Understanding Neural Nets as Splines:

Theory, Experiment, and Applications to Neuroscience

Ankit B. Patel

Joint work with Justin Sahs, Ryan Pyle, Aneel Damaraju, Josue Caro, Andy Lu, & others from Patel Lab

Baylor College of Medicine (Neuroscience Dept.)

Rice University (ECE + CS (Adj) Dept.)

ELEC/COMP 576 Lecture

Nov 4, 2020

Fabio Anselmi

Outline

- **Problem:** Many perplexing phenomena in NNs lack clear theoretical explanations:
  - Try to understand phenomena with artificial NNs
  - This leads to a new theory for understanding the representation/ learning of NNs
    - ***"NNs as Splines", "NNs as Continuous Basis Expansions"***
  - Then circle back to neuronal networks with Implications/Applications to Theoretical & Computational Neuroscience

Issues in the Learning Theory of Deep Neural Networks

Unexplained Phenomena

- **Problem:** Many perplexing phenomena lack theoretical explanations:
  - *OverParametrization:* The need for Overparametrization for training not expressivity
  - *Inits:* Lottery Ticket Hypothesis
  - *Loss Surface:* Highly non-convex and yet so many local minima near global minima
  - *Generalization:* Implicit Regularization despite highly underdetermined optimization problem
- **Potential Solution:** Pass to function space parametrization (spline)

Related Work

- **Du et al, Arora et al (2018-):** Circumvent dynamics of weights and instead track dynamics of predictions (i.e. the approx. function)

- **Neyshabur, Serebro et al (2015-).: Implicit regularization** in highly underdetermined optimization problems in ML due to parametrization and/or optimization algorithm. Focus on **low *weight* norm** solutions… recent ICLR 2020 paper characterizing class of low-weight norm ReLu NNs

- **Baraniuk et al (2018-19):** the **Spline** Perspective and the Geometry of deep nets

- **Neural Tangent Kernel:** *Jacot et al NeurIPS 2018*, NTK valid in *massively* overparametrized regime, reduces to linear dynamics

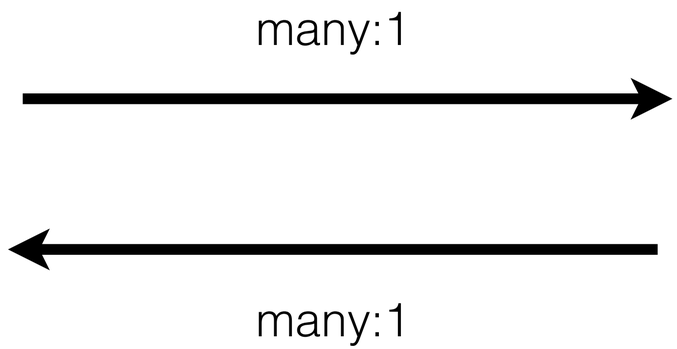
- **Double Descent Curve:** *Belkin et al 2019* show when traditional U-shaped generalization curve is replaced by a new doubled curve

- **Williams et al (2019):** Similar results re Implicit Reg in Shallow Univariate ReLu NNs; but different abstruse parametrization and no results re the initialization and Hessian of loss surface.

To Function Space: Reparametrizing Neural Nets as Splines

Motivating Function Space

Neural Net Params
(Weights & Biases)



Spline Params
(Breakpoints, Curvatures & Orientations)

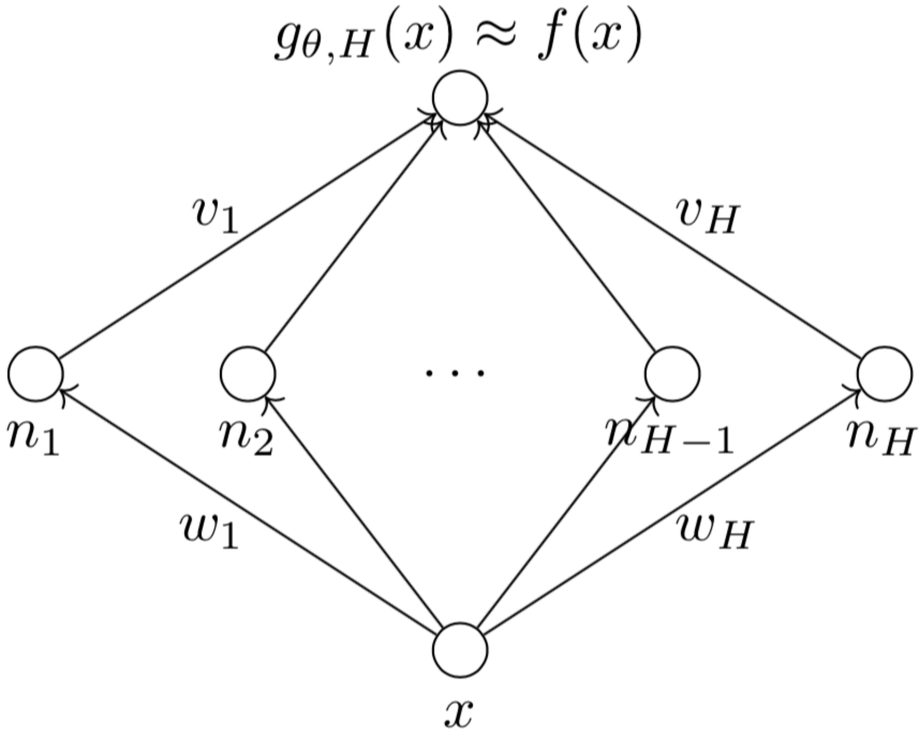
- “unfunctional” dof
- Symmetry Group(“gauge”) is large

- Mod out “gauge” group or fix gauge dof
- Every dof matters for the function / loss

What is the role of individual neurons?
A simple neural net example

We'll focus on Univariate Shallow/Deep ReLu NNs:

Shallow Univariate ReLu NN

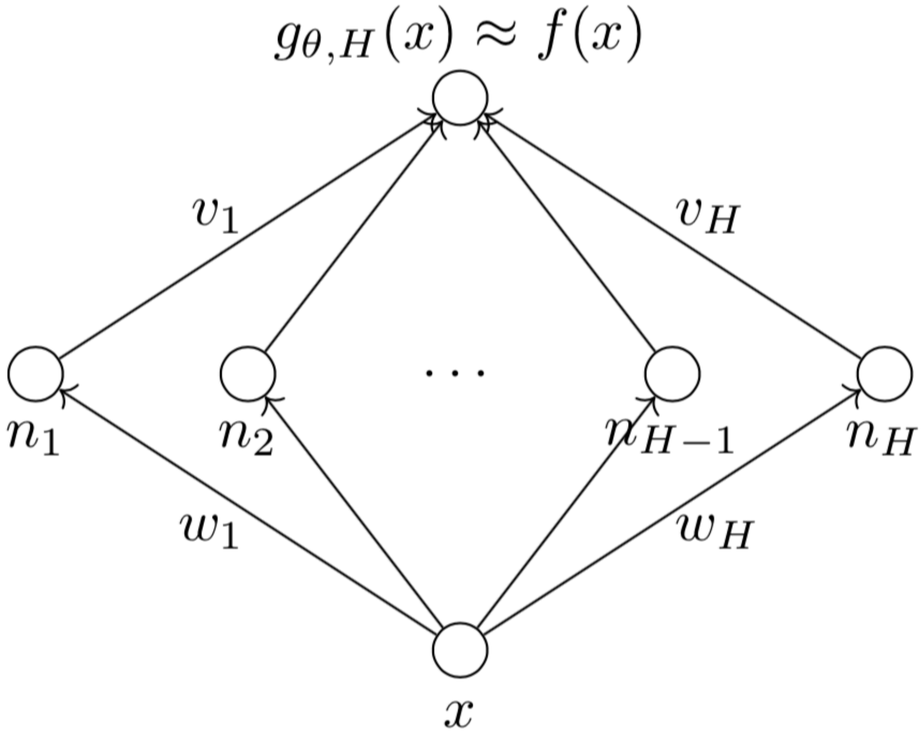


$$\hat{f}(x;\theta_{NN}) \triangleq \sum_{i=1}^{H} v_i(w_i x + b_i)_+$$

What is the role of individual neurons?
A simple neural net example

We'll focus on Univariate Shallow/Deep ReLu NNs:

Shallow Univariate ReLu NN



$$\hat{f}(x;\theta_{NN}) \triangleq \sum_{i=1}^{H} v_i(w_i x + b_i)_+$$

$$= \sum_{i=1}^{H} \mu_i(x-\beta_i) \begin{cases} [\![x > \beta_i]\!], & s_i = 1 \\ [\![x < \beta_i]\!], & s_i = -1 \end{cases}$$

$$= \sum_{p=1}^{P} [\![\beta_p \le x < \beta_{p+1}]\!] \, (m_p x + \gamma_p) \triangleq \hat{f}(x;\theta_{PWL})$$

ReLu Neural Nets as Continuous PieceWise Linear (CPWL) Functions

Reparametrization from NN to BDSO:

$$\hat{f}(x;\theta_{NN}) \triangleq \sum_{i=1}^{H} v_i(w_i x + b_i)_+$$

$$= \sum_{i=1}^{H} \mu_i(x-\beta_i) \begin{cases} [\![x > \beta_i]\!], & s_i = 1 \\ [\![x < \beta_i]\!], & s_i = -1 \end{cases}$$

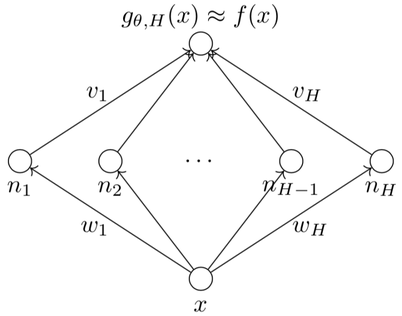
$$= \sum_{p=1}^{P} [\![\beta_p \leq x < \beta_{p+1}]\!] \, (m_p x + \gamma_p) \triangleq \hat{f}(x;\theta_{PWL})$$

Breakpoint

Orientation

Delta-Slope ("Curvature")

Shallow Univariate ReLu NN



Recasting the Simple ReLu Neural Net as a Continuous Piece-Wise Linear (CPWL) Function

We'll focus on Univariate Shallow/Deep ReLu NNs:

$$\hat{f}(x;\theta_{NN}) \triangleq \sum_{i=1}^{H} v_i(w_i x + b_i)_+$$

$$= \sum_{i=1}^{H} \mu_i(x-\beta_i) \begin{cases} [\![x > \beta_i]\!], & s_i = 1 \\ [\![x < \beta_i]\!], & s_i = -1 \end{cases}$$

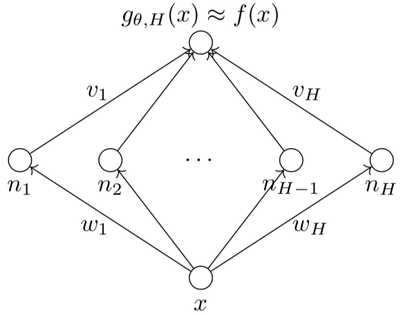
$$= \sum_{p=1}^{P} [\![\beta_p \leq x < \beta_{p+1}]\!] (m_p x + \gamma_p) \triangleq \hat{f}(x;\theta_{PWL})$$

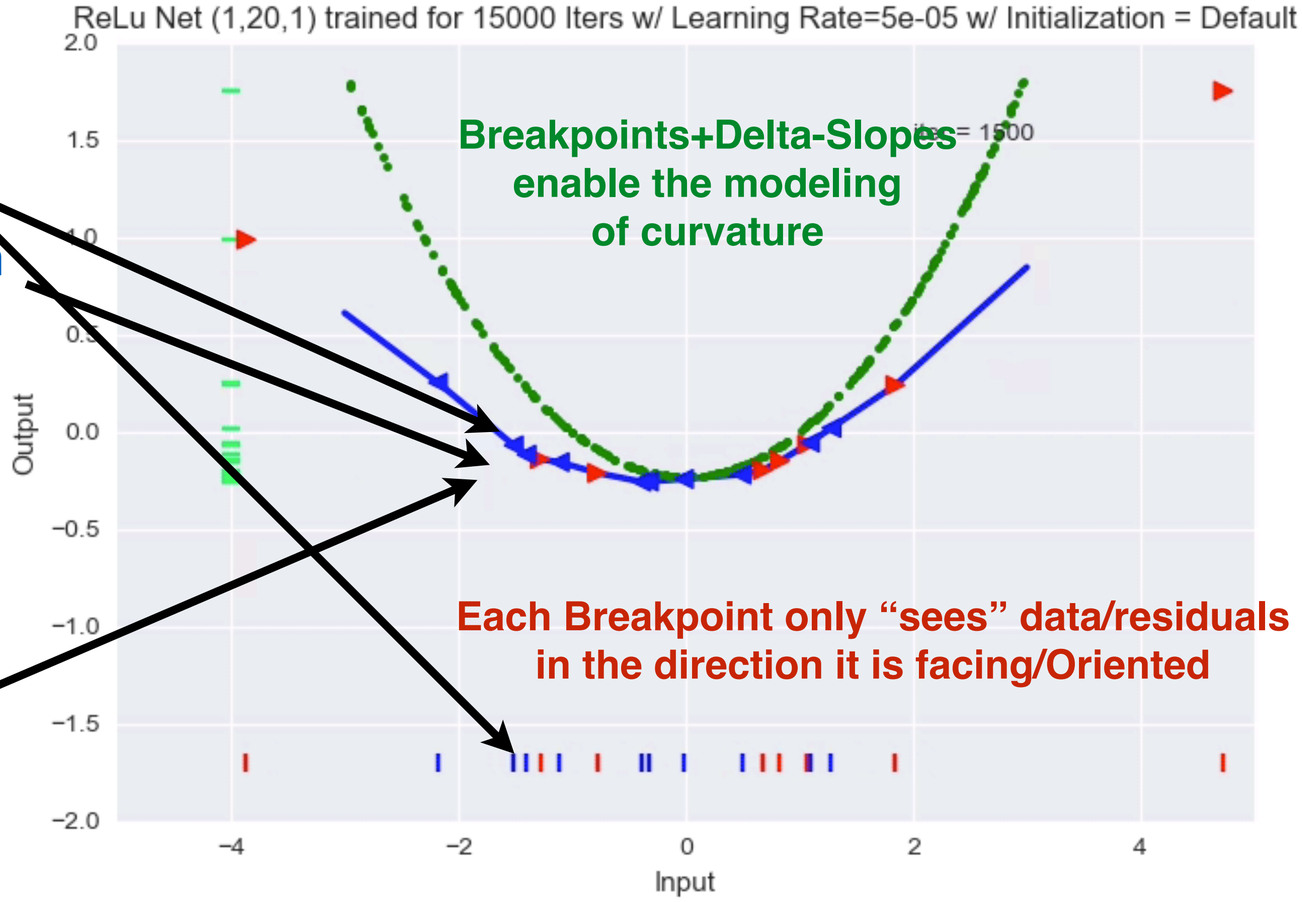
Breakpoint

Orientation (right/left)

Delta-Slope ("Curvature")

Shallow Univariate ReLu NN





Breakpoints+Delta-Slopes enable the modeling of curvature

Each Breakpoint only "sees" data/residuals in the direction it is facing/Oriented

Ongoing Work:
Generalizing ReLU NN to Multivariate Inputs

Multivariate Spline Parametrization of a ReLU NN

$$\hat{f}_H(\mathbf{x};\theta_{\text{NN}}) = \frac{1}{\sqrt{H}}\sum_{i=1}^{H} v_i\left(\langle \mathbf{w}_i, \mathbf{x}\rangle + b_i\right)_+$$

$$= \frac{1}{\sqrt{H}}\sum_{i=1}^{H} v_i\|\mathbf{w}_i\|_2\left(\left\langle \frac{\mathbf{w}_i}{\|\mathbf{w}_i\|_2}, \mathbf{x}\right\rangle + \frac{b_i}{\|\mathbf{w}_i\|_2}\right)_+$$

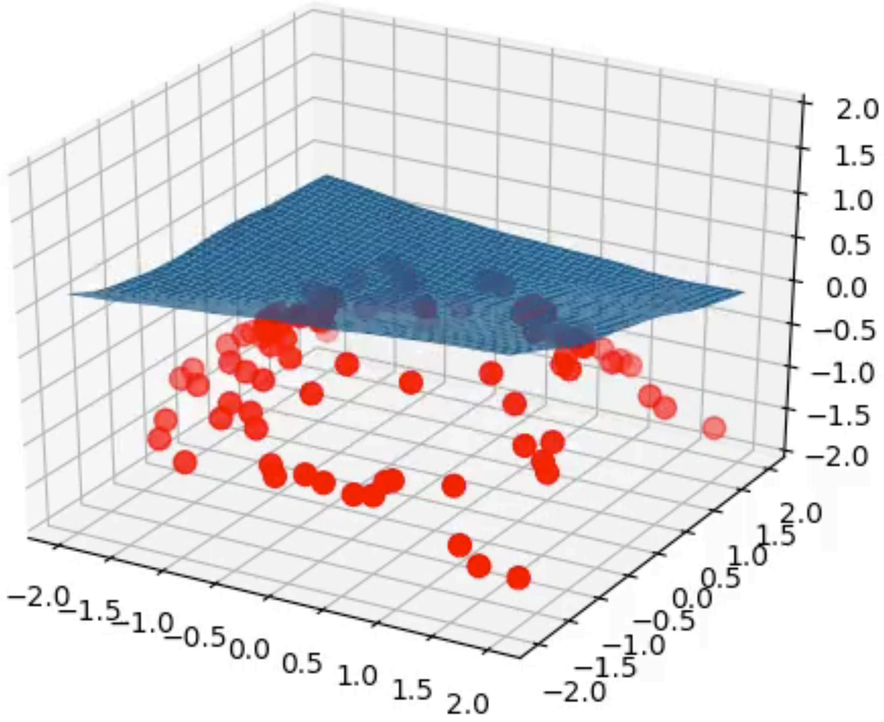
$$\hat{f}_H(\mathbf{x};\theta_{\text{BDSO}}) \triangleq \frac{1}{\sqrt{H}}\sum_{i=1}^{H} \mu_i(\langle \boldsymbol{\xi}_i, \mathbf{x}\rangle - \gamma_i)_+$$

$\mu_i \triangleq$ delta-slope $v_i\|\mathbf{w}_i\|_2$

$\boldsymbol{\xi}_i \triangleq$ unit vector normal to the break-plane $\dfrac{\mathbf{w}_i}{\|\mathbf{w}_i\|_2}$

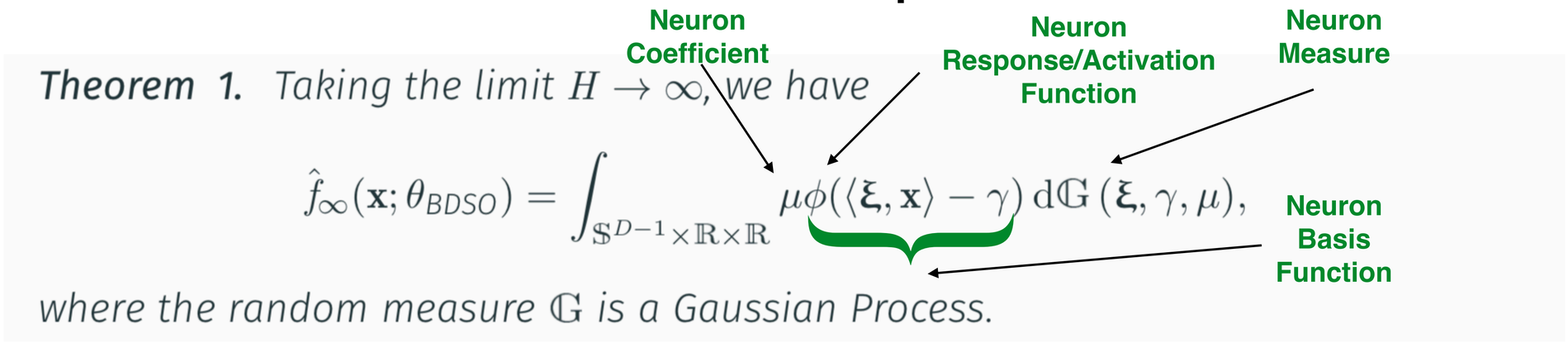
$\gamma_i \triangleq$ offset from origin $\dfrac{-b_i}{\|\mathbf{w}_i\|_2}$

2-Dimensional Inputs with Data Gap



Ongoing Work:
Generalizing to Multivariate Inputs & Arbitrary Activation Functions

Representation of NN: Approximating a Function via A Continuous Basis Expansion



Theorem 1. *Taking the limit $H \to \infty$, we have*

$$\hat{f}_\infty(\mathbf{x}; \theta_{BDSO}) = \int_{\mathbb{S}^{D-1}\times\mathbb{R}\times\mathbb{R}} \mu\phi(\langle \boldsymbol{\xi}, \mathbf{x}\rangle - \gamma)\, \mathrm{d}\mathbb{G}\,(\boldsymbol{\xi}, \gamma, \mu),$$

where the random measure $\mathbb{G}$ is a Gaussian Process.

Implication: The (well-developed) tools & mathematics of function space and basis expansions can be brought to bear on NN problems.

I'll talk about this more later if there's time and interest…

Random Initializations

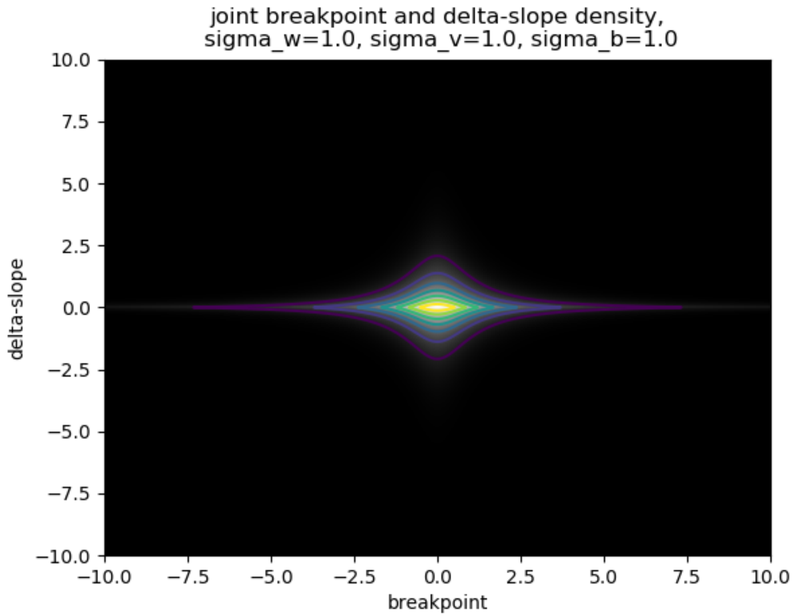
Joint + Conditional Density b/w Breakpoints and Delta-Slopes

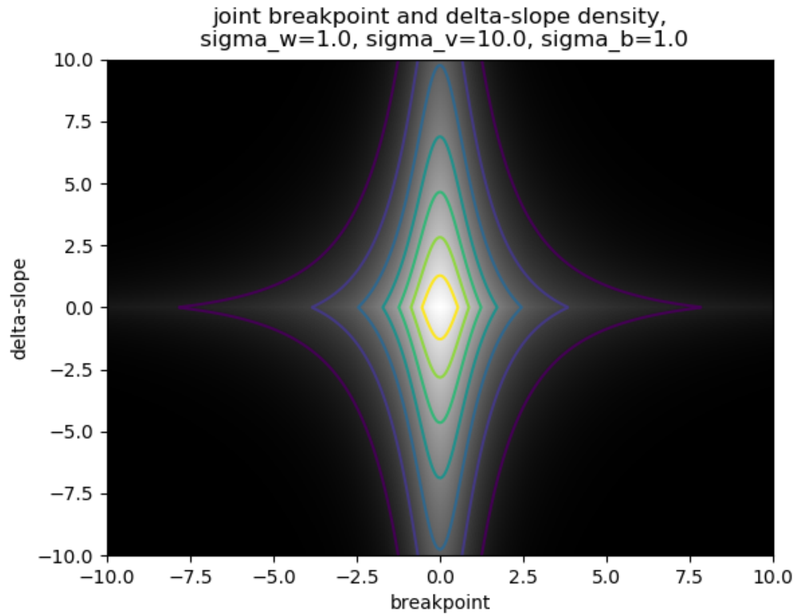
$$p_{\beta,\mu}(\beta_i, \mu_i) = \frac{1}{2\pi\sigma_v\sqrt{\sigma_b^2 + \sigma_w^2\beta_i^2}} \exp\left[-\frac{|\mu_i|\sqrt{\sigma_b^2 + \sigma_w^2\beta_i^2}}{\sigma_b\sigma_v\sigma_w}\right]$$

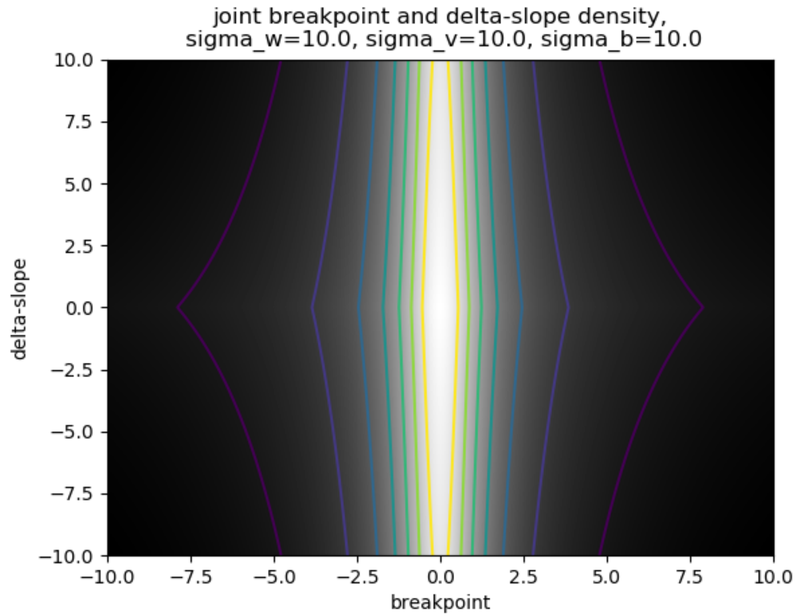
In the case of an independent Gaussian initialization,

$$p_{\mu|\beta}(\mu_i|\beta_i) = \textit{Laplace}\left(\mu_i; 0, \frac{\sigma_b\sigma_v\sigma_w}{\sqrt{\sigma_b^2 + \sigma_w^2\beta_i^2}}\right) = \frac{\sqrt{\sigma_b^2 + \sigma_w^2\beta_i^2}}{2\sigma_b\sigma_v\sigma_w} \exp\left[-\frac{|\mu_i|\sqrt{\sigma_b^2 + \sigma_w^2\beta_i^2}}{\sigma_b\sigma_v\sigma_w}\right]$$

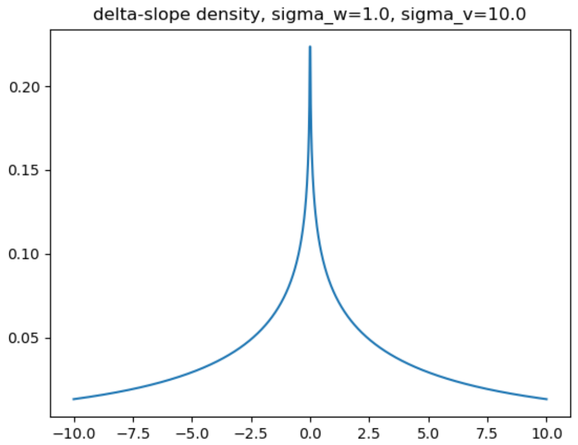
Non-obvious correlation b/w Breakpoint location and curvature







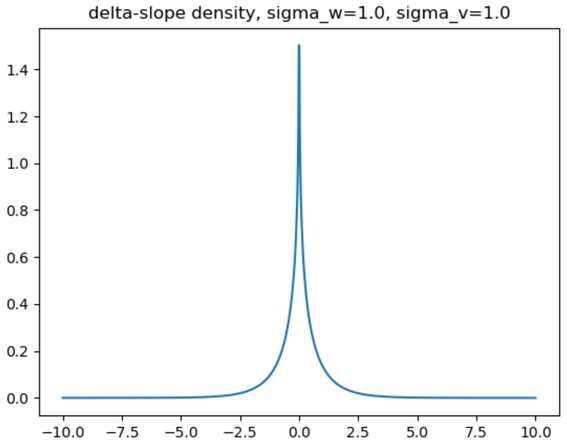
Marginal Density of Delta-Slopes

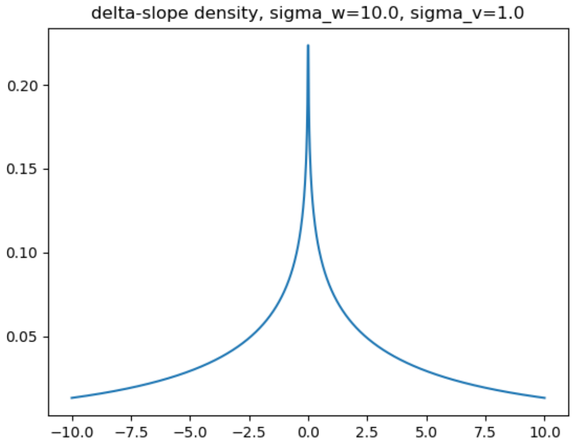


In the case of an independent Gaussian initialization,

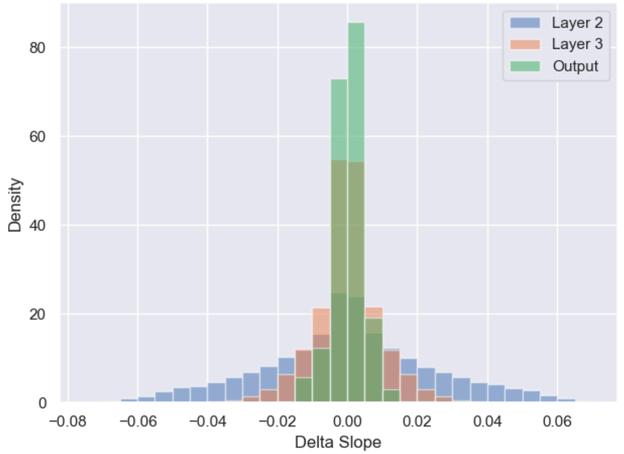
$$p_\mu(\mu_i) = \frac{1}{2\pi\sigma_v\sigma_w} G_{0,2}^{2,0}\left(\frac{\mu_i^2}{4\sigma_v\sigma_w}\middle|0,0\right) = \frac{1}{\pi\sigma_v\sigma_w} K_0\left(\frac{|\mu_i|}{\sigma_v\sigma_w}\right)$$

where $G_{pq}^{nm}(\cdot|\cdot)$ is the Meijer G-function and $K_\nu(\cdot)$ is the modified Bessel function of the second kind.



