Noncommutative Polynomials of Random Matrices in Deep Neural Networks

Algebraic Aspect of Random Matrices, CIRM

Tomohiro Hayase

October 7, 2024

Benoit Collins & TH (CIMP2023, arXiv:2103.13466)
Ryo Karakida & TH (AISTATS2021, arXiv:2006.07814)

Introduction

MLP

Let $n_0, n_1, \ldots, n_L \in \mathbb{N}$.
Parameters:

$$\theta = (W_\ell, b_\ell)_{\ell=1,\ldots,L},\ W_\ell \in \mathbb{R}^{n_\ell \times n_{\ell-1}},\ b_\ell \in \mathbb{R}^{n_\ell}.$$

Forward propagation: for $x \in \mathbb{R}^{n_0}$, set $x_0 = x$ and inductively

$$h_\ell = W_\ell x_{\ell-1} + b_\ell,\ x_\ell = \phi(h_\ell) := \phi(h_{\ell,i})_{i\in[n_\ell]}.$$

Finally, define the output by $f_\theta(x) = h_L$.

ϕ: Activation Function

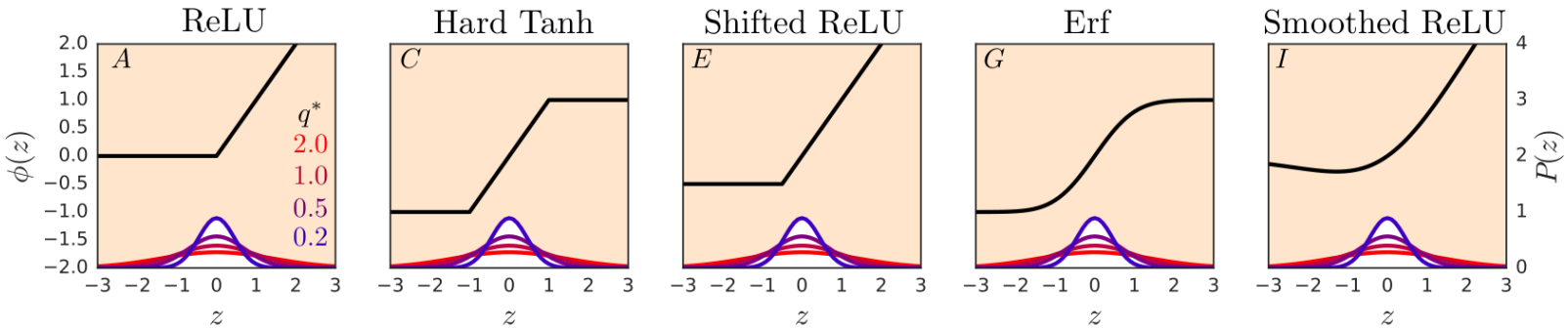


Figure: Examples of activation functions.

Deep Learning

Generally, a standard formulation of supervised deep learning is as follows:

1. We are given a finite set of input/output data pairs $(x, y) \in \mathcal{D}$.
2. We are given a *deep neural network* (DNN), a composition of parameterized transformations that maps a real vector to a real vector.
3. We are given an *object function*: e.g. mean squared loss:

$$L(x, y, \theta) = \frac{1}{2n_L} \sum_{j=1}^{n_L} (f_\theta(x)_j - y_j)^2.$$

Optimization

We minimize the loss function by gradient descent:

$$\theta_{t+1} = \theta_t - \eta_t \frac{\partial}{\partial \theta} L(x, y, \theta_t)$$

Initialization of Parameters and Random Matrices

e.g. Gaussian (Ginibre) random matrix:

$$(W_\ell)_{i,j} \sim \mathcal{N}(0, \sigma_w^2/N), \text{ i.i.d.}$$

e.g. Haar distributed orthogonal matrix:

$$W_\ell = \sigma_w O, O \sim \text{Haar Orthogonal Prob.}$$

The Infinite-dimensional Limit is Gaussian

By taking the infinite-dimensional limit ("wide limit") of hidden layers, output becomes Guassian.

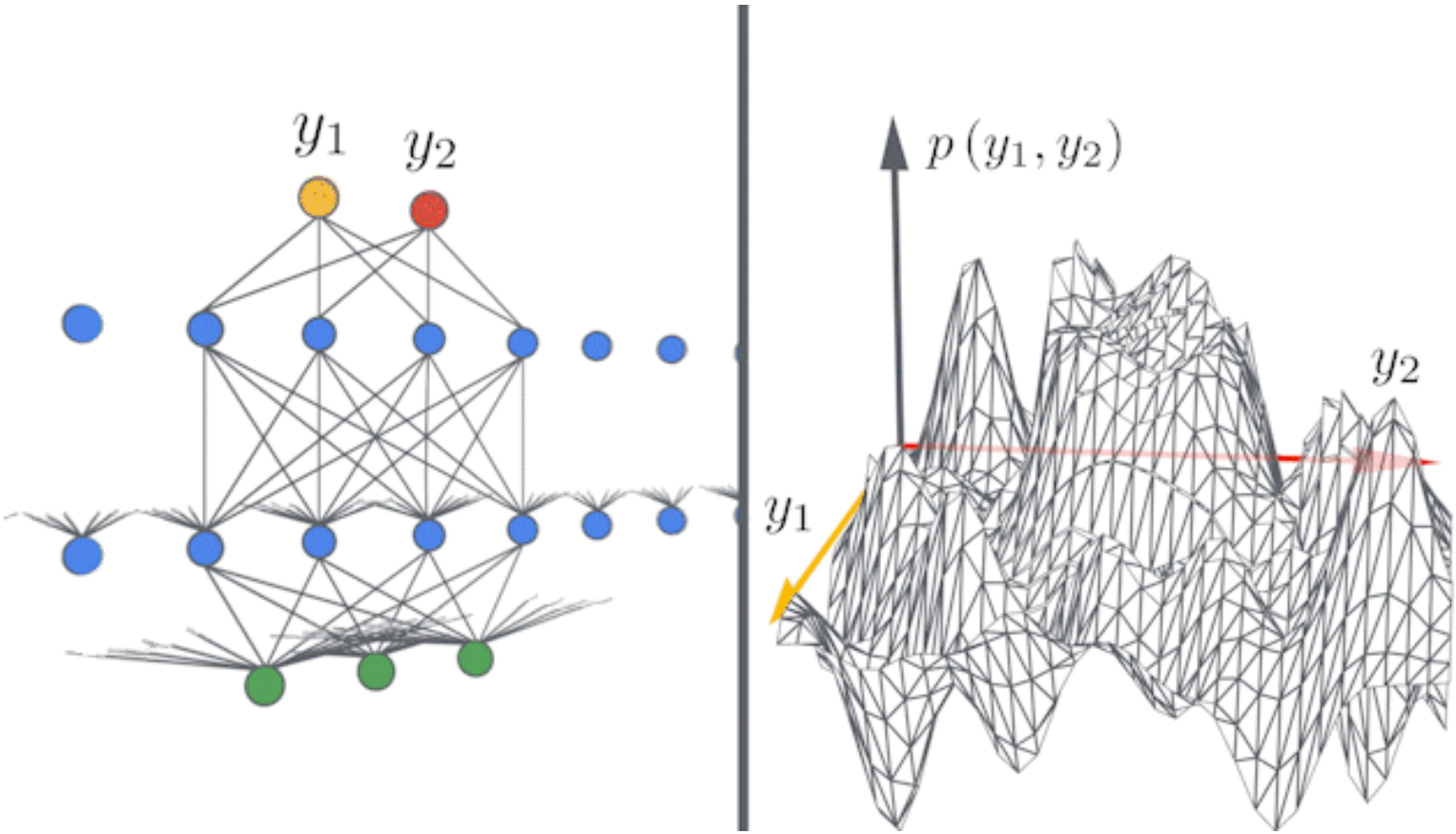


Figure: `https://ai.googleblog.com/2020/03/fast-and-easy-infinitely-wide-networks.html`

The Infinite-dimensional Limit is Gaussian

By taking the infinite-dimensional limit ("wide limit") of hidden layers, output becomes Guassian.

