Google

Understanding and Improving Deep Learning with Random Matrix Theory

Jeffrey Pennington

Google Brain, NYC

November 8, 2017

Stats 385, Stanford

Outline

1. Motivation
2. Essentials of random matrix theory
3. Geometry of neural network loss landscapes
4. Resurrecting the sigmoid in deep learning
5. Nonlinear random matrix theory
6. Conclusions

Motivation: Why Random Matrices?

Why random matrices?

- The initial weight configuration is random
  - Training may induce only low-rank perturbations around the random configuration

- An exact theory of deep learning is likely to be intractable or uninformative
  - Large complex systems are often well-modeled with random variables
    - E.g. statistical physics and thermodynamics

- Many important quantities are specific to matrix structure
  - E.g. eigenvalues and eigenvectors

Which matrices do we care about?

- Activations

$$Y^l_{i\mu} = f\left(\sum_j W^l_{ij} Y^{l-1}_{j\mu}\right)$$

- Hessians

$$H = \frac{\partial^2 \mathcal{L}}{\partial W \partial W} = \begin{array}{cc} & \begin{array}{cc} L-1 & 1 \end{array} \\ \begin{array}{c} L-1 \\ 1 \end{array} & \left(\begin{array}{c|c} * & * \\ \hline * & Y^L (Y^L)^T \end{array}\right) \end{array}$$

- Jacobians

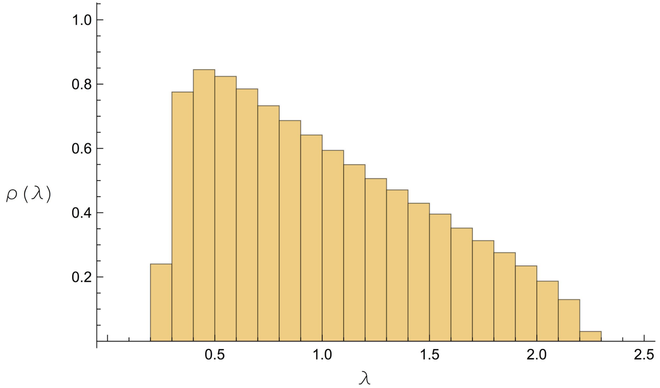
$$J = \frac{\partial y^L}{\partial z^1} = W_L D_L W_{L-1} D_{L-1} \cdots W_1 D_1$$

Essentials of random matrix theory

Spectral density

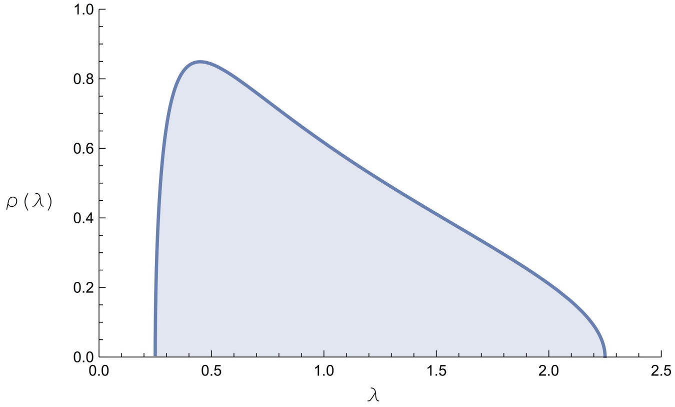
For any matrix $M_n \in \mathbb{R}^{n\times n}$, the **empirical spectral density** is:

$$\rho_n(\lambda) = \frac{1}{n}\sum_{j=1}^{n}\delta(\lambda - \lambda_j(M_n))$$



For a sequence of matrices with increasing size, $(M_n)_{n\in\mathbb{N}}$, the **limiting spectral density** is:

$$\rho(\lambda) \equiv \lim_{n\to\infty}\rho_n(\lambda)$$



Stieltjes transform

For $z \in \mathbb{C} \setminus \mathbb{R}$ the **Stieltjes transform** is defined as:

$$G(z) = \int_{\mathbb{R}} \frac{\rho(t)}{z-t} dt\,,$$

Using the identities,

$$\rho(\lambda) = \int \rho(t)\delta(\lambda - t)dt \qquad \delta(x) = \lim_{\epsilon \to 0^+} \frac{1}{\pi} \frac{\epsilon}{\epsilon^2 + x^2} = \lim_{\epsilon \to 0^+} \operatorname{Im}\left[-\frac{1}{\pi}\frac{1}{x + i\epsilon}\right]$$

The spectral density can be recovered from G using the inversion formula,

$$\rho(\lambda) = -\frac{1}{\pi} \lim_{\epsilon \to 0^+} \operatorname{Im} G(\lambda + i\epsilon)$$

R-transform and S-transform

The Stieltjes transform can be used to define two useful auxiliary objects:

the **R-transform**, defined by the functional equation,

$$\frac{1}{G(z)} + \mathcal{R}(G(z)) = z$$

and the **S-transform**, defined by a similar functional equation,

$$\mathcal{S}(zG(z) - 1) = \frac{G(z)}{zG(z) - 1}$$

Free addition and free multiplication

If **A** and **B** are **freely independent**, then the spectrum of the sum **A+B** computed using the R-transform:

$$\begin{tikzcd}
\rho_A(\lambda) \arrow[r] & G_A(z) \arrow[r] & R_A \arrow[dr] & & & \\
 & & & R_A + R_B = R_{A+B} \arrow[r] & G_{A+B}(z) \arrow[r] & \rho_{A+B}(\lambda) \\
\rho_B(\lambda) \arrow[r] & G_B(z) \arrow[r] & R_B \arrow[ur] & & &
\end{tikzcd}$$

