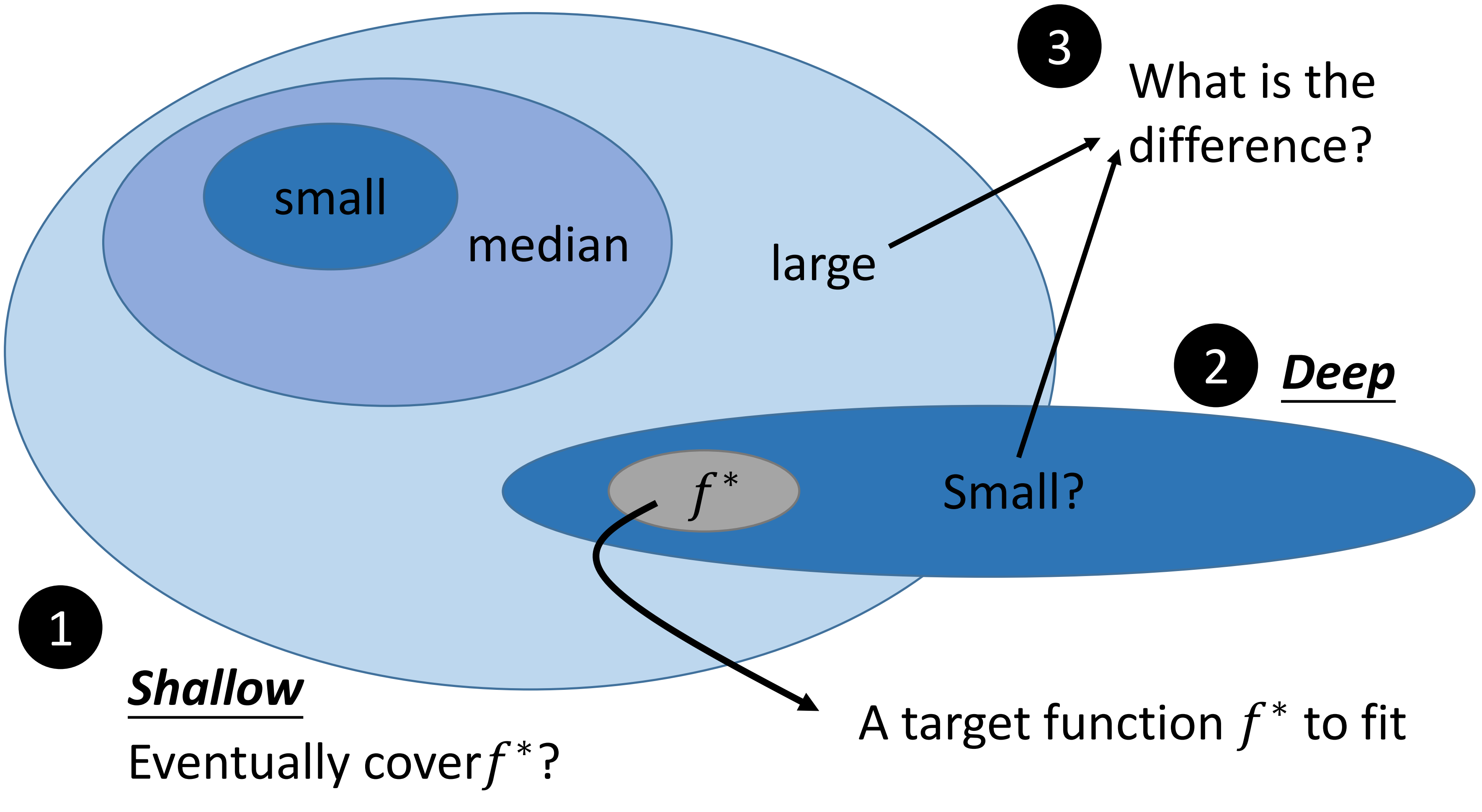
Optimization

李宏毅

Hung-yi Lee

Last time …

Optimization: Is it possible to find f^* in the function space.



Optimization

Optimization ≠ Learning

Network: $f_\theta(x)$

Training data:

$(x^1, \hat{y}^1)$

$(x^2, \hat{y}^2)$

$\vdots$

$(x^R, \hat{y}^R)$

$$L(\theta) = \sum_{r=1}^{R} l(f_\theta(x^r) - \hat{y}^r)$$

$$\theta^* = arg \min_\theta L(\theta)$$

In Deep Learning, $L(\theta)$ is not convex.

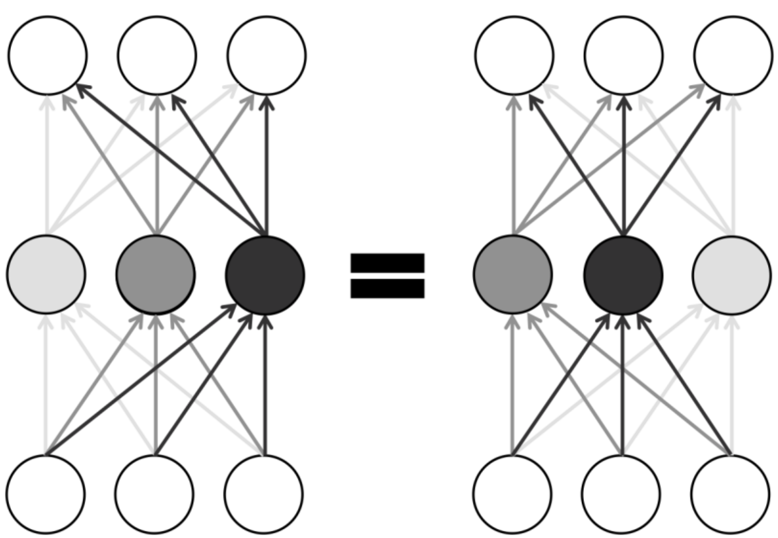
Non-convex optimization is NP-hard.

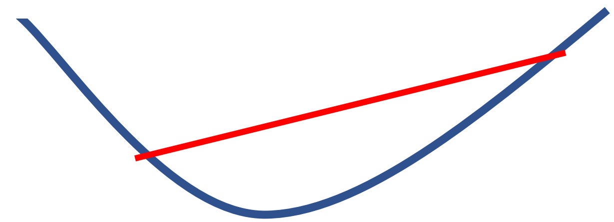
Why can we solve the problem by gradient descent?

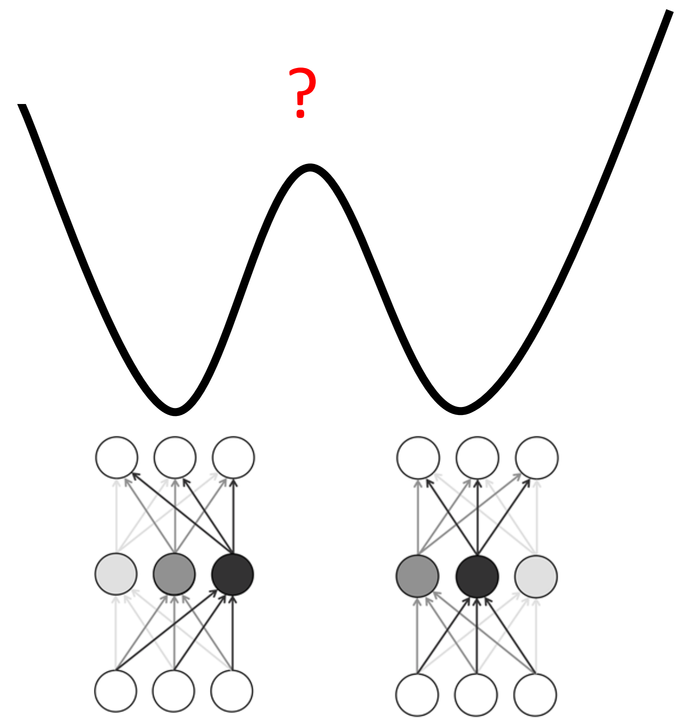
Loss of Deep Learning is not convex

There are at least exponentially many global minima for a neural net.

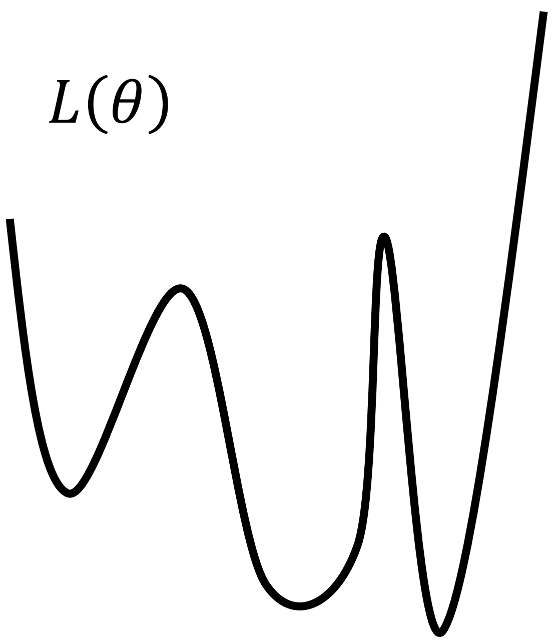
Permutating the neurons in one layer does not change the loss.



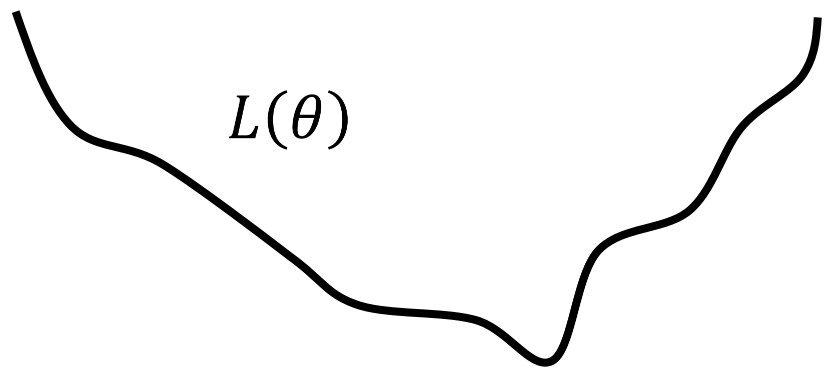


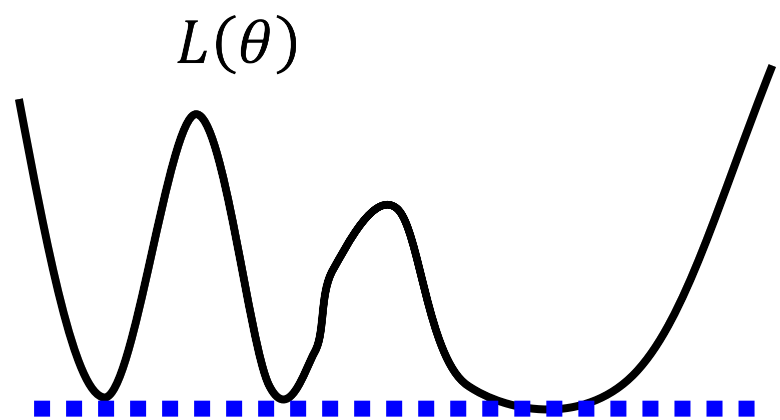


Non-convex ≠ Difficult



Not guarantee to find optimal solution by gradient descent





Outline

- Review: Hessian
- Deep Linear Model
- Deep Non-linear Model
- Conjecture about Deep Learning
- Empirical Observation about Error Surface

Hessian Matrix: When Gradient is Zero

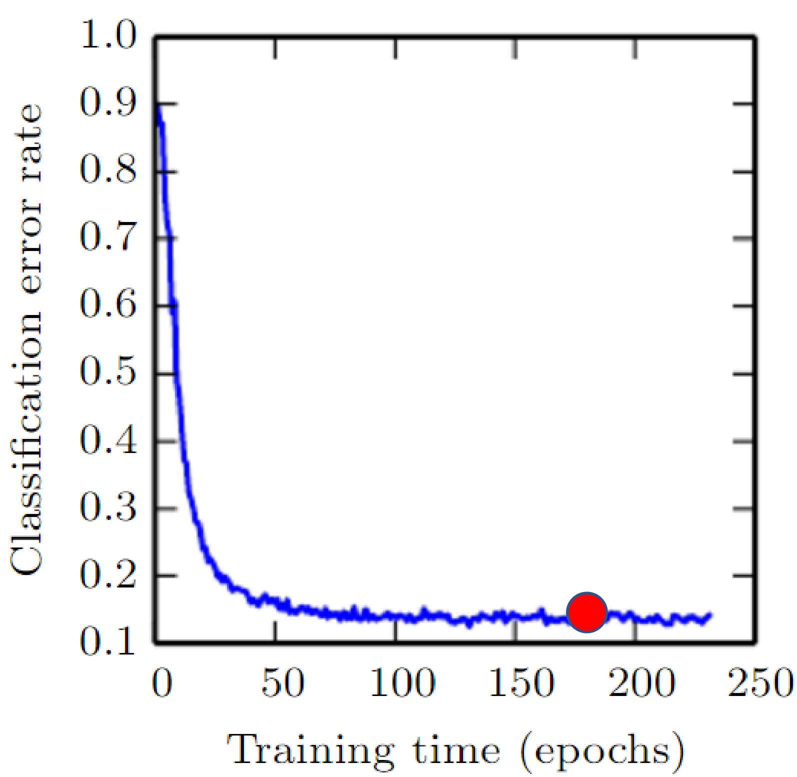
Some examples in this part are from:
https://www.math.upenn.edu/~kazdan/312F12/Notes/max-min-notesJan09/max-min.pdf

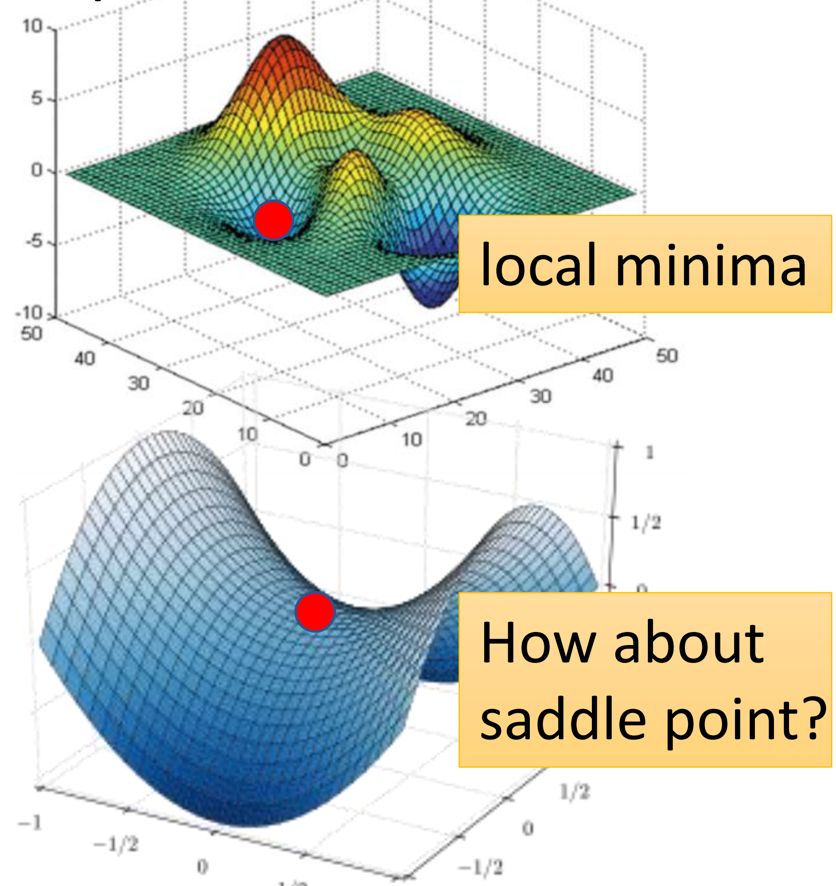
http://www.deeplearningbook.org/contents/optimization.html

Training stops ….

critical point:
gradient is zero

- People believe training stuck because the parameters are near a critical point





When Gradient is Zero

$$f(\theta) = f(\theta^0) + (\theta - \theta^0)^T g + \frac{1}{2}(\theta - \theta^0)^T H(\theta - \theta^0) + \cdots$$

Gradient g is a ***vector***

$$g_i = \frac{\partial f(\theta^0)}{\partial \theta_i} \qquad \nabla f(\theta^0)$$

Hessian H is a ***matrix***

$$H_{ij} = \frac{\partial^2}{\partial \theta_i \partial \theta_j} f(\theta^0)$$

symmetric

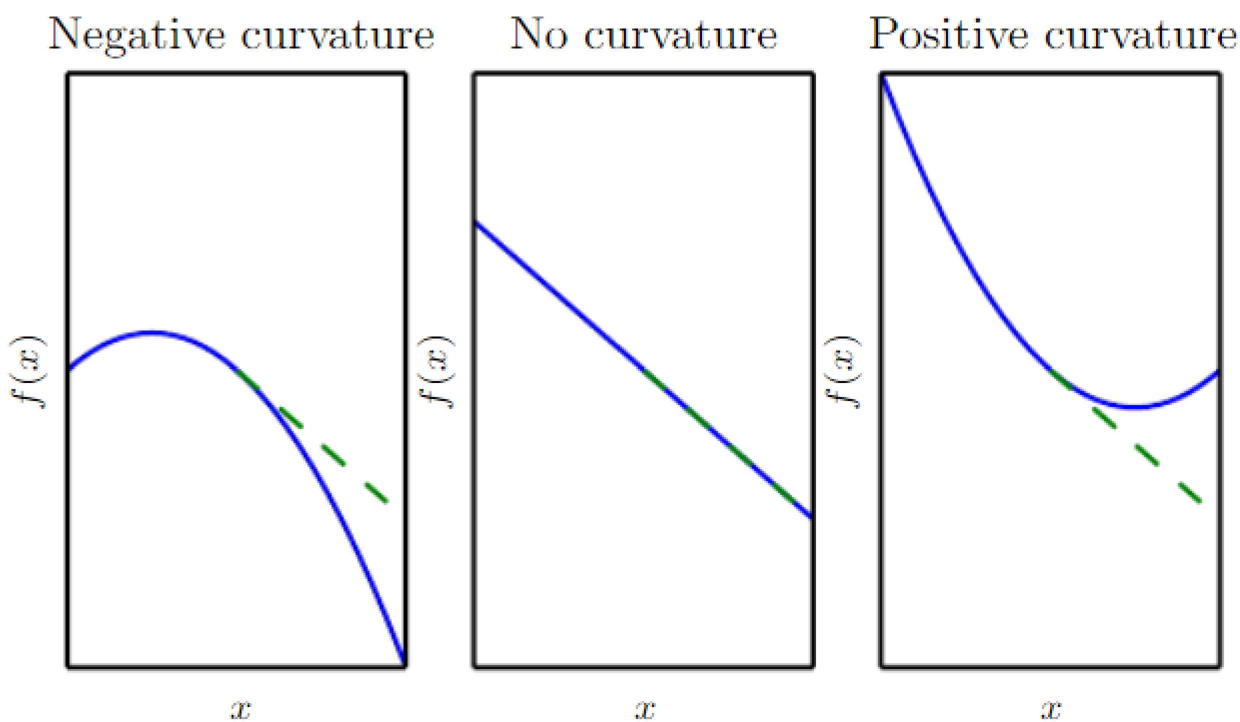
$$= \frac{\partial^2}{\partial \theta_j \partial \theta_i} f(\theta^0) = H_{ji}$$

Hessian

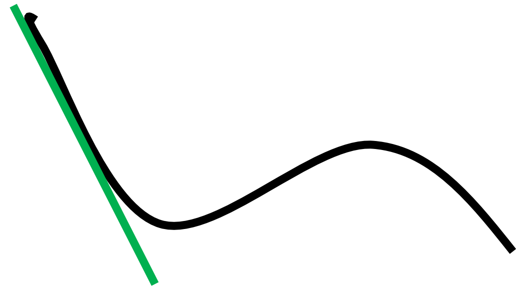
Source of image: http://www.deeplearningbook.org/contents/numerical.html

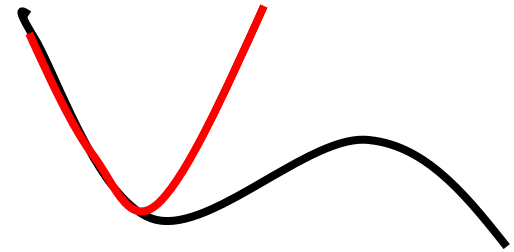
$$f(\theta) = f(\theta^0) + (\theta - \theta^0)^T g + \frac{1}{2}(\theta - \theta^0)^T H(\theta - \theta^0) + \cdots$$

H determines the curvature



Hessian





$$f(\theta) = f(\theta^0) + (\theta - \theta^0)^T g + \frac{1}{2}(\theta - \theta^0)^T H (\theta - \theta^0) + \cdots$$

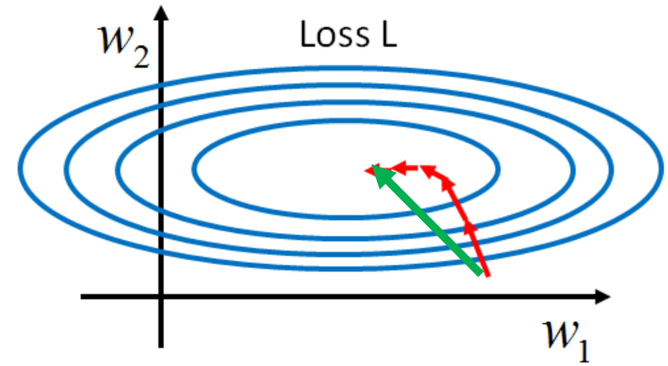
Newton's method → Find the space such that $\nabla f(\theta) = 0$

$$\nabla f(\theta) \approx \underbrace{\nabla[(\theta - \theta^0)^T g]}_{= g} + \underbrace{\nabla\left[\frac{1}{2}(\theta - \theta^0)^T H (\theta - \theta^0)\right]}_{H(\theta - \theta^0)}$$

$$\frac{\partial[(\theta - \theta^0)^T g]}{\partial \theta_i} = g_i$$

$$\frac{\partial\left[\frac{1}{2}(\theta - \theta^0)^T H (\theta - \theta^0)\right]}{\partial \theta_i}$$

Hessian



$$f(\theta) = f(\theta^0) + (\theta - \theta^0)^T g + \frac{1}{2}(\theta - \theta^0)^T H(\theta - \theta^0) + \cdots$$

<u>Newton's method</u>

$$\nabla f(\theta) \approx \underbrace{\nabla[(\theta - \theta^0)^T g]}_{= g} + \underbrace{\nabla\left[\frac{1}{2}(\theta - \theta^0)^T H(\theta - \theta^0)\right]}_{H(\theta - \theta^0)}$$

$$\nabla f(\theta) \approx g + H(\theta - \theta^0) = 0$$

$$H(\theta - \theta^0) = -g \qquad \theta = \theta^0 - \boxed{H^{-1}} g \quad \text{v.s.} \quad \theta = \theta^0 - \eta g$$

$$\theta - \theta^0 = -H^{-1} g$$

Change the direction, determine step size

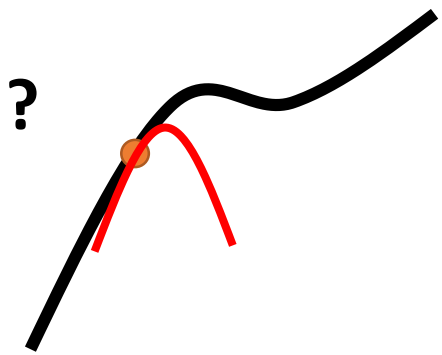
Hessian

Source of image: https://math.stackexchange.com/questions/609680/newtons-method-intuition

$$f(\theta) = f(\theta^0) + (\theta - \theta^0)^T g + \frac{1}{2}(\theta - \theta^0)^T H(\theta - \theta^0) + \cdots$$

Newton's method Not suitable for Deep Learning

Consider that $\theta = x$



If $f(x)$ is a quadratic function, obtain critical point in one step.

