Random Matrix Theory for Machine Learning

Part 1: Motivating Questions and Building Blocks

Fabian Pedregosa[1], Courtney Paquette[1,2], Tom Trogdon[3], Jeffrey Pennington[1]

[1] Google Research, [2] McGill University, [3] University of Washington

https://random-matrix-learning.github.io

About this tutorial

Objective:

- Applications of *Random Matrix Theory* (RMT) in Machine Learning.
- Proof techniques in RMT

About this tutorial

Objective:

- Applications of *Random Matrix Theory* (RMT) in Machine Learning.
- Proof techniques in RMT

Structure:

1. Motivating Questions and Building Blocks, *Fabian Pedregosa*
2. Introduction to Random Matrix Theory, *Courtney Paquette*
3. Analysis of Numerical Algorithms, *Tom Trogdon*
4. The Mystery of Generalization: Why Does Deep Learning Work?, *Jeffrey Pennington*

https://random-matrix-learning.github.io

What is a Random Matrix?

A *random matrix* is a matrix whose entries are random variables, not necessarily independent.

What is a Random Matrix?

A *random matrix* is a matrix whose entries are random variables, not necessarily independent.

Example

Realization of a random matrix:

$$Z = \begin{bmatrix} 1.066 & 0.908 & 1.026 & -0.294 & 0.879 \\ 0.908 & -1.794 & 0.596 & -1.014 & -0.103 \\ 1.026 & 0.596 & -0.246 & 0.968 & 0.750 \\ -0.294 & -1.014 & 0.968 & 0.184 & 0.812 \\ 0.879 & -0.103 & 0.750 & 0.812 & 0.210 \end{bmatrix}$$

What is a Random Matrix?

A *random matrix* is a matrix whose entries are random variables, not necessarily independent.

Example

Realization of a random matrix:

$$Z = \begin{bmatrix} 1.066 & 0.908 & 1.026 & -0.294 & 0.879 \\ 0.908 & -1.794 & 0.596 & -1.014 & -0.103 \\ 1.026 & 0.596 & -0.246 & 0.968 & 0.750 \\ -0.294 & -1.014 & 0.968 & 0.184 & 0.812 \\ 0.879 & -0.103 & 0.750 & 0.812 & 0.210 \end{bmatrix}$$

Goal of Random Matrix Theory is to understand their

- eigenvalues
- eigenvectors
- norms
- singular values
- singular vectors
- ...

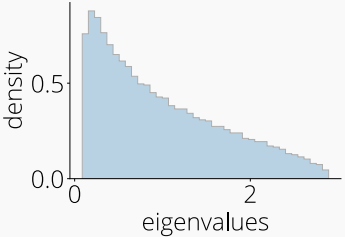
Where do Random Matrices Come From?

1928: Eigenvalues of Normal Covariance Matrices

THE GENERALISED PRODUCT MOMENT DISTRIBUTION IN SAMPLES FROM A NORMAL MULTIVARIATE POPULATION.

By JOHN WISHART, M.A., B.Sc. Statistical Department, Rothamsted Experimental Station.

John Wishart

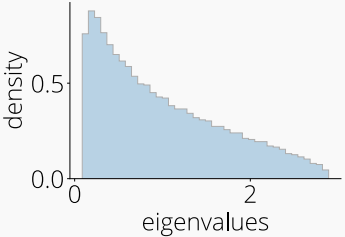


1928: Eigenvalues of Normal Covariance Matrices

THE GENERALISED PRODUCT MOMENT DISTRIBUTION IN SAMPLES FROM A NORMAL MULTIVARIATE POPULATION.

By JOHN WISHART, M.A., B.Sc. Statistical Department, Rothamsted Experimental Station.

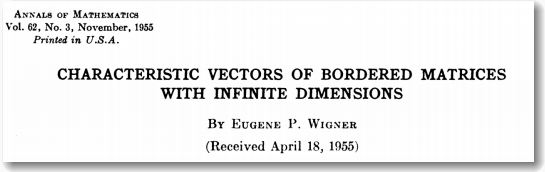
John Wishart



$$W = XX^\top, X_{ij} \sim \mathcal{N}(0,1)$$

$$p(\lambda_1, \ldots, \lambda_N) \propto e^{-\frac{1}{2}\sum_i \lambda_i} \prod \lambda_i^{(n-p-1)/2} \prod_{i<j} |\lambda_i - \lambda_j|$$

1955: Random Symmetric Matrices



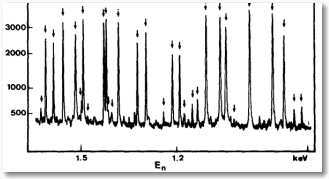
ANNALS OF MATHEMATICS
Vol. 62, No. 3, November, 1955
Printed in U.S.A.

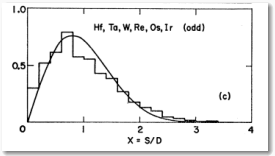
CHARACTERISTIC VECTORS OF BORDERED MATRICES WITH INFINITE DIMENSIONS

BY EUGENE P. WIGNER

(Received April 18, 1955)

Eugene Wigner



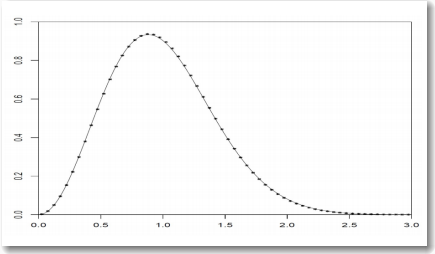


Energy levels of heavy nuclei, compared with the random matrix theory prediction.

Source: [Rosenzweig and Porter, 1960]

Model for high-dimensional phenomena

- Number Theory [Montgomery, 1973, Keating, 1993].
- Graph Theory [Erdos and Rényi, 1960].
- Finance [Bouchaud and Potters, 2009].
- Wireless communication [Tulino et al., 2004]
- Machine Learning ...

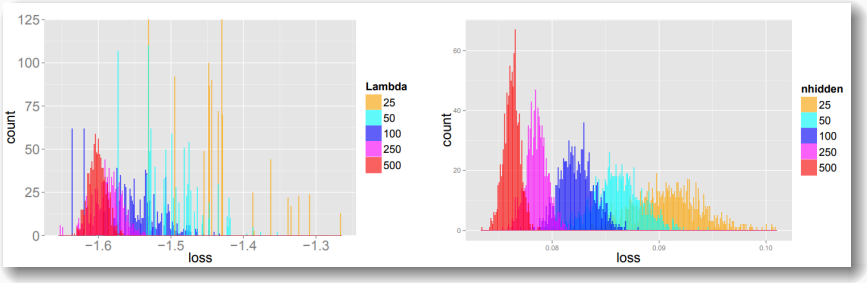


Distribution function of gaps between eigenvalues compared with histogram of gaps between ζ zeros. Source: [Odlyzko, 1987]

Spin Glass model of the Loss Landscape

Early: [Amit et al., 1985, Gardner and Derrida, 1988, Dotsenko, 1995]

Late: [Dauphin et al., 2014, Sagun et al., 2014, Choromanska et al., 2015, Baity-Jesi et al., 2018]

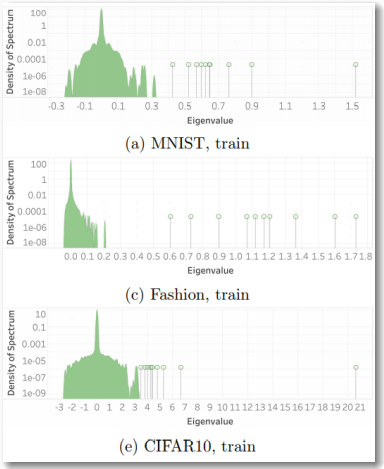


Loss study through spin-glass model. Scaled test losses for the spin-glass (left) and the neural network (right). Source: Choromanska et al. [2015] *The Loss Surfaces of Multilayer Networks*.

Random Matrices in Machine Learning: Loss Landscape

New methods and software[1,2,3] to compute Hessian eigenvalues of large models [Ghorbani et al., 2019, Yao et al., 2020, Papyan, 2020]

[1] *https://github.com/amirgholami/PyHessian*
[2] *https://github.com/google/spectral-density/*
[3] *https://github.com/deep-lab/DeepnetHessian*



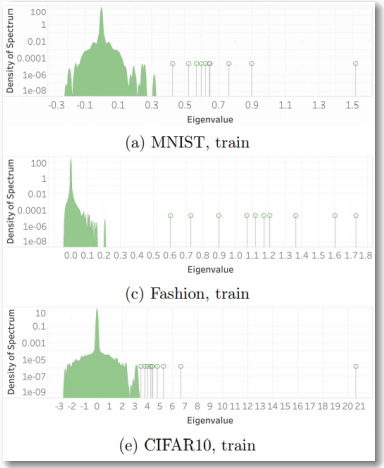
Source: [Papyan, 2020]

Random Matrices in Machine Learning: Loss Landscape

New methods and software[1,2,3] to compute Hessian eigenvalues of large models [Ghorbani et al., 2019, Yao et al., 2020, Papyan, 2020]

[1] *https://github.com/amirgholami/PyHessian*
[2] *https://github.com/google/spectral-density/*
[3] *https://github.com/deep-lab/DeepnetHessian*

RMT model for the Hessian still an open problem [Liao and Mahoney, 2021, Baskerville et al., 2021] ...

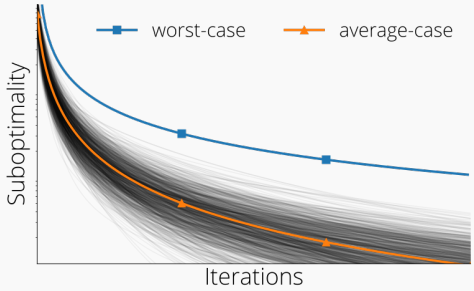


Source: [Papyan, 2020]

Random Matrices in Machine Learning: Numerical Algorithms

Analyze algorithms with **random** data.

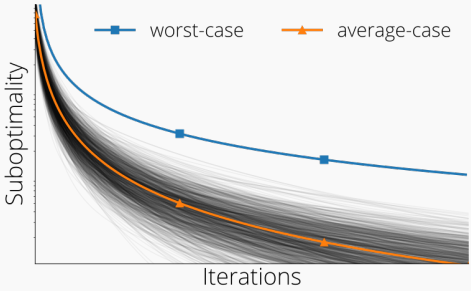
- Simplex [Borgwardt, 1987, Smale, 1983, Spielman and Teng, 2004, Vershynin, 2009] etc.
- Conjugate Gradient [Deift and Trogdon, 2017, Paquette and Trogdon, 2020]
- Acceleration [Pedregosa and Scieur, 2020, Lacotte and Pilanci, 2020]



Random Matrices in Machine Learning: Numerical Algorithms

Analyze algorithms with **random** data.

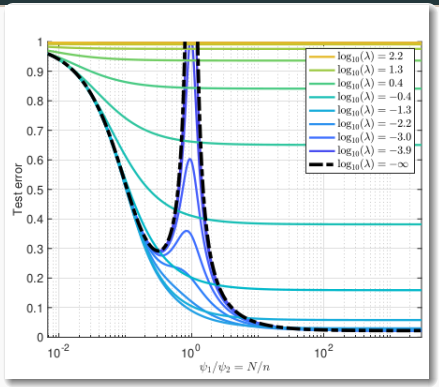
- Simplex [Borgwardt, 1987, Smale, 1983, Spielman and Teng, 2004, Vershynin, 2009] etc.
- Conjugate Gradient [Deift and Trogdon, 2017, Paquette and Trogdon, 2020]
- Acceleration [Pedregosa and Scieur, 2020, Lacotte and Pilanci, 2020]



Topic of Part 3 of this tutorial

Random Matrices in Machine Learning: Generalization

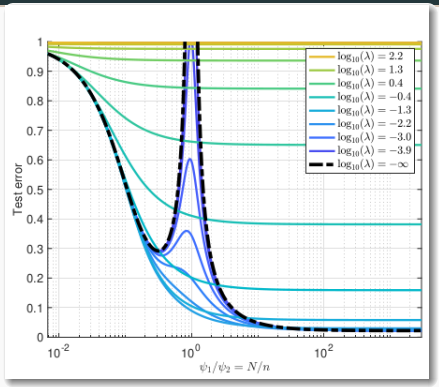
As a model for generalization [Hastie et al., 2019, Mei and Montanari, 2019, Adlam and Pennington, 2020, Liao et al., 2020]



Random Matrices can be used to model the **double descent** generalization curve. Source: [Mei and Montanari, 2019] *The generalization error of random features regression: Precise asymptotics and double descent curve*

Random Matrices in Machine Learning: Generalization

As a model for generalization [Hastie et al., 2019, Mei and Montanari, 2019, Adlam and Pennington, 2020, Liao et al., 2020]



Random Matrices can be used to model the **double descent** generalization curve. Source: [Mei and Montanari, 2019] *The generalization error of random features regression: Precise asymptotics and double descent curve*

Part 4 of this tutorial

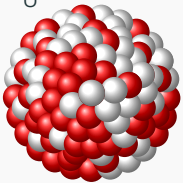
Building Blocks

Classical Random Matrix Ensembles

Gaussian Orthogonal Ensemble (GOE)

Motivation: Model Hamiltonian heavy nuclei

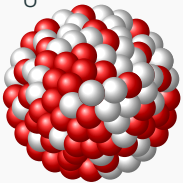
^{238}U



Gaussian Orthogonal Ensemble (GOE)

Motivation: Model Hamiltonian heavy nuclei

^{238}U



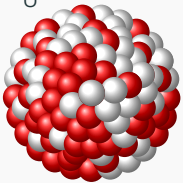
- **Rotational invariant**
  for any fixed orthogonal matrix O,

$$A \overset{\text{law}}{=} O^T A O\,.$$

Gaussian Orthogonal Ensemble (GOE)

Motivation: Model Hamiltonian heavy nuclei

^{238}U



- **Rotational invariant**
  for any fixed orthogonal matrix O,

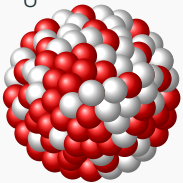
$$A \overset{\text{law}}{=} O^T A O\,.$$

- **Symmetric** matrix.

Gaussian Orthogonal Ensemble (GOE)

Motivation: Model Hamiltonian heavy nuclei

^{238}U



- **Rotational invariant**
  for any fixed orthogonal matrix O,

$$A \overset{\text{law}}{=} O^T A O\,.$$

- **Symmetric** matrix.

- **Independence**
  Entries A_{ij}, $i \leq j$ are independent.

Gaussian Orthogonal Ensemble (GOE)

- real $n \times n$ matrix

$$\frac{1}{\sqrt{n}} \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{13} & a_{14} & \cdots \\ a_{21} & a_{22} & a_{23} & a_{24} & a_{25} & \cdots \\ a_{13} & a_{32} & a_{33} & a_{34} & a_{35} & \cdots \\ a_{41} & a_{42} & a_{43} & a_{44} & a_{45} & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \end{bmatrix}$$

Gaussian Orthogonal Ensemble (GOE)

- real $n \times n$ matrix
- $\mathcal{N}(0, 1)$ above diagonal

$$\frac{1}{\sqrt{n}} \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{13} & a_{14} & \cdots \\ a_{21} & a_{22} & a_{23} & a_{24} & a_{25} & \cdots \\ a_{13} & a_{32} & a_{33} & a_{34} & a_{35} & \cdots \\ a_{41} & a_{42} & a_{43} & a_{44} & a_{45} & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \end{bmatrix}$$

Gaussian Orthogonal Ensemble (GOE)

- real $n \times n$ matrix
- $\mathcal{N}(0, 1)$ above diagonal
- Symmetric

$$\frac{1}{\sqrt{n}} \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{13} & a_{14} & \cdots \\ a_{12} & a_{22} & a_{23} & a_{24} & a_{25} & \cdots \\ a_{13} & a_{23} & a_{33} & a_{34} & a_{35} & \cdots \\ a_{14} & a_{24} & a_{34} & a_{44} & a_{45} & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \end{bmatrix}$$

Gaussian Orthogonal Ensemble (GOE)

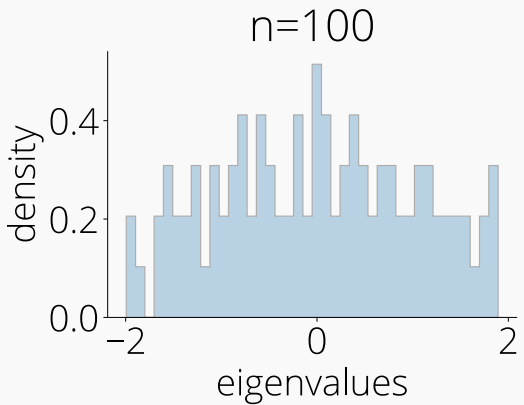
- real $n \times n$ matrix
- $\mathcal{N}(0,1)$ above diagonal
- Symmetric
- $\mathcal{N}(0,2)$ diagonal

$$\frac{1}{\sqrt{n}} \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{13} & a_{14} & \cdots \\ a_{12} & a_{22} & a_{23} & a_{24} & a_{25} & \cdots \\ a_{13} & a_{23} & a_{33} & a_{34} & a_{35} & \cdots \\ a_{14} & a_{24} & a_{34} & a_{44} & a_{45} & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \end{bmatrix}$$

Eigenvalues of GOE

Python pseudocode

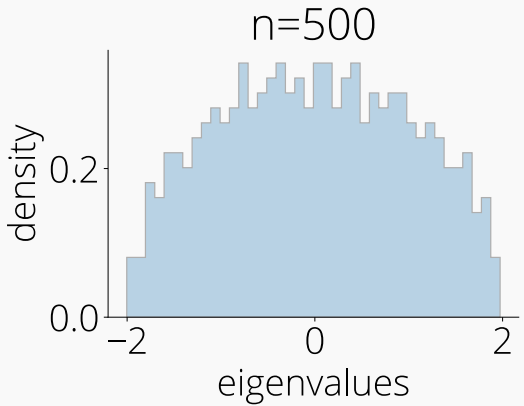
```
import numpy as np
import matplotlib.pyplot as plt

A = np.random.randn(n, n)
GOE = (A+A.T)/np.sqrt(2*n)

eig = np.linalg.eigvals(GOE)
plt.hist(eig)
```



Eigenvalues of GOE

Python pseudocode

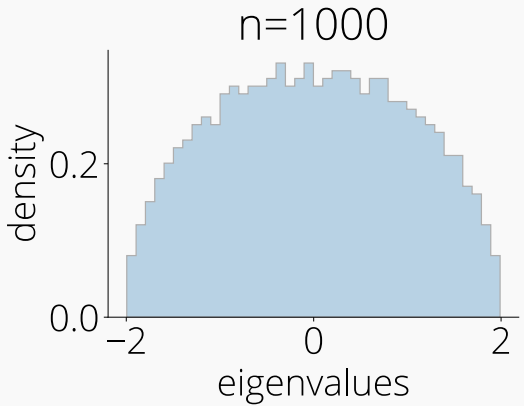
```
import numpy as np
import matplotlib.pyplot as plt

A = np.random.randn(n, n)
GOE = (A+A.T)/np.sqrt(2*n)

eig = np.linalg.eigvals(GOE)
plt.hist(eig)
```



Eigenvalues of GOE

Python pseudocode

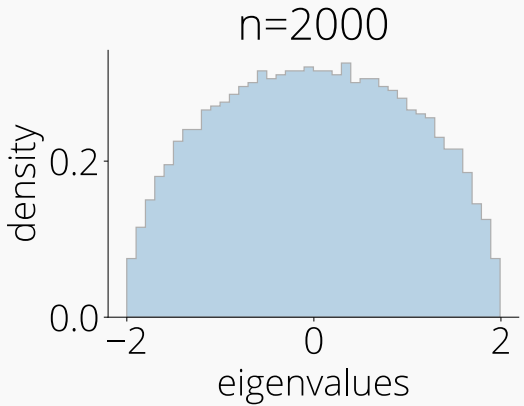
```
import numpy as np
import matplotlib.pyplot as plt

A = np.random.randn(n, n)
GOE = (A+A.T)/np.sqrt(2*n)

eig = np.linalg.eigvals(GOE)
plt.hist(eig)
```



Eigenvalues of GOE

Python pseudocode

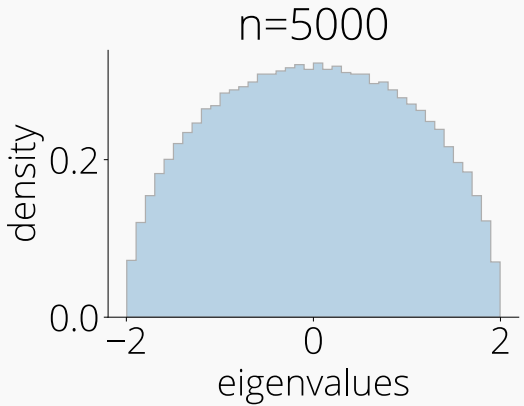
```
import numpy as np
import matplotlib.pyplot as plt

A = np.random.randn(n, n)
GOE = (A+A.T)/np.sqrt(2*n)

eig = np.linalg.eigvals(GOE)
plt.hist(eig)
```



Eigenvalues of GOE

Python pseudocode

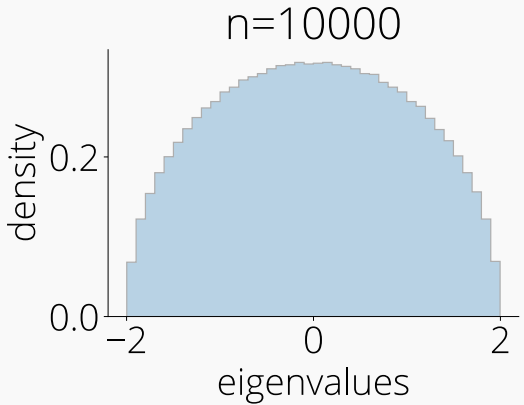
```
import numpy as np
import matplotlib.pyplot as plt

A = np.random.randn(n, n)
GOE = (A+A.T)/np.sqrt(2*n)

eig = np.linalg.eigvals(GOE)
plt.hist(eig)
```



Eigenvalues of GOE

Python pseudocode

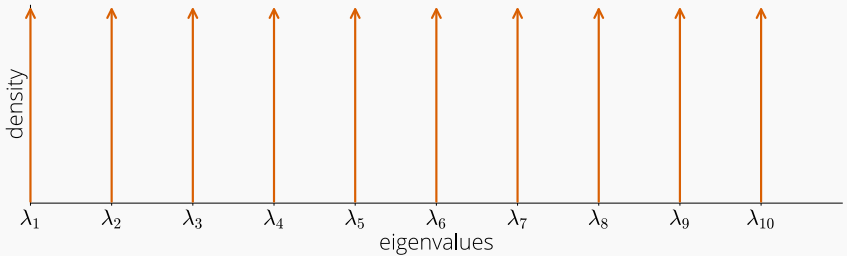
```
import numpy as np
import matplotlib.pyplot as plt

A = np.random.randn(n, n)
GOE = (A+A.T)/np.sqrt(2*n)

eig = np.linalg.eigvals(GOE)
plt.hist(eig)
```



Empirical Spectral Distribution (ESD)

ESD of matrix A_n = p.d.f. of an eigenvalue chosen uniformly at random

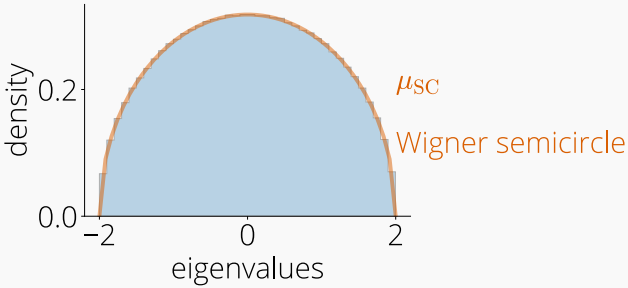
$$\mu_{\text{ESD}} = \mu_n \overset{\text{def}}{=} \frac{1}{n} \sum_{i=1}^{n} \delta_{\lambda_i(A_n)}.$$



Wigner Semicircle Law

μ_{ESD} converges as $n \to \infty$ to the semicircular distribution,

$$\mu_{\text{SC}}(x) \overset{\text{def}}{=} \frac{1}{2\pi}\sqrt{(4-x^2)_+}\,\mathrm{d}x\,.$$



To know more: [Tao, 2012, Bai and Silverstein, 2010].