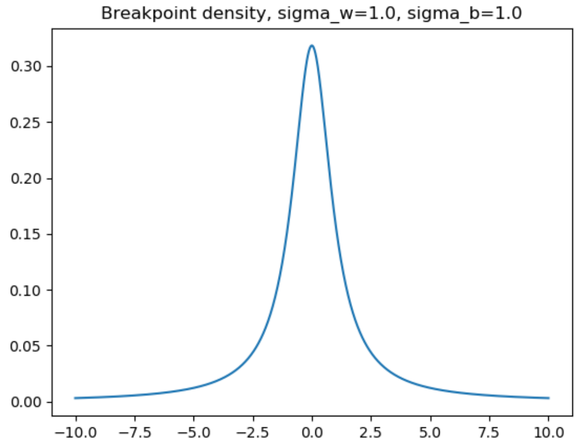


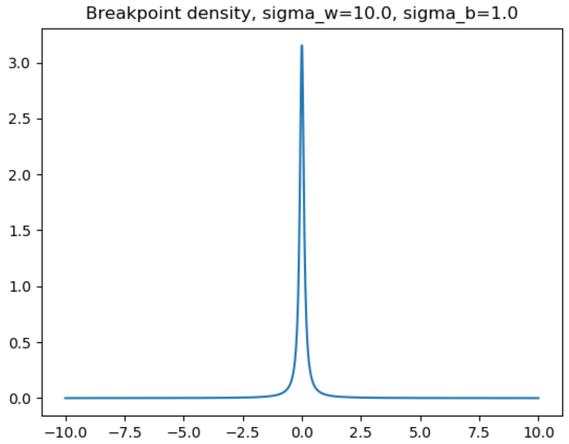
Empirical Observations

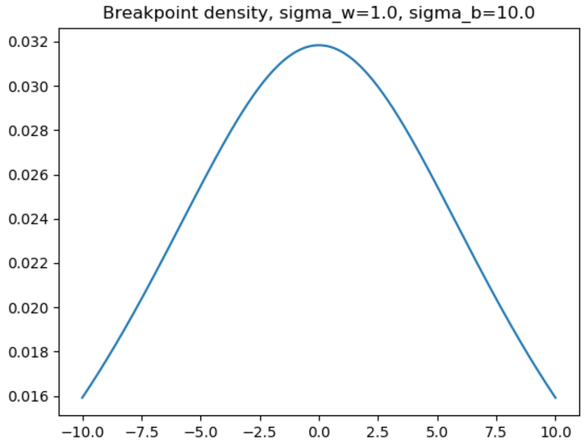


Depth yields delta-slopes closer to zero

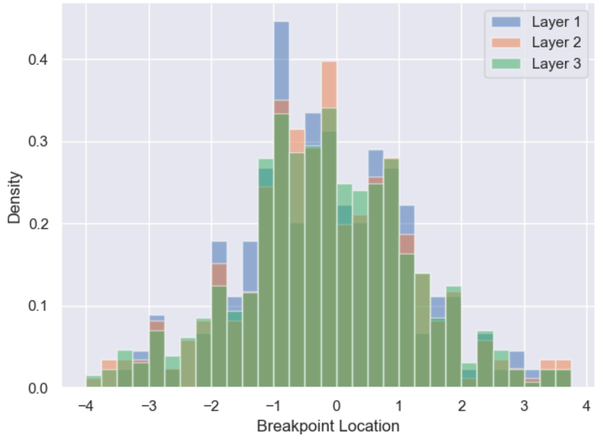
Marginal Density of Breakpoints







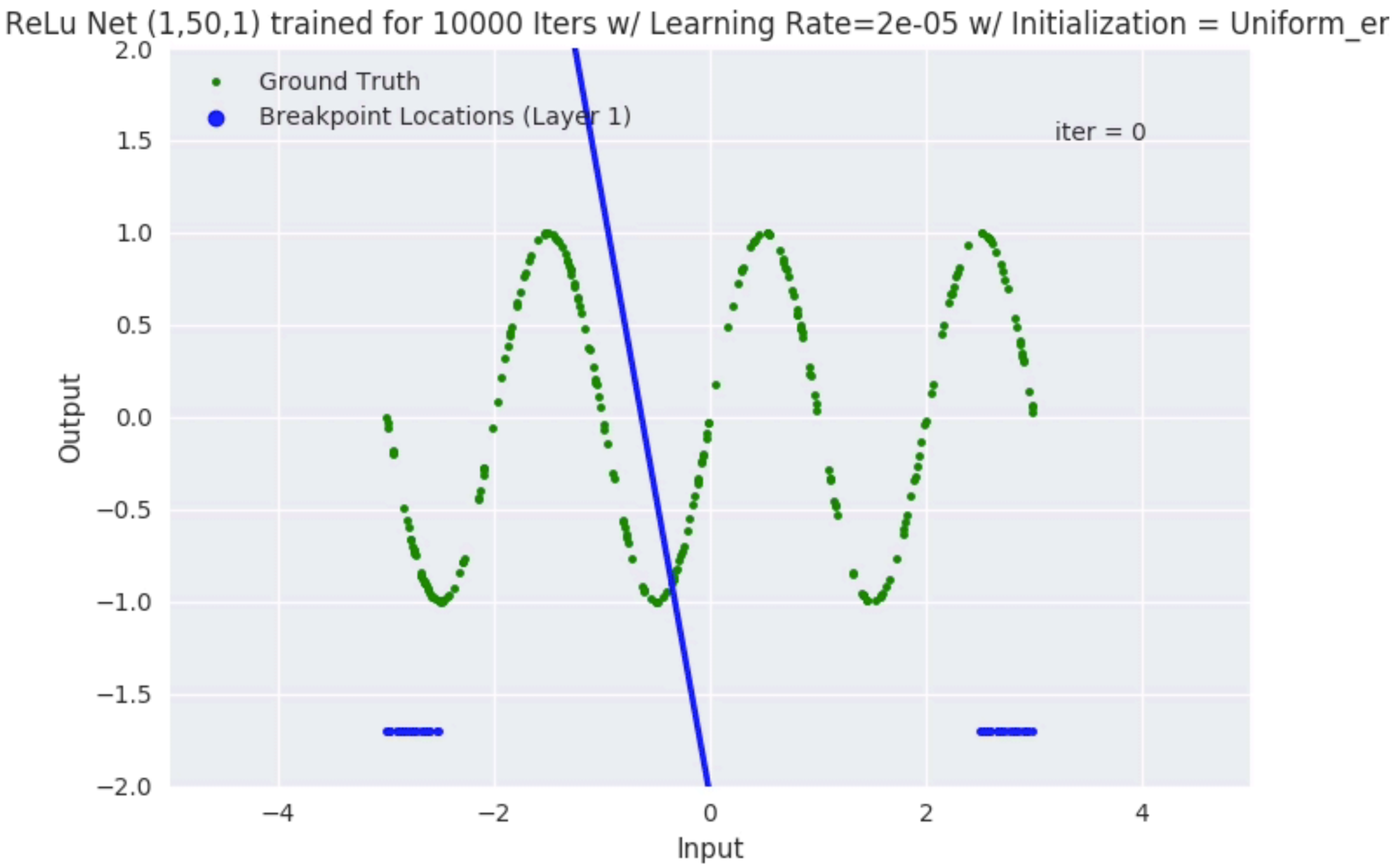
Empirical Observations



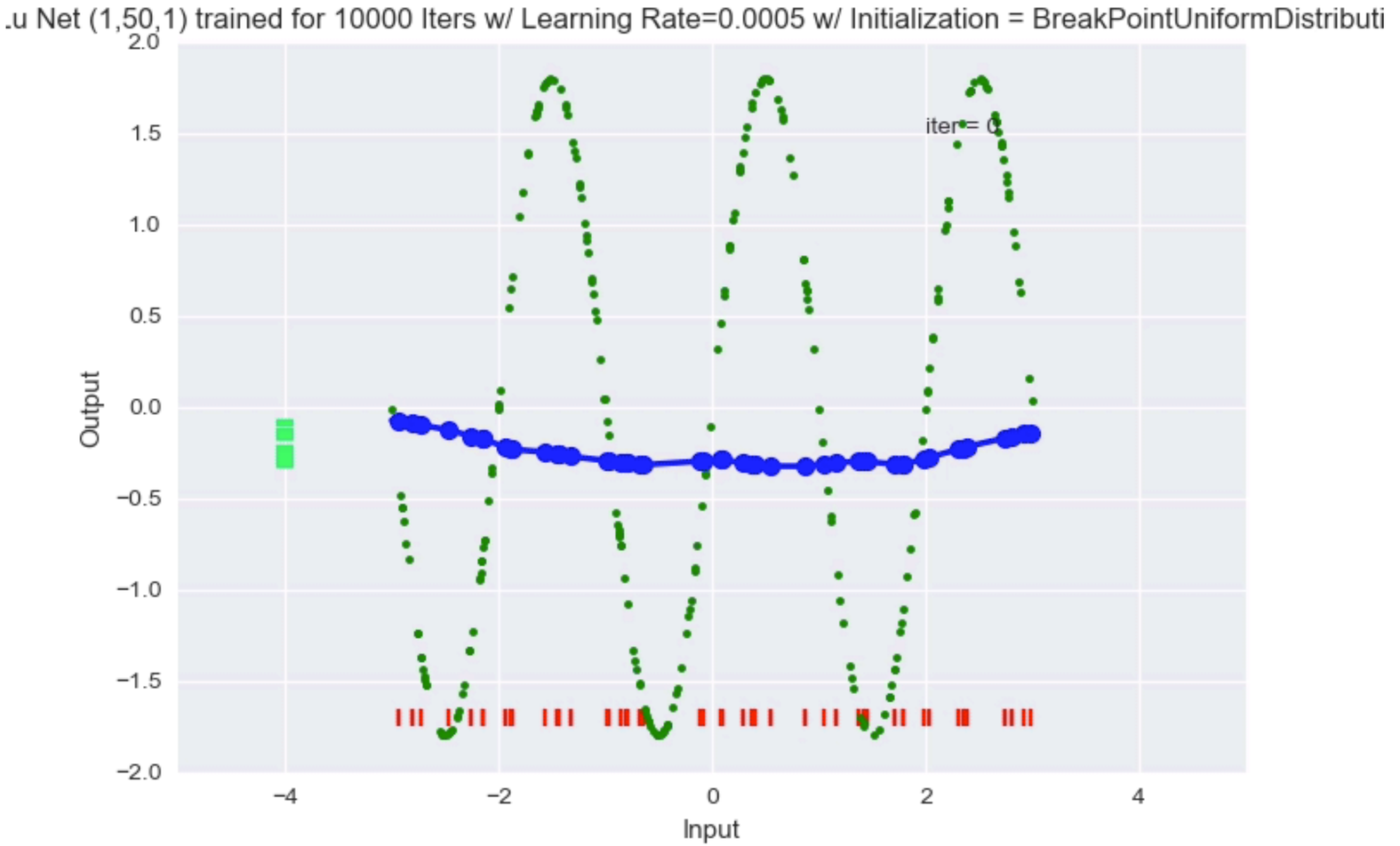
In the case of an independent Gaussian initialization,

$$p_\beta(\beta_i) = Cauchy\left(\beta_i; 0, \frac{\sigma_b}{\sigma_w}\right) = \frac{\sigma_b \sigma_w}{\pi\left(\sigma_w^2 \beta_i^2 + \sigma_b^2\right)}$$

Breaking Bad: Mismatch b/w Breakpoints and Function Curvature leads to poor optimization with GD…



.. but this can be Rescued by a Better Data-dependent Initialization of the Breakpoints



Data-Dependent Init:

Exploit Reparam. to Shape Breakpoint Density

$$\hat{f}(x;\theta_{NN}) \triangleq \sum_{i=1}^{H} v_i(w_i x + b_i)_+$$

$$= \sum_{i=1}^{H} \mu_i(x-\beta_i) \begin{cases} [\![x > \beta_i]\!], & s_i = 1 \\ [\![x < \beta_i]\!], & s_i = -1 \end{cases}$$

where $\beta_i \triangleq -\frac{b_i}{w_i}$

Init	Sine	Quadratic
Standard	4.096 ± 2.25	$.1032 \pm 0404$
Uniform	$2.280 \pm .457$	$.1118 \pm .0248$

Table 1: Test loss for standard vs uniform breakpoint initialization, on sine and quadratic $\frac{x^2}{2}$

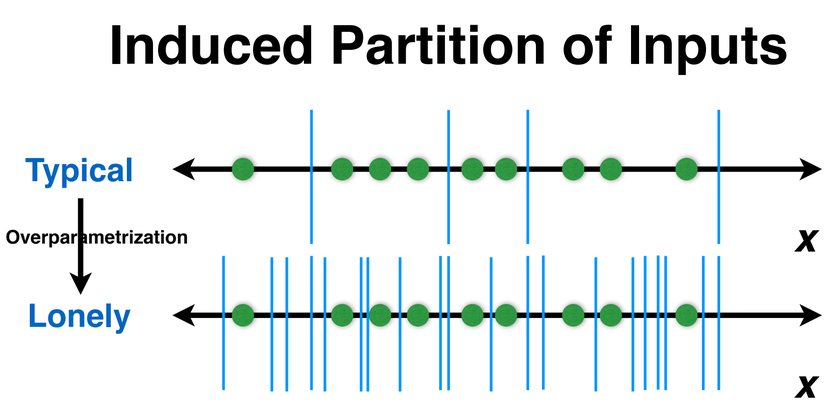
Loss Surface

Degeneracies in Loss Surface

Theorem 3. *θ^*_{BDSO} is a critical point of $\tilde{\ell}(\theta_{BDSO})$ if for all pieces $p \in [P]$ we have that $\hat{f}(\cdot;\theta_{BDSO})|_{\pi_p}$ is an Ordinary Least Squares fit of the data in piece p, and we refer to the critical point as a (C)PWL-OLS solution. Furthermore every critical point θ^*_{BDSO} of $\tilde{\ell}(\theta_{BDSO})$ corresponds to an equivalence class of critical points $\theta^*_{NN} \in [\mathcal{G}\theta^*_{BDSO}]$ of $\ell(\theta_{NN})$ where $\mathcal{G}$ is the set of transformations on the NN parameters that leaves the function (BDSO parameters) invariant.*

Overparametrization ==> Lonely Partitions ==> Global Minima

Theorem 4. *Consider the partition $\Pi_{N,H}$ as defined above. A partition is lonely if each datapoint n is alone in its own piece p. (a) The PWL OLS solution for a lonely partition is (i) CPWL, (ii) a local minima and (iii) a global minima of $\tilde{\ell}$. (b) Furthermore, if we assume that H breakpoints are uniformly spaced and that N data points are uniformly distributed within the region of breakpoints, then in the overparametrized regime $H \geq \alpha N^2$ for some constant $\alpha > 1$, the induced partition $\Pi_{N,H}$ is lonely with high probabilility $1 - e^{-N^2/(H+1)} = 1 - e^{-1/\alpha}$. Furthermore, the total number of lonely partitions, and thus (a lower bound on) the total number of global minima of $\tilde{\ell}$ is $\binom{H+1}{N} = O(N^{\alpha N})$*



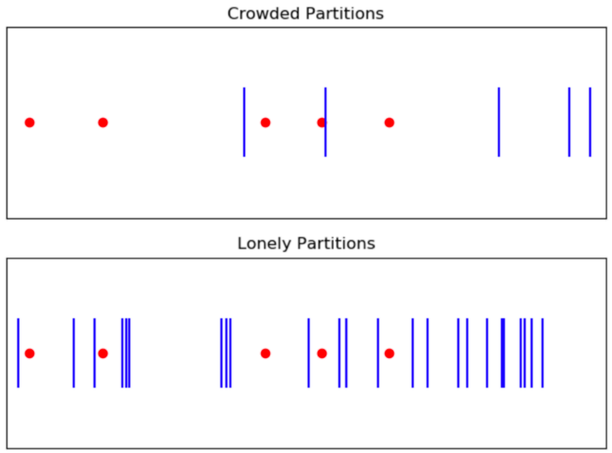


Figure 1. Breakpoints (blue bars) vs datapoints (red points). A lonely partition is when a datapoint is isolated - overparameterization makes this increasingly likely

Overparametrization ==> Lonely Partitions ==> Global Minima

Theorem 4. *Consider the partition $\Pi_{N,H}$ as defined above. A partition is lonely if each datapoint n is alone in its own piece p. (a) The PWL OLS solution for a lonely partition is (i) CPWL, (ii) a local minima and (iii) a global minima of $\tilde{\ell}$. (b) Furthermore, if we assume that H breakpoints are uniformly spaced and that N data points are uniformly distributed within the region of breakpoints, then in the overparametrized regime $H \geq \alpha N^2$ for some constant $\alpha > 1$, the induced partition $\Pi_{N,H}$ is lonely with high probabilility $1 - e^{-N^2/(H+1)} = 1 - e^{-1/\alpha}$. Furthermore, the total number of lonely partitions, and thus (a lower bound on) the total number of global minima of $\tilde{\ell}$ is $\binom{H+1}{N} = O(N^{\alpha N})$*

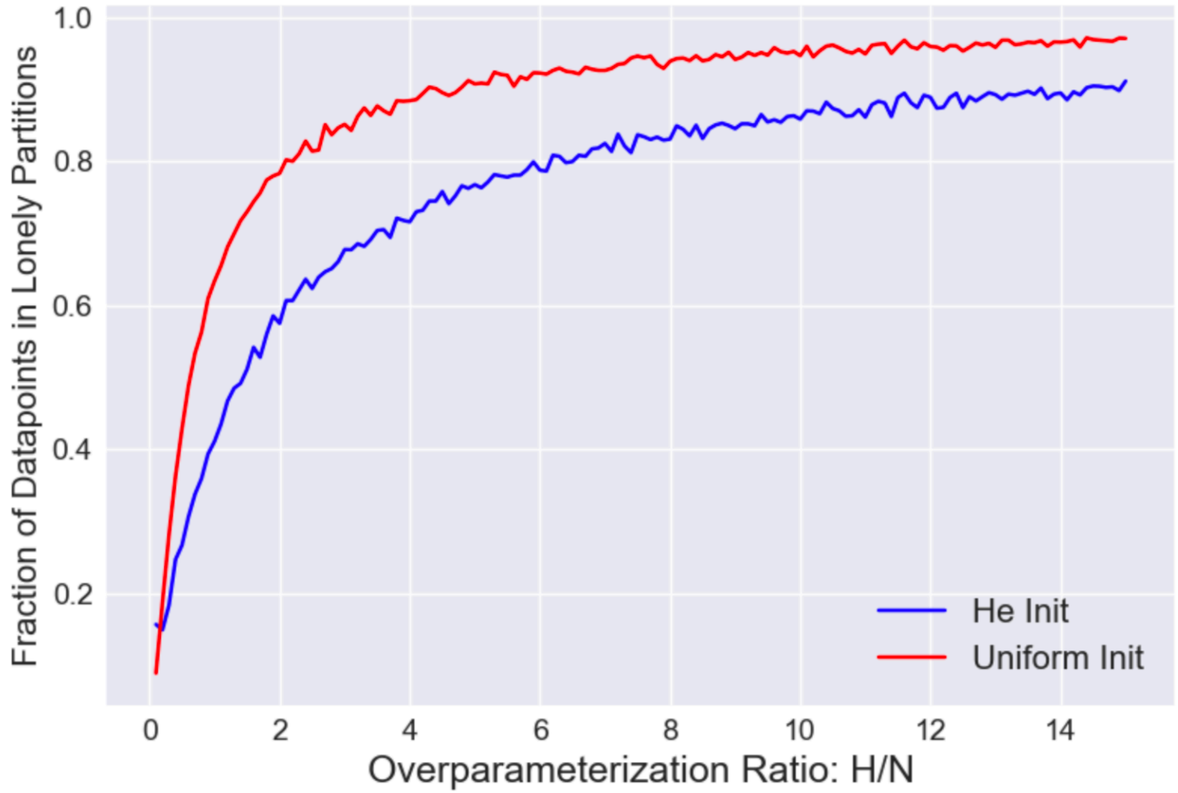


Figure 11. Percentage of datapoints which are in a lonely partition as a function of overparameterization ratio $\frac{H}{N}$ for both a standard (H
and uniform init. Massive overparameterization leads w.h.p. to lonely partitions.

Hessian of the Loss and Degenerate Directions

If Two Neurons have the same Activation Pattern, Then there will be a Zero Eigenvalue

Simplified Hessian is a Gram Matrix —> Pos.Semi-Def….

$$= \begin{pmatrix}
\langle v_i\mathbf{x}_i, v_j\mathbf{x}_j\rangle & \langle w_i\mathbf{x}_i + b_i\mathbf{1}_i, v_j\mathbf{x}_j\rangle & \langle v_i\mathbf{1}_i, v_j\mathbf{x}_j\rangle \cdots \langle \mathbf{1}, v_j\mathbf{x}_j\rangle \\
\langle v_i\mathbf{x}_i, w_j\mathbf{x}_j + b_j\mathbf{1}_j\rangle & \langle w_i\mathbf{x}_i + b_i\mathbf{1}_i, w_j\mathbf{x}_j + b_j\mathbf{1}_j\rangle & \langle v_i\mathbf{1}_i, w_j\mathbf{x}_j + b_j\mathbf{1}_j\rangle \cdots \langle \mathbf{1}, w_j\mathbf{x}_j + b_j\mathbf{1}_j\rangle \\
\langle v_i\mathbf{x}_i, v_j\mathbf{1}_j\rangle & \langle w_i\mathbf{x}_i + b_i\mathbf{1}_i, v_j\mathbf{1}_j\rangle & \langle v_i\mathbf{1}_i, v_j\mathbf{1}_j\rangle \cdots \langle \mathbf{1}, v_j\mathbf{1}_j\rangle \\
\vdots & & \vdots \\
\langle v_i\mathbf{x}_i, v_i\mathbf{x}_i\rangle & \langle w_i\mathbf{x}_i + b_i\mathbf{1}_i, v_i\mathbf{x}_i\rangle & \langle v_i\mathbf{1}_i, v_i\mathbf{x}_i\rangle \\
\langle v_i\mathbf{x}_i, w_i\mathbf{x}_i + b_i\mathbf{1}_i\rangle & \langle w_i\mathbf{x}_i + b_i\mathbf{1}_i, w_i\mathbf{x}_i + b_i\mathbf{1}_i\rangle & \langle v_i\mathbf{1}_i, w_i\mathbf{x}_i + b_i\mathbf{1}_i\rangle \\
\langle v_i\mathbf{x}_i, v_i\mathbf{1}_i\rangle & \langle w_i\mathbf{x}_i + b_i\mathbf{1}_i, v_i\mathbf{1}_i\rangle & \langle v_i\mathbf{1}_i, v_i\mathbf{1}_i\rangle \\
\vdots & & \vdots \\
\langle \mathbf{1}, v_i\mathbf{x}_i\rangle & \langle \mathbf{1}, w_i\mathbf{x}_i + b_i\mathbf{1}_i\rangle & \langle \mathbf{1}, v_i\mathbf{1}_i\rangle \cdots \langle \mathbf{1}, \mathbf{1}\rangle
\end{pmatrix}$$

…whose generating vectors are…

$$\{v_i\mathbf{x}_i, w_i\mathbf{x}_i + b_i\mathbf{1}_i, v_i\mathbf{1}_i\}_{i=1}^{H} \cup \{\mathbf{1}\}$$

Properties of Loss Hessian:

- PSD
- PD iff gen. vectors are Linearly Independent
- Has 0 eigenvalues iff vectors are Linearly Dependent (flat in that direction, valley)

Critical Points fall into a 6 types: Local minima (1), Degenerate (5)

Thus, except possibly at a set of measure zero (the $\beta_i = x_n$ event), $\mathbf{H}_\ell$ is the (positive semi-definite) Gram matrix of the set

$$\{v_i\mathbf{x}_i, w_i\mathbf{x}_i + b_i\mathbf{1}_i, v_i\mathbf{1}_i\}_{i=1}^{H} \cup \{\mathbf{1}\}$$

Then, $\mathbf{H}_\ell \succ 0$ iff the vectors of this set are linearly independent. This is the case *unless*

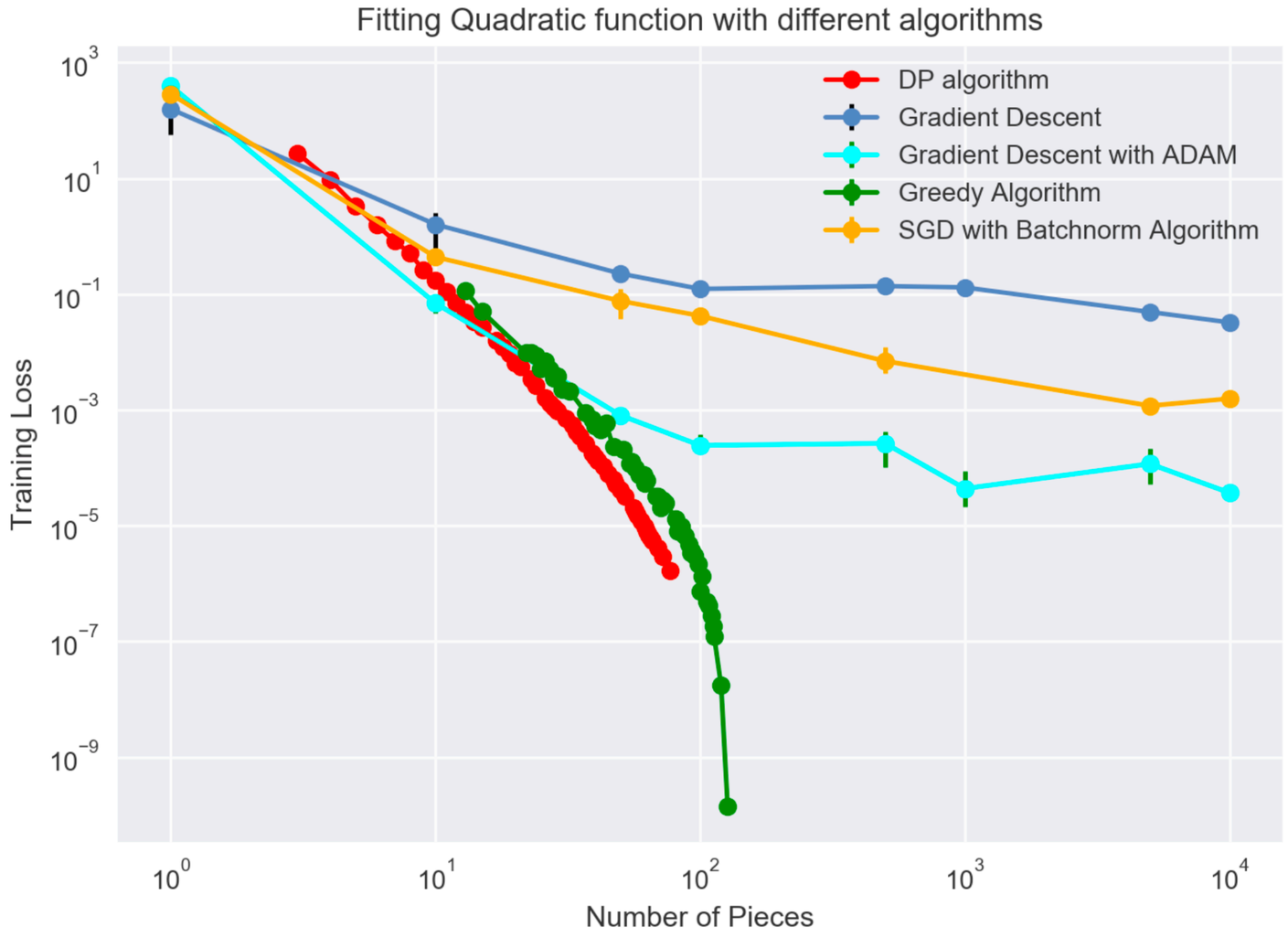
Remember these neurons…

1. any neurons share activation patterns
   - the loss function doesn't care about what the function does between data points, so any change in one neuron that is "cancelled out" by other neuron(s) between it and the data leaves the loss unchanged
2. any neuron is active on the entire data
   - its bias is redundant with the global bias, leading to a 1-dimensional subspace of constant loss
3. any neuron is active on no data (i.e. $\mathbf{x}_i = \mathbf{1}_i = \mathbf{0}$)
   - The breakpoint will be facing away from the data, so any change of its parameters that doesn't move the breakpoint into the data will have no effect on the loss (v_i can change arbitrarily, w_i and b_i will have half-spaces of constant loss)
4. if for any i, $\mathbf{x}_i \propto \mathbf{1}_i$
   - i.e. $w_i\mathbf{x}_i + b_i\mathbf{1}_i \equiv \alpha_i\mathbf{x}_i \propto v_i\mathbf{x}_i \propto v_i\mathbf{1}_i$
   - e.g. if i has only one active data point or has multiple data points *all* with the same x-value
   - you can "rotate" the line segment through its value at x_n
5. some of (w_i, v_i, b_i) are 0
   - if $v_i = 0$, the delta-slope is 0, so the location of the breakpoint, and thus the values of w_i and b_i do not matter
   - if $w_i = 0$, the breakpoint is at infinity, so the values of v_i and b_i do not matter
   - if $b_i = 0$, changing w_i will not move the breakpoint, so w_i and v_i can change so long as w_iv_i remains constant