And the spectrum of the product **AB** can be computed using the S-transform:

$$\begin{tikzcd}
\rho_A(\lambda) \arrow[r] & G_A(z) \arrow[r] & S_A \arrow[dr] & & & \\
 & & & S_A S_B = S_{AB} \arrow[r] & G_{AB}(z) \arrow[r] & \rho_{AB}(\lambda) \\
\rho_B(\lambda) \arrow[r] & G_B(z) \arrow[r] & S_B \arrow[ur] & & &
\end{tikzcd}$$

Free Independence

Classical independence

X, Y are independent if one has

$$\mathbf{E} f(X) g(Y) = 0$$

whenever f and g are such that

$$\mathbf{E} f(X) = 0$$
$$\mathbf{E} g(Y) = 0$$

Free independence

X, Y are freely independent if

$$\mathbf{E} f_1(X) g_1(Y) \dots f_k(X) g_k(Y) = 0$$

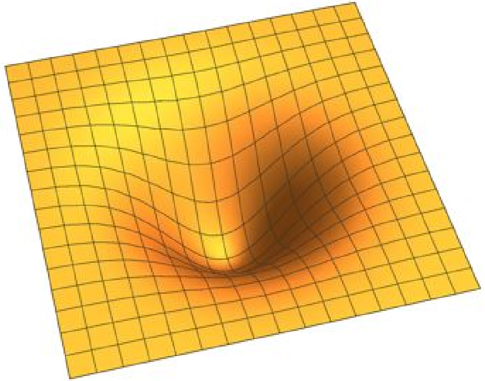
whenever f_i and g_i are such that

$$\mathbf{E} f_i(X) = 0$$
$$\mathbf{E} g_i(Y) = 0$$

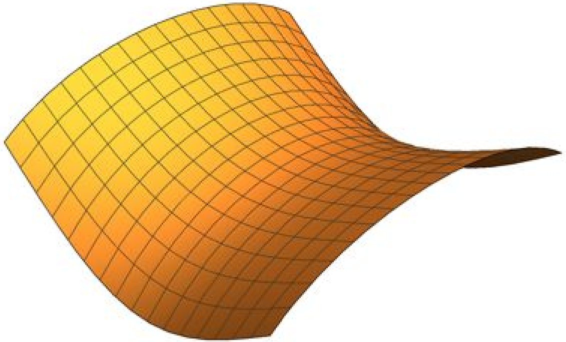
Geometry of neural network loss surfaces

with Yasaman Bahri

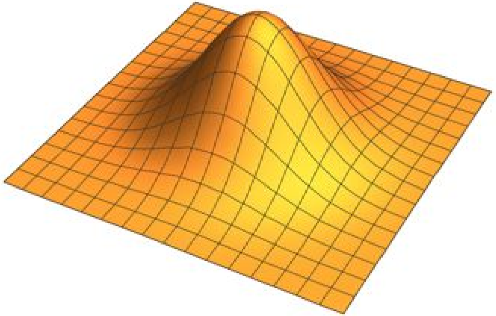
Single critical point



$\{1, 1\}$

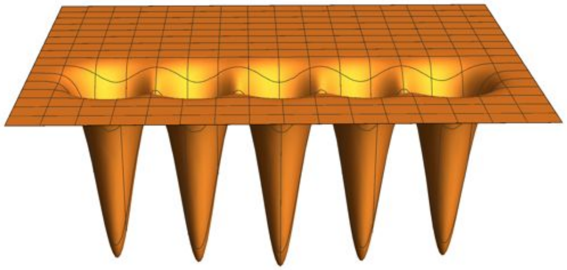


$\{1, -1\}$

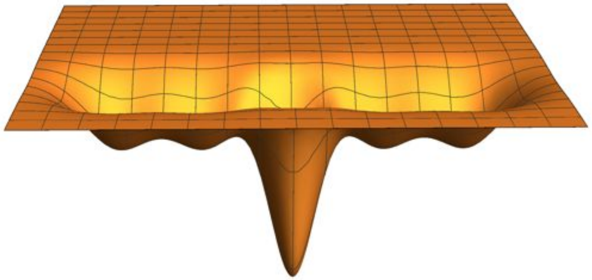


$\{-1, -1\}$

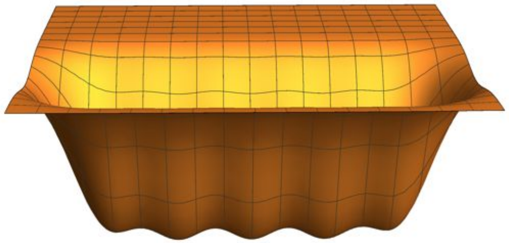
Multiple critical points



A) All minima are roughly equivalent, but index 1 critical points have higher loss.



B) Global minimum much lower than local minima and index 1 critical points.



C) All minima and index 1 critical points are roughly equivalent.

The geometry of high-dimensional loss landscapes

Very little is known for general network architectures!

