NeurIPS Tutorial on “Training Instability”
Part 2: Generalization

Maryam Fazel and Yu-Xiang Wang

Part 1 of the tutorial is about “Rethinking Optimization”

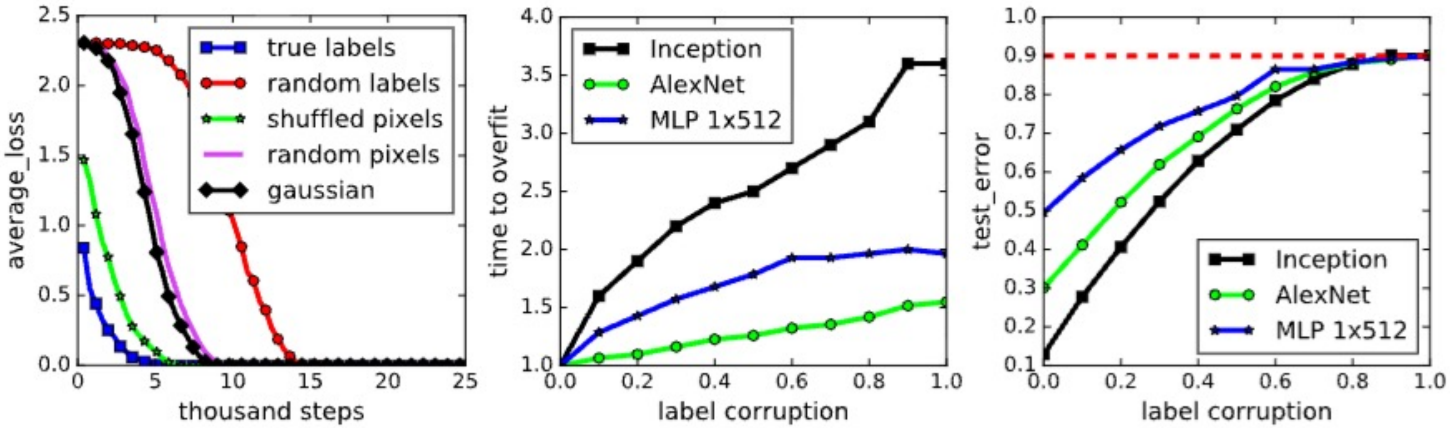
- Go beyond the “stable regime”

- Gradient descent can often converge faster!
  - Linear convergence
  - Nesterov Accelerated Rates
  - (Sometimes) arbitrarily fast (constant iteration complexity)

Part 2 of the tutorial is about "Rethinking Generalization"

Understanding deep learning requires rethinking generalization

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, Oriol Vinyals



(a) learning curves (b) convergence slowdown (c) generalization error growth

- **Deep learning models in practice are NOT capacity limited**
- "generalization" depends on many factors

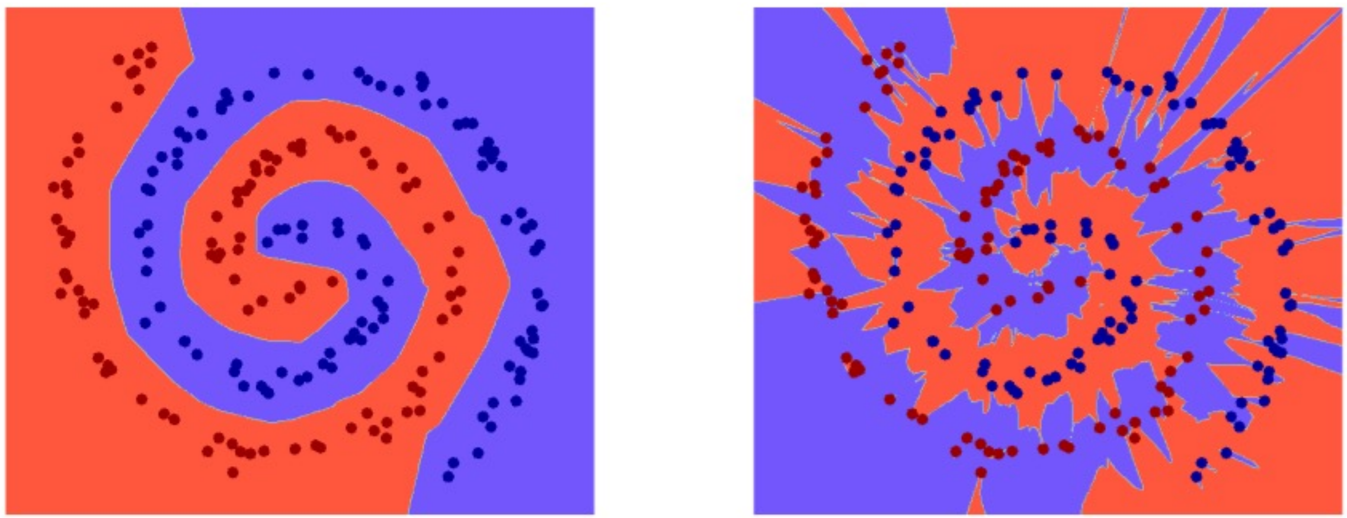
We ask: how does large stepsize affects generalization in overparameterized models?

Let's say the labels are clean… there are many "interpolating" solutions

[Submitted on 7 Jun 2019 (v1), last revised 15 Nov 2020 (this version, v6)]

Understanding Generalization through Visualizations

W. Ronny Huang, Zeyad Emam, Micah Goldblum, Liam Fowl, J. K. Terry, Furong Huang, Tom Goldstein



(a) 100% train, 100% test   (b) 100% train, 7% test

Question #1: Does GD with Large Stepsize *find* the generalizing solutions or overfitting solutions?

Things become even more interesting when the labels are noisy.

Benign overfitting (Belkin, Bartlett et al.) : you may have 0 training loss on noisy labels, yet test error / loss → 0

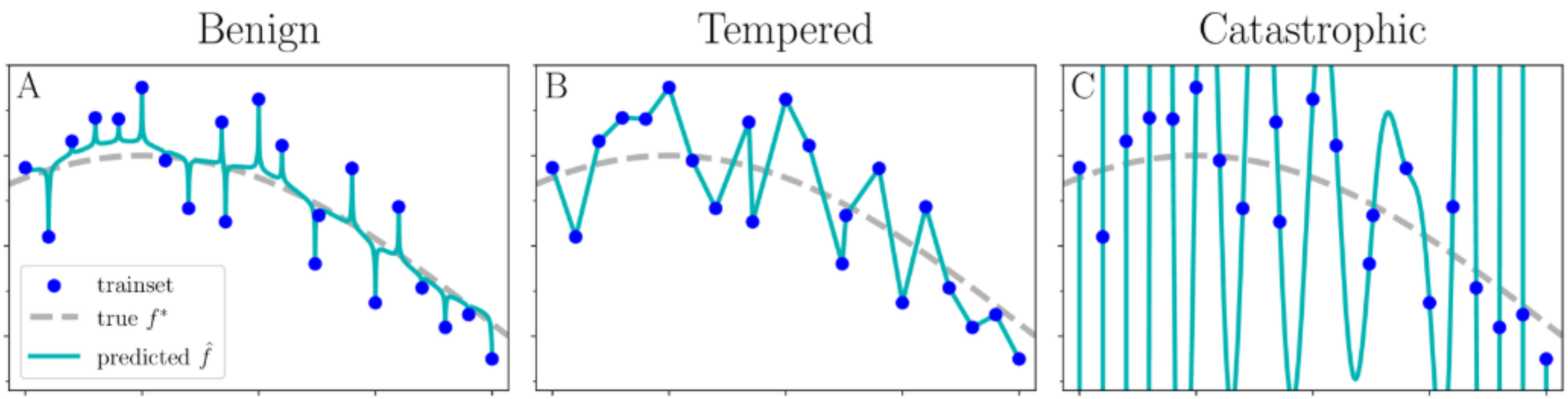


Figure 1: **As $n \to \infty$, interpolating methods can exhibit three types of overfitting. (A)** In *benign overfitting*, the predictor asymptotically approaches the ground-truth, Bayes-optimal function. Nadaraya-Watson kernel smoothing with a singular kernel, shown here, is asymptotically benign. **(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.

Illustration from (Mallinar et al. 2022)

Question #2: What solutions does GD with Large Stepsize find when labels are noisy?

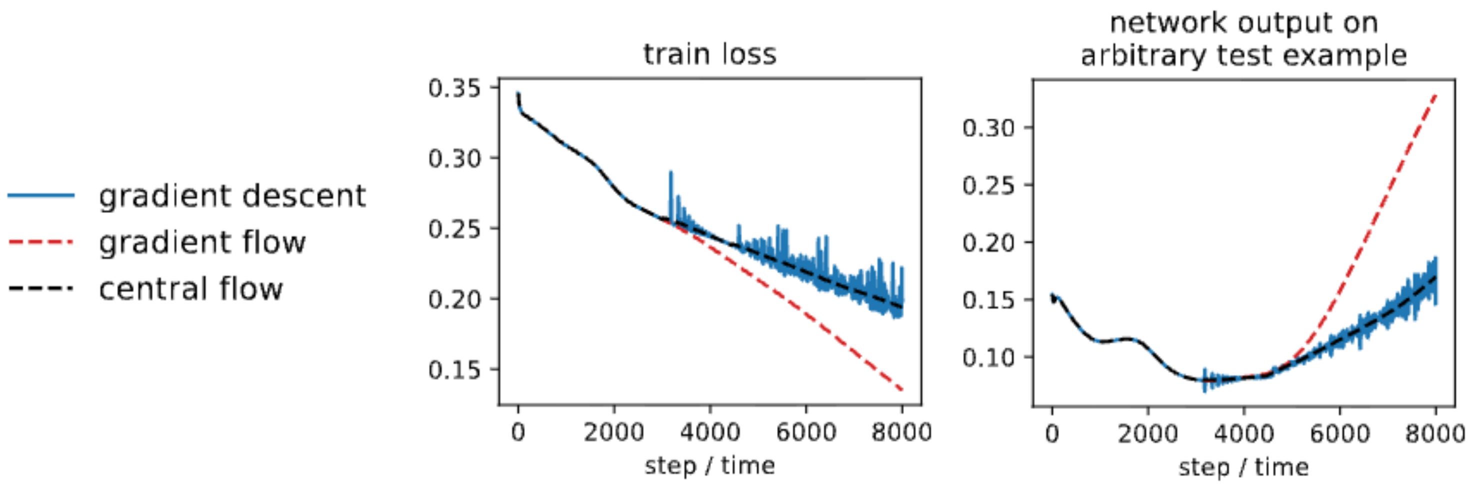
The implicit bias of "Large Stepsize" does not function in isolation.

- Data distribution
  - e.g., Low-dimensional structure, data-augmentation
- Choice of loss functions
  - e.g., Square loss, logistic loss
- Model architecture
  - e.g., with or without "bias", "residual connection", "batch-norm"
- Hyperparameters in training:
  - e.g., weight decay, momentum, adaptive optimizers

Question #3: How does GD with Large Stepsize interact with other "*forces of nature*"?

Gradient descent with *constant stepsize* is qualitatively different from gradient flow.

Cohen, Damian, Talwalkar, Kolter, Lee (2025) "Central Flows"

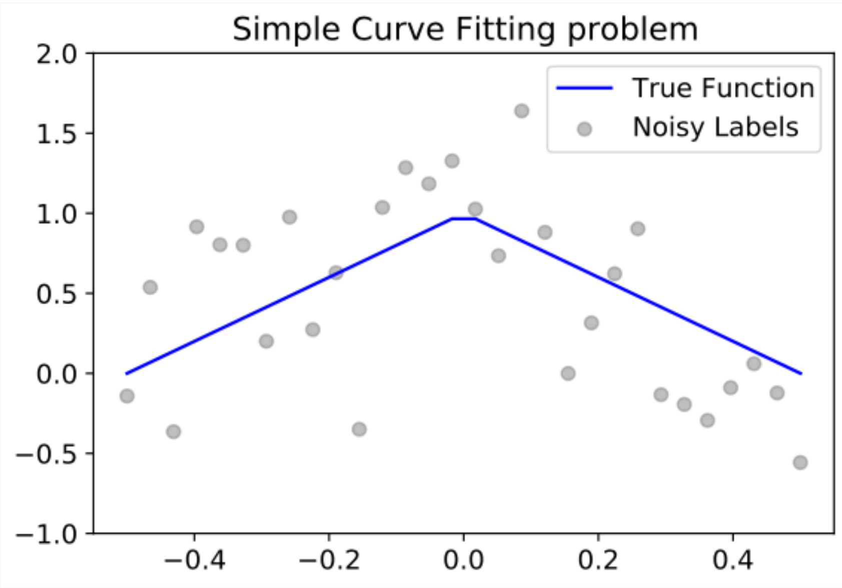


The dynamics is complex and **chaotic**. In: Kong and Tao (2020) "Stochasticity of Deterministic Gradient Descent"

What does the GD solution look like?

Let's start with a simple example.

Let us train an overparameterized ReLU NN on this “curve fitting” problem



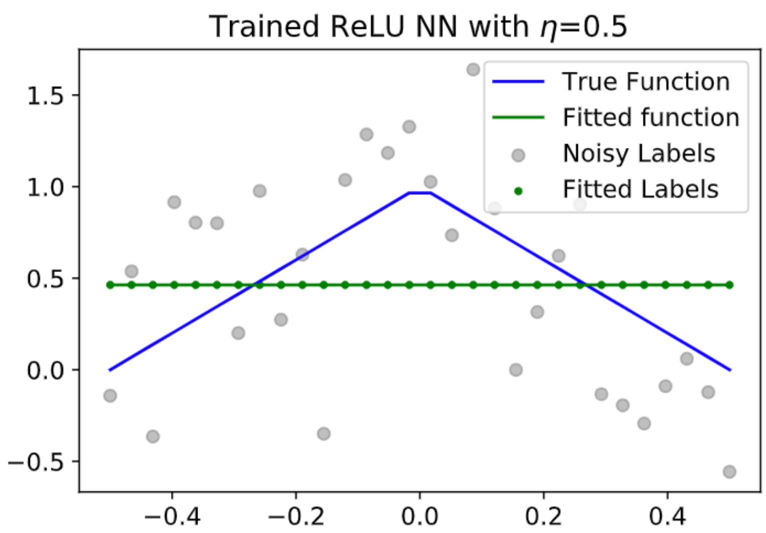
Global optimal solution has 0-loss, i.e., interpolating.