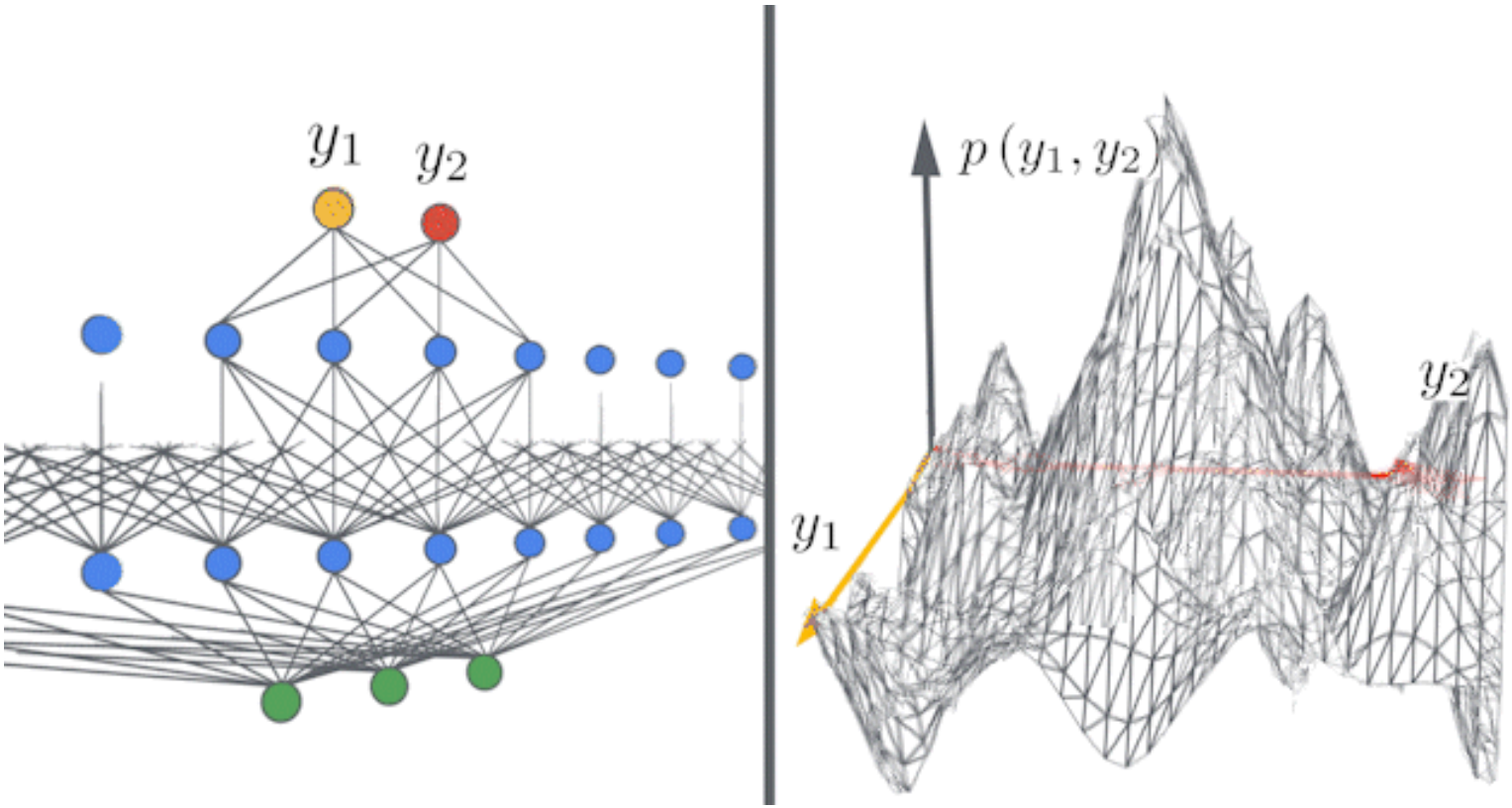


Figure: `https://ai.googleblog.com/2020/03/fast-and-easy-infinitely-wide-networks.html`

The Infinite-dimensional Limit is Gaussian

By taking the infinite-dimensional limit ("wide limit") of hidden layers, output becomes Guassian.

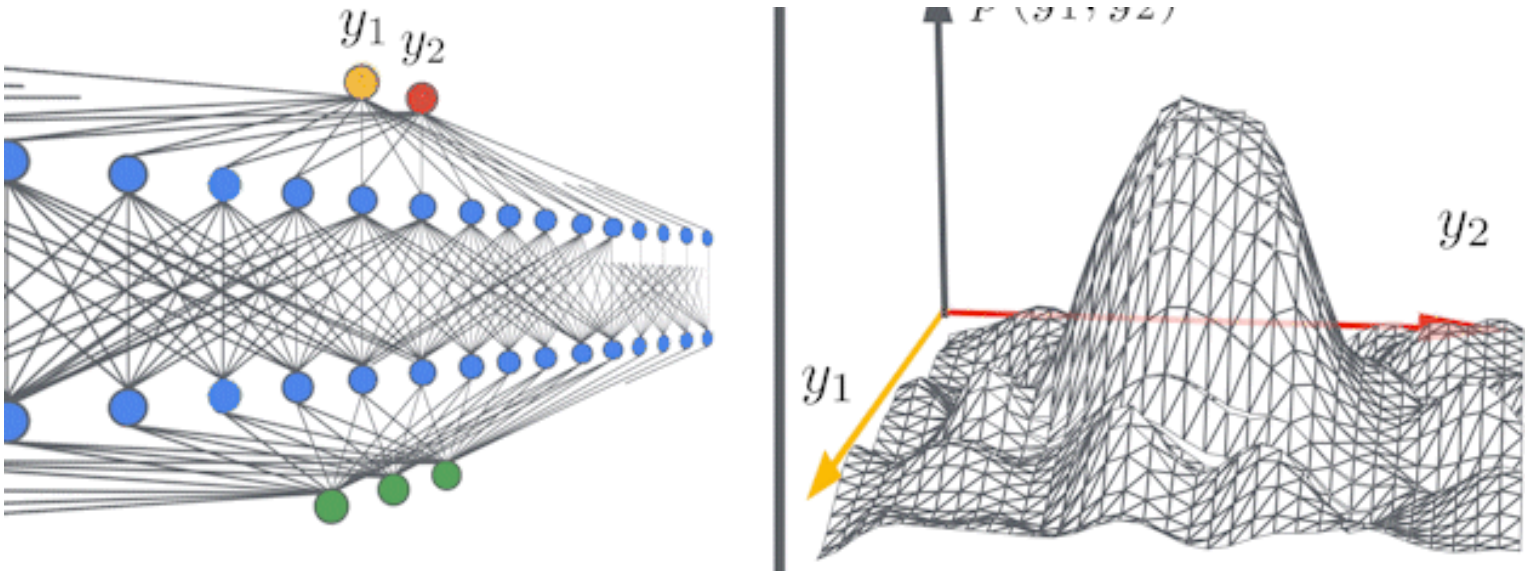


Figure: `https://ai.googleblog.com/2020/03/fast-and-easy-infinitely-wide-networks.html`

The Infinite-dimensional Limit is Gaussian

By taking the infinite-dimensional limit ("wide limit") of hidden layers, output becomes Guassian.