What is the problem?

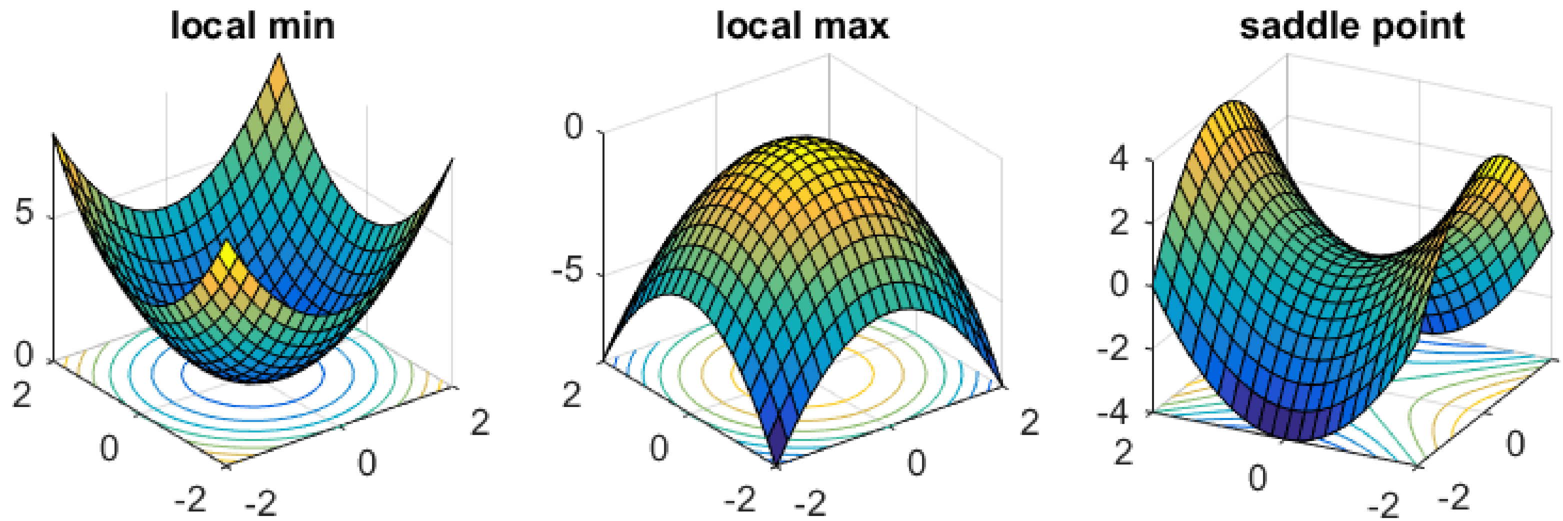
Source of image:
http://www.offconvex.org/2016/03/22/saddlepoints/

Hessian

$$f(\theta) = f(\theta^0) + \cancel{(\theta - \theta^0)^T g} + \frac{1}{2}(\theta - \theta^0)^T H (\theta - \theta^0) + \cdots$$

At critical point ($g = 0$)

H tells us the properties of critical points.



http://speech.ee.ntu.edu.tw/~tlkagk/courses/LA_2016/Lecture/eigen.pdf

Review: Linear Algebra

- If $Av = \lambda v$ (v is a vector, λ is a scalar)
  - v is an eigenvector of A excluding zero vector
  - λ is an eigenvalue of A that corresponds to v

A must be square

Eigen value

$$\begin{bmatrix} 5 & 2 & 1 \\ -2 & 1 & -1 \\ 2 & 2 & 4 \end{bmatrix} \begin{bmatrix} 1 \\ -1 \\ 1 \end{bmatrix} = \begin{bmatrix} 4 \\ -4 \\ 4 \end{bmatrix} = 4 \begin{bmatrix} 1 \\ -1 \\ 1 \end{bmatrix}$$

Eigen vector

Review: Positive/Negative Definite

- An nxn matrix A is symmetric.
- For every non-zero vector x ($x \neq 0$)

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

positive definite: $x^T A x > 0$ $\Leftrightarrow$ All eigen values are positive.

positive semi-definite: $x^T A x \geq 0$ $\Leftrightarrow$ All eigen values are non-negative.

negative definite: $x^T A x < 0$ $\Leftrightarrow$ All eigen values are negative.

negative semi-definite: $x^T A x \leq 0$ $\Leftrightarrow$ All eigen values are non-positive.

Hessian

At critical point: $x^T H x$

$$f(\theta) \approx f(\theta^0) + \frac{1}{2}\underline{(\theta - \theta^0)^T H (\theta - \theta^0)}$$

H is positive definite $\Rightarrow$ $x^T H x > 0$ $\qquad x^T H x \geq 0?$

All eigen values are positive.

$\Rightarrow$ Around θ^0: $f(\theta) > f(\theta^0)$ $\Rightarrow$ Local minima

H is negative definite $\Rightarrow$ $x^T H x < 0$ $\qquad x^T H x \leq 0?$

All eigen values are negative

$\Rightarrow$ Around θ^0: $f(\theta) < f(\theta^0)$ $\Rightarrow$ Local maxima

Sometimes$x^T H x > 0$, sometimes$x^T H x < 0$

$\Rightarrow$ Saddle point

Hessian

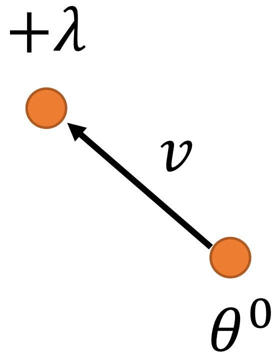
At critical point:

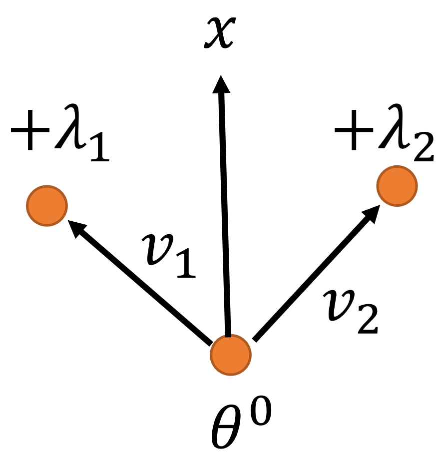
$$f(\theta) \approx f(\theta^0) + \frac{1}{2}(\theta - \theta^0)^T H(\theta - \theta^0)$$

v is an eigen vector $\Rightarrow$ $v^T H v = v^T(\lambda v) = \lambda \|v\|^2$

Unit vector $= \lambda$

(Ignore 1/2 for simplicity)





$$x = a_1 v_1 + a_2 v_2$$

$x^T H x$

$= (a_1 v_1 + a_2 v_2)^T H (a_1 v_1 + a_2 v_2)$

v_1 and v_2 are orthogonal

$= (a_1)^2 \lambda_1 + (a_x)^2 \lambda_2$?

Because H is an nxn symmetric matrix,

H can have eigen vectors $\{v_1, v_2, \dots, v_n\}$ form a orthonormal basis.

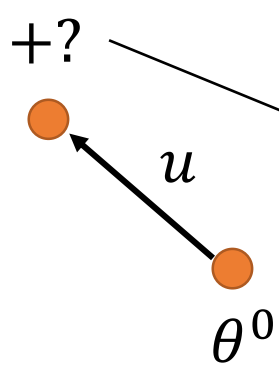
Hessian

At critical point:

$$f(\theta) \approx f(\theta^0) + \frac{1}{2}(\theta - \theta^0)^T H (\theta - \theta^0)$$

v is an eigen vector ➡ $v^T H v = v^T(\lambda v) = \lambda \|v\|^2$

Unit vector $= \lambda$



$$u = a_1 v_1 + a_2 v_2 + \cdots + a_n v_n$$

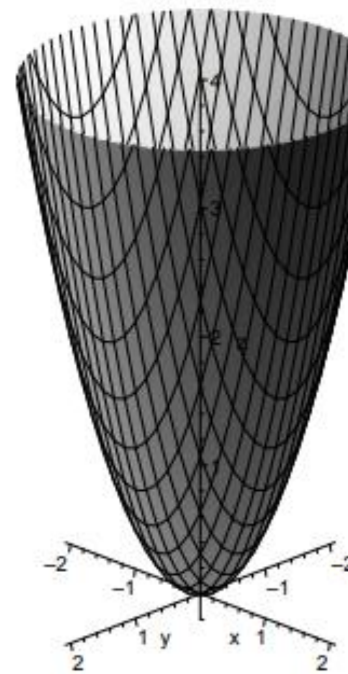
$$(a_1)^2 \lambda_1 + (a_2)^2 \lambda_2 + \cdots + (a_n)^2 \lambda_n$$

Because H is an nxn symmetric matrix,

H can have eigen vectors $\{v_1, v_2, \ldots, v_n\}$ form a orthonormal basis.

Examples

$$f(x, y) = x^2 + 3y^2$$



$$\frac{\partial f(x, y)}{\partial x} = 2x \qquad \frac{\partial f(x, y)}{\partial y} = 6y$$

$$x = 0, y = 0$$

$$\frac{\partial^2}{\partial x \partial x} f(x, y) = 2 \qquad \frac{\partial^2}{\partial x \partial y} f(x, y) = 0$$

$$\frac{\partial^2}{\partial y \partial x} f(x, y) = 0 \qquad \frac{\partial^2}{\partial y \partial y} f(x, y) = 6$$

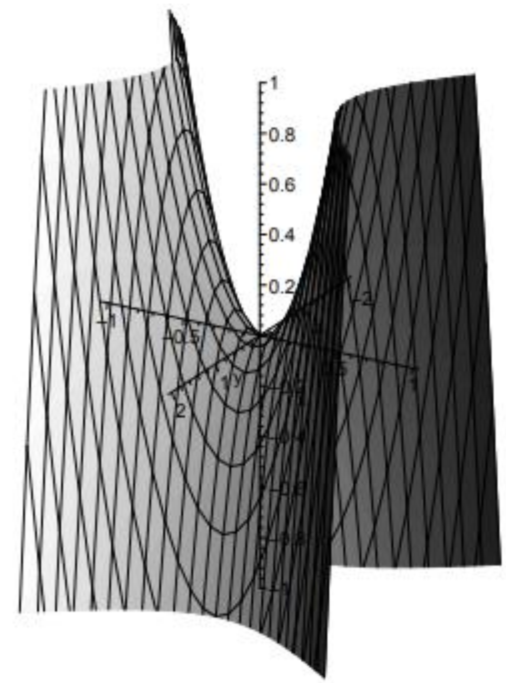
$$H = \begin{bmatrix} 2 & 0 \\ 0 & 6 \end{bmatrix}$$

Positive-definite

Local minima

$$f(x,y) = -x^2 + 3y^2$$

Examples



$$\frac{\partial f(x,y)}{\partial x} = -2x \quad \frac{\partial f(x,y)}{\partial y} = 6y$$

$$x = 0, y = 0$$

$$\frac{\partial^2}{\partial x \partial x} f(x,y) = -2 \qquad \frac{\partial^2}{\partial x \partial y} f(x,y) = 0$$

$$\frac{\partial^2}{\partial y \partial x} f(x,y) = 0 \qquad \frac{\partial^2}{\partial y \partial y} f(x,y) = 6$$

$$H = \begin{bmatrix} -2 & 0 \\ 0 & 6 \end{bmatrix}$$

Saddle

Degenerate

- Degenerate Hessian has at least one zero eigen value

$$f(x,y) = x^2 + y^4$$

$$\frac{\partial f(x,y)}{\partial x} = 2x \qquad \frac{\partial f(x,y)}{\partial y} = 4y^3 \qquad x = y = 0$$

$$\frac{\partial^2}{\partial x \partial x} f(x,y) = 2 \qquad \frac{\partial^2}{\partial x \partial y} f(x,y) = 0 \qquad H = \begin{bmatrix} 2 & 0 \\ 0 & 0 \end{bmatrix}$$

$$\frac{\partial^2}{\partial y \partial x} f(x,y) = 0 \qquad \frac{\partial^2}{\partial y \partial y} f(x,y) = 12y^2$$

Degenerate

- Degenerate Hessian has at least one zero eigen value

$$f(x, y) = x^2 + y^4 \qquad\qquad g(x, y) = x^2 - y^4$$

$$x = y = 0 \qquad\qquad x = y = 0$$

$$g = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \qquad\qquad g = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$H = \begin{bmatrix} 2 & 0 \\ 0 & 0 \end{bmatrix} \qquad\qquad H = \begin{bmatrix} 2 & 0 \\ 0 & 0 \end{bmatrix}$$

No Difference

Degenerate

$$h(x, y) = 0$$

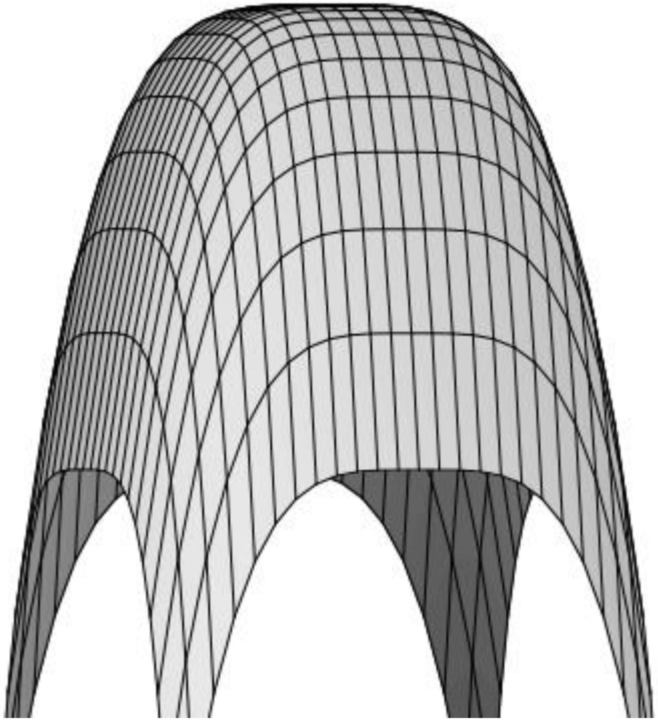
$$x = y = 0$$

$$g = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$H = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$$

$$f(x, y) = -x^4 - y^4$$

$$\frac{\partial f(x, y)}{\partial x} = -4x^3 \quad \frac{\partial f(x, y)}{\partial y} = -4y^3$$



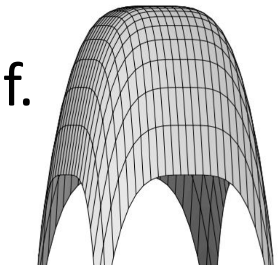
$$\frac{\partial^2}{\partial x \partial x} f(x, y) = -12x^2 \quad \frac{\partial^2}{\partial x \partial y} f(x, y) = 0$$

$$\frac{\partial^2}{\partial y \partial x} f(x, y) = 0 \quad \frac{\partial^2}{\partial y \partial y} f(x, y) = -12y^2$$

$$H = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$$

http://homepages.math.uic.edu/~julius/monkeysaddle.html

c.f.



Monkey Saddle

$$\frac{\partial f(x,y)}{\partial x} = 3x^2 - 3y^2$$

$$\frac{\partial f(x,y)}{\partial y} = -6xy$$

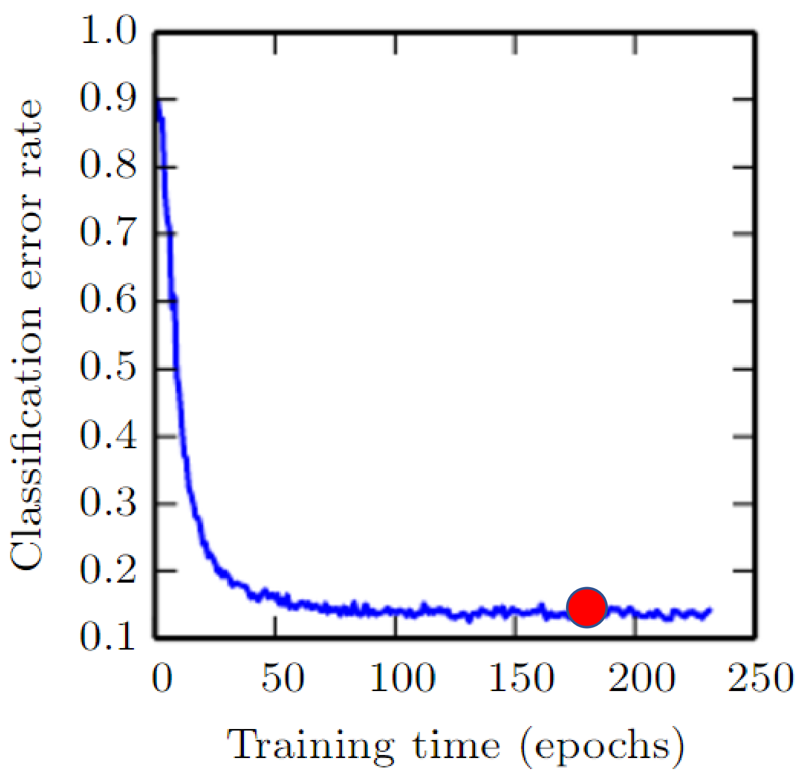
$$\frac{\partial^2}{\partial x \partial x} f(x,y) = 6x \qquad \frac{\partial^2}{\partial x \partial y} f(x,y) = -6y$$