But does GD find these “interpolating” solution?

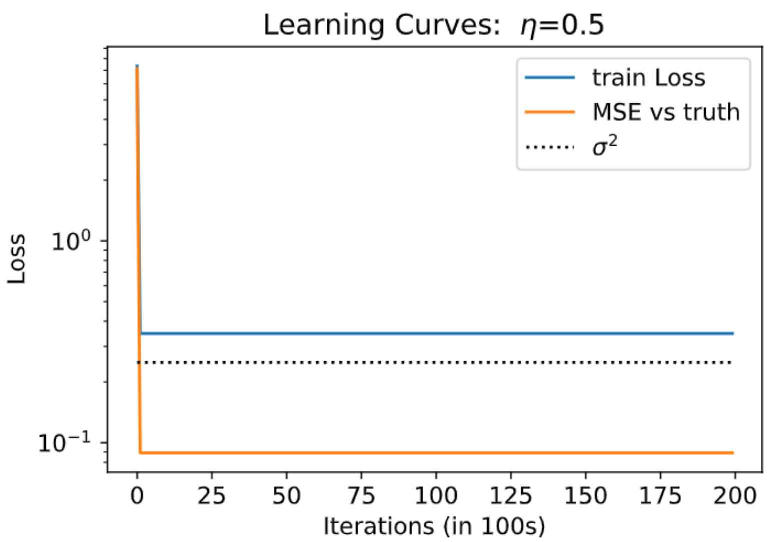
If so, does GD solution satisfies “Benign overfitting”?

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

Stepsize = 0.5

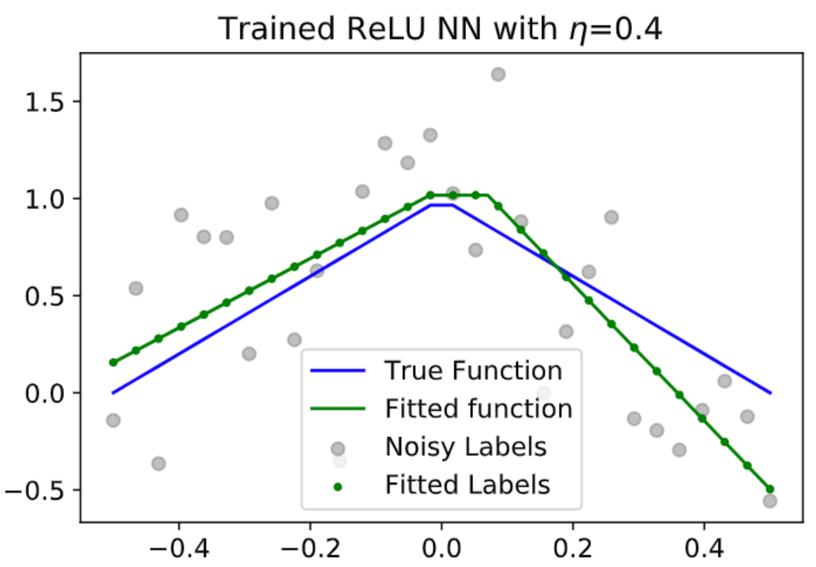


Trained ReLU NN with η=0.5
True Function
Fitted function
Noisy Labels
Fitted Labels
1.5
1.0
0.5
0.0
−0.5
−0.4
−0.2
0.0
0.2
0.4

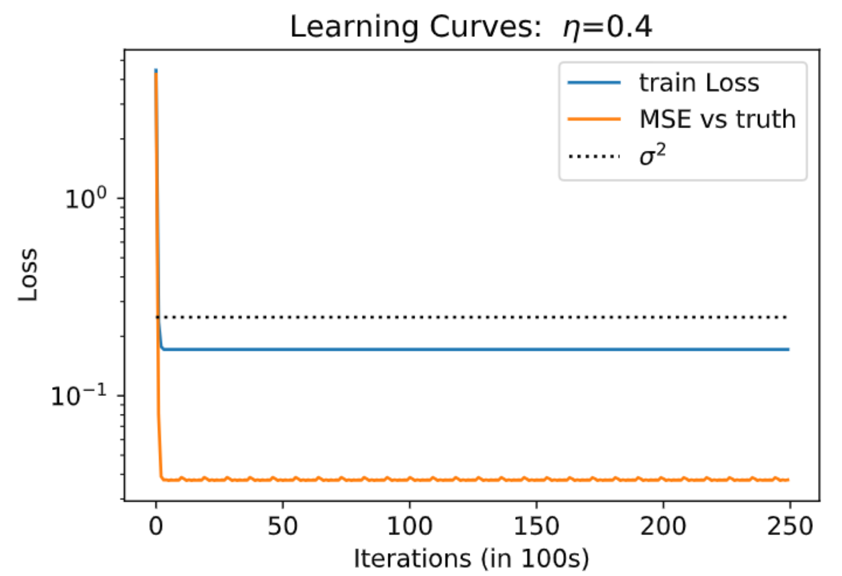


Learning Curves: η=0.5
train Loss
MSE vs truth
σ^2
Loss
10^0
10^{-1}
0
25
50
75
100
125
150
175
200
Iterations (in 100s)

Stepsize = 0.4

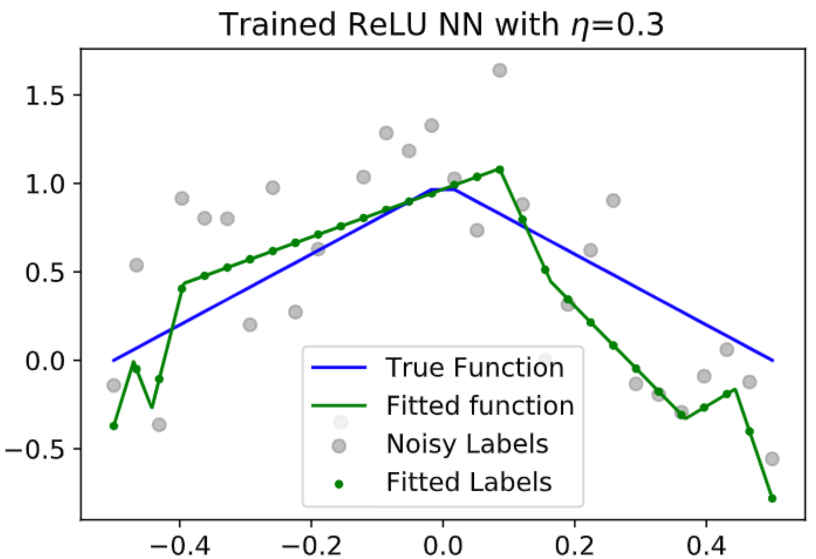


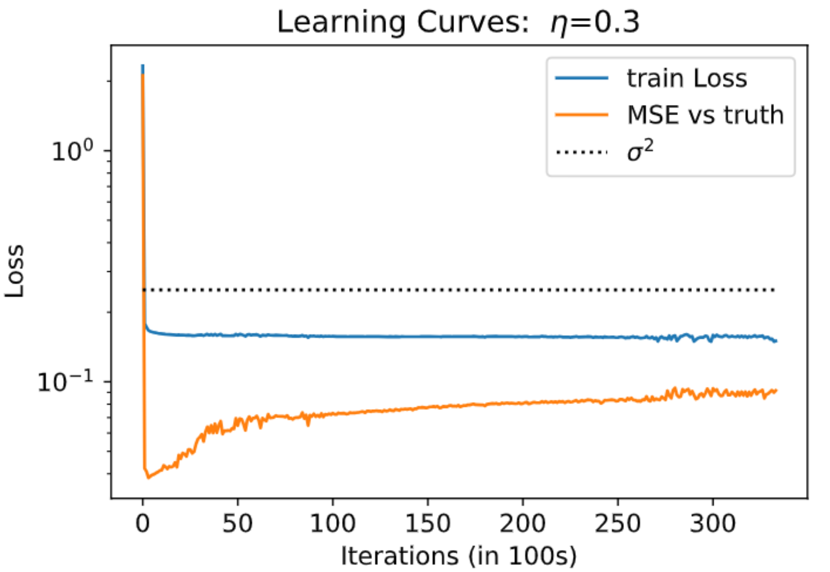
Trained ReLU NN with η=0.4
1.5
1.0
0.5
0.0
−0.5
−0.4
−0.2
0.0
0.2
0.4
True Function
Fitted function
Noisy Labels
Fitted Labels



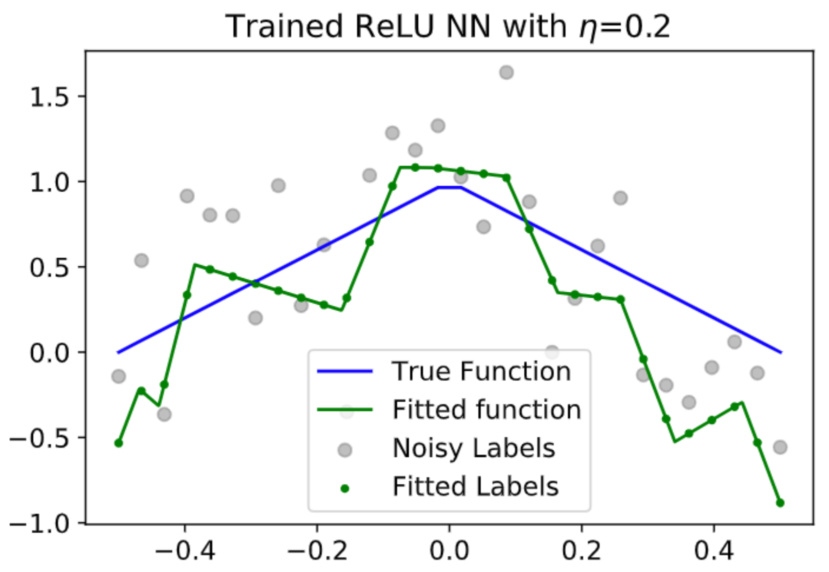
Learning Curves: η=0.4
Loss
10^0
10^{-1}
0
50
100
150
200
250
Iterations (in 100s)
train Loss
MSE vs truth
σ^2

Stepsize = 0.3

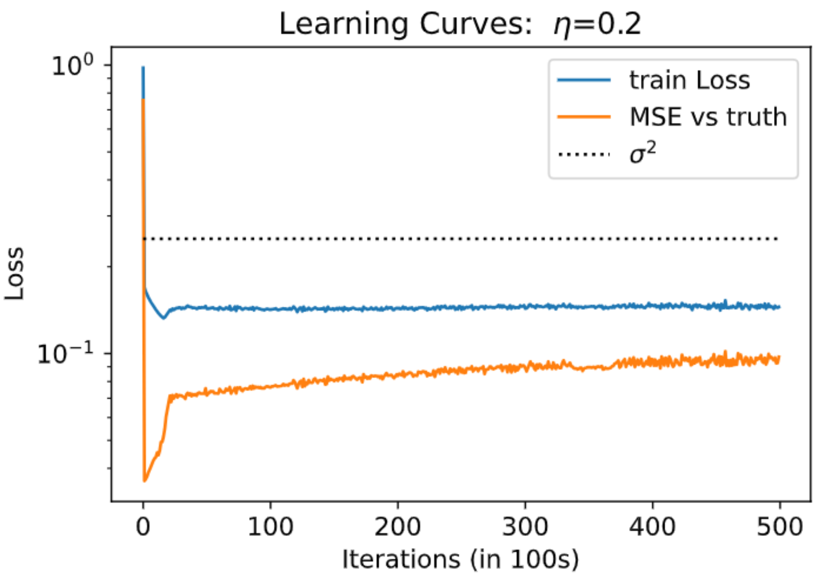

Trained ReLU NN with η=0.3
1.5
1.0
0.5
0.0
−0.5
−0.4
−0.2
0.0
0.2
0.4
True Function
Fitted function
Noisy Labels
Fitted Labels


Learning Curves: η=0.3
train Loss
MSE vs truth
σ^2
Loss
10^0
10^{-1}
0
50
100
150
200
250
300
Iterations (in 100s)