- Are all minima global?
    - True for deep linear networks [Kawaguchi 2016]
    - Almost true for non-linear networks with pyramidal structure [Nguyen & Hein 2017]
- What is the distribution of critical points (points where grad vanishes)?
    - Connection to certain spin glass systems [Choromanska, et al. 2014]
    - Field theory of Gaussian random functions [Bray & Dean 2007]
        - Expression for index as a function of error:

“Energy” or loss value at a critical point

Index = # of negative eigenvalues of the Hessian matrix

$$\alpha \approx \frac{4\sqrt{2}}{3\pi}\left|\frac{\epsilon-\epsilon_c}{\epsilon_c}\right|^{\frac{3}{2}}$$

Energy of minimizer

Notation

Consider a neural net parameterized by θ with d-dimensional output $\hat{y}$.

$$\mathcal{L} = \frac{1}{n}\sum_{\mu=1}^{n} \ell(\hat{y}^{\mu}, y^{\mu}) \equiv \epsilon$$

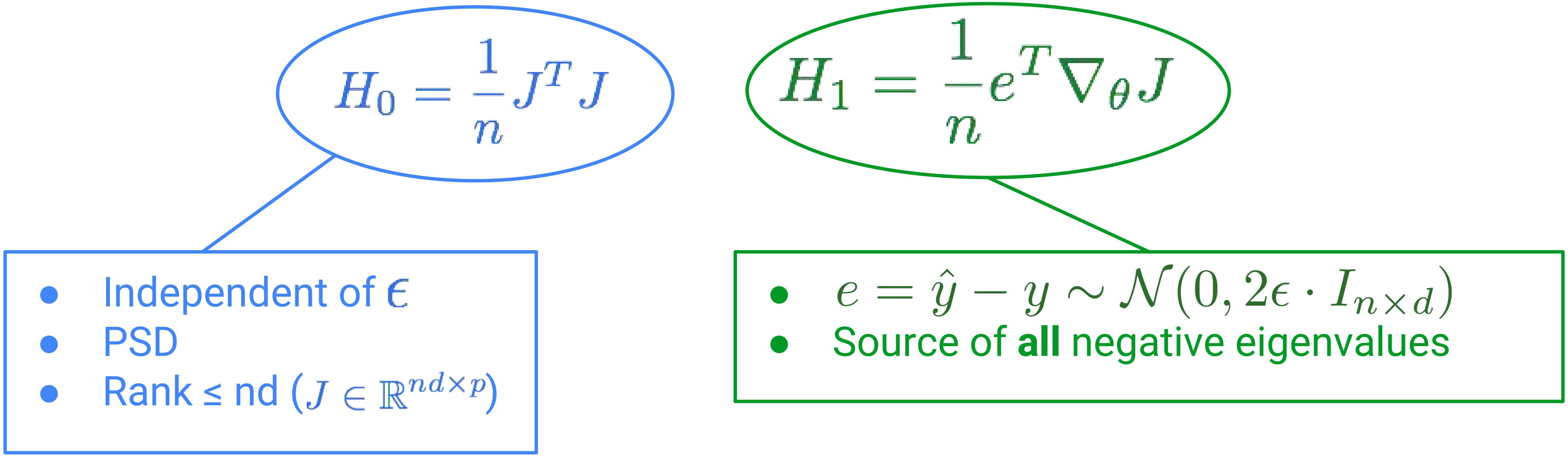
Jacobian: $J_a^{i,\mu} = \dfrac{\partial \hat{y}_i^{\mu}}{\partial \theta_a}$

Grad: $\dfrac{\partial \mathcal{L}}{\partial \theta_a} = \dfrac{1}{n}\displaystyle\sum_{\mu,i=1}^{n,d} \dfrac{\partial \ell^{\mu}}{\partial \hat{y}_i^{\mu}} J_a^{i,\mu}$

Hessian: $$H = \frac{\partial^2 \mathcal{L}}{\partial \theta_a \partial \theta_b} = \frac{1}{n}\sum_{\mu,i=1}^{n,d}\left[\sum_{j=1}^{d} \underbrace{\frac{\partial^2 \ell^{\mu}}{\partial \hat{y}_i^{\mu} \hat{y}_j^{\mu}} J_a^{i,\mu} J_b^{j,\mu}}_{H_0} + \underbrace{\frac{\partial \ell^{\mu}}{\partial \hat{y}_i^{\mu}} \frac{\partial J_a^{i,\mu}}{\partial \theta_b}}_{H_1}\right]$$

Form of the Hessian

$$H = \frac{1}{n}\left[J^T J + (\hat{y} - y)^T \nabla_\theta J\right] = H_0 + H_1$$



$$H_0 = \frac{1}{n} J^T J$$

- Independent of ϵ
- PSD
- Rank ≤ nd ($J \in \mathbb{R}^{nd \times p}$)

$$H_1 = \frac{1}{n} e^T \nabla_\theta J$$

- $e = \hat{y} - y \sim \mathcal{N}(0, 2\epsilon \cdot I_{n \times d})$
- Source of **all** negative eigenvalues

Random matrix approximations

1. Regard H as a **structured** random matrix
2. Approximate H_0 as a **Wishart** matrix, i.e. assume elements of J are iid
3. Depending on the architecture, approximate H_1 as:
   - A **Wigner** matrix, for generic architecture
   - A **product Wishart** matrix, for ReLU activation functions
4. Assume H_0 and H_1 are **freely independent**, i.e. they are independent and their eigenspaces are not aligned in any special way
   - Compute spectrum using **R-transform**

Predictions for the number of negative eigenvalues

The individual R-transforms are known or are possible to derive:

Wishart + Wigner (generic):

$$\mathcal{R}_H = \frac{\sigma}{1-\sigma z\phi} + 2\epsilon z$$

Wishart + product-Wishart (ReLU):

$$\mathcal{R}_H = \frac{\sigma}{1-\sigma z\phi} + \frac{\epsilon\phi z}{2-\epsilon\phi^2 z^2}$$

σ = scale of J (set σ = 1 wlog)

ϵ = loss value (energy)