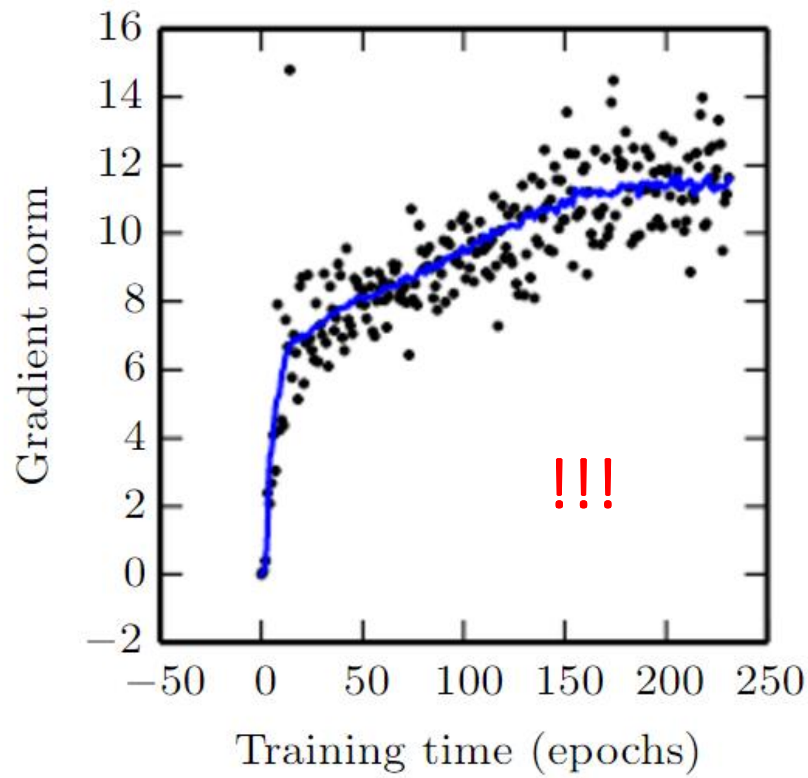
$$\frac{\partial^2}{\partial y \partial x} f(x,y) = -6y \qquad \frac{\partial^2}{\partial y \partial y} f(x,y) = -6x$$

http://www.deeplearningbook.org/contents/optimization.html

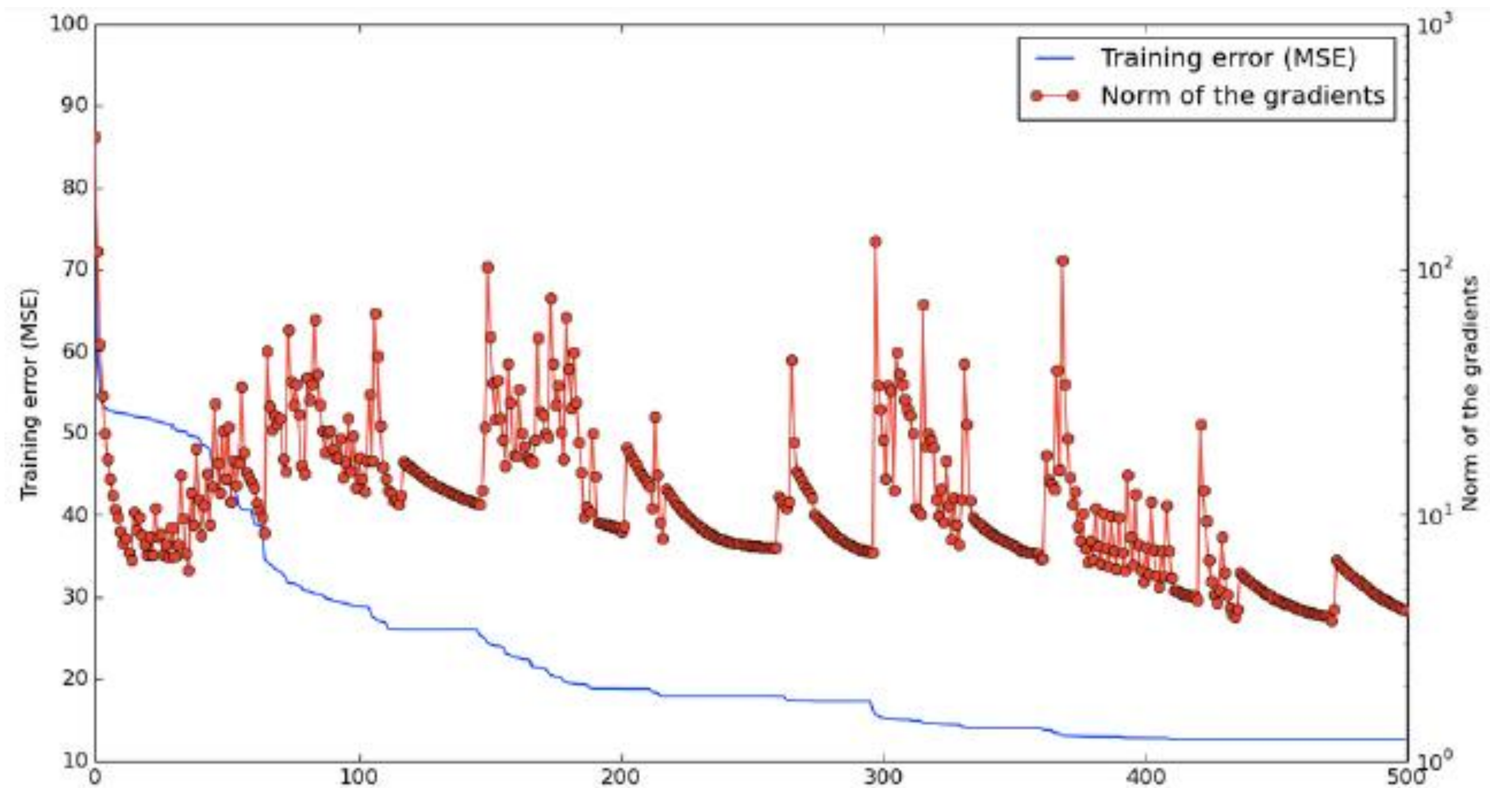
Training stuck ≠ Zero Gradient

- People believe training stuck because the parameters are around a critical point





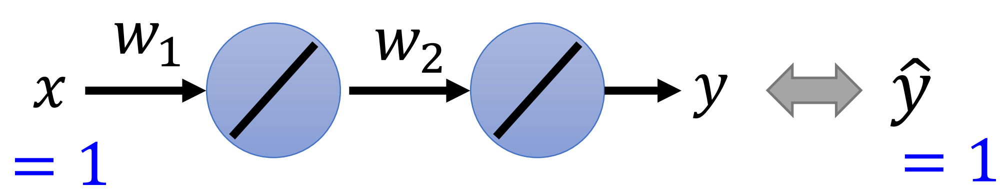
Training stuck ≠ Zero Gradient

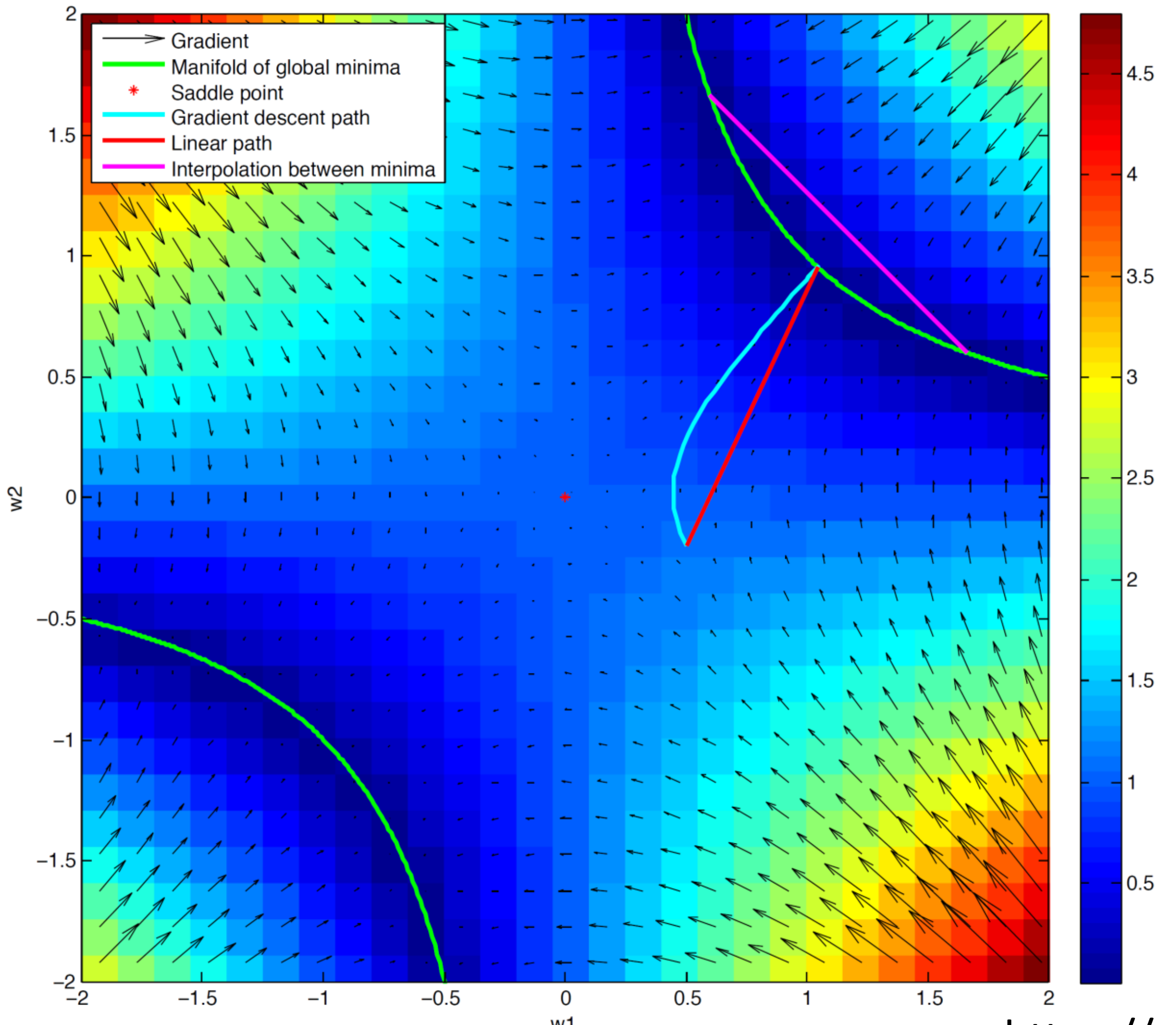


Approach a saddle point, and then escape

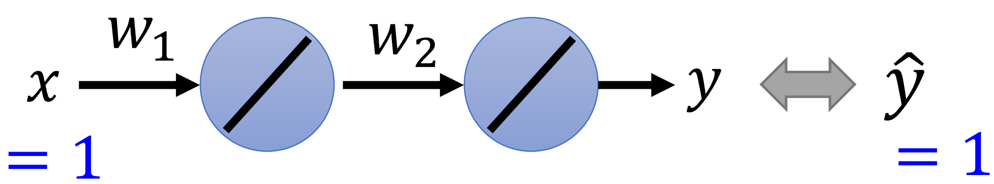
http://videolectures.net/deeplearning2015_bengio_theoretical_motivations/

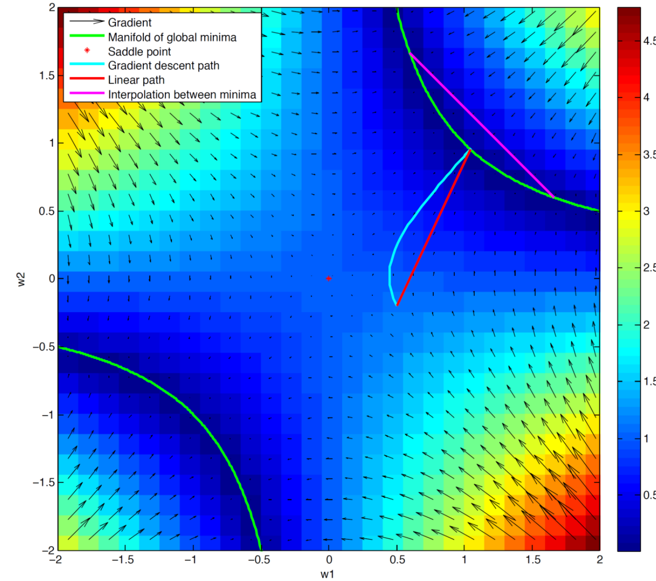
Deep Linear Network





https://arxiv.org/abs/1412.6544





$$L = (\hat{y} - w_1 w_2 x)^2 \quad = (1 - w_1 w_2)^2$$

$$\frac{\partial L}{\partial w_1} = 2(1 - w_1 w_2)(-w_2)$$

$$\frac{\partial L}{\partial w_2} = 2(1 - w_1 w_2)(-w_1)$$

The probability of stuck as saddle point is almost zero.

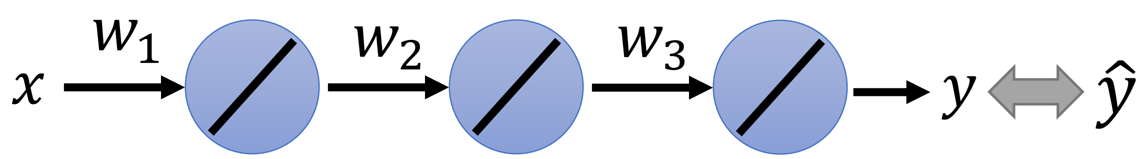
Easy to escape

$$\frac{\partial^2 L}{\partial {w_1}^2} = 2(-w_2)(-w_2) \qquad \frac{\partial^2 L}{\partial w_1 \partial w_2} = -2 + 4 w_1 w_2$$

$$\frac{\partial^2 L}{\partial w_2 \partial w_1} = -2 + 4 w_1 w_2 \qquad \frac{\partial^2 L}{\partial {w_2}^2} = 2(-w_1)(-w_1)$$

2-hidden layers

$$L = (1 - w_1 w_2 w_3)^2$$



$$\frac{\partial L}{\partial w_1} = 2(1 - w_1 w_2 w_3)(-w_2 w_3)$$

$$\frac{\partial L}{\partial w_2} = 2(1 - w_1 w_2 w_3)(-w_1 w_3)$$

$$\frac{\partial L}{\partial w_3} = 2(1 - w_1 w_2 w_3)(-w_1 w_2)$$

$$\frac{\partial^2 L}{\partial {w_1}^2} = 2(-w_2 w_3)^2$$

$$\frac{\partial^2 L}{\partial w_2 \partial w_1} = -2w_3 + 4w_1 w_2 (w_3)^2$$

$w_1 w_2 w_3 = 1$ global minima

$$w_1 = w_2 = w_3 = 0$$

$$H = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

So flat

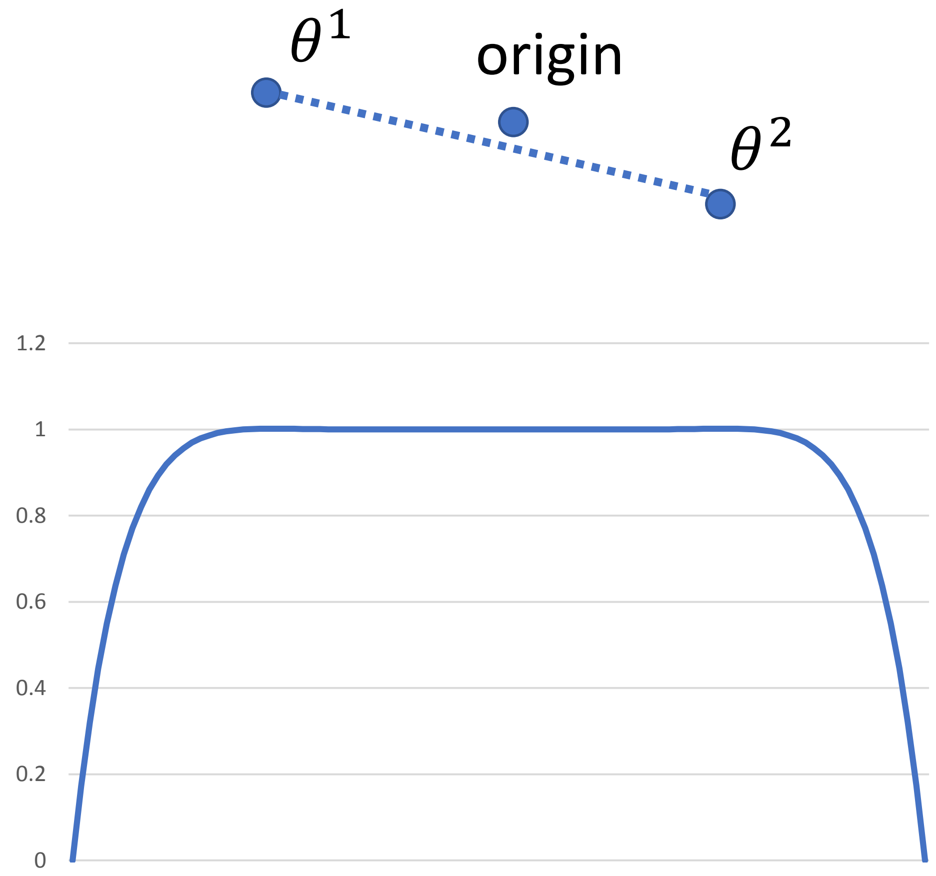
$$w_1 = w_2 = 0, w_3 = k$$

$$H = \begin{bmatrix} 0 & -2 & 0 \\ -2 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

Saddle point

All minima are global, some critical points are “bad”.

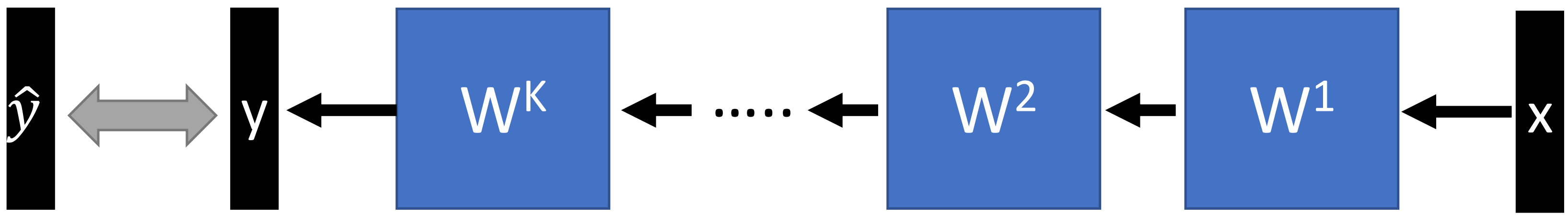
10 hidden layers



10-hidden layers

Demo

Deep Linear Network



$$y = W^K W^{K-1} \cdots W^2 W^1 x \qquad L = \sum_{n=1}^{N} (x^n - \hat{y}^n)^2$$

Hidden layer size $\geq$ Input dim, output dim

More than two hidden layers can produce saddle point without negative eigenvalues.

Reference

- Kenji Kawaguchi, Deep Learning without Poor Local Minima, NIPS, 2016
- Haihao Lu, Kenji Kawaguchi, Depth Creates No Bad Local Minima, arXiv, 2017
- Thomas Laurent, James von Brecht, Deep linear neural networks with arbitrary loss: All local minima are global, arXiv, 2017
- Maher Nouiehed, Meisam Razaviyayn, Learning Deep Models: Critical Points and Local Openness, arXiv, 2018

Non-linear Deep Network

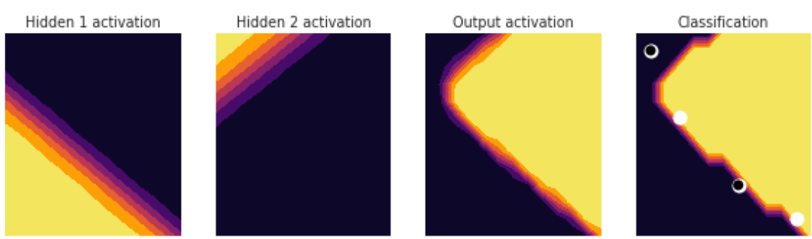
Does it have local minima?

證明事情不存在很難，證明事情存在相對容易

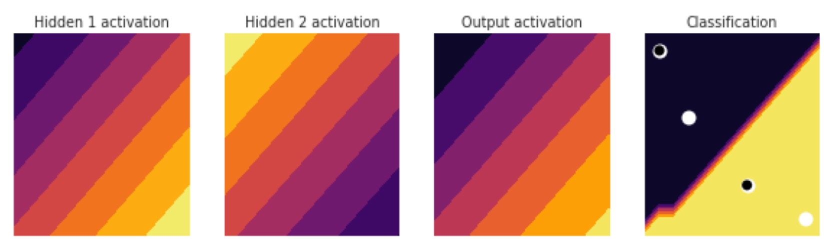
感謝曾子家同學發現投影片上的錯字

Even Simple Task can be Difficult

h		XOR ReLU	XOR Sigmoid	Jellyfish ReLU	Jellyfish Sigmoid		XOR ReLU	XOR Sigmoid	Jellyfish ReLU	Jellyfish Sigmoid
2	Adam	28%	79%	7%	0%	GD	23%	90%	16%	62%
3	Adam	52%	98%	34%	0%	GD	47%	100%	33%	100%
4	Adam	68%	100%	50%	2%	GD	70%	100%	66%	100%
5	Adam	81%	100%	51%	27%	GD	80%	100%	68%	100%
6	Adam	91%	100%	61%	17%	GD	89%	100%	69%	100%
7	Adam	97%	100%	69%	58%	GD	89%	100%	86%	100%

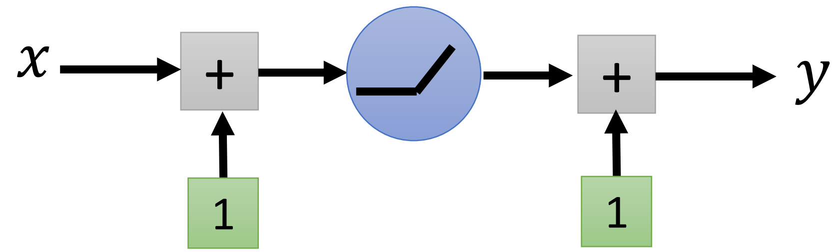


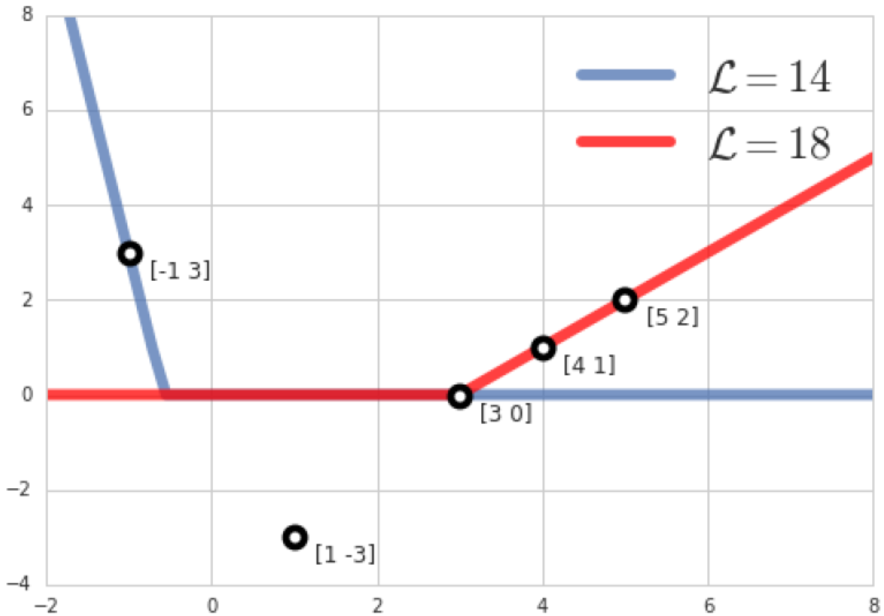
(a) Optimally converged net for Jellyfish.