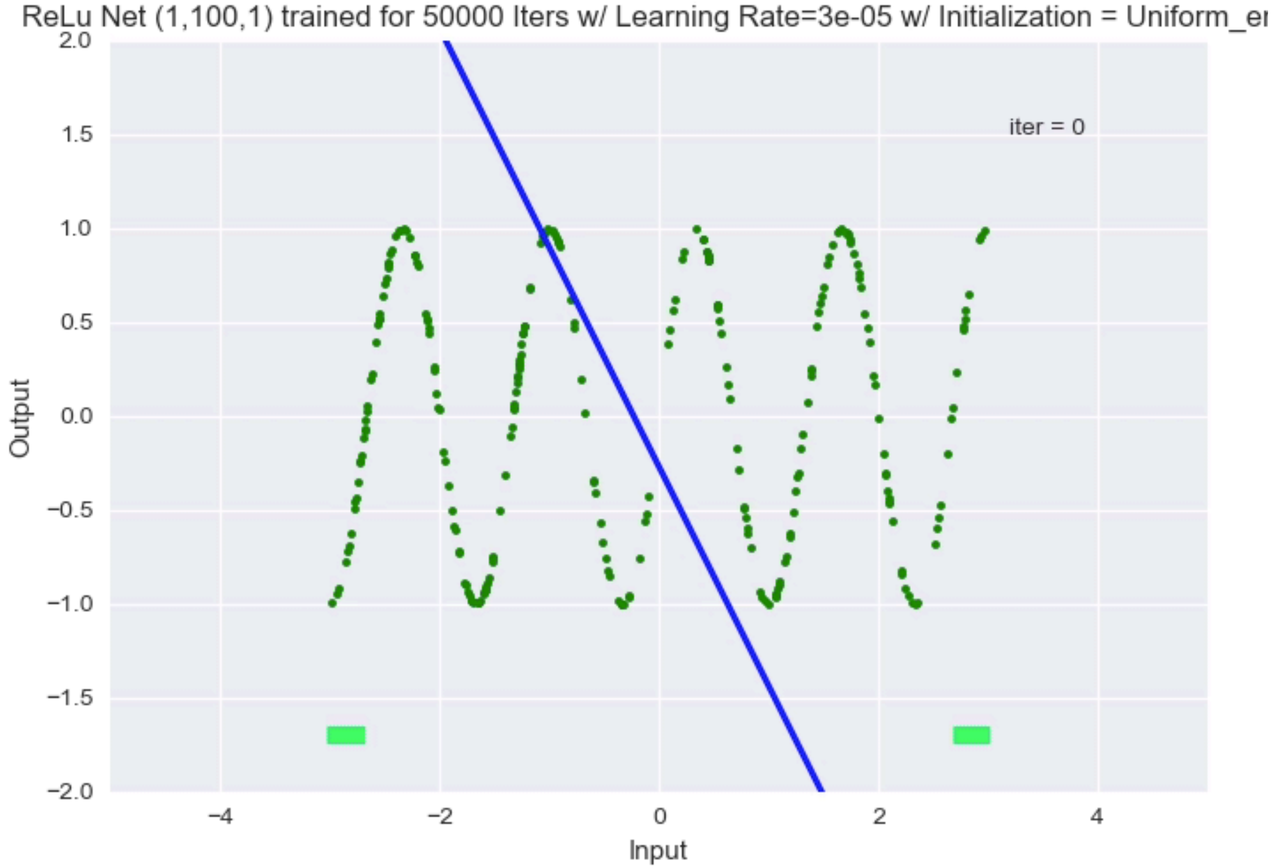
Gradient Descent:

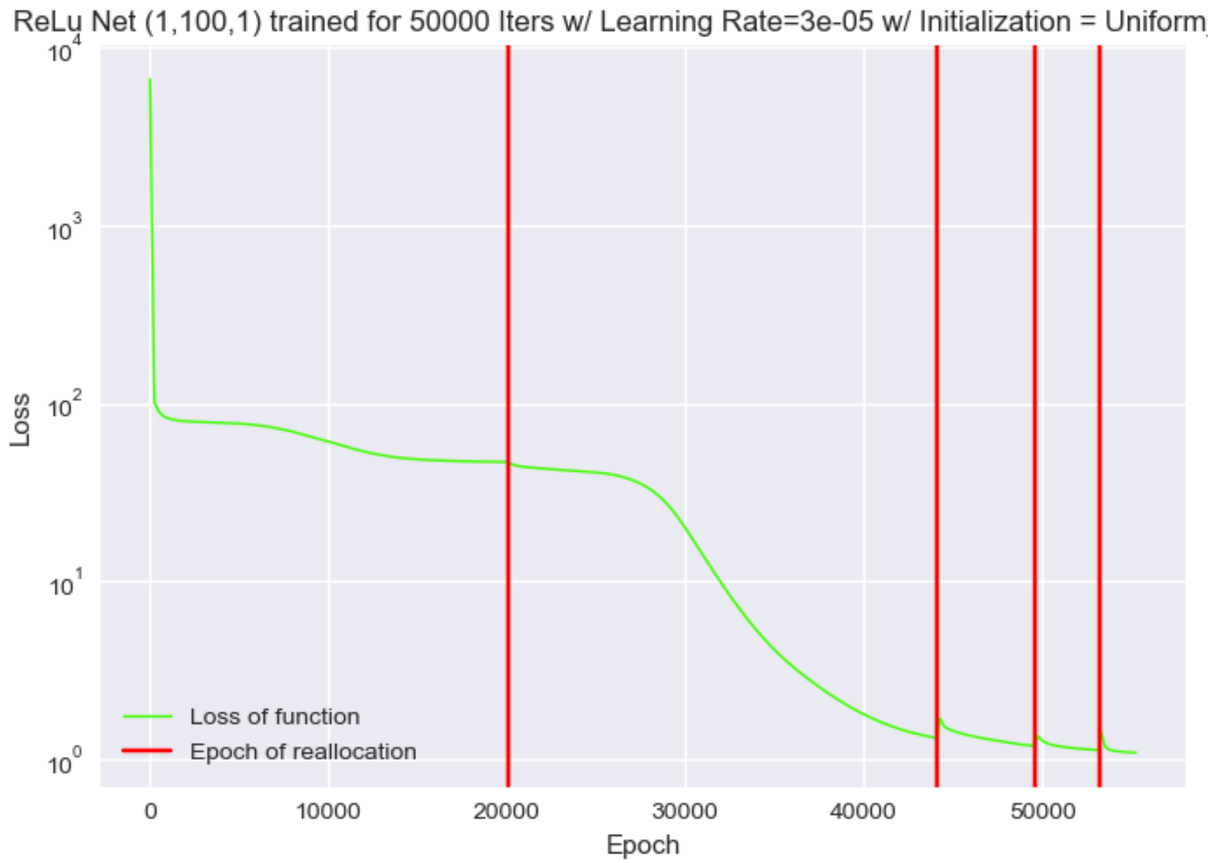
Suboptimality,
Breakpoint + Delta-Slope Dynamics,
& the Role of Depth

How *Suboptimal* is Gradient Descent?
Comparisons to Globally Optimal PWL Regression Algos



Can we improve GD by including global moves?
Relocating "Bad" Breakpoints During Training Rescues GD





Dynamical Laws for Breakpoints, Curvatures, Fcn

2.4 Gradient Descent Dynamics in the Function Space

Theorem 5. *For a one hidden layer univariate ReLU network trained with gradient descent with respect to the neural network parameters* $\theta_{NN} = \{(w_i, b_i, v_i)\}_{i=1}^{H}$*, the gradient flow dynamics of the function space parameters* $\theta_{BDSO} = \{(\beta_i, \mu_i)\}_{i=1}^{H}$ *are governed by the following laws:*

$$\frac{\mathrm{d}\beta_i}{\mathrm{d}t} = -\frac{\partial \ell(\theta_{NN})}{\partial \beta_i} = \frac{v_i(t)}{w_i(t)} \Big[\underbrace{\langle \hat{\boldsymbol{\epsilon}}(t) \odot \mathbf{a}_i(t), \mathbf{1} \rangle}_{\text{net relevant residual}} + \beta_i(t) \underbrace{\langle \hat{\boldsymbol{\epsilon}}(t) \odot \mathbf{a}_i(t), \mathbf{x} \rangle}_{\text{correlation}}\Big] \tag{4}$$

$$\frac{\mathrm{d}\mu_i(t)}{\mathrm{d}t} = -\frac{\partial \ell(\theta_{NN})}{\partial \mu_i} = -(v_i^2(t) + w_i^2(t)) \langle \hat{\boldsymbol{\epsilon}}(t) \odot \mathbf{a}_i(t), \mathbf{x} \rangle - w_i(t) b_i(t) \langle \hat{\boldsymbol{\epsilon}}(t) \odot \mathbf{a}_i(t), \mathbf{1} \rangle \tag{5}$$

Dynamical Laws for the Function (Neural Outputs):

Relation to the Neural Tangent Kernel (NTK)

2.4 Gradient Descent Dynamics in the Function Space

Theorem 5. *For a one hidden layer univariate ReLU network trained with gradient descent with respect to the neural network parameters* $\theta_{NN} = \{(w_i, b_i, v_i)\}_{i=1}^{H}$*, the gradient flow dynamics of the function space parameters* $\theta_{BDSO} = \{(\beta_i, \mu_i)\}_{i=1}^{H}$ *are governed by the following laws:*

$$\frac{\mathrm{d}\beta_i}{\mathrm{d}t} = -\frac{\partial \ell(\theta_{NN})}{\partial \beta_i} = \frac{v_i(t)}{w_i(t)} \Big[\underbrace{\langle \hat{\boldsymbol{\epsilon}}(t) \odot \mathbf{a}_i(t), \mathbf{1} \rangle}_{\text{net relevant residual}} + \beta_i(t) \underbrace{\langle \hat{\boldsymbol{\epsilon}}(t) \odot \mathbf{a}_i(t), \mathbf{x} \rangle}_{\text{correlation}}\Big] \tag{4}$$

$$\frac{\mathrm{d}\mu_i(t)}{\mathrm{d}t} = -\frac{\partial \ell(\theta_{NN})}{\partial \mu_i} = -(v_i^2(t) + w_i^2(t)) \langle \hat{\boldsymbol{\epsilon}}(t) \odot \mathbf{a}_i(t), \mathbf{x} \rangle - w_i(t) b_i(t) \langle \hat{\boldsymbol{\epsilon}}(t) \odot \mathbf{a}_i(t), \mathbf{1} \rangle \tag{5}$$

$$\begin{aligned}
\frac{\mathrm{d}\hat{y}_i(t)}{\mathrm{d}t} &= \sum_{j=1}^{n} \hat{\epsilon}_j(t)(\mathbf{H}'_{ij}(t) + \mathbf{G}_{ij}(t)) \\
&= \sum_{j=1}^{n} \text{similarity of } \mathbf{x}_i \text{ and } \mathbf{x}_j \text{ according to the network, weighted by the residual at } \mathbf{x}_j \\
&= \sum_{j=1}^{n} \text{residual at } \mathbf{x}_j, \text{ weighted by similarity to } \mathbf{x}_i \\
&= \langle \boldsymbol{\epsilon}, \mathbf{H}'_i(t) + \mathbf{G}_i(t) \rangle
\end{aligned}$$

Relation to Neural Tangent Kernel

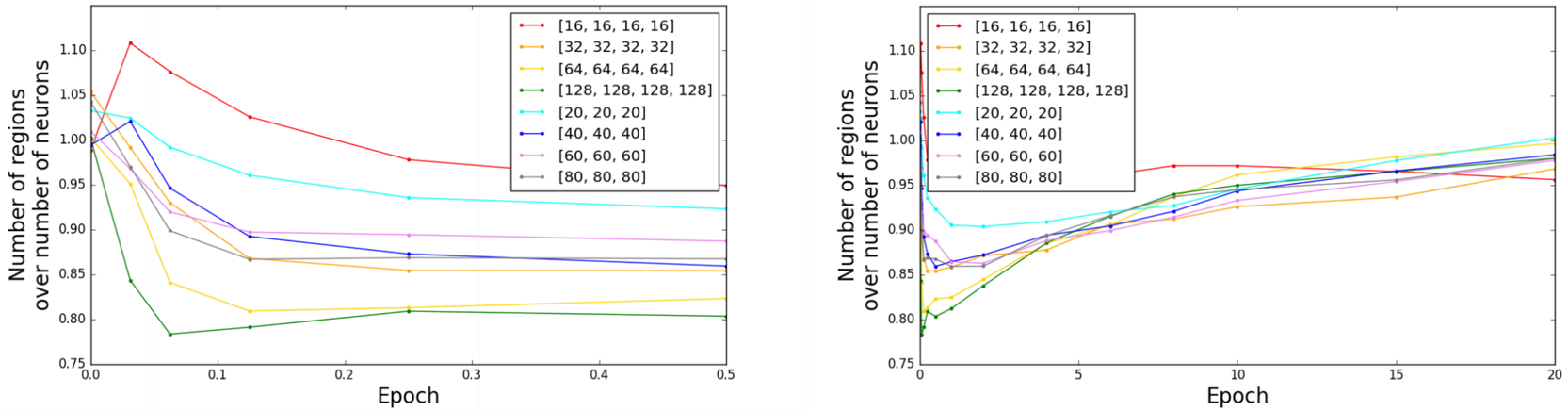
The Value of Depth: Expressivity or Learnability?

Depth doesn't add much Expressivity…

L	Sine	5 piece poly	Sawtooth	Arctan	Exponential	Quadratic
1	40 ± 0	40 ± 0	40 ± 0	40 ± 0	40 ± 0	40 ± 0
2	55.5 ± 2.9	52 ± 1.414	$50 \pm .7$	49.25 ± 3.3	51.25 ± 6.1	49.25 ± 4.5
4	68 ± 3.1	57.25 ± 6.8	48.5 ± 2.5	42.5 ± 4.8	40.25 ± 3.9	40.25 ± 3.3
5	62.25 ± 15.1	49 ± 3.5	44.5 ± 5.1	38 ± 5.1	33.75 ± 1.1	31.5 ± 1.7

Table 2: Comparison of the number of pieces induced in a network of up to depth 5, with 40 units evenly distributed across layers, trained to fit varying target functions.

Consistent with recent surprising findings from *Hanin & Rolnick 2019:*

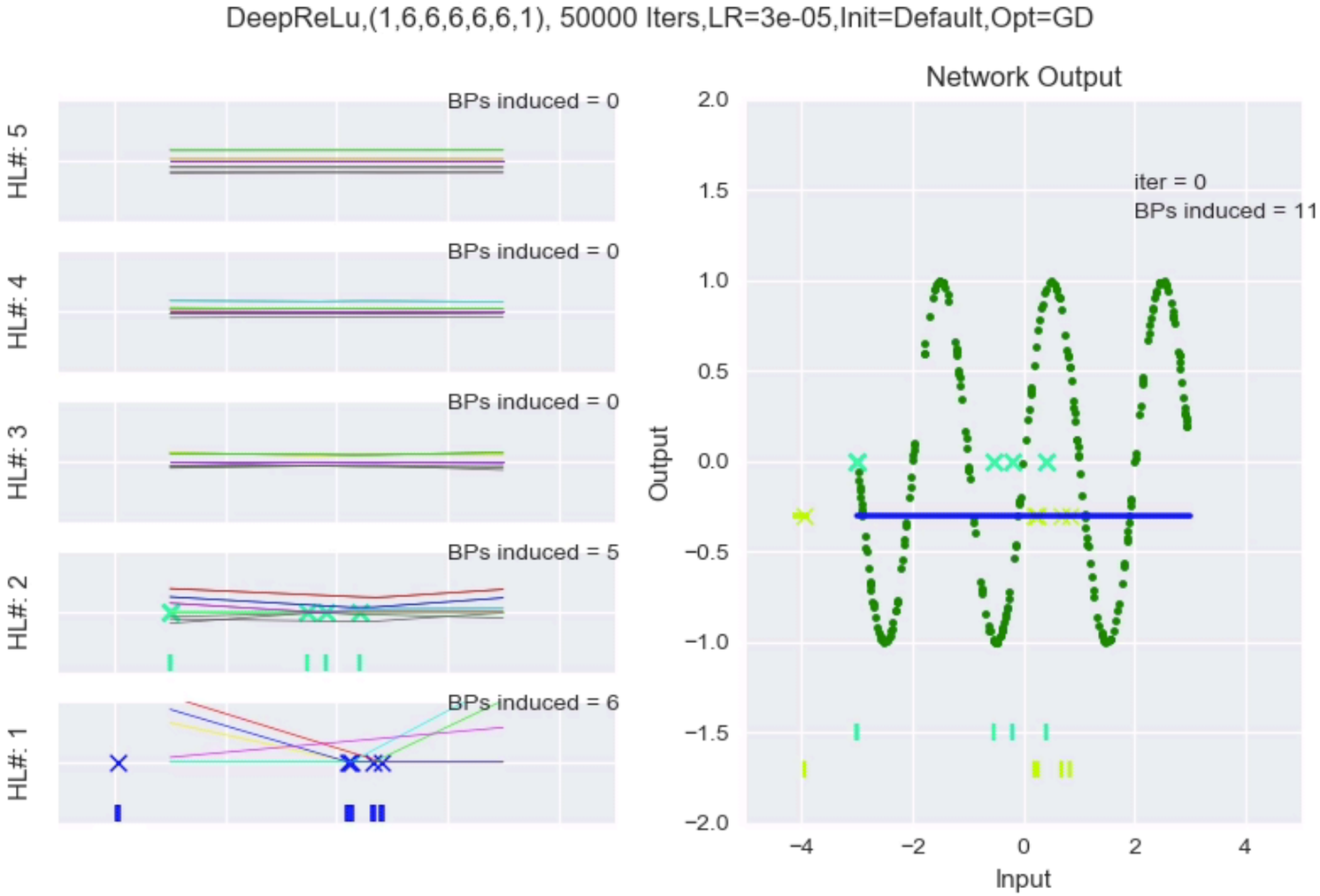


…then what is Depth good for?
Defining & Visualizing Breakpoints in Deep Nets

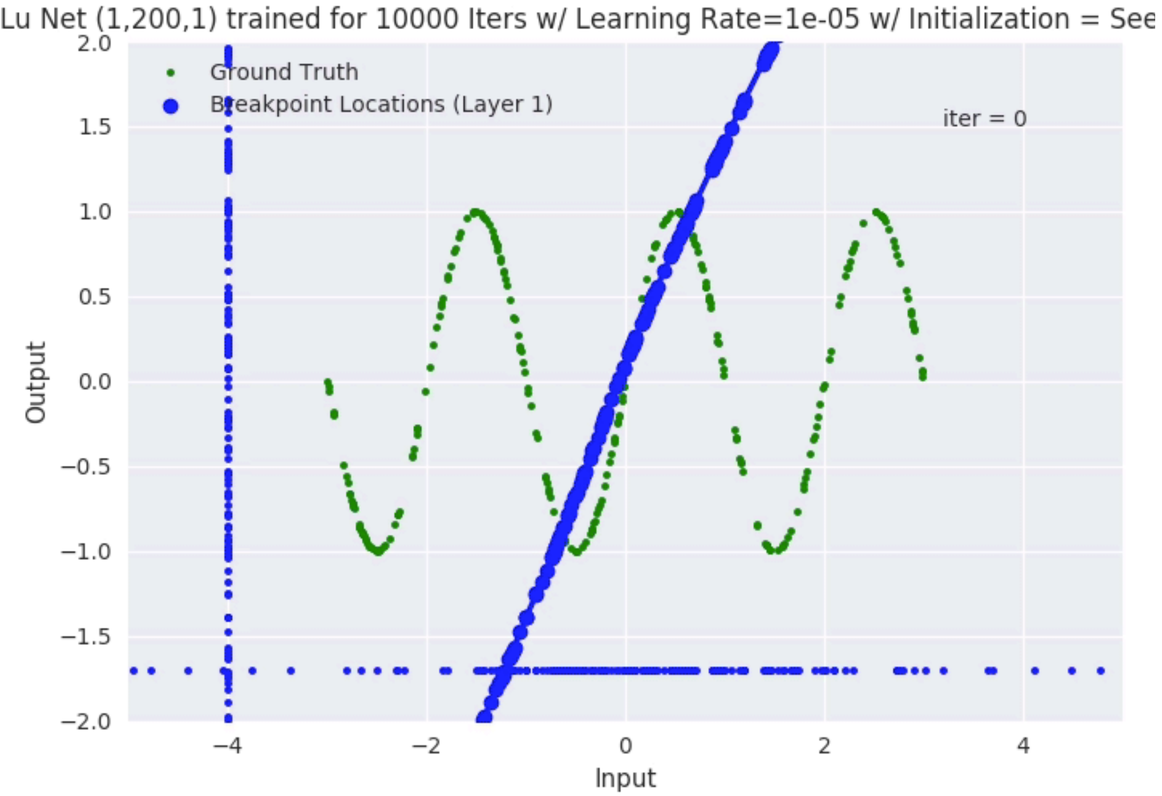
The Fine Print: Definition of Active Breakpoints is more subtle for Deep Nets

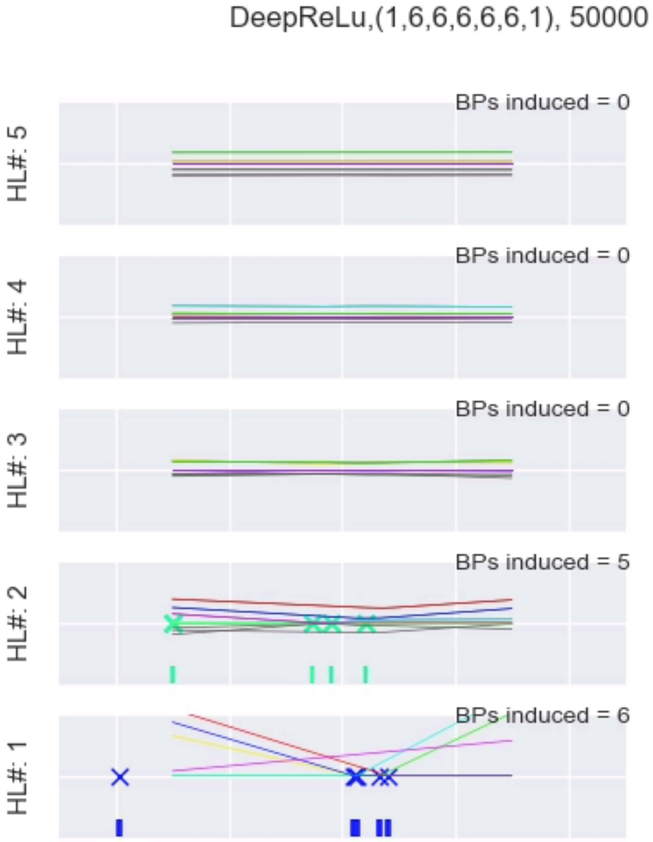
Definition 1. $\beta_i^{(\ell)}$ *is a* breakpoint induced by neuron i in layer ℓ *if* $z_i^{(\ell)}(\beta_i^{(\ell)}) = 0$. *Since the function* $z_i^{(\ell)}(\cdot)$ *is nonlinear, neuron* i *may induce multiple breakpoints, which we denote* $\beta_{i,k}^{(\ell)}$. *A breakpoint* $\beta_{i,k}^{(\ell)}$ *is* active *if there exists some path* π *through neuron* i *such that for all other neurons* $j \neq i \in \pi$, $z_j^{(\ell(j))} > 0$, *i.e.* $\widehat{a}_j(x) = 1$. *If two neurons* i *and* j *in layers* ℓ *and* ℓ' *induce the same breakpoint(s),* $\beta_{i,k}^{(\ell)} = \beta_{j,k'}^{(\ell')}$, *then both are referred to as* degenerate *breakpoints.*

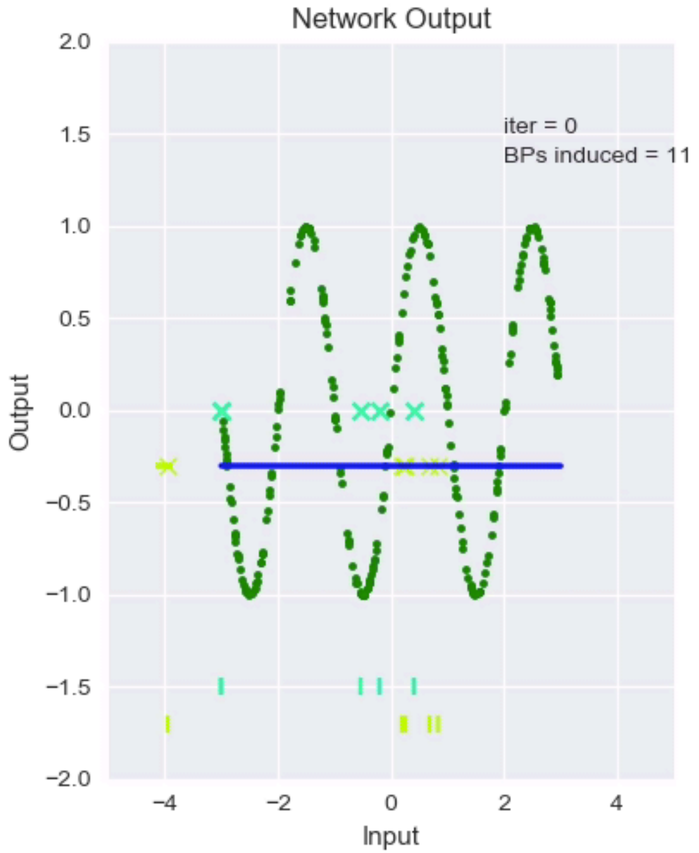
Let $\widehat{a}_\pi(x) = \prod_{i \in \pi} \widehat{a}_i$. Then, $\beta_i^{(\ell)}$ is active iff there exists some path π such that $\widehat{a}_\pi$ is discontinuous at $x = \beta_i^{(\ell)}$. Thus, $g_\theta(x)$ is non-differentiable at x if $x = \beta_i^{(\ell)}$ for some (ℓ, i). If no degenerate breakpoints exist, then the converse also holds. (If there do exist degenerate breakpoints $\beta_i^{(\ell)}$ and $\beta_j^{(\ell')}$, then it is possible that $\mu_i^{(\ell)} = -\mu_j^{(\ell')}$, i.e. the changes in slope cancel out and $g_\theta(x)$ remains linear and differentiable.)