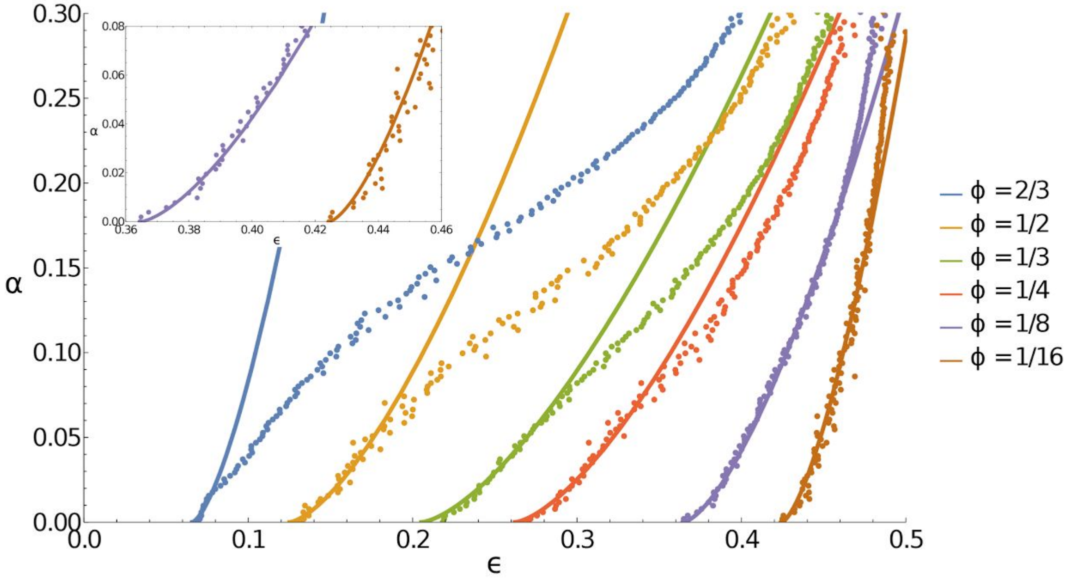
ϕ = #parameters / #samples (ratio of dimensions of J)

Recovering the spectral densities, we find the same behavior in both cases:

$$\alpha(\epsilon,\phi) \equiv \int_{-\infty}^{0} \rho(\lambda;\epsilon,\phi) \quad \approx \tilde{\alpha}_0(\phi)\left|\frac{\epsilon-\epsilon_c}{\epsilon_c}\right|^{3/2}$$

Distribution of critical points

- Train many networks using a heuristic method attracted to saddle points
- For each obtained critical point, note the index and the energy
- For each unique value of the index, report the mean energy
- Fits well to theory!



(a) Index of critical points versus energy

Resurrecting the sigmoid in deep learning

with Sam Schoenholz
and Surya Ganguli

Deep learning vs very deep learning

- Deep learning: <25 layers
  - Challenging but possible to train.
- Very deep learning: O(100) layers
  - Nearly impossible to train without hacks/tricks.
    - ResNets/batch norm
    - LSTMs
- Are these techniques/architectures useful beyond their ability to improve training?
- What is the intrinsic difficulty in training very deep models?

Necessary conditions for trainability

[Poole (2016); Schoenholz (2016)]

1. **The forward-propagated signals (activations) should not explode or vanish**

$$z^{l+1} = W^{l+1} f(z^l) + b^l$$

$$b_i \sim \mathcal{N}(0, \sigma_b^2)$$
$$W_{ij} \sim \mathcal{N}(0, \sigma_w^2/n)$$

Variance of activations should approach a fixed-point:

$$\mathbb{E}[(z_1^{l+1})^2] = \mathbb{E}[(z_1^l)^2] \equiv q^* \qquad \Rightarrow \qquad z^l \sim \mathcal{N}(0, q^*)$$

Possible under appropriate conditions:

tanh: $\sigma_w > 1 \text{ or } \sigma_b > 0$

ReLU: $\sigma_w \leq \sqrt{2}$

Necessary conditions for trainability

[Poole (2016); Schoenholz (2016)]

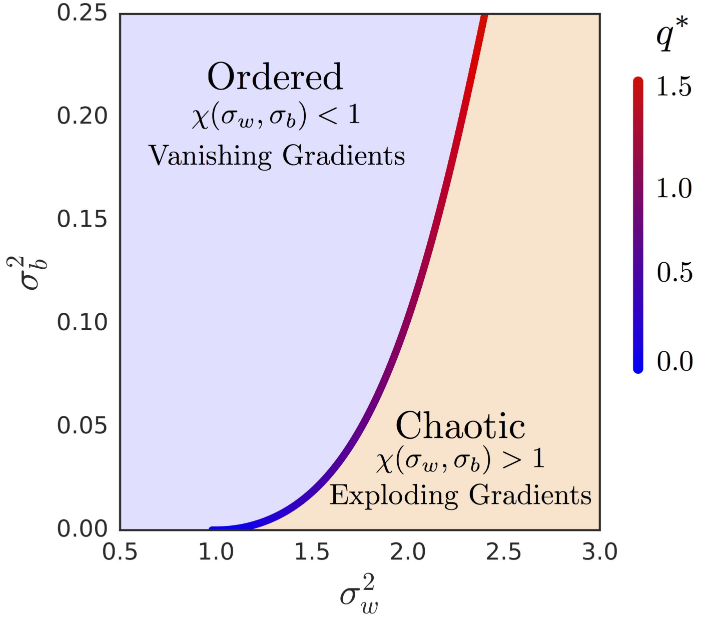
2. **The backward-propagated signals (error signal) should not explode/vanish**

$$\delta_j^{l-1} = \sum_i \delta_i^l W_{ij}^l f'(z_j^{l-1}) \qquad \delta^L = \epsilon$$

$$\mathbb{E}\left[(\delta_1^{l-1})^2\right] = \mathbb{E}\left[(\delta_1^l)^2\right] \underbrace{\sigma_w^2 \mathbb{E}\left[f'(z_1^{l-1})^2\right]}_{\chi(\sigma_w, \sigma_b)}$$