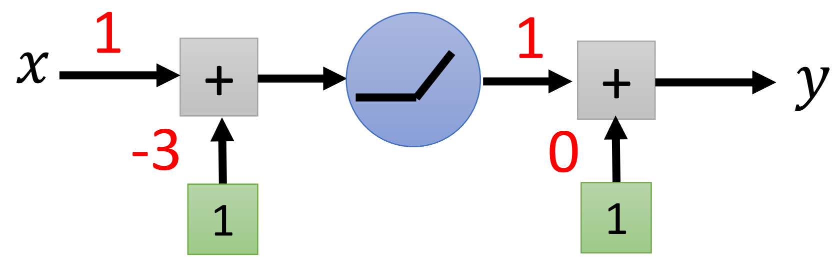
(b) Stuck net for Jellyfish.

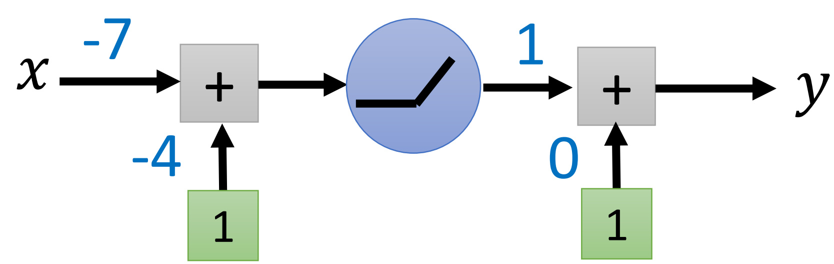
ReLU has local





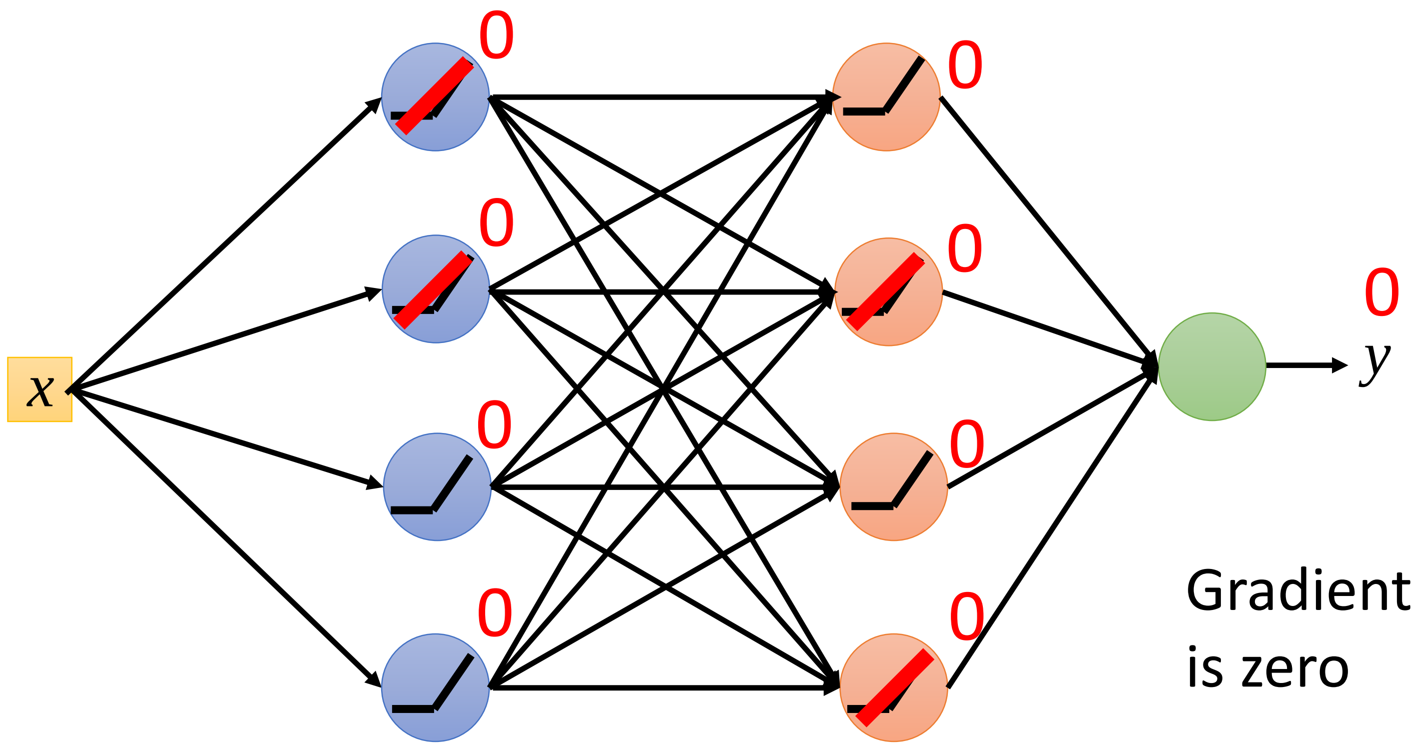
(-1.3) (1,-3) (3,0) (4,1) (5,2)





This relu network has local minima.

“Blind Spot” of ReLU



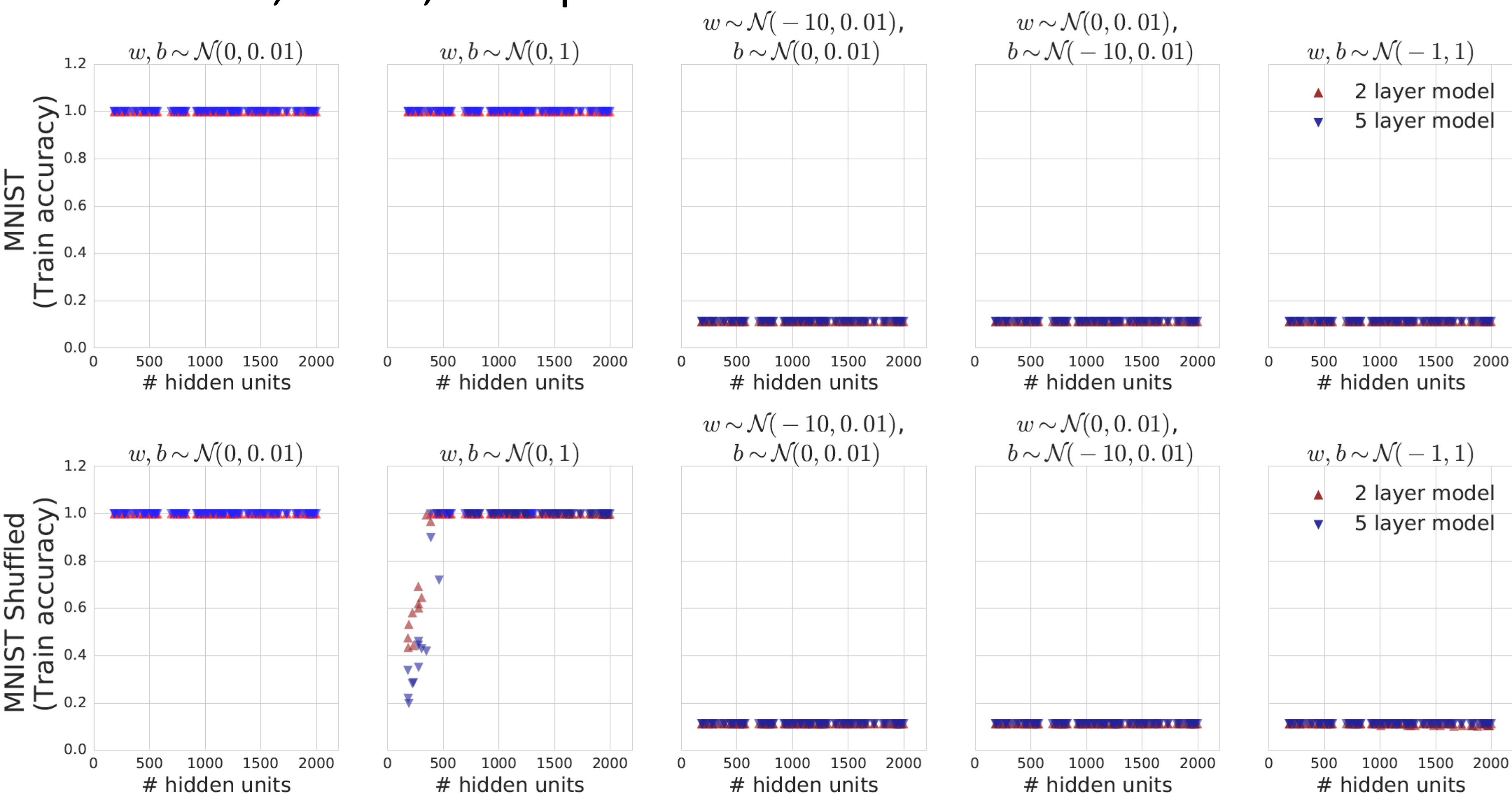
Gradient is zero

It is pretty easy to make this happens ……

“Blind Spot” of ReLU

Consider your initialization

- MNIST, Adam, 1M updates

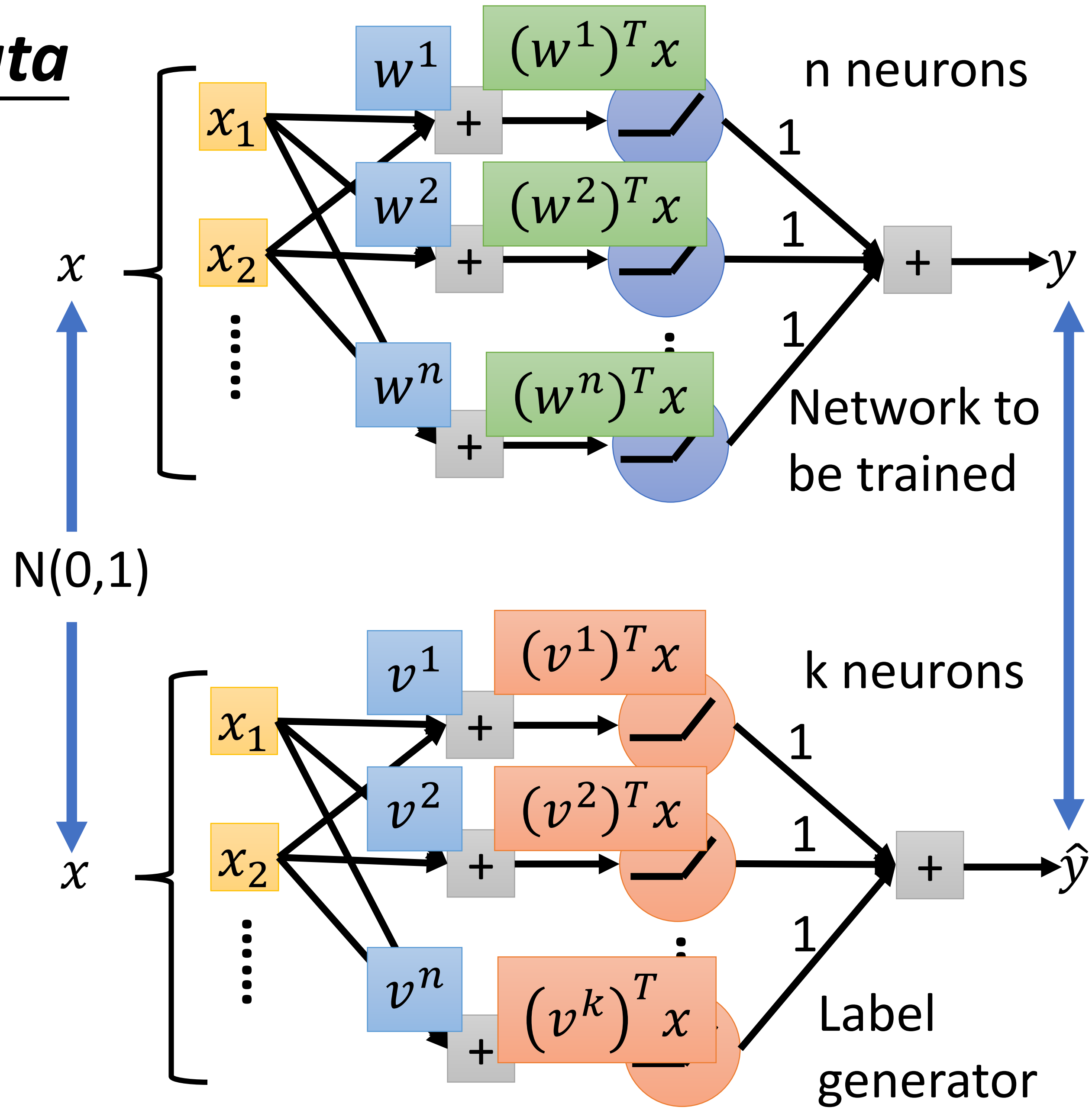


Considering Data

If $n \geq k$ and
$w^i = v^i$

We obtain
global minima

The number of
k and n matters



Considering Data

Table 1: Spurious local minima found for $n = k$

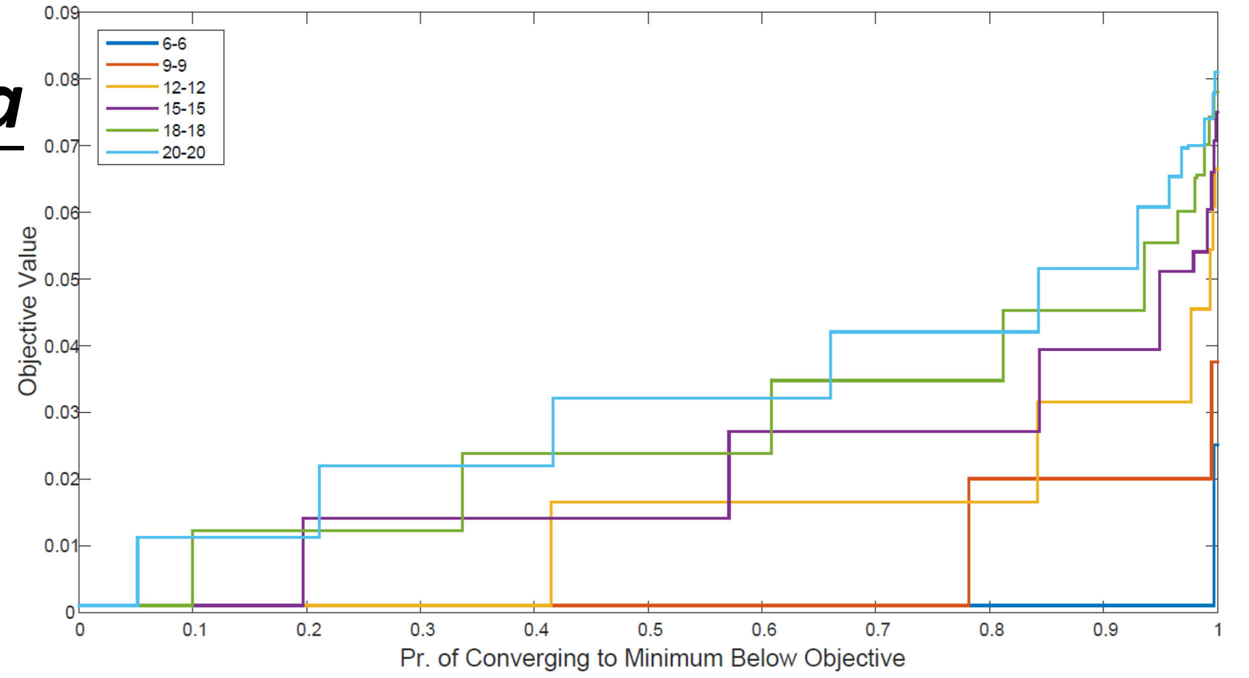
k	n	% of runs converging to local minima	Average minimal eigenvalue	Average objective value
6	6	0.3%	0.0047	0.025
7	7	5.5%	0.014	0.023
8	8	12.6%	0.021	0.021
9	9	21.8%	0.027	0.02
10	10	34.6%	0.03	0.022
11	11	45.5%	0.034	0.022
12	12	58.5%	0.035	0.021
13	13	73%	0.037	0.022
14	14	73.6%	0.038	0.023
15	15	80.3%	0.038	0.024
16	16	85.1%	0.038	0.027
17	17	89.7%	0.039	0.027
18	18	90%	0.039	0.029
19	19	93.4%	0.038	0.031
20	20	94%	0.038	0.033

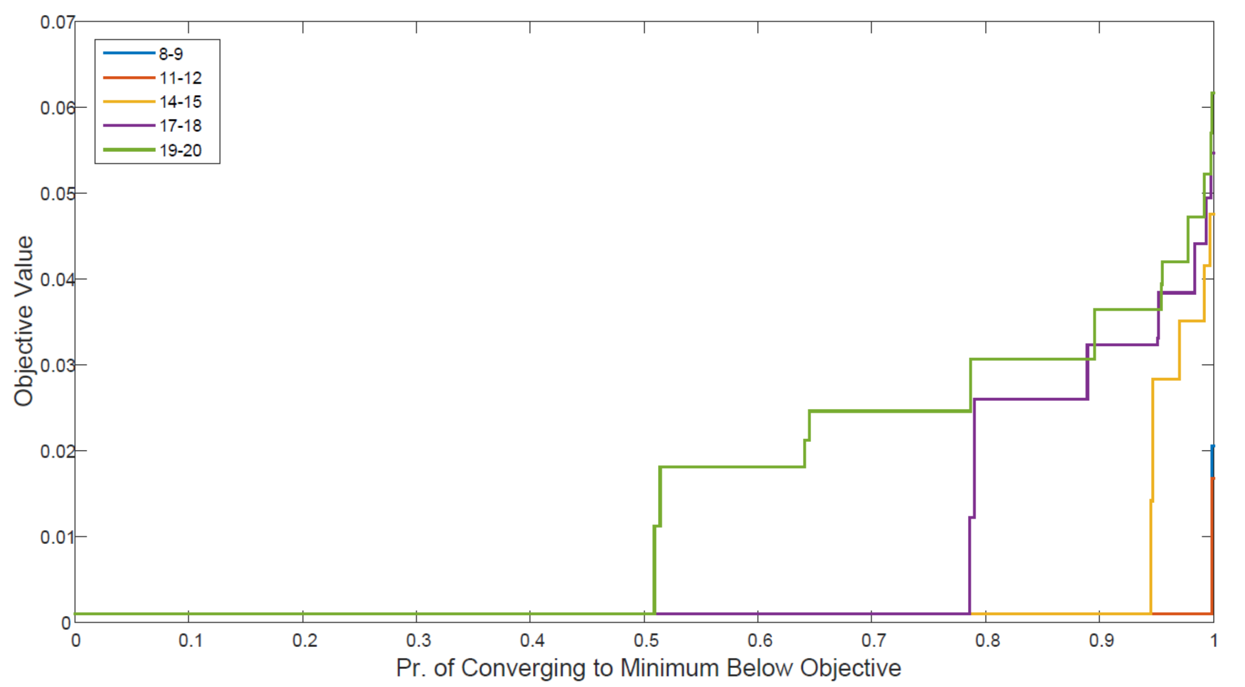
Table 2: Spurious local minima found for $n \neq k$

k	n	% of runs converging to local minima	Average minimal eigenvalue	Average objective value
8	9	0.1%	0.0059	0.021
10	11	0.1%	0.0057	0.018
11	12	0.1%	0.0056	0.017
12	13	0.3%	0.0054	0.016
13	14	1.5%	0.0015	0.038
14	15	5.5%	0.002	0.033
15	16	10.1%	0.004	0.032
16	17	18%	0.0055	0.031
17	18	20.9%	0.007	0.031
18	19	36.9%	0.0064	0.028
19	20	49.1%	0.0077	0.027

No local for $n \geq k + 2$

Considering Data





Reference

- Grzegorz Swirszcz, Wojciech Marian Czarnecki, Razvan Pascanu, “Local minima in training of neural networks”, arXiv, 2016
- Itay Safran, Ohad Shamir, “Spurious Local Minima are Common in Two-Layer ReLU Neural Networks”, arXiv, 2017
- Yi Zhou, Yingbin Liang, “Critical Points of Neural Networks: Analytical Forms and Landscape Properties”, arXiv, 2017
- Shai Shalev-Shwartz, Ohad Shamir, Shaked Shammah, “Failures of Gradient-Based Deep Learning”, arXiv, 2017

The theory should looks like …

Under some conditions (initialization, data, ……),

We can find global optimal.