Wishart ensemble

Wishart

- X = random $(d \times n)$ matrix with entries i.i.d. $\mathcal{N}(0,1)$
- Wishart $(d \times d)$ matrix, $W = \frac{XX^T}{n}$

Remarks

- W is symmetric, positive semi-definite

Wishart ensemble

Wishart

- X = random $(d \times n)$ matrix with entries i.i.d. $\mathcal{N}(0,1)$
- Wishart $(d \times d)$ matrix, $W = \frac{XX^T}{n}$

Remarks

- W is symmetric, positive semi-definite
- Same non-zero eigenvalues than $\frac{1}{n}X^TX$

Wishart ensemble

Wishart

- X = random $(d \times n)$ matrix with entries i.i.d. $\mathcal{N}(0,1)$
- Wishart $(d \times d)$ matrix, $W = \frac{XX^T}{n}$

Remarks

- W is symmetric, positive semi-definite
- Same non-zero eigenvalues than $\frac{1}{n}X^TX$
- $\frac{1}{n}X^TX$ is Hessian of the least squares problem $\frac{1}{2n}\|Xw - y\|^2$

Wishart ensemble

Wishart

- X = random $(d \times n)$ matrix with entries i.i.d. $\mathcal{N}(0,1)$
- Wishart $(d \times d)$ matrix, $W = \frac{XX^T}{n}$

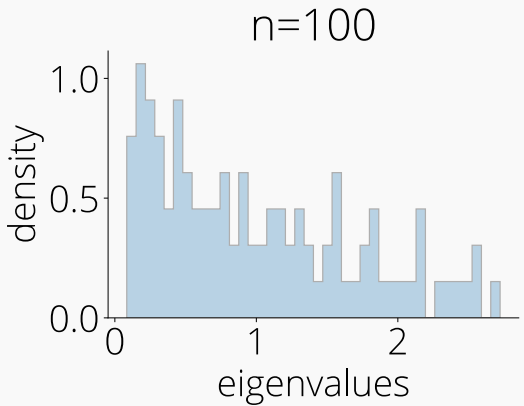
Remarks

- W is symmetric, positive semi-definite
- Same non-zero eigenvalues than $\frac{1}{n}X^TX$
- $\frac{1}{n}X^TX$ is Hessian of the least squares problem $\frac{1}{2n}\|Xw - y\|^2$
- Parameter $r = \frac{d}{n}$

Wishart ensemble

Python pseudocode

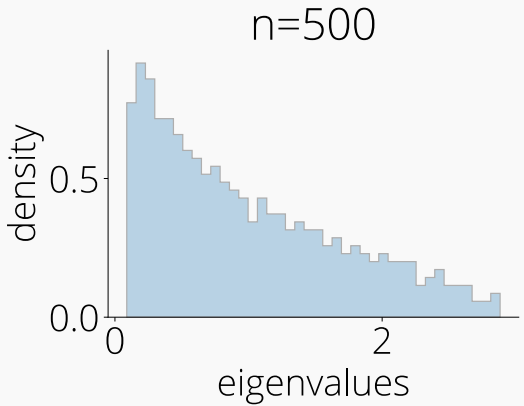
```
import numpy as np
import matplotlib.pyplot as plt

r = 1/2  # for example
X = np.random.randn(n * r, n)
W = np.dot(X, X.T) / n

eig = np.linalg.eigvals(W)
plt.hist(eig)
```



Wishart ensemble

Python pseudocode

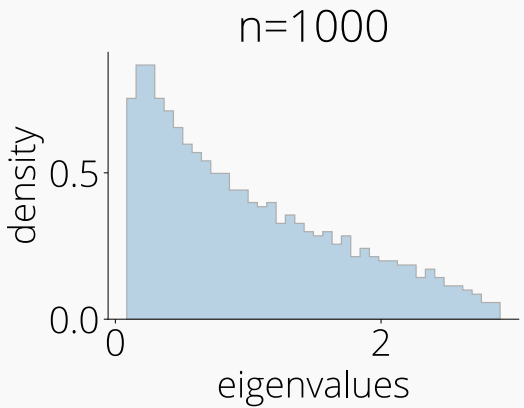
```
import numpy as np
import matplotlib.pyplot as plt

r = 1/2  # for example
X = np.random.randn(n * r, n)
W = np.dot(X, X.T) / n

eig = np.linalg.eigvals(W)
plt.hist(eig)
```



Wishart ensemble

Python pseudocode

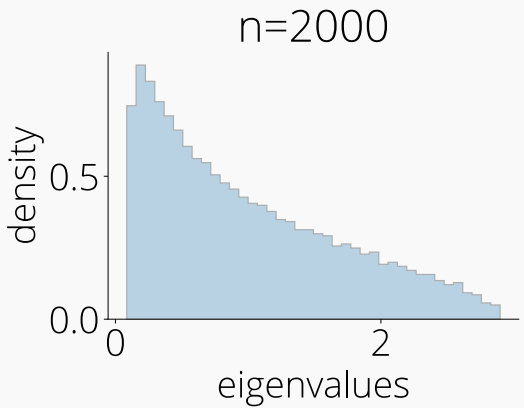
```
import numpy as np
import matplotlib.pyplot as plt

r = 1/2  # for example
X = np.random.randn(n * r, n)
W = np.dot(X, X.T) / n

eig = np.linalg.eigvals(W)
plt.hist(eig)
```



Wishart ensemble

Python pseudocode

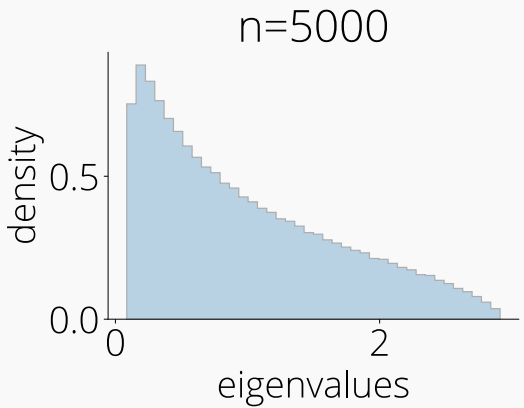
```
import numpy as np
import matplotlib.pyplot as plt

r = 1/2  # for example
X = np.random.randn(n * r, n)
W = np.dot(X, X.T) / n

eig = np.linalg.eigvals(W)
plt.hist(eig)
```



Wishart ensemble

Python pseudocode

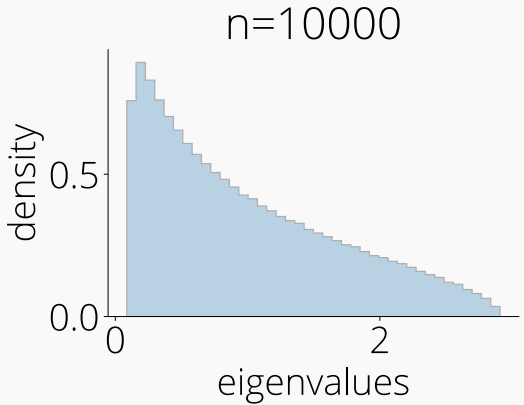
```
import numpy as np
import matplotlib.pyplot as plt

r = 1/2  # for example
X = np.random.randn(n * r, n)
W = np.dot(X, X.T) / n

eig = np.linalg.eigvals(W)
plt.hist(eig)
```



Wishart ensemble

Python pseudocode

```
import numpy as np
import matplotlib.pyplot as plt

r = 1/2  # for example
X = np.random.randn(n * r, n)
W = np.dot(X, X.T) / n

eig = np.linalg.eigvals(W)
plt.hist(eig)
```



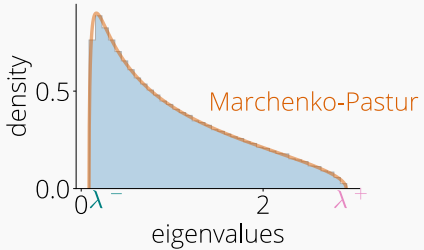
Limit of Wishart matrices

Marchenko-Pastur (MP) law [Marčenko and Pastur, 1967]

As $n, d \to \infty$, $\frac{d}{n} \to r$, μ_{ESD} converges to the Marchenko-Pastur distribution:

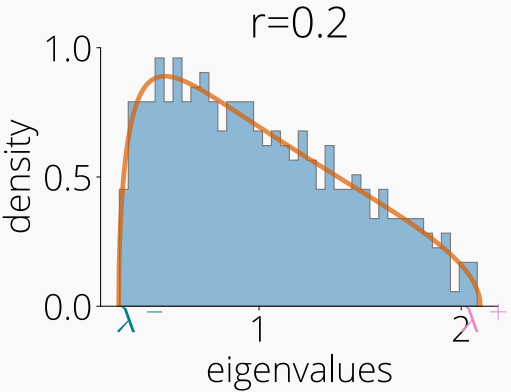
$$\mu_{\text{MP}}(x) \overset{\text{def}}{=} \underbrace{(1 - \tfrac{1}{r})_+ \delta_0(x)}_{\text{nonzero if } r > 1} + \frac{\sqrt{(\lambda^+ - x)(x - \lambda^-)}}{2\pi r x} 1_{x \in [\lambda^-, \lambda^+]} \, \mathrm{d}x \,.$$

$$\text{with } \lambda^- = (1 - \sqrt{r})^2, \lambda^+ = (1 + \sqrt{r})^2$$



The $r = \frac{d}{n}$ parameter

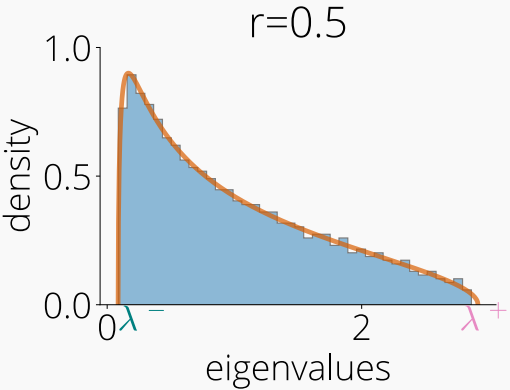
- $r < 1 \implies d < n \implies W$ is product of two **fat** matrices.
- $r > 1 \implies d > n \implies W$ is product of two **thin** matrices (rank-deficient).



$$\mu_{\mathrm{MP}}(x) \overset{\text{def}}{=} \underbrace{(1 - \tfrac{1}{r})_+ \delta_0(x)}_{\text{nonzero if } r > 1} + \frac{\sqrt{(\lambda^+ - x)(x - \lambda^-)}}{2\pi r x} 1_{x \in [\lambda^-, \lambda^+]} \, \mathrm{d}x \, .$$

The $r = \frac{d}{n}$ parameter

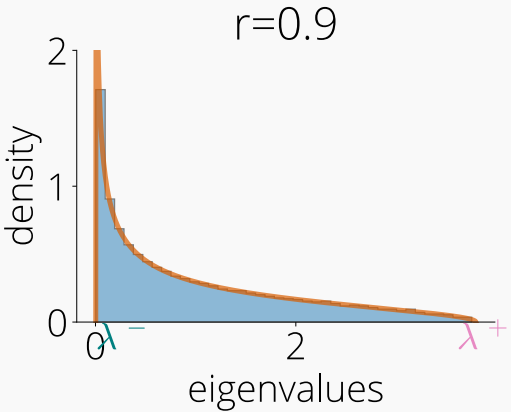
- $r < 1 \implies d < n \implies W$ is product of two **fat** matrices.
- $r > 1 \implies d > n \implies W$ is product of two **thin** matrices (rank-deficient).



$$\mu_{\mathrm{MP}}(x) \stackrel{\text{def}}{=} \underbrace{(1-\tfrac{1}{r})_+ \delta_0(x)}_{\text{nonzero if } r>1} + \frac{\sqrt{(\lambda^+ - x)(x-\lambda^-)}}{2\pi r x} 1_{x\in[\lambda^-,\lambda^+]}\,\mathrm{d}x\,.$$

The $r = \frac{d}{n}$ parameter

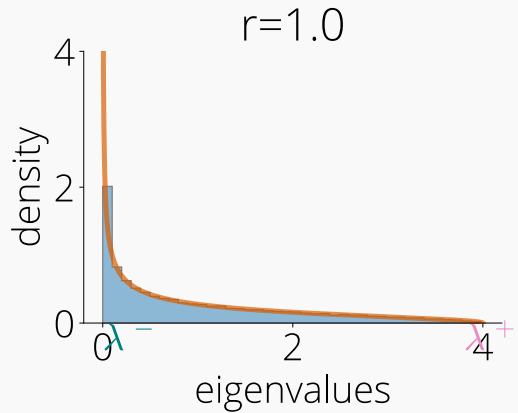
- $r < 1 \implies d < n \implies W$ is product of two **fat** matrices.
- $r > 1 \implies d > n \implies W$ is product of two **thin** matrices (rank-deficient).



$$\mu_{\mathrm{MP}}(x) \stackrel{\text{def}}{=} \underbrace{(1-\tfrac{1}{r})_+\delta_0(x)}_{\text{nonzero if } r>1} + \frac{\sqrt{(\lambda^+ - x)(x-\lambda^-)}}{2\pi r x} 1_{x\in[\lambda^-,\lambda^+]}\,\mathrm{d}x\,.$$

The $r = \frac{d}{n}$ parameter

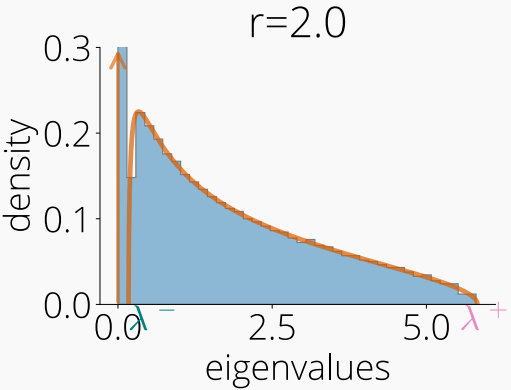
- $r < 1 \implies d < n \implies W$ is product of two **fat** matrices.
- $r > 1 \implies d > n \implies W$ is product of two **thin** matrices (rank-deficient).



$$\mu_{\mathrm{MP}}(x) \stackrel{\text{def}}{=} \underbrace{(1 - \tfrac{1}{r})_+ \delta_0(x)}_{\text{nonzero if } r > 1} + \frac{\sqrt{(\lambda^+ - x)(x - \lambda^-)}}{2\pi r x} 1_{x \in [\lambda^-, \lambda^+]} \, \mathrm{d}x \,.$$

The $r = \frac{d}{n}$ parameter

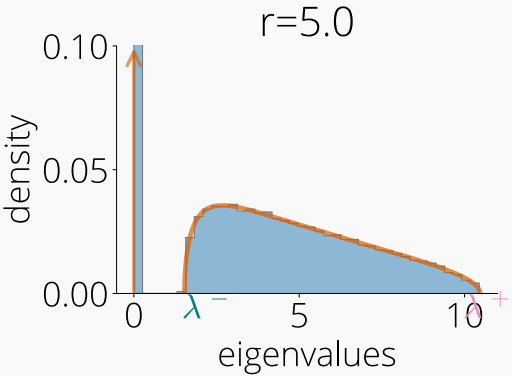
- $r < 1 \implies d < n \implies W$ is product of two **fat** matrices.
- $r > 1 \implies d > n \implies W$ is product of two **thin** matrices (rank-deficient).



$$\mu_{\mathrm{MP}}(x) \stackrel{\text{def}}{=} \underbrace{(1 - \tfrac{1}{r})_+ \delta_0(x)}_{\text{nonzero if } r > 1} + \frac{\sqrt{(\lambda^+ - x)(x - \lambda^-)}}{2\pi r x} 1_{x \in [\lambda^-, \lambda^+]} \, \mathrm{d}x \,.$$

The $r = \frac{d}{n}$ parameter

- $r < 1 \implies d < n \implies W$ is product of two **fat** matrices.
- $r > 1 \implies d > n \implies W$ is product of two **thin** matrices (rank-deficient).



$$\mu_{\mathrm{MP}}(x) \overset{\text{def}}{=} \underbrace{(1 - \tfrac{1}{r})_+ \delta_0(x)}_{\text{nonzero if } r > 1} + \frac{\sqrt{(\lambda^+ - x)(x - \lambda^-)}}{2\pi r x} 1_{x \in [\lambda^-, \lambda^+]} \, \mathrm{d}x \,.$$

Universality

Covariance matrices $W = \frac{1}{n} X X^\top$

What happens if we replace the $\mathcal{N}(0,1)$ with a different distribution?

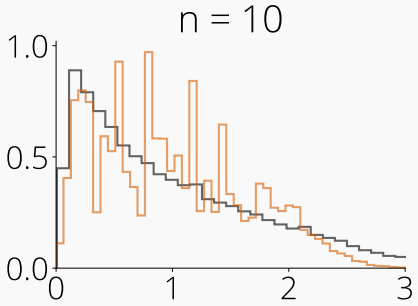
- $X_{ij} \sim \mathcal{N}(0,1)$
- $X_{ij} \sim$ Rademacher $\Pr(X_{ij} = -1) = \Pr(X_{ij} = 1) = \frac{1}{2}$

Universality

Covariance matrices $W = \frac{1}{n} XX^\top$

What happens if we replace the $\mathcal{N}(0,1)$ with a different distribution?

- $X_{ij} \sim \mathcal{N}(0,1)$
- $X_{ij} \sim$ Rademacher $\Pr(X_{ij} = -1) = \Pr(X_{ij} = 1) = \frac{1}{2}$

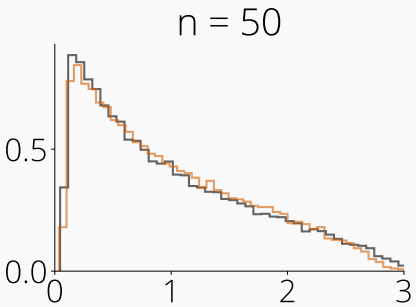


Universality

Covariance matrices $W = \frac{1}{n} XX^\top$

What happens if we replace the $\mathcal{N}(0,1)$ with a different distribution?

- $X_{ij} \sim \mathcal{N}(0,1)$
- $X_{ij} \sim$ Rademacher $\Pr(X_{ij} = -1) = \Pr(X_{ij} = 1) = \frac{1}{2}$

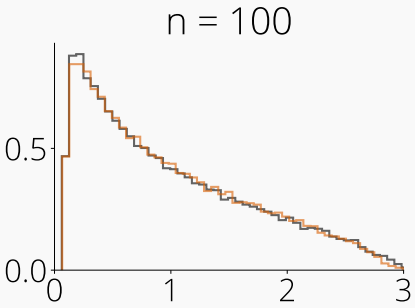


Universality

Covariance matrices $W = \frac{1}{n} XX^\top$

What happens if we replace the $\mathcal{N}(0,1)$ with a different distribution?

- $X_{ij} \sim \mathcal{N}(0,1)$
- $X_{ij} \sim$ Rademacher $\Pr(X_{ij} = -1) = \Pr(X_{ij} = 1) = \frac{1}{2}$

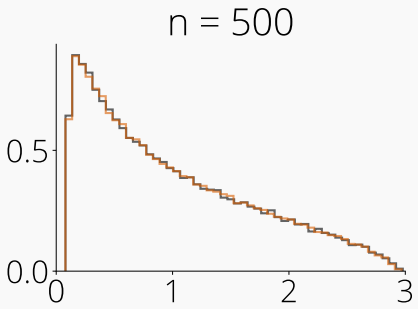


Universality

Covariance matrices $W = \frac{1}{n} XX^\top$

What happens if we replace the $\mathcal{N}(0,1)$ with a different distribution?