$$\mathbb{E}\left[(\delta_1^1)^2\right] = \mathbb{E}\left[(\delta_1^L)^2\right] \chi(\sigma_w, \sigma_b)^L$$

$$\boxed{\chi(\sigma_w, \sigma_b) = 1}$$



Beyond gradient norms

Along the critical line, the gradient norms are well-behaved on average:

$$\delta^1 = \epsilon^T J$$

$$J = \frac{\partial y^L}{\partial z^1} = W_L D_L W_{L-1} D_{L-1} \cdots W_1 D_1 \qquad D_l = \mathrm{diag}(f'(z^l))$$

$$\mathbb{E}_\epsilon[||\delta^1||^2] = \mathbb{E}_\epsilon[\epsilon^T J J^T \epsilon] = \mathrm{tr}[JJ^T] = \sum_{i=1}^n \sigma_i^2$$

$$\chi(\sigma_w, \sigma_b) = 1 \iff \frac{1}{n}\sum_{i=1}^n \sigma_i^2 = 1$$

Singular values of input-output Jacobian

The singular values of

$$J = \frac{\partial y^L}{\partial z^1} = W_L D_L W_{L-1} D_{L-1} \cdots W_1 D_1$$

tell us how much the error signals get stretched and skewed as they propagate back through the network.

Even if the mean squared singular value is one, there can be directions with large singular values, which can derail learning.

Computing the distribution of singular values

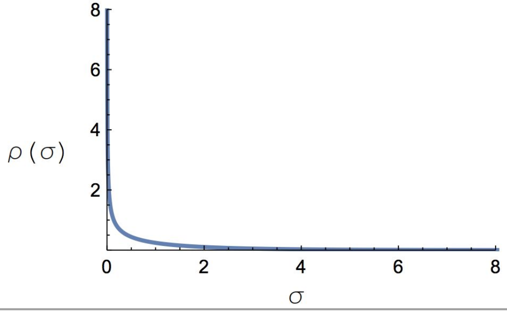
We have seen how to initialize the weight and bias variance so as to prevent gradient signals from exploding or vanishing.

Can we also manipulate the singular value distribution so that it is better conditioned?

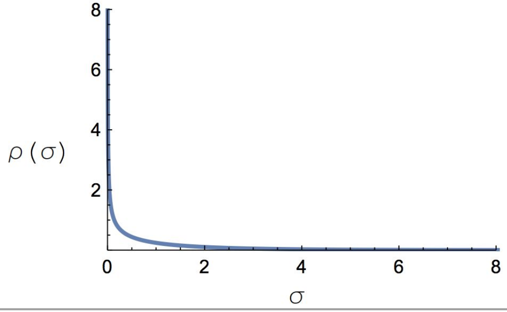
We would need to know how to compute the singular values of a product of random matrices --> **S-transforms**!