Stepsize = 0.2

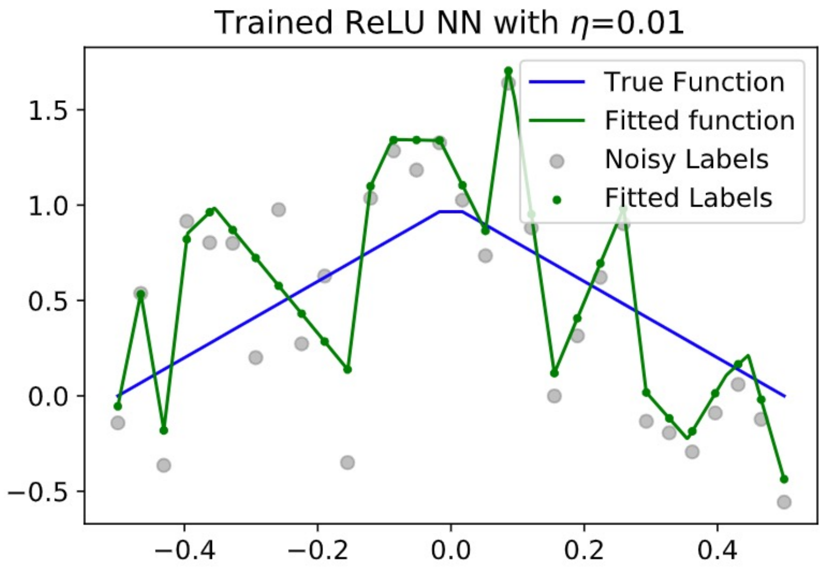


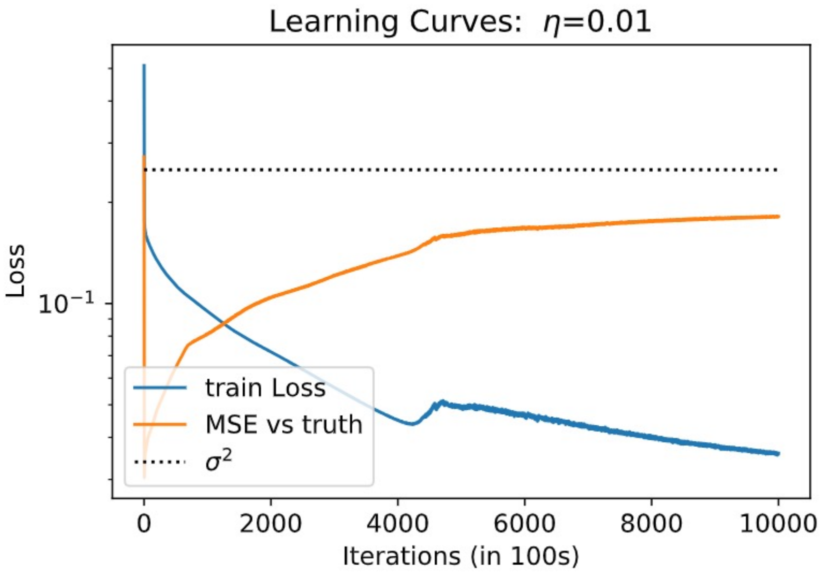
Trained ReLU NN with η=0.2
1.5
1.0
0.5
0.0
−0.5
−1.0
−0.4
−0.2
0.0
0.2
0.4
True Function
Fitted function
Noisy Labels
Fitted Labels



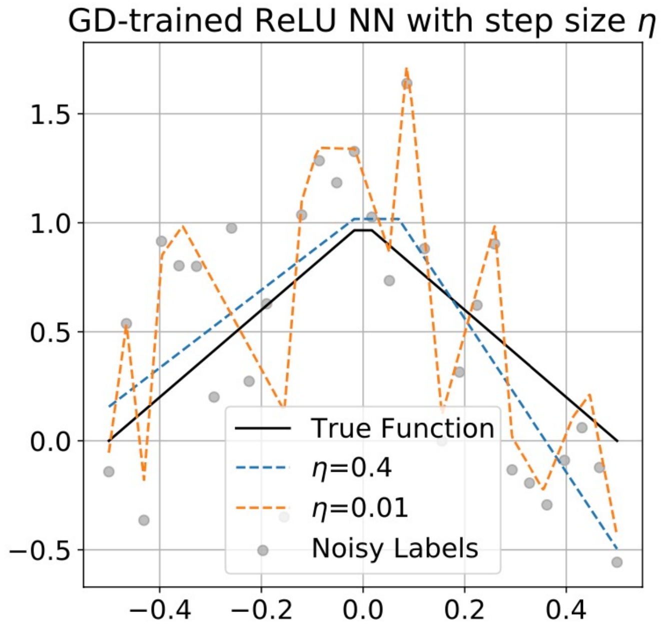
Learning Curves: η=0.2
10^0
10^{-1}
Loss
0
100
200
300
400
500
Iterations (in 100s)
train Loss
MSE vs truth
σ^2

Stepsize = 0.01





Observation: By tuning the stepsize, we are effectively tuning the number of “linear pieces”. GD with larger stepsize learns **simpler functions**.

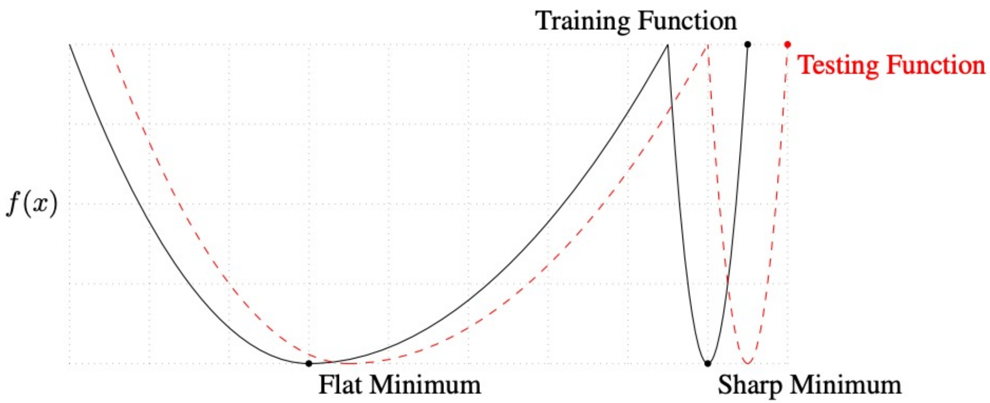


But how did “sparsity” emerge?

Is this a general phenomenon?
Did we get lucky?

Can we prove anything about this phenomenon rigorously?

Large stepsize is intimately connected **flat minima**, and *low-curvature* regions



Minima stability theory:
(Wu et al. 2018, Mulayoff et al. 2021)

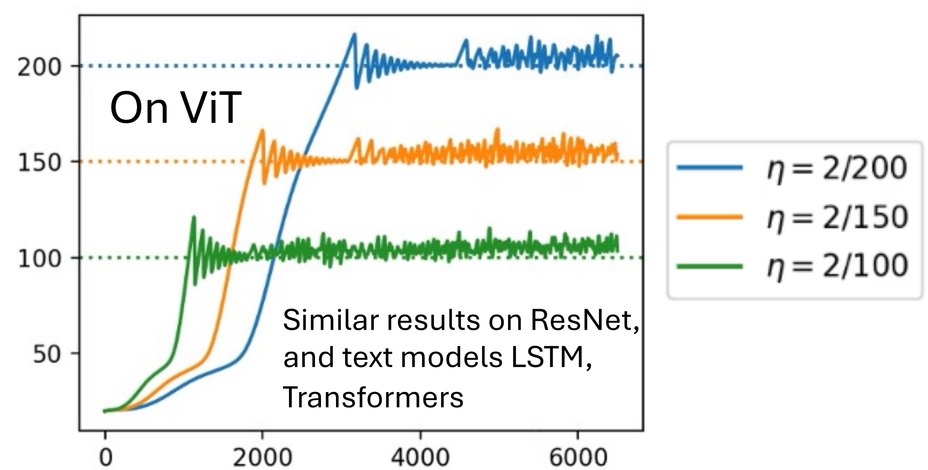
GD tend to diverge at sharp minima. The set of points GD can stabilize around:

$$\{f_\theta | \lambda_{\max}(\nabla^2 \mathcal{L}(\theta)) \leq 2/\eta, \nabla \mathcal{L}(\theta) = 0\}$$

Edge-of-Stability phenomenon
(Cohen et al, 2021; 2025)

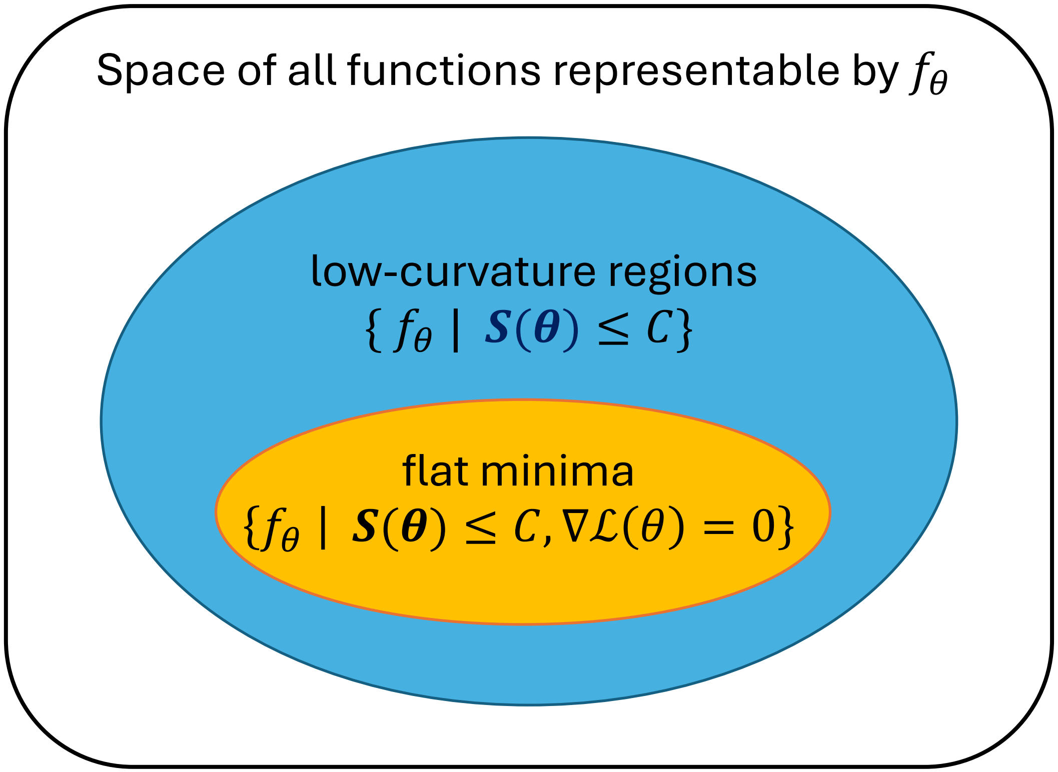
Entire GD trajectory stays inside the following set

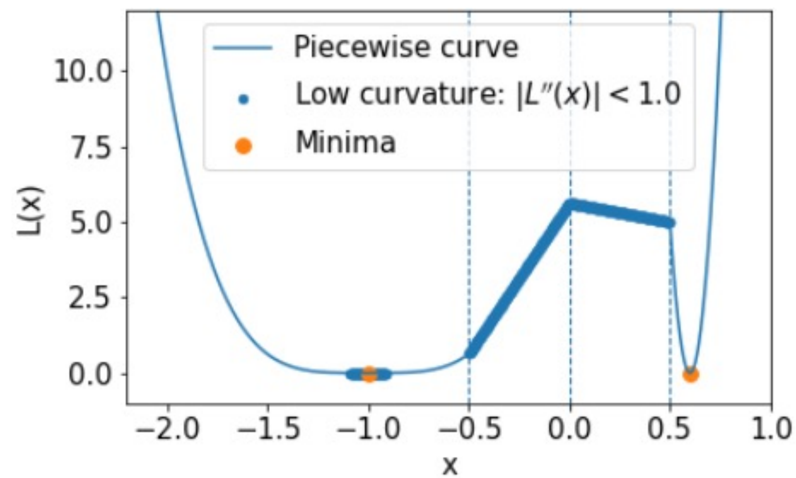
$$\{f_\theta | \lambda_{\max}(\nabla^2 \mathcal{L}(\theta)) \lesssim 2/\eta\}$$



(illustration from "Central Flow" Cohen et al, 2025)

Flat minima and flat points (low-curvature regions)





$\boldsymbol{S}(\boldsymbol{\theta}) := \lambda_{\max}\left(\nabla^2\mathcal{L}(\theta)\right)$ for Gradient Descent $C = 2/\eta$

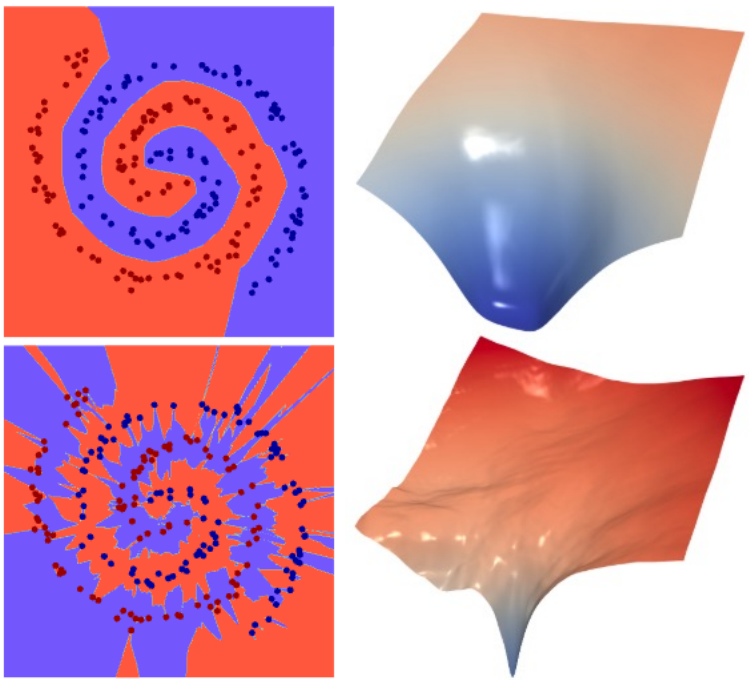
$\boldsymbol{S}(\boldsymbol{\theta}) := \mathrm{trace}\left(\nabla^2\mathcal{L}(\theta)\right)$ for Stochastic gradient descent $C = O(1/\eta)$

with stepsize = η

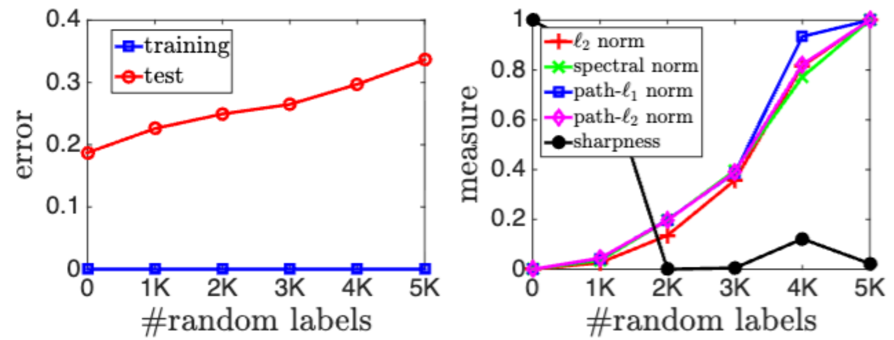
Do flat minima generalize better?