- $X_{ij} \sim \mathcal{N}(0,1)$
- $X_{ij} \sim$ Rademacher $\Pr(X_{ij} = -1) = \Pr(X_{ij} = 1) = \frac{1}{2}$

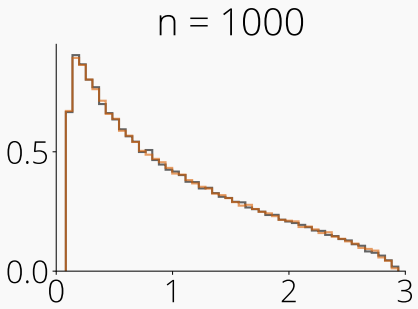


Universality

Covariance matrices $W = \frac{1}{n} XX^\top$

What happens if we replace the $\mathcal{N}(0,1)$ with a different distribution?

- $X_{ij} \sim \mathcal{N}(0,1)$
- $X_{ij} \sim$ Rademacher $\Pr(X_{ij} = -1) = \Pr(X_{ij} = 1) = \frac{1}{2}$



Universality

Universality

- Statistics only mildly depend on the lower order moments of distribution of the entries

Universality

Universality

- Statistics only mildly depend on the lower order moments of distribution of the entries

Example: Marchenko-Pastur [Marčenko and Pastur, 1967]

Let X be a $d \times n$ random matrix with i.i.d. entries that verifies

$$\mathbb{E}[X_{ij}] = 0, \quad \mathbb{E}[X_{ij}^2] = 1, \quad \mathbb{E}[X_{ij}^4] < \infty$$

Universality: As $n, d \to \infty$ with $\frac{d}{n} \to r$, the ESD of $W = \frac{XX^T}{n}$ converges to Marchenko-Pastur(r)

May others ...

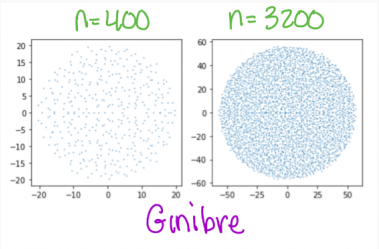
Other matrix ensembles

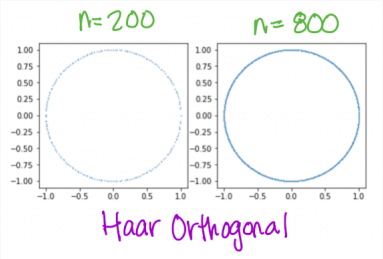
- **Ginibre.** Let G_n be $n \times n$ matrix of i.i.d. $N(0,1)$, (bilinear games [Domingo-Enrich et al., 2020])

$$\text{(Circle law)} \quad \text{ESD of } G_n/\sqrt{n} \to \text{Unif(disk)}\,.$$

- Uniform probability measure on **orthogonal matrices**. $V \sim \text{Unif}(O(n))$,

$$\text{ESD of } V \to \text{Unif}(\mathbb{S}^1)\,.$$





References

Ben Adlam and Jeffrey Pennington. The neural tangent kernel in high dimensions: Triple descent and a multi-scale theory of generalization. In *International Conference on Machine Learning*. PMLR, 2020. URL https://arxiv.org/pdf/2008.06786.pdf.

Daniel J Amit, Hanoch Gutfreund, and Haim Sompolinsky. Spin-glass models of neural networks. *Physical Review A*, 1985. URL https://doi.org/10.1103/PhysRevA.32.1007.

Z. Bai and J. Silverstein. *Spectral analysis of large dimensional random matrices*, volume 20. Springer, 2010.

Marco Baity-Jesi, Levent Sagun, Mario Geiger, Stefano Spigler, Gérard Ben Arous, Chiara Cammarota, Yann LeCun, Matthieu Wyart, and Giulio Biroli. Comparing dynamics: Deep neural networks versus glassy systems. In *International Conference on Machine Learning*. PMLR, 2018. URL https://arxiv.org/pdf/1803.06969.pdf.

Nicholas P Baskerville, Diego Granziol, and Jonathan P Keating. Applicability of random matrix theory in deep learning. *arXiv preprint arXiv:2102.06740*, 2021. URL https://arxiv.org/pdf/2102.06740.pdf.

K H Borgwardt. *The simplex method: A probabilistic analysis*. Springer–Verlag, Berlin, Heidelberg, 1987. URL https://www.springer.com/gp/book/9783540170969.

References ii

Jean-Philippe Bouchaud and Marc Potters. Financial applications of random matrix theory: a short review. *arXiv preprint arXiv:0910.1205*, 2009. URL https://arxiv.org/pdf/0910.1205.

Anna Choromanska, Mikael Henaff, Michael Mathieu, Gérard Ben Arous, and Yann LeCun. The loss surfaces of multilayer networks. In *Artificial intelligence and statistics*. PMLR, 2015. URL https://arxiv.org/pdf/1412.0233.pdf.

Yann Dauphin, Razvan Pascanu, Caglar Gulcehre, Kyunghyun Cho, Surya Ganguli, and Yoshua Bengio. Identifying and attacking the saddle point problem in high-dimensional non-convex optimization. *arXiv preprint arXiv:1406.2572*, 2014. URL https://arxiv.org/pdf/1406.2572.pdf.

P Deift and T Trogdon. Universality for Eigenvalue Algorithms on Sample Covariance Matrices. *SIAM Journal on Numerical Analysis*, 2017. URL http://arxiv.org/abs/1701.01896.

Carles Domingo-Enrich, Fabian Pedregosa, and Damien Scieur. Average-case acceleration for bilinear games and normal matrices. *arXiv preprint arXiv:2010.02076*, 2020.

Viktor Dotsenko. *An introduction to the theory of spin glasses and neural networks*. World Scientific, 1995. URL https://doi.org/10.1142/2460.

Paul Erdos and Alfréd Rényi. On the evolution of random graphs. *Publ. Math. Inst. Hung. Acad. Sci*, 1960. URL http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.348.530&rep=rep1&type=pdf.

Elizabeth Gardner and Bernard Derrida. Optimal storage properties of neural network models. *Journal of Physics A: Mathematical and general*, 1988. URL http://www.phys.ens.fr/~derrida/PAPIERS/1988/optimal-storage.pdf.

References iii

Behrooz Ghorbani, Shankar Krishnan, and Ying Xiao. An investigation into neural net optimization via hessian eigenvalue density. In *International Conference on Machine Learning*. PMLR, 2019. URL https://arxiv.org/pdf/1901.10159.pdf.

Trevor Hastie, Andrea Montanari, Saharon Rosset, and Ryan J Tibshirani. Surprises in high-dimensional ridgeless least squares interpolation. *arXiv preprint arXiv:1903.08560*, 2019. URL https://arxiv.org/pdf/1903.08560.pdf.

J Keating. The Riemann zeta-function and quantum chaology. In *Quantum chaos*. Elsevier, 1993. URL https://arxiv.org/pdf/0708.4223.pdf.

Jonathan Lacotte and Mert Pilanci. Optimal randomized first-order methods for least-squares problems. In *International Conference on Machine Learning*. PMLR, 2020. URL https://arxiv.org/pdf/2002.09488.pdf.

Z Liao and M W Mahoney. Hessian Eigenspectra of More Realistic Nonlinear Models. mar 2021. URL http://arxiv.org/abs/2103.01519.

Zhenyu Liao, Romain Couillet, and Michael W Mahoney. A random matrix analysis of random fourier features: beyond the gaussian kernel, a precise phase transition, and the corresponding double descent. *arXiv preprint arXiv:2006.05013*, 2020. URL https://arxiv.org/pdf/2006.05013.pdf.

Vladimir A Marčenko and Leonid Andreevich Pastur. Distribution of eigenvalues for some sets of random matrices. *Mathematics of the USSR-Sbornik*, 1967. URL https://iopscience.iop.org/article/10.1070/SM1967v001n04ABEH001994/meta.

Song Mei and Andrea Montanari. The generalization error of random features regression: Precise asymptotics and double descent curve. *arXiv preprint arXiv:1908.05355*, 2019. URL https://arxiv.org/pdf/1908.05355.pdf.

References iv

Hugh L Montgomery. The pair correlation of zeros of the zeta function. In *Proc. Symp. Pure Math*, 1973. URL http://www-personal.umich.edu/~hlm/paircor1.pdf.

Andrew M Odlyzko. On the distribution of spacings between zeros of the zeta function. *Mathematics of Computation*, 1987. URL https://doi.org/10.1090/S0025-5718-1987-0866115-0.

Vardan Papyan. Traces of class/cross-class structure pervade deep learning spectra. *Journal of Machine Learning Research*, 2020. URL http://jmlr.org/papers/v21/20-933.html.

Elliot Paquette and Thomas Trogdon. Universality for the conjugate gradient and minres algorithms on sample covariance matrices. *arXiv preprint arXiv:2007.00640*, 2020. URL https://arxiv.org/pdf/2007.00640.pdf.

Fabian Pedregosa and Damien Scieur. Acceleration through spectral density estimation. In *International Conference on Machine Learning*. PMLR, 2020. URL https://arxiv.org/pdf/2002.04756.pdf.

Norbert Rosenzweig and Charles E. Porter. "repulsion of energy levels" in complex atomic spectra. *Phys. Rev.*, 1960. URL https://link.aps.org/doi/10.1103/PhysRev.120.1698.

Levent Sagun, V Ugur Guney, Gerard Ben Arous, and Yann LeCun. Explorations on high dimensional landscapes. *arXiv preprint arXiv:1412.6615*, 2014. URL https://arxiv.org/pdf/1412.6615.pdf.

S Smale. On the average number of steps of the simplex method of linear programming. *Mathematical Programming*, 27(3): 241–262, oct 1983. ISSN 0025-5610. doi: 10.1007/BF02591902. URL http://link.springer.com/10.1007/BF02591902.

References v

D A Spielman and S-H Teng. Smoothed analysis of algorithms. *Journal of the ACM*, 51(3):385–463, may 2004. ISSN 00045411. doi: 10.1145/990308.990310. URL *http://dl.acm.org/citation.cfm?id=990308.990310*.

Terence Tao. *Topics in random matrix theory*. American Mathematical Soc., 2012.

Antonia M Tulino, Sergio Verdú, and Sergio Verdu. *Random matrix theory and wireless communications*. Now Publishers Inc, 2004. URL *http://dx.doi.org/10.1561/0100000001*.

R Vershynin. Beyond Hirsch Conjecture: Walks on Random Polytopes and Smoothed Complexity of the Simplex Method. *SIAM Journal on Computing*, 39, 2009. URL *http://epubs.siam.org/doi/10.1137/070683386*.

Eugene Wigner. Characteristic vectors of bordered matrices with infinite dimensions. *Annals of Mathematics*, 1955. URL *https://doi.org/10.1007/978-3-662-02781-3_35*.

John Wishart. The generalised product moment distribution in samples from a normal multivariate population. *Biometrika*, 1928. URL *https://doi.org/10.2307/2331939*.

Zhewei Yao, Amir Gholami, Kurt Keutzer, and Michael W Mahoney. Pyhessian: Neural networks through the lens of the hessian. In *2020 IEEE International Conference on Big Data (Big Data)*. IEEE, 2020. URL *https://github.com/amirgholami/PyHessian*.

Random Matrix Theory for Machine Learning

Introduction to Random Matrix Theory

Fabian Pedregosa, Courtney Paquette, Tom Trogdon, Jeffrey Pennington

https://random-matrix-learning.github.io

Table of contents

1. Stieltjes Transform

2. *R*-Transform

Stieltjes Transform

Maximum entropy principle

A disordered (real world) system will be random in all ways that are not explicitly prevented.

Conversely, a matrix is interesting only in those ways it fails to look like a random matrix.

Notes of Elliot Paquette and the thesis, *A random matrix framework for large dimensional machine learning and neural networks* by Zhenyu Liao

Example MNIST

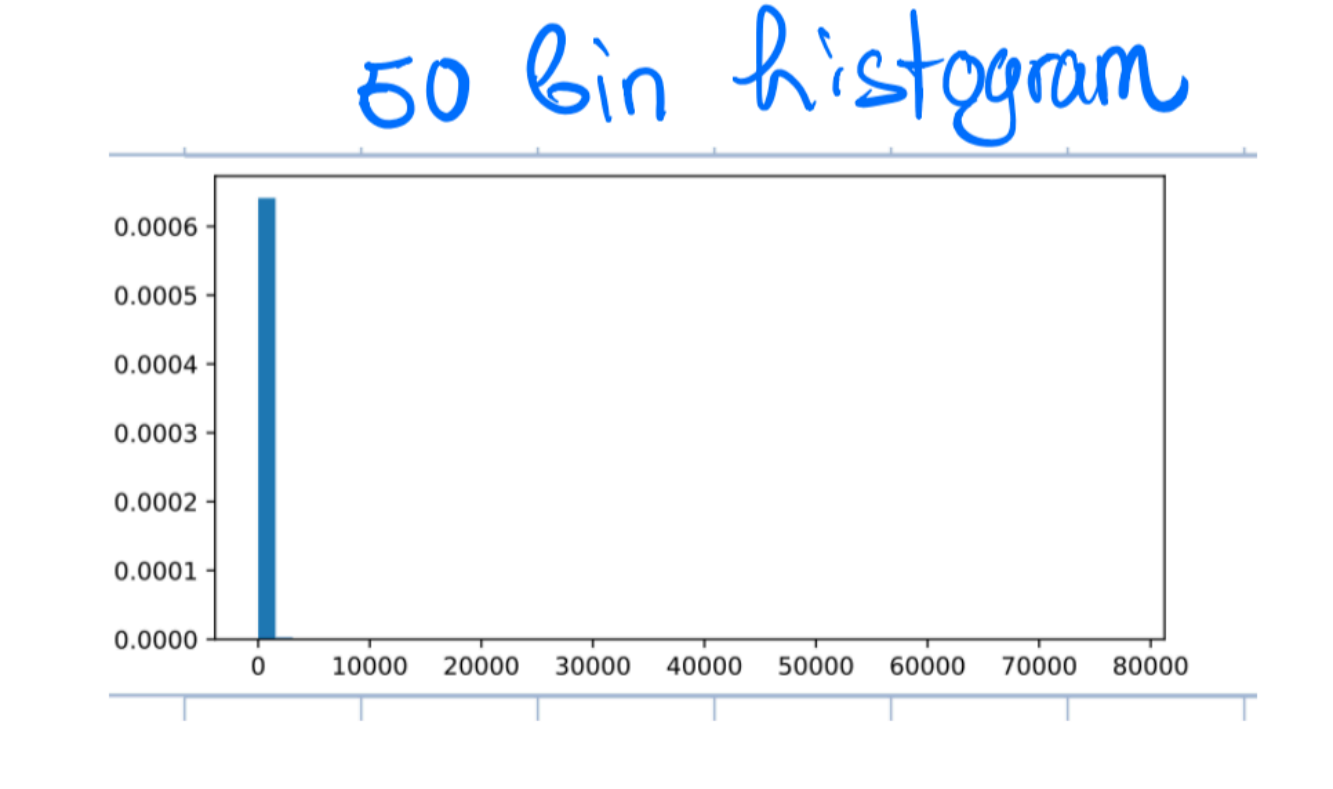
MNIST $\boldsymbol{M}$ $(60,000 \times 28 \times 28)$, form sample covariance matrix, $\boldsymbol{S} = \boldsymbol{M}\boldsymbol{M}^T$

Does $\boldsymbol{S}$ look like a random matrix?

Example MNIST

MNIST M $(60,000 \times 28 \times 28)$, form sample covariance matrix, $S = MM^T$

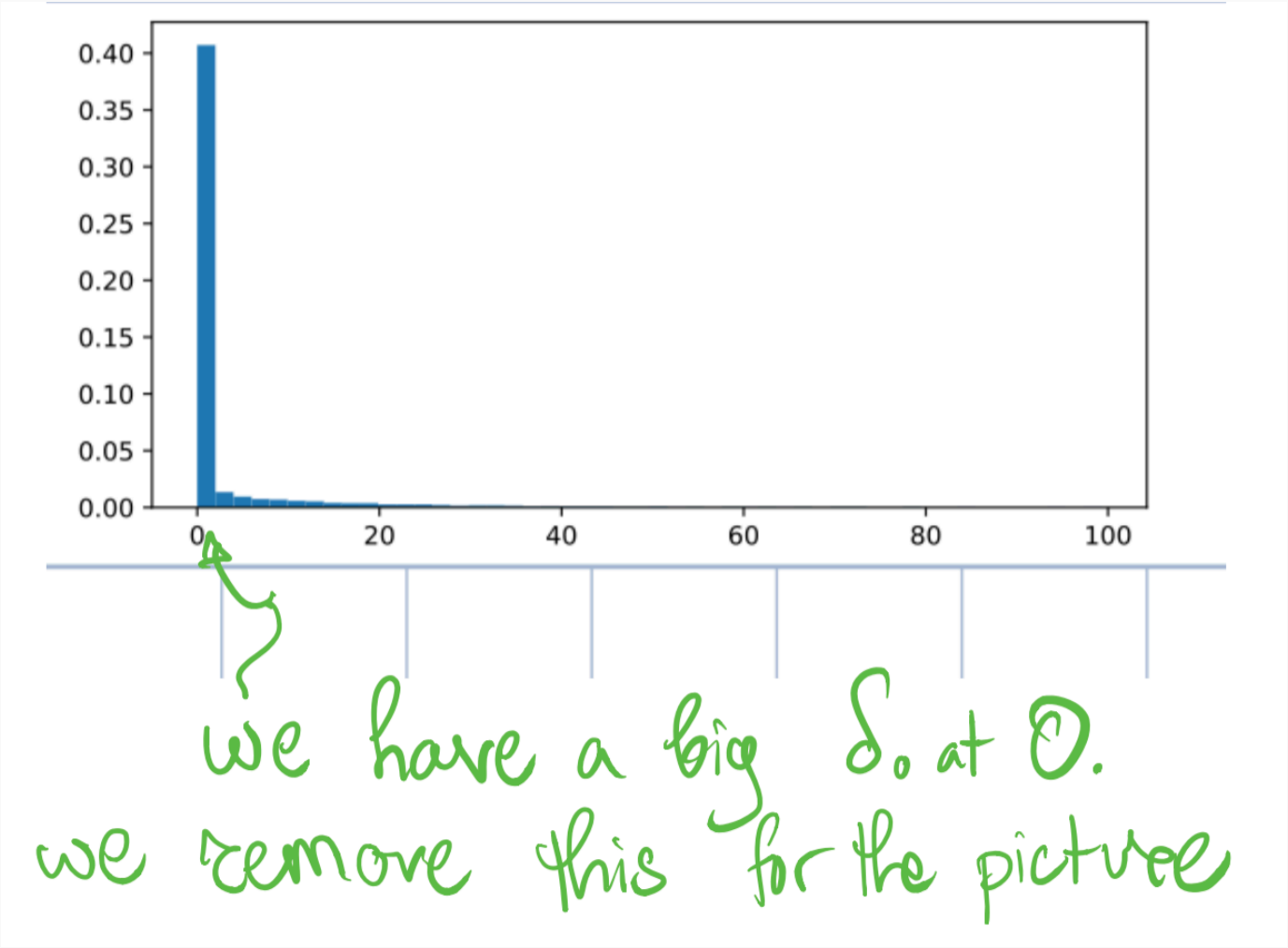
Does S look like a random matrix?