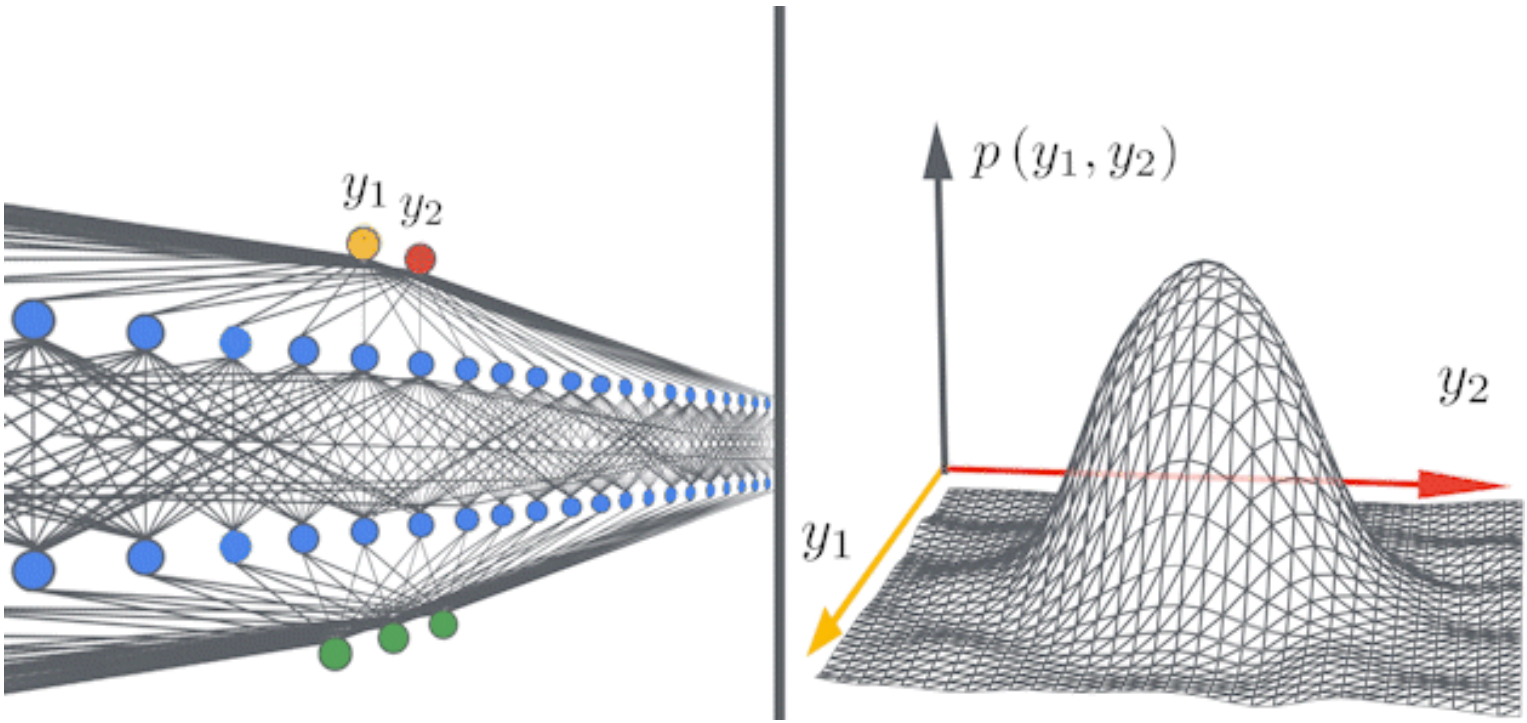


Figure: `https://ai.googleblog.com/2020/03/fast-and-easy-infinitely-wide-networks.html`

The Infinite-dimensional Limit is Gaussian

By taking the infinite-dimensional limit ("wide limit") of hidden layers, output becomes Guassian.

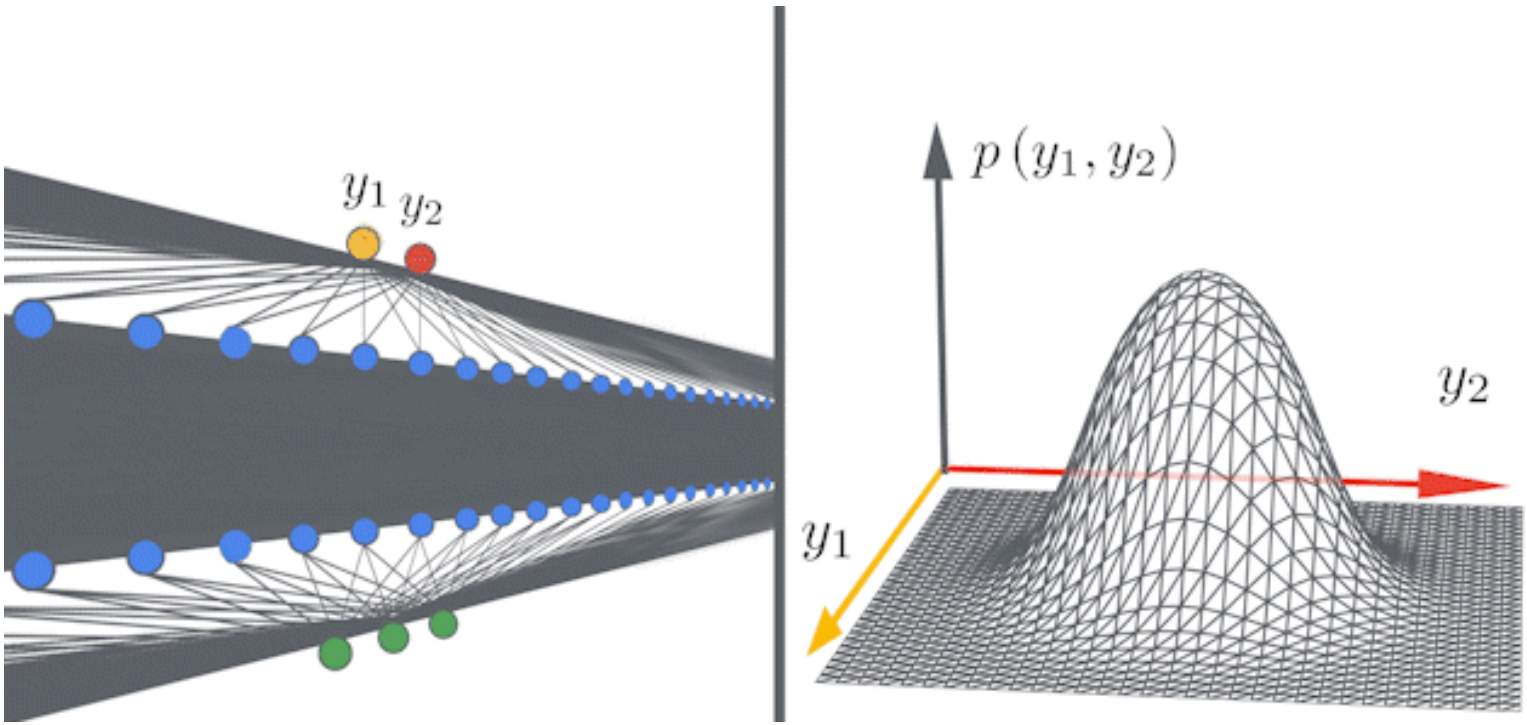


Figure: `https://ai.googleblog.com/2020/03/fast-and-easy-infinitely-wide-networks.html`