Deep learning folklore that **flat minima generalize better**.

(Hochreiter and Schmidhuber, 1997)



(Huang et al. 2018)



Very flat minima could also overfit.

"Exploring generalization in Deep Learning"
Neyshabur et al. 2017

Sharp Minima Can Generalize For Deep Nets

Laurent Dinh, Razvan Pascanu, Samy Bengio, Yoshua Bengio

How do we make sense of these conflicting observations?

Remainder of this tutorial

1. Flat minima **exactly recover** weights in Matrix Sensing and 2-layer Neural Nets (Maryam)

2. Does **flatness imply generalization** in 2-layer ReLU Neural Networks? (Yu-Xiang)

3. Discussion and Open problems. (Both)

Flat Minima and Generalization: Case studies in Low-rank Recovery and a 2-Layer Network

Outline of this part:

- Overparameterization, generalization & flatness
- Flatness via trace of Hessian
- Prove “flat minima generalize” in 2-layer test cases, including:
  - matrix sensing
  - a 2-layer neural net

Over-parameterization and some consequences

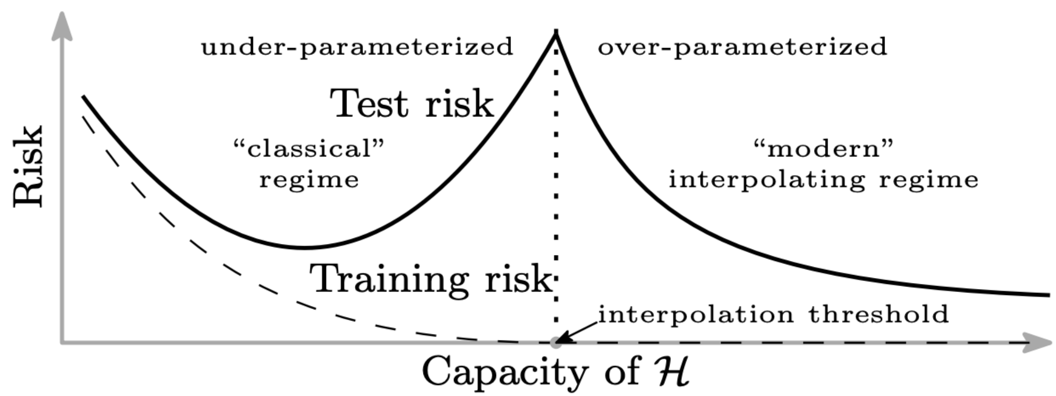
Recall: deep learning seeks **overparameterized** models

$$\min_{\theta \in \mathbb{R}^d} \; \mathcal{L}(\theta) := \frac{1}{n} \sum_{i=1}^{n} \ell(y_i, f_\theta(x_i))$$

where

$$\underbrace{\#\text{parameters}}_{d} \qquad \gg \qquad \underbrace{\#\text{samples}}_{n}$$

Evidence of **double descent phenomena** (or benign overfitting) in practice and in simple theory models



(Belkin, Hsu, Ma, Mandal '18)

Overparameterization $\implies$ **many** zero-loss solutions

Question: Why do some zero-loss (interpolating) solutions generalize, and others do not?

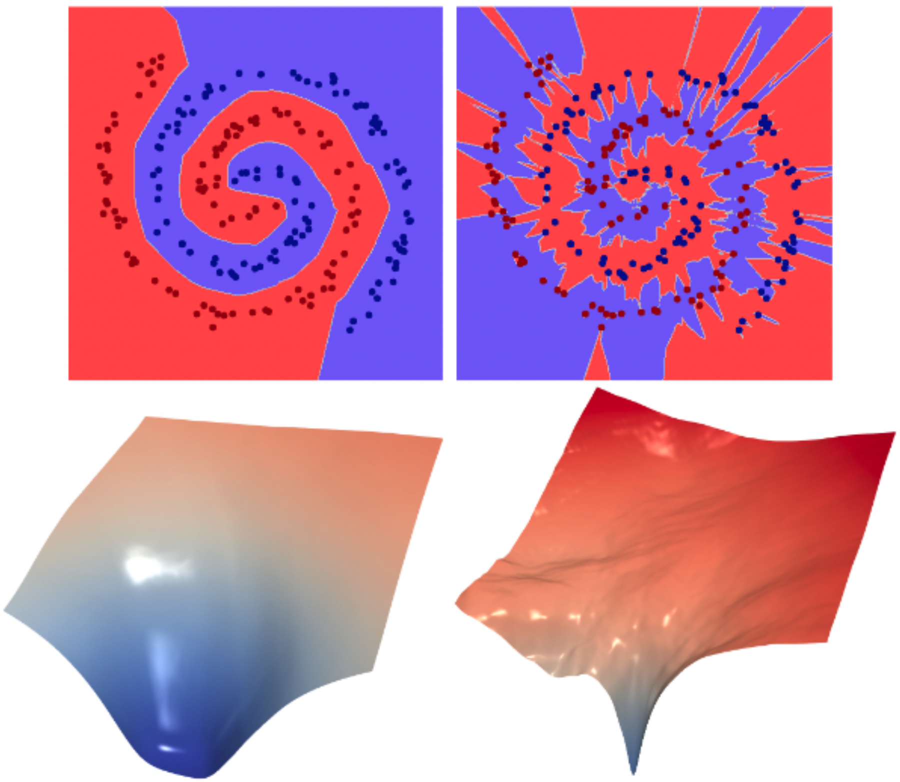
Value of training loss is **not enough**; other properties that predict good generalization?

1. explicit or implicit regularization (training algorithm)
2. **flatness** (loss function + architecture, ℓ and f_θ) $\rightarrow$ **this part**
   algorithm-agnostic, focus on loss landscape $\mathcal{L}(\theta)$

Empirical evidence favoring flatness

(Huang, Emam, Goldblum, Fowl, Terry, Huang, Goldstein '2020)

As seen earlier: Binary classification, with swiss-roll data:



- Classification boundaries (top), training loss landscapes (bottom), 6-layer network: left generalizes well (& more robust), right has perfect train accuracy but *bad generalization*

Can we prove flat minimizers generalize?

For many **over-parametrized low-rank matrix** recovery problems: Yes!

- matrix recovery/sensing
- matrix completion (approximate recovery)
- phase retrieval
- bilinear matrix sensing
- robust PCA
- one-hidden-layer NN with quadratic activation

Flat minima **exactly** recover the ground-truth generative model under standard statistical assumptions, i.e., they generalize (in a strong sense)

Ref: L. Ding, D. Drusvyatskiy, M. Fazel, Z. Harchaoui, *IMA Journal on Information and Inference*, 2024.

“Matrix sensing” problem

Problem: recover matrix $M_\sharp \in \mathbb{R}^{d\times d}$ from $b_i = \langle A_i, M_\sharp \rangle = \mathrm{Tr}$, where

$$\mathcal{A}(X) = (\langle A_1, X\rangle, \langle A_2, X\rangle, \ldots, \langle A_m, X\rangle)$$

and $r_\sharp := \mathrm{rank}(M_\sharp) \ll d$.

Classical approach: (Fazel et al. 01, '02, Recht-Fazel-Parrilo '10)

$$\min_{X\in\mathbb{R}^{d\times d}} \quad \underbrace{\|X\|_*}_{\text{complexity}} \qquad \text{subject to} \qquad \mathcal{A}(X) = b$$

- Explicit nuclear norm regularization: well-understood by now
- Possible to pick **low-complexity solutions** without this regularizer and just via 'flatness'?

Case study in nonconvex matrix sensing

Problem: recover matrix $M_\sharp \in \mathbb{R}^{d\times d}$ from $b = \mathcal{A}(M_\sharp)$, where

$$\mathcal{A}(X) = (\langle A_1, X\rangle, \langle A_2, X\rangle, \ldots, \langle A_m, X\rangle)$$

and $r_\sharp := \mathrm{rank}(M_\sharp) \ll d$.

Rewrite as **over-parametrized low-rank matrix recovery**:
Let $X = LR^T$,

$$\min_{L,R\in\mathbb{R}^{d\times k}} \mathcal{L}(L,R) = \|\mathcal{A}(LR^\top) - b\|_2^2$$

where $b = \mathcal{A}(M_\sharp)$ and

$$k \gg \mathrm{rank}(M_\sharp) := r_\sharp$$

'Learning' interpretation: A two-layer linear network

- (L,R) are the model parameters (layer weights)
- A_i, b_i are the data
- $M_\sharp$ captures the generative model (teacher network)

▶ a prototype for nonconvex learning (Gunasekar et al, '17, Du et al. '18, Li et al. '18, Tian and Du '18)

Flatness measure

(Zero-loss) solution set: $\mathcal{S} = \{(L,R) : \mathcal{A}(LR^\top) = b\}$

Second-order expansion around $(L,R) \in \mathcal{S}$:

$$\mathcal{L}(L+U, R+V) \quad \approx \quad \tfrac{1}{2} D^2\mathcal{L}(L,R)[U,V]$$

Flatness measure: $\operatorname{tr}(D^2\mathcal{L}(L,R))$

An **average measure of curvature**:

$$\operatorname{tr}(D^2\mathcal{L}(L,R)) = \; c \cdot \mathop{\mathbb{E}}_{U,V\sim\mathcal{N}(0,I)} \mathcal{L}(L+U, R+V)$$

Flat (flattest) solutions are the argmin of:

$$\boxed{\min_{L,R\in\mathbb{R}^{d\times k}} \; \underbrace{\operatorname{tr}(D^2\mathcal{L}(L,R))}_{\text{quadratic}} \qquad \text{subject to} \qquad \underbrace{\mathcal{A}(LR^\top)}_{\text{quadratic}} = b}$$

Warm-up: $\mathcal{A} = \mathcal{I}$

$$\min_{L,R \in \mathbb{R}^{d\times k}} \mathcal{L}(L,R) = \|LR^\top - M_\sharp\|_F^2$$

Second-order expansion around $(L,R) \in \mathcal{S}$:

$$D^2\mathcal{L}(L,R)[U,V] = 4\langle \underbrace{LR^\top - M_\sharp}_{=0}, UV^\top\rangle + 2\|LV^\top + UR\|_F^2$$

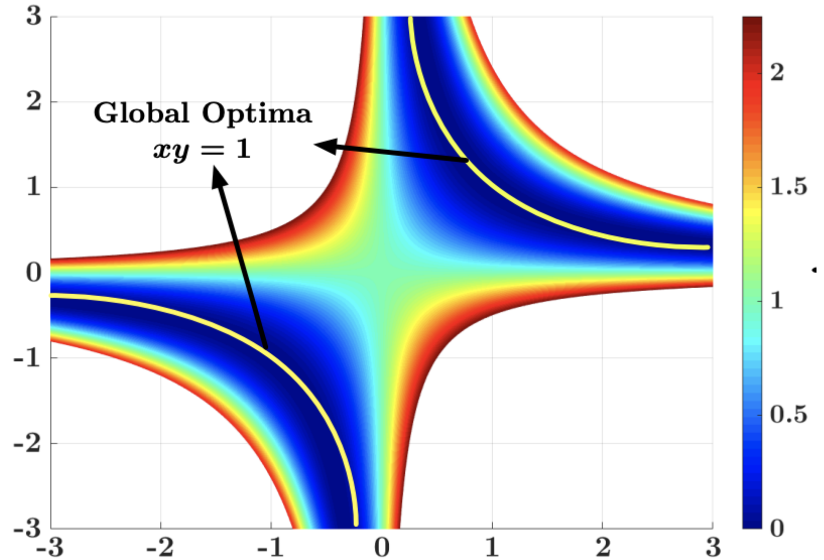


Figure: $l(x,y) = (xy-1)^2$. (1,1), (-1,-1) are flat solutions.

(we prove when $\mathcal{A} = \mathcal{I}$, flat is equivalent to "norm minimal" and "balanced")

Back to $\mathcal{A} \neq \mathcal{I}$

Goal: (Exact recovery)
Show that under standard statistical assumptions (on measurement map $\mathcal{A}$, i.e., randomness of data A_i) flat solutions $(L, R) \in \mathcal{S}$ satisfy $LR^\top = M_\sharp$.

Strategy: Show that $M_\sharp$ is the **unique solution** of the following convex relaxation of flatness maximization:

$$\boxed{\min_{X \in \mathbb{R}^{d \times d}} \quad \|D_1 X D_2\|_* \qquad \text{subject to} \qquad \mathcal{A}(X) = b}$$

where D_1 and D_2 are data-dependent weights, hence both objective and constraints are **data-dependent**.