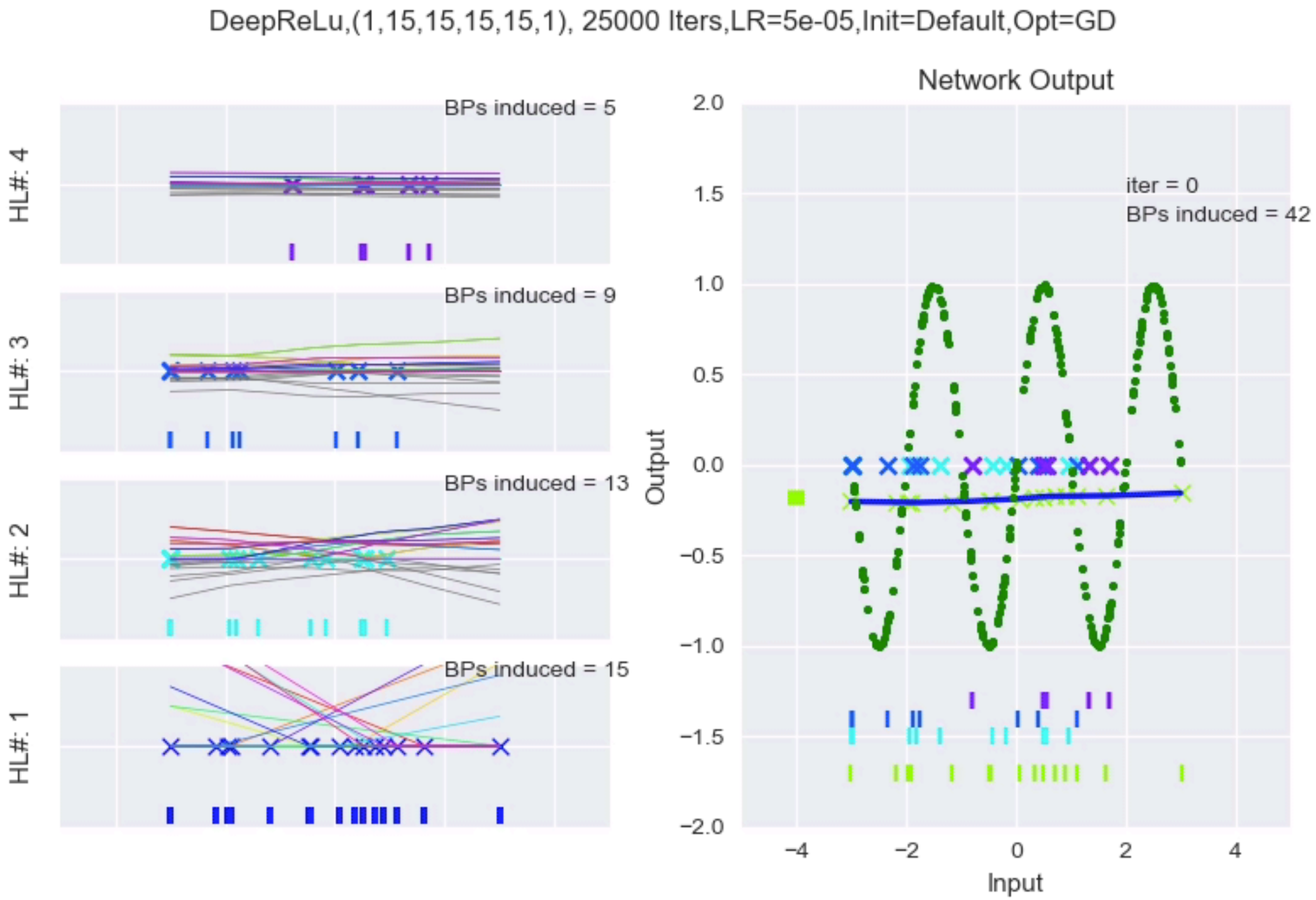
What is Depth good for?
Depth helps with Breakpoint Mobility

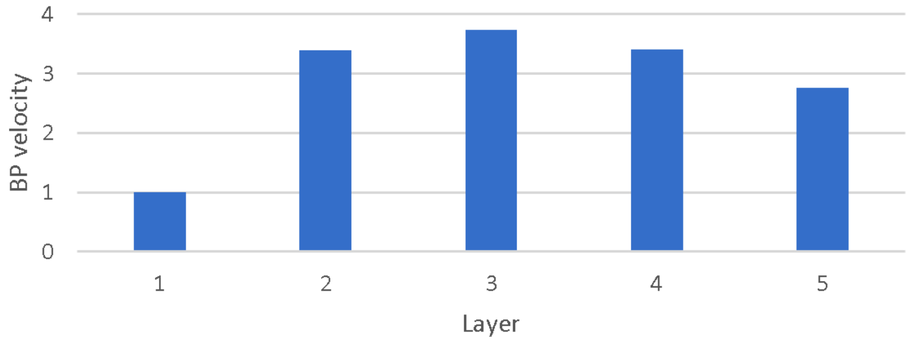


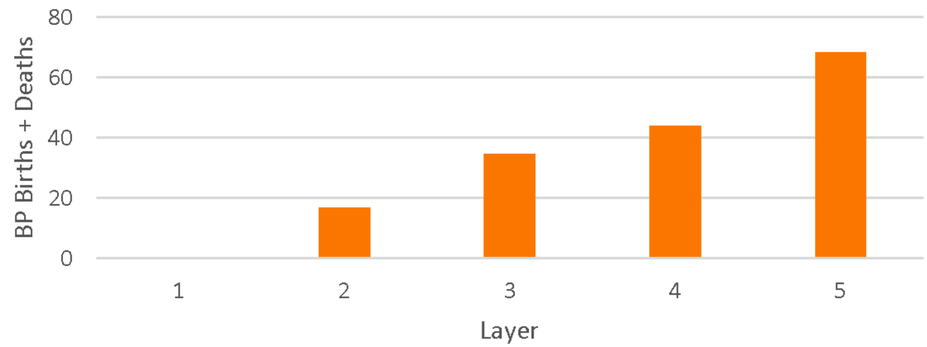




What is Depth good for?
Depth —> Breakpoint Speed, Birth & Death

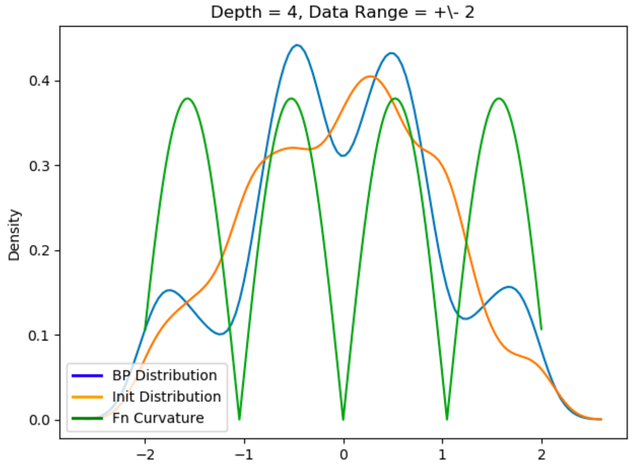




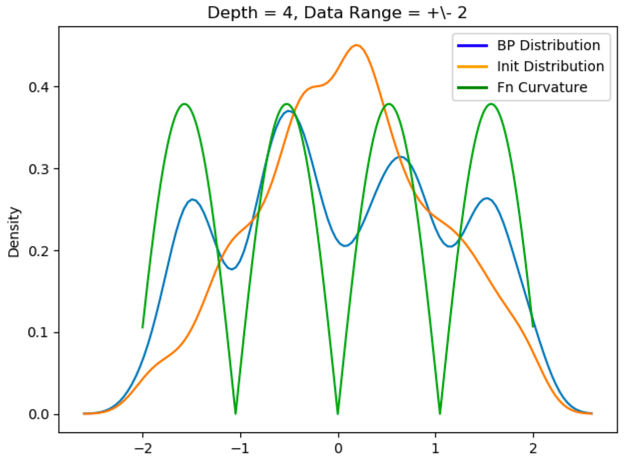


What is Depth good for?
Breakpoints in Deeper Nets are more Attracted to Curvature

For Shallow ReLu Net (1 layer):
Corr(Final BPs, Target Fcn Curvature) = 0.32



Deep ReLu Net (4 layers):
Corr(Final BPs, Target Fcn Curvature) = 0.49



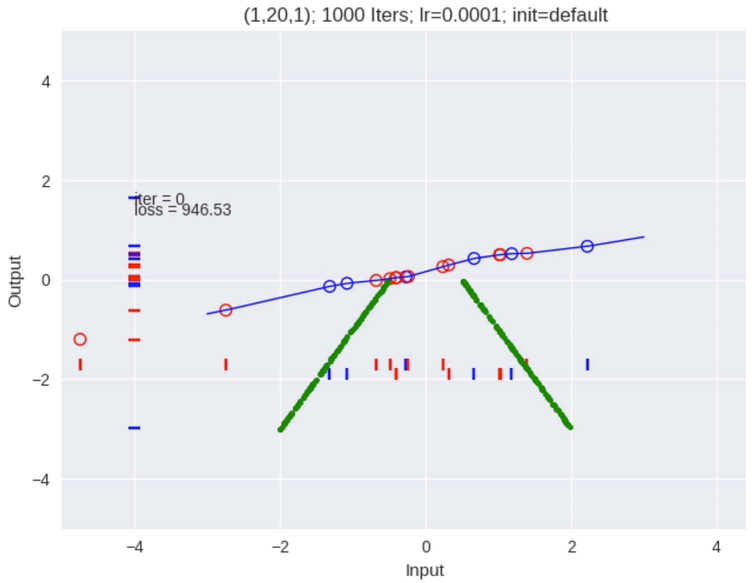
Depth/Type	Sine	x^3	$Sin(\frac{x^2}{2})$	Cubic Spline
1 - Initial	-0.0770	-0.904	-0.754	-0.164
1 - Final	0.324	-0.891	-0.592	0.289
4 - Initial	-0.171	-0.900	-0.752	-0.133
4 - Final	0.494	0.688	-0.212	0.798
1 - Δ	0.401	0.0130	0.162	0.452
4 - Δ	0.665	1.59	0.540	0.931

Deep nets have higher correlation b/w Breakpoint and Curvature locations than Shallow nets

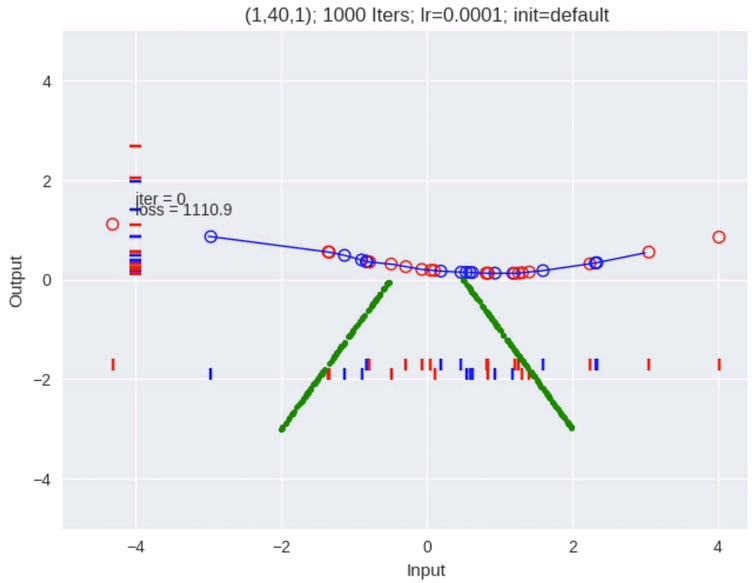
Generalization:
Explaining
Implicit Regularization

Implicit Reg.: Impact of Width for Two Lines

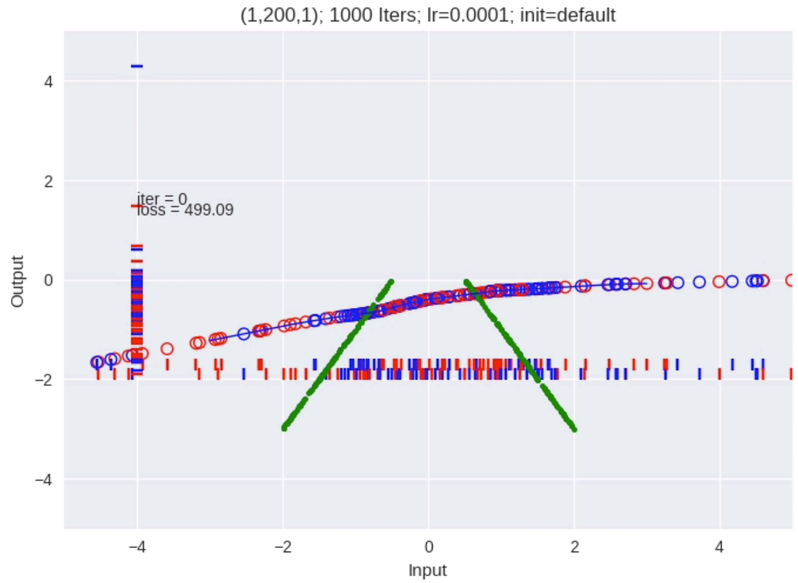
Width = 20 units



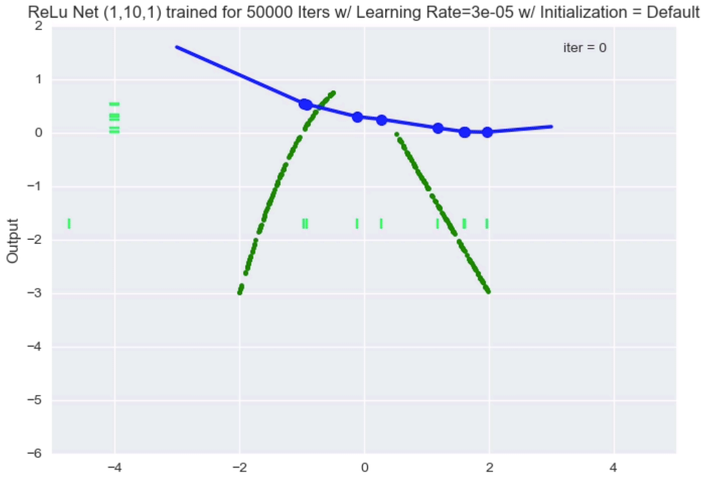
Width = 40 units

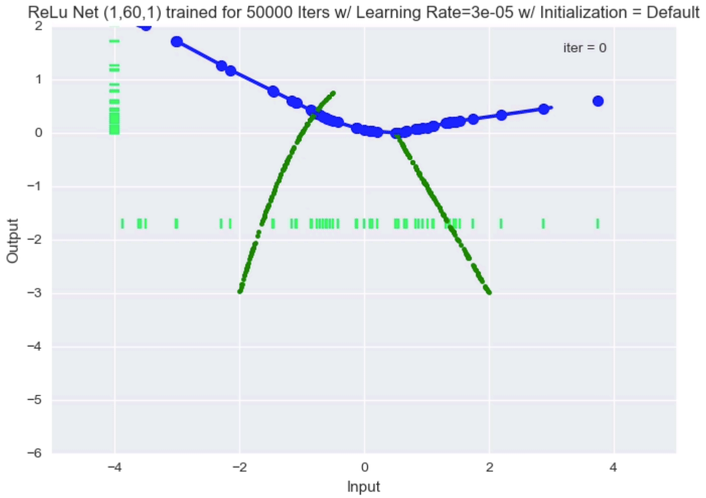


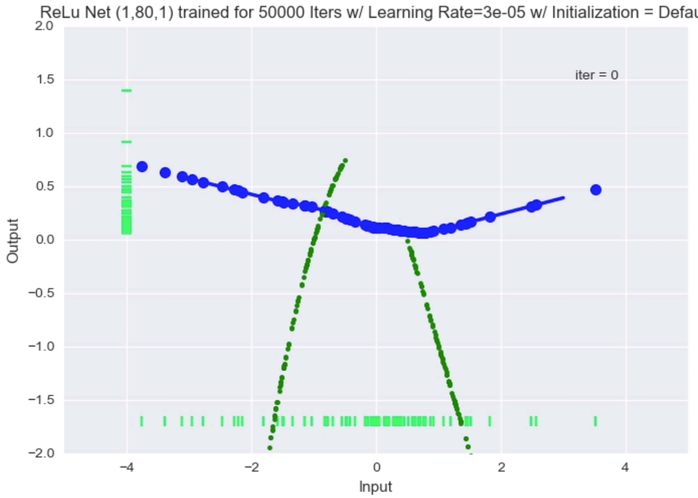
Width = 200 units

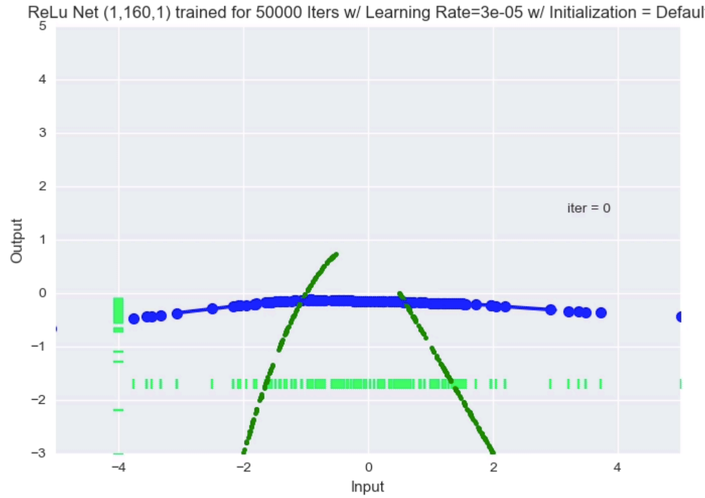


Implicit Reg.: Impact of Width for Smooth Target

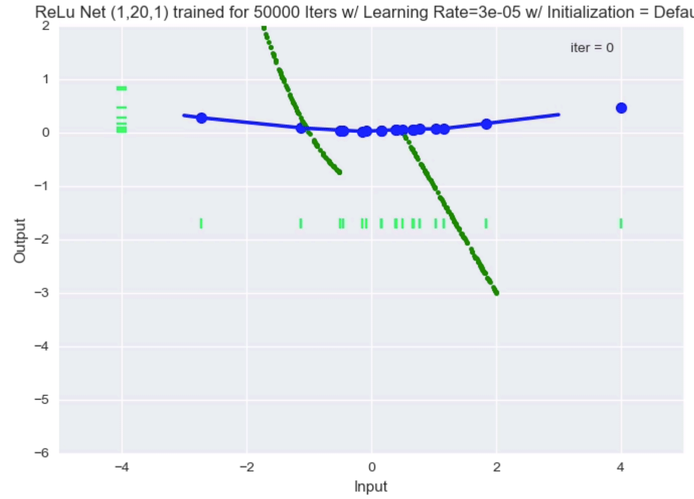


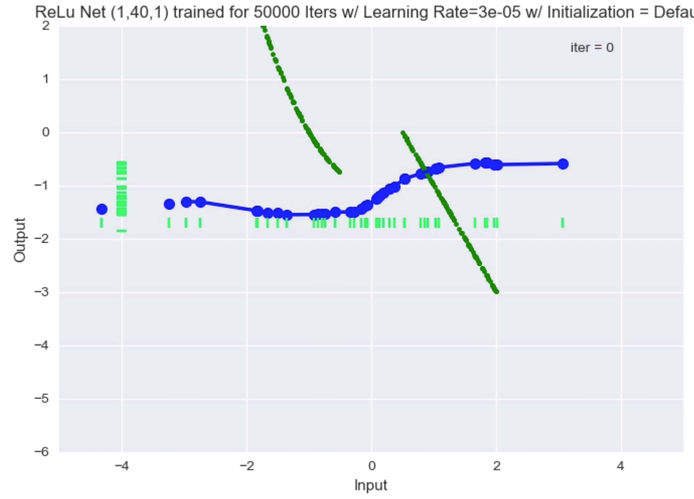


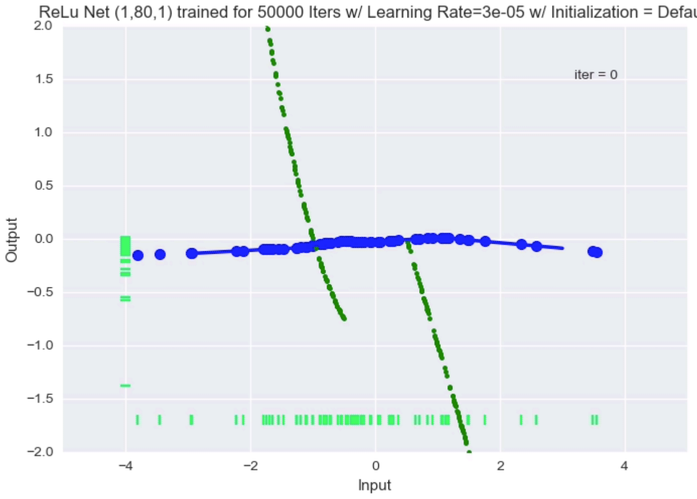




Implicit Reg.: Impact of Width for Sharp Target



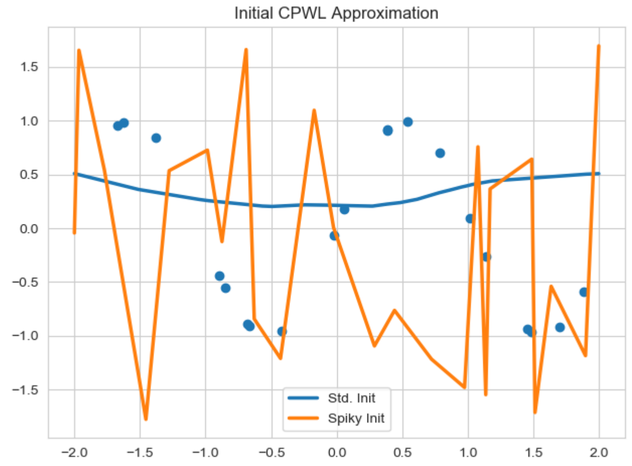


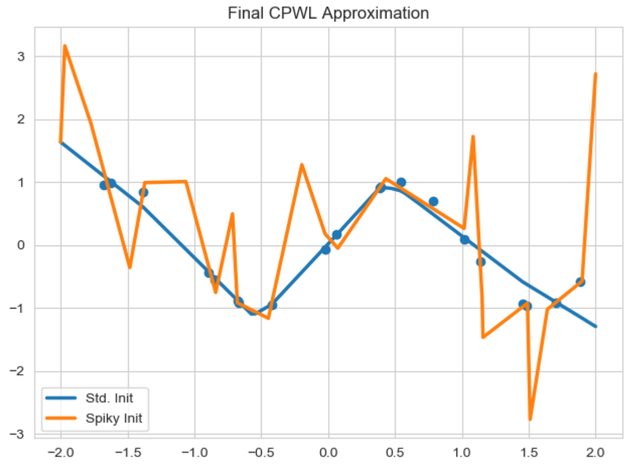


Implicit Reg.: Impact of Init Smoothness

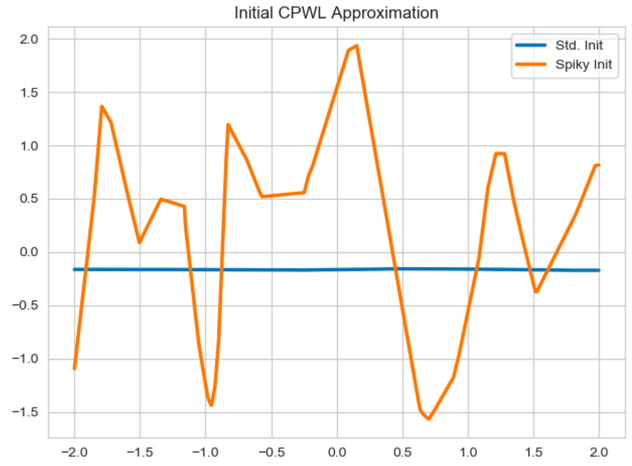
Initial | **Final**

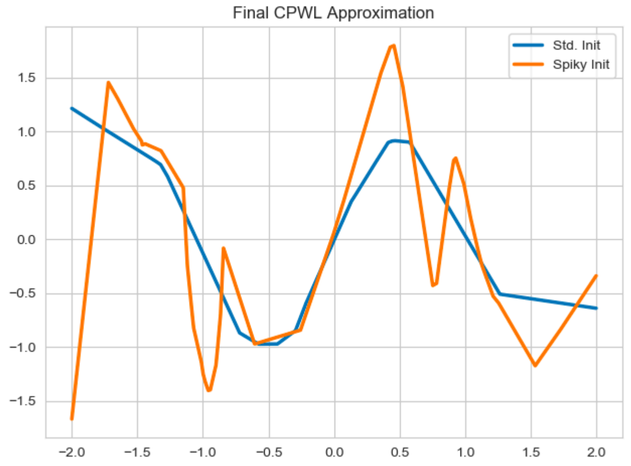
Shallow





Deep





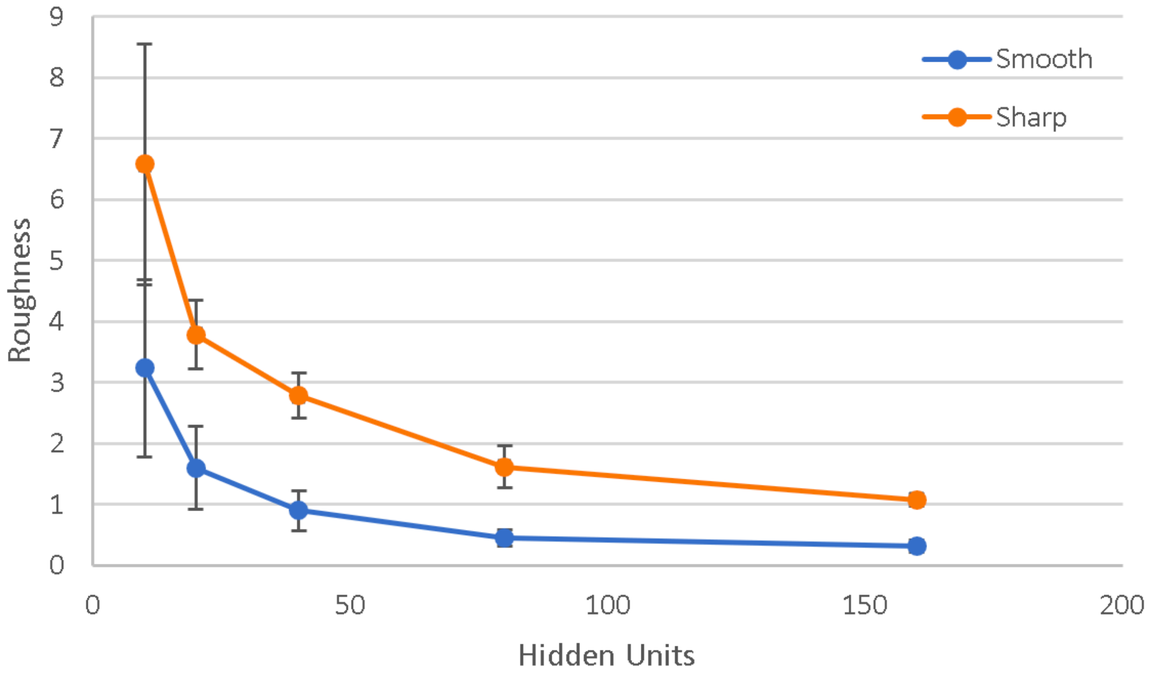
1. Spiky Inits are remembered thru training and significantly increase generalization error

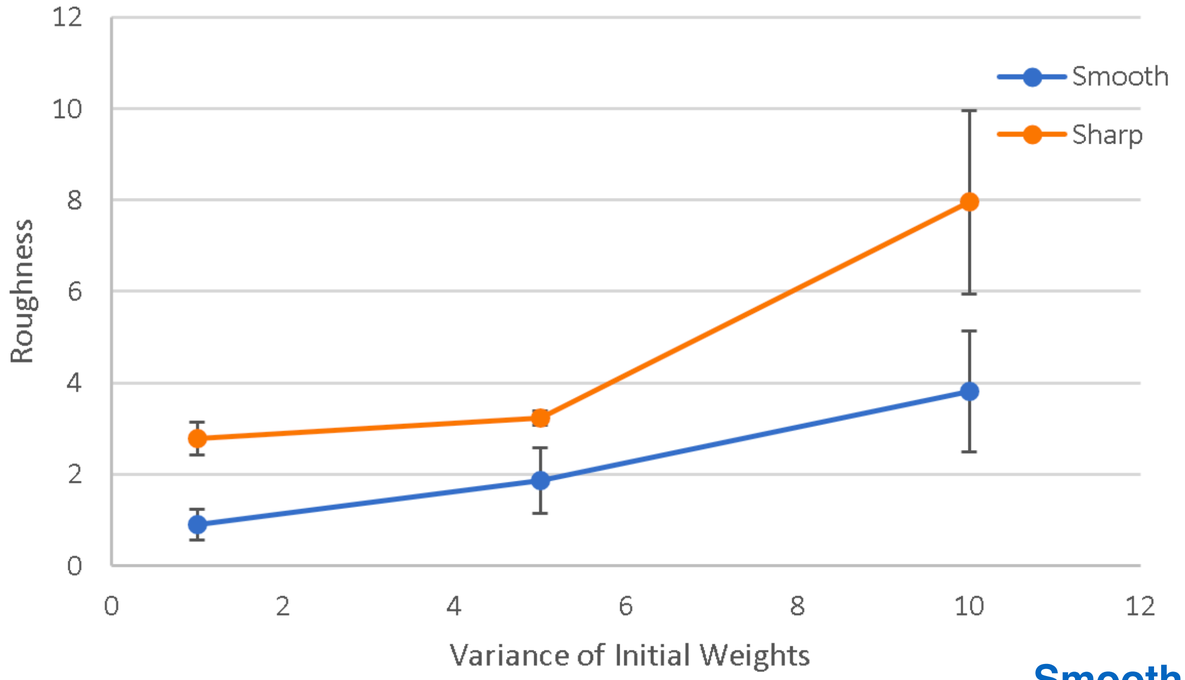
Function	Shallow	Spiky Shallow	Deep	Spiky Deep
Sine	42.95 ± 6.406	157.5 ± 60.27	31.48 ± 7.078	122.0 ± 128.2
Arctan	$.01252 \pm .07650$	2.499 ± 1.257	0.9795 ± 0.9355	32.57 ± 26.10
Sawtooth	156.9 ± 12.45	150.1 ± 61.48	148.1 ± 8.755	198.0 ± 170.9
Cubic	3.608 ± 1.683	136.7 ± 124.1	56.77 ± 98.91	191.6 ± 114.1
Quadratic	3.559 ± 4.553	150.6 ± 49.00	1.741 ± 1.296	46.02 ± 19.42
Exp	$.6509 \pm .5928$	$181.1 \pm 75.36 \pm$	1.339 ± 1.292	54.50 ± 37.77

Table 3: Comparison of testing loss (generalization ability) of various network shallow a networks with a standard vs 'spiky' initialization

2. Going Deeper can improve error but doesn't always help

Quantifying Implicit Regularization: Final Smoothness vs. Init Smoothness & Width





(proportional to Init. Roughness)

Smoothness Measured via Roughness:

$$\rho \equiv \sum_i \mu_i^2$$

Dynamical Laws for Breakpoints, Curvatures, Fcn

2.4 Gradient Descent Dynamics in the Function Space

Theorem 5. *For a one hidden layer univariate ReLU network trained with gradient descent with respect to the neural network parameters* $\theta_{NN} = \{(w_i, b_i, v_i)\}_{i=1}^{H}$*, the gradient flow dynamics of the function space parameters* $\theta_{BDSO} = \{(\beta_i, \mu_i)\}_{i=1}^{H}$ *are governed by the following laws:*

$$\frac{\mathrm{d}\beta_i}{\mathrm{d}t} = -\frac{\partial \ell(\theta_{NN})}{\partial \beta_i} = \frac{v_i(t)}{w_i(t)} \Big[\underbrace{\langle \hat{\boldsymbol{\epsilon}}(t) \odot \mathbf{a}_i(t), \mathbf{1} \rangle}_{\text{net relevant residual}} + \beta_i(t) \underbrace{\langle \hat{\boldsymbol{\epsilon}}(t) \odot \mathbf{a}_i(t), \mathbf{x} \rangle}_{\text{correlation}}\Big] \tag{4}$$

$$\frac{\mathrm{d}\mu_i(t)}{\mathrm{d}t} = -\frac{\partial \ell(\theta_{NN})}{\partial \mu_i} = -(v_i^2(t) + w_i^2(t)) \langle \hat{\boldsymbol{\epsilon}}(t) \odot \mathbf{a}_i(t), \mathbf{x} \rangle - w_i(t) b_i(t) \langle \hat{\boldsymbol{\epsilon}}(t) \odot \mathbf{a}_i(t), \mathbf{1} \rangle \tag{5}$$

Dynamical Laws for the Function (Neural Outputs)

2.4 Gradient Descent Dynamics in the Function Space

Theorem 5. *For a one hidden layer univariate ReLU network trained with gradient descent with respect to the neural network parameters* $\theta_{NN} = \{(w_i, b_i, v_i)\}_{i=1}^{H}$*, the gradient flow dynamics of the function space parameters* $\theta_{BDSO} = \{(\beta_i, \mu_i)\}_{i=1}^{H}$ *are governed by the following laws:*

$$\frac{\mathrm{d}\beta_i}{\mathrm{d}t} = -\frac{\partial \ell(\theta_{NN})}{\partial \beta_i} = \frac{v_i(t)}{w_i(t)} \Big[\underbrace{\langle \hat{\boldsymbol{\epsilon}}(t) \odot \mathbf{a}_i(t), \mathbf{1} \rangle}_{\textit{net relevant residual}} + \beta_i(t) \underbrace{\langle \hat{\boldsymbol{\epsilon}}(t) \odot \mathbf{a}_i(t), \mathbf{x} \rangle}_{\textit{correlation}} \Big] \tag{4}$$

$$\frac{\mathrm{d}\mu_i(t)}{\mathrm{d}t} = -\frac{\partial \ell(\theta_{NN})}{\partial \mu_i} = -(v_i^2(t) + w_i^2(t)) \langle \hat{\boldsymbol{\epsilon}}(t) \odot \mathbf{a}_i(t), \mathbf{x} \rangle - w_i(t) b_i(t) \langle \hat{\boldsymbol{\epsilon}}(t) \odot \mathbf{a}_i(t), \mathbf{1} \rangle \tag{5}$$

A Theoretical Explanation for Implicit Reg.:
1. Standard Random Inits are <u>very</u> Flat

Smoothness Measured via Roughness:

$$\rho \equiv \sum_i \mu_i^2$$

Smoothness is Highly Concentrated near zero —> Delta-Slopes near zero w.h.p.