Conjecture about Deep Learning

Almost all local minimum have very similar loss to the global optimum, and hence finding a local minimum is good enough.

Analyzing Hessian

- When we meet a critical point, it can be saddle point or local minima.
- Analyzing H

If the network has N parameters

$$
\begin{array}{ccccc}
v_1 & v_2 & v_3 & \ldots\ldots & v_N \\
\lambda_1 & \lambda_2 & \lambda_3 & \ldots\ldots & \lambda_N
\end{array}
$$

We assume λ has 1/2 (?) to be positive, 1/2 (?) to be negative.

Analyzing Hessian

- If N=1: v_1 λ_1

  1/2 local minima, 1/2 local maxima,
  Saddle point is almost impossible

- If N=2: v_1 v_2 λ_1 λ_2

  + + - -

  1/4 local minima, 1/4 local maxima,
  1/2 Saddle points

  + -, - +

- If N=10:

  1/1024 local minima, 1/1024 local maxima,
  Almost every critical point is saddle point

When a network is very large,

It is almost impossible to meet a local minima.

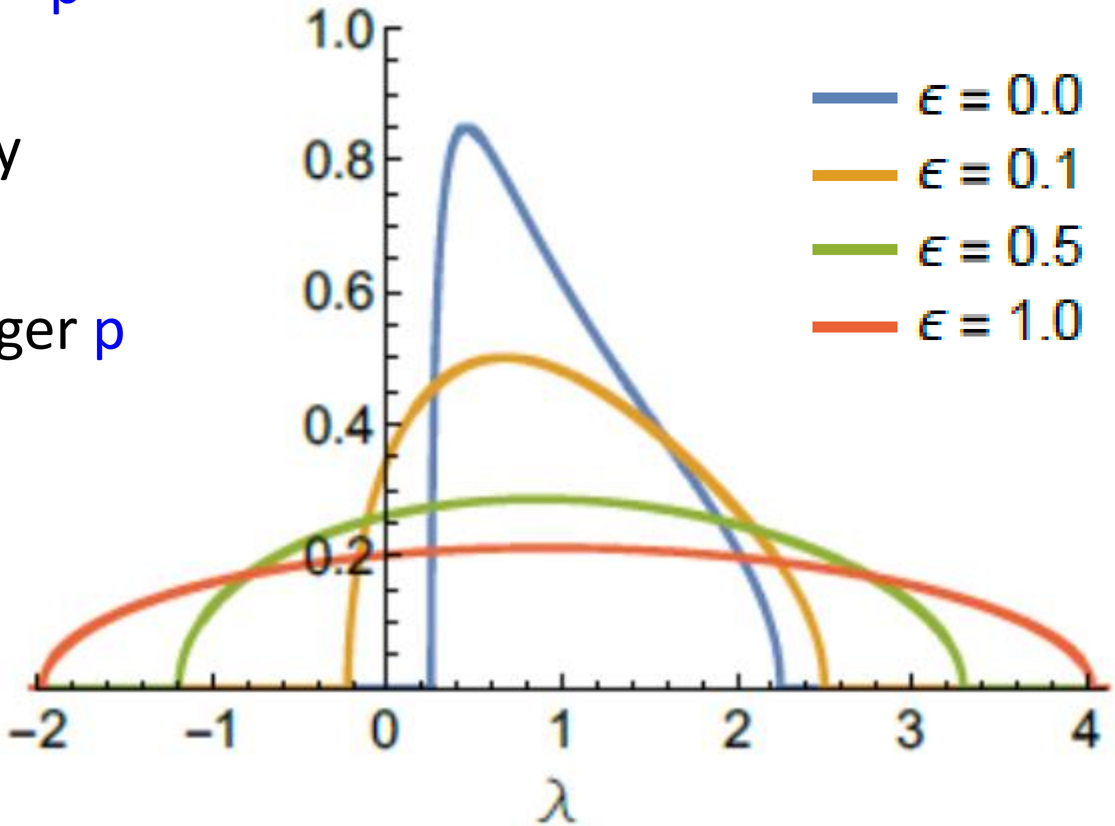
Saddle point is what you need to worry about.

Error v.s. Eigenvalues

We assume λ has ~~1/2 (?)~~ p to be negative.

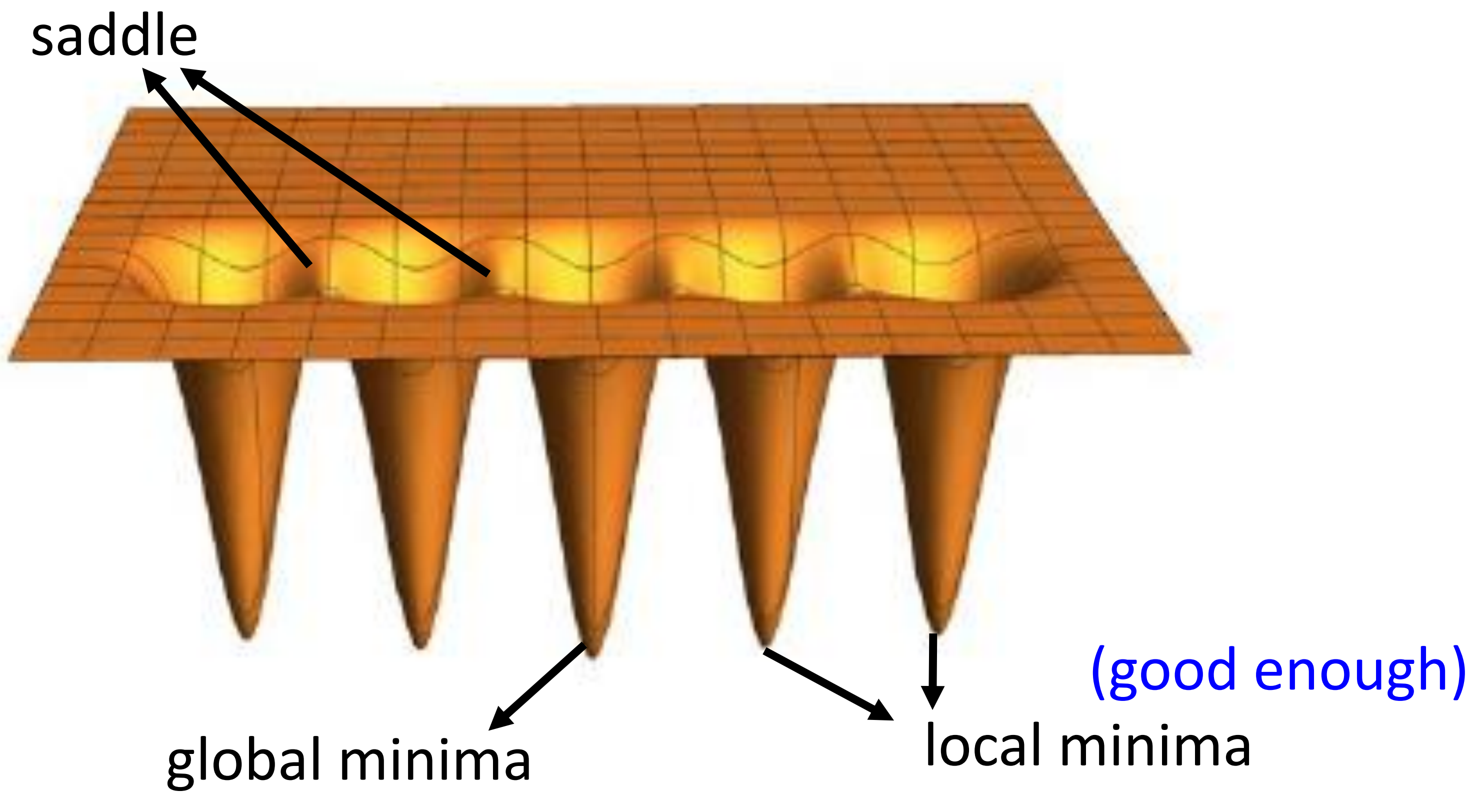
p is a probability related to error

Larger error, larger p



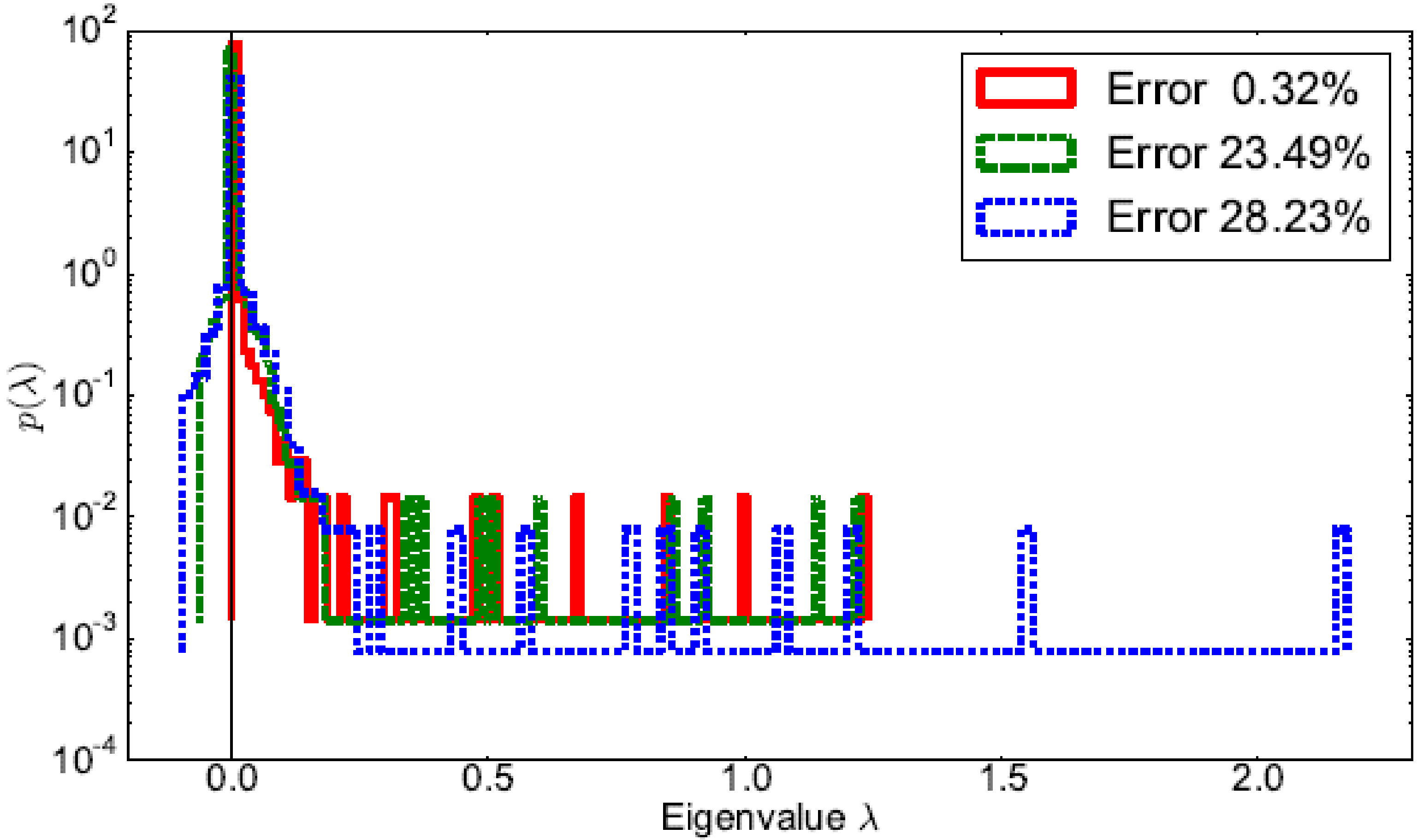
Source of image:
http://proceedings.mlr.press/v70/pennington17a/pennington17a.pdf

Guess about Error Surface



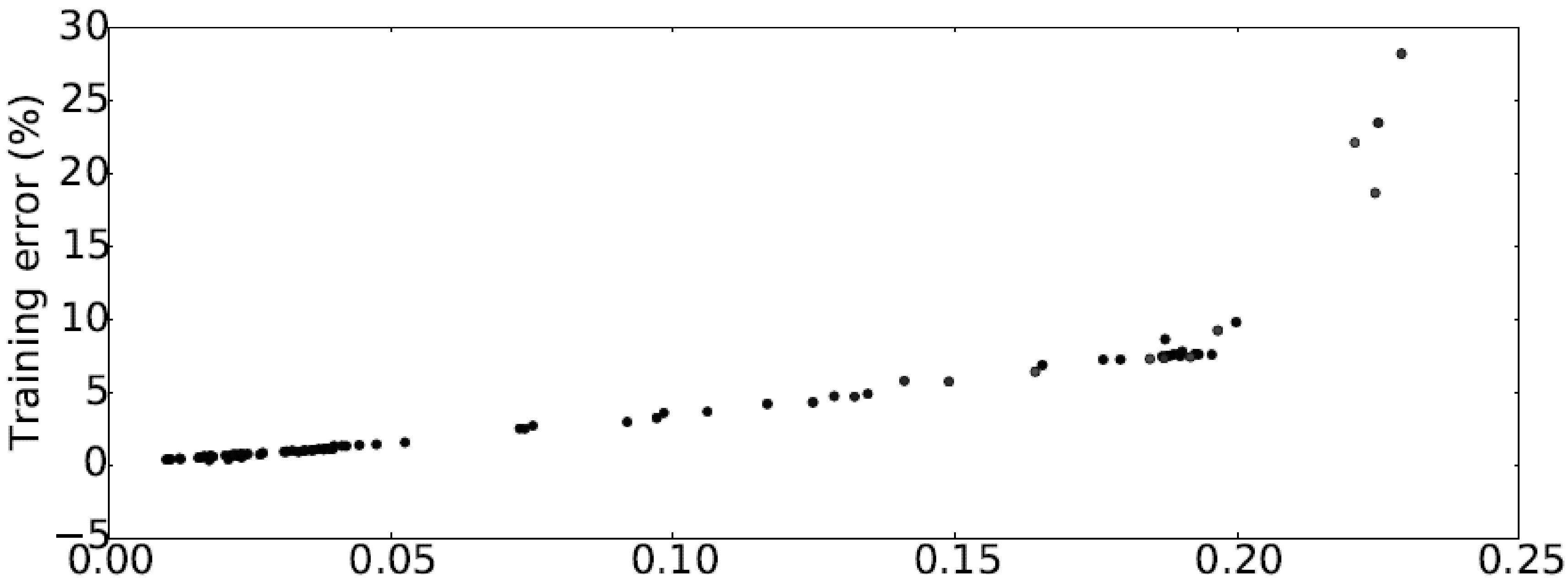
https://stats385.github.io/assets/lectures/Understanding_and_improving_deep_lea
ring_with_random_matrix_theory.pdf

Training Error v.s. Eigenvalues



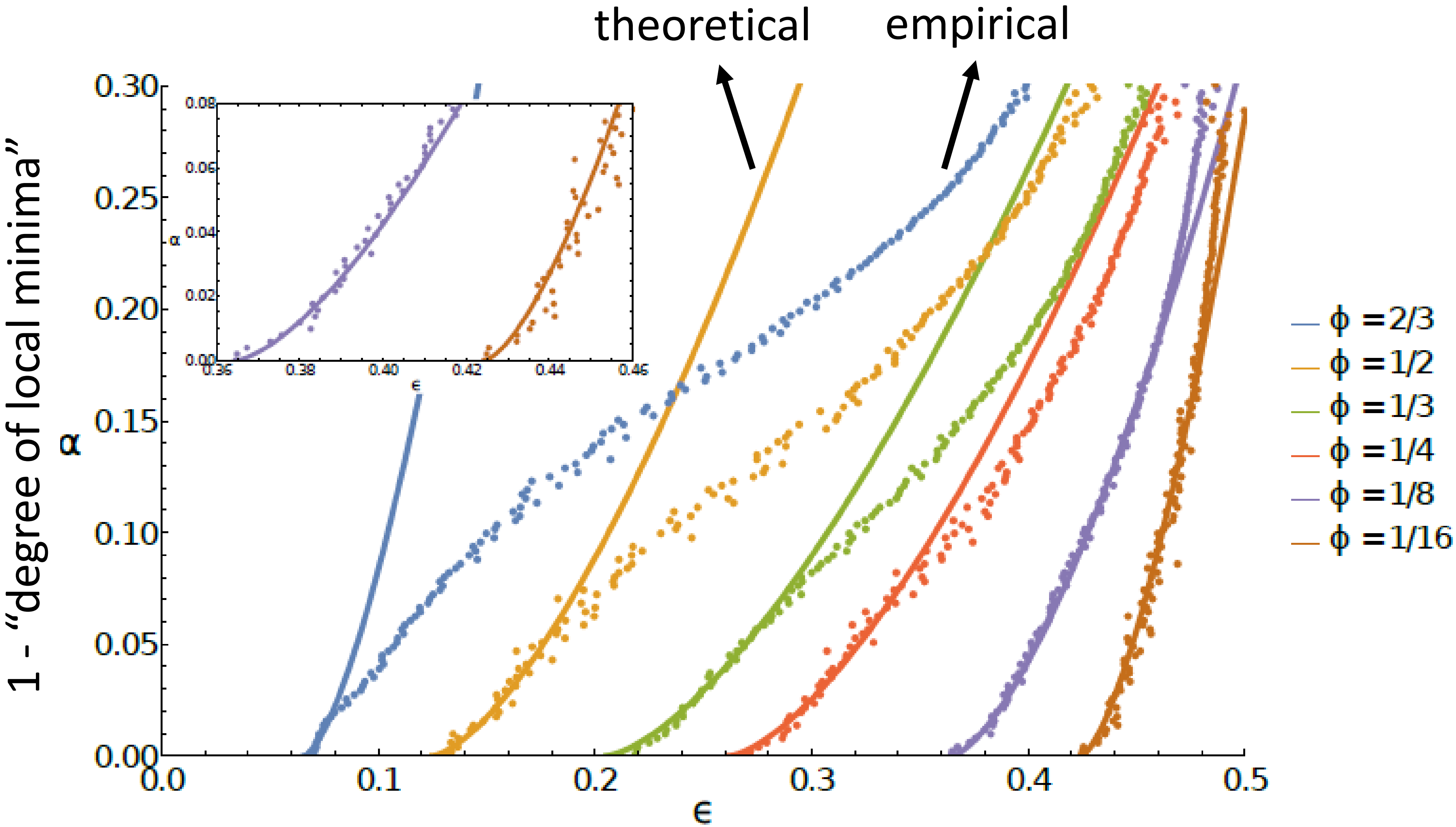
Training Error v.s. Eigenvalues

Portion of positive eigenvalues “Degree of Local Minima”



1 - “degree of local minima”

(portion of negative eigen values)

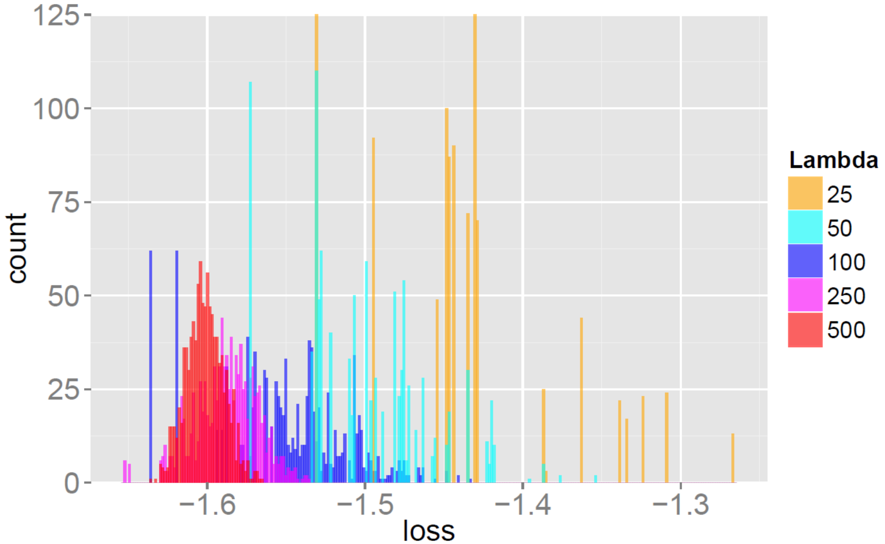


1 - “degree of local minima”
(portion of negative eigen values)

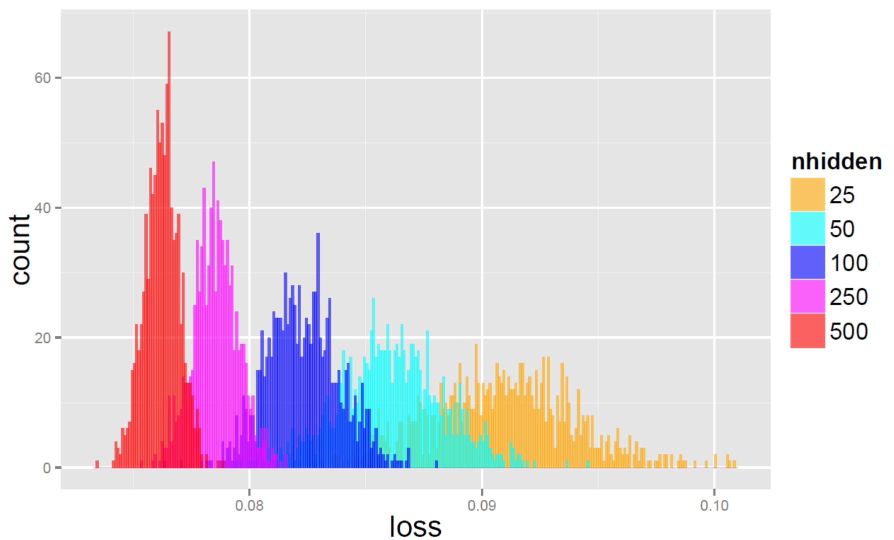
$$\alpha \propto \left(\frac{\varepsilon}{c} - 1\right)^{3/2}$$

Spin Glass v.s. Deep Learning

- Deep learning is the same as spin glass model with ***7 assumptions***.