Matrix sensing

Random data/measurements:

$$\mathcal{A}(X) = (\mathrm{tr}(A_1X), \mathrm{tr}(A_2X), \ldots, \mathrm{tr}(A_mX))$$

- Gaussian ensemble: A_i are i.i.d standard Gaussian (also holds for many more cases via matrix Restricted Isometry Property) (Recht-Fazel-Parrilo '10)

Theorem (Matrix sensing)

When $m \gtrsim r_\sharp d$, with probability at least $1 - e^{-\Omega(m)}$, any flat solution (L_f, R_f) satisfies

$$L_f R_f^\top = M_\sharp.$$

Moreover, for any $\delta > 0$ w.h.p. we have

$$\|L_f\|_F^2 + \|R_f\|_F^2 \leq (1+\delta)\|M_\sharp\|_* \qquad \text{[Norm-minimal]}$$

$$\|L_f^\top L_f - R_f^\top R_f\|_* \leq \delta\|M_\sharp\|_* \qquad \text{[Balanced]}$$

- matches sample complexity for nuclear norm minimization (though not the same solution)
- result extends to **noisy labels** (recovery up to noise level)

Case study: Single hidden-layer NN (quadratic activation)

Problem: (Li-Ma-Zhang '18, Soltanolkotabi et al. '18)

Given data $x \in \mathbb{R}^d$, output $y(x)$ is given by the "teacher" network

$$y(U_\sharp, x) = v^\top q(U_\sharp^\top x)$$

- $U_\sharp$ is $d \times r_\sharp$; $v \in \mathbb{R}^{r_\sharp}$ has r_1 positive and r_2 negative entries
- $q(s) = s^2$ applied coordinate-wise

Prediction $\hat{y}$ of the "student" NN on x can be expressed as

$$\hat{y}(U, x) = u^\top q(U^\top x)$$

with a fixed u, so problem simplifies to seeking U.

Overparameterized problem:

$$\min_{U \in \mathbb{R}^{d \times k}} \mathcal{L}(U) := \frac{1}{n} \sum_{i=1}^{n} (\hat{y}(U, x_i) - y_i)^2$$

Flatness: $U_f \in \mathcal{S}$ is **flat** if it solves the problem

$$\min_{U \in \mathcal{S}} \operatorname{tr}(D^2 \mathcal{L}(U)).$$

Exact recovery

Lemma (Reduction to matrix sensing): We can reformulate the loss as

$$\mathcal{L}([U_1, U_2]) = \frac{1}{n}\|\mathcal{A}(U_1U_1^\top - U_2U_2^\top - M_\sharp)\|_2^2,$$

where $A_i = x_i x_i^\top$ and $M_\sharp = U_\sharp \mathrm{diag}(v) U_\sharp^\top$.

Theorem (Exact recovery)

When $m \gtrsim r_\sharp d$, with probability at least $1 - e^{-\Omega(d)}$, any flat solution U_f recovers the teacher model $U_\sharp$.

Summary & take-away

- For a family of overparameterized nonconvex problems, flat minima do generalize!
- Relation to other properties: norm minimality ("weight decay"), balancedness
- Ideas from *compressed sensing*, *low-rank recovery* are useful
- Some implications:
  - regularization: (approximate) Hessian trace can serve as a good regularizer
  - algorithmic: a theoretical basis for methods that bias iterates towards flat solutions

Remainder of this tutorial

1. Flat minima **exactly recover** weights in Matrix Sensing and 2-layer Neural Nets (Maryam)

2. Does **flatness imply generalization** in 2-layer ReLU Neural Networks? (Yu-Xiang)

3. Discussion and Open problems. (Yu-Xiang and Maryam)

So far, we considered "exact recovery" and "stable recovery" by flat minima.

- Can we weaken the data assumptions?
  - No assumption on the labeling function
- What can we say about other points GD discovers?
  - No interpolation. Not even local minima, e.g., early stopping.
- Can we obtain results for more realistic neural networks?
  - ReLU activation? Training all weights.

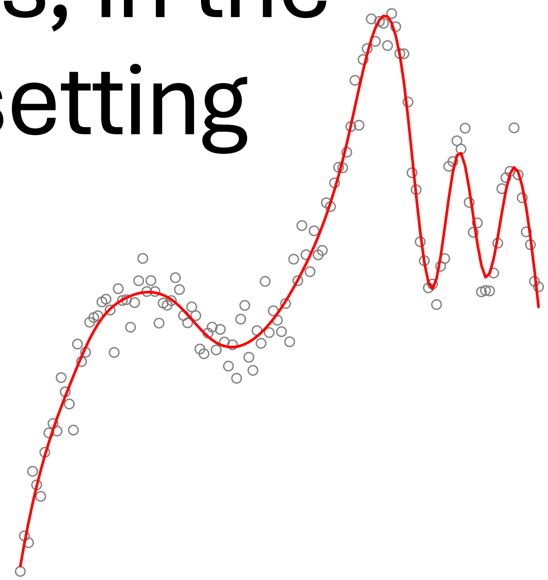
Problem setup: statistical theory of ML

- Data $(x_1, y_1), \ldots, (x_n, y_n) \in \mathcal{X} \times \mathcal{Y}$
- A family of models $\mathcal{F}$ parameter space Θ
- Each element $f_\theta : \mathcal{X} \to \mathcal{Y}$
- Loss function $\ell : (\mathcal{X} \times \mathcal{Y}) \times \mathcal{F} \to \mathbb{R}$
- Training: try to minimize the loss on training data

How do we measure generalization?

- Loss function ℓ
- Train loss (empirical risk): $\frac{1}{\mathrm{n}}\sum_i \ell(train\_data_i, f)$
- Test loss (aka risk): $\mathbb{E}_{data\sim P}[\ell(data, f)]$
- **Generalization Gap = |Training Loss - Test Loss|**
  - Useful when we do not make strong assumptions about the data.

In the case of the square loss, in the non-parametric regression setting



- If $\boldsymbol{y_i = f_0(x_i) + N(0, \sigma^2)}$

- Then:

$$\mathrm{MSE}(f) \coloneqq \mathbb{E}\left[\left(f(x) - f_0(x)\right)^2\right]$$

$$= \underbrace{\mathbb{E}[(f(x) - y)^2] - \overbrace{\mathbb{E}[(f_0(x) - y)^2]}^{\sigma^2}}_{\text{"Excess Risk", aka "Regret"}}$$

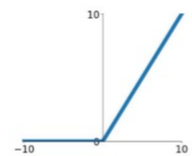
$$\leq TrainLoss(f) \; - \sigma^2 + Gen.Gap(f)$$

We consider two-Layer ***overparameterized*** *ReLU*-Neural Networks

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

- ReLU activation

ReLU
$\max(0, x)$



- Square loss

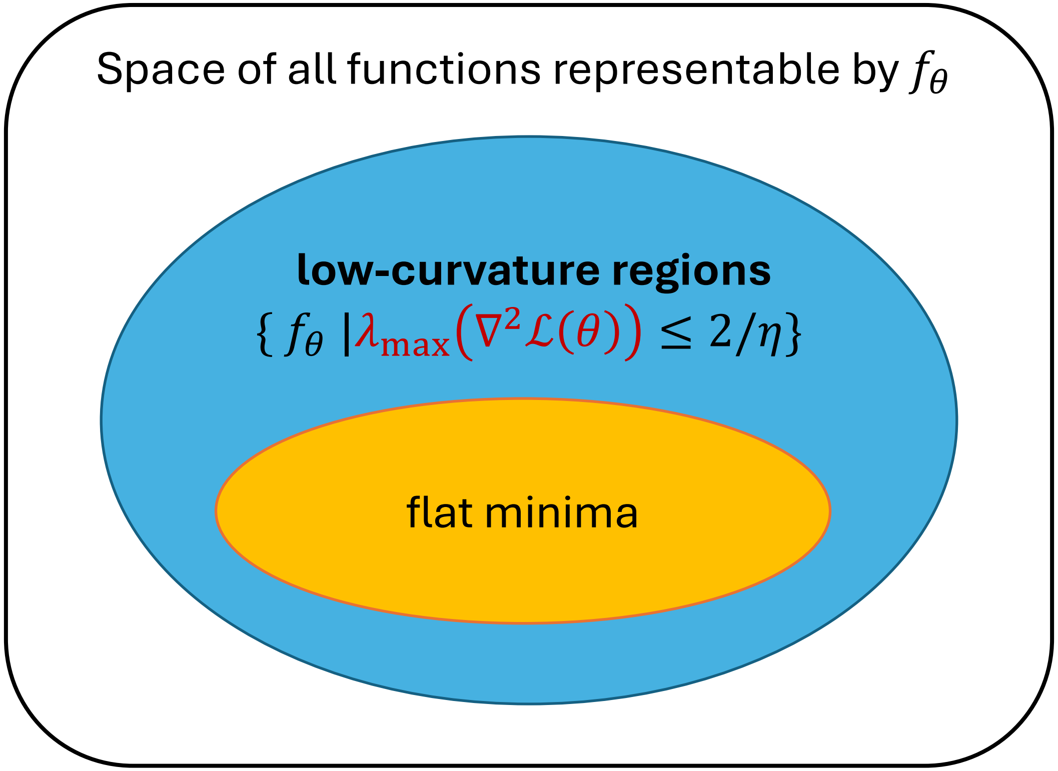
$$\mathcal{L}(\theta) = \frac{1}{2n} \sum_{i=1}^{n} (f_\theta(x_i) - y_i)^2$$

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

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

Stepsize (aka learning rate) parameter

Recall that GD finds points in low-curvature region: $\{f_\theta | \lambda_{\max}(\nabla^2 \mathcal{L}(\theta)) \leq 2/\eta\}$

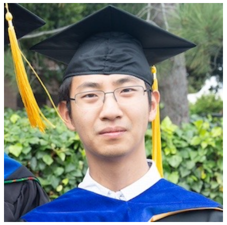


We will study the generalization of the whole class via **Uniform Convergence.**

Note: The set is data-dependent, since $\mathcal{L}$ depends on training data.

Our plan is to focus on the following work.

- Univariate–input + Square loss
  - Qiao, Zhang, Singh, Soudry, Wang. (2024) **Stable Minima Cannot Overfit in Univariate ReLU Networks: Generalization by Large Step Sizes**: https://arxiv.org/abs/2406.06838

- (If time permit) more general cases
  - Logistic loss: (Qiao et al. 2025)
  - High-dimension: (Liang et al. 2025a)
  - Adaptation and data-geometry: (Liang et al. 2025b)

What does class look like? **A Weighted TV1** class.

$$\left\{ f_\theta \middle| \lambda_{\max}(\nabla^2 \mathcal{L}(\theta)) \leq 2/\eta \right\}$$

$$\subseteq$$

$$\left\{ f \;\middle|\; \int |f''(x)| g(x) dx \leq C \right\} =: \mathrm{TV}_g^{(1)}(C)$$

where $C = 2/\eta + \tilde{O}(1)$

Mulayoff, Rotem, Tomer Michaeli, and Daniel Soudry. "The implicit bias of minima stability: A view from function space." *NeurIPS'2021*

Qiao et al. (2024) Stable Minima Cannot Overfit in Univariate ReLU Networks: Generalization by Large Step Sizes. NeurIPS'2024

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

Theorem (Qiao, Zhang, Singh, Soudry and W., 2024): Let f be any function represented by a ReLU activated two-layer NN f_θ. Let $\mathcal{L}(\theta)$ be the square (training) loss.

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

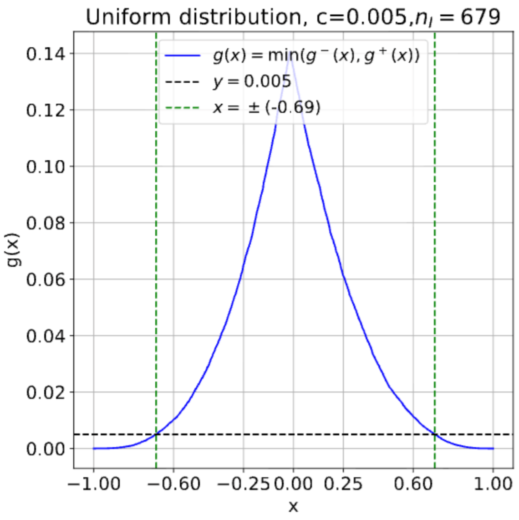
Assume data is coming from $\boldsymbol{y_i = f_0(x_i) + noise}$, then w.h.p.

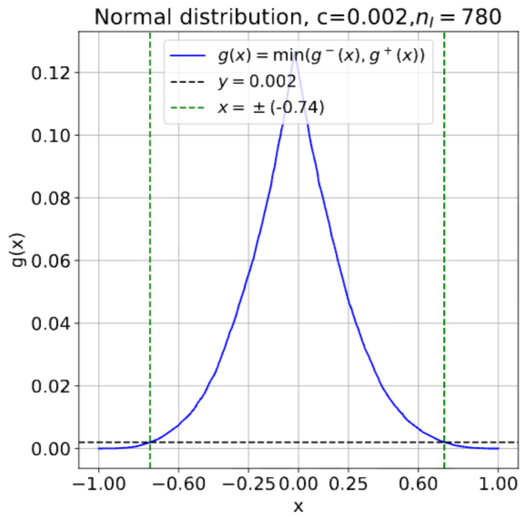
$$\int_{-x_{\max}}^{x_{\max}} |f''(x)| g(x) dx \leq \frac{\lambda_{\max}(\nabla_\theta^2 \mathcal{L}(\theta))}{2} - \frac{1}{2} + \widetilde{O}\left(\sigma x_{\max} \cdot \min\left\{1, \sqrt{\frac{k}{n}}\right\}\right) + x_{\max}\sqrt{\mathrm{MSE}(f)}.$$

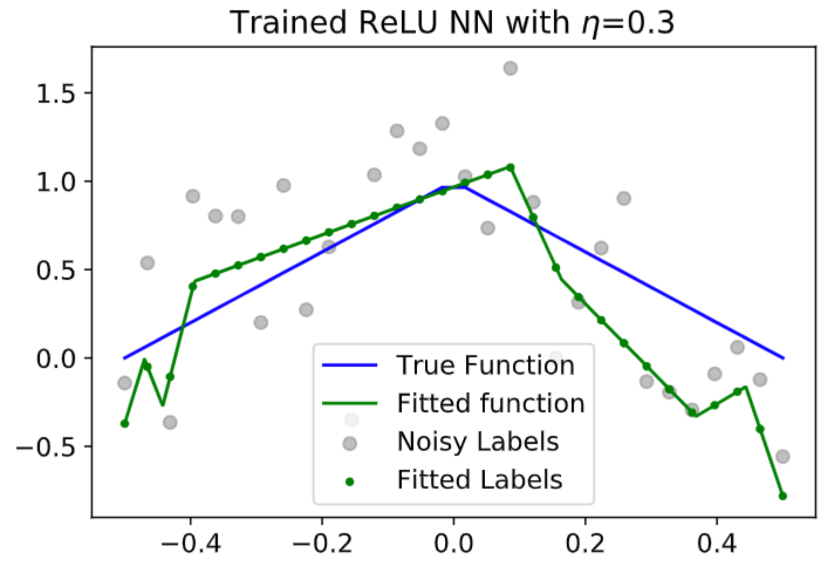
- Tune learning rate => select smoothness of f
- Smoothness of f => Generalization bounds

The weighting function g(x) depends only on the distribution of x.