Jacobian spectra for large depth

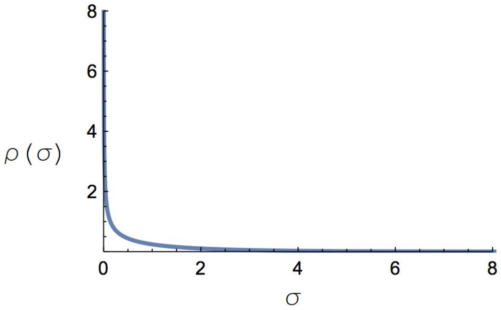
Gaussian W, any f



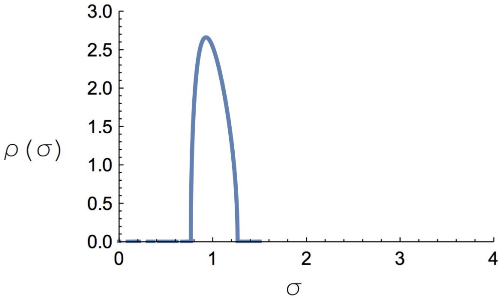
Orthogonal W, ReLU



Orthogonal W, tanh, $\sigma_w \gg 1$

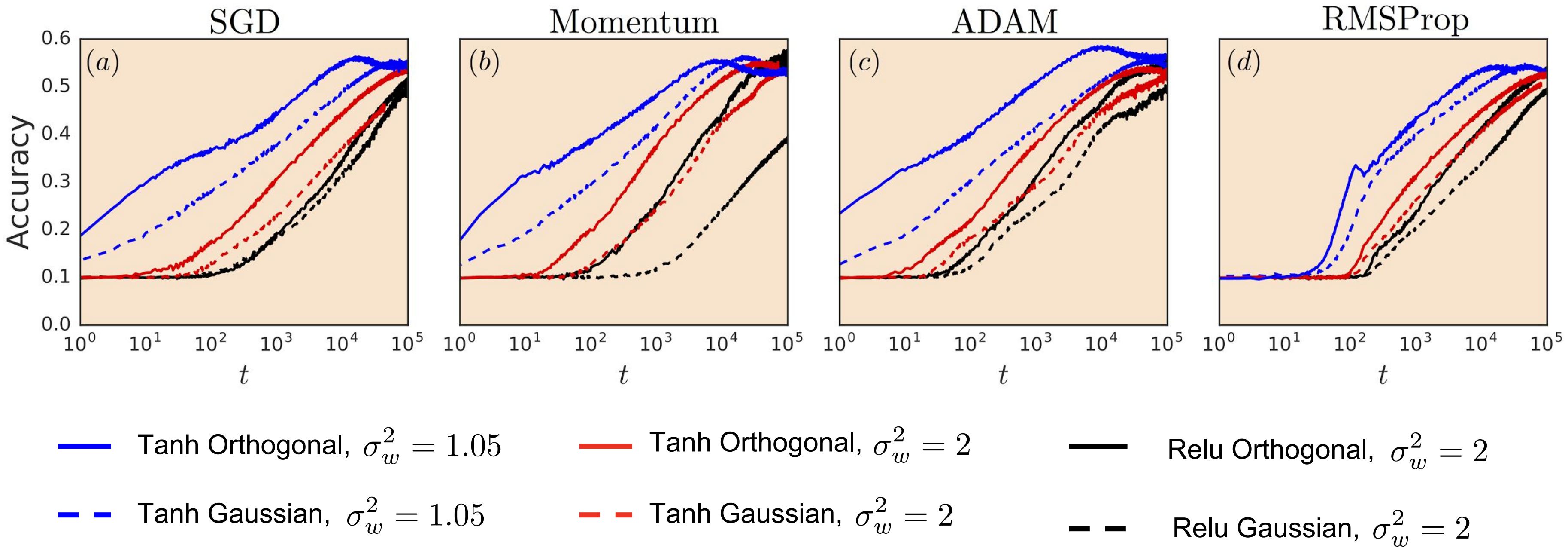


Orthogonal W, tanh, $\sigma_w \sim 1+1/L$



Implications for training speed

Test accuracy on CIFAR-10



Nonlinear random matrix theory

with Pratik Worah

Problem setup

One of the basic building blocks of many of the salient matrices for deep learning is

$$YY^T = f(WX)f(WX)^T$$

From the mathematical perspective, this is also one of the simplest random matrices with non-linear dependencies for which an explicit representation of the eigenvalues is unknown.

Spectral density and moments

For any matrix $M_n \in \mathbb{R}^{n \times n}$, recall that the empirical density of eigenvalues is:

$$\rho_n(\lambda) = \frac{1}{n} \sum_{j=1}^{n} \delta(\lambda - \lambda_j(M_n))$$

The moments of this distribution are related to the traces of the matrix:

$$m_k^{(n)} = \mathbb{E}[\frac{1}{n}\mathrm{tr}(M_n^k)] = \int \rho_n(\lambda)\lambda^k \, d\lambda$$

For a sequence of matrices with increasing size, $(M_n)_{n\in\mathbb{N}}$, define:

$$\rho(\lambda) \equiv \lim_{n\to\infty} \rho_n(\lambda) \qquad\qquad m_k \equiv \lim_{n\to\infty} m_k^{(n)}$$

Stieltjes transform

Can we construct the spectral density from the moments?

$$G(z) = \sum_{k=0}^{\infty} \frac{m_k}{z^{k+1}} = \sum_{k=0}^{\infty} \frac{1}{z^{k+1}} \mathbb{E}[\frac{1}{n}\mathrm{tr}(M^k)]$$

$$= \int \frac{\rho(\lambda)}{z-\lambda} d\lambda = -\mathbb{E}[\frac{1}{n}\mathrm{tr}(M - zI)^{-1}]$$

The spectral density can be recovered through the inversion formula

$$\rho(\lambda) = -\frac{1}{\pi} \lim_{\epsilon \to 0+} \operatorname{Im} G(\lambda + i\epsilon)$$

Moment method

$$\mathbb{E}[\mathrm{tr}(M^k)] \to G(z) \to \rho(\lambda)$$

$$\mathbb{E}[\mathrm{tr}(M^k)] = \sum_{i_1,\ldots,i_k=1}^{n} \mathbb{E}[M_{i_1 i_2} M_{i_2 i_3} \cdots M_{i_{k-1} i_k} M_{i_k i_1}]$$

$$M_{ij} = \sum_{k=1}^{n} Y_{ik} Y_{jk} = \sum_{k=1}^{n} f\Big(\sum_{l=1}^{n} W_{il} X_{lk}\Big) f\Big(\sum_{l=1}^{n} W_{jl} X_{lk}\Big)$$