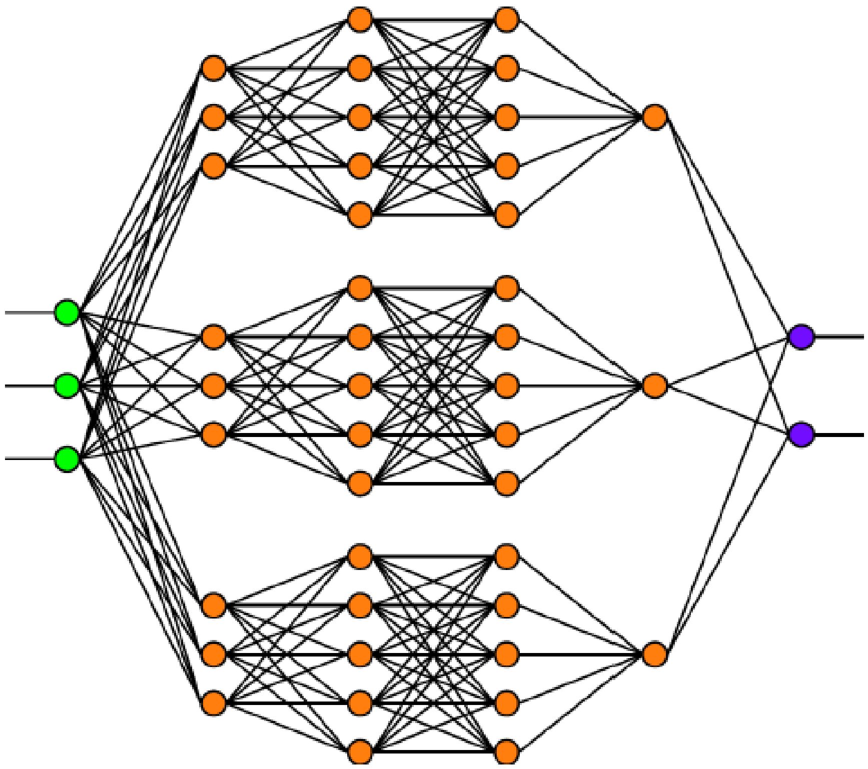
spin glass model

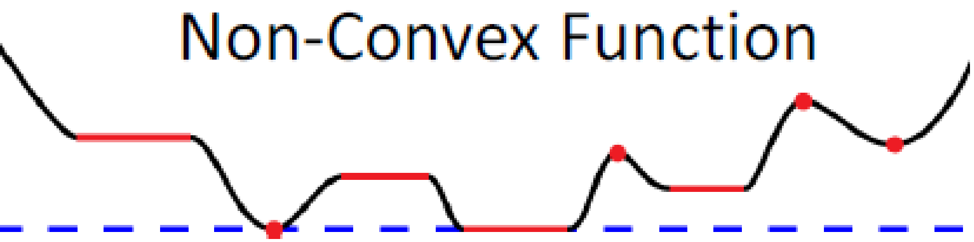


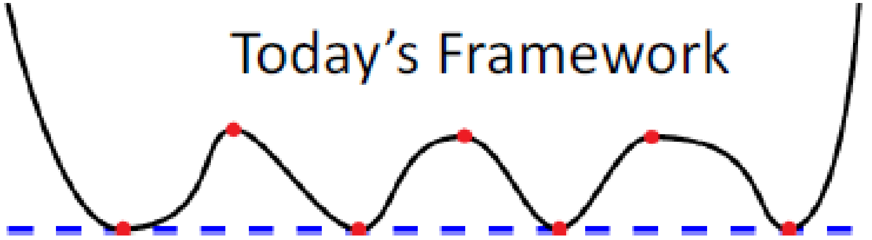
network

More Theory

- If the size of network is **large enough**, we can find global optimal by gradient descent
  - Independent to initialization





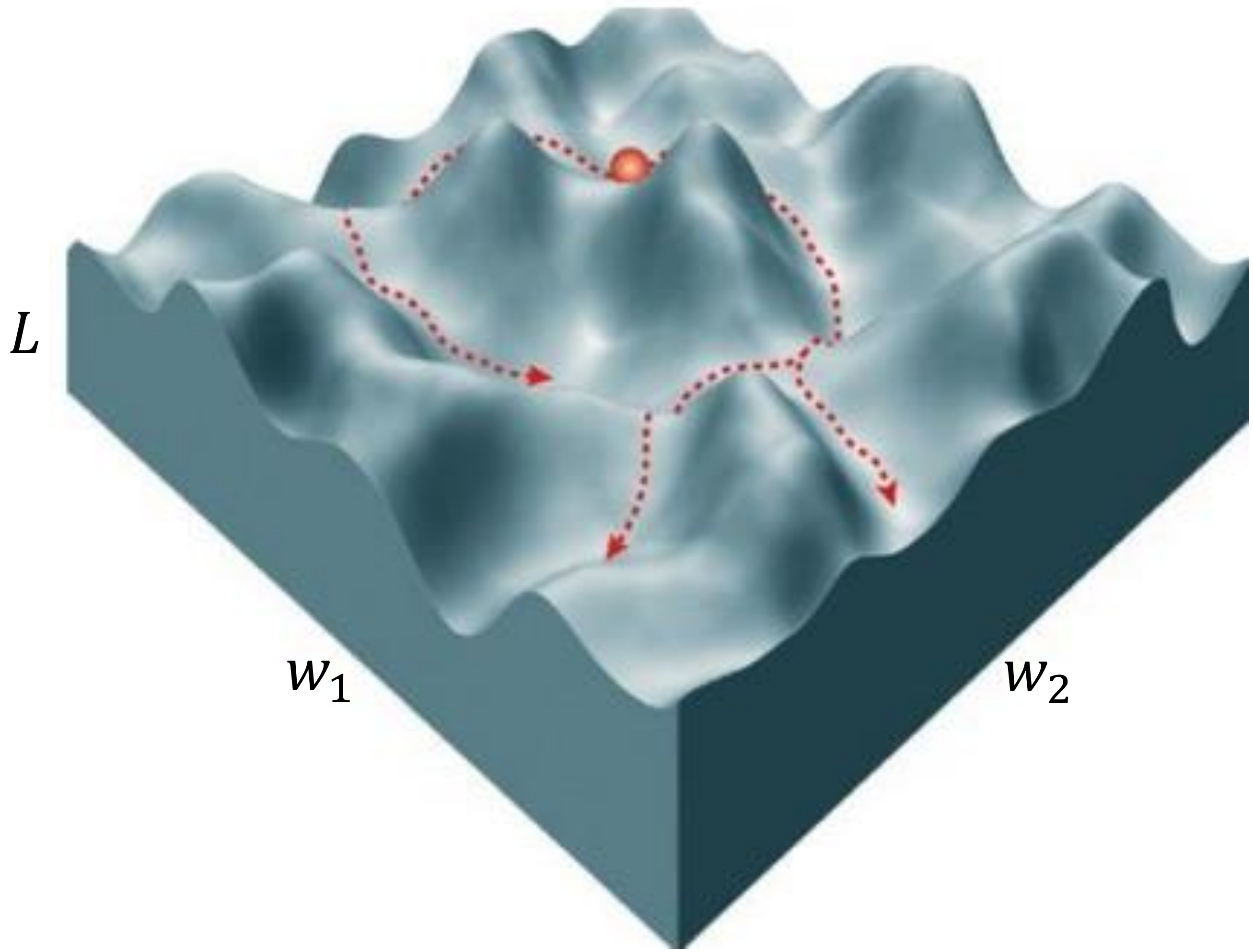


Reference

- Razvan Pascanu, Yann N. Dauphin, Surya Ganguli, Yoshua Bengio, On the saddle point problem for non-convex optimization, arXiv, 2014
- Yann Dauphin, Razvan Pascanu, Caglar Gulcehre, Kyunghyun Cho, Surya Ganguli, Yoshua Bengio, “Identifying and attacking the saddle point problem in high-dimensional non-convex optimization”, NIPS, 2014
- Anna Choromanska, Mikael Henaff, Michael Mathieu, Gérard Ben Arous, Yann LeCun, “The Loss Surfaces of Multilayer Networks”, PMLR, 2015
- Jeffrey Pennington, Yasaman Bahri, “Geometry of Neural Network Loss Surfaces via Random Matrix Theory”, PMLR, 2017
- Benjamin D. Haeffele, Rene Vidal, ” Global Optimality in Neural Network Training”, CVPR, 2017

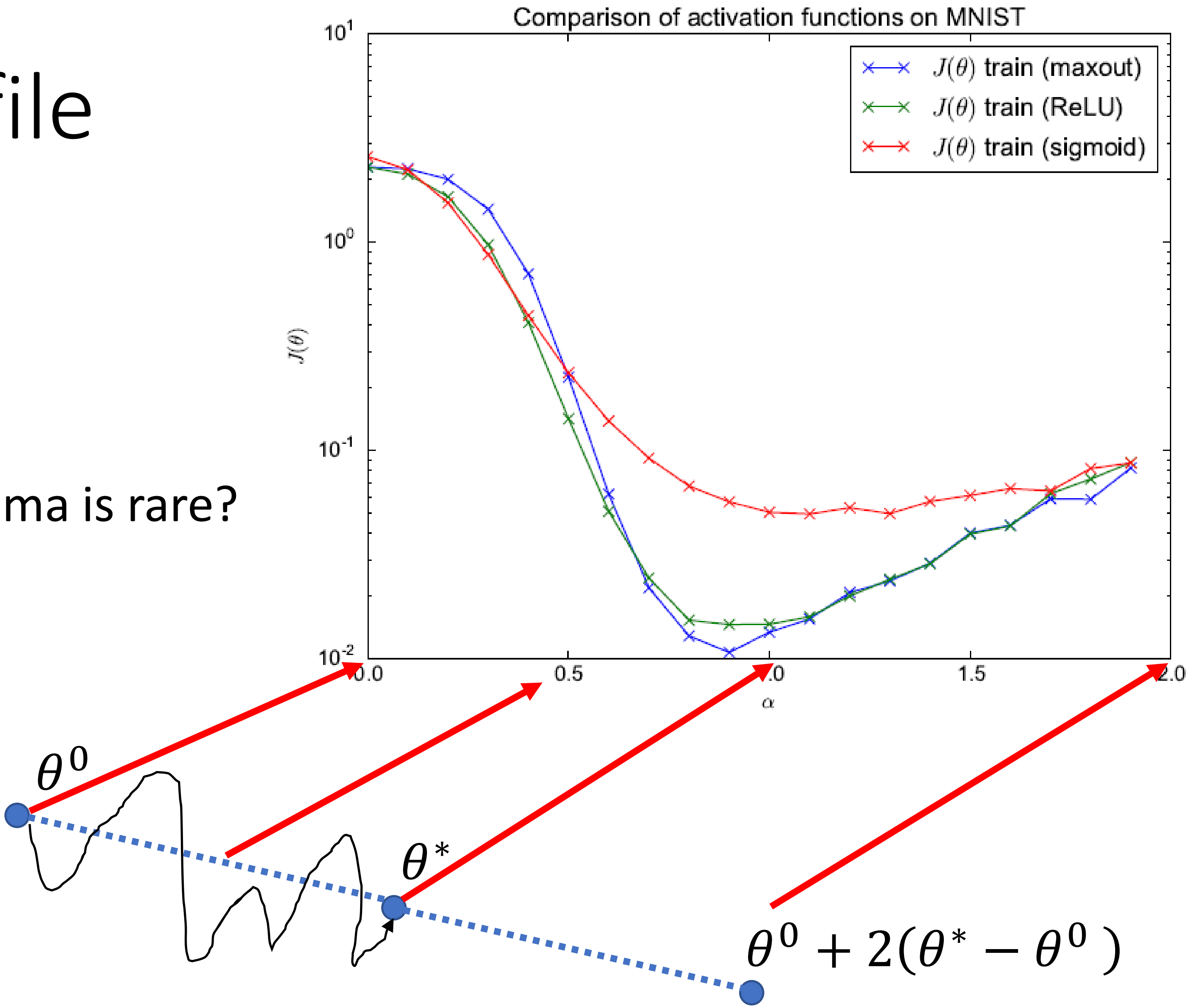
What does the Error Surface look like?

Error Surface

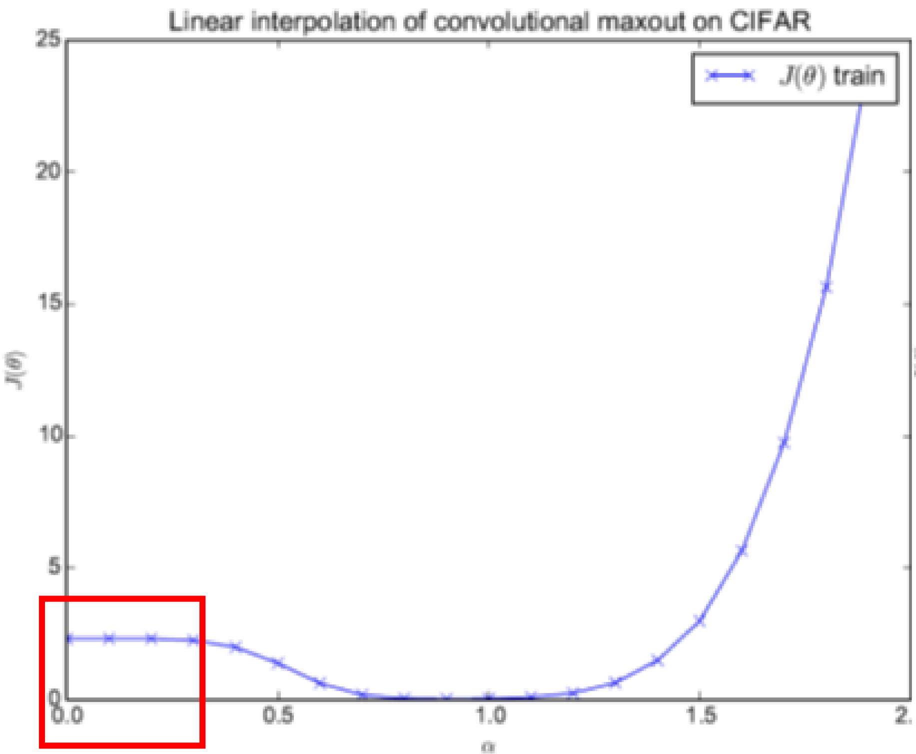


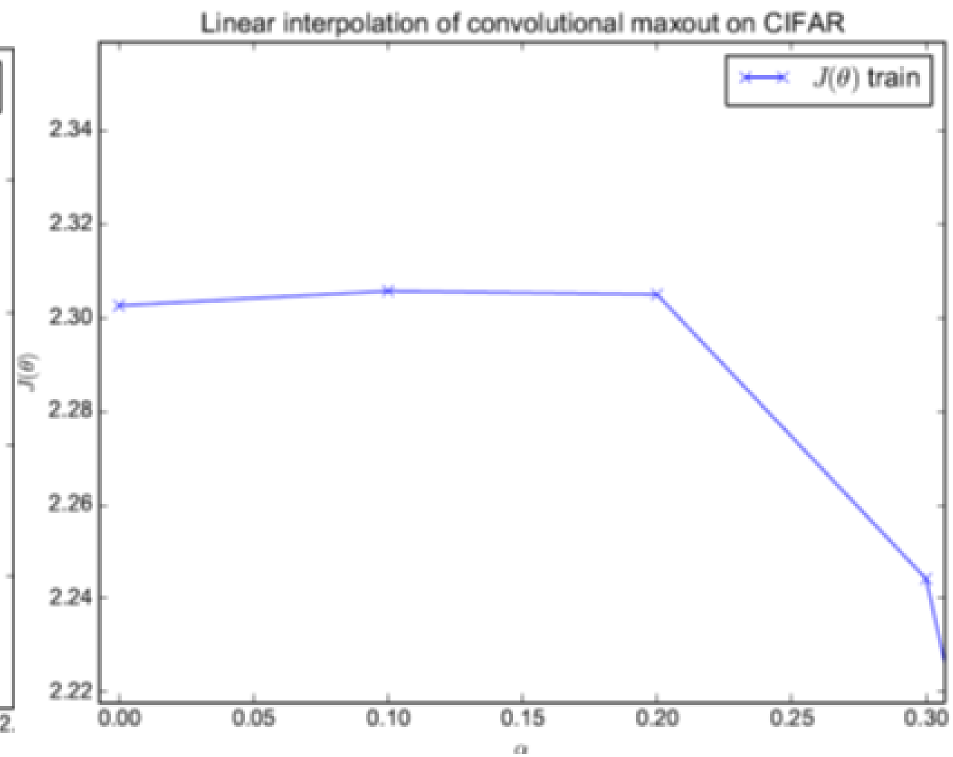
Profile

local minima is rare?

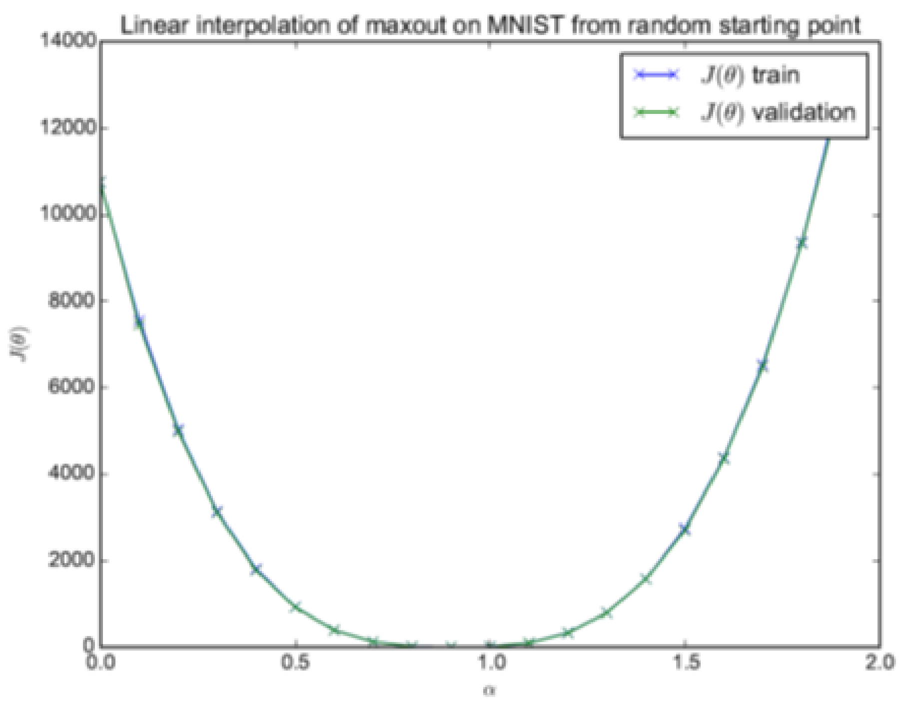


Profile

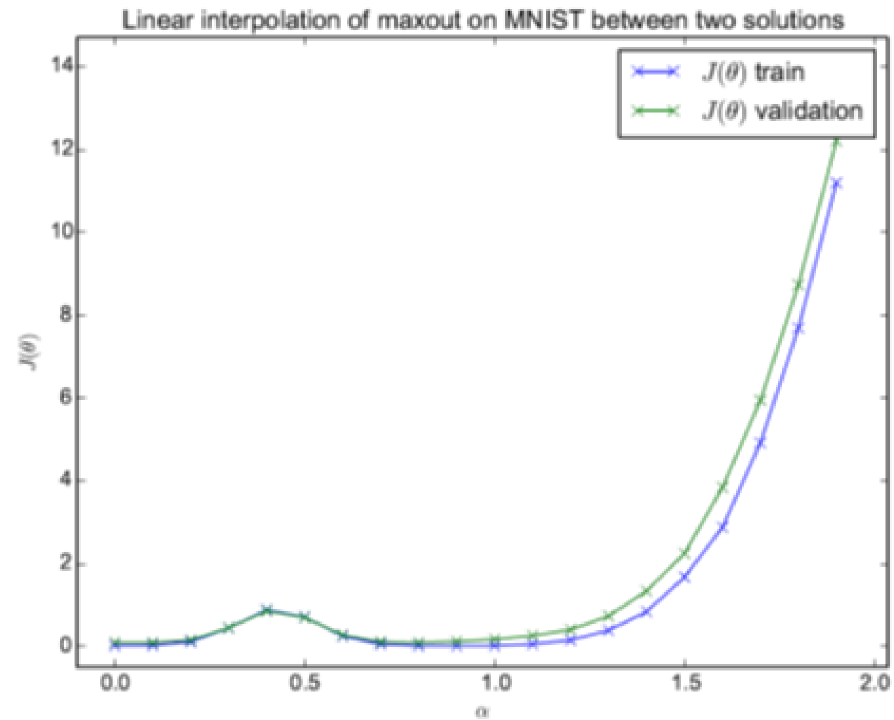

Linear interpolation of convolutional maxout on CIFAR
$J(\theta)$ train
$J(\theta)$
α
0
5
10
15
20
25
0.0
0.5
1.0
1.5
2.