Observation 1: 1 giant eigenvalue, M has non-zero mean

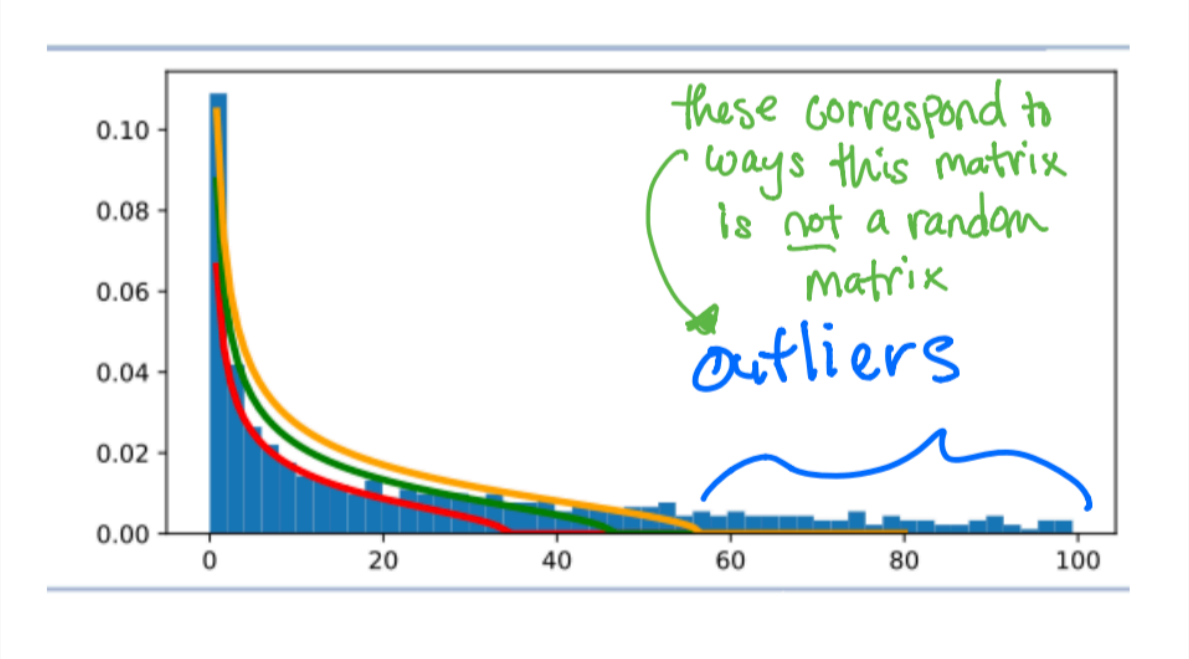
Example MNIST

Remove large eigenvalue



Example MNIST

Observation 2: There is a bulk component



- Fit bulk with Marchenko-Pastur(1) to correspond to point mass
- Large eigenvalue correspond to interesting outliers
- May be ”hidden” (weak) outliers in the bulk eigenvalues

Method 1: Stieltjes transform

Stieltjes transform

$$m_\mu(z) = \int_{\mathbb{R}} \frac{1}{t-z}\,\mu(\mathrm{d}t)$$

$$z \in \mathbb{C}, \quad \Im(z) > 0, \quad \mu \text{ is } \mathbb{P}\text{-measure on } \mathbb{R}$$

Theorem (Stieltjes inversion)

$$\lim_{\varepsilon \downarrow 0} \frac{1}{\pi} \Im\left(m_\mu(x+i\varepsilon)\right) \xrightarrow[\varepsilon \to 0]{} \mu$$

Method 1: Stieltjes transform

Stieltjes transform

$$m_\mu(z) = \int_{\mathbb{R}} \frac{1}{t-z}\,\mu(\mathrm{d}t)$$

$$z \in \mathbb{C}, \quad \Im(z) > 0, \quad \mu \text{ is } \mathbb{P}\text{-measure on } \mathbb{R}$$

Theorem (Stieltjes inversion)

$$\lim_{\varepsilon \downarrow 0} \frac{1}{\pi} \Im\left(m_\mu(x+i\varepsilon)\right) \xrightarrow[\varepsilon \to 0]{} \mu$$

Example: μ is law of Unif$([-1,1])$

Stieltjes transform:

$$m_\mu(z) = \frac{1}{2}\int_{\mathbb{R}} \frac{1}{t-z} 1\left(\{|t| \le 1\}\right) \mathrm{d}t = \frac{1}{2}\int_{-1}^{1} \frac{\mathrm{d}t}{t-z} = \frac{1}{2}\log\left(\frac{1-z}{-1-z}\right)$$

Inversion

$$\lim_{\varepsilon \downarrow 0} \frac{\Im}{\pi}\left(\frac{1}{2}\log\left(\frac{1-z}{-1-z}\right)\right)\Big|_{z=x+i\varepsilon} = \begin{cases} \frac{1}{2}, & \text{if } |x| < 1 \\ 0, & \text{if } |x| > 1 \\ \star, & |x| = 1 \end{cases}$$

Empirical spectral distribution and Stieltjes

$$\text{ESD of } \boldsymbol{A}: \quad \mu_n = \frac{1}{n}\sum_{i=1}^{n} \delta_{\lambda_i(\boldsymbol{A})}$$

Stieltjes transform of ESD

$$m_{\mu_n}(z) = \frac{1}{n}\sum_{i=1}^{n} \frac{1}{\lambda_i(\boldsymbol{A}) - z} = \frac{1}{n}\mathrm{tr}\left((\boldsymbol{A} - z\boldsymbol{I}_d)^{-1}\right)$$

Theorem

If for each $z \in \mathbb{C}$, $\Im(z) > 0$, and

$$m_{\mu_n}(z) \xrightarrow[n\to\infty]{} m_{\mu}(z) \quad \Rightarrow \quad \mu_n \xrightarrow[n\to\infty]{} \mu.$$

Resolvent

Resolvent

$$Q(z) \stackrel{\text{def}}{=} \text{Resolvent of } A = (A - zI_d)^{-1}$$

Remarks

For nice random matrices (GOE, Wishart, sample covariance),

$$\text{Resolvent of } A \approx m_A(z) I_d$$

where m_A is the Stieltjes transform of A. That is, for any unit vector u independent of A,

$$u^T (A - zI_d)^{-1} u \cong m_A(z) \qquad \text{(weak sense)}$$

*This gives not only eigenvalues but also eigenvectors

Marchenko-Pastur and Stieltjes

Lemma:
Suppose $x \in \mathbb{R}^p$ has i.i.d. entries of mean zero, unit variance. Then

$$x^T A x - \mathrm{tr} A \to 0$$

Wishart: $W = \frac{1}{n} X X^T, \quad X \in \mathbb{R}^{d \times n}, \quad d/n \to r \in (0 \infty)$

$$X = \begin{bmatrix} | & | & \cdots & | \\ x_1 & x_2 & \cdots & x_n \\ | & | & \cdots & | \end{bmatrix} \Rightarrow \text{Resolvent of } W = Q_n(z) = \left(\frac{1}{n} \sum_{i=1}^{n} x_i x_i^T - z I_d \right)^{-1}$$

Question: What is the Stieltjes transform of Wishart W?

Suppose $\exists\ Q(z) \in \mathbb{C}^{d \times d}$ s.t. $\frac{1}{d} \mathrm{tr}(Q_n(z) - Q(z)) \to 0$ (Stieltjes of MP $= \mathrm{tr}(Q(z))$)

Fact: $\left| \frac{1}{d} \mathrm{tr}\left(Q_n(z) (Q(z)^{-1} + z I_d) Q(z) \right) - \frac{1}{n} \frac{1}{d} \sum_{i=1}^{n} x_i^T Q(z) Q_n(z) x_i \right| \to 0$

Marchenko-Pastur and Stieltjes

Linear algebra to construct self-consistent equation for $\mathrm{tr}(Q_n(z))$:

Remove 1 column and 1 row:

$$Q_n = \left(\frac{1}{n}\sum_{i=1}^{n} x_i x_i^T - zI_d\right)^{-1} = \left(\frac{1}{n}x_1 x_1^T + \overbrace{\frac{1}{n}\sum_{i=2}^{n} x_i x_i^T - zI_d}^{Q_n^{(1)}}\right)^{-1}$$

$$= Q_n^{(1)} - \frac{n^{-1}Q_n^{(1)}x_1 x_1^T Q_n^{(1)}}{1 + n^{-1}x_1^T Q_n^{(1)} x_1} \quad \text{(Sherman-Morrison)}$$

Marchenko-Pastur and Stieltjes

Linear algebra to construct self-consistent equation for $\mathrm{tr}(Q_n(z))$:

Remove 1 column and 1 row:

$$Q_n = \left(\frac{1}{n}\sum_{i=1}^{n} x_i x_i^T - zI_d\right)^{-1} = \left(\frac{1}{n}x_1x_1^T + \overbrace{\frac{1}{n}\sum_{i=2}^{n} x_i x_i^T - zI_d}^{Q_n^{(1)}}\right)^{-1}$$

$$= Q_n^{(1)} - \frac{n^{-1}Q_n^{(1)}x_1x_1^TQ_n^{(1)}}{1+n^{-1}x_1^TQ_n^{(1)}x_1} \quad \text{(Sherman-Morrison)}$$

$Q_n^{(1)}, x_1$ independent $\Rightarrow x_1^TQQ_n^{(1)}x_1 = tr(QQ_n^{(1)})$ and $x_1^TQ_n^{(1)}x_1 = tr(Q_n^{(1)})$

$$d^{-1}x_1^TQQ_nx_1 \quad = \quad \frac{d^{-1}\mathrm{tr}(QQ_n^{(1)})}{1+n^{-1}\mathrm{tr}(Q_n^{(1)})} \approx \frac{d^{-1}\mathrm{tr}(QQ_n)}{1+n^{-1}\mathrm{tr}(Q_n)}$$

Marchenko-Pastur and Stieltjes

Linear algebra to construct self-consistent equation for $\mathrm{tr}(Q_n(z))$:

Remove 1 column and 1 row:

$$Q_n = \left(\frac{1}{n}\sum_{i=1}^{n} x_i x_i^T - zI_d\right)^{-1} = \left(\frac{1}{n}x_1x_1^T + \overbrace{\frac{1}{n}\sum_{i=2}^{n} x_i x_i^T - zI_d}^{Q_n^{(1)}}\right)^{-1}$$

$$= Q_n^{(1)} - \frac{n^{-1}Q_n^{(1)}x_1x_1^TQ_n^{(1)}}{1+n^{-1}x_1^TQ_n^{(1)}x_1} \quad \text{(Sherman-Morrison)}$$

$Q_n^{(1)}, x_1$ independent $\Rightarrow x_1^TQQ_n^{(1)}x_1 = tr(QQ_n^{(1)})$ and $x_1^TQ_n^{(1)}x_1 = tr(Q_n^{(1)})$

$$d^{-1}x_1^TQQ_nx_1 = \frac{d^{-1}\mathrm{tr}(QQ_n^{(1)})}{1+n^{-1}\mathrm{tr}(Q_n^{(1)})} \approx \frac{d^{-1}\mathrm{tr}(QQ_n)}{1+n^{-1}\mathrm{tr}(Q_n)}$$

$$\begin{array}{ccc} \frac{1}{nd}\sum_{i=1}^{n} x_i^TQQ_nx_i & \approx & \frac{d^{-1}\mathrm{tr}(QQ_n)}{1+n^{-1}\mathrm{tr}(Q_n)} \\ \downarrow & & \downarrow \\ d^{-1}\mathrm{tr}\big(Q_n(Q^{-1}+zI_d)Q\big) & \approx & \frac{d^{-1}\mathrm{tr}(QQ_n)}{1+n^{-1}\mathrm{tr}(Q_n)} \end{array} \Rightarrow \quad Q^{-1} + zI_d = \frac{1}{1+n^{-1}\mathrm{tr}(Q_n)}I_d$$

$$Q = \left(-z + \frac{1}{1+n^{-1}\mathrm{tr}(Q_n)}\right)^{-1} I_d$$

Marchenko-Pastur and Stieltjes

$$\text{Stieltjes of } W_n = m_{W_n}(z) = d^{-1}\mathrm{tr}(Q_n) \approx d^{-1}\mathrm{tr}(Q) = \left(-z + \frac{1}{1 + r \cdot d^{-1}\mathrm{tr}(Q_n)}\right)^{-1}$$

Marchenko-Pastur and Stieltjes

Stieltjes of $W_n = m_{W_n}(z) = d^{-1}\mathrm{tr}(Q_n) \approx d^{-1}\mathrm{tr}(Q) = \left(-z + \frac{1}{1 + r \cdot d^{-1}\mathrm{tr}(Q_n)}\right)^{-1}$

Fixed point equation for trace ($r = \lim_{n\to\infty} \frac{d}{n}$):

$$m_{W_n}(z) = \frac{1}{-z + \frac{1}{1+r\cdot m_{W_n}(z)}} + \varepsilon(n,z), \qquad \varepsilon(n,z) \xrightarrow[n\to\infty]{} 0$$

Fixed point for Marchenko-Pastur, MP:

$$m_{\mathrm{MP}}(z) = \frac{1}{-z + \frac{1}{1+r\cdot m_{\mathrm{MP}}(z)}}, \qquad \text{Note: } m(z) \sim \frac{-1}{z} \text{ as } z \to \infty$$

✓ solve numerically ✓ many dist. satisfy fixed point eqn.

Marchenko-Pastur and Stieltjes

Stieltjes of $W_n = m_{W_n}(z) = d^{-1}\mathrm{tr}(Q_n) \approx d^{-1}\mathrm{tr}(Q) = \left(-z + \frac{1}{1 + r \cdot d^{-1}\mathrm{tr}(Q_n)}\right)^{-1}$

Fixed point equation for trace ($r = \lim_{n\to\infty} \frac{d}{n}$):

$$m_{W_n}(z) = \frac{1}{-z + \frac{1}{1+r\cdot m_{W_n}(z)}} + \varepsilon(n,z), \qquad \varepsilon(n,z) \xrightarrow[n\to\infty]{} 0$$

Fixed point for Marchenko-Pastur, MP:

$$m_{\mathrm{MP}}(z) = \frac{1}{-z + \frac{1}{1+r\cdot m_{\mathrm{MP}}(z)}}, \qquad \text{Note: } m(z) \sim \tfrac{-1}{z} \text{ as } z \to \infty$$

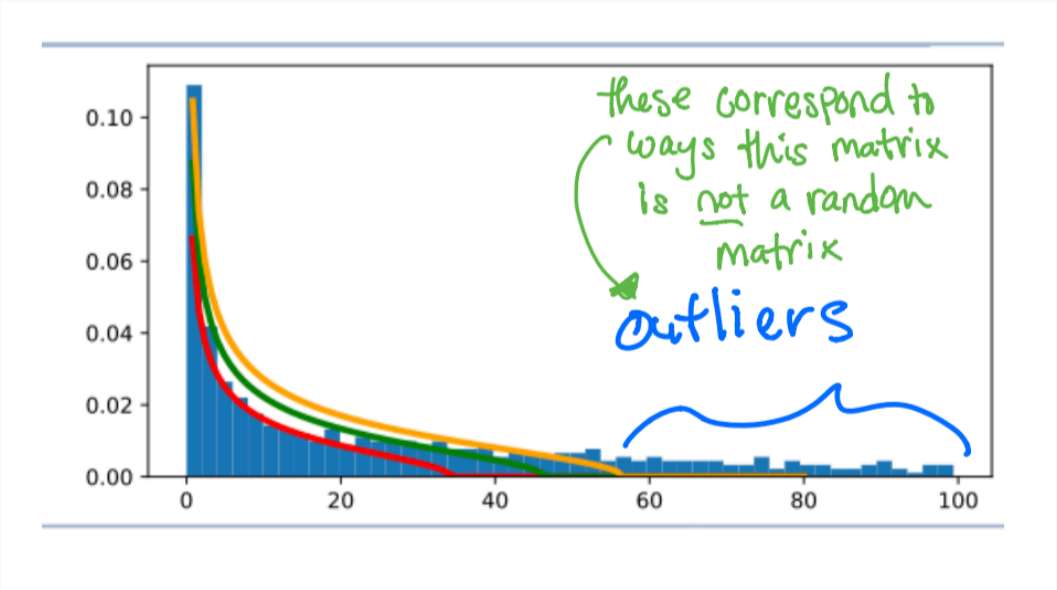
✓ solve numerically ✓ many dist. satisfy fixed point eqn.

Stieltjes transform of Marchenko-Pastur

$$m_{\mathrm{MP}}(z) = \frac{1 - r - z + \sqrt{((1+\sqrt{r})^2 - z)((1-\sqrt{r})^2 - z)}}{2rz}$$

Bulk + outliers

How do we model this?



	Bulk	+	Outliers
	↓		↓
Model:	Marchenko-Pastur	+	Spikes

Sample covariance matrices

Set-up

- Covariance matrix, C, (symmetric, positive semi-definite matrix)
- Noise matrix, $Z \in \mathbb{R}^{d \times n}$ (mean 0, variance 1, i.i.d.)
- $n^{1/\delta} \leq d \leq n^{\delta}$ for some $\delta > 0$
- $X = C^{1/2}Z$; Form $\dfrac{XX^T}{n}$
- $\|C\|_2 \leq$ constant, independent of n

Sample covariance matrices

Set-up

- Covariance matrix, C, (symmetric, positive semi-definite matrix)
- Noise matrix, $Z \in \mathbb{R}^{d \times n}$ (mean 0, variance 1, i.i.d.)
- $n^{1/\delta} \leq d \leq n^{\delta}$ for some $\delta > 0$
- $X = C^{1/2}Z$; Form $\dfrac{XX^T}{n}$
- $\|C\|_2 \leq$ constant, independent of n

Theorem (Bai, Krishnaiah, Silverstein, Yin, '80s-'90s)

$$\text{Stieltjes of } \frac{XX^T}{n} = \operatorname{tr}\left[\left(I_d - z\frac{XX^T}{n}\right)^{-1}\right] \approx \frac{n}{d}\widetilde{m}(z) + \frac{1-\frac{d}{n}}{\frac{d}{n}z}$$

$$\text{where} \qquad \widetilde{m}(z) = \left(-z + \frac{1}{n}\operatorname{tr}[C(I_d - \widetilde{m}(z)C)^{-1}]\right)^{-1}$$

✓ $\widetilde{m}(z) \approx$ Stieltjes transform $\frac{X^TX}{n}$ ✓ implicit eqn, solved numerically

Examples of Sample Covariances

See Colab for details

R-Transform

Example Hessian of 2-layer Network Model

Setup

- $W^{(1)} \in \mathbb{R}^{n_1 \times n_0}$ and $W^{(2)} \in \mathbb{R}^{n_2 \times n_1}$ weight matrices, i.i.d. $N(0,1)$
- $x \in \mathbb{R}^{n_0 \times m}$ is input data and $y \in \mathbb{R}^{n_2 \times m}$ targets
- $g : \mathbb{R} \to \mathbb{R}$ activation function
- $n = n_0 = n_1 = n_2$ and $\phi = \frac{2n}{m}$

outputs: $\widehat{y} = W^{(2)} g(W^{(1)} x)$ residuals: $e_{ij} = \widehat{y}_{ij} - y_{ij}$

Goal: $$\min_{\theta = [W^{(1)}, W^{(2)}]} \left\{ f(\theta) = \frac{1}{2m} \| W^{(2)} g(W^{(1)} x) - y \|^2 \right\}$$

Example Hessian of 2-layer Network Model

Setup

- $W^{(1)} \in \mathbb{R}^{n_1 \times n_0}$ and $W^{(2)} \in \mathbb{R}^{n_2 \times n_1}$ weight matrices, i.i.d. $N(0,1)$
- $x \in \mathbb{R}^{n_0 \times m}$ is input data and $y \in \mathbb{R}^{n_2 \times m}$ targets
- $g : \mathbb{R} \to \mathbb{R}$ activation function
- $n = n_0 = n_1 = n_2$ and $\phi = \frac{2n}{m}$

$$\text{outputs:} \quad \widehat{y} = W^{(2)} g(W^{(1)} x) \qquad \text{residuals:} \quad e_{ij} = \widehat{y}_{ij} - y_{ij}$$

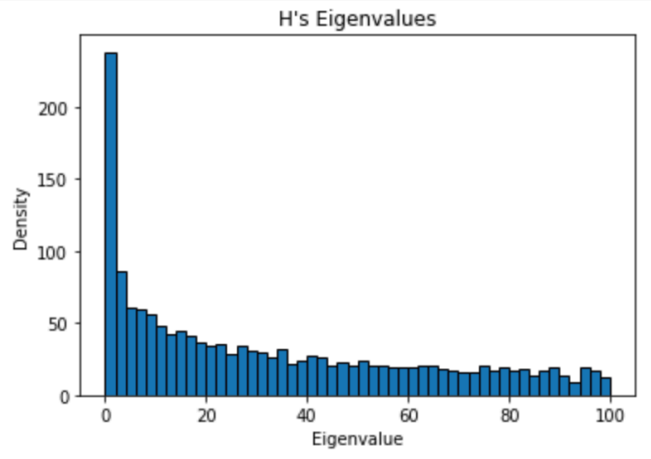
$$\text{Goal:} \qquad \min_{\theta = [W^{(1)}, W^{(2)}]} \left\{ f(\theta) = \frac{1}{2m} \| W^{(2)} g(W^{(1)} x) - y \|^2 \right\}$$

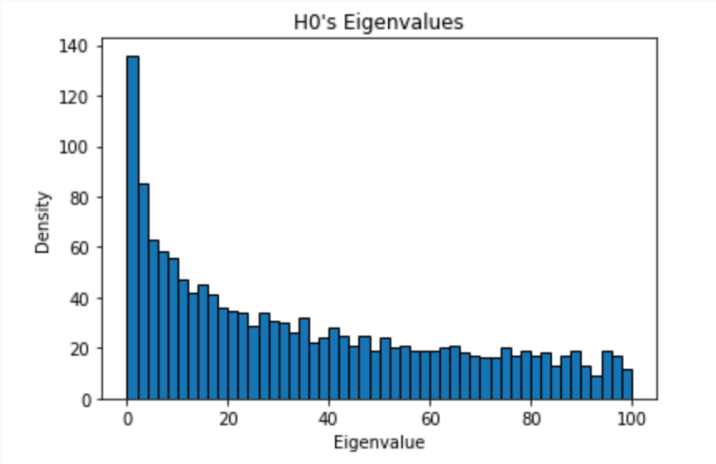
Hessian: $H = H_0 + H_1$

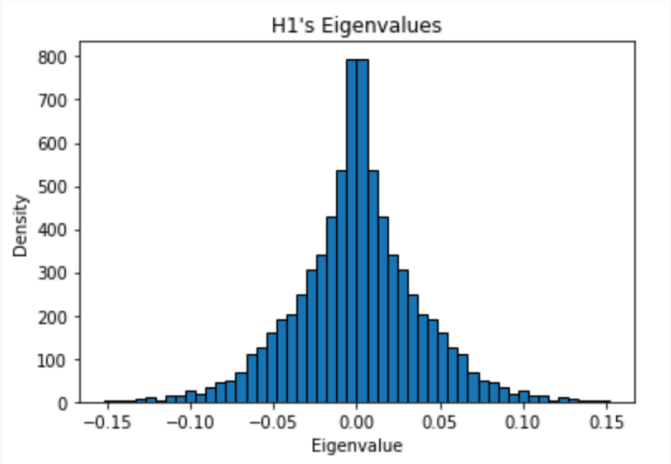
$$[H_0]_{\alpha\beta} = \frac{1}{m} \sum_{i,j=1}^{n_2,m} \frac{\partial \hat{y}_{ij}}{\partial \theta_\alpha} \frac{\partial \hat{y}_{ij}}{\partial \theta_\beta} \quad \text{and} \quad [H_1]_{\alpha\beta} = \frac{1}{m} \sum_{i,j}^{n_2,m} e_{ij} \frac{\partial^2 \hat{y}_{ij}}{\partial \theta_\alpha \partial \theta_\beta}$$

Snapshots of the Hessian during Training

Hessian: $H = H_0 + H_1$







Question: How do you model eigenvalues of H from H_0 and H_1?

Images by McGill undergraduate, Ria Stevens

R-Transform

R-Transform

Tool for writing simple formulas for densities from known densities

Definitions

Free convolution of measures, $\mu_A \boxplus \mu_B$

If A, B two random matrices with ESD, μ_A and μ_B,

$$\text{ESD of } A + B = \mu_A \boxplus \mu_B$$

provided matrix sizes large and matrices asymptotically free.