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







The implicit regularization is **stronger in the interior** of the data distribution...
Nearly no regularization towards the boundaries.

Interpolating solutions must have high curvature (must be sharp)

- Theorem from the previous slide

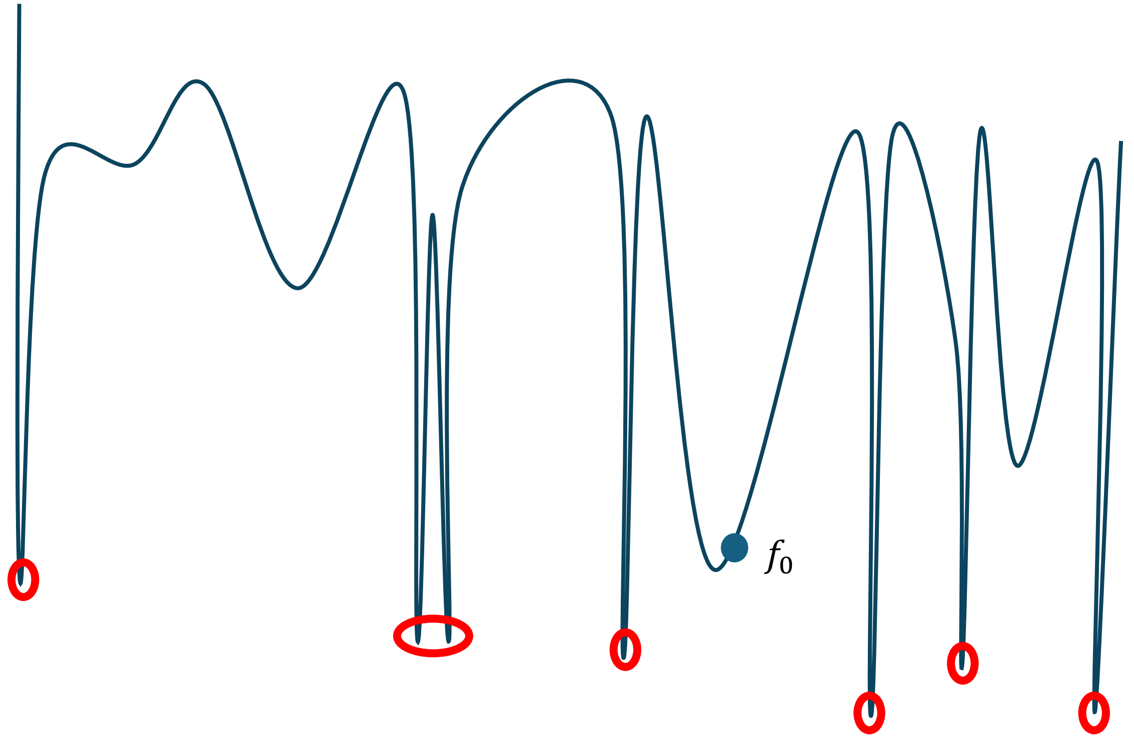
$$\int_{-x_{\max}}^{x_{\max}} |f''(x)| g(x) dx \leq \frac{\lambda_{\max}(\nabla_\theta^2 \mathcal{L}(\theta))}{2} - \frac{1}{2} + x_{\max}\sqrt{2\mathcal{L}(\theta)},$$

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

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

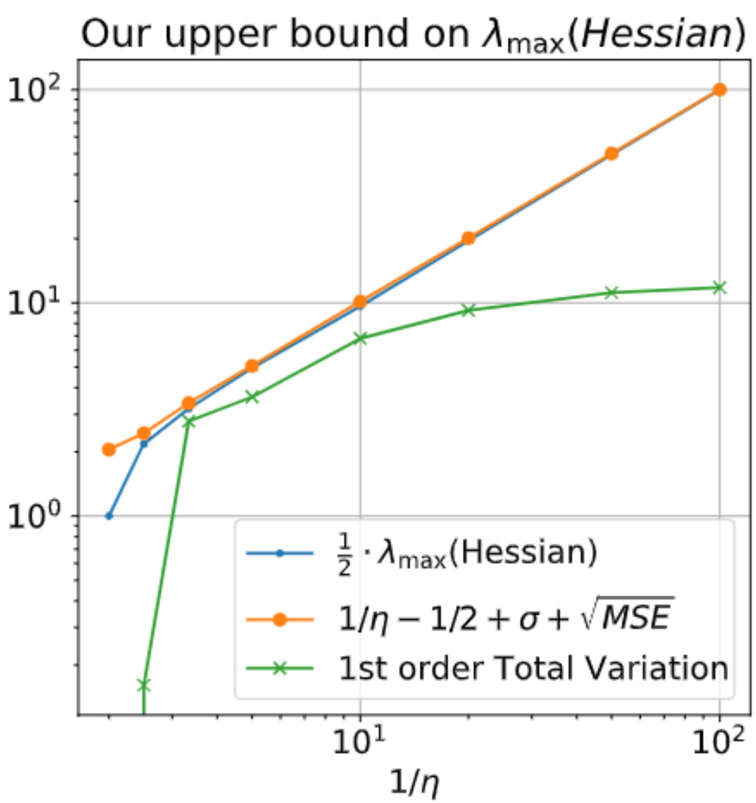
- Implies that stepsize η needs to be extremely small $O\left(\frac{1}{n^2\sigma}\right)$ for GD to stably converge to interpolating solutions.

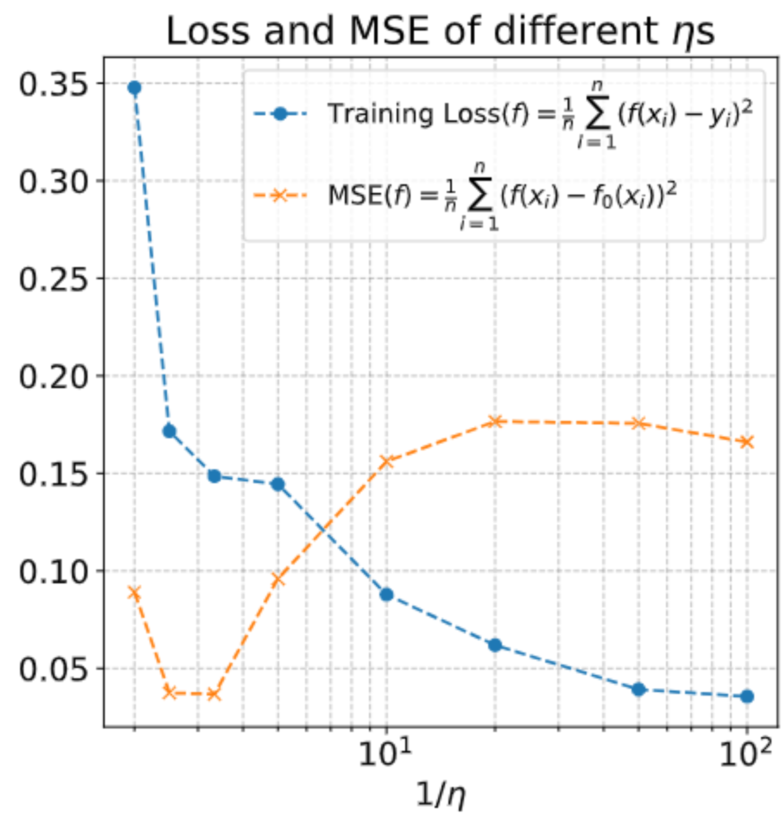
It tells us something new about the energy landscape of overparameterized NN training on noisy problems



Training with GD automatically avoids these sharp and overfitting solutions

Edge-of-Stability appears to hold. 2/η 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:**

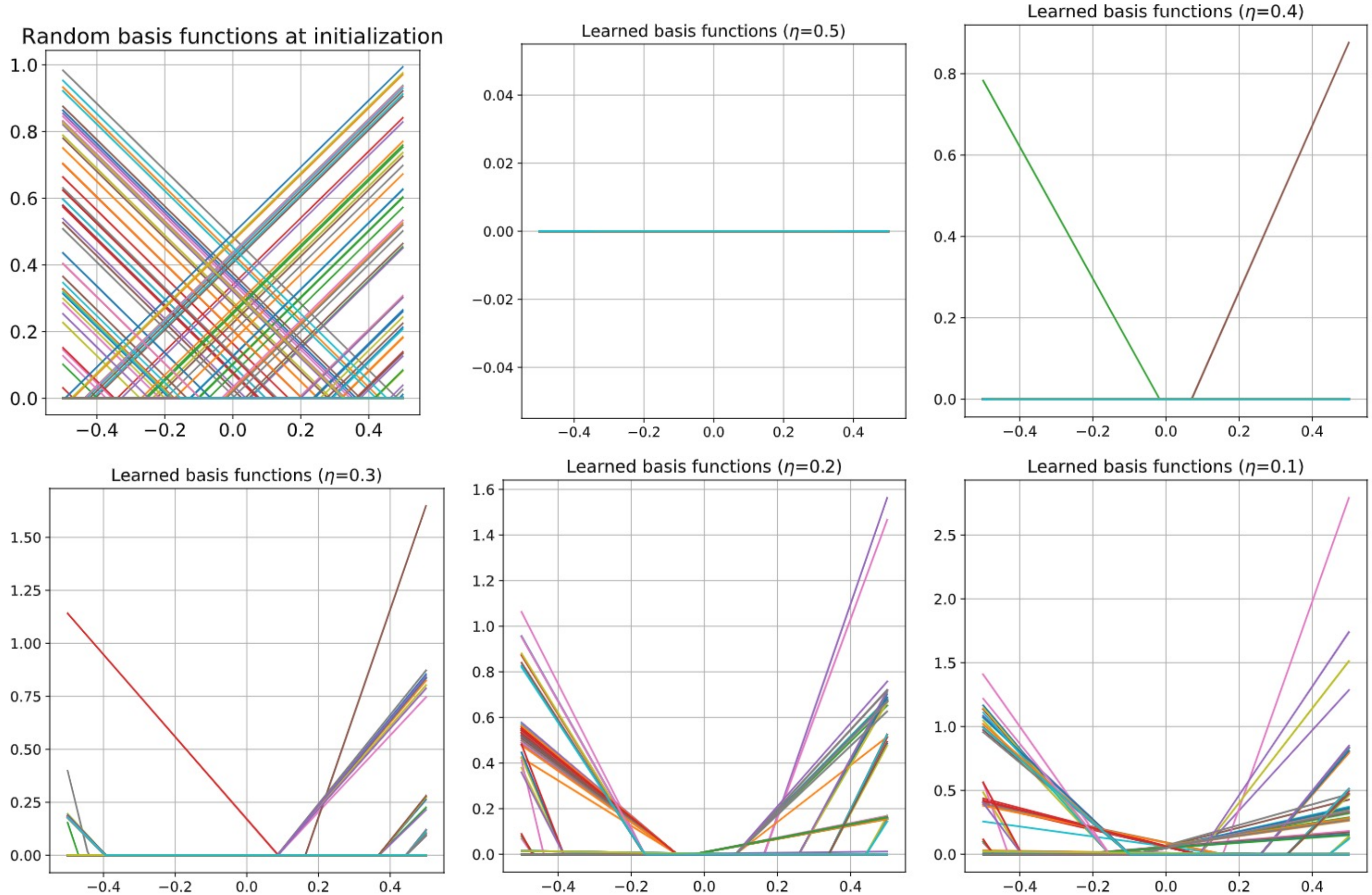
1. Agnostic case: generalization gap = O(n^{-2/5})
2. In the non-parametric regression setting, if training loss smaller than σ^2 then w.h.p., get an MSE

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

* near minimax optimal (for estimating TV1-functions).

	NN with optimally tuned stepsize	Kernel ridge regression (any RKHS)
MSE	$O(n^{-4/5})$	$\Omega(n^{-3/4})$

Large-stepsize generalizes better due to extensive "Feature learning": only a few neurons are active!



Checkpoint:

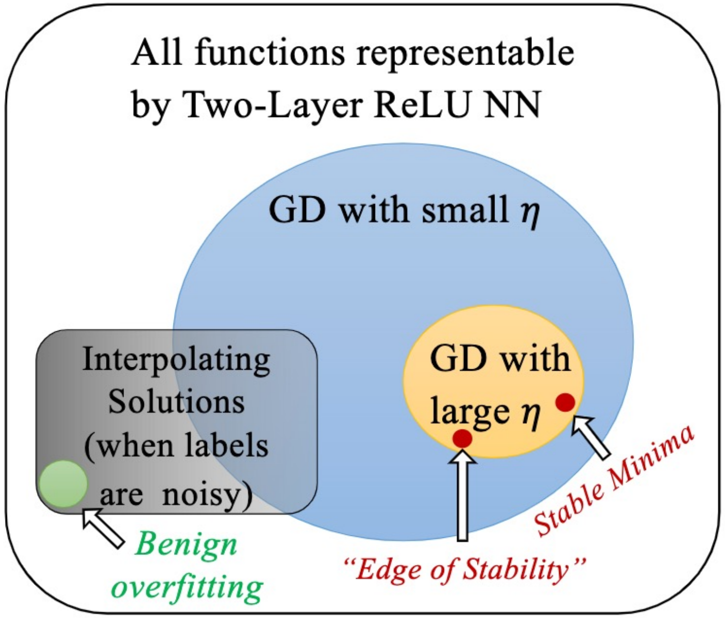
Qiao et al. (2024) **Stable Minima Cannot Overfit in Univariate ReLU Networks: Generalization by Large Step Sizes:**

https://arxiv.org/abs/2406.06838

- 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.