Linear interpolation of convolutional maxout on CIFAR
$J(\theta)$ train
$J(\theta)$
α
2.22
2.24
2.26
2.28
2.30
2.32
2.34
0.00
0.05
0.10
0.15
0.20
0.25
0.30

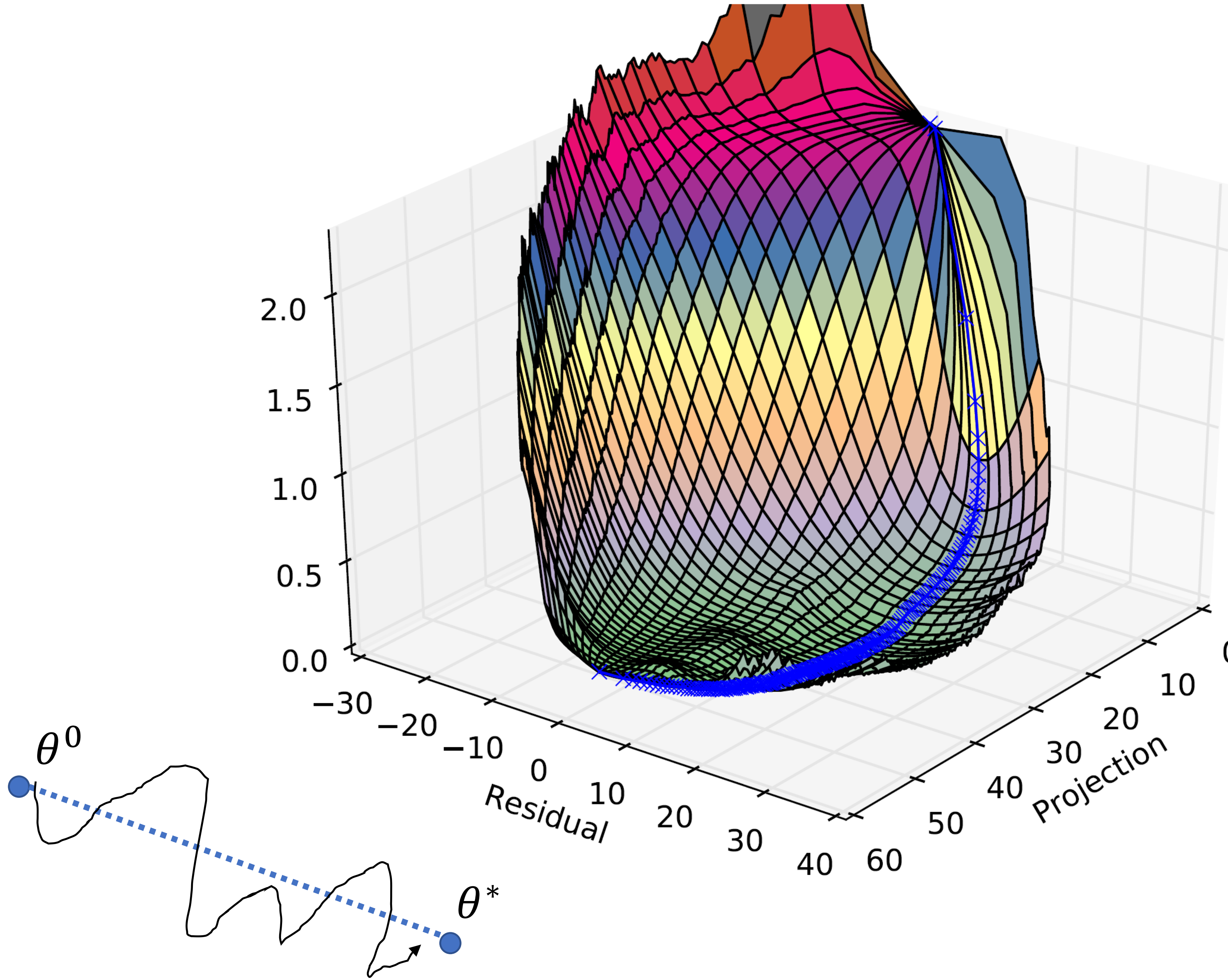
Profile

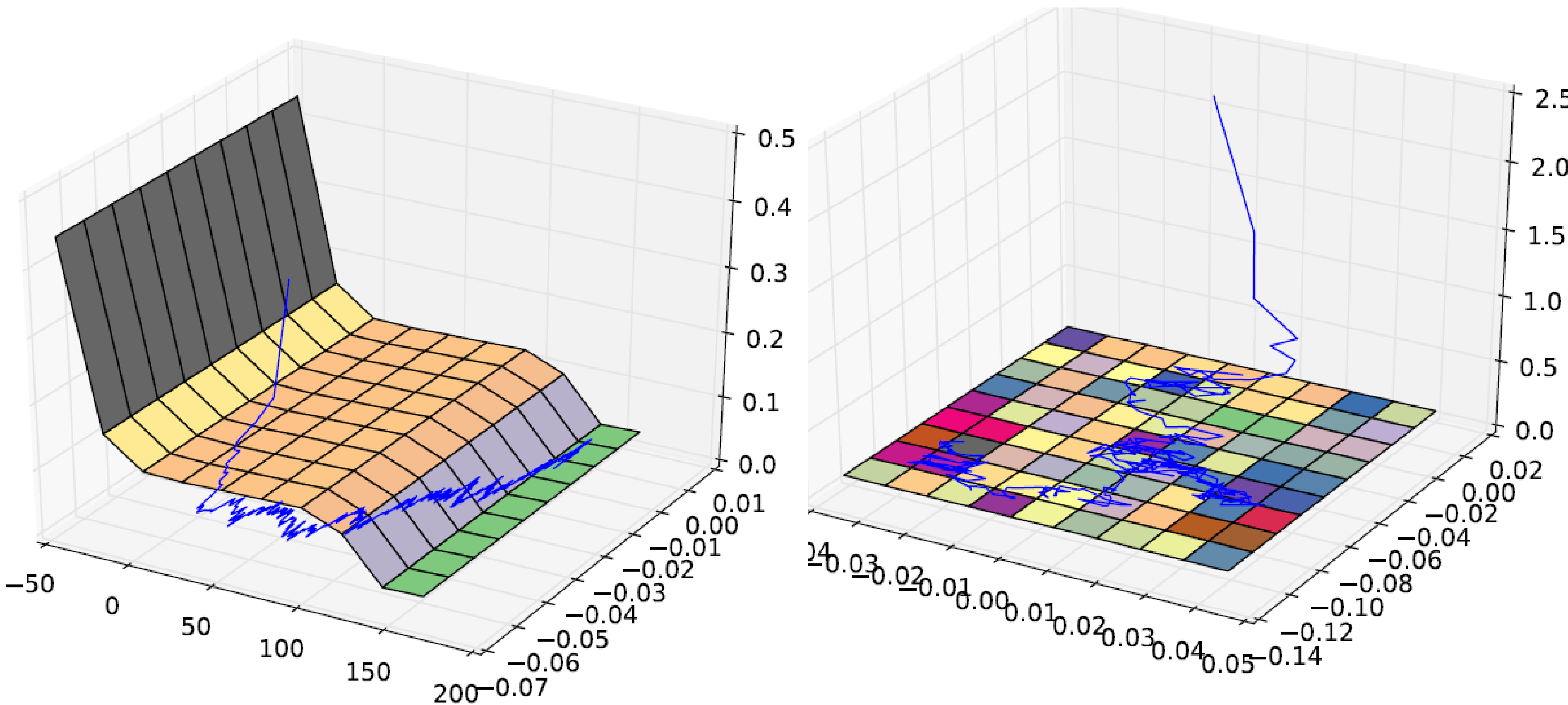


two random starting points

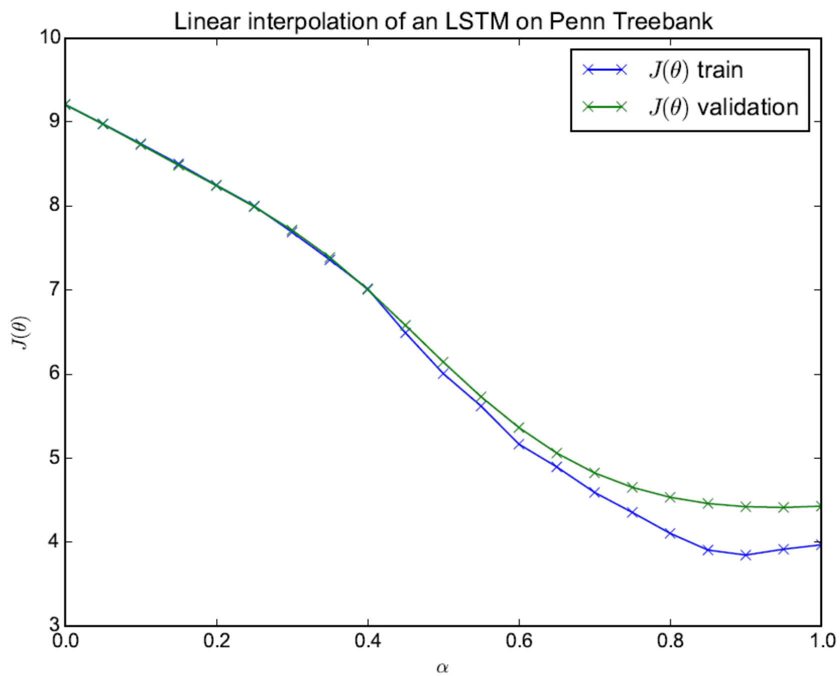


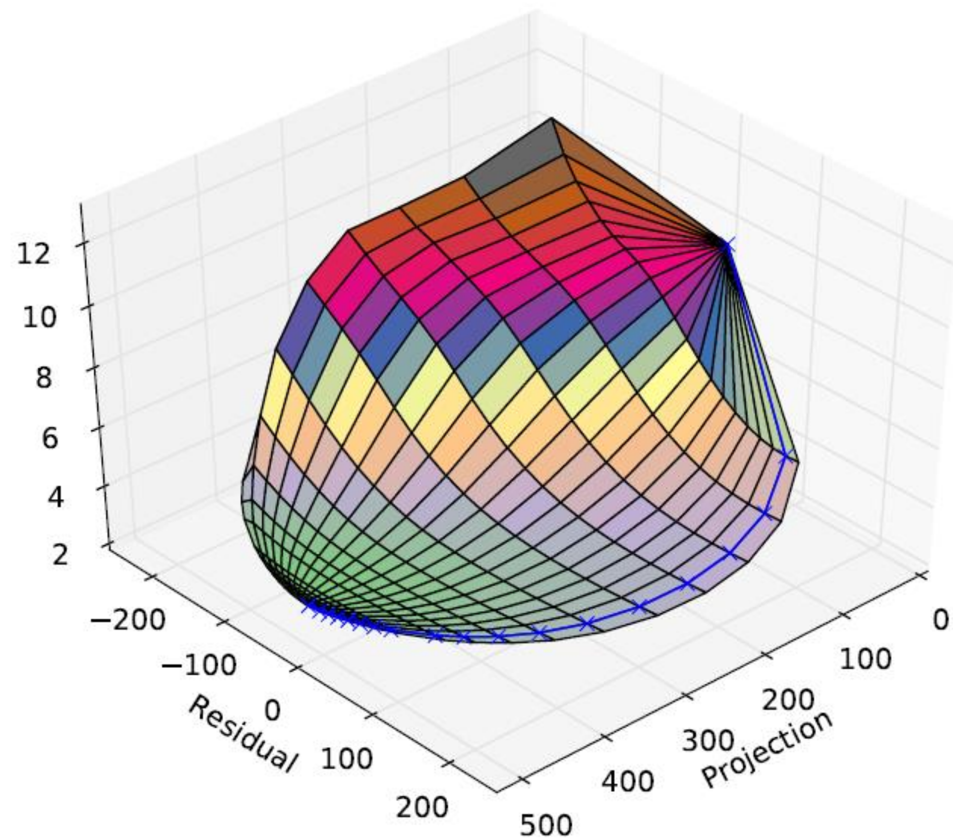
two “solutions”


2.0
1.5
1.0
0.5
0.0
−30
−20
−10
0
10
20
30
40
Residual
0
10
20
30
40
50
60
Projection
θ^0
θ^*


0.5
0.4
0.3
0.2
0.1
0.0
−50
0
50
100
150
200
0.01
0.00
−0.01
−0.02
−0.03
−0.04
−0.05
−0.06
−0.07
2.5
2.0
1.5
1.0
0.5
0.0
0.02
0.00
−0.02
−0.04
−0.06
−0.08
−0.10
−0.12
−0.14
−0.03
−0.02
−0.01
0.00
0.01
0.02
0.03
0.04
0.05

Profile - LSTM



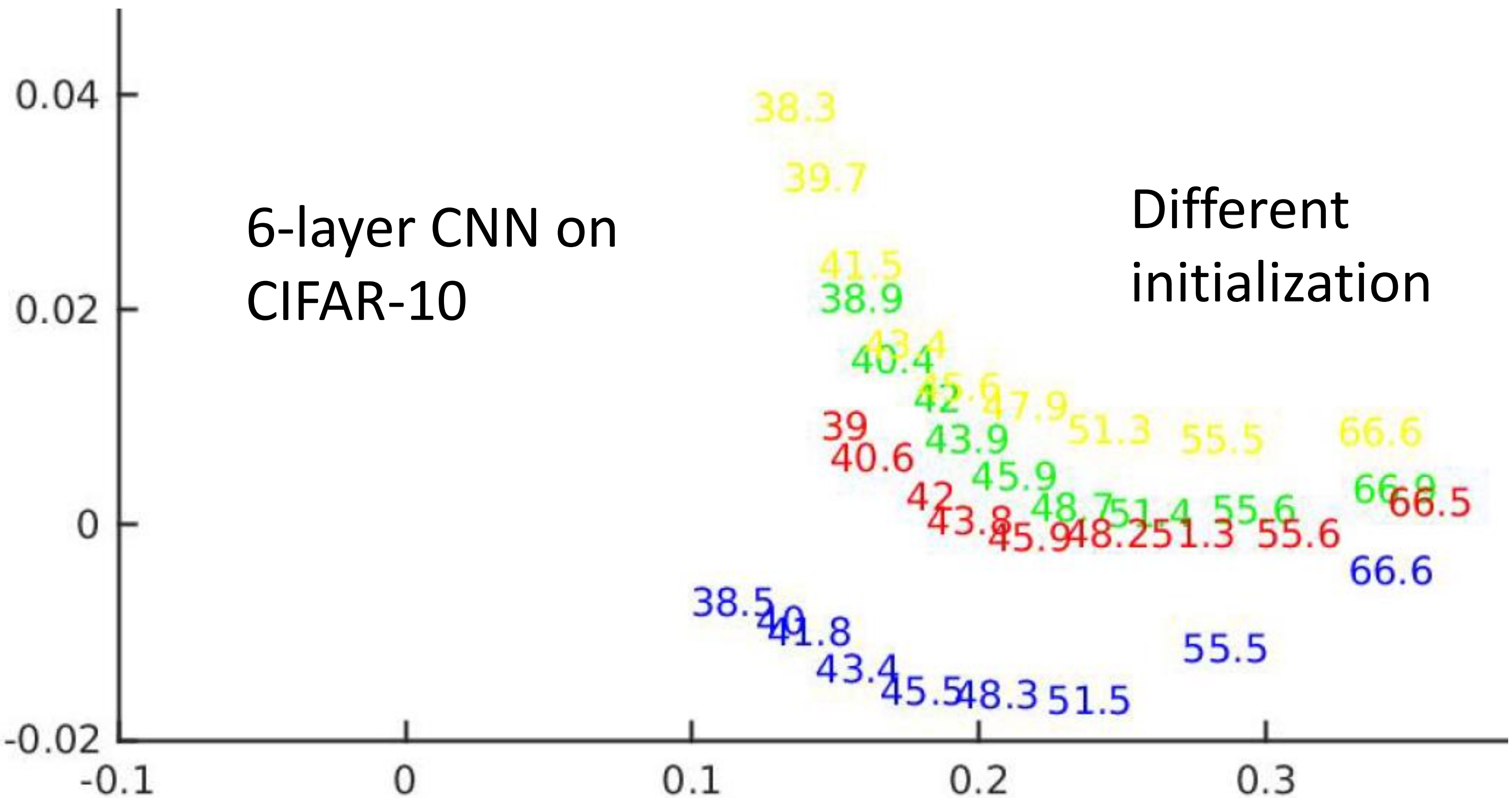


Training Processing

Different initialization / different strategies usually lead to similar loss (there are some exceptions).

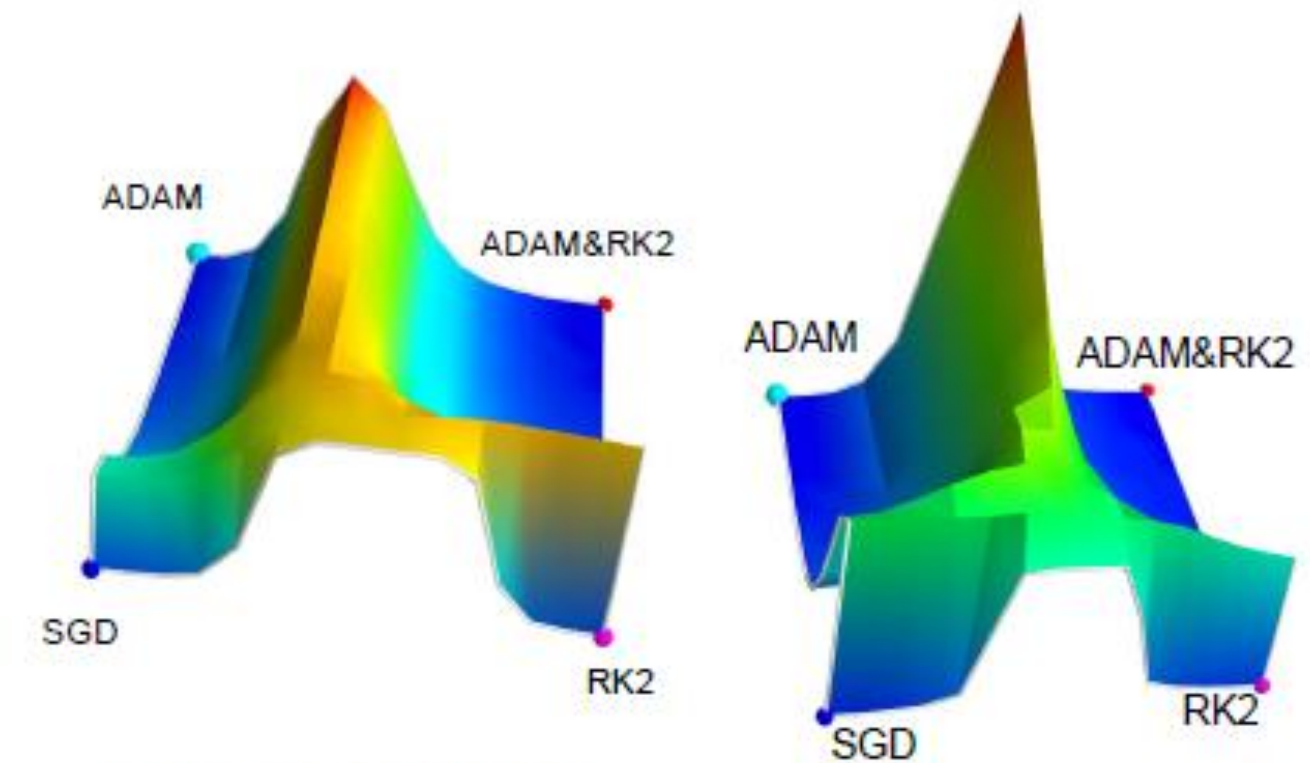
6-layer CNN on CIFAR-10

Different initialization



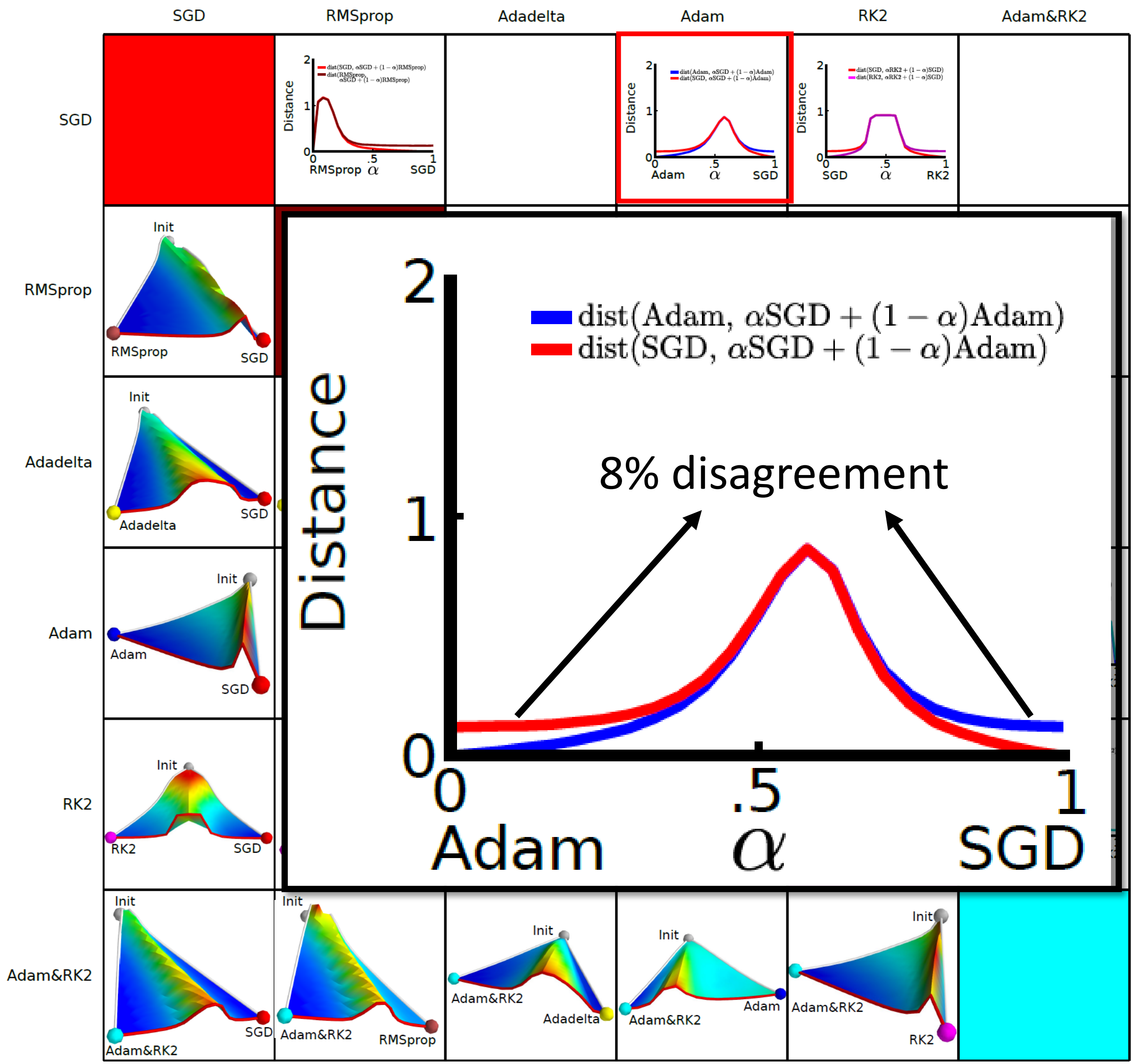
Training Processing

- Different strategies (the same initialization)