(side note: product of matrices version)

Asymptotically free matrices

Orthogonal invariance

An A random symmetric matrix is orthogonally invariance if for any fixed orthogonal matrix O

$$O^T A O \overset{\text{law}}{=} A$$

Examples

- Multiples of the Identity
- Wishart with Gaussian entries
- GOE with Gaussian entries

Asymptotically free matrices

Sufficient condition

Suppose $\{A_n\}$ and $\{B_n\}$ are $n \times n$ random matrices. If the following

- A_n and B_n are independent
- A_n is orthogonally invariant

Then A_n and B_n are asymptotically free and $\mu_{A_n} \boxplus \mu_{B_n} \cong \mu_{A_n+B_n}$

Intuition: Eigenvectors of A_n are completely unaligned from B_n

Stieltjes and R-transform

R-transform

R-transform is inverse of Stieltjes of m

$$R(-m(z)) - \frac{1}{m(z)} = z.$$

Examples

- $\beta \boldsymbol{u}\boldsymbol{u}^T$ is $R_{\beta \boldsymbol{u}\boldsymbol{u}^T} = \frac{\beta}{n(1-s\beta)}$
- $R_{\text{Marchenko-Pastur(r)}}(s) = \frac{1}{1-sr}$
- $R_{\text{semicircle}}(s) = s$

Calculus for spectral densities

Theorem

If A and B are asymptotically free,

$$R_{\mu_{A+B}} = R_{\mu_A \boxplus \mu_B} = R_{\mu_A}(s) + R_{\mu_B}(s).$$

Remark

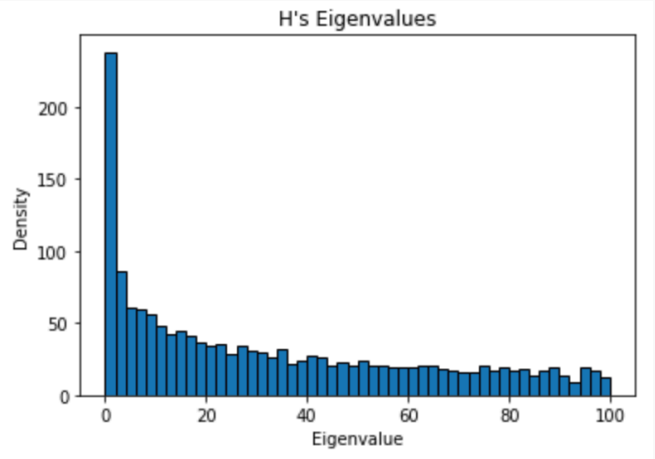
R-transform $\Leftrightarrow$ Stieltjes transform $\Leftrightarrow$ Spectral density

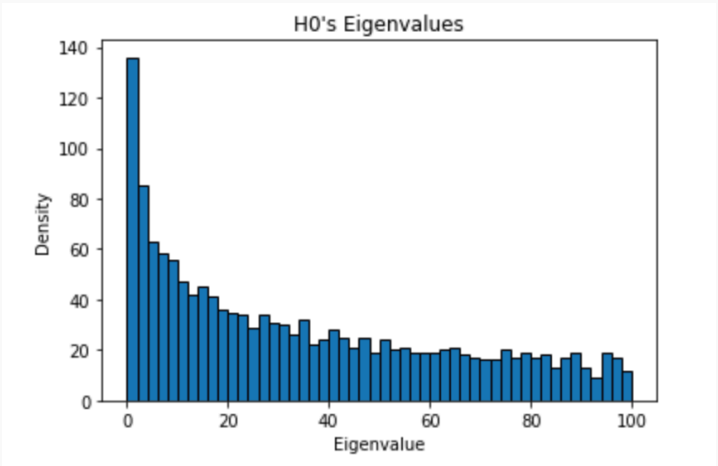
Example

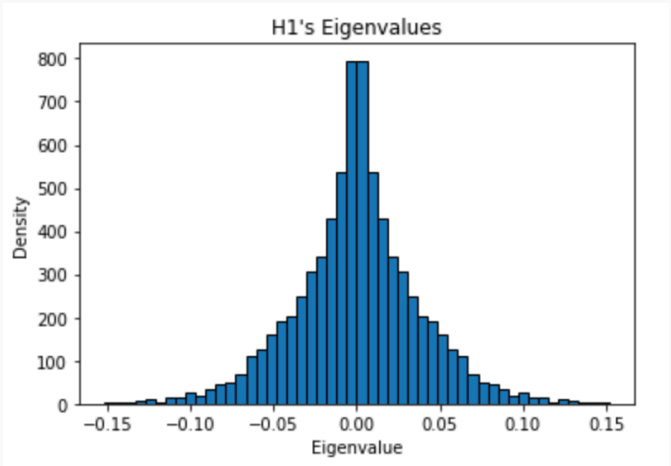
$$R_{\text{GOE + Wishart}} = R_{\text{semicircle}} + R_{\text{MP}} = \underbrace{s}_{\text{GOE}} + \underbrace{\frac{1}{1 - sr}}_{\text{Wishart}}$$

Snapshots of the Hessian during Training

Hessian: $H = H_0 + H_1$







Question: How do you model eigenvalues of H from H_0 and H_1?

Images by McGill undergraduate, Ria Stevens

Model for the Hessian

$$n_2\varepsilon \overset{\text{def}}{=} f(\boldsymbol{\theta}) \quad \text{Hessian:} \quad \boldsymbol{H} = \boldsymbol{H}_0 + \boldsymbol{H}_1$$

Modeling Assumptions

- $\boldsymbol{H}_0$ is Marchenko-Pastur
- $\boldsymbol{H}_0$ and $\boldsymbol{H}_1$ are asymptotically free
- $n = n_0 = n_1 = n_2$ and $\phi = \frac{2n}{m}$

R-transform of $\boldsymbol{H}_1$: $$R_{\boldsymbol{H}_1}(s) = \frac{\varepsilon\phi s}{2 - \varepsilon\phi^2 s^2}$$

R-transform of $\boldsymbol{H}$

$$R_{\boldsymbol{H}}(s) = \frac{1}{1 - s\phi} + \frac{\varepsilon\phi s}{2 - \varepsilon\phi^2 s^2}$$

Questions?

References i

Z. Bai and J. Silverstein.

Spectral analysis of large dimensional random matrices, volume 20.

Springer, 2010.

Z. D. Bai and Jack W. Silverstein.

CLT for linear spectral statistics of large-dimensional sample covariance matrices.

Ann. Probab., 32(1A):553–605, 2004.

ISSN 0091-1798.

doi: 10.1214/aop/1078415845.

URL *https://10.1214/aop/1078415845*.

References ii

Jinho Baik and Jack W. Silverstein.

Eigenvalues of large sample covariance matrices of spiked population models.

J. Multivariate Anal., 97(6):1382–1408, 2006.

doi: 10.1016/j.jmva.2005.08.003.

URL *https://10.1016/j.jmva.2005.08.003*.

Joël Bun, Jean-Philippe Bouchaud, and Marc Potters.

Cleaning large correlation matrices: tools from random matrix theory.

Phys. Rep., 666:1–109, 2017.

ISSN 0370-1573.

doi: 10.1016/j.physrep.2016.10.005.

URL *https://10.1016/j.physrep.2016.10.005*.

References iii

Zhenyu Liao.

A random matrix framework for large dimensional machine learning and neural networks.

PhD thesis, Université Paris-Saclay, 2019.

Jeffrey Pennington and Yasaman Bahri.

Geometry of neural network loss surfaces via random matrix theory.

In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 2798–2806. PMLR, 2017.

URL *http://proceedings.mlr.press/v70/pennington17a.html*.

Random Matrix Theory for Machine Learning

Part 3: Analysis of numerical algorithms

Fabian Pedregosa[1], Courtney Paquette[1,2], Tom Trogdon[3], Jeffrey Pennington[1]

[1] Google Research, [2] McGill University, [3] University of Washington

https://random-matrix-learning.github.io

Notation

In this section:

- A is typically a rectangular matrix with more rows than columns
- W is a symmetric (square) matrix
- Often $W \propto A^T A$

Motivation: Average-case versus worst-case in high dimensions

Issues with worst-case bounds in high dimensions

In some very specific cases, the high-dimensionality of a given problem provides it with enough degrees of freedom to “conspire against” a given algorithm.

Issues with worst-case bounds in high dimensions

In some very specific cases, the high-dimensionality of a given problem provides it with enough degrees of freedom to "conspire against" a given algorithm.

For, example, consider solving a $n \times n$ linear system $\boldsymbol{W}\boldsymbol{x} = \boldsymbol{b}$ using the conjugate gradient (CG) algorithm where

$$W = \begin{bmatrix} \mathfrak{r} & \sqrt{\mathfrak{r}} & & & & & \\ \sqrt{\mathfrak{r}} & 1+\mathfrak{r} & \sqrt{\mathfrak{r}} & & & & \\ & \sqrt{\mathfrak{r}} & 1+\mathfrak{r} & \sqrt{\mathfrak{r}} & & & \\ & & \sqrt{\mathfrak{r}} & \ddots & \ddots & & \\ & & & \ddots & & & \\ & & & & & \sqrt{\mathfrak{r}} & \\ & & & & \sqrt{\mathfrak{r}} & 1+\mathfrak{r} & \end{bmatrix}, \quad b = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \quad 0 < \mathfrak{r} < 1.$$

Issues with worst-case bounds in high dimensions

The CG algorithm is iterative and produces approximations x_k that satisfy:

$$x_k = \underset{y \in \mathbb{R}^n}{\arg\min} \left\{ (x-y)^T W (x-y)^T : y \in \operatorname{span}\{b, Wb, \ldots, W^{k-1}b\} \right\}.$$

Issues with worst-case bounds in high dimensions

The CG algorithm is iterative and produces approximations x_k that satisfy:

$$x_k = \underset{y \in \mathbb{R}^n}{\arg\min} \left\{ (x-y)^T W (x-y)^T : y \in \operatorname{span}\{b, Wb, \ldots, W^{k-1}b\} \right\}.$$

It can be shown that for the above choice of W, b, and $1 \leq k < n$

$$\|b - Wx_k\|^2 = \left(\frac{1}{\mathfrak{r}}\right)^k \quad \text{but} \quad \|b - Wx_n\| = 0.$$

Issues with worst-case bounds in high dimensions

The CG algorithm is iterative and produces approximations x_k that satisfy:

$$x_k = \underset{y \in \mathbb{R}^n}{\arg\min} \left\{ (x-y)^T W (x-y)^T : y \in \operatorname{span}\{b, Wb, \ldots, W^{k-1}b\} \right\}.$$

It can be shown that for the above choice of W, b, and $1 \leq k < n$

$$\|b - Wx_k\|^2 = \left(\frac{1}{\mathfrak{r}}\right)^k \quad \text{but} \quad \|b - Wx_n\| = 0.$$

The residuals (or norms of the gradient) appear to diverge exponentially before the iteration finally converges!

Issues with worst-case bounds in high dimensions

The CG algorithm is iterative and produces approximations x_k that satisfy:

$$x_k = \underset{y \in \mathbb{R}^n}{\arg\min} \left\{ (x-y)^T W (x-y)^T : y \in \operatorname{span}\{b, Wb, \ldots, W^{k-1}b\} \right\}.$$

It can be shown that for the above choice of W, b, and $1 \leq k < n$

$$\|b - Wx_k\|^2 = \left(\frac{1}{\mathfrak{r}}\right)^k \quad \text{but} \quad \|b - Wx_n\| = 0.$$

The residuals (or norms of the gradient) appear to diverge exponentially before the iteration finally converges!

And as n increases, this becomes worse. And a worst-case bound needs to account for this pathological example.

Instead, we may want to choose W and b to be random and consider

$$\mathbb{E}\|b - Wx_k\|^2.$$

If one chooses W to be distributed according to the Wishart distribution, as $n \to \infty$,

$$\mathbb{E}\|b - Wx_k\|^2 = \mathfrak{r}^k + o(1), \quad \mathfrak{r} = r^{-1} = \lim_{n\to\infty} \frac{n}{d}.$$

Introducing distributions

Instead, we may want to choose W and b to be random and consider

$$\mathbb{E}\|b - Wx_k\|^2.$$

If one chooses W to be distributed according to the Wishart distribution, as $n \to \infty$,

$$\mathbb{E}\|b - Wx_k\|^2 = \mathfrak{r}^k + o(1), \quad \mathfrak{r} = r^{-1} = \lim_{n\to\infty} \frac{n}{d}.$$

But a valid important open problem is: To model optimization in a ML context, what distribution is relevant for W?

This is an open problem. See Liao and Mahoney [2021] for work in this direction.

Main RMT tool: Matrix moments

Main linear algebra tool

Recall Cauchy's integral formula: If f is analytic in a sufficiently large region and $\mathcal{C}$ is smooth, simple, closed curve then

$$f(z) = \frac{1}{2\pi \mathrm{i}} \int_{\mathcal{C}} \frac{f(z')}{z' - z} \, \mathrm{d}z'.$$

Main linear algebra tool

Recall Cauchy's integral formula: If f is analytic in a sufficiently large region and $\mathcal{C}$ is smooth, simple, closed curve then

$$f(z) = \frac{1}{2\pi \mathrm{i}} \int_{\mathcal{C}} \frac{f(z')}{z' - z} \, \mathrm{d}z'.$$

Suppose the eigenvalues of an $n \times n$ matrix W are enclosed by $\mathcal{C}$, then

$$f(W) := U f(\Lambda) U^{-1} = \frac{1}{2\pi \mathrm{i}} \int_{\mathcal{C}} f(z) (z I_n - W)^{-1} \, \mathrm{d}z.$$

Main linear algebra tool

Recall Cauchy's integral formula: If f is analytic in a sufficiently large region and $\mathcal{C}$ is smooth, simple, closed curve then

$$f(z) = \frac{1}{2\pi \mathrm{i}} \int_{\mathcal{C}} \frac{f(z')}{z' - z} \, \mathrm{d}z'.$$

Suppose the eigenvalues of an $n \times n$ matrix W are enclosed by $\mathcal{C}$, then

$$f(W) := U f(\Lambda) U^{-1} = \frac{1}{2\pi \mathrm{i}} \int_{\mathcal{C}} f(z) (z I_n - W)^{-1} \, \mathrm{d}z.$$

In particular,

$$W^k = \frac{1}{2\pi \mathrm{i}} \int_{\mathcal{C}} z^k (z I_n - W)^{-1} \, \mathrm{d}z.$$

Two consequences

$$\frac{1}{n}\operatorname{tr} W^k = \frac{1}{2\pi n \mathrm{i}} \int_{\mathcal{C}} z^k \operatorname{tr}\left(z I_n - W\right)^{-1} \mathrm{d}z$$

Two consequences

$$\frac{1}{n}\operatorname{tr} W^k = \frac{1}{n}\sum_{j=1}^{n} \frac{1}{2\pi \mathrm{i}} \int_C z^k (z-\lambda_j)^{-1}\,\mathrm{d}z$$

Two consequences

$$\frac{1}{n}\operatorname{tr} W^k = \frac{1}{n}\sum_{j=1}^{n} \frac{1}{2\pi \mathrm{i}} \int_{\mathcal{C}} z^k (z-\lambda_j)^{-1}\,\mathrm{d}z = \frac{1}{2\pi \mathrm{i}} \int_{\mathcal{C}} z^k \underbrace{m_{\mathrm{ESD}}(z)}_{\substack{\text{Stieltjes transform of} \\ n^{-1}\sum_j \delta_{\lambda_j}}} \mathrm{d}z$$

Two consequences

$$\frac{1}{n}\operatorname{tr} W^k = \frac{1}{n}\sum_{j=1}^{n} \frac{1}{2\pi \mathrm{i}} \int_{\mathcal{C}} z^k (z-\lambda_j)^{-1}\,\mathrm{d}z = \frac{1}{2\pi \mathrm{i}} \int_{\mathcal{C}} z^k \underbrace{m_{\mathrm{ESD}}(z)}_{\substack{\text{Stieltjes transform of} \\ n^{-1}\sum_j \delta_{\lambda_j}}} \mathrm{d}z$$

$$u^T W^k u =$$

Two consequences

$$\frac{1}{n}\operatorname{tr} W^k = \frac{1}{n}\sum_{j=1}^{n} \frac{1}{2\pi \mathrm{i}} \int_{\mathcal{C}} z^k (z - \lambda_j)^{-1} \,\mathrm{d}z = \frac{1}{2\pi \mathrm{i}} \int_{\mathcal{C}} z^k \underbrace{m_{\mathrm{ESD}}(z)}_{\substack{\text{Stieltjes transform of} \\ n^{-1}\sum_j \delta_{\lambda_j}}} \,\mathrm{d}z$$

$$u^T W^k u = \frac{1}{2\pi \mathrm{i}} \int_{\mathcal{C}} z^k u^T (z I_n - W)^{-1} u \,\mathrm{d}z$$

Two consequences

$$\frac{1}{n}\operatorname{tr} W^k = \frac{1}{n}\sum_{j=1}^{n} \frac{1}{2\pi \mathrm{i}} \int_{\mathcal{C}} z^k (z-\lambda_j)^{-1}\,\mathrm{d}z = \frac{1}{2\pi \mathrm{i}} \int_{\mathcal{C}} z^k \underbrace{m_{\mathrm{ESD}}(z)}_{\substack{\text{Stieltjes transform of} \\ n^{-1}\sum_j \delta_{\lambda_j}}} \mathrm{d}z$$

$$u^T W^k u = \frac{1}{2\pi \mathrm{i}} \int_{\mathcal{C}} z^k u^T (zI_n - W)^{-1} u\,\mathrm{d}z = \frac{1}{2\pi \mathrm{i}} \int_{\mathcal{C}} z^k \underbrace{m_{\mathrm{ESD}}(z)}_{\substack{\text{Stieltjes transform of} \\ \sum_j w_j \delta_{\lambda_j}}} \mathrm{d}z$$

$$\sum_j w_j \delta_{\lambda_j}, \quad w_j = (v_j^T u)^2$$

Analysis of matrix moments

A main RMT takeaway:

Matrix moments $\Leftrightarrow$ Classical moments of ESD $\Leftrightarrow$ Contour integrals of Stieltjes transform

Analysis of matrix moments

A main RMT takeaway:

$$\tfrac{1}{n}\operatorname{tr} W^k \approx \int_{\mathbb{R}} x^k \mu_{\mathrm{ESD}}(\mathrm{d}x) \approx \frac{1}{2\pi \mathrm{i}} \int_{\mathcal{C}} z^k m_{\mathrm{ESD}}(z)\,\mathrm{d}z$$

A main RMT takeaway:

$$\tfrac{1}{n}\operatorname{tr} W^k = \int_{\mathbb{R}} x^k \mu_{\mathrm{ESD}}(\mathrm{d}x) \approx \frac{1}{2\pi \mathrm{i}} \int_{\mathcal{C}} z^k m_{\mathrm{ESD}}(z)\,\mathrm{d}z$$

If $\mu_{\mathrm{ESD}} = \frac{1}{n}\sum_{j=1}^{n} \delta_{\lambda_j(W)}$ then the first $\approx$ becomes $=$.

A main RMT takeaway:

$$\tfrac{1}{n}\operatorname{tr} W^k \approx \int_{\mathbb{R}} x^k \mu_{\mathrm{ESD}}(\mathrm{d}x) = \frac{1}{2\pi \mathrm{i}} \int_{\mathcal{C}} z^k m_{\mathrm{ESD}}(z)\,\mathrm{d}z$$

If μ_{ESD} is the limiting ESD then the second $\approx$ becomes $=$.

A main RMT takeaway:

$$\tfrac{1}{n}\operatorname{tr} W^k \approx \int_{\mathbb{R}} x^k \mu_{\mathrm{ESD}}(\mathrm{d}x) = \frac{1}{2\pi \mathrm{i}} \int_{\mathcal{C}} z^k m_{\mathrm{ESD}}(z)\,\mathrm{d}z$$

If μ_{ESD} is the limiting ESD then errors are typically on the order of $1/n$.

Analysis of matrix moments

A main RMT takeaway:

$$u^T W^k u \approx \int_{\mathbb{R}} x^k \mu_{\mathrm{ESD}}(\mathrm{d}x) \approx \frac{1}{2\pi \mathrm{i}} \int_{\mathcal{C}} z^k m_{\mathrm{ESD}}(z)\, \mathrm{d}z$$

A main RMT takeaway:

$$\boldsymbol{u}^T \boldsymbol{W}^k \boldsymbol{u} = \int_{\mathbb{R}} x^k \mu_{\mathrm{ESD}}(\mathrm{d}x) \approx \frac{1}{2\pi \mathrm{i}} \int_{\mathcal{C}} z^k m_{\mathrm{ESD}}(z)\, \mathrm{d}z$$

If $\mu_{\mathrm{ESD}} = \sum_{j=1}^{n} w_j \delta_{\lambda_j(\boldsymbol{W})}$, $w_j = (\boldsymbol{v}_j^T \boldsymbol{u})^2$ for eigenvectors $\boldsymbol{v}_j$ then the first $\approx$ becomes $=$.

A main RMT takeaway:

$$u^T W^k u \approx \int_{\mathbb{R}} x^k \mu_{\mathrm{ESD}}(\mathrm{d}x) = \frac{1}{2\pi \mathrm{i}} \int_{\mathcal{C}} z^k m_{\mathrm{ESD}}(z)\, \mathrm{d}z$$

If μ_{ESD} is the limiting ESD then errors are typically on the order of $1/\sqrt{n}$.

A main RMT takeaway:

$$u^T W^k u \approx \int_{\mathbb{R}} x^k \mu_{\mathrm{ESD}}(\mathrm{d}x) \approx \frac{1}{2\pi \mathrm{i}} \int_{\mathcal{C}} z^k m_{\mathrm{ESD}}(z)\, \mathrm{d}z$$

If a statistic, (which might be the error encountered in, or the runtime of, an algorithm) depends strongly on these generalized moments, it may be analyzable directly using RMT.

A main RMT takeaway:

$$u^T W^k u = \int_{\mathbb{R}} x^k \mu_{\mathrm{ESD}}(\mathrm{d}x) \approx \frac{1}{2\pi \mathrm{i}} \int_{\mathcal{C}} z^k m_{\mathrm{ESD}}(z)\,\mathrm{d}z \quad \mu_{\mathrm{ESD}} = \sum_{j=1}^{d} w_j \delta_{\lambda_j(W)}$$

Theorem (Knowles and Yin [2017])

For a large class of sample covariance matrices W there exists a deterministic measure μ_{SCM} with Stieltjes transform m_{SCM} such that

$$\Pr\left(\left|a^T(W - zI_n)^{-1}b - (a^T b)m_{\mathrm{SCM}}(z)\right| \geq \|a\|\|b\|t\right) = O(n^{-D})$$

for any $D > 0$, uniformly in a large subset of the complex plane.