- Provably stronger than NTK. New insight into representation learning.



Extension of the theory

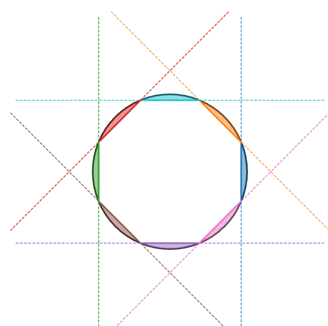
Qiao and W. (2025) **Does Flatness imply Generalization for Logistic Loss in Univariate Two-Layer ReLU Network?:** https://arxiv.org/abs/2512.01473

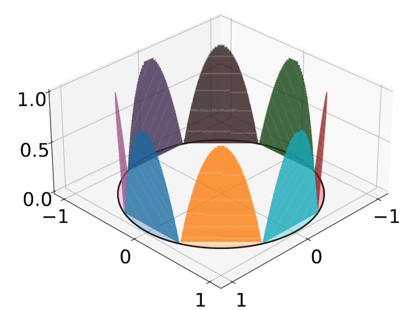
$$\{ f_\theta \mid \lambda_{\max}\left(\nabla^2 \mathcal{L}(\theta)\right) \leq 2/\eta\}$$ insufficient for generalization.

$$\{ f_\theta \mid \lambda_{\max}\left(\nabla^2 \mathcal{L}(\theta)\right) \leq \frac{2}{\eta},\ \|\boldsymbol{\theta}\| = \boldsymbol{o}(\boldsymbol{n})\}$$ works.

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

(NeurIPS 2025 Spotlight)

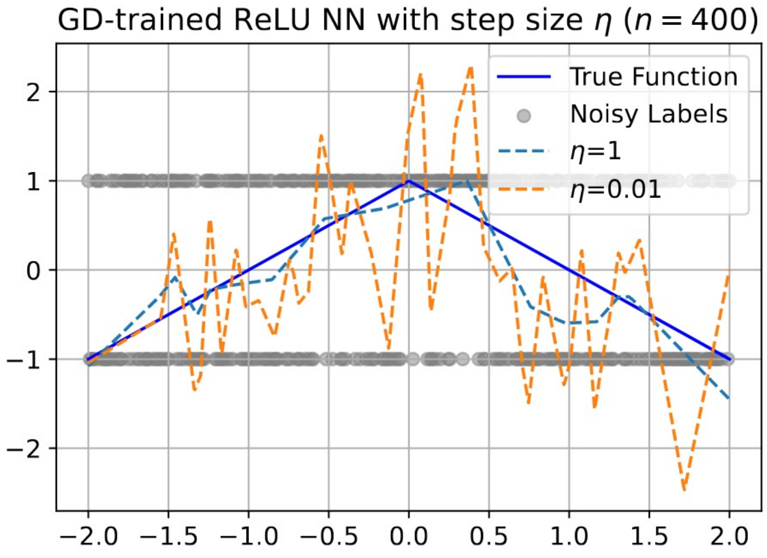




Liang, Cloninger, Parhi and W. (2025) **Generalization Below the Edge of Stability: The Role of Data Geometry**: https://arxiv.org/abs/2506.20779

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

- Empirically, kinda yes.
- Data: y ~ Bernoulli(Sigmoid(f_0(x))



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

$$\left\{ f_\theta \middle| \lambda_{\max}(\nabla^2 \mathcal{L}(\theta)) \leq 2/\eta \right\}$$

$$\subseteq$$

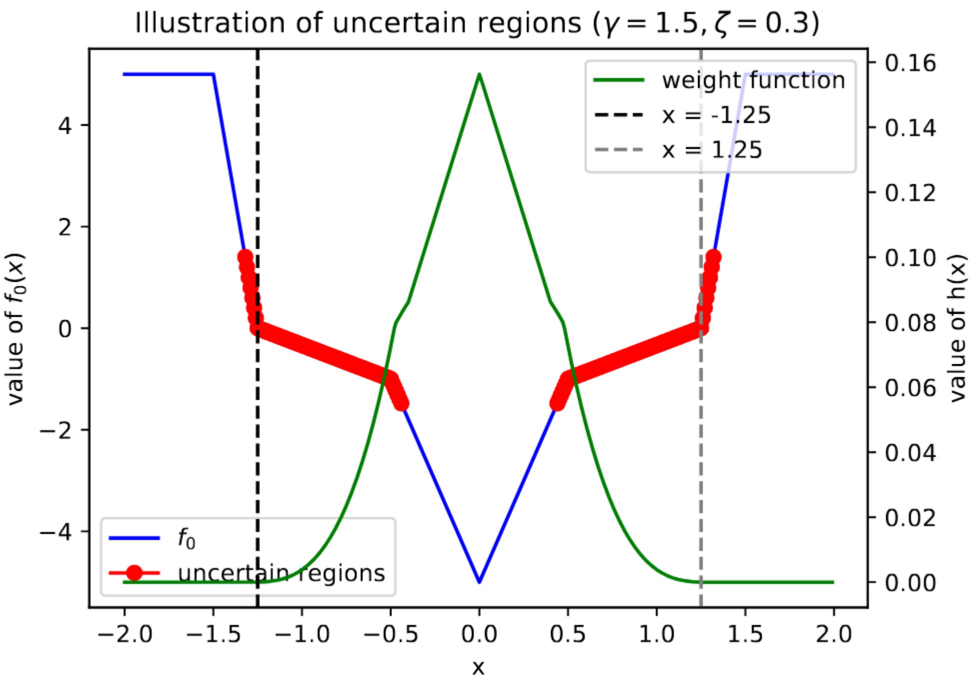
$$\left\{ f \middle| \int |f''(x)| \boxed{g(x)} dx \leq \frac{2}{\eta} \right\}$$

But the weighting function g now depends on f!

Qiao and W. (2025) "Does Flatness imply Generalization for Logistic Loss in Univariate Two-Layer ReLU Network?" New manuscript.

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

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



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

1. interpolating

2. arbitrarily flat loss

"flat" when simple and generalizing

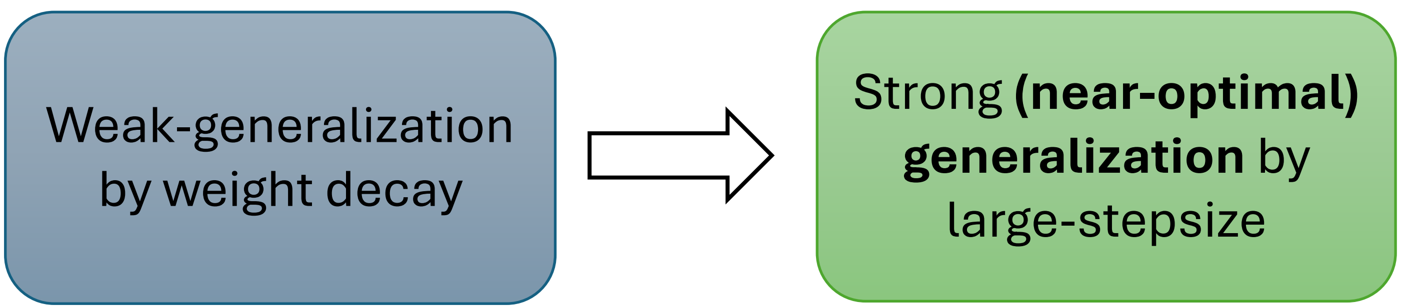
But also "flat" if you are **confidently interpolating** training data.

$\{ f_\theta \mid \lambda_{\max}(\nabla^2 \mathcal{L}(\theta)) \leq 2/\eta \}$ **insufficient** for generalization.

Why does it still generalize in the non-parametric classification setting?

- Assumption: y ~ Bernoulli(Sigmoid(f_0(x))
  - f_0 is bounded.

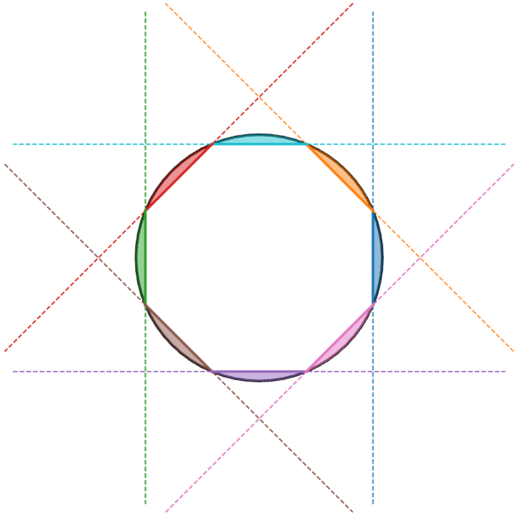
(Informal) Claim: within the convex hull of the uncertain region of f_0, near ***optimal excess risk*** for an "optimized" $f \in \{ f_\theta \mid \lambda_{\max}\left(\nabla^2 \mathcal{L}(\theta)\right) \leq \frac{2}{\eta},\ \|\boldsymbol{\theta}\| = \boldsymbol{o}(\boldsymbol{n}) \}$

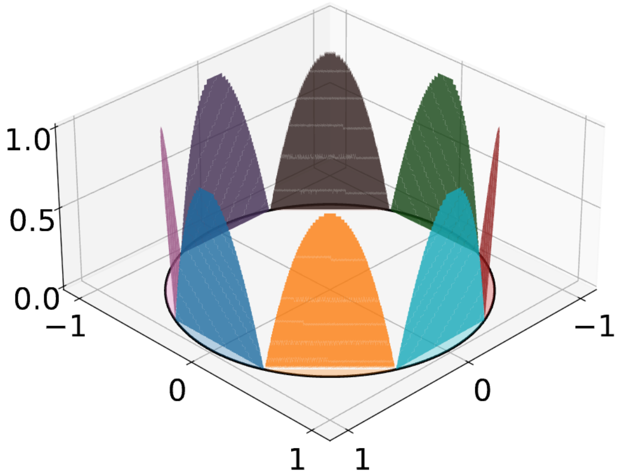


Qiao and W. (2025) **Does Flatness imply Generalization for Logistic Loss in Univariate Two-Layer ReLU Network?**

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

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

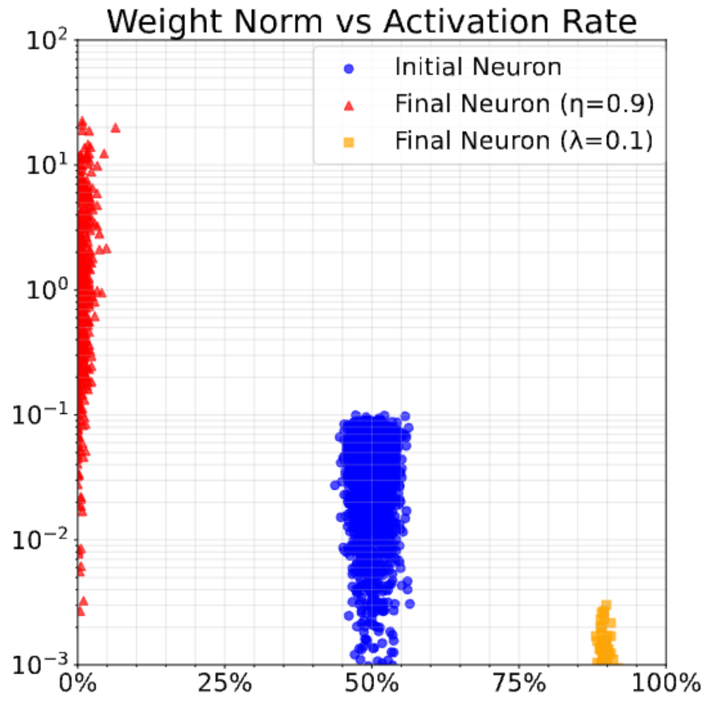




Liang, T., Qiao, D., Wang, Y. X., & Parhi, R. (2025). Stable Minima of ReLU Neural Networks Suffer from the Curse of Dimensionality: The Neural Shattering Phenomenon. *NeurIPS'25*.