Neural Network Gaussian Process (NNGP)

Consider two inputs x, x' and corresponding hidden units x_ℓ, x'_ℓ and h_ℓ, h'_ℓ in MLP.
Taking an infinite dimensional limit at the initial state, we have [Lee+ICLR2018]

$$(h_\ell, h'_\ell) \sim \mathcal{N}(0, \sigma_w^2 K_\ell(x, x') + \sigma_b^2)$$

where

$$K_\ell(x, x') := \lim_{n_\ell \to \infty} \frac{1}{n_\ell} \sum_{j=1}^{n_\ell} x_{\ell,j} x'_{\ell,j}.$$

We have the following **Kernel Propagation**:

$$K_{\ell+1}(x, x') = \int \phi(z_1)\phi(z_2)p_{\mathcal{N}}(z)dz,$$

where

$$p_{\mathcal{N}} = \mathcal{N}(0, \sigma_w^2 \begin{pmatrix} K_\ell(x, x) & K_\ell(x, x') \\ K_\ell(x, x') & K_\ell(x', x') \end{pmatrix} + \sigma_b^2).$$

For some activation functions, we can compute the integral explicitly.

Applicatoin: NNGP Estimation

Generally, consider B samples. Set
$X = (x(a))_{a=1,\dots,B}$, $Y = (y(a))_{a=1,\dots,B}$ be input/output samples.

$$K(x^*, X) := (K_L(x^*, x(a)))_{n=1,\dots,a} \in \mathbb{R}^B$$

$$K(X, X) := (K_L(x(a), x(b)))_{a,b} \in M_B(\mathbb{R})$$

Then the posterior mean/ var is given by the following: for a new input x^*,

$$m(y^*) = K(x^*, X)K(X, X)^{-1}Y$$

$$v(y^*) = K(x^*, x^*) - K(x^*, X)K(X, X)^{-1}K(x^*, X)$$

[Lee et.al., Deep Neural Networks as Gaussian Process, ICLR 2018]

2. Jacobian

Vanishing/Exploding Gradients

The optimization of DNN needs its parameter derivations.
Since a DNN is a function composition, the chain rule computes the parameter derivations. The input-output Jacobian is defined as

$$J = \frac{\partial f_\theta(x)}{\partial x} = \frac{\partial h_L}{\partial x}$$

In the case of MLP, we have

$$J = W_L D_{L-1} \dots W_2 D_1 W_1,$$

where

$$D_\ell = \frac{\partial x_\ell}{\partial h_\ell} = \mathrm{diag}(\phi'(h_{\ell,1}), \dots, \phi'(h_{\ell,n_\ell}))$$

Dynamical Isometry

A DNN is said to achieve *dynamical isometry* If the eigenvalue distribution is concentrated around one.

Dynamical Isometry

A DNN is said to achieve *dynamical isometry* If the eigenvalue distribution is concentrated around one.
Dynamical Isotmetry prevents the exploding/vanishing gradients.
If we set the parameter initialization to be Haar orthogonal and choose the appropriate activation function, then we can make the DNN to achieve the dynamical isometry
[Pennington+(AISTATS2018)].

Dynamical Isometry

A DNN is said to achieve *dynamical isometry* If the eigenvalue distribution is concentrated around one.
Dynamical Isotmetry prevents the exploding/vanishing gradients.
If we set the parameter initialization to be Haar orthogonal and choose the appropriate activation function, then we can make the DNN to achieve the dynamical isometry [Pennington+(AISTATS2018)].
Set μ_L, ν be limit spectral distributions of $JJ^\top, D^2$ as wide limits respectively.
Under the assumption of the **asymptotic freeness of Jacobians** (I'll touch this later section!), it holds that

$$\mu_L = [(\sigma^2 \cdot)_* \nu]^{\boxtimes L}$$

where $\boxtimes$ is the free multiplicative convolution.

Distribution of $D = \mathrm{diag}\phi'$

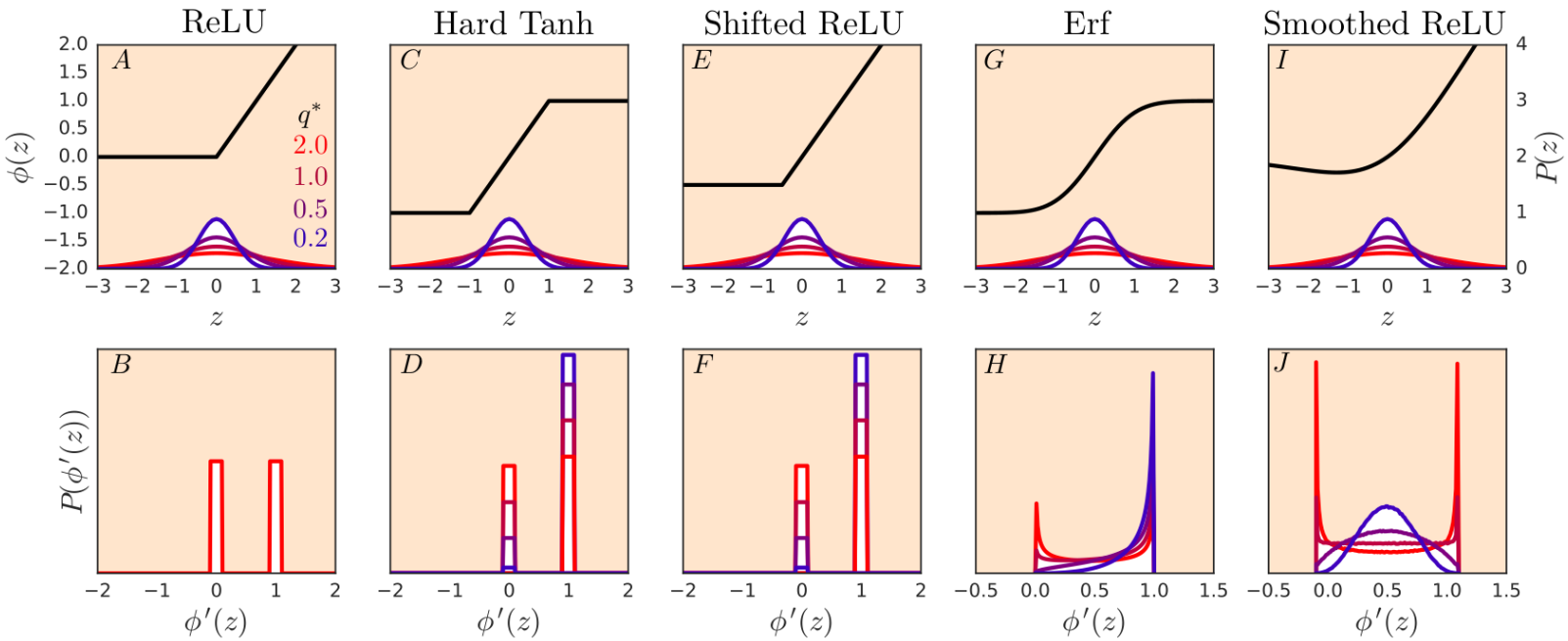


Figure: Distribution of $\phi'(z)$ with several input variance $z \sim N(0, q^*)$ with $q^* = 0.2, 0.5, 1.0, 2.0$. (Figure from PSG(AISTATS2018).)