Theorem 2. *Consider the initial roughness ρ_0 under a Gaussian initialization. In the He tion, we have that the tail probability is given by*

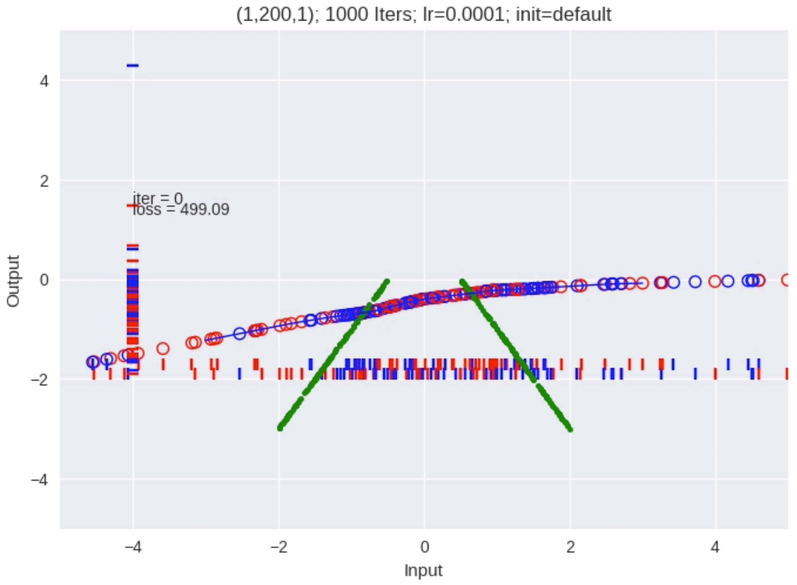
$$\mathbb{P}[\rho_0 - \mathbb{E}[\rho_0] \geq \lambda] \leq \frac{1}{1 + \frac{\lambda^2 H}{128}},$$

where $\mathbb{E}[\rho_0] = 4$. In the Glorot initialization, we have that the tail probability is given by

$$\mathbb{P}[\rho_0 - \mathbb{E}[\rho_0] \geq \lambda] \leq \frac{1}{1 + \frac{\lambda^2 (H+1)^4}{128 H}},$$

where $\mathbb{E}[\rho_0] = \frac{4H}{(H+1)^2} = O\left(\frac{1}{H}\right)$.

Width = 200 units



A Theoretical Explanation for Implicit Reg. in *Kernel Regime*:
2. Flat Init + GD + ReLu Parametrization —> Cubic Spline
Main Theoretical Results

Theorem 6. *Let $\boldsymbol{\mu}^*$ be the converged $\boldsymbol{\mu}$ parameter after gradient flow on the BDSO model Equation (7) starting from $\boldsymbol{\mu}_0 = \mathbf{0}$, with $\boldsymbol{\beta}$ held constant. And furthermore suppose that the model perfectly interpolates the training data $\tilde{\ell}(\theta_{BDSO}) = 0$. Then,*

$$\boldsymbol{\mu}^* = \arg\min_{\boldsymbol{\mu}} \|\boldsymbol{\mu}\|_2^2 \text{ s.t. } \mathbf{y} = \boldsymbol{\Phi}(\mathbf{x}; \boldsymbol{\beta})\boldsymbol{\mu}.$$

Corollary 4. *Consider the setting of Theorem 6, with the additional assumption that the breakpoints are uniformly spaced, and let $H \to \infty$. Then the learned function $\hat{f}_\infty(x; \boldsymbol{\mu}^*, \boldsymbol{\beta}^*)$ is the global minimizer of*

$$\inf_f \int_{-\infty}^{\infty} f''(x)^2 \, \mathrm{d}x \text{ s.t. } y_n = f(x_n) \; \forall n \in [N],$$

As such, $\hat{f}_\infty(x; \boldsymbol{\mu}^*, \boldsymbol{\beta}^*)$ is a natural cubic smoothing spline with N degrees of freedom (Ahlberg et al., 1967).

A Theoretical Explanation for Implicit Reg. in *Kernel Regime*:
2. Flat Init + GD + ReLu Parametrization —> Cubic Spline
Intuitive Proof Sketch

GD Dynamics of Individual (Discrete) Curvatures:

$$\dot{\mu}_i(t) = -\langle \hat{\epsilon}(t) \odot \mathbf{a}_i(t), \mathbf{x} \rangle + \beta_i(t) \langle \hat{\epsilon}(t) \odot \mathbf{a}_i(t), \mathbf{1} \rangle$$

Vectorize:

$$\dot{\boldsymbol{\mu}}_s = r_{2,s}(t)\mathbf{1} + r_{3,s}(t)\boldsymbol{\beta}_s$$

Neurons in data gap *all* see very similar gradients

Take Continuum Limit (Overparametrization) + Solve:

$$\dot{\mu}_s(x,t) = r_{2,s}(t) + r_{3,s}(t)x$$

$$\mu(x, t=\infty) = \mu(x, t=0) + R^*_{2,s} + R^*_{3,s}x$$

Init Curvature is Flat (essential for smoothness)

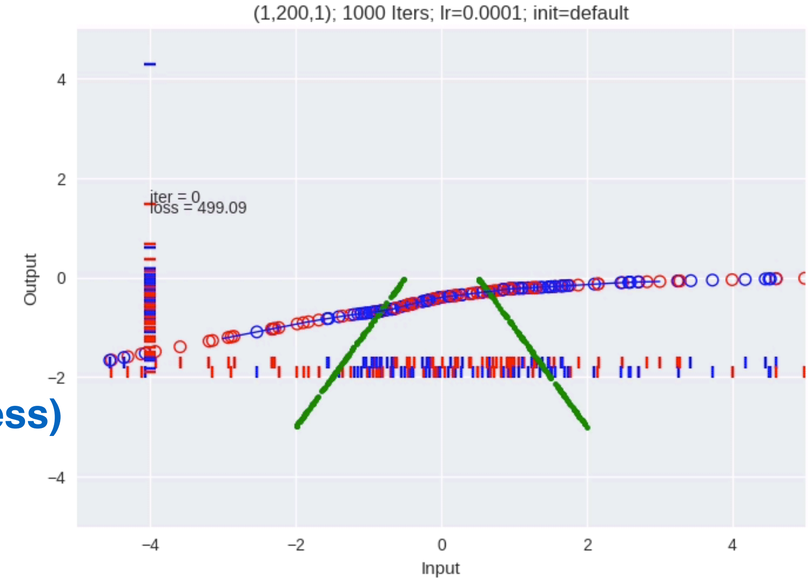
Integrate twice in x (Curvature-based Param)

$$\hat{f}(x,t) = c_{0,s} + c_{1,s}x + c_{2,s}(x-\xi_s)^2_s/2! + c_{3,s}(x-\xi_s)^3_s/3!$$

0th order PBCs (loss): $\hat{f}(x=\xi_s, t=\infty) = \sum_s f_s(x=\xi_s)$

1st order PBCs (param): $\hat{f}'(x=\xi_s, t=\infty) =$ **(Continuity constraint for slope of approximation)**

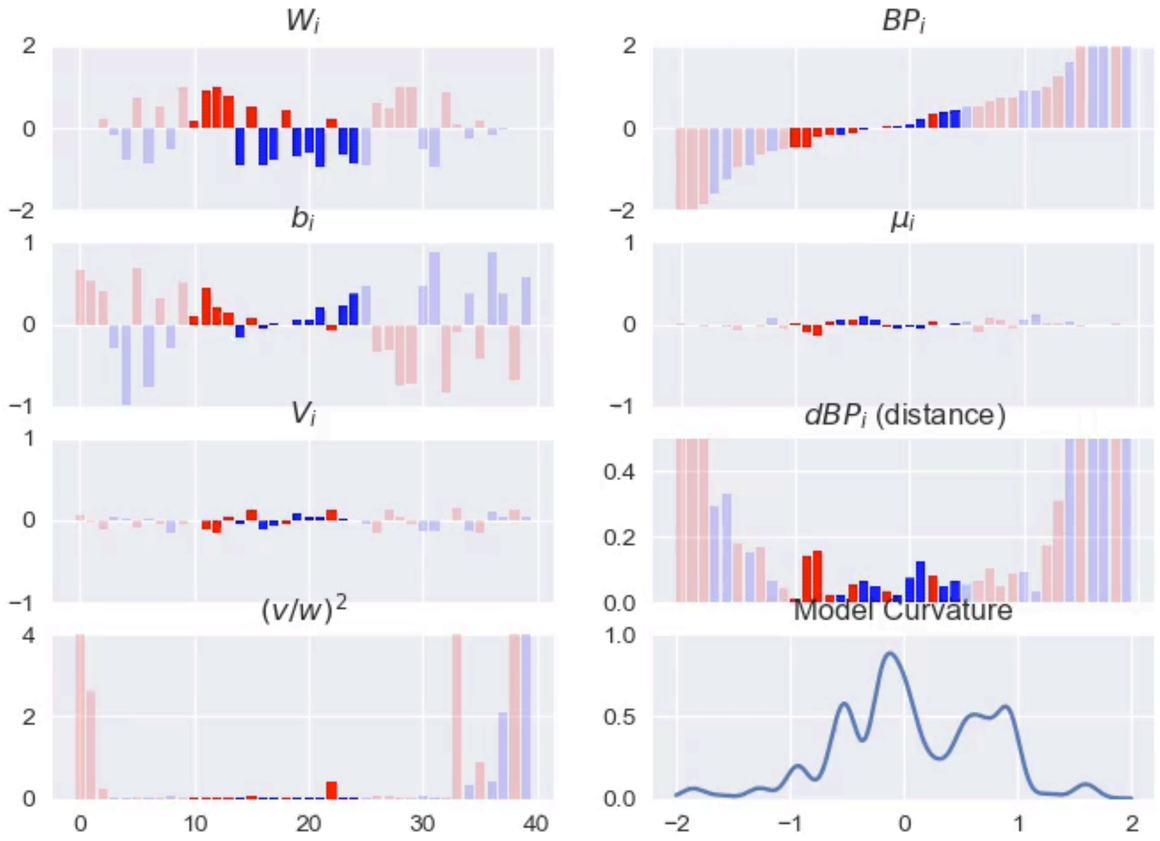
Width = 200 units



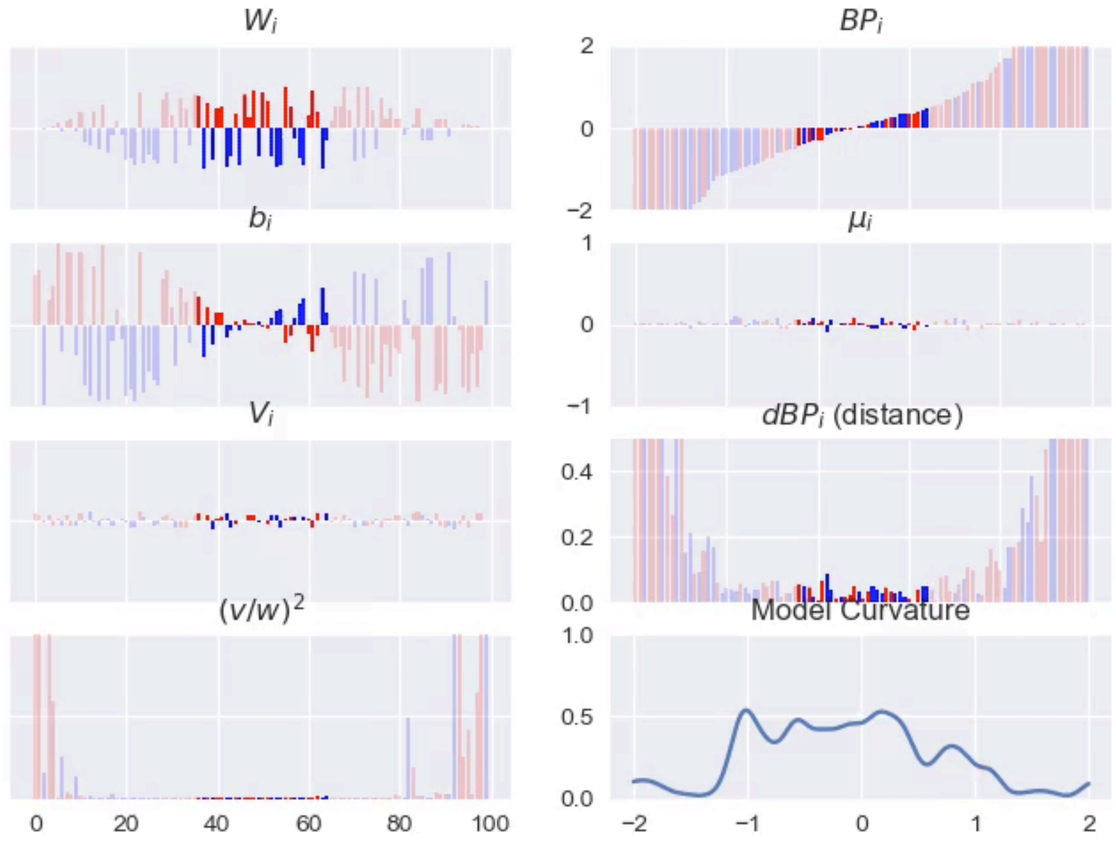
Surprise: 2nd+3rd order terms —> a piecewise *cubic* spline

A Theoretical Explanation for Implicit Reg. in *Kernel Regime*:
3. Test theory with simulations

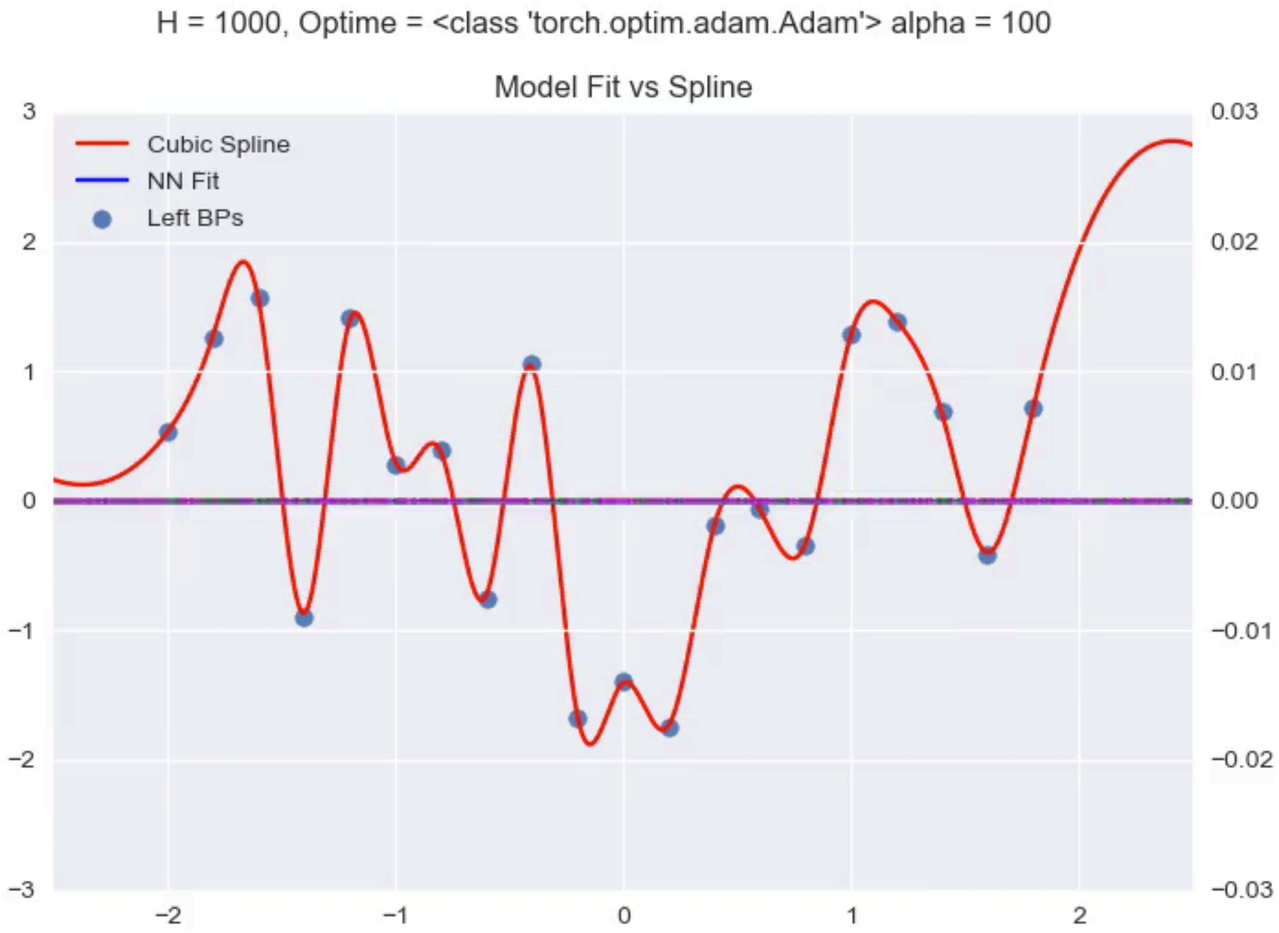
Width = 40 units



Width = 100 units



A Theoretical Explanation for Implicit Reg. in *Kernel Regime*:
3. Compare Predicted Spline to Trained NN



A Theoretical Explanation for Implicit Reg. in *Rich Regime*: *What Happens in the Rich Regime?*

Rich Regime (alpha=1)

Rich Regime (alpha=0.1)

A Theoretical Explanation for Implicit Reg. in *Rich Regime*: *What happens in the Rich Regime?*

Less Smoothness, More Concentration of Curvature amongst Breakpoints

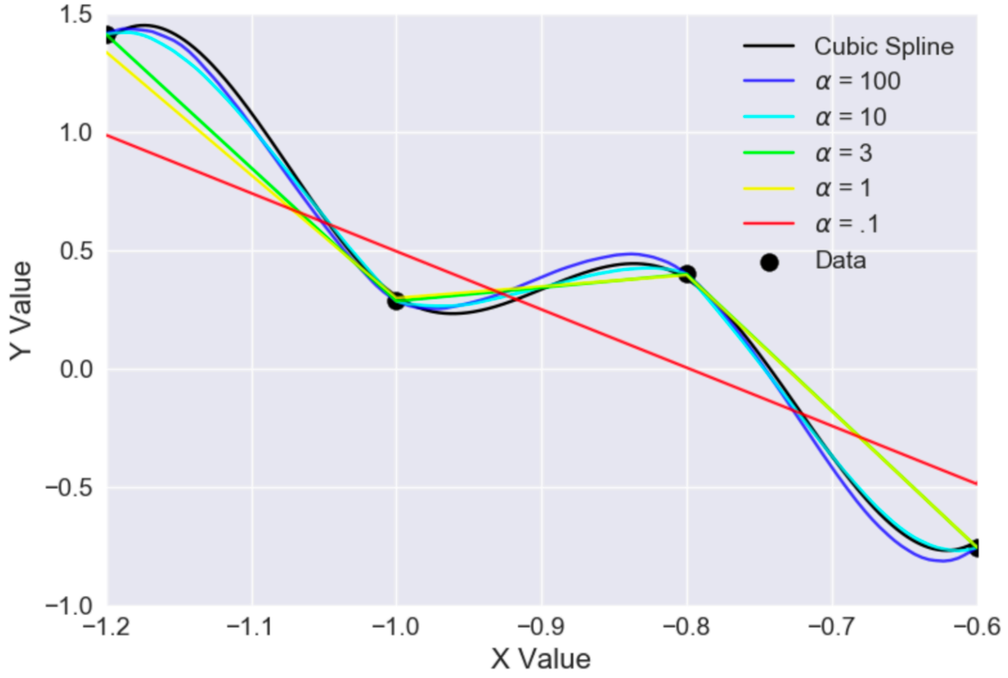


Figure 9. Effect of α, using same experimental protocol as *Table 2*, zoomed in on a small area of a single seed.

Why? Init Weight Scale controls Relative Learning Rate of Breakpoints vs. Delta-Slopes

2.4 GRADIENT DESCENT DYNAMICS IN THE FUNCTION SPACE

Theorem 5. *For a one hidden layer univariate ReLU network trained with gradient descent with respect to the neural network parameters* $\theta_{NN} = \{(w_i, b_i, v_i)\}_{i=1}^{H}$*, the gradient flow dynamics of the function space parameters* $\theta_{BDSO} = \{(\beta_i, \mu_i)\}_{i=1}^{H}$ *are governed by the following laws:*

$$\frac{\mathrm{d}\beta_i}{\mathrm{d}t} = -\frac{\partial \ell(\theta_{NN})}{\partial \beta_i} = \frac{v_i(t)}{w_i(t)} \Big[\underbrace{\langle \hat{\boldsymbol{\epsilon}}(t) \odot \mathbf{a}_i(t), \mathbf{1} \rangle}_{\textit{net relevant residual}} + \beta_i(t) \underbrace{\langle \hat{\boldsymbol{\epsilon}}(t) \odot \mathbf{a}_i(t), \mathbf{x} \rangle}_{\textit{correlation}}\Big] \tag{4}$$

$$\frac{\mathrm{d}\mu_i(t)}{\mathrm{d}t} = -\frac{\partial \ell(\theta_{NN})}{\partial \mu_i} = -(v_i^2(t) + w_i^2(t)) \langle \hat{\boldsymbol{\epsilon}}(t) \odot \mathbf{a}_i(t), \mathbf{x} \rangle - w_i(t) b_i(t) \langle \hat{\boldsymbol{\epsilon}}(t) \odot \mathbf{a}_i(t), \mathbf{1} \rangle \tag{5}$$

A Theoretical Explanation for Implicit Reg. in *Rich Regime*: *What happens in the Rich Regime?*

Theoretical Explanation: Piecewise Quadratic Loss Surface exhibits three types Of Critical Points, corresponding to the three types of Breakpoint attractors

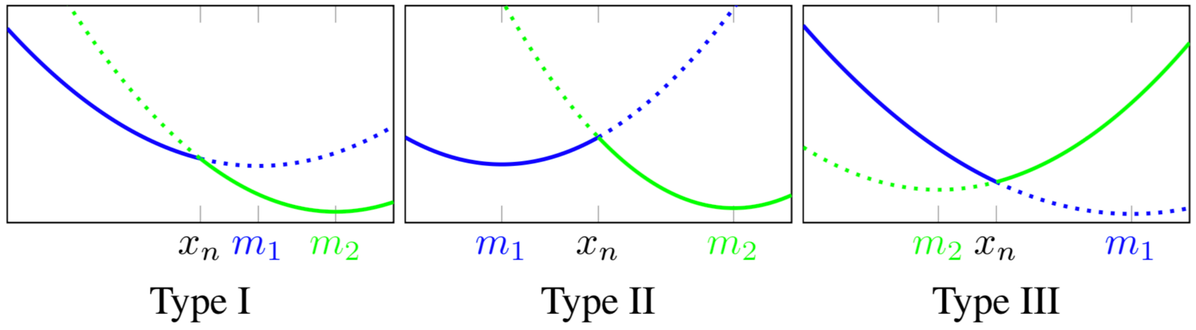


Figure 2. Classification of Critical Points

B.6. Loss Surface in the Spline Parametrization

Theorem 3. *The loss function $\tilde{\ell}(\theta_{BDSO} = (\boldsymbol{\beta}, \boldsymbol{\mu}), \mathbf{s})$ is a continuous piecewise quadratic spline. Furthermore, the 1-dimensional slice $\tilde{\ell}(\beta_i; \boldsymbol{\beta}_{-i}, \boldsymbol{\mu}, \mathbf{s})$ is also a continuous piecewise quadratic spline in β_i with knots at datapoints $\{x_n\}_{n=1}^N$. Let $p_1(\beta_i)$ ($p_2(\beta_i)$) be the left (right) piece at knot x_n, which both have positive curvature, and let $m_j \triangleq \arg\min p_j(\beta_i)$. Then, with measure 1, the knots x_n fall into one of four types as shown in Figure 2: (Type I) $m_1, m_2 < x_n$, or $x_n < m_1, m_2$, (Type II) $m_1 < x_n < m_2$ (Type III) $m_2 < x_n < m_1$.*

Proof. First, consider the following function:

$$g(\boldsymbol{\beta}, \boldsymbol{\mu}) = \sum_{n=1}^{N} \left(\sum_{i=1}^{H} \mu_i (x_n - \beta_i) - y_n \right)^2.$$

As a sum of squares of linear terms, this is clearly quadratic in $(\boldsymbol{\beta}, \boldsymbol{\mu})$. Then, note that the loss

$$\tilde{\ell}(\boldsymbol{\beta}, \boldsymbol{\mu}, \mathbf{s}) = \sum_{n=1}^{N} \left(\sum_{i=1}^{H} \mu_i (x_n - \beta_i)_{s_i} - y_n \right)^2$$

A Theoretical Explanation for Implicit Reg. in *Rich Regime*: *What Happens in Rich Regime?*

Rich Regime (alpha=1)

Rich Regime (alpha=0.1)

Implications for Theoretical and Computational Neuroscience

& Future Directions

The ***Spline PoV*** strongly encourages us to re-examine **Neuronal** Networks

- **Inference:** The role of individual neurons in representing/approximating a function:
  - Each neuron is a **basis function;** Each neuronal cell type is a type of basis function
  - ==> A population of neurons forms an **over complete basis:** distributed code with mixed selectivity is “normal” and efficient; IR determines how distributed/concentrated the code is
  - **Saturating Neuronal response** functions and **neuronal cell types** will have a dramatic impact on the representation and IR due to basis functions being saturated (vs. non-saturating e.g. ReLu)
- **Learning: Implicit Regularization** should occur for any **gradual learning** algorithm (described by a corresponding **PDE for the neuron population spline dynamics**)
  - **NN initialization is critical;** strong pressure for Evolution to select for specific implicit biases (stored in genome & established during development/lifetime via cell types, plasticity rules, etc.)
  - **Neuronal cell types** could be a very efficient way to genetically store this implicit bias; precludes the need to train from scratch in each lifetime; **E/I balance** should have a dramatic impact on representation/IR
  - **Neuron-Neuron response correlations are a *Feature* not a Bug**: they do not need to be “decorrelated”, instead they could signal IR

Summary & Future Work

- **Function space** is very useful for:
  - Understanding **Overparametrization, Loss Surface, Implicit Regularization**
  - Visualization and Probing
  - Developing New Inits & Learning Algorithms
- **Future Work:**
  - Develop theory for Deeper FFNNs, RNNs, GANs, etc.
  - Scale theory to high dimensions
  - Fast (approximate) Viz tools
  - Apply theory and tools to understanding Inductive Bias of neurally consistent models (Conductance-weighted averaging, Dale's Law)

(We're hiring!)

Web: *ankitlab.co/*
Twitter: @abp4_ankit

Thanks!