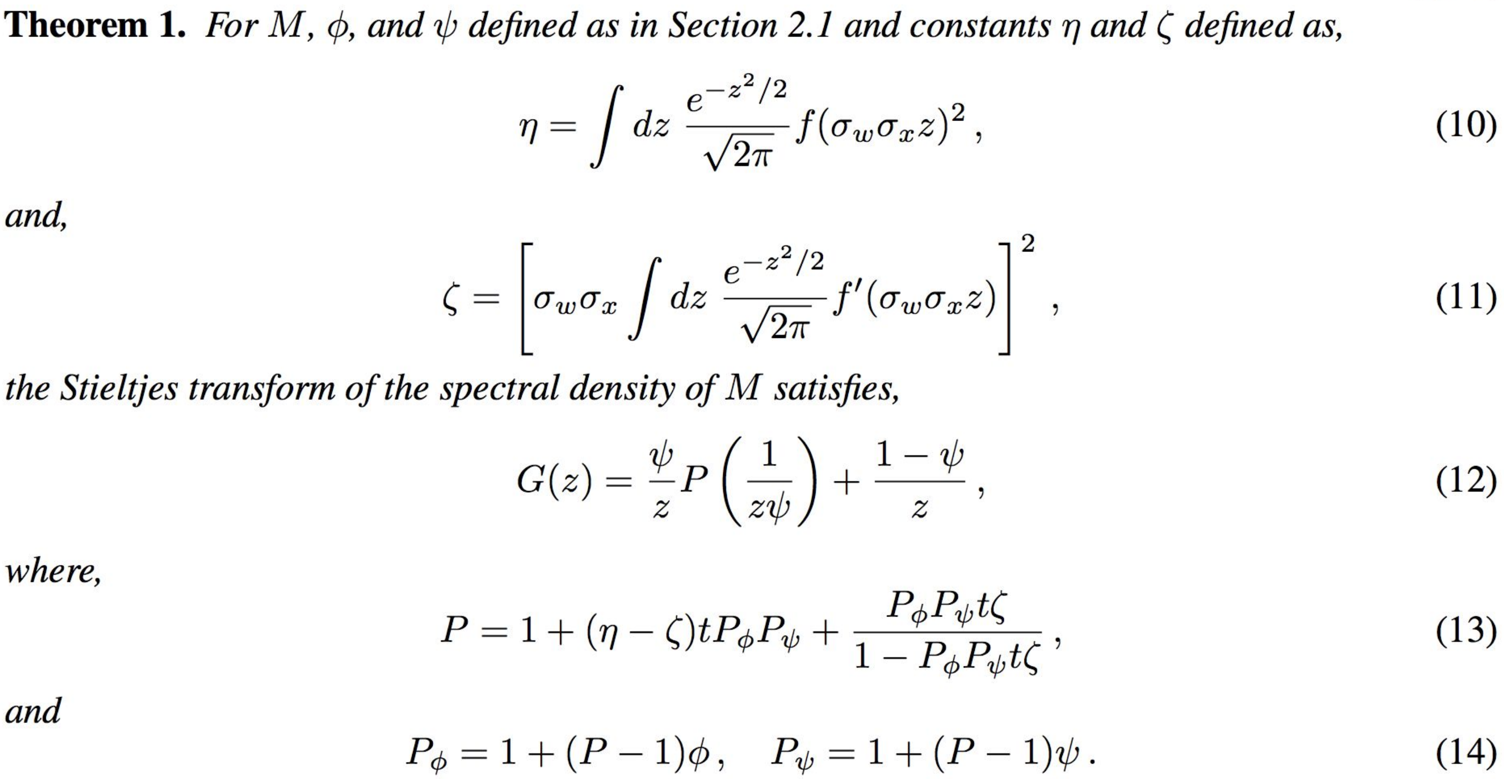
$$\mathbb{E} = \int \prod_{i,j=1}^{n} dW_{ij} \frac{e^{-W_{ij}^2/\sigma_w^2}}{\sqrt{2\pi\sigma_w^2}} \int \prod_{i,j=1}^{n} dX_{ij} \frac{e^{-X_{ij}^2/\sigma_x^2}}{\sqrt{2\pi\sigma_x^2}}$$

Result for Stieltjes transform

$$\phi \equiv \frac{n_0}{m}, \quad \psi \equiv \frac{n_0}{n_1}$$

$$X \in \mathbb{R}^{n_0 \times m}$$

$$W \in \mathbb{R}^{n_1 \times n_0}$$



Theorem 1. *For M, ϕ, and ψ defined as in Section 2.1 and constants η and ζ defined as,*

$$\eta = \int dz \, \frac{e^{-z^2/2}}{\sqrt{2\pi}} f(\sigma_w \sigma_x z)^2 \,, \tag{10}$$

and,

$$\zeta = \left[\sigma_w \sigma_x \int dz \, \frac{e^{-z^2/2}}{\sqrt{2\pi}} f'(\sigma_w \sigma_x z) \right]^2 \,, \tag{11}$$

the Stieltjes transform of the spectral density of M satisfies,

$$G(z) = \frac{\psi}{z} P\left(\frac{1}{z\psi}\right) + \frac{1-\psi}{z} \,, \tag{12}$$

where,

$$P = 1 + (\eta - \zeta) t P_\phi P_\psi + \frac{P_\phi P_\psi t \zeta}{1 - P_\phi P_\psi t \zeta} \,, \tag{13}$$

and

$$P_\phi = 1 + (P - 1)\phi \,, \quad P_\psi = 1 + (P - 1)\psi \,. \tag{14}$$

Asymptotic performance of random features

Targets (random Gaussian)

Random feature matrix

Random (Gaussian) weights

$$L(W_2) = \frac{1}{2n_2m}\|\mathcal{Y} - W_2^TY\|_F^2 + \gamma\|W_2\|_F^2, \quad Y = f(WX)$$

Input (random Gaussian)

$$W_2^* = \frac{1}{m}YQ\mathcal{Y}^T, \quad Q = \left(\frac{1}{m}Y^TY + \gamma I_m\right)^{-1}$$

$$G(z) = -\mathbb{E}[\frac{1}{n}\mathrm{tr}(YY^T - zI)^{-1}] \ \Rightarrow L(W_2^*) = F[G(-\gamma)]$$