The Limit Spectral Distribution μ_L of $JJ^\top$

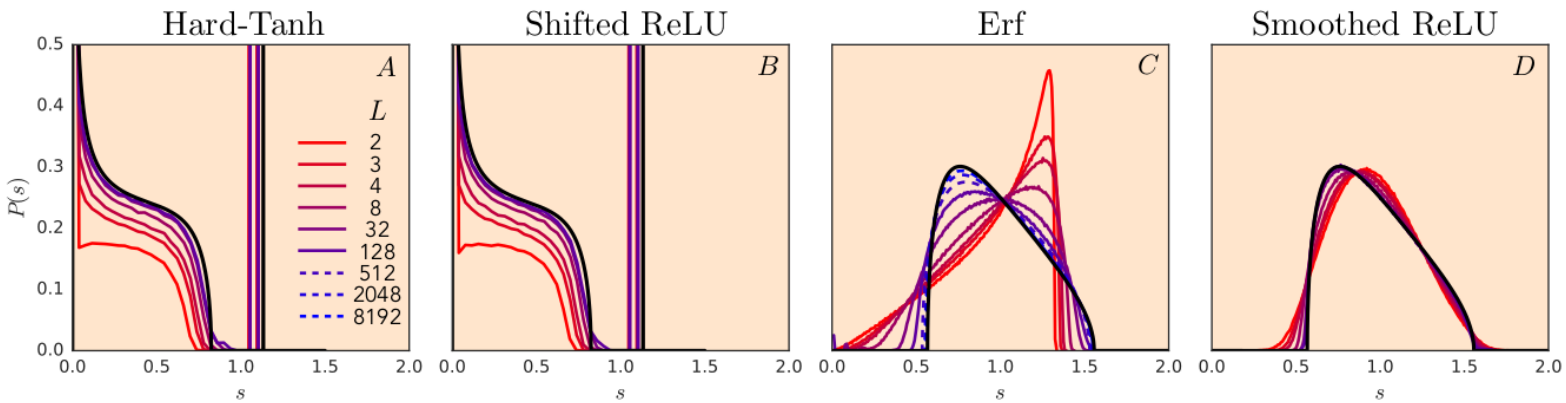


Figure: Limit spectral distribution (black line) with $L \to \infty$ after $N \to \infty$ and empirical results for $L = 2, 3, 4, 8, 32, 128, 512, 2048, 8192$. (Figure from PSG(AISTATS2018))

3. Neural Tangent Kernel

Neural Tangent Kernel

Under continual version of GD, learning dynamics of parameters is given by:

$$\frac{d\theta_t}{dt} = \eta(\nabla_\theta f_{\theta_t})^\top (y - f_{\theta_t})$$

(* The learning rate η is fixed.) Then, learning dynamics of DNN is given by:

$$\frac{df_\theta}{dt} = \eta_t \Theta_t (y - f_{\theta_t})$$

where

$$\Theta_t = \nabla_\theta f_{\theta_t} (\nabla_\theta f_{\theta_t})^\top$$

Neural Tangent Kernel and Dynamics

Informal[Jacot+NeurIPS2018, Lee+NeruIPS2019] Under the wide limit $n \to \infty$, the learning dynamics of DNN are approximated by

$$\frac{df_\theta}{dt} = \eta \Theta (y - f_{\theta_t})$$

where the *neural tangent kernel* is defined as

$$\Theta := \lim_{n_1, \ldots, n_{L-1} \to \infty} \Theta_0.$$

The neural Tangent Kernel is A Surrogate Model of DNN+GD

Based of NTK, we can do Bayesian estimation in the same way as NNGP. Moreover, with NTK, we can simulate the gradient descent at any step t of ensemble networks.

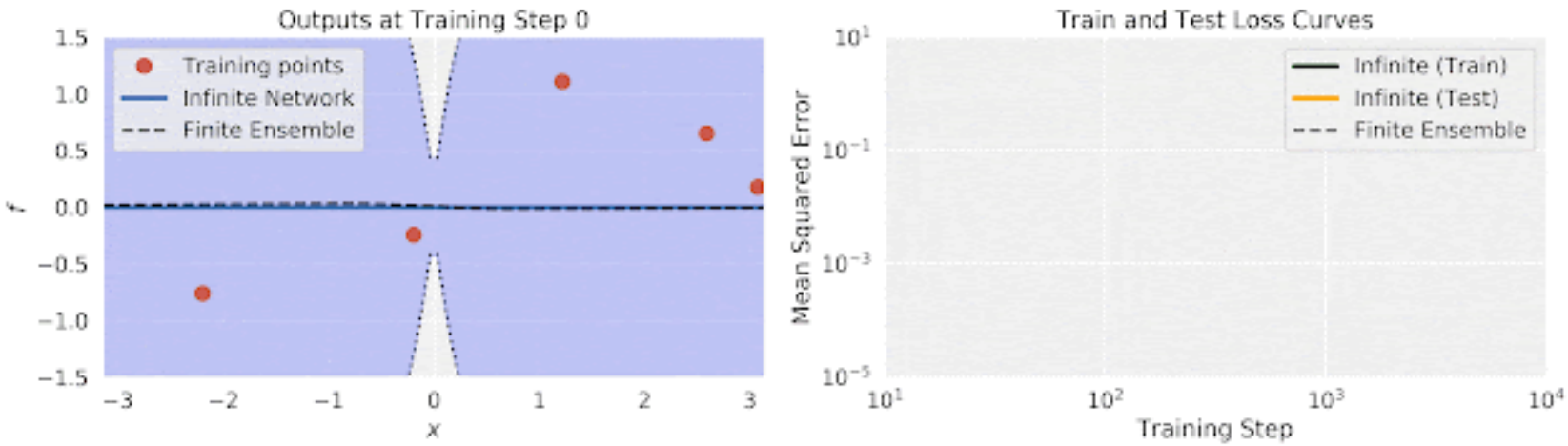


Figure: from Google "Fast and Easy Infinitely Wide Networks with Neural Tangents"

The neural Tangent Kernel is A Surrogate Model of DNN+GD

Based of NTK, we can do Bayesian estimation in the same way as NNGP. Moreover, with NTK, we can simulate the gradient descent at any step t of ensemble networks.