A main RMT takeaway:

$$u^T W^k u = \int_{\mathbb{R}} x^k \mu_{\mathrm{ESD}}(\mathrm{d}x) \approx \frac{1}{2\pi \mathrm{i}} \int_{\mathcal{C}} z^k m_{\mathrm{ESD}}(z)\,\mathrm{d}z \quad \mu_{\mathrm{ESD}} = \sum_{j=1}^{d} w_j \delta_{\lambda_j(W)}$$

Theorem (Knowles and Yin [2017])

For a large class of sample covariance matrices W there exists a deterministic measure μ_{SCM} with Stieltjes transform m_{SCM} such that

$$\Pr\left(|m_{\mathrm{ESD}}(z) - m_{\mathrm{SCM}}(z)| \geq t\right) = O(n^{-D})$$

for any $D > 0$, uniformly in a large subset of the complex plane.

Resolvent estimates lead to moment estimates

$$u^T W^k u \approx \frac{1}{2\pi \mathrm{i}} \int_{\mathcal{C}} z^k m_{\mathrm{ESD}}(z)\,\mathrm{d}z \approx \frac{1}{2\pi \mathrm{i}} \int_{\mathcal{C}} z^k m_{\mathrm{SCM}}(z)\,\mathrm{d}z = \int_{\mathbb{R}} x^k \mu_{\mathrm{SCM}}(\mathrm{d}x)$$

Resolvent estimates lead to moment estimates

$$u^T W^k u \approx \frac{1}{2\pi \mathrm{i}} \int_{\mathcal{C}} z^k m_{\mathrm{ESD}}(z)\,\mathrm{d}z \approx \frac{1}{2\pi \mathrm{i}} \int_{\mathcal{C}} z^k m_{\mathrm{SCM}}(z)\,\mathrm{d}z = \int_{\mathbb{R}} x^k \mu_{\mathrm{SCM}}(\mathrm{d}x)$$

$$u^T P(W) u \approx \int_{\mathbb{R}} P(x) \mu_{\mathrm{SCM}}(\mathrm{d}x)$$

Algorithm halting times (runtimes)

Statistics of algorithm runtimes

Our abstract setup to analyze algorithms is as follows. Suppose first that there is a intrinsic notion of dimension n.

Statistics of algorithm runtimes

Our abstract setup to analyze algorithms is as follows. Suppose first that there is a intrinsic notion of dimension n.

- Let $\mathcal{A}$ be an iterative algorithm to solve a problem P_n (e.g., an $n \times n$ linear system with gradient descent).

Statistics of algorithm runtimes

Our abstract setup to analyze algorithms is as follows. Suppose first that there is a intrinsic notion of dimension n.

- Let $\mathcal{A}$ be an iterative algorithm to solve a problem P_n (e.g., an $n \times n$ linear system with gradient descent).
- Included with the algorithm, we assume there is a measure of error $E_k(P_n; \mathcal{A})$ at iteration k (e.g., norm of the gradient).

Statistics of algorithm runtimes

Our abstract setup to analyze algorithms is as follows. Suppose first that there is a intrinsic notion of dimension n.

- Let $\mathcal{A}$ be an iterative algorithm to solve a problem P_n (e.g., an $n \times n$ linear system with gradient descent).
- Included with the algorithm, we assume there is a measure of error $E_k(P_n; \mathcal{A})$ at iteration k (e.g., norm of the gradient).
- Let $\mathcal{E}$ represent a distribution from which problems P_n are drawn (e.g., a random matrix and vector for a linear system).

Our abstract setup to analyze algorithms is as follows. Suppose first that there is a intrinsic notion of dimension n.

- Let $\mathcal{A}$ be an iterative algorithm to solve a problem P_n (e.g., an $n \times n$ linear system with gradient descent).
- Included with the algorithm, we assume there is a measure of error $E_k(P_n; \mathcal{A})$ at iteration k (e.g., norm of the gradient).
- Let $\mathcal{E}$ represent a distribution from which problems P_n are drawn (e.g., a random matrix and vector for a linear system).
- The halting time is then defined as

$$T_{\mathcal{A}}(P_n, \varepsilon) = \min\{k : E_k(P_n; \mathcal{A}) < \varepsilon\}.$$

Some history

Probably the most famous, and maybe the most influential, instance of the probabilistic analysis of an algorithm, was the analysis of the simplex algorithm developed by Dantzig [1951] for linear programming (see also Dantzig [1990]).

Some history

Probably the most famous, and maybe the most influential, instance of the probabilistic analysis of an algorithm, was the analysis of the simplex algorithm developed by Dantzig [1951] for linear programming (see also Dantzig [1990]).

For many years after its inception the simplex method had no provable complexity guarantees. Indeed, with a fixed pivot rule, there typically exists a problem on which the simplex method takes an exponentially large number of steps.

Some history

Probably the most famous, and maybe the most influential, instance of the probabilistic analysis of an algorithm, was the analysis of the simplex algorithm developed by Dantzig [1951] for linear programming (see also Dantzig [1990]).

For many years after its inception the simplex method had no provable complexity guarantees. Indeed, with a fixed pivot rule, there typically exists a problem on which the simplex method takes an exponentially large number of steps.

Despite the existence of other algorithms for linear programming with provable polynomial runtime guarantees, the simplex method persisted as the most widely used algorithm.

Some history

Borgwardt [1987] and, independently, Smale [1983] proved that under certain probabilistic assumptions and under certain pivot rules, the expected runtime of the simplex algorithm is polynomial:

$$\mathbb{E}T_{\text{Simplex}}(P_n; \varepsilon) \leq \text{polynomial in } n.$$

Some history

Borgwardt [1987] and, independently, Smale [1983] proved that under certain probabilistic assumptions and under certain pivot rules, the expected runtime of the simplex algorithm is polynomial:

$$\mathbb{E}T_{\text{Simplex}}(P_n; \varepsilon) \leq \text{polynomial in } n.$$

Limited only by their statistical assumptions, these analyses demonstrated, at least conceptually, why the simplex algorithm typically behaves well and is efficient.

Some history

Borgwardt [1987] and, independently, Smale [1983] proved that under certain probabilistic assumptions and under certain pivot rules, the expected runtime of the simplex algorithm is polynomial:

$$\mathbb{E}T_{\text{Simplex}}(P_n; \varepsilon) \leq \text{polynomial in } n.$$

Limited only by their statistical assumptions, these analyses demonstrated, at least conceptually, why the simplex algorithm typically behaves well and is efficient.

The subsequent analysis by Spielman and Teng [2004] improved these analyses by providing estimates for randomly perturbed linear programs. This analysis has since been improved, see [Dadush and Huiberts [2020], Vershynin [2009], Deshpande and Spielman [2005]], for example.

This highlights something we will face here: While we will go through the precise average case analysis of some optimization algorithms, one can always take issue with the underlying statistical assumptions we make.

This highlights something we will face here: While we will go through the precise average case analysis of some optimization algorithms, one can always take issue with the underlying statistical assumptions we make.

For any average-case analysis one hopes to continue to:

- Expand the class of distributions that can be considered.
- Increase the precision of the resulting estimates.
- Collect additional algorithms that can be analyzed with the same or similar techniques.

Some history

We also highlight two other success stories in average-case analysis. These are of a different flavor because randomization is introduced to algorithms to improve their performance. And subsequently, one has a natural distribution over which to compute averages, but the problem being solved is deterministic.

We also highlight two other success stories in average-case analysis. These are of a different flavor because randomization is introduced to algorithms to improve their performance. And subsequently, one has a natural distribution over which to compute averages, but the problem being solved is deterministic.

The first algorithm is the power method with randomized starting. The power method is an algorithm to compute the dominant eigenvalue (provided it exists) of a matrix. It also also approximates the dominant eigenvector.

The power method

1. x_0 is the initial vector, $\|x_0\| = 1$ and W is given.
2. For $k = 1, 2, \ldots$
   - 2.1 Compute $v_k = Wx_{k-1}$
   - 2.2 Compute $\mu_k = v_k^T x_{k-1}$
   - 2.3 Compute $x_k = v_k / \|v_k\|$

The iterates μ_k, under mild assumptions, will converge to the dominant eigenvalue of W. It is well-known that the power method will converge at a exponential rate depending on the ratio of the largest-to-next-largest eigenvalue (a relative spectral gap).

If W is positive semi-definite and x_0 is chosen randomly ($x_0 =$ `np.random.randn(n)`, $x_0 \leftarrow x_0/\|x_0\|$), then it was shown in Kuczyński and Woźniakowski [1992] that a spectral gap is not need to get average-case error bounds of the form:

$$\mathbb{E} \underbrace{\frac{|\mu_k - \lambda_{\max}|}{|\lambda_{\max} - \lambda_{\min}|}}_{E_k(P_n;\mathrm{Power})} \leq 0.871 \frac{\log n}{k-1}.$$

The power method can also be analyzed on random matrices, see Kostlan [1988], Deift and Trogdon [2017].

Lastly, a discussion that is closer to the heart of the matter is the work of Strohmer and Vershynin [2009] on the randomized version of the original Kaczmarz algorithm [Kaczmarz [1937]] for the solution of overdetermined linear systems.

The Kaczmarz Algorithm

1. x_0 is the initial vector and A is given.
2. For $k = 1, 2, \ldots$
   2.1 Select a row a_j of A (add randomness here!)
   2.2 Compute $x_k = x_{k-1} - \frac{b_j - a_j^T x_{k-1}}{\|a_j\|^2} a_j$

For a consistent overdetermined system $Ax = b$ it was shown that the method satisfies

$$\mathbb{E} \underbrace{\|x_k - x\|^2}_{E_k(P_n;\mathrm{Kaczmarz})} \leq \left(1 - \frac{1}{\kappa(A^TA)}\right)^k \|x_0 - x\|^2$$

where $\kappa(A^TA)$ is the condition number of A^TA (to be discussed more later).

The power of random matrix theory (RMT) is that one can ask and answer more involved questions:

- If P_n is drawn randomly, then

$$T_{\mathcal{A}}(P_n, \varepsilon)$$

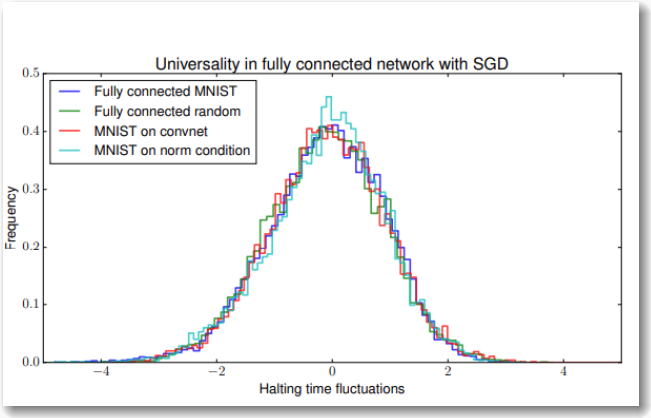
  is an integer-valued random variable. While it is important bound its expectation or moments, what about its distribution as $n \to \infty$?

- With the same considerations

$$E_k(P_n; \mathcal{A})$$

  is a random variable. Can we understand its distribution?

Universality of the halting time



Sagun et al. [2017] present experiments to demonstrate that the halting time $T_{\mathrm{SGD}}(P_n; \varepsilon)$ for a number of neural network architectures exhibits universality. That is, after proper centering and rescaling, the resulting statistics do not depend (in the limit) on the distribution on P_n.

A wide variety of numerical algorithms have been demonstrated (both empirically and rigorously) to have universal halting times (i.e., runtimes, iteration counts, etc.). The study of universality in halting time was initiated by Pfrang et al. [2014] and broaded in Deift et al. [2014].

Universality in halting time is the statement that for a given $\mathcal{A}$, and a wide class of ensembles $\mathcal{E}$, there are constants $\mu = \mu(\mathcal{E}, \varepsilon, n)$ and $\sigma = \sigma(\mathcal{E}, \varepsilon, n)$ and $\varepsilon = \varepsilon(\mathcal{E}, n)$ such that

$$\lim_{n\to\infty} \mathbb{P}_{\mathcal{E}} \left(\frac{T_{\mathcal{A}}(P_n, \varepsilon) - \mu}{\sigma} \leq t \right) = F_{\mathcal{A}}(t).$$

A wide variety of numerical algorithms have been demonstrated (both empirically and rigorously) to have universal halting times (i.e., runtimes, iteration counts, etc.). The study of universality in halting time was initiated by Pfrang et al. [2014] and broaded in Deift et al. [2014].

Universality in halting time is the statement that for a given $\mathcal{A}$, and a wide class of ensembles $\mathcal{E}$, there are constants $\mu = \mu(\mathcal{E}, \varepsilon, n)$ and $\sigma = \sigma(\mathcal{E}, \varepsilon, n)$ and $\varepsilon = \varepsilon(\mathcal{E}, n)$ such that

$$\lim_{n\to\infty} \mathbb{P}_{\mathcal{E}} \left(\frac{T_{\mathcal{A}}(P_n, \varepsilon) - \mu}{\sigma} \leq t \right) = F_{\mathcal{A}}(t).$$

The limiting distribution is independent of the choice for $\mathcal{E}$.

A case study: Regression

A natural first step

A natural first place to combine RMT and optimization/ML with a view toward universality is in the study of linear regression:

$$\underset{x\in\mathbb{R}^n}{\arg\min}\left\{\mathcal{L}(x) := \frac{1}{2d}\|Ax-b\|^2, \quad b = Aa+\eta\right\},$$

where a is the signal, η is additive noise, and

$$\boxed{A \text{ is a } d\times n \text{ matrix}}.$$

A natural first step

A natural first place to combine RMT and optimization/ML with a view toward universality is in the study of linear regression:

$$\arg\min_{x\in\mathbb{R}^n}\left\{\mathcal{L}(x) := \frac{1}{2d}\|Ax-b\|^2, \quad b = Aa+\eta\right\},$$

where a is the signal, η is additive noise, and

$$\boxed{A \text{ is a } d\times n \text{ matrix}}.$$

There are, of course, many iterative algorithms to solve this problem and we focus on two:

1. the conjugate gradient algorithm (CG) [Hestenes and Steifel [1952]], and
2. the gradient descent algorithm (GD).

The Conjugate Gradient Algorithm

1. x_0 is the initial guess.
2. Set $r_0 = A^T b - A^T A x_0$, $p_0 = r_0$.
3. For $k = 1, 2, \ldots, n$
   - 3.1 Compute $a_{k-1} = \dfrac{r_{k-1}^* r_{k-1}}{r_{k-1}^* A^T A p_{k-1}}$.
   - 3.2 Set $x_k = x_{k-1} + a_{k-1} p_{k-1}$.
   - 3.3 Set $r_k = r_{k-1} - a_{k-1} A^T A p_{k-1}$.
   - 3.4 Compute $b_{k-1} = -\dfrac{r_k^* r_k}{r_{k-1}^* r_{k-1}}$.
   - 3.5 Set $p_k = r_k - b_{k-1} p_{k-1}$.

A natural first step

Why consider CG?

A natural first step

Why consider CG?

CG is a highly-structured algorithm with connections to the Lanczos iteration and the theory of orthogonal polynomials. While we do not discuss many of these details, they play an important role in the analysis. CG is also a method-of-choice in the broader computational mathematics community.

A natural first step

Why consider CG?

CG is a highly-structured algorithm with connections to the Lanczos iteration and the theory of orthogonal polynomials. While we do not discuss many of these details, they play an important role in the analysis. CG is also a method-of-choice in the broader computational mathematics community.

A simplification.

Why consider CG?

CG is a highly-structured algorithm with connections to the Lanczos iteration and the theory of orthogonal polynomials. While we do not discuss many of these details, they play an important role in the analysis. CG is also a method-of-choice in the broader computational mathematics community.

A simplification.

Instead of considering CG on the normal equations, $A^T A x = A^T b$, we first consider a slightly simpler problem:

CG applied to $A^T A x = c$.

Scaling regions

Scaling regions show up here in the relationship between n and d in a sample covariance matrix $W = \frac{A^T A}{d}$ (A is $d \times n$).

Scalings of sample covariance matrices

- $d = \lfloor nr \rfloor$ for $r > 1$
- $d = n$
- $d = \lfloor n + cn^{\alpha} \rfloor$ for $0 < \alpha < 1$

Scaling regions

Scaling regions show up here in the relationship between n and d in a sample covariance matrix $W = \frac{A^T A}{d}$ (A is $d \times n$).

Scalings of sample covariance matrices

- $d = \lfloor nr \rfloor$ for $r > 1$
- $d = n$
- $d = \lfloor n + cn^{\alpha} \rfloor$ for $0 < \alpha < 1$

Recall that condition number of a matrix W is defined to be

$$\kappa(W) = \frac{\sigma_1(W)}{\sigma_n(W)},$$

i.e., the ratio of the largest to the smallest singular value of W.

Matrices with small condition numbers are said to be well conditioned while those with larger condition numbers are said to be ill conditioned.

Scaling regions

Scaling regions show up here in the relationship between n and d in a sample covariance matrix $W = \frac{A^T A}{d}$ (A is $d \times n$).

Scalings of sample covariance matrices

- $d = \lfloor nr \rfloor$ for $r > 1$ (well conditioned)
- $d = n$
- $d = \lfloor n + cn^\alpha \rfloor$ for $0 < \alpha < 1$

Recall that condition number of a matrix W is defined to be

$$\kappa(W) = \frac{\sigma_1(W)}{\sigma_n(W)},$$

i.e., the ratio of the largest to the smallest singular value of W.

Matrices with small condition numbers are said to be well conditioned while those with larger condition numbers are said to be ill conditioned.

Scaling regions

Scaling regions show up here in the relationship between n and d in a sample covariance matrix $W = \frac{A^T A}{d}$ (A is $d \times n$).

Scalings of sample covariance matrices

- $d = \lfloor nr \rfloor$ for $r > 1$ (well conditioned)
- $d = n$ (ill conditioned, but still invertible)
- $d = \lfloor n + cn^\alpha \rfloor$ for $0 < \alpha < 1$

Recall that condition number of a matrix W is defined to be

$$\kappa(W) = \frac{\sigma_1(W)}{\sigma_n(W)},$$

i.e., the ratio of the largest to the smallest singular value of W.

Matrices with small condition numbers are said to be well conditioned while those with larger condition numbers are said to be ill conditioned.

Scaling regions

Scaling regions show up here in the relationship between n and d in a sample covariance matrix $W = \frac{A^T A}{d}$ (A is $d \times n$).

Scalings of sample covariance matrices

- $d = \lfloor nr \rfloor$ for $r > 1$ (well conditioned)
- $d = n$ (ill conditioned, but still invertible)
- $d = \lfloor n + cn^\alpha \rfloor$ for $0 < \alpha < 1$ (somewhere in between)

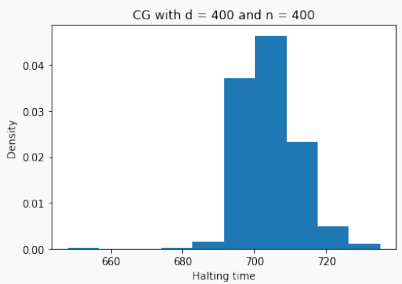
Recall that condition number of a matrix W is defined to be

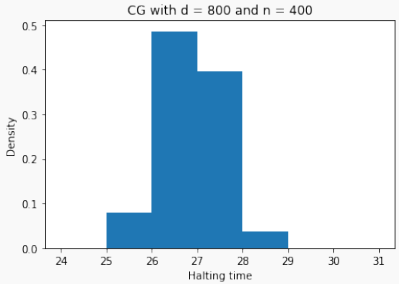
$$\kappa(W) = \frac{\sigma_1(W)}{\sigma_n(W)},$$

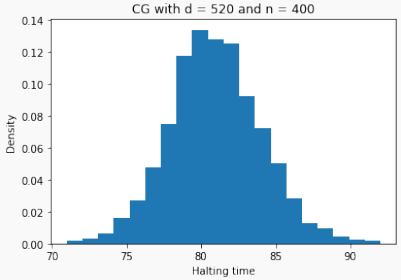
i.e., the ratio of the largest to the smallest singular value of W.

Matrices with small condition numbers are said to be well conditioned while those with larger condition numbers are said to be ill conditioned.