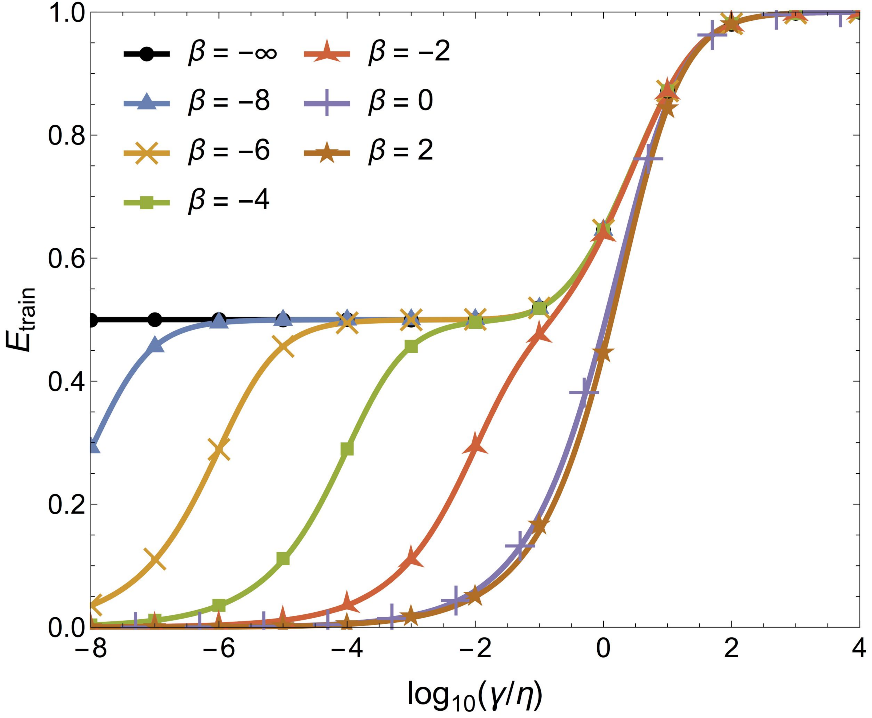
Experiments

$$\eta = \mathbb{E}[f^2]$$

$$\zeta = \mathbb{E}[f']^2$$

$$\beta = \log_{10}(\eta/\zeta - 1)$$

- Excellent agreement between theory and simulation
- Depends on a single scalar statistic of the non-linearity
- For fixed regularization, better performance for nonlinearities which are close to even functions



Google

Conclusions

Conclusions

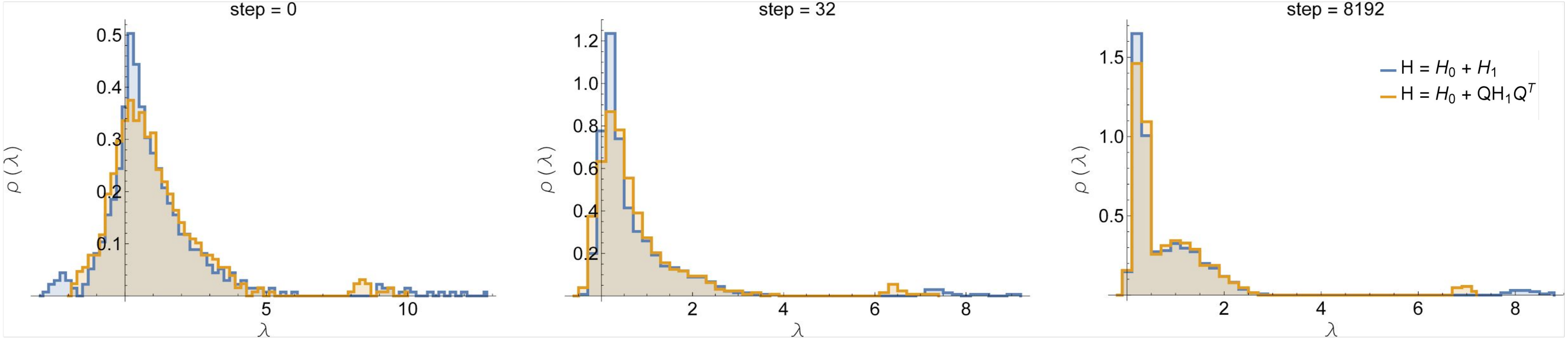
Random matrix theory provides powerful tools for studying deep learning!

1. Demonstrated how to examine the geometry of the loss landscape of neural networks
2. Showed how to equilibrate the distribution of singular values of the input-output Jacobian for faster training
3. Developed techniques for studying random matrices with nonlinear dependencies, showed how to leverage the results to achieve predictive power

Extra slides

Free independence throughout training

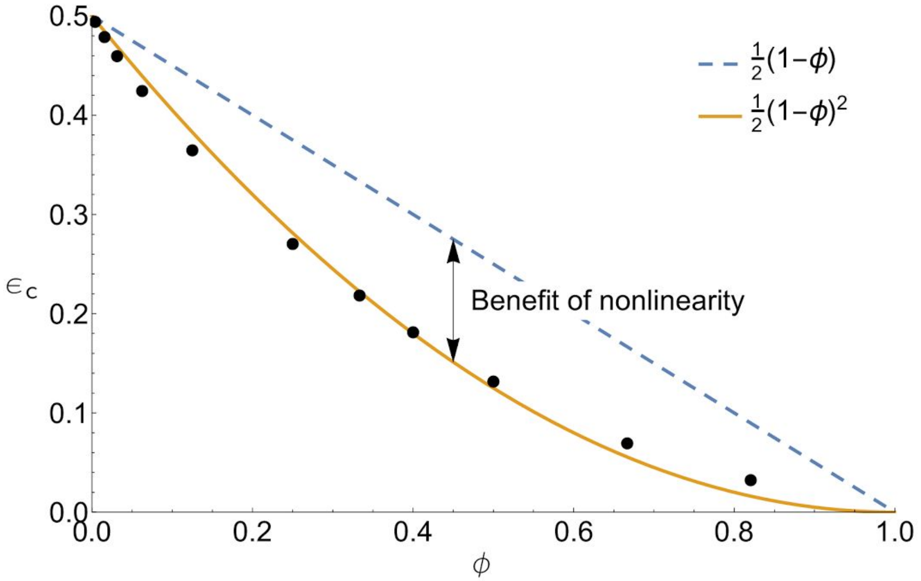
- Track the eigenvalues of $H = H_0 + H_1$ over the course of learning
- Test free independence by comparing to $H = H_0 + QH_1Q^T$ for random orthogonal Q [Chen et al 2012]:



- Good agreement, especially near end of training

Energy of minimizers

- Empirically extract the loss value at minimizers for problems of various sizes
- Simple relationship emerges between loss value and ϕ, the ratio of parameters to samples



(b) Energy of minimizers versus parameters/data points