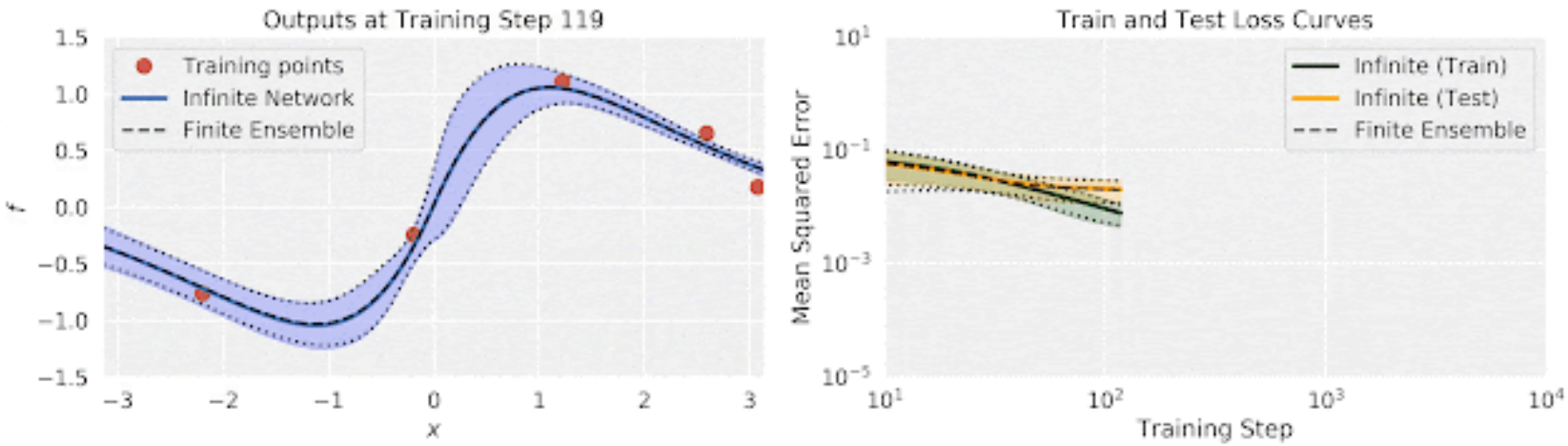


Figure: from Google "Fast and Easy Infinitely Wide Networks with Neural Tangents"

The neural Tangent Kernel is A Surrogate Model of DNN+GD

Based of NTK, we can do Bayesian estimation in the same way as NNGP. Moreover, with NTK, we can simulate the gradient descent at any step t of ensemble networks.

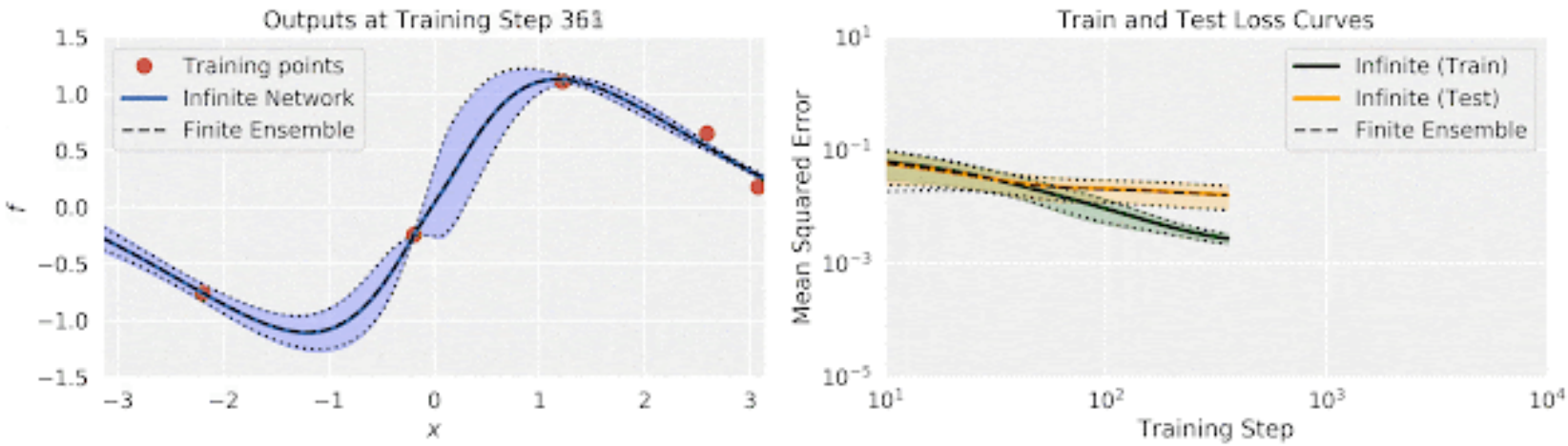


Figure: from Google "Fast and Easy Infinitely Wide Networks with Neural Tangents"

The neural Tangent Kernel is A Surrogate Model of DNN+GD

Based of NTK, we can do Bayesian estimation in the same way as NNGP. Moreover, with NTK, we can simulate the gradient descent at any step t of ensemble networks.

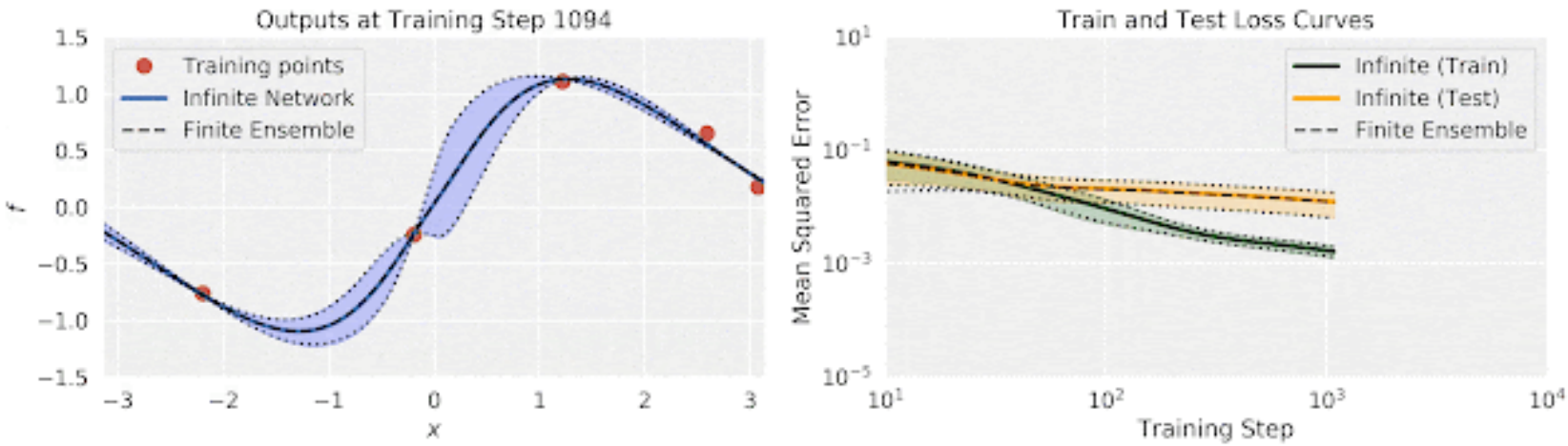


Figure: from Google "Fast and Easy Infinitely Wide Networks with Neural Tangents"

The neural Tangent Kernel is A Surrogate Model of DNN+GD

Based of NTK, we can do Bayesian estimation in the same way as NNGP. Moreover, with NTK, we can simulate the gradient descent at any step t of ensemble networks.