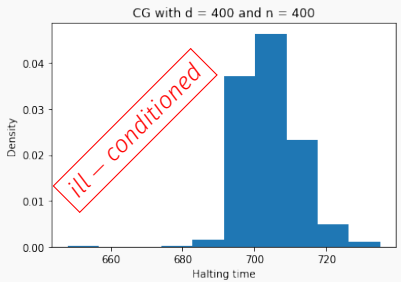
Three distinct behaviors of CG depending on scaling region

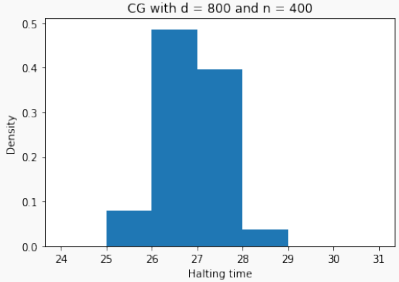


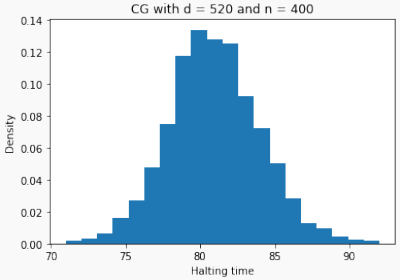




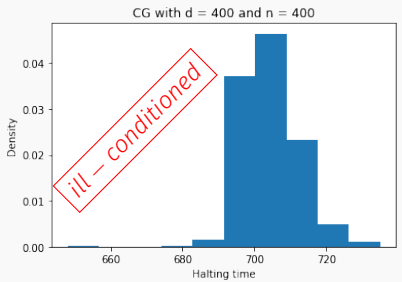
Three distinct behaviors of CG depending on scaling region

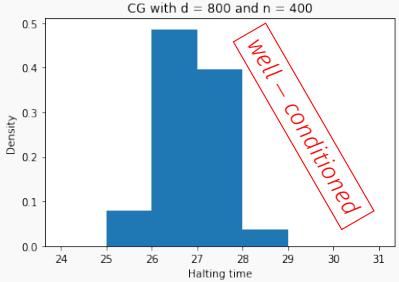


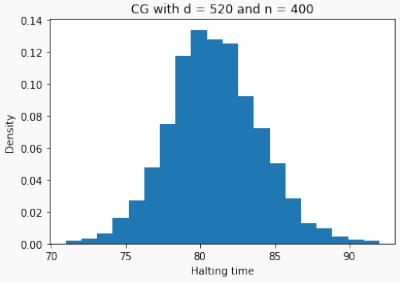




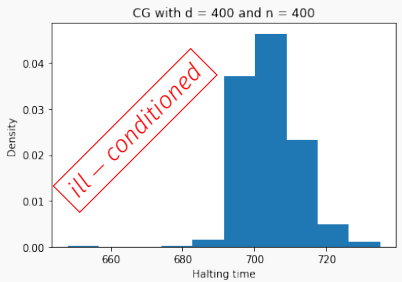
Three distinct behaviors of CG depending on scaling region

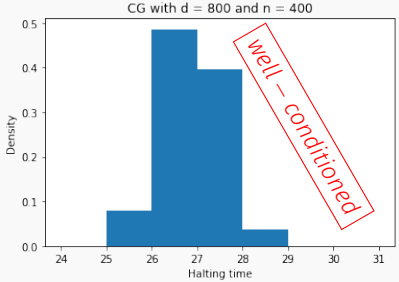


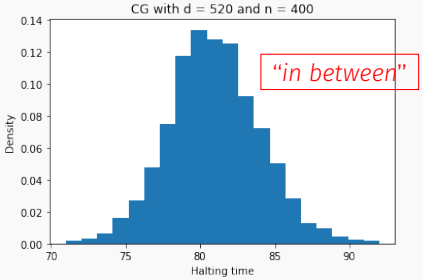




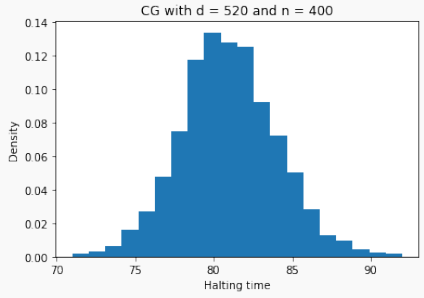
Three distinct behaviors of CG depending on scaling region

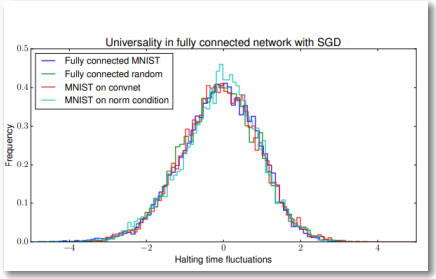






Qualitative comparison with SGD





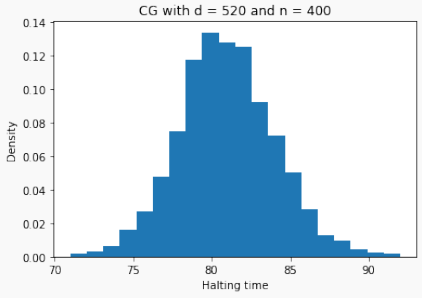
While the mechanisms behind these behaviors are surely different, we see a non-trivial histogram in each setting.

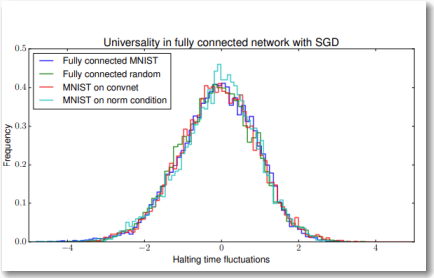
For CG on Wishart matrices, it can be shown that

$$\|r_k\| = \|c - A^T A x_k\| \overset{(\text{dist})}{=} \prod_{j=0}^{k-1} \frac{\chi_{n-j-1}}{\chi_{d-j}},$$

for independent chi-distributed random variables.

Qualitative comparison with SGD





So, if we set

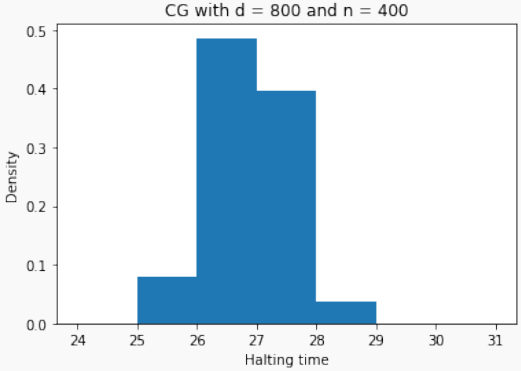
$$E_k(\mathrm{Wishart}; \mathrm{CG}) = \|r_k\|$$

we can analyze the halting time to see that

$$T_{\mathrm{CG}}(\mathrm{Wishart}, \varepsilon) \approx \frac{2}{c} n^{1-\alpha} \log \varepsilon^{-1} + O(n^{3/2-2\alpha}) \mathcal{N}(0,1),$$

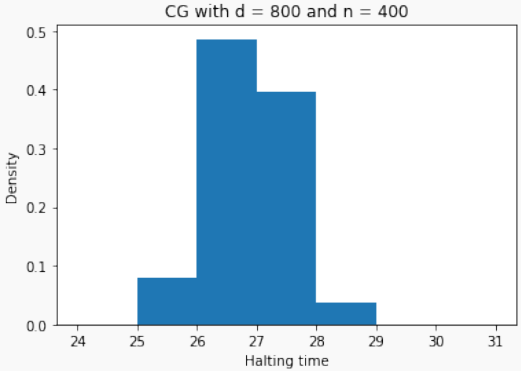
for $1/2 < \alpha < 1$.

To the well-conditioned regime!



It turns out that the errors $E_k(P_N; \mathcal{A})$ for iterative methods for a linear system involving A^TA are often analyzable in the well-conditioned, ill-conditioned and "in between" regimes. But the analysis of the halting time can be much more involved because the halting time $T_{\mathcal{A}}(P_n, \varepsilon)$ can tend to infinity with n!

To the well-conditioned regime!



So, we, for the time being, let $d = \lfloor nr \rfloor$ for $r > 1$.

The Gradient Descent Algorithm

1. x_0 is the initial vector.
2. For $k = 1, 2, \ldots$
   2.1 Select step size γ_k
   2.2 Compute $x_k = x_{k-1} - \gamma_k \nabla \mathcal{L}(x_{k-1})$

The Gradient Descent Algorithm

1. x_0 is the initial vector.
2. For $k = 1, 2, \ldots$
   2.1 Select step size γ_k
   2.2 Compute $x_k = x_{k-1} - \gamma_k \nabla \mathcal{L}(x_{k-1})$

Recall that the gradient of the regression functional is

$$\nabla \mathcal{L}(x) = Wx - c, \quad W = \frac{A^T A}{d}.$$

The Gradient Descent Algorithm

1. x_0 is the initial vector.
2. For $k = 1, 2, \ldots$
   2.1 Select step size γ_k
   2.2 Compute $x_k = x_{k-1} - \gamma_k \nabla \mathcal{L}(x_{k-1})$

Recall that the gradient of the regression functional is

$$\nabla \mathcal{L}(x) = Wx - c, \quad W = \frac{A^T A}{d}.$$

A direction calculation reveals that

$$x_k = Q_k(W)c,$$

for a polynomial Q_k of degree $k-1$ with coefficients that depend on γ_j, $j = 1, 2, \ldots, k$.

For simplicity, suppose that W is full rank. Then if x is the true minimizer, a crucial calculation is that

$$x - x_k = W^{-1}c - Q_k(W)c = W^{-1}\underbrace{\left(I_n - WQ_k(W)\right)}_{R_k(W)}c.$$

Note that R_k is a polynomial of degree k satisfying $R_k(0) = 1$.

Then

$$\nabla \mathcal{L}(x_k) = Wx_k - Wx = R_k(W)c,$$
$$\|R_k(W)c\|^2 = c^T R_k(W)^2 c.$$

Example: GD

For GD follows that the difference $\boldsymbol{x}_k - \boldsymbol{x}$ satisfies

$$\boldsymbol{x}_k - \boldsymbol{x} = \boldsymbol{x}_{k-1} - \boldsymbol{x} - \gamma_k(W\boldsymbol{x}_{k-1} - W\boldsymbol{x}) = (\boldsymbol{I}_n - \gamma_k W)(\boldsymbol{x}_{k-1} - \boldsymbol{x}).$$

And so,

$$R_k(x) = \prod_{j=1}^{k}(1 - \gamma_j x).$$

Example: GD

For GD follows that the difference $\boldsymbol{x}_k - \boldsymbol{x}$ satisfies

$$\boldsymbol{x}_k - \boldsymbol{x} = \boldsymbol{x}_{k-1} - \boldsymbol{x} - \gamma_k(W\boldsymbol{x}_{k-1} - W\boldsymbol{x}) = (\boldsymbol{I}_n - \gamma_k W)(\boldsymbol{x}_{k-1} - \boldsymbol{x}).$$

And so,

$$R_k(x) = \prod_{j=1}^{k}(1 - \gamma_j x).$$

For CG the polynomial R_k is best characterized using the theory of orthogonal polynomials.

$$\|R_k(W)\boldsymbol{c}\|^2 = \boldsymbol{c}^T R_k(W)^2 \boldsymbol{c}$$

The error analysis of GD (and, as it turns out, CG) is reduced to:

1. The determination/characterization of the polynomial R_k.
2. The estimation of $\boldsymbol{c}^T R_k(W)^2 \boldsymbol{c}$.

For many methods of interest (CG and GD included), the coefficients of R_k depend continuously on the eigenvalues and eigenvectors of $\boldsymbol{W}$ in a sufficiently strong sense that

$$R_k(x) \xrightarrow[n\to\infty]{\Pr} \mathcal{R}_k(x) \longleftarrow \text{deterministic}.$$

Then, one can conclude

$$\boldsymbol{c}^T R_k(\boldsymbol{W})^2 \boldsymbol{c} \xrightarrow[n\to\infty]{\Pr} \int_{\mathbb{R}} \mathcal{R}_k(x)^2 \mu_{\mathrm{SCM}}(\mathrm{d}x).$$

This provides a deterministic limit for the (random) errors that are encountered throughout the algorithm.

Note: This is true only if $\boldsymbol{c}$ is independent of $\boldsymbol{W}$ and in the regression problem it is not.

Building back to true regression

For the regression problem, we have

$$c = \frac{1}{n}\left[A^T A a + A^T \eta\right].$$

Then

$$\|\mathcal{L}(x_k)\|^2 = a^T W^2 R_k(W)^2 a^T + \frac{1}{n^2}\eta^T A R_k(W)^2 A^T \eta + \underbrace{\frac{2}{n} a^T W R_k(W)^2 A^T \eta}$$

Building back to true regression

For the regression problem, we have

$$c = \frac{1}{n}\left[A^T A a + A^T \eta\right].$$

Then

$$\|\mathcal{L}(x_k)\|^2 = a^T W^2 R_k(W)^2 a^T + \frac{1}{n^2}\eta^T A R_k(W)^2 A^T \eta + \underbrace{\frac{2}{n} a^T W R_k(W)^2 A^T \eta}_{\approx 0 \quad \text{if} \quad a, \eta \text{ indep.}}$$

Building back to true regression

For the regression problem, we have

$$c = \frac{1}{n}\left[A^T A a + A^T \eta\right].$$

Then

$$\|\mathcal{L}(x_k)\|^2 = a^T W^2 R_k(W)^2 a^T + \frac{1}{n^2}\eta^T A R_k(W)^2 A^T \eta + \underbrace{\frac{2}{n} a^T W R_k(W)^2 A^T \eta}_{\approx 0 \quad \text{if} \quad a,\eta \text{ indep.}}$$

$$\xrightarrow[n\to\infty]{\Pr} \underbrace{R\int_{\mathbb{R}} x^2 \mathcal{R}_k(x)^2 \mu_{\mathrm{SCM}}(\mathrm{d}x) + \tilde{R}\int_{\mathbb{R}} x \mathcal{R}_k(x)^2 \mu_{\mathrm{SCM}}(\mathrm{d}x)}_{\epsilon_k^2}.$$

Building back to true regression

For the regression problem, we have

$$c = \frac{1}{n}\left[A^T A a + A^T \eta\right].$$

Then

$$\|\mathcal{L}(x_k)\|^2 = a^T W^2 R_k(W)^2 a^T + \frac{1}{n^2}\eta^T A R_k(W)^2 A^T \eta + \underbrace{\frac{2}{n} a^T W R_k(W)^2 A^T \eta}_{\approx 0 \text{ if } a,\eta \text{ indep.}}$$

$$\xrightarrow[n\to\infty]{\Pr} \underbrace{R\int_{\mathbb{R}} x^2 \mathcal{R}_k(x)^2 \mu_{\mathrm{SCM}}(\mathrm{d}x) + \tilde{R}\int_{\mathbb{R}} x\mathcal{R}_k(x)^2 \mu_{\mathrm{SCM}}(\mathrm{d}x)}_{\epsilon_k^2}.$$

Important features:

- This demonstrates that the entire spectrum of W contributes via μ_{SCM}
- Nearly all probabilistic analyses of algorithms give inequalities whereas this gives true leading-order behavior.

$$\|\mathcal{L}(x_k)\|^2 \xrightarrow[n\to\infty]{\Pr} R \int_{\mathbb{R}} x^2 \mathcal{R}_k(x)^2 \mu_{\mathrm{SCM}}(\mathrm{d}x) + \tilde{R} \int_{\mathbb{R}} x \mathcal{R}_k(x)^2 \mu_{\mathrm{SCM}}(\mathrm{d}x)$$

If one has a good guess as to what the limiting distribution μ_{SCM} is then the γ_k's in GD can be chosen based on this limit — to minimize this expression, see Pedregosa and Scieur [2020].

$$\|\mathcal{L}(x_k)\|^2 \xrightarrow[n\to\infty]{\Pr} R \int_{\mathbb{R}} x^2 \mathcal{R}_k(x)^2 \mu_{\mathrm{SCM}}(\mathrm{d}x) + \tilde{R} \int_{\mathbb{R}} x \mathcal{R}_k(x)^2 \mu_{\mathrm{SCM}}(\mathrm{d}x)$$

If one has a good guess as to what the limiting distribution μ_{SCM} is then the γ_k's in GD can be chosen based on this limit — to minimize this expression, see Pedregosa and Scieur [2020].

Furthermore, by preconditioning one can make such a guess valid, see Lacotte and Pilanci [2020].

Provided that $\mathfrak{e}_k \xrightarrow{k\to\infty} 0$, one finds that

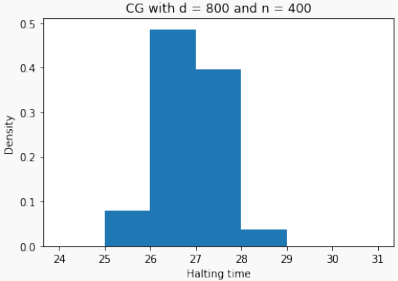
$$\lim_{n\to\infty} \mathbb{P}\left(T_{\mathcal{A}}(P_n;\varepsilon) = \min\{k : \mathfrak{e}_k < \varepsilon\}\right) = 1,$$

for most choices of ε.

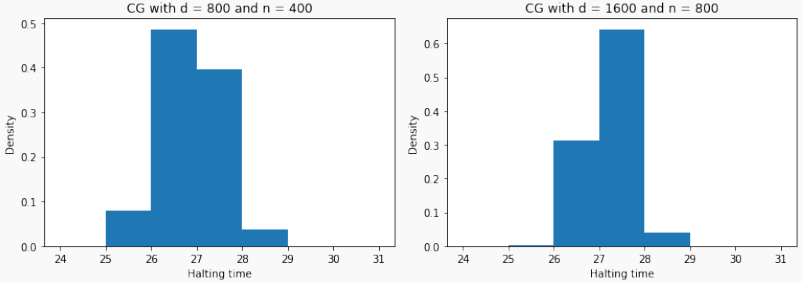
This turns out to be true for all $d \geq n$, $n \to \infty$, for the regression problem with CG or GD.

Deterministic halting for CG with $r = 2$, $\varepsilon = 10^{-4}$

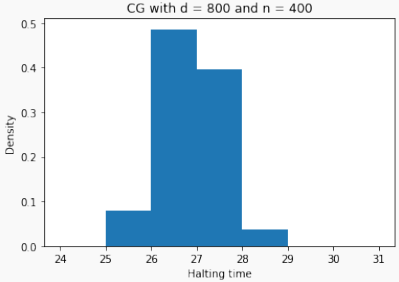
Deterministic halting for CG with $r = 2$, $\varepsilon = 10^{-4}$

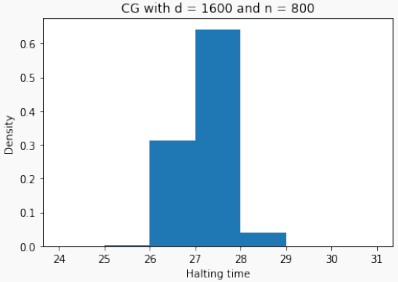


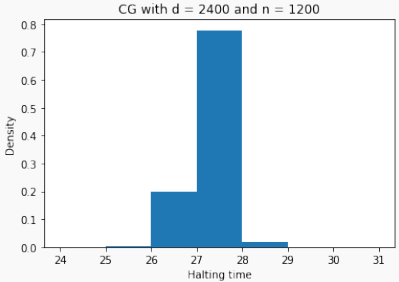
Deterministic halting for CG with $r = 2$, $\varepsilon = 10^{-4}$



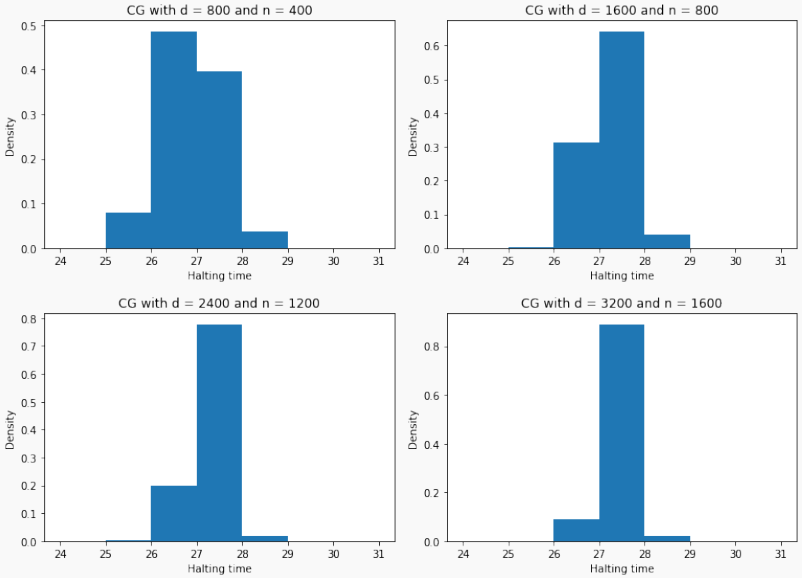
Deterministic halting for CG with $r = 2$, $\varepsilon = 10^{-4}$







Deterministic halting for CG with $r = 2$, $\varepsilon = 10^{-4}$



Outlook

RMT provides non-trivial tractable models to analyze the statistics of optimization algorithms.

Other algorithms are analyzable:

- MINRES algorithm
- Polyak algorithm
- Nesterov accelerated algorithm
- SGD for regression
- ...

See the preprints: Paquette and Trogdon [2020], Paquette et al. [2021], Ding and Trogdon [2021], Paquette et al. [2020]

Outlook

Other ensembles are analyzable using the following results from RMT:

- Spiked random matrices (see Baik et al. [2005], Bloemendal and Virág [2013], Ding and Yang [2019], and many more)
- Nonlinear models (see Part 4)
- Random graphs (see Erdős et al. [2013], for example)
- Invariant ensembles (see Bourgade et al. [2014], Deift [2000] and many more)

Open questions

Many open questions remain:

Open questions

Many open questions remain:

- To what extent can one move these ideas beyond regression? To a two-layer network? Rank-one matrix completion problem?

Open questions

Many open questions remain:

- To what extent can one move these ideas beyond regression? To a two-layer network? Rank-one matrix completion problem?
- What is a good probability distribution to study? Wishart is clearly the place to start but what is relevant in practice?

A CG demo

See Colab for a CG demo

https://colab.research.google.com/drive/1UZRSK665b8sqq0NQFwMCwrVabPlB-7nK?usp=sharing

References

J Baik, G Ben Arous, and S Péché. Phase transition of the largest eigenvalue for nonnull complex sample covariance matrices. *The Annals of Probability*, 33(5), sep 2005. ISSN 0091-1798. doi: 10.1214/009117905000000233. URL *https://projecteuclid.org/journals/annals-of-probability/volume-33/issue-5/Phase-transition-of-the-largest-eigenvalue-for-nonnull-complex-sample/10.1214/009117905000000233.full*.

A Bloemendal and B Virág. Limits of spiked random matrices I. *Probability Theory and Related Fields*, 156(3-4):795–825, aug 2013. ISSN 0178-8051. doi: 10.1007/s00440-012-0443-2. URL *http://link.springer.com/10.1007/s00440-012-0443-2*.

K H Borgwardt. *The simplex method: A probabilistic analysis*. Springer–Verlag, Berlin, Heidelberg, 1987. URL *https://www.springer.com/gp/book/9783540170969*.

P Bourgade, L Erdős, and H-T Yau. Edge Universality of Beta Ensembles. *Communications in Mathematical Physics*, 332(1):261–353, nov 2014. ISSN 0010-3616. doi: 10.1007/s00220-014-2120-z. URL *http://link.springer.com/10.1007/s00220-014-2120-z*.

D Dadush and S Huiberts. A Friendly Smoothed Analysis of the Simplex Method. *SIAM Journal on Computing*, 49(5):STOC18–449–STOC18–499, jan 2020. ISSN 0097-5397. doi: 10.1137/18M1197205. URL *https://epubs.siam.org/doi/10.1137/18M1197205*.