Future Work:

Developing Probing & Visualization Tools

Extending to Deep & Multivariate NNs

Arbitrary Activation Functions

Neuronal Consistent Nets with Cell Types and Saturating Responses

Ongoing Work:
Generalizing ReLU NN to Multivariate Inputs

Multivariate Spline Parametrization of a ReLU NN

$$\hat{f}_H(\mathbf{x};\theta_{\text{NN}}) = \frac{1}{\sqrt{H}}\sum_{i=1}^{H} v_i\left(\langle \mathbf{w}_i, \mathbf{x}\rangle + b_i\right)_+$$

$$= \frac{1}{\sqrt{H}}\sum_{i=1}^{H} v_i\|\mathbf{w}_i\|_2\left(\left\langle \frac{\mathbf{w}_i}{\|\mathbf{w}_i\|_2}, \mathbf{x}\right\rangle + \frac{b_i}{\|\mathbf{w}_i\|_2}\right)_+$$

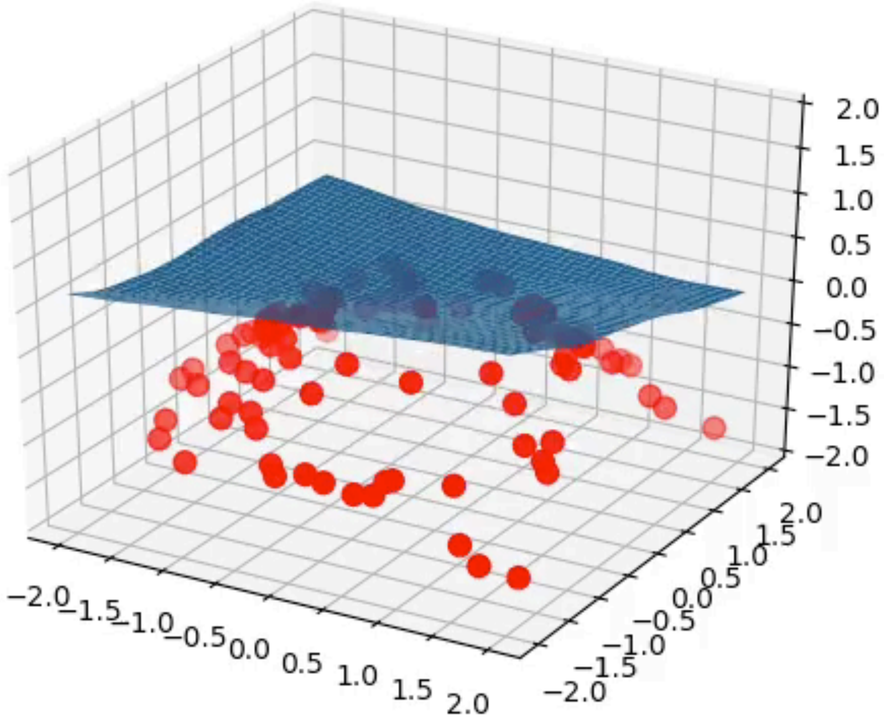
$$\hat{f}_H(\mathbf{x};\theta_{\text{BDSO}}) \triangleq \frac{1}{\sqrt{H}}\sum_{i=1}^{H} \mu_i(\langle \boldsymbol{\xi}_i, \mathbf{x}\rangle - \gamma_i)_+$$

$\mu_i \triangleq$ delta-slope $v_i\|\mathbf{w}_i\|_2$

$\boldsymbol{\xi}_i \triangleq$ unit vector normal to the break-plane $\dfrac{\mathbf{w}_i}{\|\mathbf{w}_i\|_2}$

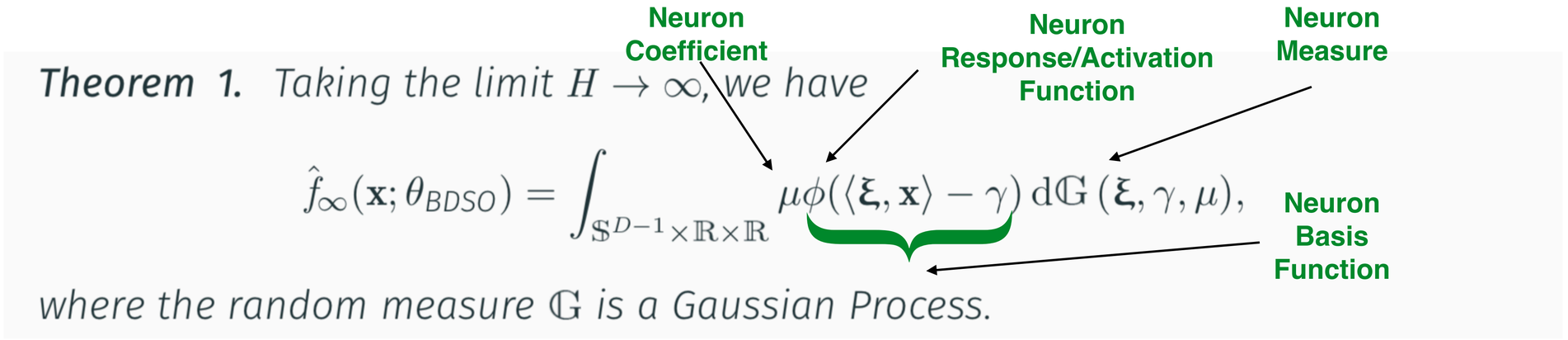
$\gamma_i \triangleq$ offset from origin $\dfrac{-b_i}{\|\mathbf{w}_i\|_2}$

2-Dimensional Inputs with Data Gap



Ongoing Work:
Generalizing to Multivariate Inputs w/ Arbitrary Activation Functions

Representation of NN: Approximating a Function via A Continuous Basis Expansion



Theorem 1. *Taking the limit $H \to \infty$, we have*

$$\hat{f}_\infty(\mathbf{x}; \theta_{BDSO}) = \int_{\mathbb{S}^{D-1}\times\mathbb{R}\times\mathbb{R}} \mu\phi(\langle \boldsymbol{\xi}, \mathbf{x}\rangle - \gamma)\, \mathrm{d}\mathbb{G}\,(\boldsymbol{\xi}, \gamma, \mu),$$

where the random measure $\mathbb{G}$ is a Gaussian Process.

Implication: The (well-developed) tools & mathematics of function space and basis expansions can be brought to bear on NN problems.

Ongoing Work:
Generalizing to Multivariate Inputs

Representation of NN: Approximating a Function via A Continuous Basis Expansion

Neuron Coefficient

Neuron Response Function

Neuron Measure

Neuron Basis Function

Theorem 1. *Taking the limit $H \to \infty$, we have*

$$\hat{f}_\infty(\mathbf{x}; \theta_{BDSO}) = \int_{\mathbb{S}^{D-1} \times \mathbb{R} \times \mathbb{R}} \mu \phi(\langle \boldsymbol{\xi}, \mathbf{x} \rangle - \gamma) \, \mathrm{d}\mathbb{G}(\boldsymbol{\xi}, \gamma, \mu),$$

where the random measure $\mathbb{G}$ is a Gaussian Process. Letting $c(\boldsymbol{\xi}, \gamma) = \mathbb{E}_{\mathbb{G}}[\mu | \boldsymbol{\xi}, \gamma]$, we have

$$\hat{f}_\infty(\mathbf{x}; \theta_{BDSO}) = \mathcal{R}^* \underbrace{\{(\phi *_\gamma c(\boldsymbol{\xi}, \cdot)\eta_0(\boldsymbol{\xi}, \cdot))(\gamma)\}}_{\phi \text{ is convolved with } c \cdot \eta_0 \text{ along the } \gamma \text{ direction}}(\mathbf{x}),$$

where $\mathcal{R}^*\{\cdot\}$ is the dual radon transform.

A NN representation aan be written as a *(dual) Radon Transform (related to Fourier Transform)*

Ongoing Work:
Generalizing to Multivariate Inputs

Representation of NN:
The role of each neuron is to represent coefficient x basis function

Theorem 2. *Under certain (strong) assumptions, we have*

$$c(\boldsymbol{\xi},\gamma) = \frac{-\gamma_D}{\eta_0(\boldsymbol{\xi},\gamma)}\mathcal{F}_\gamma^{-1}\left[\frac{\vartheta^{D-1}}{\mathcal{F}_\gamma[\phi](\vartheta)}\mathcal{F}_D\left[\hat{f}_\infty\right](\vartheta\boldsymbol{\xi})\right](\gamma)$$
$$\triangleq \frac{1}{\eta_0(\boldsymbol{\xi},\gamma)}\left(\mathcal{R}^*\right)^{-1}\left\{\mathbf{L}_{\boldsymbol{\xi}}^{-1}\hat{f}_\infty\right\}(\boldsymbol{\xi},\gamma),$$

where $\mathbf{L}_{\boldsymbol{\xi}}^{-1}$ *is the convolutional inverse of* ϕ*, i.e. the linear operator such that* $\mathbf{L}_{\boldsymbol{\xi}}^{-1}\phi = \delta$*, applied in the direction of* $\boldsymbol{\xi}$

Ongoing Work:
Generalizing to Multivariate Inputs
Neuron Response Functions Control the Basis:
ReLu

Corollary 4. *Suppose $\phi(\cdot) = (\cdot)_+$ (ReLU activation). Then,*

$$c(\boldsymbol{\xi}, \gamma) = \frac{-\gamma_D}{\eta_0(\boldsymbol{\xi}, \gamma)} \mathcal{F}_\gamma^{-1} \left[\vartheta^{D-1} (-\vartheta^2) \mathcal{F}_D \left[\hat{f}_\infty \right] (\vartheta \boldsymbol{\xi}) \right] (\gamma)$$
$$= \frac{1}{\eta_0(\boldsymbol{\xi}, \gamma)} (\mathcal{R}^*)^{-1} \left\{ \nabla^2 \hat{f}_\infty \right\} (\boldsymbol{\xi}, \gamma)$$

Ongoing Work:
Generalizing to Multivariate Inputs

Neuron Response Functions Control the Basis:
Step

Corollary 5. *Suppose $\phi(\cdot) = \Theta(\cdot)$ (Step function/Heaviside activation). Then,*

$$c(\boldsymbol{\xi}, \gamma) = \frac{-\gamma_D}{\eta_0(\boldsymbol{\xi}, \gamma)} \mathcal{F}_\gamma^{-1} \left[\vartheta^{D-1} i\vartheta \mathcal{F}_D \left[\hat{f}_\infty \right] (\vartheta \boldsymbol{\xi}) \right] (\gamma)$$
$$= \frac{1}{\eta_0(\boldsymbol{\xi}, \gamma)} (\mathcal{R}^*)^{-1} \left\{ \partial_\xi \hat{f}_\infty \right\} (\boldsymbol{\xi}, \gamma)$$

Ongoing Work:
Generalizing to Multivariate Inputs

Neuron Response Functions Control the Basis: Saturating ReLu

Lemma 6. *Suppose $\phi(\cdot) = (\cdot)_+ - (\cdot - \Delta)_+$ (fixed-width un-normalized saturating ReLU activation). Then,*

$$c(\boldsymbol{\xi}, \gamma) = \frac{-\gamma_D}{\eta_0(\boldsymbol{\xi}, \gamma)} \mathcal{F}_\gamma^{-1} \left[\frac{\vartheta^{D-1}(-\vartheta^2)}{1 - e^{-i\Delta\vartheta}} \mathcal{F}_D \left[\hat{f}_\infty \right] (\vartheta \boldsymbol{\xi}) \right] (\gamma)$$

$$= \frac{1}{\eta_0(\boldsymbol{\xi}, \gamma)} (\mathcal{R}^*)^{-1} \left\{ \nabla^2 \left[\sum_{k=0}^{\infty} \hat{f}_\infty(\cdot - \Delta \boldsymbol{\xi} k) \right] \right\} (\boldsymbol{\xi}, \gamma)$$

Ongoing Work:
Generalizing to Multivariate Inputs

Neuron Response Functions Control the Basis: PowerReLU

Lemma 3. *Suppose* $\phi(x) = \frac{(x)_+^{\lambda-1}}{\Gamma(\lambda)}$ *(Power-ReLU activation,* $\lambda > 0$*). Then,*

$$\begin{aligned} c(\boldsymbol{\xi}, \gamma) &= \frac{-\gamma_D}{\eta_0(\boldsymbol{\xi}, \gamma)} \mathcal{F}_\gamma^{-1} \left[\frac{\vartheta^{D-1}}{(i\vartheta)^{-\lambda}} \mathcal{F}_D \left[\hat{f}_\infty \right] (\vartheta \boldsymbol{\xi}) \right] (\gamma) \\ &= \frac{-\gamma_D}{\eta_0(\boldsymbol{\xi}, \gamma)} \mathcal{F}_\gamma^{-1} \left[\vartheta^{D-1} \mathcal{F}_D \left[\square_{+,\mathbf{x}}^{\lambda} \hat{f}_\infty \right] (\vartheta \boldsymbol{\xi}) \right] (\gamma) \end{aligned}$$

where

$$\square_{+,\mathbf{x}}^{\lambda} \triangleq \frac{\sum_{d=1}^{D} \mathbb{D}_{+,\mathbf{e}_d}^{\lambda}}{\langle \boldsymbol{\xi}^{\odot\lambda}, \mathbf{1} \rangle},$$

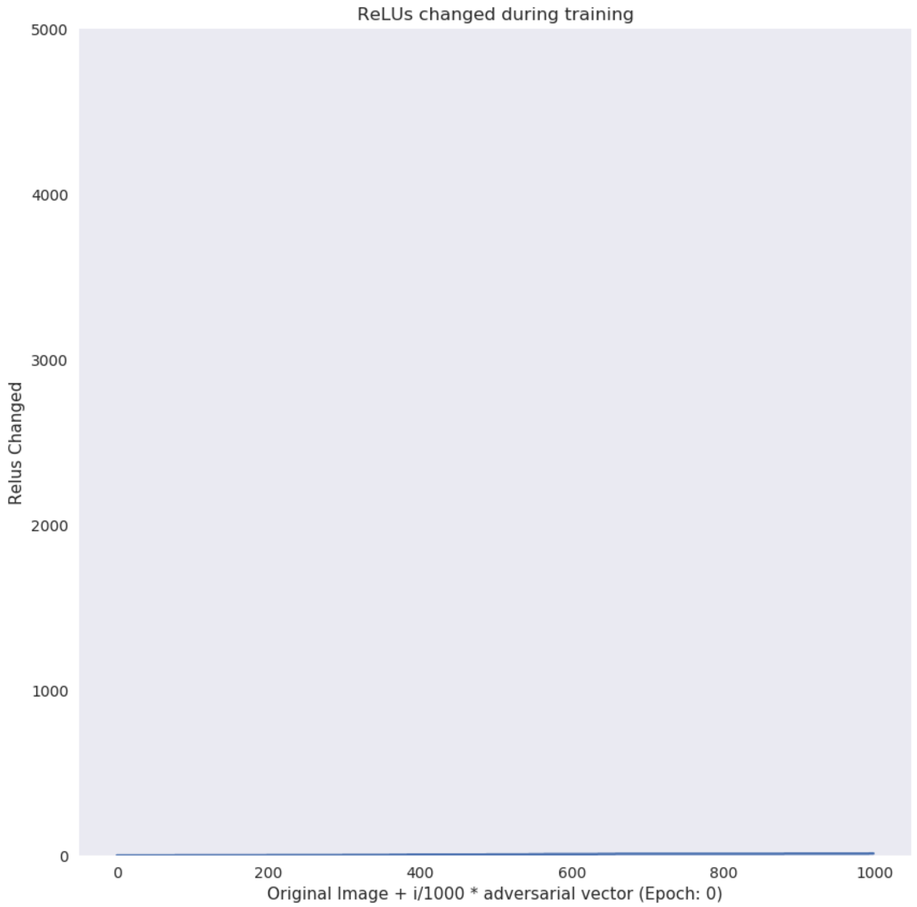
and $\mathbb{D}_{+,\mathbf{e}_d}^{\lambda}$ *is the right-sided Riemann-Liouiville Fractional Derivative of order* λ*, in the direction of the basis vector* $\mathbf{e}_d$*.*

Future Work:

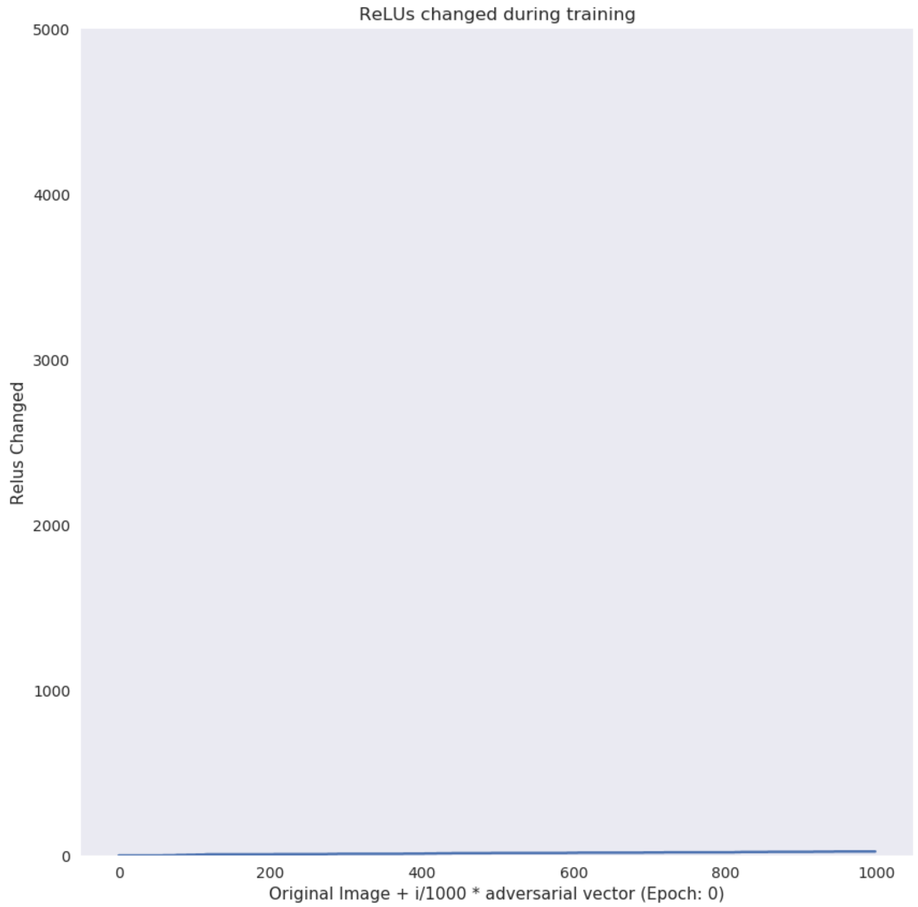
Developing Probing & Visualization Tools
To better understand and improve DL

Breakpoint Dynamics during Training: Random vs. Adversarial (*Targeted*) Directions

Random Direction BPs

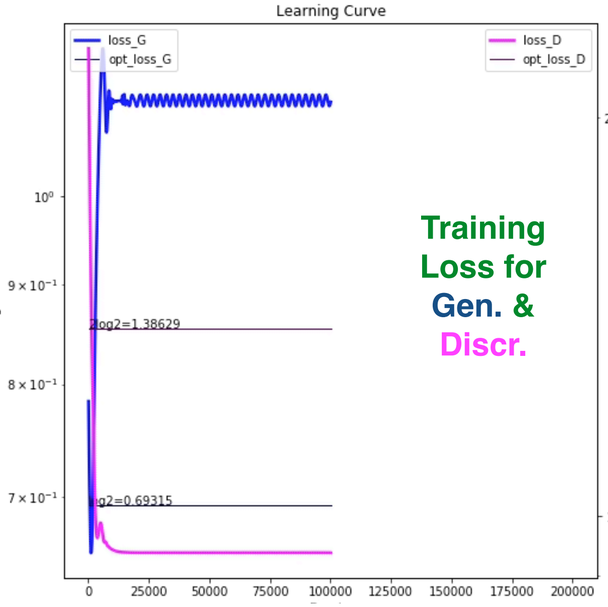


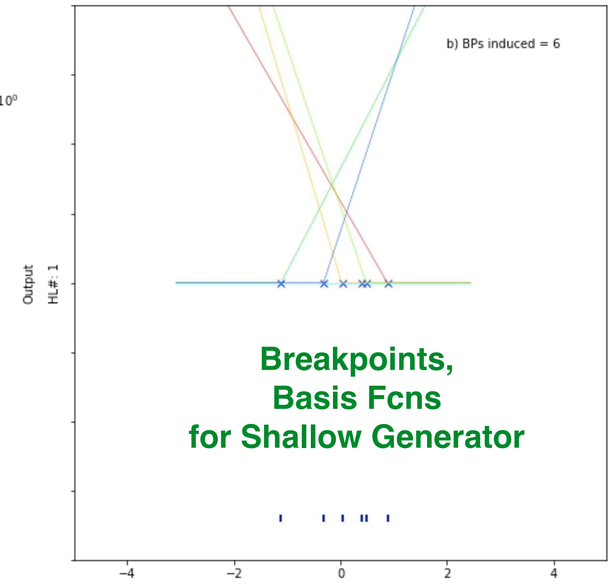
Adversarial Direction BPs

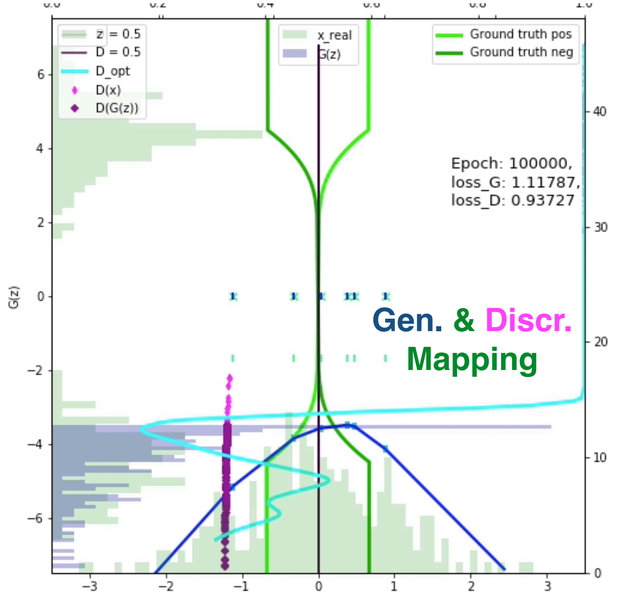


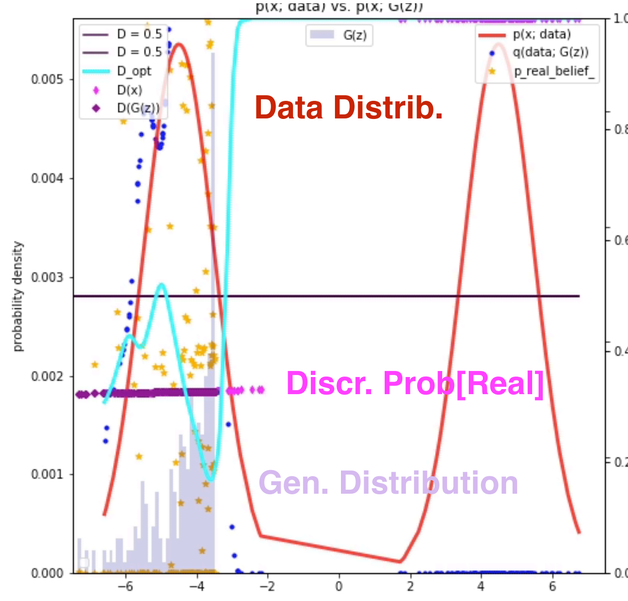
Learning Dynamics in GANs: Deeper Discriminators mitigate Mode Collapse

Shallow Discriminator(1,8,1)



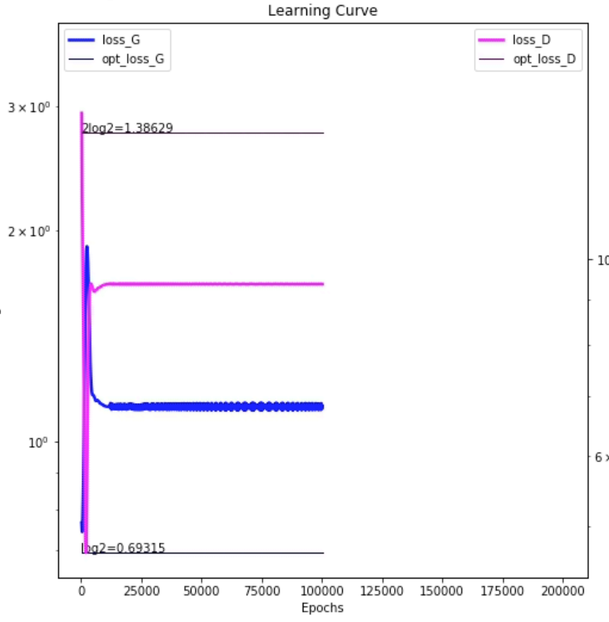


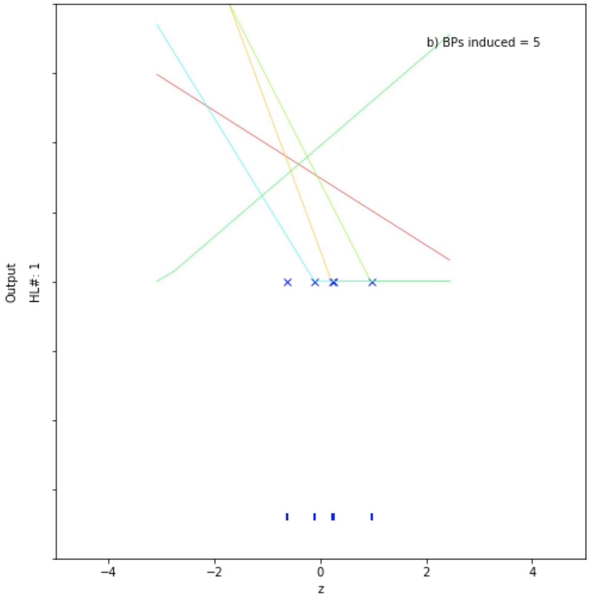


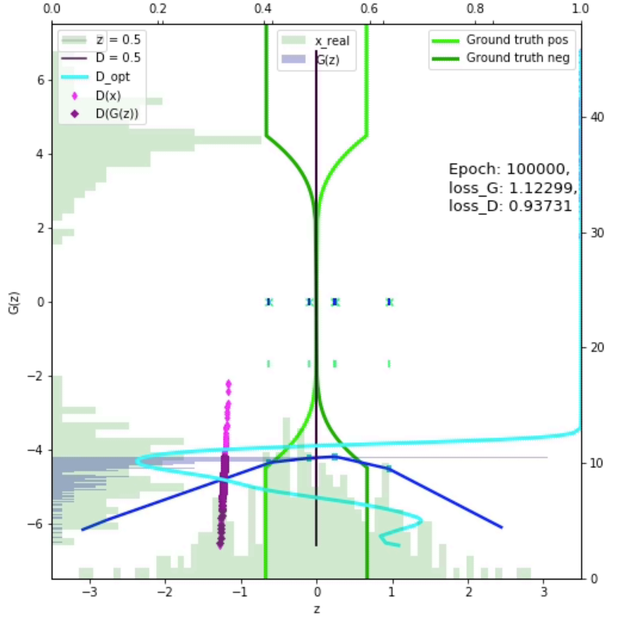


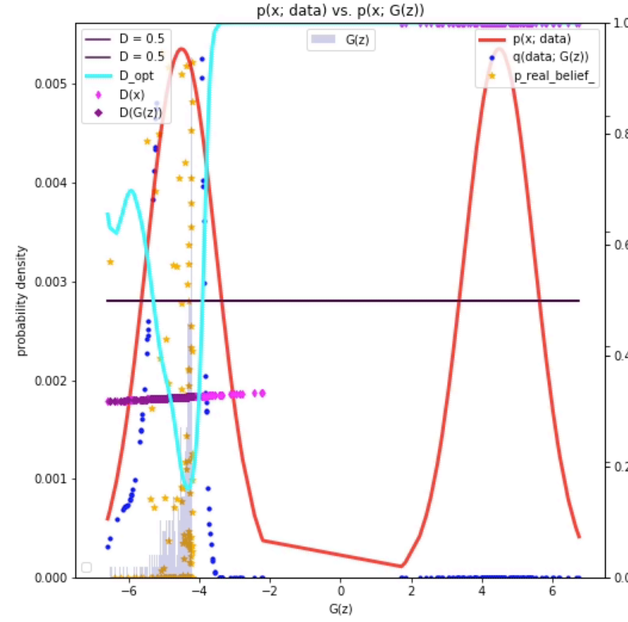
DeepReLU(1, 8, 1), D(1, 16, 16, 1), Opt=GD, Pi[0.5_0.5]_Mu[-4.5_ 4.5]_Sigma2[1._1.], NSamples: 256, NIters: 200000, lrG: 0.0001, lrD: 0.0001
torchSeed: 0, Htot: 8, NHiddens: 1, HtotD: 32, NHiddensD: 2, kdeSigma2: 0.2, noiseSigma2: 0.0, releps: 0.000, rKL: False, sinit: True, gStep: 1, batchNorm: False, NBends: 0.0, fl: False, nn

Deep Discriminator(1,16,16,1)