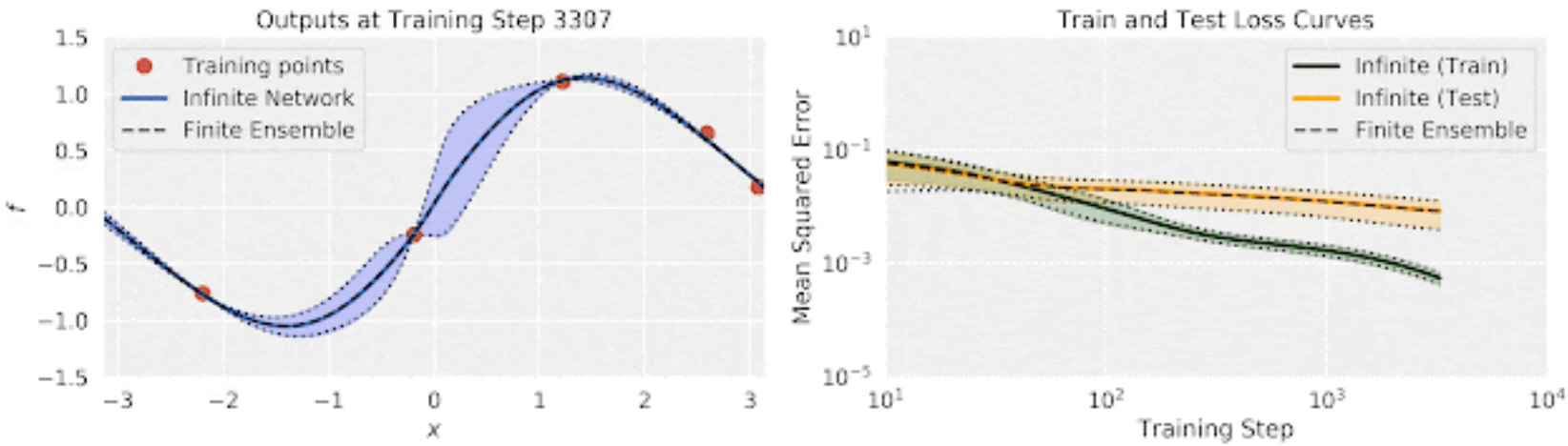


Figure: from Google "Fast and Easy Infinitely Wide Networks with Neural Tangents"

The neural Tangent Kernel is A Surrogate Model of DNN+GD

Based of NTK, we can do Bayesian estimation in the same way as NNGP. Moreover, with NTK, we can simulate the gradient descent at any step t of ensemble networks.

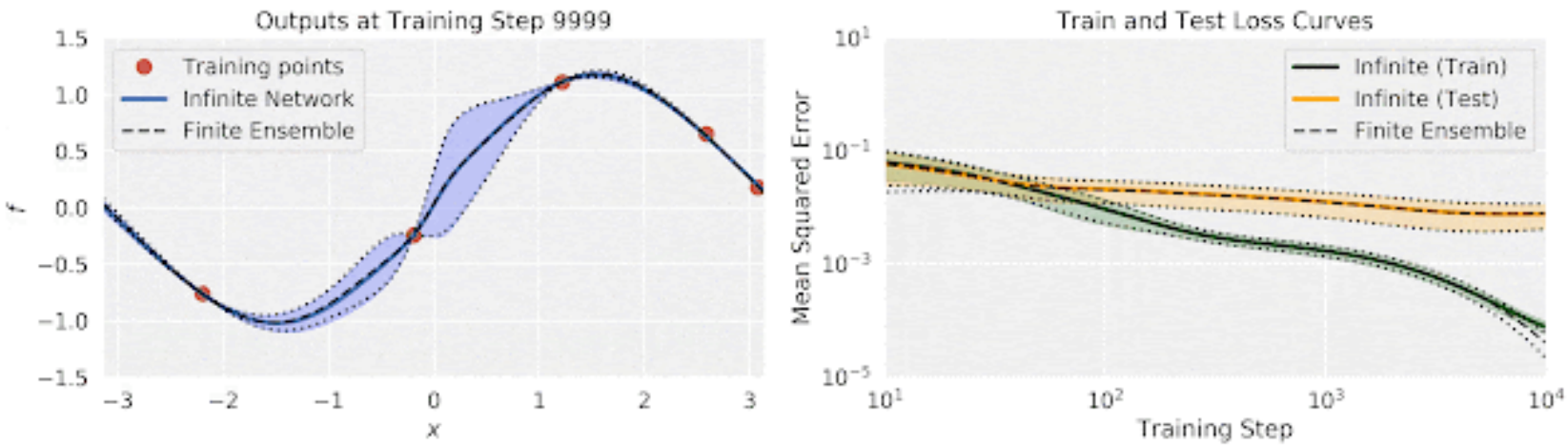


Figure: from Google "Fast and Easy Infinitely Wide Networks with Neural Tangents"

Eigenvalue Spectrum of NTK

"Spectra of the Conjugate Kernel and Neural Tangent Kernel for linear-width neural networks", Z. Fan & Z. Wang (arXiv:2005.11879).
They treat the standard formulation: Gaussian Initialization & Multi-samples & Small output dimension, and they get a recurrence equation of the limit spectral distribution of NTK.

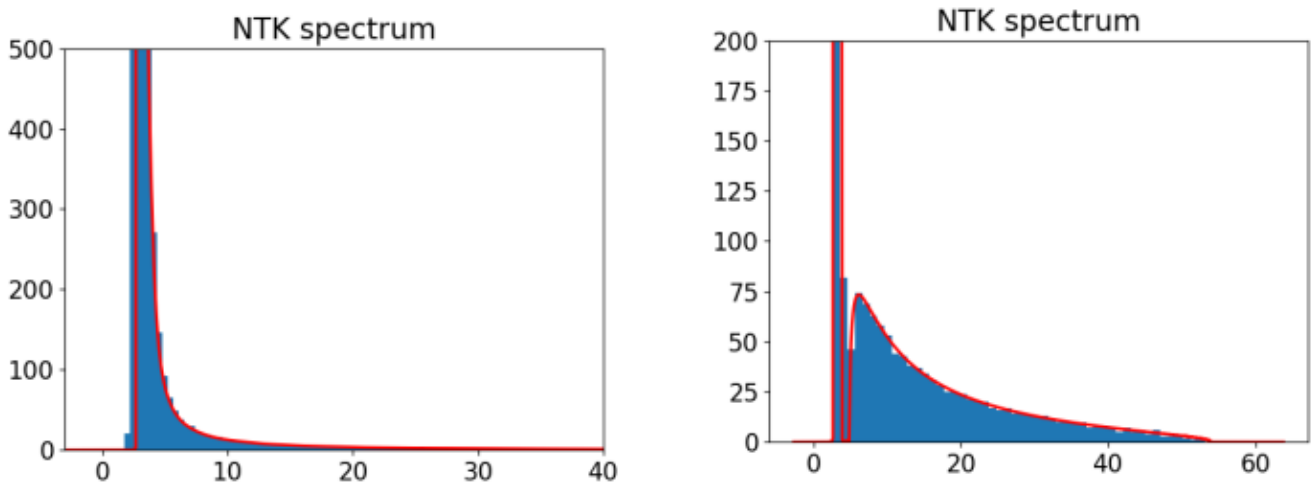


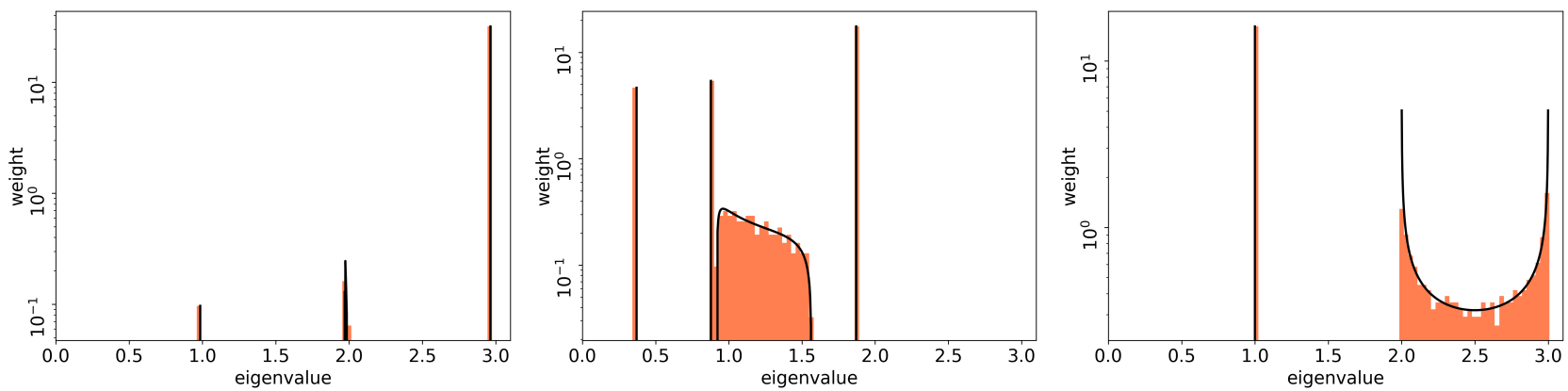
Figure: Spectrum of full NTK (red line: theory, blue line: empirical spectral distribution.)