(a) NIN, CIFAR10

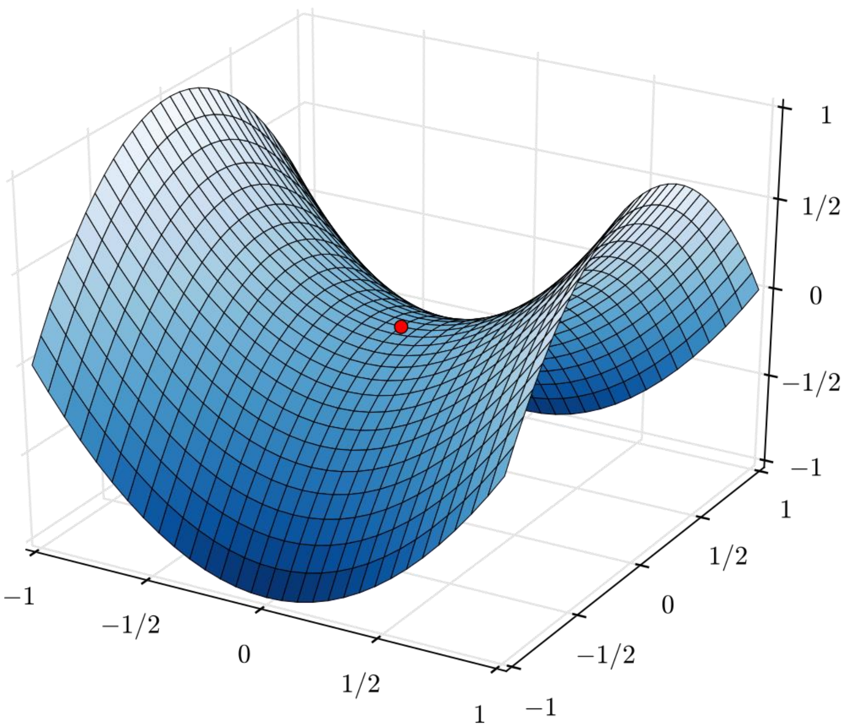
(b) VGG, CIFAR10



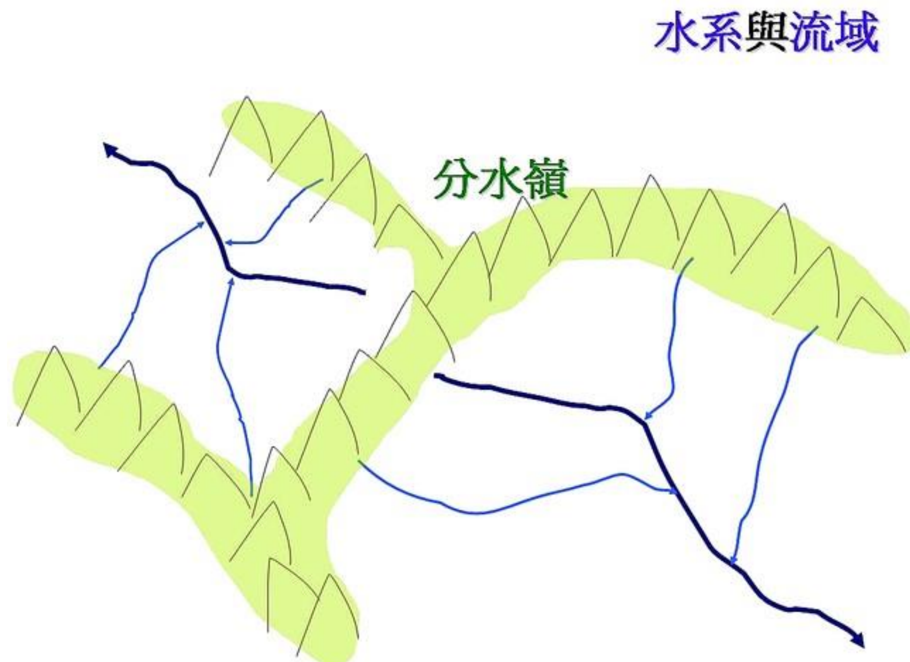
(c) VGG, CIFAR10

http://mypaper.pchome.com.tw/ccschoolgeo/post/1311484084

Training Processing



何時分道揚鑣？

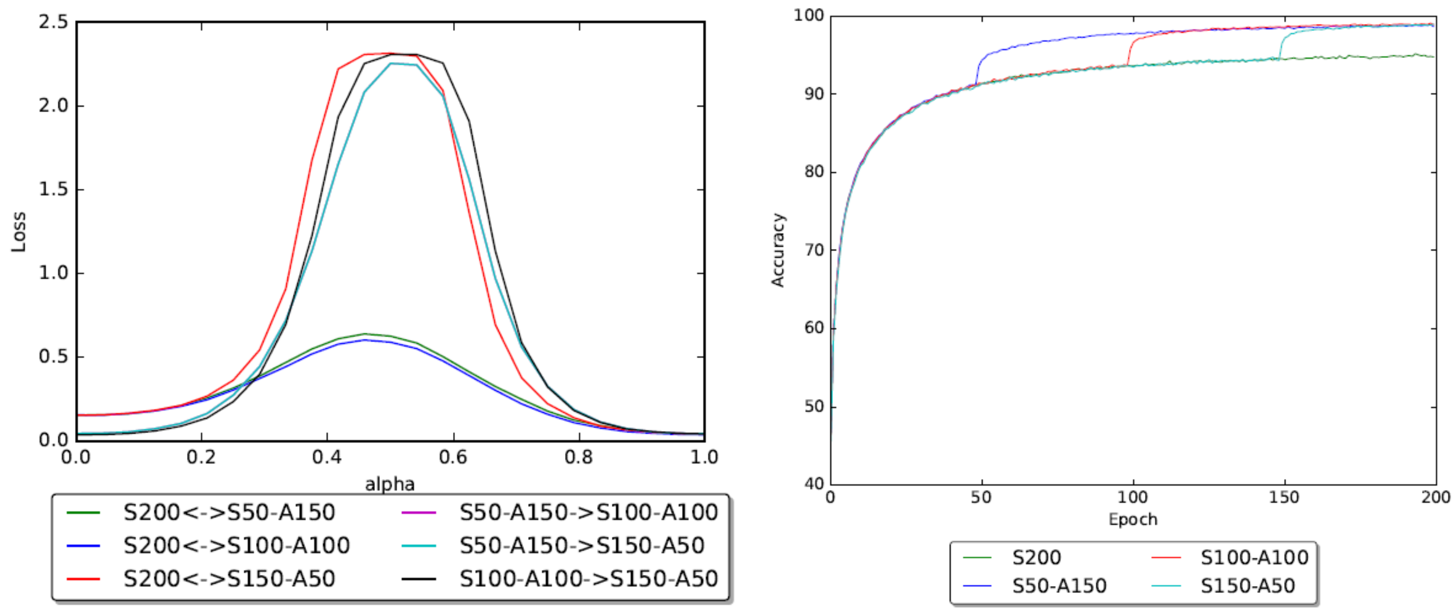


Different training strategies

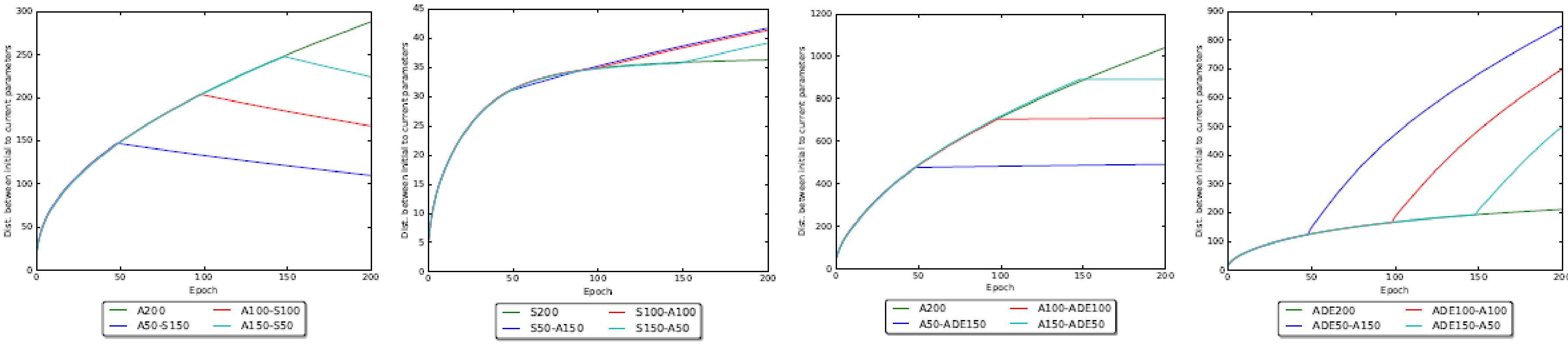
Different basins

Training Processing

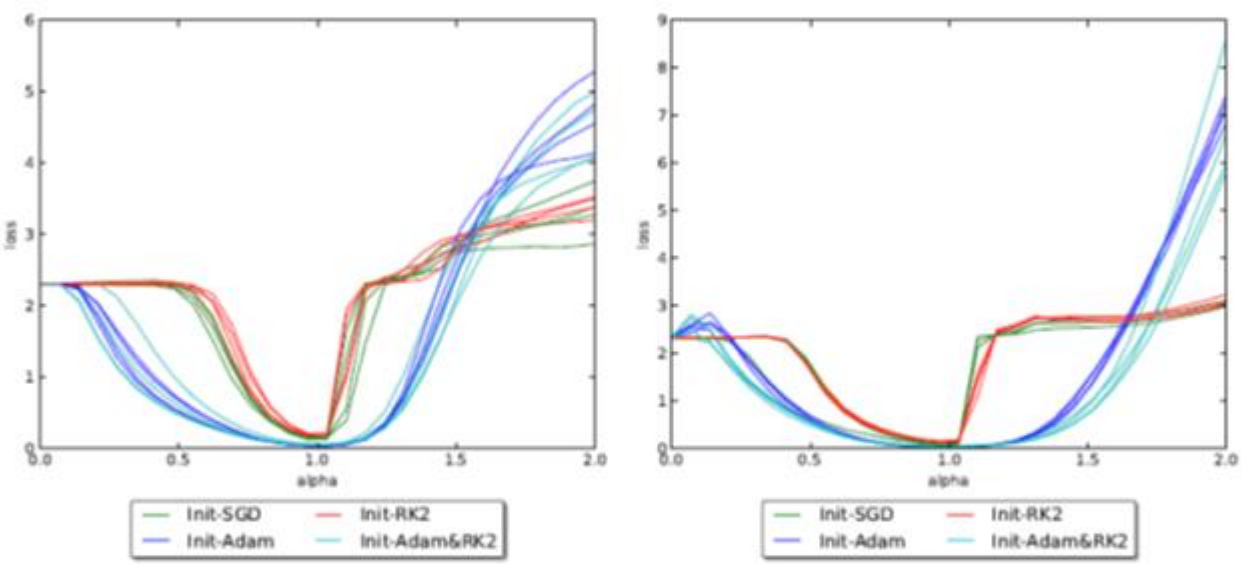
- Training strategies make difference at all stages of training



(b) NIN: Switching from SGD (S, $\eta = .1$) to Adam (A, $\eta = .0001$).



(a) (b) (c) (d)

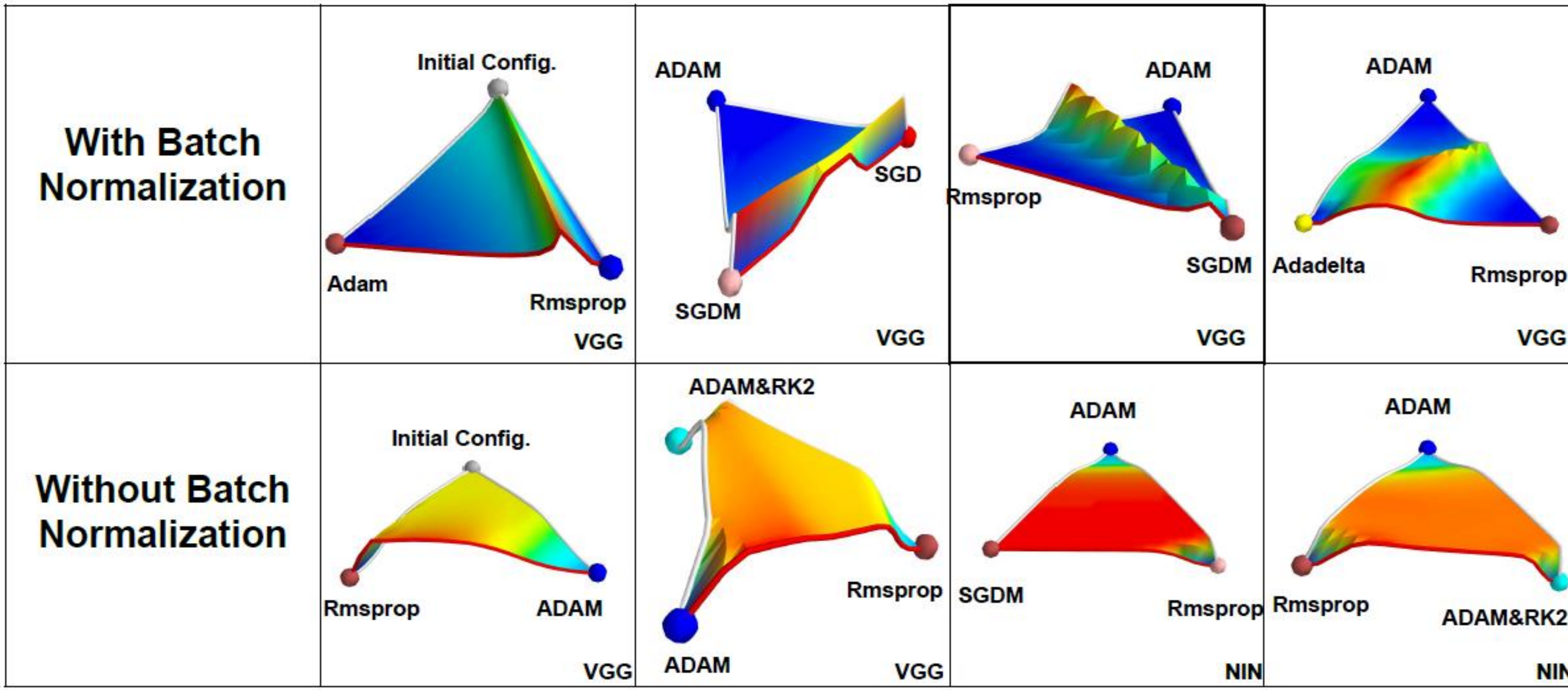


(a) CIFAR10, NIN

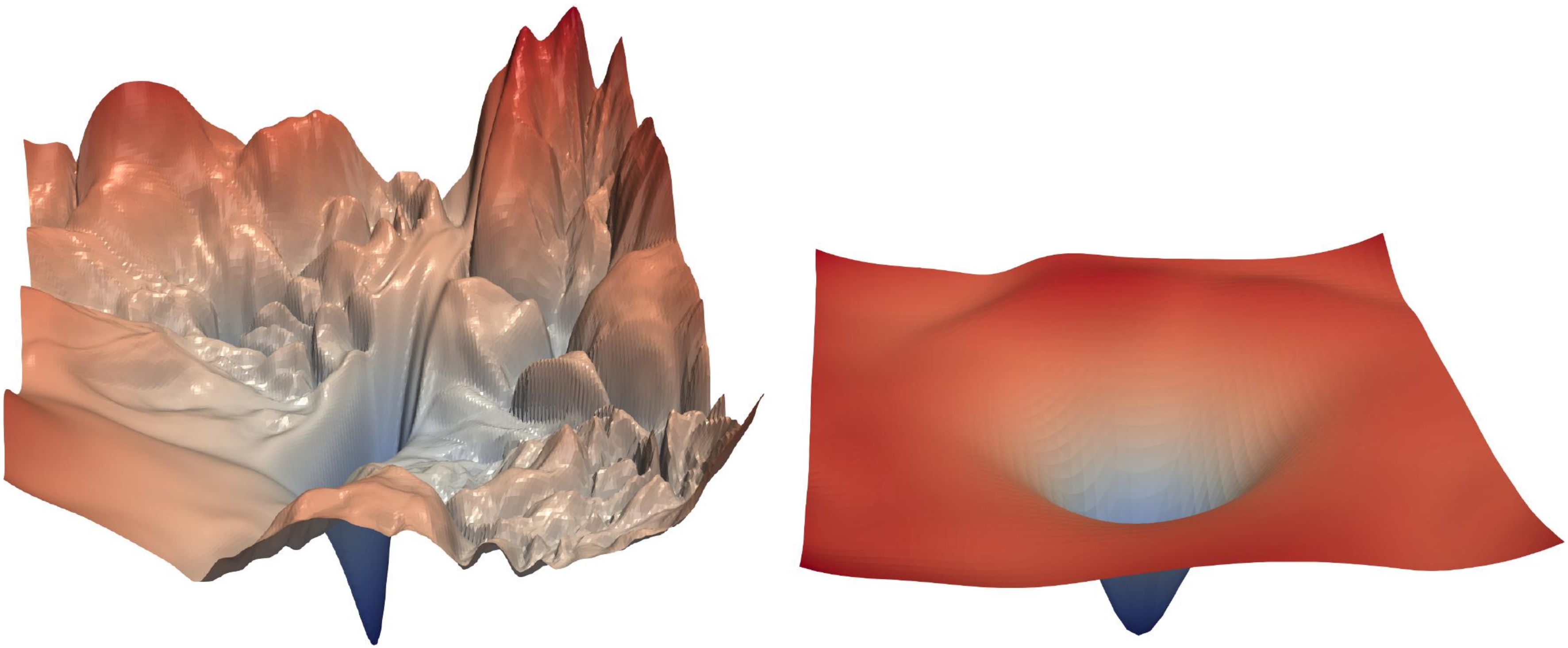
(b) CIFAR10, VGG

Larger basin for Adam

Batch Normalization



Skip Connection



(a) without skip connections (b) with skip connections

Figure 1: The loss surfaces of ResNet-56 with/without skip connections. The vertical axis is logarithmic to show dynamic range. The proposed filter normalization scheme is used to enable comparisons of sharpness/flatness between the two figures.

Reference

- Ian J. Goodfellow, Oriol Vinyals, Andrew M. Saxe, “Qualitatively characterizing neural network optimization problems”, ICLR 2015
- Daniel Jiwoong Im, Michael Tao, Kristin Branson, “An Empirical Analysis of Deep Network Loss Surfaces”, arXiv 2016
- Qianli Liao, Tomaso Poggio, “Theory II: Landscape of the Empirical Risk in Deep Learning”, arXiv 2017
- Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, Tom Goldstein, “Visualizing the Loss Landscape of Neural Nets”, arXiv 2017

Concluding Remarks

Concluding Remarks

- Deep linear network is not convex, but all the local minima are global minima.
    - There are saddle points which are hard to escape
- Deep network has local minima.
    - We need more theory in the future
- Conjecture:
    - When training a larger network, it is rare to meet local minima.
    - All local minima are almost as good as global
- We can try to understand the error surface by visualization.
    - The error surface is not as complexed as imagined.