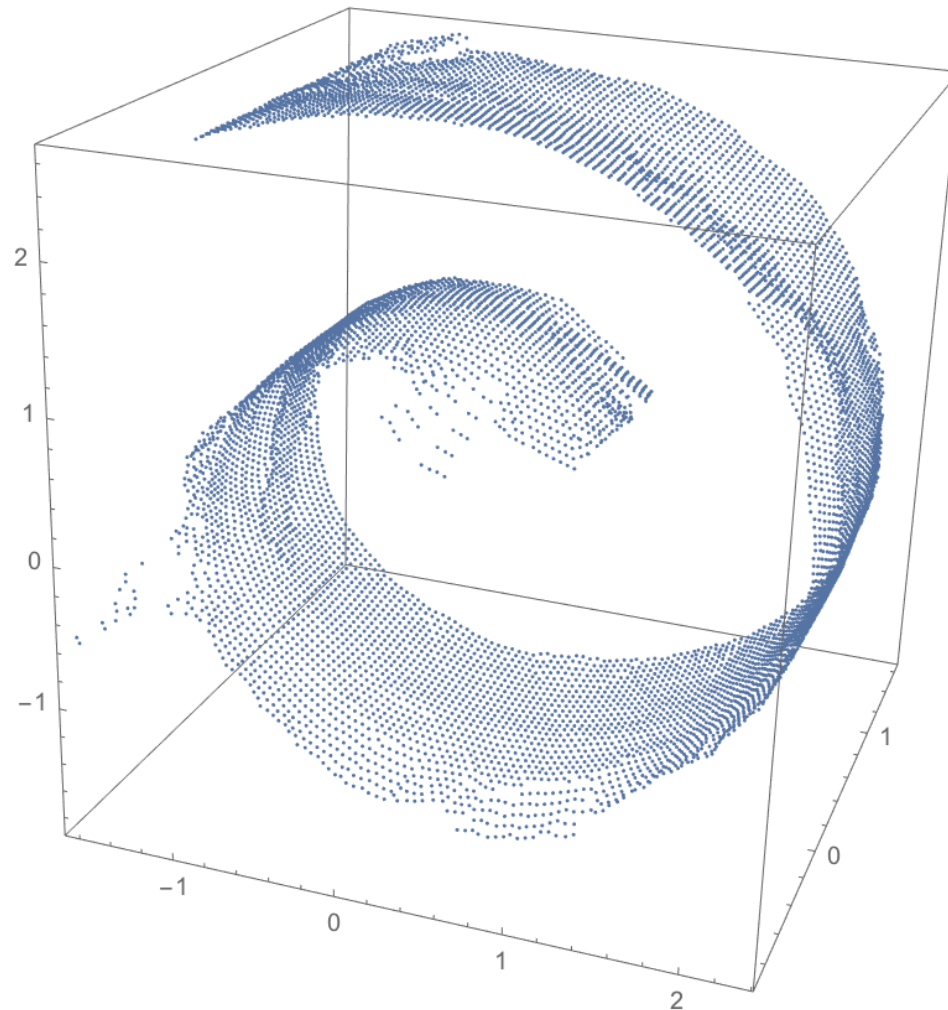
Checkpoint: PNN with weight decay enjoys amazing adaptivity.

- Overparameterize then tune weight decay
 - Adapts to a broad class of functions
 - Learns a sparse linear combination of learned functions
 - New insight into depth, width and representation

$$n^{-\frac{2\alpha}{2\alpha+d}}$$

- But, does not solve the curse of dimensionality?

Weight decay in Parallel NN adapts to low-dimensional manifold structures in data distribution.



We prove near-optimal rate that adapts to intrinsic dimension by overparameterization + weight decay (Zhang et al, 2024)

Theorem: ResNeXt, or ConvResNeXt (or PNNs). Lipschitz loss functions, bounded outputs. Choose depth $L = \log(n)$, $MN = O(n)$, then:

$$\mathbb{E}_{\mathcal{D}}[\text{Loss}(\hat{f}(\mathbf{x}), y)] \leq \mathbb{E}_{\mathcal{D}}[\text{Loss}(f^*)] + \tilde{O}\left(n^{-\frac{\alpha/d}{2\alpha/d+1}(1-o(1))}\right),$$

- No (exponential) dependence on the ambient dimension D .
- No need to know intrinsic dim d , and other parameters
- Tune only the weight decay

Remainder of the talk

- Generalization by Large Learning Rate
- Generalization by weight decay
- Discussion

Implications on modern generalization theory?

Classical generalization theory

VC-theory

Rademacher complexity

Metric entropy

Other empirical process theory
for getting Uniform convergence

limitations

- Nothing about computation
- Uniform convergence might not be needed

In between

Algorithmic stability

Minima stability

Edge-of-Stability

...

Modern generalization theory

NTK theory

Benign overfitting

Everything to do with
optimization.

limitations

Often restrictive in model classes
/ data distribution
Dynamic theory is hard for
complex architectures.

Weight Decay vs Large Learning Rate

- Both gives us similar inductive biases in function space (a sparse L1 regularization that induces sparsity).
 - Allow them to generalize near optimally and outperform NTK.
- But the regularization effect of Large Learning Rate (or flatness) become fragile for logistic loss and in high-dimension while weight decay remains strong.
- Are there other benefits of Large Learning Rate?

Open problems

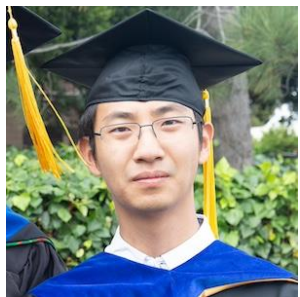
- Flatness implied generalization in multi-layer models.
- Can “flatness” discover low-dimensional manifold structures?
- Understanding optimization and the effects of initialization more in the representation learning regime?

Thank you for your attention!



References:

- Qiao et al. (2024) **Stable Minima Cannot Overfit in Univariate ReLU Networks: Generalization by Large Step Sizes:** <https://arxiv.org/abs/2406.06838>
- Qiao and W. (2025) **Does Flatness imply Generalization for Logistic Loss in Univariate Two-Layer ReLU Network?:** In preparation coming soon.
- Liang, Qiao, W. and Parhi (2025) **Stable Minima of ReLU Neural Networks Suffer from the Curse of Dimensionality: The Neural Shattering Phenomenon:** <https://arxiv.org/abs/2506.20779>
- Zhang et al. (2022) **Deep Learning meets Nonparametric Regression: Are Weight-Decayed DNNs Locally Adaptive?** <https://arxiv.org/abs/2204.09664>
- Zhang et al. (2023) **Nonparametric Classification on Low Dimensional Manifolds using Overparameterized Convolutional Residual Networks:** <https://arxiv.org/abs/2307.01649>
- Work partially supported by
 - NSF DMS Award “SCALE MoDL: The Adaptivity of Deep Learning”



Kaiqi Zhang, Dan Qiao, Tongtong Liang, Esha Singh, Zixuan Zhang, Minshuo Chen, Tuo Zhao, Daniel Soudry, Rahul Parhi