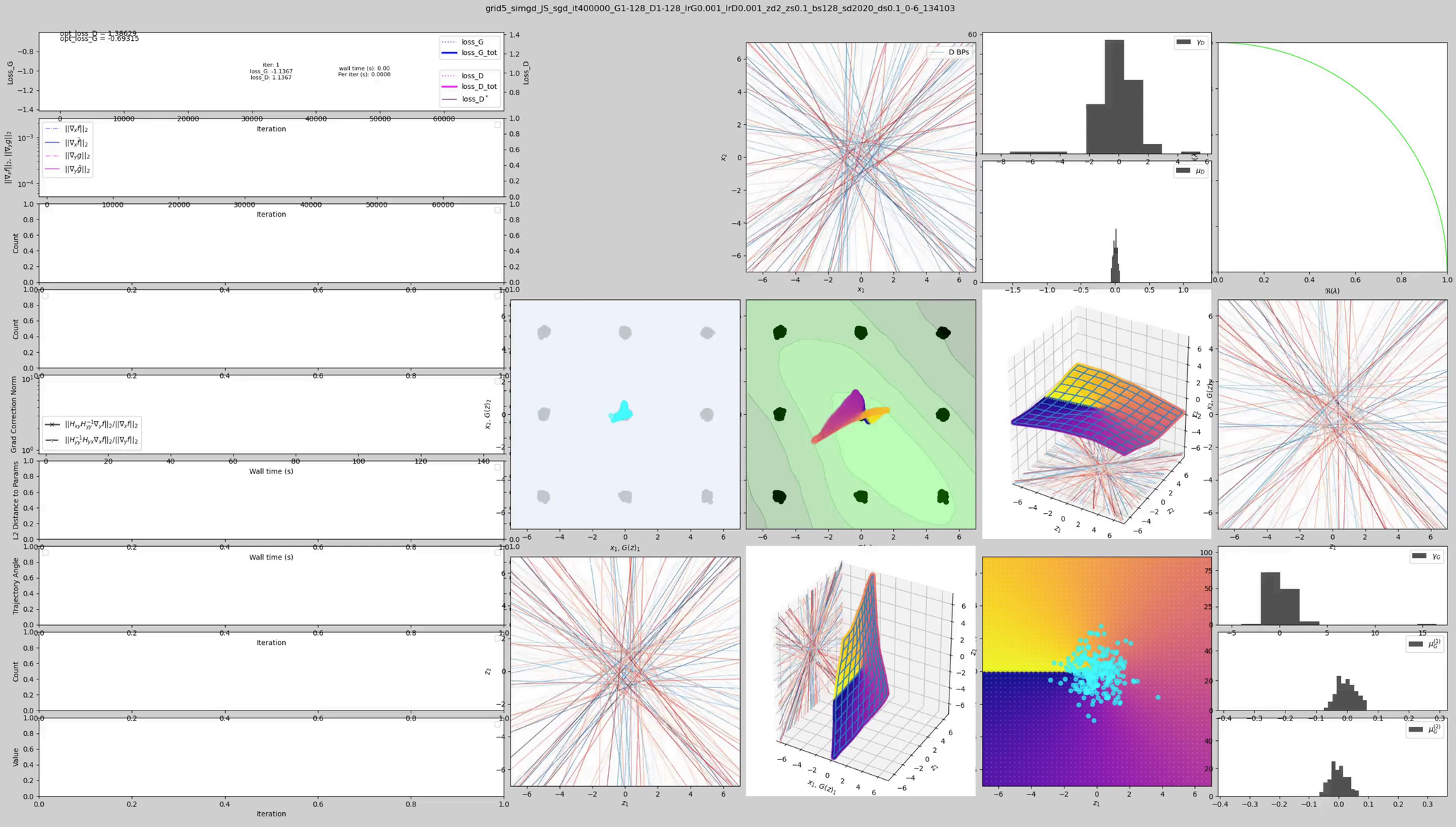




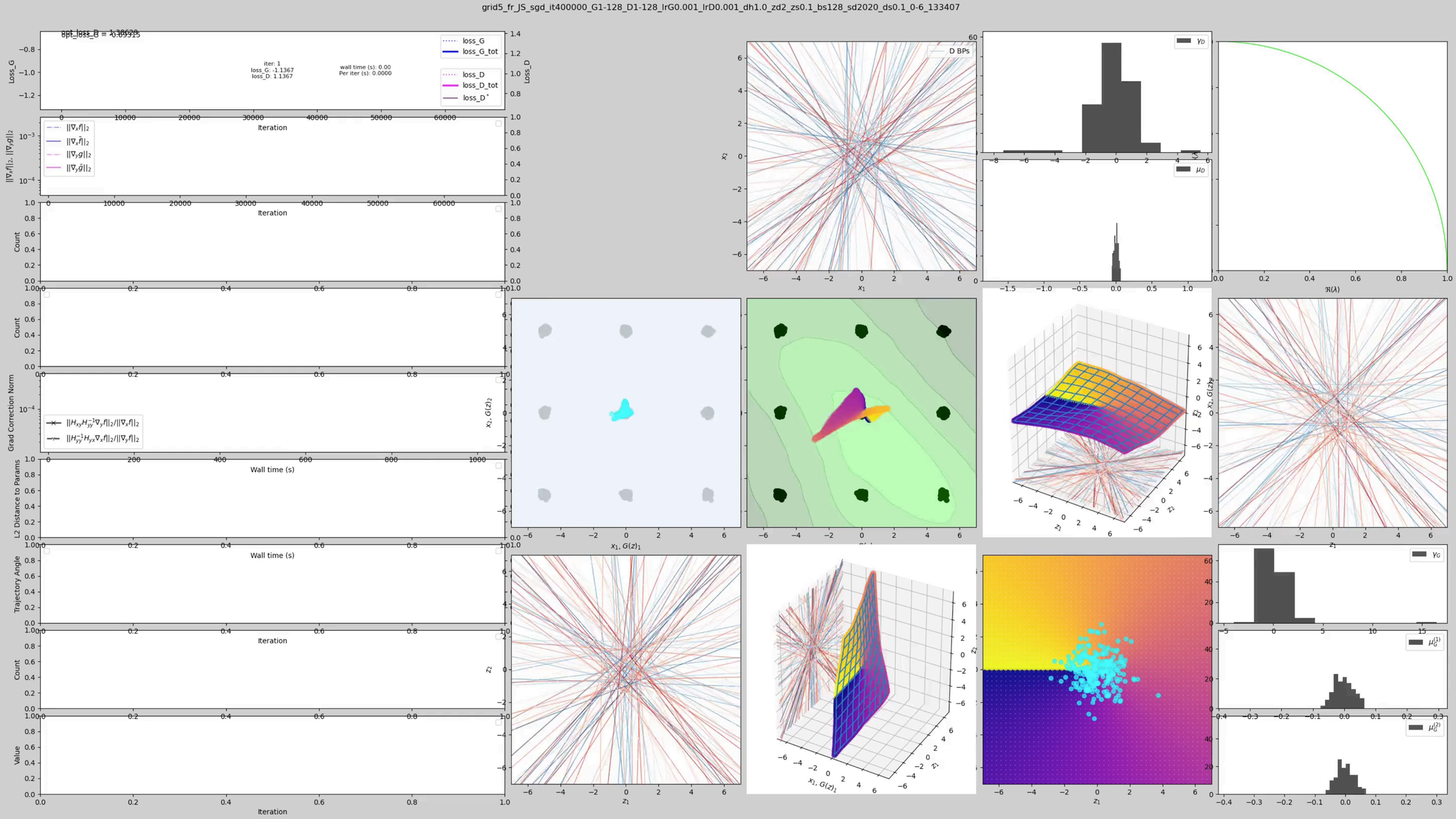




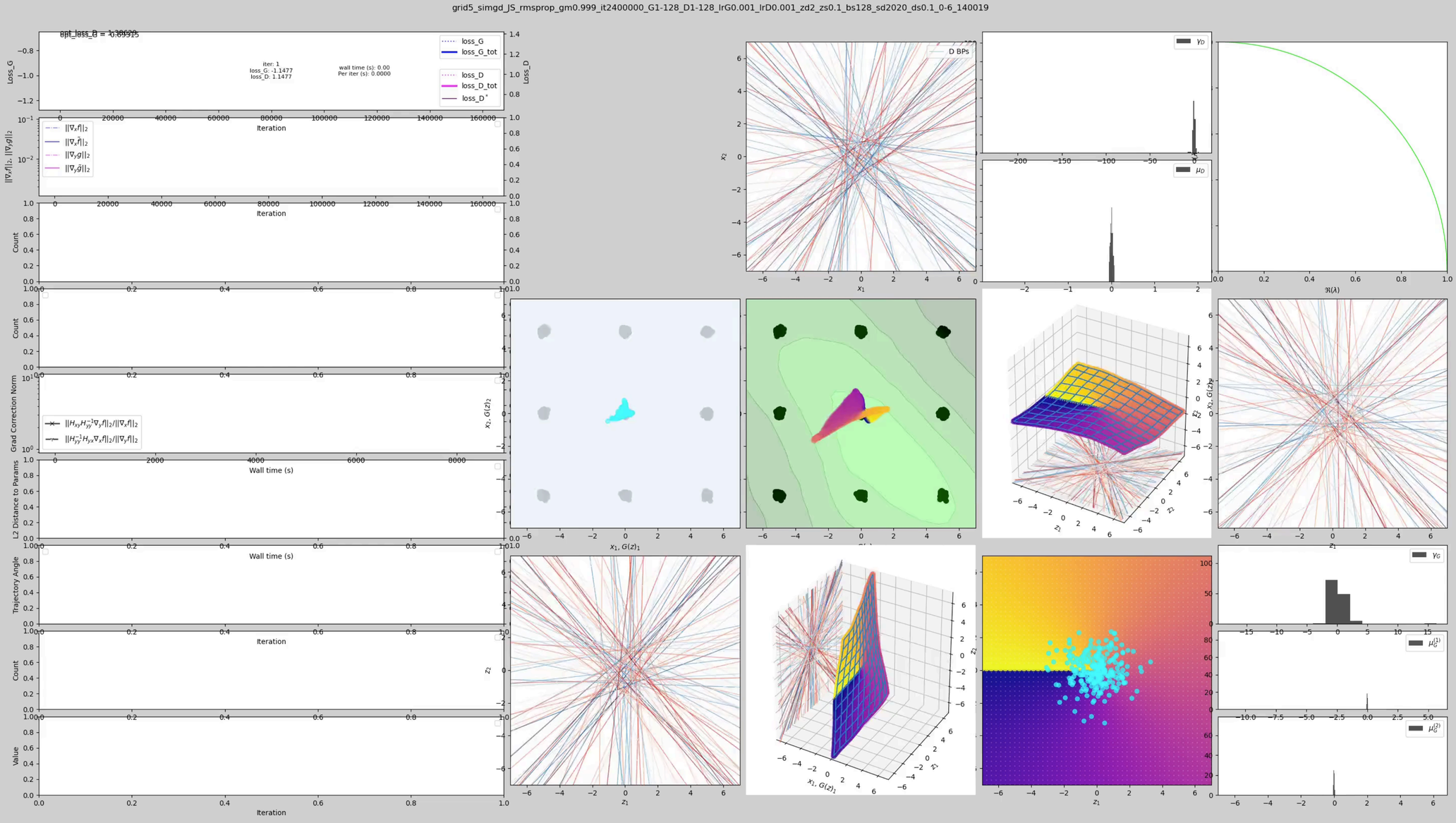
Learning Dynamics in GANs: Implicit Regularization makes it difficult for SimGD algorithm to approximate Discontinuities —> Mode Collapse



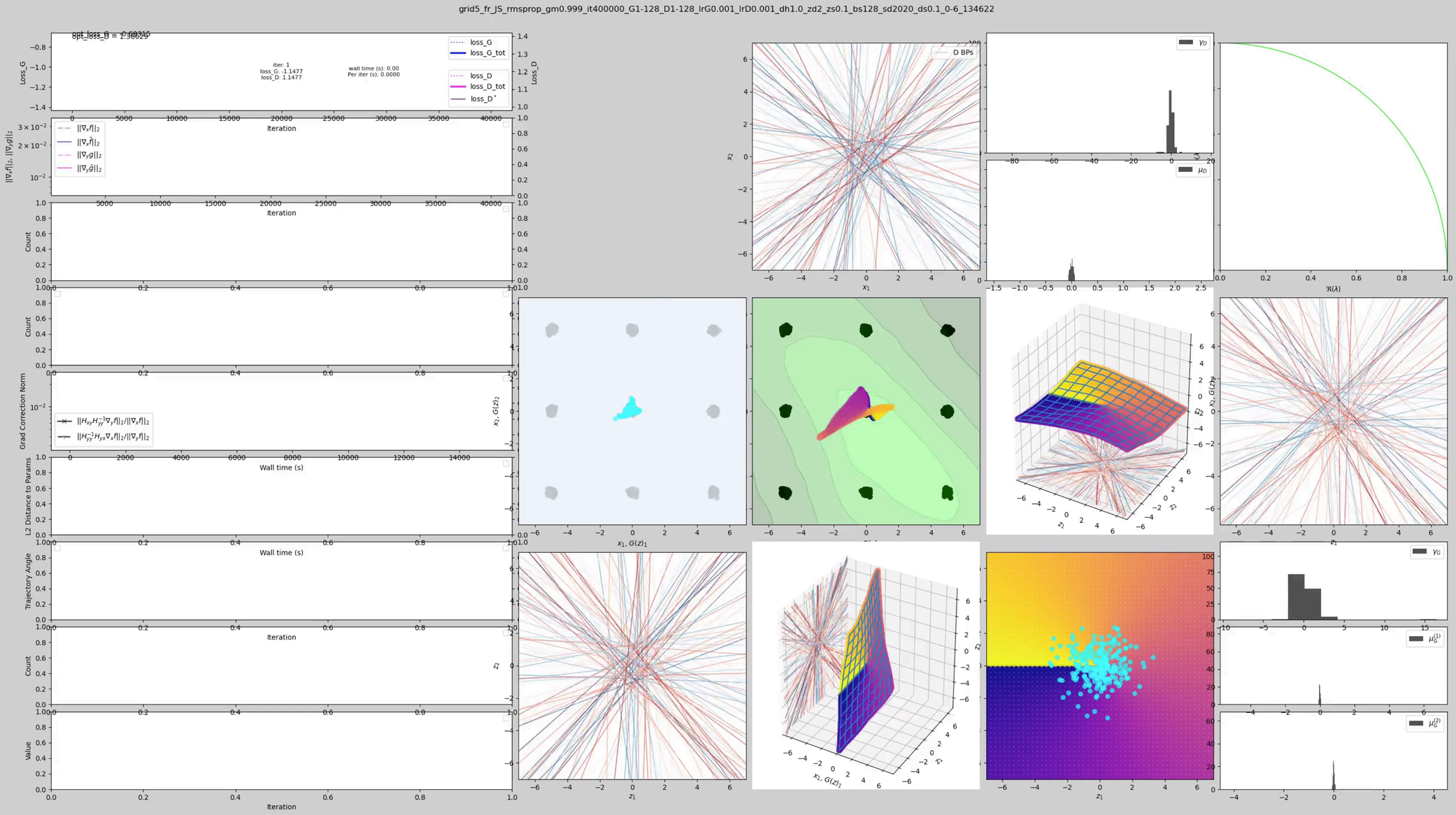
Learning Dynamics in GANs: Switching to a minimax-guaranteed algorithm (Follow-the-Ridge) helps improve mode coverage but still not good enough…



Learning Dynamics in GANs: Adding a Preconditioner not only helps SimGD optimization/training speed
But more importantly it induced implicit regularization to be more adaptive…



Learning Dynamics in GANs: Combining MiniMax + Preconditioning together —> Adaptive Regularization —> Discontinuities approximated more sharply and quickly —> greatly improved mode coverage very early on.



Thanks!

- DL is a ***powerful tool for approximating functions*** which are too complex to specify directly but can instead by specified by many input-output examples…
  - … ***BUT it is not a magical blackbox*** — sometimes it fails badly. We must develop a theory that specifies the underlying modeling assumptions. We are also studying the brain to see how to alleviate many current limitations.
  - ***We can and must combine principled domain knowledge with the flexibility of DL***, in order to achieve ***generalization/extrapolation, low sample complexity, few and interpretable parameters***.
-

Thanks!

- **Future Directions:**
  - Univariate Deep
  - Multivariate BDSO Expansions (ongoing)
    - Other Activation Functions (sigmoid, step, saturating ReLu)
  - Neuronal Networks and Cell Types

Extra Slides

Calculating Gradients & Hessian of Loss Surface

Gradients & Critical Points of the Loss

(for Shallow Univariate ReLu NNs)

Gradients of Function/Residuals:

$$\begin{aligned}
\frac{\partial \hat{\mathbf{f}}}{\partial b_0} &= \frac{\partial}{\partial b_0}\left(\sum_{i=1}^{H} v_i \phi(w_i \mathbf{x} + b_i \mathbf{1}) + b_0 \mathbf{1}\right) \\
&= \mathbf{1} \\
\frac{\partial \hat{\mathbf{f}}}{\partial w_i} &= \frac{\partial}{\partial w_i}\left(\sum_{i=1}^{H} v_i \phi(w_i \mathbf{x} + b_i \mathbf{1}) + b_0 \mathbf{1}\right) \\
&= v_i \phi'(w_i \mathbf{x} + b_i \mathbf{1}) \odot \mathbf{x} \\
&= v_i \mathbf{x} \odot [\![w_i \mathbf{x} + b_i \mathbf{1} > 0]\!] \\
&= v_i \mathbf{x}_i \\
\frac{\partial \hat{\mathbf{f}}}{\partial v_i} &= \frac{\partial}{\partial v_i}\left(\sum_{i=1}^{H} v_i \phi(w_i \mathbf{x} + b_i \mathbf{1}) + b_0 \mathbf{1}\right) \\
&= \phi(w_i \mathbf{x} + b_i \mathbf{1}) \\
&= (w_i \mathbf{x} + b_i \mathbf{1}) \odot [\![w_i \mathbf{x} + b_i \mathbf{1} > 0]\!] \\
&= w_i \mathbf{x}_i + b_i \mathbf{1}_i \\
\frac{\partial \hat{\mathbf{f}}}{\partial b_i} &= \frac{\partial}{\partial b_i}\left(\sum_{i=1}^{H} v_i \phi(w_i \mathbf{x} + b_i \mathbf{1}) + b_0 \mathbf{1}\right) \\
&= v_i \phi'(w_i \mathbf{x} + b_i \mathbf{1}) \\
&= v_i [\![w_i \mathbf{x} + b_i \mathbf{1} > 0]\!] \\
&= v_i \mathbf{1}_i
\end{aligned}$$

Gradients of Loss:

$$\begin{aligned}
\ell(\mathbf{x}) &= \frac{1}{2}\langle \hat{\epsilon}, \hat{\epsilon} \rangle \\
\frac{\partial \ell}{\partial \varphi} &= \langle \hat{\epsilon}, \frac{\partial}{\partial \varphi} \hat{\epsilon} \rangle \\
&= -\langle \hat{\epsilon}, \frac{\partial}{\partial \varphi} \hat{\mathbf{f}} \rangle \\
\frac{\partial \ell}{\partial b_0} &= -\langle \hat{\epsilon}, \mathbf{1} \rangle \\
\frac{\partial \ell}{\partial w_i} &= -\langle \hat{\epsilon}, v_i \mathbf{x}_i \rangle \\
\frac{\partial \ell}{\partial v_i} &= -\langle \hat{\epsilon}, w_i \mathbf{x}_i + b_i \mathbf{1}_i \rangle \\
\frac{\partial \ell}{\partial b_i} &= -\langle \hat{\epsilon}, v_i \mathbf{1}_i \rangle
\end{aligned}$$

Calculating the Hessian of the Loss

(for Shallow Univariate ReLu NNs)

Full Hessian (with Dirac Delta Function terms):

$$\mathbf{H}_\ell \triangleq \mathrm{Hess}(\ell) \triangleq \begin{pmatrix} \ddots & & & & & \\ & \frac{\partial^2 \ell}{\partial w_j \partial w_i} & \frac{\partial^2 \ell}{\partial w_j \partial v_i} & \frac{\partial^2 \ell}{\partial w_j \partial b_i} & \cdots & \frac{\partial^2 \ell}{\partial w_j \partial b_0} \\ & \frac{\partial^2 \ell}{\partial v_j \partial w_i} & \frac{\partial^2 \ell}{\partial v_j \partial v_i} & \frac{\partial^2 \ell}{\partial v_j \partial b_i} & \cdots & \frac{\partial^2 \ell}{\partial v_j \partial b_0} \\ & \frac{\partial^2 \ell}{\partial b_j \partial w_i} & \frac{\partial^2 \ell}{\partial b_j \partial v_i} & \frac{\partial^2 \ell}{\partial b_j \partial b_i} & \cdots & \frac{\partial^2 \ell}{\partial b_j \partial b_0} \\ & \vdots & & & \ddots & \vdots \\ & \frac{\partial^2 \ell}{\partial b_0 \partial w_i} & \frac{\partial^2 \ell}{\partial b_0 \partial v_i} & \frac{\partial^2 \ell}{\partial b_0 \partial b_i} & \cdots & \frac{\partial^2 \ell}{\partial b_0^2} \end{pmatrix}$$

$$= \begin{pmatrix} \ddots & & & & \\ & \langle v_j \mathbf{x}_j, v_i \mathbf{x}_i \rangle - \langle \hat{\epsilon}, \delta_{ij} v_i \mathbf{x}_{i,=}^{\odot 2} \rangle & \langle v_j \mathbf{x}_j, w_i \mathbf{x}_i + b_i \mathbf{1}_i \rangle - \langle \hat{\epsilon}, \delta_{ij} \mathbf{x}_i \rangle & \langle v_j \mathbf{x}_j, v_i \mathbf{1}_i \rangle - \langle \hat{\epsilon}, \delta_{ij} v_i \mathbf{1}_{i,=} \rangle & \cdots & \langle v_j \mathbf{x}_j, \mathbf{1} \rangle \\ & \langle w_j \mathbf{x}_j + b_j \mathbf{1}_j, v_i \mathbf{x}_i \rangle - \langle \hat{\epsilon}, \delta_{ij} \mathbf{x}_i \rangle & \langle w_j \mathbf{x}_j + b_j \mathbf{1}_j, w_i \mathbf{x}_i + b_i \mathbf{1}_i \rangle & \langle w_j \mathbf{x}_j + b_j \mathbf{1}_j, v_i \mathbf{1}_i \rangle - \langle \hat{\epsilon}, \delta_{ij} \mathbf{1}_i \rangle & \cdots & \langle w_j \mathbf{x}_j + b_j \mathbf{1}_j, \mathbf{1} \rangle \\ & \langle v_j \mathbf{1}_j, v_i \mathbf{x}_i \rangle - \langle \hat{\epsilon}, \delta_{ij} v_i \mathbf{x}_{i,=} \rangle & \langle v_j \mathbf{1}_j, w_i \mathbf{x}_i + b_i \mathbf{1}_i \rangle - \langle \hat{\epsilon}, \delta_{ij} \mathbf{1}_i \rangle & \langle v_j \mathbf{1}_j, v_i \mathbf{1}_i \rangle - \langle \hat{\epsilon}, \delta_{ij} \mathbf{1}_{i,=} \rangle & \cdots & \langle v_j \mathbf{1}_j, \mathbf{1} \rangle \\ & \vdots & & & \ddots & \vdots \\ & \langle \mathbf{1}, v_i \mathbf{x}_i \rangle & \langle \mathbf{1}, w_i \mathbf{x}_i + b_i \mathbf{1}_i \rangle & \langle \mathbf{1}, v_i \mathbf{1}_i \rangle & \cdots & \langle \mathbf{1}, \mathbf{1} \rangle \end{pmatrix}$$

Simplifying the Hessian of the Loss

(for Shallow Univariate ReLu NNs)

Assuming Datapoints and Breakpoints don't Coincide…
(this excludes Dirac Delta Function terms):

Assume $\beta_i \neq x_n$ for all i, n (i.e. exclude a set of measure 0)

$$= \begin{pmatrix} \ddots & & & & \\ \langle v_j\mathbf{x}_j, v_i\mathbf{x}_i\rangle & \langle v_j\mathbf{x}_j, w_i\mathbf{x}_i + b_i\mathbf{1}_i\rangle - \langle \hat{\epsilon}, \delta_{ij}\mathbf{x}_i\rangle & \langle v_j\mathbf{x}_j, v_i\mathbf{1}_i\rangle & \cdots & \langle v_j\mathbf{x}_j, 1\rangle \\ \langle w_j\mathbf{x}_j + b_j\mathbf{1}_j, v_i\mathbf{x}_i\rangle - \langle \hat{\epsilon}, \delta_{ij}\mathbf{x}_i\rangle & \langle w_j\mathbf{x}_j + b_j\mathbf{1}_j, w_i\mathbf{x}_i + b_i\mathbf{1}_i\rangle & \langle w_j\mathbf{x}_j + b_j\mathbf{1}_j, v_i\mathbf{1}_i\rangle - \langle \hat{\epsilon}, \delta_{ij}\mathbf{1}_i\rangle & \cdots & \langle w_j\mathbf{x}_j + b_j\mathbf{1}_j, 1\rangle \\ \langle v_j\mathbf{1}_j, v_i\mathbf{x}_i\rangle & \langle v_j\mathbf{1}_j, w_i\mathbf{x}_i + b_i\mathbf{1}_i\rangle - \langle \hat{\epsilon}, \delta_{ij}\mathbf{1}_i\rangle & \langle v_j\mathbf{1}_j, v_i\mathbf{1}_i\rangle & \cdots & \langle v_j\mathbf{1}_j, 1\rangle \\ \vdots & & & \ddots & \vdots \\ \langle \mathbf{1}, v_i\mathbf{x}_i\rangle & \langle \mathbf{1}, w_i\mathbf{x}_i + b_i\mathbf{1}_i\rangle & \langle \mathbf{1}, v_i\mathbf{1}_i\rangle & \cdots & \langle \mathbf{1}, 1\rangle \end{pmatrix}$$

Assuming we are at Critical Point…

At a critical point, i.e. $\langle \hat{\epsilon}_i, \mathbf{x}\rangle = \langle \hat{\epsilon}_i, \mathbf{1}\rangle = 0$:

Hessian is a Gram Matrix —> PSD iff generating vectors are Linearly Independent

$$= \begin{pmatrix} \ddots & & & & \\ \langle v_j\mathbf{x}_j, v_i\mathbf{x}_i\rangle & \langle v_j\mathbf{x}_j, w_i\mathbf{x}_i + b_i\mathbf{1}_i\rangle & \langle v_j\mathbf{x}_j, v_i\mathbf{1}_i\rangle & \cdots & \langle v_j\mathbf{x}_j, 1\rangle \\ \langle w_j\mathbf{x}_j + b_j\mathbf{1}_j, v_i\mathbf{x}_i\rangle & \langle w_j\mathbf{x}_j + b_j\mathbf{1}_j, w_i\mathbf{x}_i + b_i\mathbf{1}_i\rangle & \langle w_j\mathbf{x}_j + b_j\mathbf{1}_j, v_i\mathbf{1}_i\rangle & \cdots & \langle w_j\mathbf{x}_j + b_j\mathbf{1}_j, 1\rangle \\ \langle v_j\mathbf{1}_j, v_i\mathbf{x}_i\rangle & \langle v_j\mathbf{1}_j, w_i\mathbf{x}_i + b_i\mathbf{1}_i\rangle & \langle v_j\mathbf{1}_j, v_i\mathbf{1}_i\rangle & \cdots & \langle v_j\mathbf{1}_j, 1\rangle \\ \vdots & & & \ddots & \vdots \\ \langle \mathbf{1}, v_i\mathbf{x}_i\rangle & \langle \mathbf{1}, w_i\mathbf{x}_i + b_i\mathbf{1}_i\rangle & \langle \mathbf{1}, v_i\mathbf{1}_i\rangle & \cdots & \langle \mathbf{1}, 1\rangle \end{pmatrix}$$

Hessian of the Loss and Degenerate Directions

(for Shallow Univariate ReLu NNs)

Simplified Hessian is a Gram Matrix—> Pos.Semi-Def….

$$= \begin{pmatrix} \ddots & & \ddots \\ \langle v_i\mathbf{x}_i, v_j\mathbf{x}_j\rangle & \langle w_i\mathbf{x}_i + b_i\mathbf{1}_i, v_j\mathbf{x}_j\rangle & \langle v_i\mathbf{1}_i, v_j\mathbf{x}_j\rangle \cdots \langle \mathbf{1}, v_j\mathbf{x}_j\rangle \\ \langle v_i\mathbf{x}_i, w_j\mathbf{x}_j + b_j\mathbf{1}_j\rangle & \langle w_i\mathbf{x}_i + b_i\mathbf{1}_i, w_j\mathbf{x}_j + b_j\mathbf{1}_j\rangle & \langle v_i\mathbf{1}_i, w_j\mathbf{x}_j + b_j\mathbf{1}_j\rangle \cdots \langle \mathbf{1}, w_j\mathbf{x}_j + b_j\mathbf{1}_j\rangle \\ \langle v_i\mathbf{x}_i, v_j\mathbf{1}_j\rangle & \langle w_i\mathbf{x}_i + b_i\mathbf{1}_i, v_j\mathbf{1}_j\rangle & \langle v_i\mathbf{1}_i, v_j\mathbf{1}_j\rangle \cdots \langle \mathbf{1}, v_j\mathbf{1}_j\rangle \\ \vdots & & \vdots \\ \langle v_i\mathbf{x}_i, v_i\mathbf{x}_i\rangle & \langle w_i\mathbf{x}_i + b_i\mathbf{1}_i, v_i\mathbf{x}_i\rangle & \langle v_i\mathbf{1}_i, v_i\mathbf{x}_i\rangle \\ \langle v_i\mathbf{x}_i, w_i\mathbf{x}_i + b_i\mathbf{1}_i\rangle & \langle w_i\mathbf{x}_i + b_i\mathbf{1}_i, w_i\mathbf{x}_i + b_i\mathbf{1}_i\rangle & \langle v_i\mathbf{1}_i, w_i\mathbf{x}_i + b_i\mathbf{1}_i\rangle \\ \langle v_i\mathbf{x}_i, v_i\mathbf{1}_i\rangle & \langle w_i\mathbf{x}_i + b_i\mathbf{1}_i, v_i\mathbf{1}_i\rangle & \langle v_i\mathbf{1}_i, v_i\mathbf{1}_i\rangle \\ \vdots & & \vdots \\ \langle \mathbf{1}, v_i\mathbf{x}_i\rangle & \langle \mathbf{1}, w_i\mathbf{x}_i + b_i\mathbf{1}_i\rangle & \langle \mathbf{1}, v_i\mathbf{1}_i\rangle \cdots \langle \mathbf{1}, \mathbf{1}\rangle \end{pmatrix}$$

…whose generating vectors are…

$$\{v_i\mathbf{x}_i, w_i\mathbf{x}_i + b_i\mathbf{1}_i, v_i\mathbf{1}_i\}_{i=1}^{H} \cup \{\mathbf{1}\}$$

Hessian is PSD and will be PD iff these vectors are Linearly Independent

—> Hessian will have 0 eigenvalues iff vectors are Linearly Dependent

Critical Points fall into a 6 types: Local minima (1), Degenerate (5)

Thus, except possibly at a set of measure zero (the $\beta_i = x_n$ event), $\mathbf{H}_\ell$ is the (positive semi-definite) Gram matrix of the set

$$\{v_i\mathbf{x}_i, w_i\mathbf{x}_i + b_i\mathbf{1}_i, v_i\mathbf{1}_i\}_{i=1}^{H} \cup \{\mathbf{1}\}$$

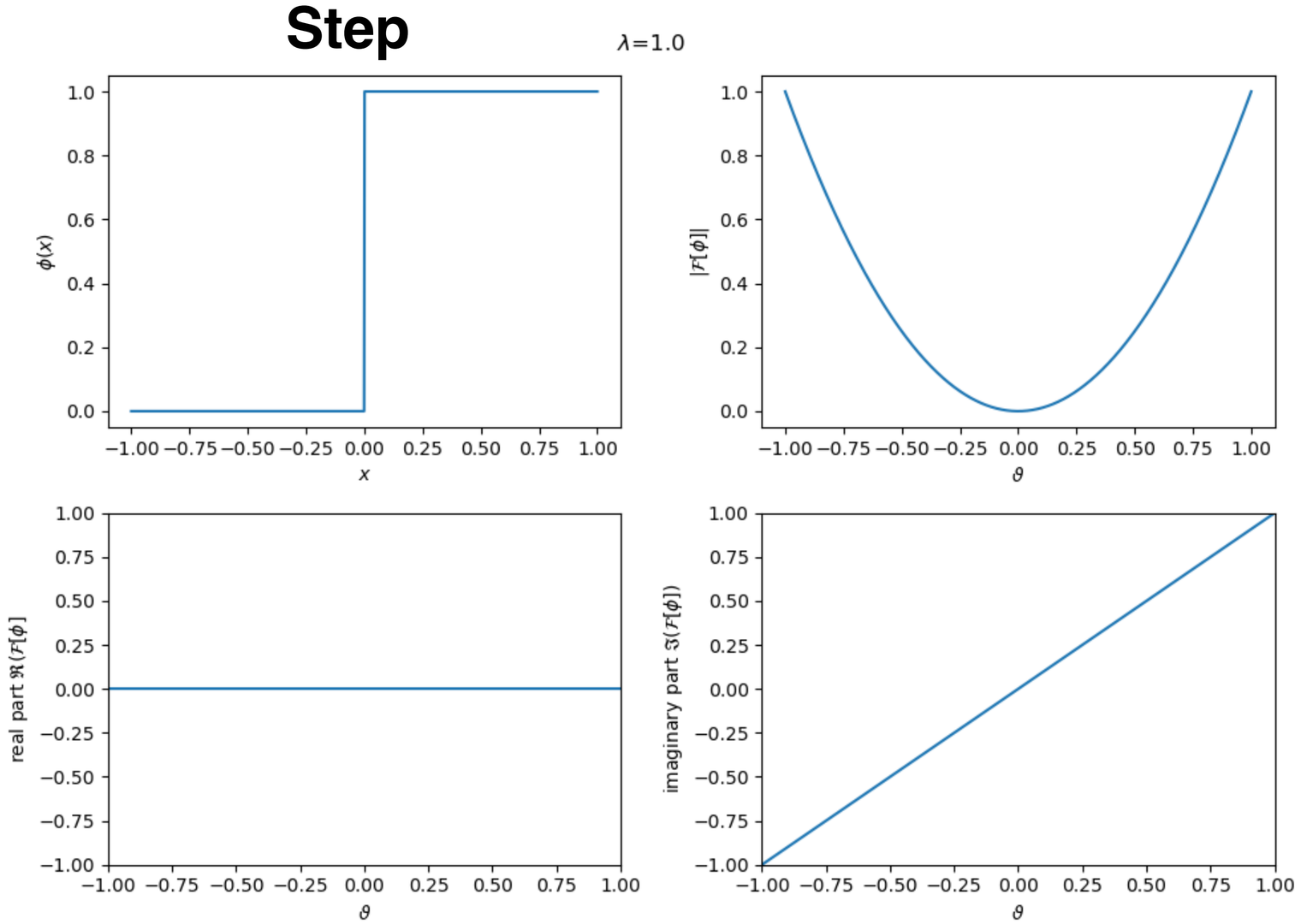
Then, $\mathbf{H}_\ell \succ 0$ iff the vectors of this set are linearly independent. This is the case *unless*