One-Sample NTK

When the DNN achieves dynamical isometry, the spectrum of the one-sample NTK concentrates around the maximal value, and the maximal value is $O(L)$. [Ryo Karakida & TH (arXiv:2006.07814, AISTATS2021)]

(Sketch)Under an assumption on **Asymptotic Freeness**, we have the following recursive equations:

$$\Theta_{\ell+1} = q_\ell + W_{\ell+1} D_\ell \Theta_\ell D_\ell W_{\ell+1}^\top,$$
$$\mu_{\ell+1} = (q_\ell + \sigma_{\ell+1}^2 \cdot)_* (\nu_\ell \boxtimes \mu_\ell)$$



One-Sample NTK & Learning Rate

The spectrum (eigenvalues) of the NTK has a vital role in tuning the learning dynamics.
e.g. $\eta > 1/\lambda_{\max}(\Theta) \Longrightarrow$ The learning dynamics do not converge.
e.g. The conditional number $c = \lambda_{\min}/\lambda_{\max}$ detemines the converge speed.

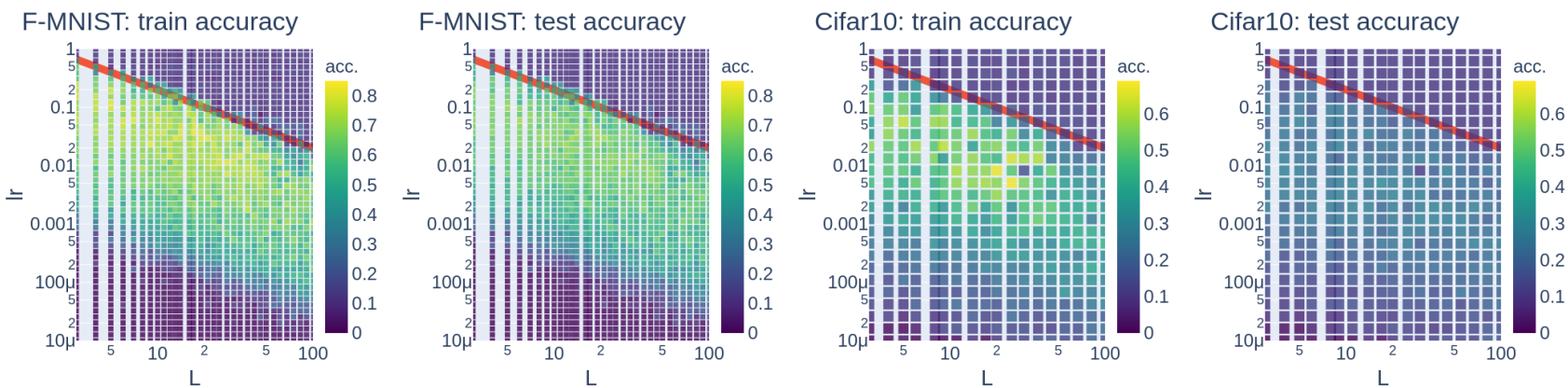


Figure: Redline (the borderline of the exploding gradients). Our theory expects this line!

Out of NTK ball

Original NTK theory requires a small learning rate. G. Yang expanded the scaling of weights in NTK to a more general one: maximal update parametrization (μP)! [G. Yang, Tensor Programs V: arXiv:2203.03466]

4 Asymptotic Freeness

Asymptotic Freeness and Free Probability Theory

Definition (Asymtptotic freeness, a C*-version)
Let $(A_j(n), A_j(n)^*)_{j\in J}$ be a family of $n \times n$ random matrices and adjoints.
The family is said to be *asymptotically free almost surely* if there exist C*-probability spaces $(\mathfrak{A}_j, \tau_j)_{j\in J}$ and elements $(a_j \in \mathfrak{A}_j)_{j\in J}$ so that for any $Q \in \mathbb{C}\langle X_j, X_j^* \mid j \in J\rangle$, the following holds:

$$\lim_{n\to\infty} \mathrm{tr}_n \left[Q(A_j(n), A_j(n)^* \mid j \in J)\right] = (*_{j\in J}\tau_j)\left[Q(a_j, a_j^* \mid j \in J)\right],$$

where $*_{j\in J}\tau_j$ is the free product of the tracial states.

Example

For $N \in \mathbb{N}$, let
- $W(N)$ be Ginibre or Haar orthogonal random matrix,
- $D(N)$ be a constant diagonal matrix with a limit distribution as $N \to \infty$.

Then (W, W^*) and D are a.s. asymptotically free as $N \to \infty$.

Asumptotic Freeness of Jacobians

Let $W_\ell, D_\ell (\ell = 1, 2, \ldots, L)$ be weight matrices in MLP and W_ℓ be scaled Haar orthogonal random matrices. (The Gaussian case is treated by : [B. Hanin and M. Nica.], [L. Pastur.], [G.Yang])

Theorem. [CH22]

Assuming that $D_1, \ldots, D_{L-1}$ have limit joint moments. Then

$$(W_1, W_1^\top), \ldots, (W_L, W_L^\top), (D_1, \ldots, D_{L-1})$$

are asymptotically free as $n \to \infty$ almost surely.

Difficulty: Entries of D_ℓ and W_ℓ are **not** independent.

$D_\ell = \mathrm{diag}(\phi'(h_\ell))$,

$h_\ell = W_\ell x_\ell$.

The heart of Proof: Invariance of MLP.

Consider arbitrary orthogonal random matrices U_ℓ fixing x_ℓ , i.e.,

$$U_\ell x_\ell = x_\ell,$$

with

$$(U_0, \ldots, U_\ell) \perp\!\!\!\perp (W_{\ell+1}, \ldots, W_L),$$

for each $\ell = 0, \ldots, L-1$. Then the joint distribution does not change as follows:

$$\begin{aligned}&(W_1 U_0, W_2 U_1, \ldots, W_L U_{L-1}, D_1, \ldots, D_{L-1}) \\ &\sim (W_1, W_2, \ldots, W_L, D_1, \ldots, D_{L-1}).\end{aligned}$$

The matrix U has only a one-dimensional constraint, and the remaining N-1 dimensions can be taken independently. The large N limit can neglect the error of ignoring one dimension.

5. Summary

Summary

1. Random matrices appear as initialization of weights in DNN.

Summary

1. Random matrices appear as initialization of weights in DNN.
2. Random matrices also appear in the analysis of learned weights (NTK).

Summary

1. Random matrices appear as initialization of weights in DNN.
2. Random matrices also appear in the analysis of learned weights (NTK).
3. Infinite dimensional limit and free probability are helpful in the analysis of NTK, Dynamical Isometry since noncommutative polynomials of random matrices appear in the analysis.