Neural Shattering does not happen if there is **weight decay** or if we **remove "bias"** parameter from MLP

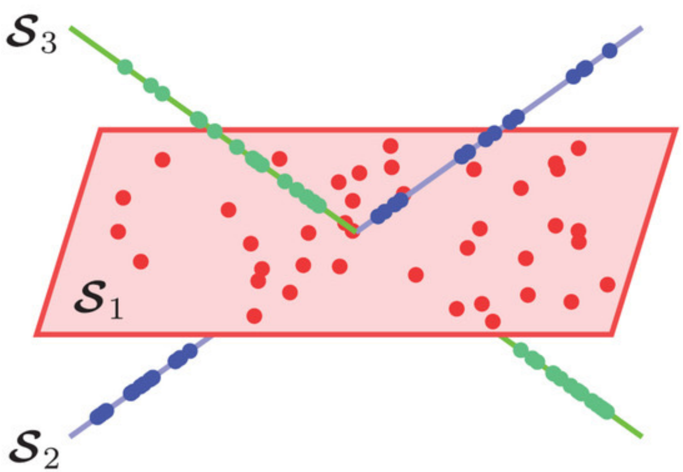
Weights $|w_i^{(1)} w_i^{(2)}|$

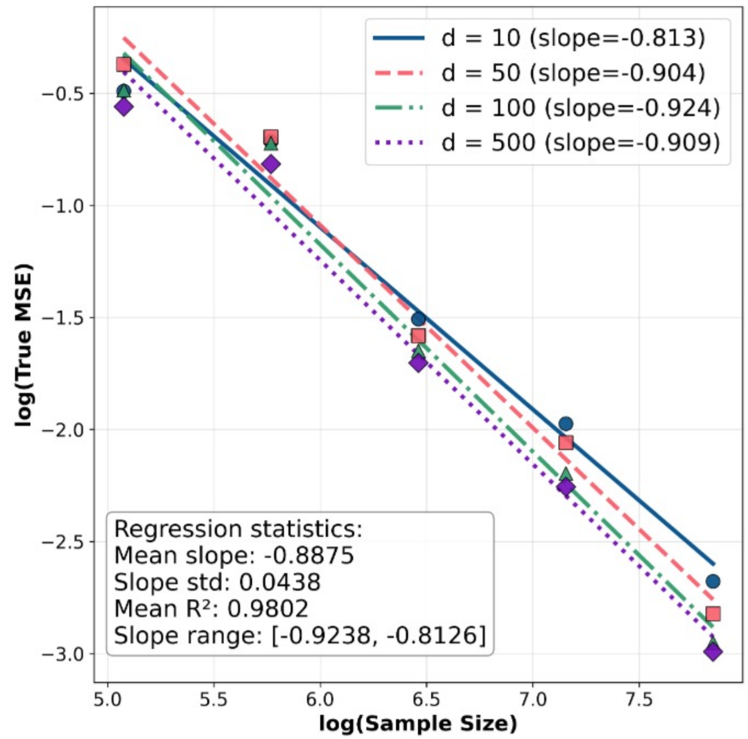


Percentage of training data where a neuron is active

What happens if the input data is secretly low-dimensional (embedded in a high-dim ambient space)

- Assumption: data comes from a union of low-dim subspaces

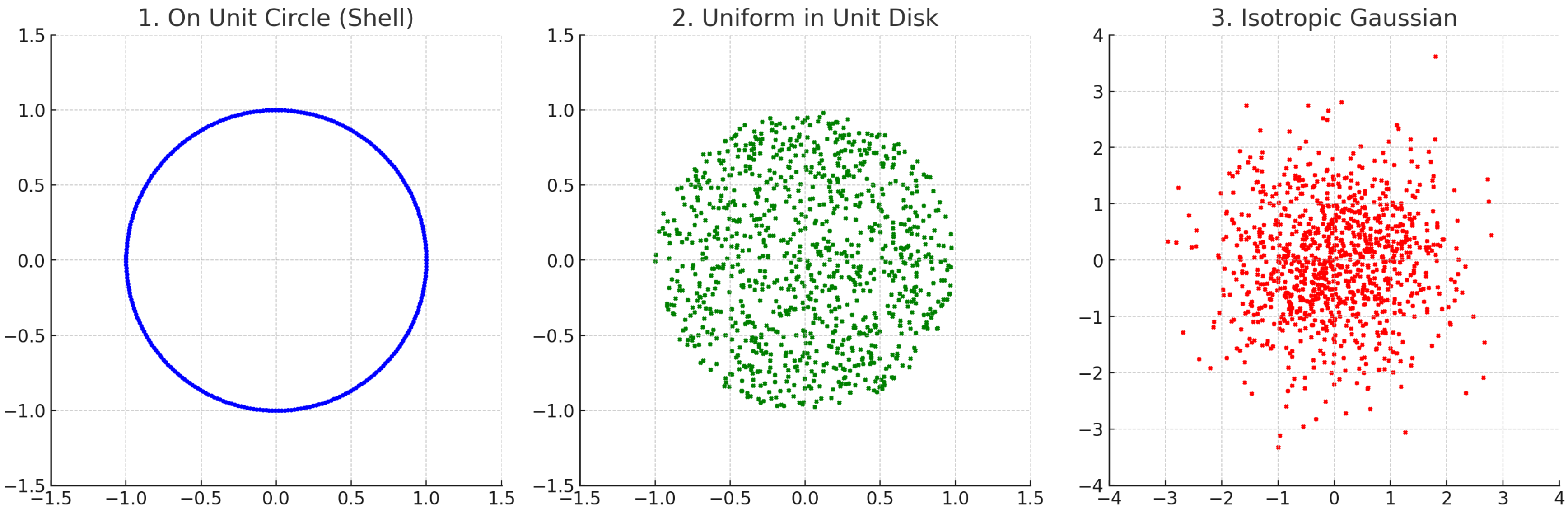




(a) Adaptation to intrinsic dimension

Liang et al (2025) **Generalization Below the Edge of Stability: The Role of Data Geometry.** https://arxiv.org/abs/2510.18120

The shape of data distribution matters in flatness induced generalization



Cannot generalize at all

Generalize but suffer from Curse-of-Dimensionality

Liang et al (2025) **Generalization Below the Edge of Stability: The Role of Data Geometry.**
https://arxiv.org/abs/2510.18120

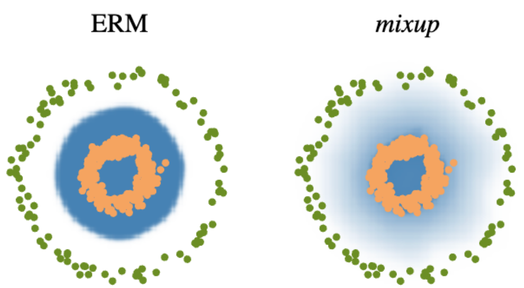
Mixup: a prominent approach for data augmentation.

mixup: BEYOND EMPIRICAL RISK MINIMIZATION

by H Zhang · 2017 · Cited by 13963 — We have proposed **mixup, a data-agnostic and straightforward data augmentation principle**. We have shown that mixup is a form of vicinal risk ...

```python
# y1, y2 should be one-hot vectors
for (x1, y1), (x2, y2) in zip(loader1, loader2):
    lam = numpy.random.beta(alpha, alpha)
    x = Variable(lam * x1 + (1. - lam) * x2)
    y = Variable(lam * y1 + (1. - lam) * y2)
    optimizer.zero_grad()
    loss(net(x), y).backward()
    optimizer.step()
```

(a) One epoch of *mixup* training in PyTorch.



(b) Effect of *mixup* ($\alpha = 1$) on a toy problem. Green: Class 0. Orange: Class 1. Blue shading indicates $p(y = 1|x)$.

Figure 1: Illustration of *mixup*, which converges to ERM as $\alpha \to 0$.

Our theory explains “mixup" quite well. But can we do better?

Checkpoint: provable generalization bounds for low-curvature points, but..

- Trickier in high-dimension and beyond square loss.
- Known fixes: Data-augmentation, Weight Decay, Architecture tweaks.
- Many interesting theoretical / empirical directions to explore.

Remainder of this tutorial

1. Flat minima **exactly recover** weights in Matrix Sensing and 2-layer Neural Nets (Maryam)

2. Does **flatness imply generalization** in 2-layer ReLU Neural Networks? (Yu-Xiang)

3. Discussion and Open problems. (Both)

Flat minima / regions in **Multi-layer** Neural networks appears to behave qualitatively different.

- For two-layers networks:
  - Mostly similar to weight decay, give L1-type sparsity (or low nuclear norm)

- For L-layer diagonal linear networks
  - As L → large, weight decay => ||. ||_2/L norm. (sparser!)

  - But flat minima => ||. ||_$\{2-\frac{2}{L}\}$ norm (denser!)

(Lemma 9.2, Ding et al., 2024)

What do we know and what's open?

- L-layer linear (non-diagonal) neural networks
  (Gatmiry et al, NeurIPS'22).
  similar to when L=2, i.e., nuclear-norm.

- What happens with nonlinear activations?

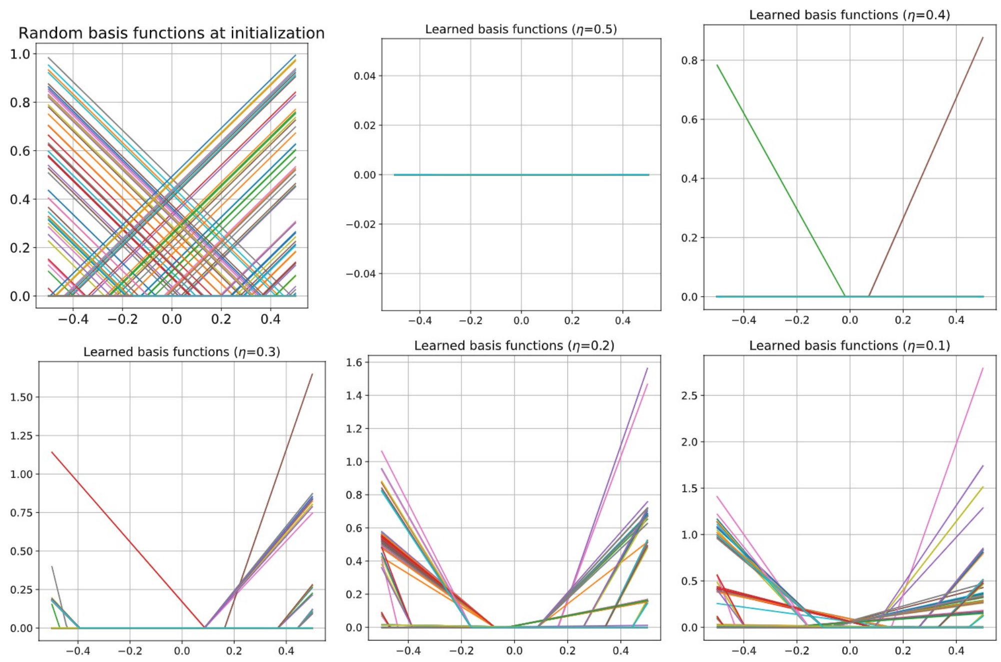
- In between diagonal vs fully-connected weights?
  - Convolutional layers?
  - Block-diagonal weights?

Interaction with architecture choices.

- BERT models have biases
- GPT models do not use biases
- Provably better generalization when there is no bias?

The modality of representation learning is quite interesting

- It's pushing neurons out of data support.
- "Dead" neurons will never recover.
- They may be active on OOD data.
  - Culprits of non-robustness



How can we characterize the dynamics?

Thank you for your attention!

Jingfeng Wu
UC Berkeley

Yu-Xiang Wang
UC San Diego

Maryam Fazel
UW

References and other materials on the website:
https://uuujf.github.io/instability/

Supplementary slides

What about depth?

Overparameterized sparse recovery:

$$\min_{v_1,\ldots,v_k \in \mathbb{R}^d} \quad f(v) := \frac{1}{m}\|A(\underbrace{v_1 \odot \cdots \odot v_k}_{x}) - b\|_2^2,$$

where $b = A(x_\sharp)$ and we seek x that's $r_\sharp$-sparse.

Flat $(v_1, \ldots, v_k)$ are those solving:

$$\min_{v_i \in \mathbb{R}^d, i=1,\ldots,k} \mathrm{tr}(D^2 f(v_1, \ldots, v_k)) \quad \text{s.t.} \quad A(v_1 \odot \cdots \odot v_k) = b.$$

Lemma: For Gaussian A, any flat solution $(v_1, \ldots, v_k)$ yields a minimizer $x = v_1 \odot \cdots \odot v_k$ of the problem:

$$\min_{x \in \mathbb{R}^d} \sum_{i=1}^{d} |D_{ii}||x_i|^{2-\frac{2}{k}} \quad \text{s.t.} \quad Ax = b.$$

Conclusion: Exact recovery for $k = 2$ and poor recovery as $k \to \infty$.

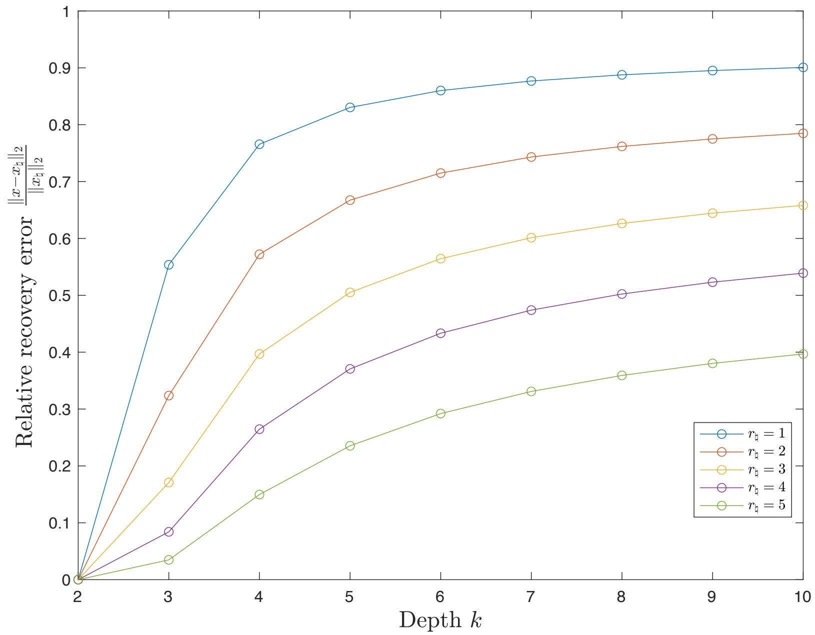


Figure: The effect of depth for different choice of sparsity $r_\sharp$

- (Gatmiry et al. Neurips'23) showed approximate recovery bounds for k-layer but **non-diagnonal** linear network
- Theoretical explanation is still open for $k > 2$ for networks with nonlinear activation