1. any neurons share activation patterns
   - the loss function doesn't care about what the function does between data points, so any change in one neuron that is "cancelled out" by other neuron(s) between it and the data leaves the loss unchanged
2. any neuron is active on the entire data
   - its bias is redundant with the global bias, leading to a 1-dimensional subspace of constant loss
3. any neuron is active on no data (i.e. $\mathbf{x}_i = \mathbf{1}_i = \mathbf{0}$)
   - The breakpoint will be facing away from the data, so any change of its parameters that doesn't move the breakpoint into the data will have no effect on the loss (v_i can change arbitrarily, w_i and b_i will have half-spaces of constant loss)
4. if for any i, $\mathbf{x}_i \propto \mathbf{1}_i$
   - i.e. $w_i\mathbf{x}_i + b_i\mathbf{1}_i \equiv \alpha_i\mathbf{x}_i \propto v_i\mathbf{x}_i \propto v_i\mathbf{1}_i$
   - e.g. if i has only one active data point or has multiple data points *all* with the same x-value
   - you can "rotate" the line segment through its value at x_n
5. some of (w_i, v_i, b_i) are 0
   - if $v_i = 0$, the delta-slope is 0, so the location of the breakpoint, and thus the values of w_i and b_i do not matter
   - if $w_i = 0$, the breakpoint is at infinity, so the values of v_i and b_i do not matter
   - if $b_i = 0$, changing w_i will not move the breakpoint, so w_i and v_i can change so long as w_iv_i remains constant

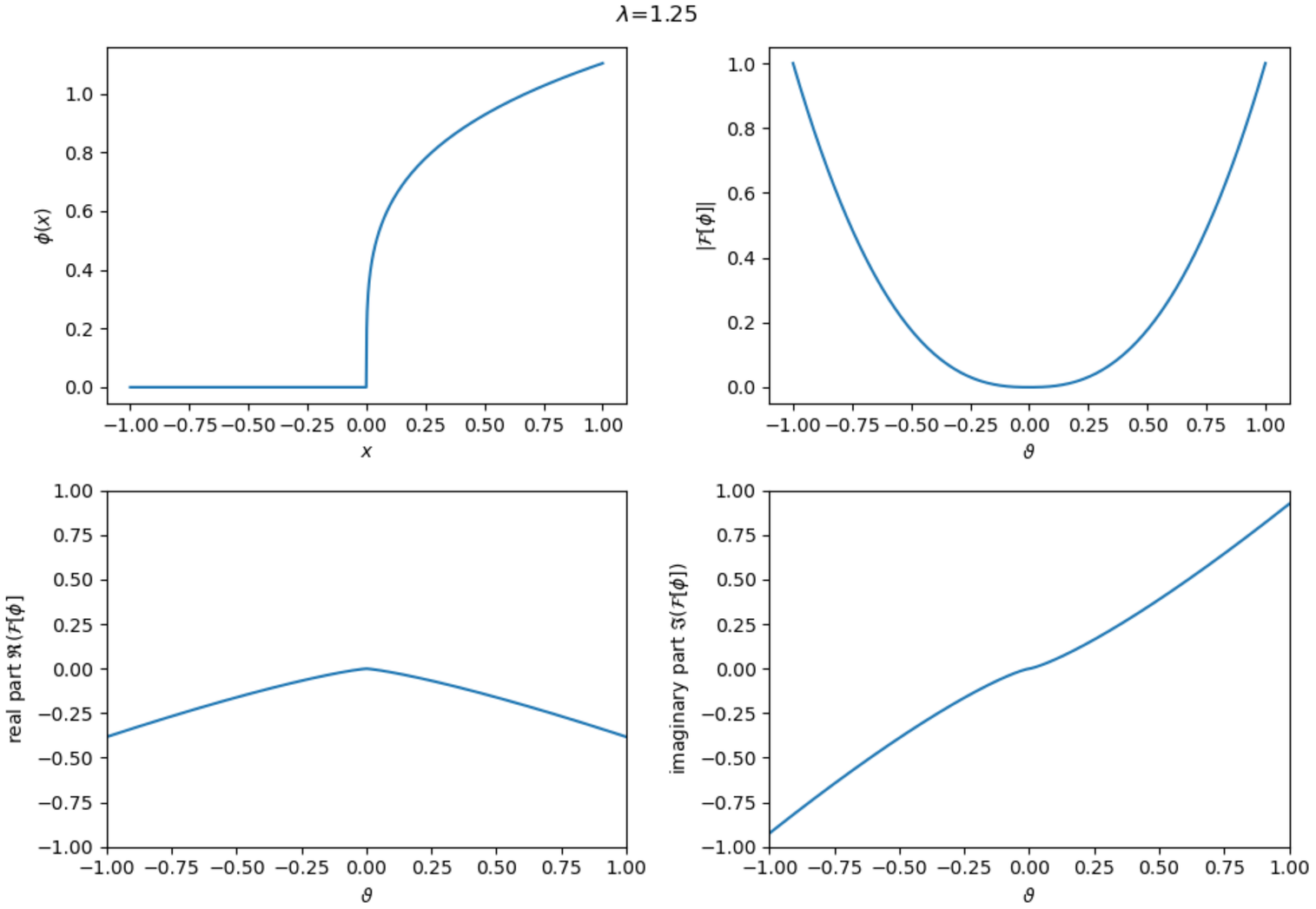
Implicit Reg. for Neuronal Networks with Saturating Response Functions

Generalizing to Other Activation Functions

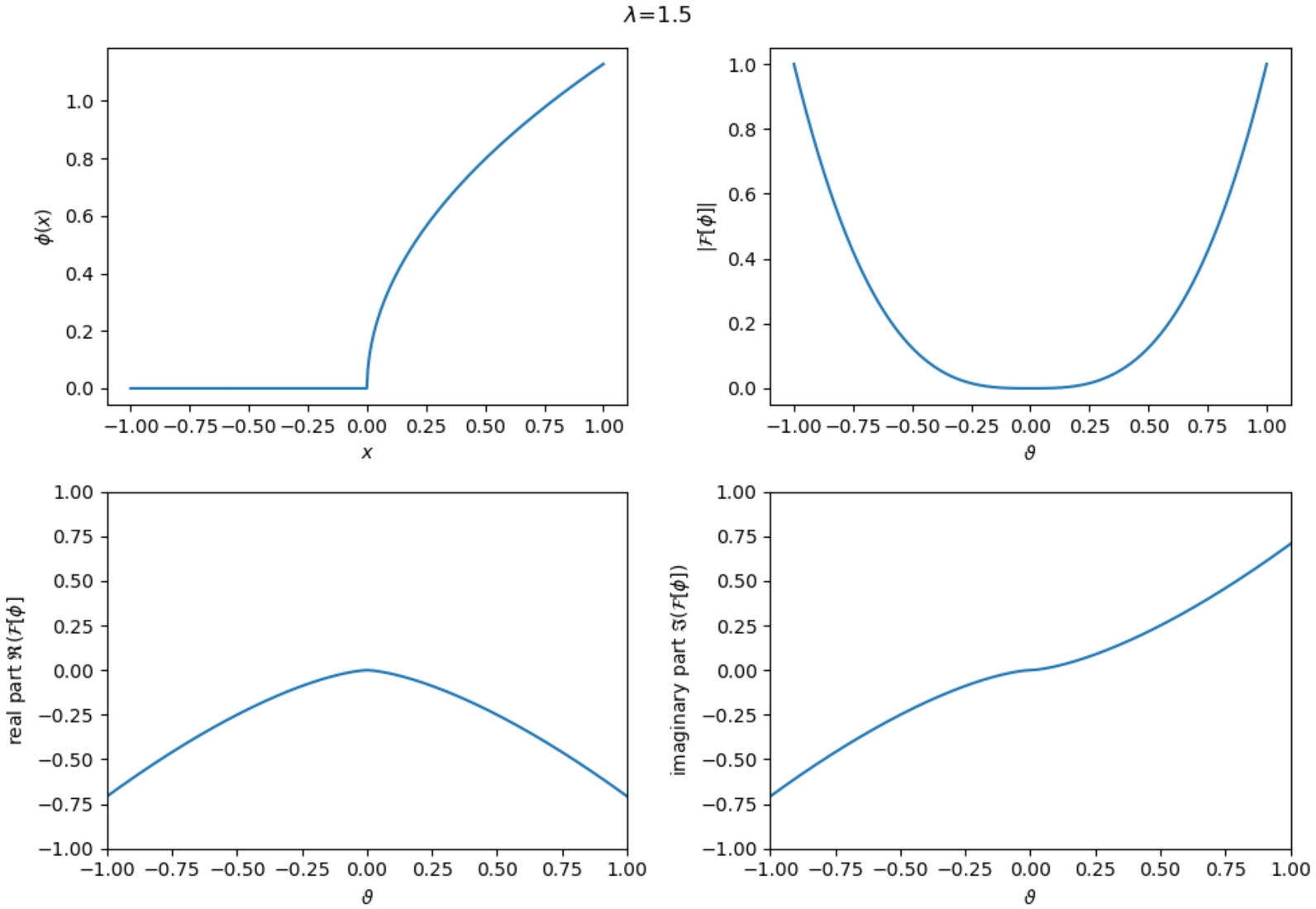
For General Non-Decreasing Activation Functions



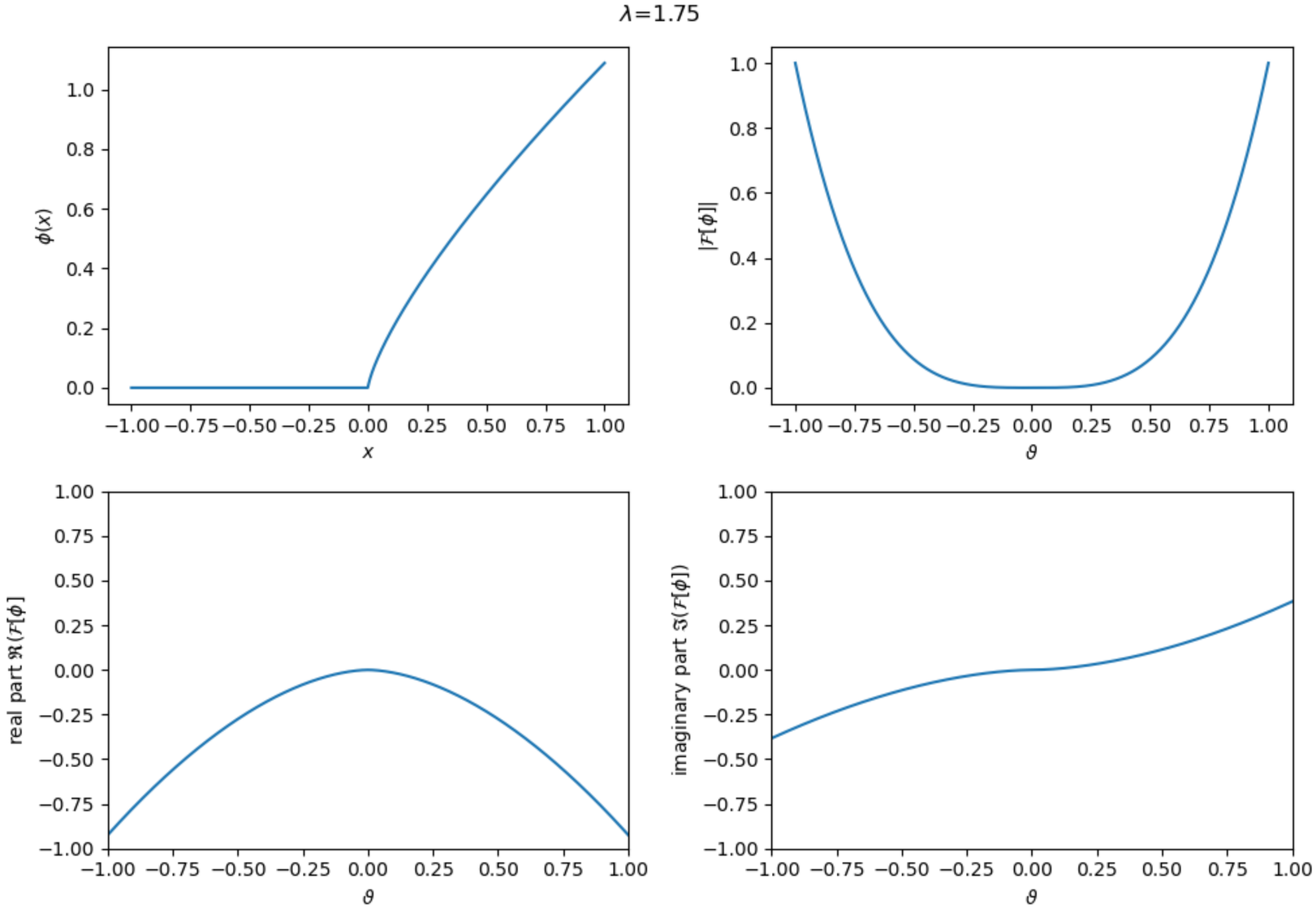
For General Non-Decreasing Activation Functions



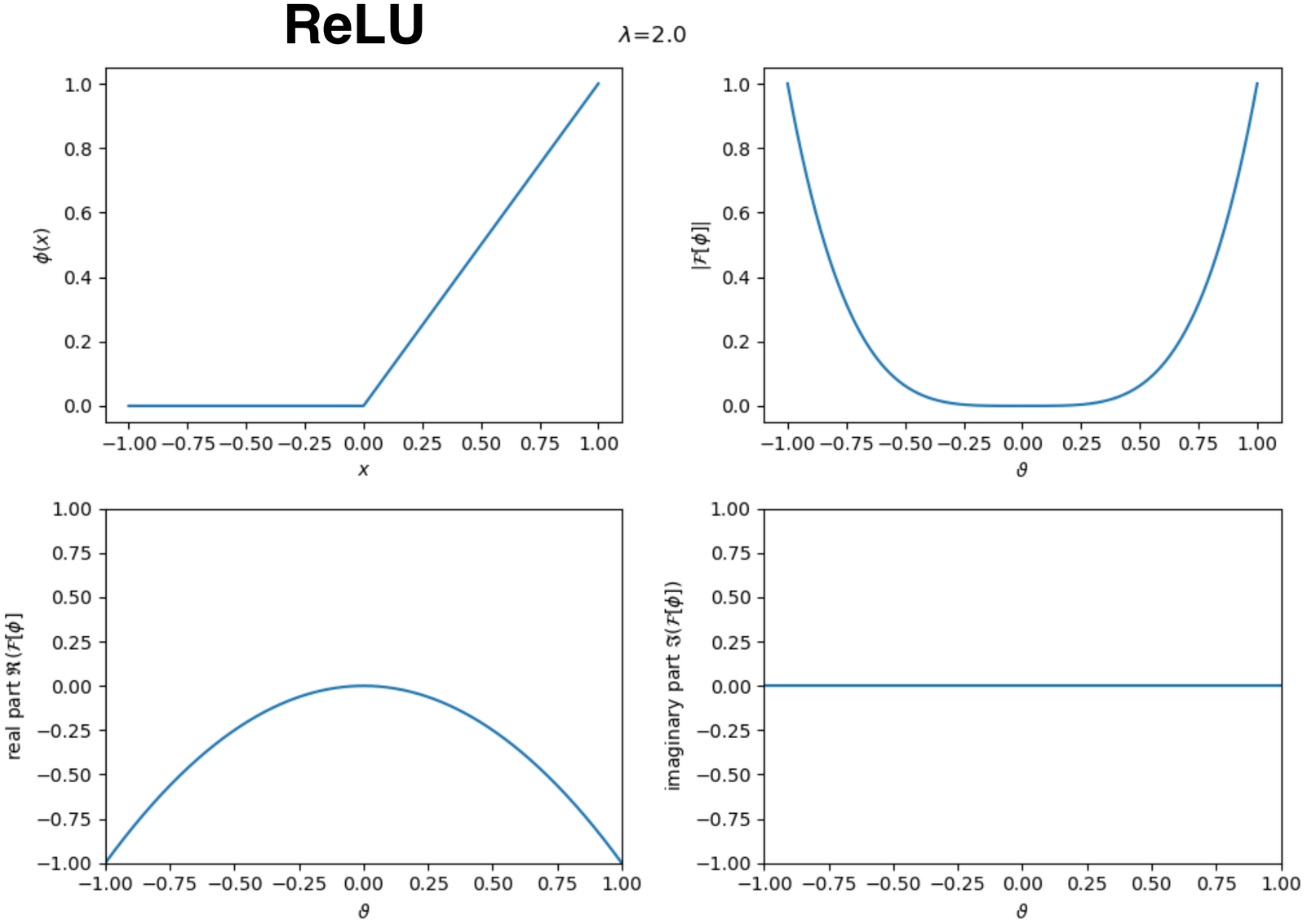
For General Non-Decreasing Activation Functions



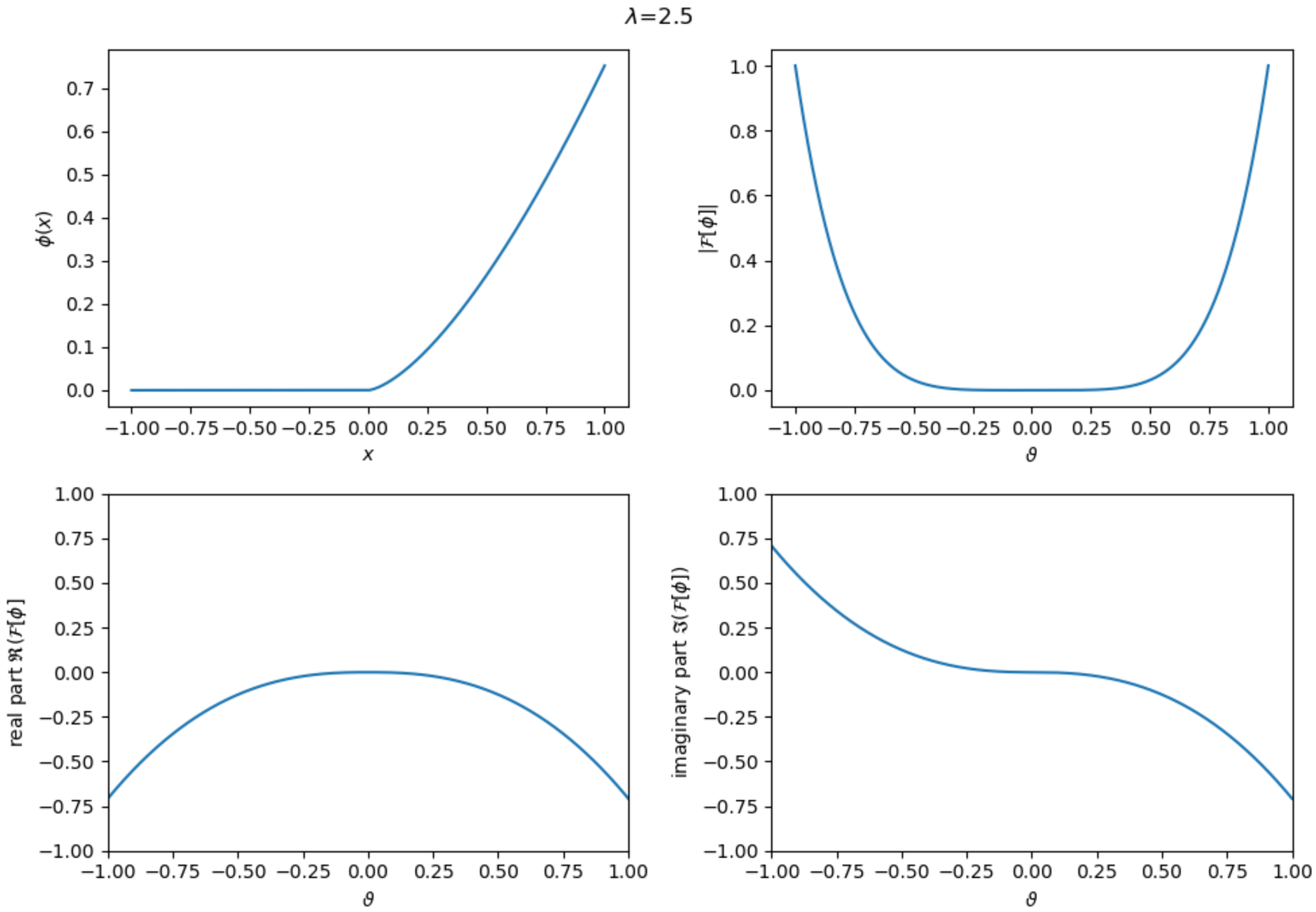
For General Non-Decreasing Activation Functions



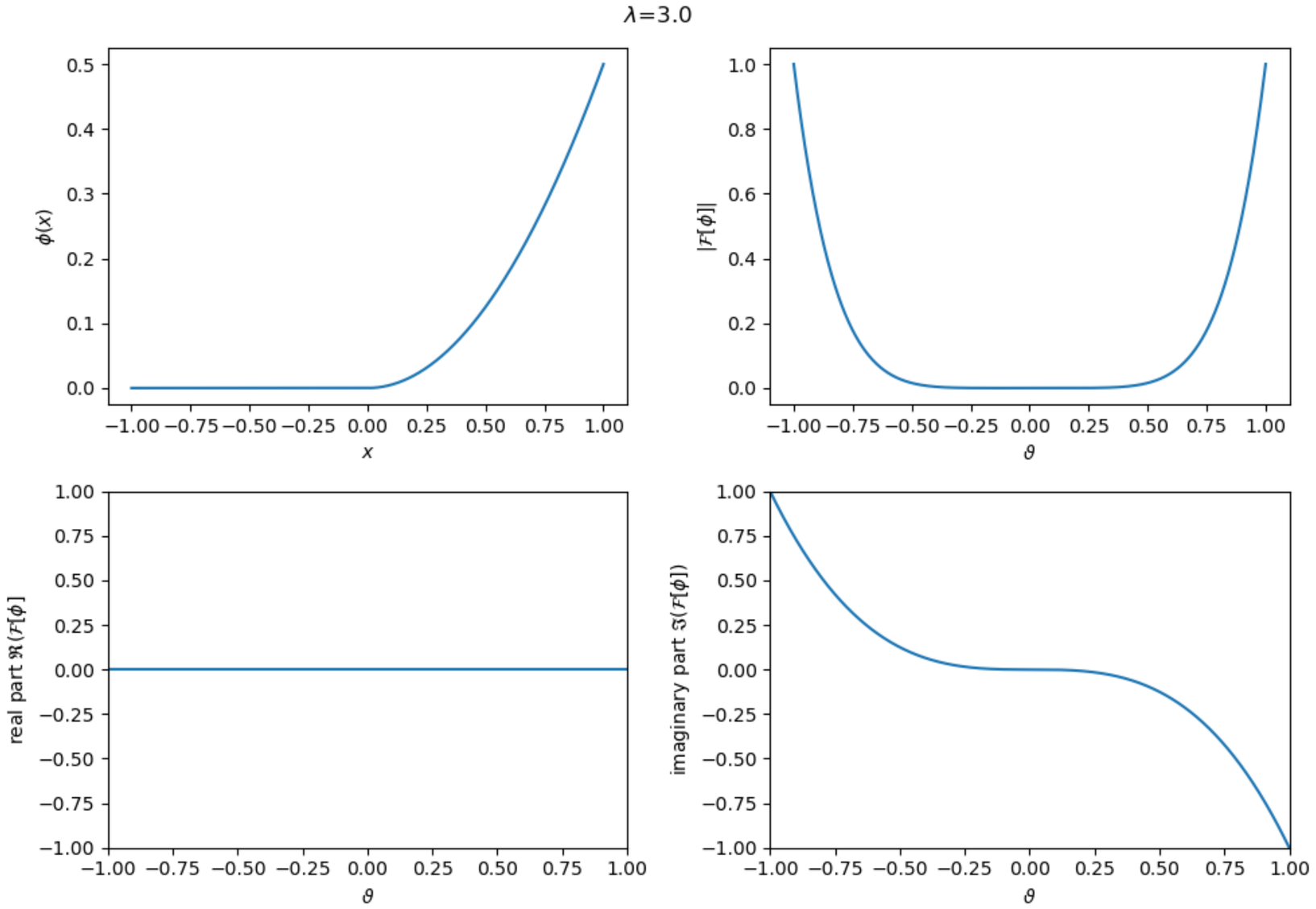
For General Non-Decreasing Activation Functions



For General Non-Decreasing Activation Functions

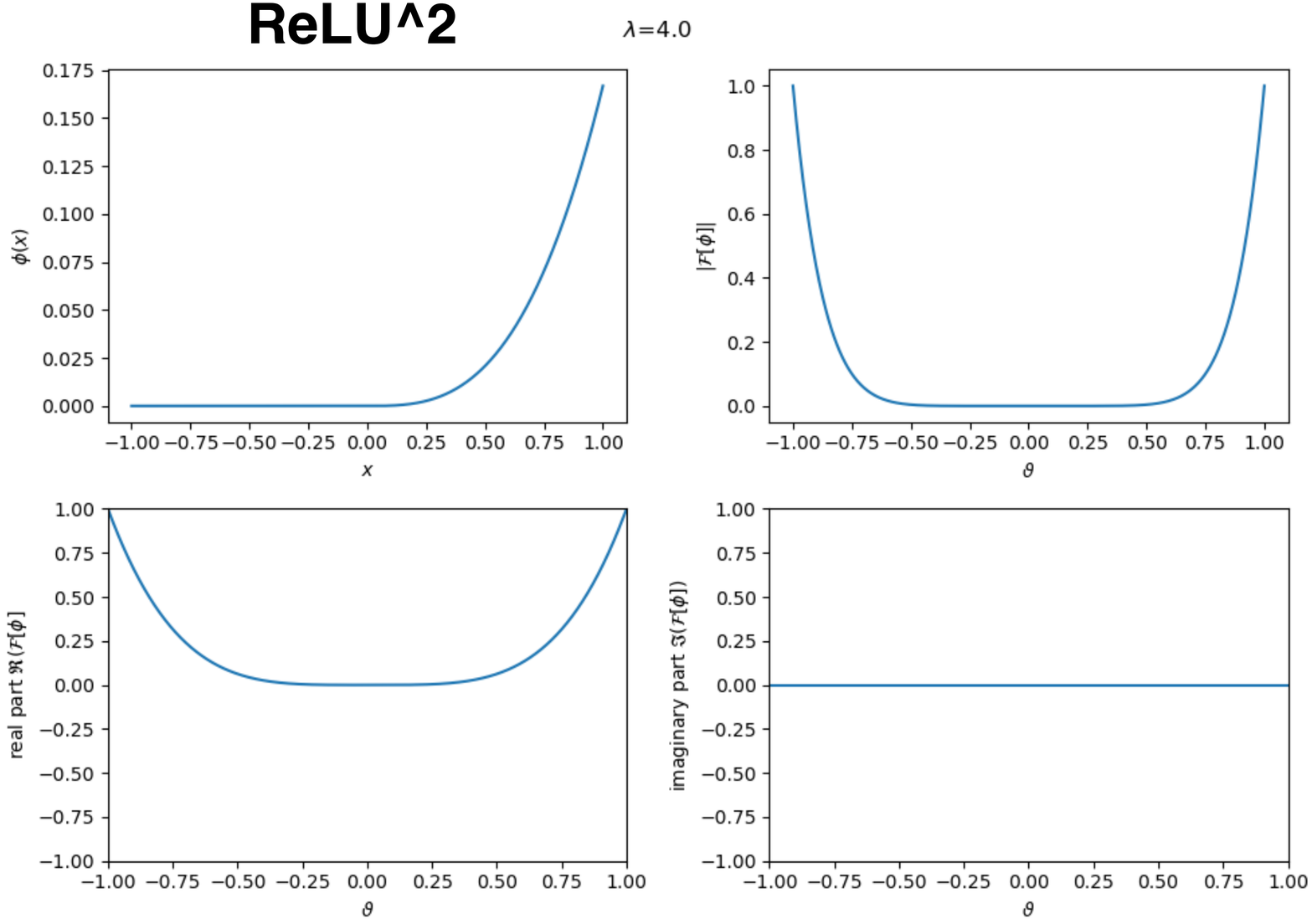


For General Non-Decreasing Activation Functions



For General Non-Decreasing Activation Functions

ReLU^2



Implicit Fourier Regularization is responsible for High-Frequency Low-Amplitude Adversarial Attacks