G B Dantzig. Maximization of a linear function of variables subject to linear inequalities. In *Activity Analysis of Production and Allocation*, pages 339–347. 1951. ISBN 0804748349. URL *http://cowles.econ.yale.edu/P/cm/m13/m13-21.pdf*.

G B Dantzig. Origins of the simplex method. In *A history of scientific computing*, pages 141–151. ACM, New York, NY, USA, jun 1990. doi: 10.1145/87252.88081. URL *http://dl.acm.org/doi/10.1145/87252.88081*.

P Deift. *Orthogonal Polynomials and Random Matrices: a Riemann-Hilbert Approach*. Amer. Math. Soc., Providence, RI, 2000.

References ii

P Deift and T Trogdon. Universality for Eigenvalue Algorithms on Sample Covariance Matrices. *SIAM Journal on Numerical Analysis*, 2017. URL http://arxiv.org/abs/1701.01896.

Percy A Deift, Govind Menon, Sheehan Olver, and Thomas Trogdon. Universality in numerical computations with random data. *Proceedings of the National Academy of Sciences*, 2014. URL https://doi.org/10.1073/pnas.1413446111.

A Deshpande and D A Spielman. Improved Smoothed Analysis of the Shadow Vertex Simplex Method. In *46th Annual IEEE Symposium on Foundations of Computer Science (FOCS'05)*, pages 349–356. IEEE, 2005. ISBN 0-7695-2468-0. doi: 10.1109/SFCS.2005.44. URL http://ieeexplore.ieee.org/document/1530727/.

X Ding and T Trogdon. The conjugate gradient algorithm on a general class of spiked covariance matrices. *arXiv preprint arXiv:2106.13902*, jun 2021. URL http://arxiv.org/abs/2106.13902.

X Ding and F Yang. Spiked separable covariance matrices and principal components. *arXiv preprint arXiv:1905.13060*, may 2019. URL http://arxiv.org/abs/1905.13060.

L Erdős, A Knowles, H-T Yau, and J Yin. Spectral statistics of Erdős–Rényi graphs I: Local semicircle law. *The Annals of Probability*, 41(3B), may 2013. ISSN 0091-1798. doi: 10.1214/11-AOP734. URL https://projecteuclid.org/journals/annals-of-probability/volume-41/issue-3B/Spectral-statistics-of-ErdősR{é}nyi-graphs-I-Local-semicircle-law/10.1214/11-AOP734.full.

M Hestenes and E Steifel. Method of Conjugate Gradients for Solving Linear Systems. *J. Research Nat. Bur. Standards*, 20:409–436, 1952.

S Kaczmarz. Angenäherte auflösung von systemen linearer gleichungen (english translation by jason stockmann): Bulletin international de l'académie polonaise des sciences et des lettres. 1937.

A Knowles and J Yin. Anisotropic local laws for random matrices. *Probability Theory and Related Fields*, 169(1-2):257–352, oct 2017. ISSN 0178-8051. doi: 10.1007/s00440-016-0730-4. URL http://link.springer.com/10.1007/s00440-016-0730-4.

E Kostlan. Complexity theory of numerical linear algebra. *Journal of Computational and Applied Mathematics*, 22(2-3):219–230, jun 1988. ISSN 03770427. doi: 10.1016/0377-0427(88)90402-5. URL http://www.sciencedirect.com/science/article/pii/0377042788904025.

References iii

J Kuczyński and H Woźniakowski. Estimating the Largest Eigenvalue by the Power and Lanczos Algorithms with a Random Start. *SIAM Journal on Matrix Analysis and Applications*, 13(4):1094–1122, oct 1992. ISSN 0895-4798. doi: 10.1137/0613066. URL *http://epubs.siam.org/doi/10.1137/0613066*.

Jonathan Lacotte and Mert Pilanci. Optimal randomized first-order methods for least-squares problems. In *International Conference on Machine Learning*. PMLR, 2020. URL *https://arxiv.org/pdf/2002.09488.pdf*.

Z Liao and M W Mahoney. Hessian Eigenspectra of More Realistic Nonlinear Models. mar 2021. URL *http://arxiv.org/abs/2103.01519*.

C Paquette, B van Merriënboer, and F Pedregosa. Halting Time is Predictable for Large Models: A Universality Property and Average-case Analysis. *arXiv preprint arXiv:2006.04299*, jun 2020. URL *http://arxiv.org/abs/2006.04299*.

C Paquette, K Lee, F Pedregosa, and E Paquette. SGD in the Large: Average-case Analysis, Asymptotics, and Stepsize Criticality. *arXiv preprint 2102.04396*, feb 2021. URL *http://arxiv.org/abs/2102.04396*.

Elliot Paquette and Thomas Trogdon. Universality for the conjugate gradient and minres algorithms on sample covariance matrices. *arXiv preprint arXiv:2007.00640*, 2020. URL *https://arxiv.org/pdf/2007.00640.pdf*.

Fabian Pedregosa and Damien Scieur. Acceleration through spectral density estimation. In *International Conference on Machine Learning*. PMLR, 2020. URL *https://arxiv.org/pdf/2002.04756.pdf*.

C W Pfrang, P Deift, and G Menon. How long does it take to compute the eigenvalues of a random symmetric matrix? *Random matrix theory, interacting particle systems, and integrable systems, MSRI Publications*, 65:411–442, 2014. URL *http://arxiv.org/abs/1203.4635*.

Levent Sagun, Thomas Trogdon, and Yann LeCun. Universal halting times in optimization and machine learning. *Quarterly of Applied Mathematics*, 2017. URL *https://doi.org/10.1090/qam/1483*.

S Smale. On the average number of steps of the simplex method of linear programming. *Mathematical Programming*, 27(3):241–262, oct 1983. ISSN 0025-5610. doi: 10.1007/BF02591902. URL *http://link.springer.com/10.1007/BF02591902*.

References iv

D A Spielman and S-H Teng. Smoothed analysis of algorithms. *Journal of the ACM*, 51(3):385–463, may 2004. ISSN 00045411. doi: 10.1145/990308.990310. URL http://dl.acm.org/citation.cfm?id=990308.990310.

T Strohmer and R Vershynin. A Randomized Kaczmarz Algorithm with Exponential Convergence. *Journal of Fourier Analysis and Applications*, 15(2): 262–278, apr 2009. ISSN 1069-5869. doi: 10.1007/s00041-008-9030-4. URL http://link.springer.com/10.1007/s00041-008-9030-4.

R Vershynin. Beyond Hirsch Conjecture: Walks on Random Polytopes and Smoothed Complexity of the Simplex Method. *SIAM Journal on Computing*, 39, 2009. URL http://epubs.siam.org/doi/10.1137/070683386.

Random Matrix Theory for Machine Learning

The Mystery of Generalization: Why Does Deep Learning Work?

Fabian Pedregosa, Courtney Paquette, Tom Trogdon, Jeffrey Pennington

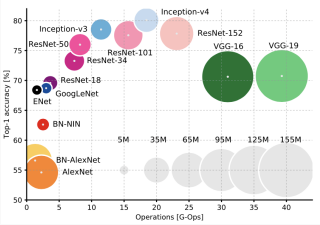
https://random-matrix-learning.github.io

Why does deep learning work?

Deep neural networks define a flexible and expressive class of functions.

State-of-the-art models have millions or billions of parameters

- Meena: 2.6 billion
- Turing NLG: 17 billion
- GPT-3: 175 billion



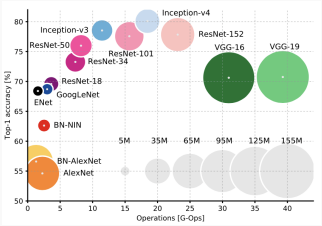
Source: [Canziani et al., 2016]

Why does deep learning work?

Deep neural networks define a flexible and expressive class of functions.

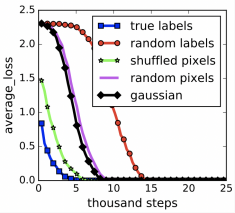
State-of-the-art models have millions or billions of parameters

- Meena: 2.6 billion
- Turing NLG: 17 billion
- GPT-3: 175 billion



Source: [Canziani et al., 2016]

Models that perform well on real data can easily memorize noise

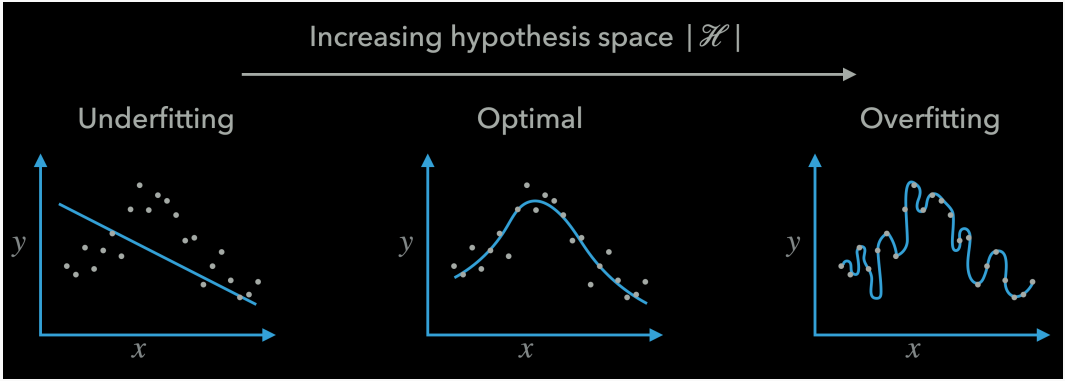


Source: [Zhang et al., 2021]

Why does deep learning work?

Deep neural networks define a flexible and expressive class of functions.

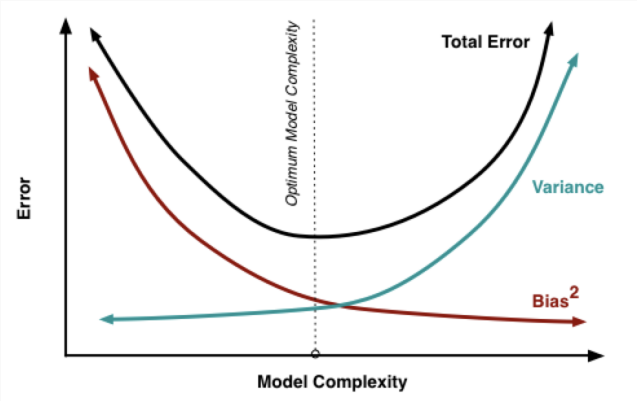
$\Rightarrow$ Standard wisdom suggests they should overfit



Why does deep learning work?

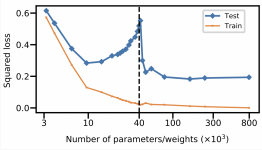
Deep neural networks define a flexible and expressive class of functions.

$\Rightarrow$ Standard wisdom suggests they should overfit

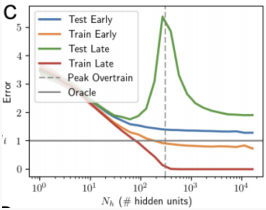


Double descent

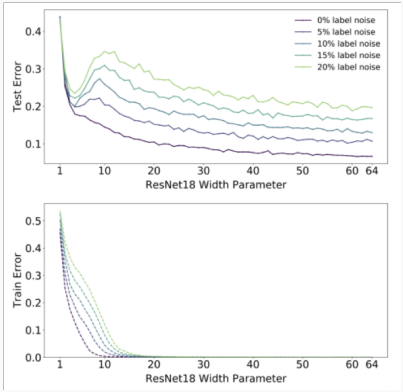
However, large neural networks do not obey the classical theory:



Source: [Belkin et al., 2019]



Source: [Advani et al., 2020]



Source: [Nakkiran et al., 2019]

The emerging paradigm of *double descent* seeks to explain this phenomenon.

Models of Double Descent

History of double descent: Kernel interpolation

1) Interpolating kernels (trained to zero error) generalize well [Belkin et al., 2018]

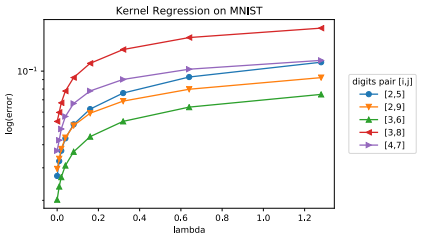
$\Rightarrow$ Double descent is not unique to deep neural networks

History of double descent: Kernel interpolation

1) Interpolating kernels (trained to zero error) generalize well [Belkin et al., 2018]

⇒ Double descent is not unique to deep neural networks

2) Kernels can implicitly regularize in high dimensions [Liang and Rakhlin, 2020]



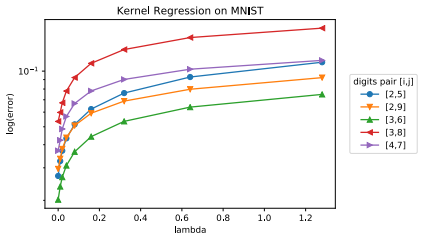
Source: [Liang and Rakhlin, 2020]

History of double descent: Kernel interpolation

1) Interpolating kernels (trained to zero error) generalize well [Belkin et al., 2018]

⇒ Double descent is not unique to deep neural networks

2) Kernels can implicitly regularize in high dimensions [Liang and Rakhlin, 2020]



Source: [Liang and Rakhlin, 2020]

3) Consistency is a high-dimensional phenomenon [Rakhlin and Zhai, 2019]:

The estimation error of the minimum-norm kernel interpolant

$$\arg\min_{f \in \mathcal{H}} \|f\|_{\mathcal{H}} \quad \text{s.t.} \quad f(x_i) = y_i\,, \quad i = 1 \dots n$$

does not converge to zero as n grows, unless d is proportional to n.

Models of double descent: High-dimensional linear regression

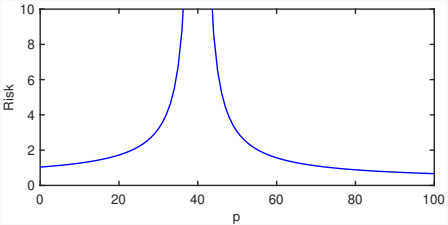
What is the simplest theoretically tractable model that exhibits double descent?

Models of double descent: High-dimensional linear regression

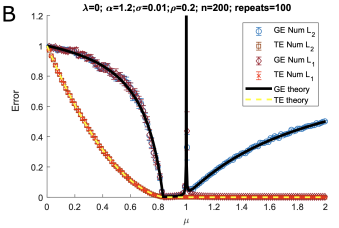
What is the simplest theoretically tractable model that exhibits double descent?

Linear regression suffices, but requires a mechanism to vary the effective number of parameters or samples:

- The size of randomly selected subsets of features [Belkin et al., 2020]
- The dimensionality of the low-variance subspace [Bartlett et al., 2020]
- The sparsity of the generative model [Mitra, 2019]



Source: [Belkin et al., 2020]



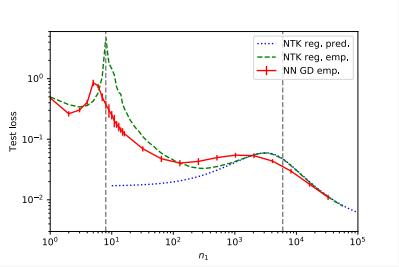
Source: [Mitra, 2019]

Models of double descent: Random feature models

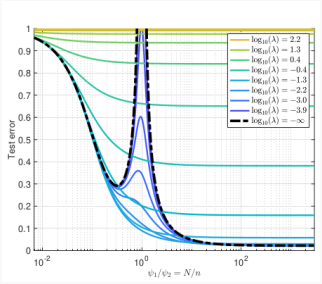
In random feature regression, the number of random features controls the model capacity, and can be tuned independently from the data.

Exact asymptotic results in high dimensions exist in many settings, including:

- Unstructured random features [Mei and Montanari, 2019]
- NTK-structured random features [Adlam and Pennington, 2020]
- Random Fourier features [Liao et al., 2020]



Source: [Adlam and Pennington, 2020]



Source: [Mei and Montanari, 2019]

Random feature models

Random feature regression: definition

Random feature regression is just linear regression on a transformed feature matrix, $F = f(\frac{1}{\sqrt{d}} WX) \in \mathbb{R}^{m \times n}$, where $W \in \mathbb{R}^{m \times d}$, $W_{ij} \sim \mathcal{N}(0, 1)$.

- Model given by $\beta^\top F$ (instead of $\beta^\top X$) — variable capacity (m vs d parameters)
- $f(\cdot)$ is a nonlinear activation function, acting elementwise
- F is equivalent to first post-activation layer of a NN at init

Random feature regression: definition

Random feature regression is just linear regression on a transformed feature matrix, $F = f(\frac{1}{\sqrt{d}} WX) \in \mathbb{R}^{m \times n}$, where $W \in \mathbb{R}^{m \times d}$, $W_{ij} \sim \mathcal{N}(0, 1)$.

- Model given by $\beta^\top F$ (instead of $\beta^\top X$) — variable capacity (m vs d parameters)
- $f(\cdot)$ is a nonlinear activation function, acting elementwise
- F is equivalent to first post-activation layer of a NN at init

For targets $Y \in \mathbb{R}^{1 \times n}$, the ridge-regularized loss is,

$$L(\beta; X) = \|Y - \tfrac{1}{\sqrt{m}} \beta^\top F\|_2^2 + \lambda \|\beta\|_2^2 \,,$$

and the optimal regression coefficients $\hat{\beta}$ are given by,

$$\hat{\beta} = \tfrac{1}{\sqrt{m}} Y (K + \lambda I_n)^{-1} F^\top \,, \quad K = \tfrac{1}{m} F^\top F \,.$$

Random feature regression: definition

Random feature regression is just linear regression on a transformed feature matrix, $F = f(\frac{1}{\sqrt{d}} WX) \in \mathbb{R}^{m \times n}$, where $W \in \mathbb{R}^{m \times d}$, $W_{ij} \sim \mathcal{N}(0, 1)$.

- Model given by $\beta^\top F$ (instead of $\beta^\top X$) — variable capacity (m vs d parameters)
- $f(\cdot)$ is a nonlinear activation function, acting elementwise
- F is equivalent to first post-activation layer of a NN at init

For targets $Y \in \mathbb{R}^{1 \times n}$, the ridge-regularized loss is,

$$L(\beta; X) = \|Y - \tfrac{1}{\sqrt{m}} \beta^\top F\|_2^2 + \lambda \|\beta\|_2^2 \,,$$

and the optimal regression coefficients $\hat{\beta}$ are given by,

$$\hat{\beta} = \tfrac{1}{\sqrt{m}} Y (K + \lambda I_n)^{-1} F^\top \,, \quad K = \tfrac{1}{m} F^\top F \,.$$

Note that $Q \equiv (K + \lambda I_n)^{-1}$ is the *resolvent* of the kernel matrix K.

Random feature regression: training error

Training error is determined by the resolvent matrix $Q \equiv (K + \lambda I_n)^{-1}$:

$$\begin{aligned} E_{\text{train}}(\lambda) &= \tfrac{1}{n} \|Y - \tfrac{1}{\sqrt{m}} \hat{\boldsymbol{\beta}}^\top F\|_2^2 \\ &= \tfrac{1}{n} \|Y - \tfrac{1}{m} YQF^\top F\|_2^2 \\ &= \tfrac{1}{n} \|Y - YQK\|_2^2 \\ &= \tfrac{1}{n} \|Y(I_n - QK)\|_2^2 \\ &= \lambda^2 \tfrac{1}{n} \|YQ\|_2^2 \,, \end{aligned}$$

where we used that $I_n - QK = I_n - Q(Q^{-1} - \lambda I_n) = I_n - (I_n - \lambda Q) = \lambda Q$.

Random feature regression: training error

Training error is determined by the resolvent matrix $Q \equiv (K + \lambda I_n)^{-1}$:

$$\begin{aligned} E_{\text{train}}(\lambda) &= \tfrac{1}{n} \| Y - \tfrac{1}{\sqrt{m}} \hat{\beta}^\top F \|_2^2 \\ &= \tfrac{1}{n} \| Y - \tfrac{1}{m} Y Q F^\top F \|_2^2 \\ &= \tfrac{1}{n} \| Y - Y Q K \|_2^2 \\ &= \tfrac{1}{n} \| Y (I_n - Q K) \|_2^2 \\ &= \lambda^2 \tfrac{1}{n} \| Y Q \|_2^2 \,, \end{aligned}$$

where we used that $I_n - QK = I_n - Q(Q^{-1} - \lambda I_n) = I_n - (I_n - \lambda Q) = \lambda Q$.

So we see that the training error measures the alignment between the resolvent and the label vector.

What about the test error?

Aside: Generalized cross validation (GCV)

A model's performance on the training set, or subsets thereof, can be useful for estimating its performance on the test set.

- Leave-one-out cross validation (LOOCV)

$$E_{LOOCV}(\lambda) = \frac{1}{n}\|\boldsymbol{YQ} \cdot \mathrm{diag}(\boldsymbol{Q})^{-1}\|_2^2$$

- Generalized cross validation (GCV)

$$E_{GCV}(\lambda) = \frac{1}{n}\|\boldsymbol{YQ}\|_2^2 / (\frac{1}{n}\mathrm{tr}(\boldsymbol{Q}))^2$$

Aside: Generalized cross validation (GCV)

A model's performance on the training set, or subsets thereof, can be useful for estimating its performance on the test set.

- Leave-one-out cross validation (LOOCV)

$$E_{LOOCV}(\lambda) = \frac{1}{n}\|\boldsymbol{Y}\boldsymbol{Q} \cdot \mathrm{diag}(\boldsymbol{Q})^{-1}\|_2^2$$

- Generalized cross validation (GCV)

$$E_{GCV}(\lambda) = \frac{1}{n}\|\boldsymbol{Y}\boldsymbol{Q}\|_2^2 / (\frac{1}{n}\mathrm{tr}(\boldsymbol{Q}))^2$$

In certain high-dimensional limits, $E_{GCV}(\lambda) = E_{LOOCV}(\lambda) = E_{\text{test}}(\lambda)$:

- Ridge regression [Hastie et al., 2019]
- Kernel ridge regression [Jacot et al., 2020]
- Random feature regression [Adlam and Pennington, 2020]

Random feature regression: high-dimensional asymptotics

To develop an analytical model of double descent, we study the high-dimensional asymptotics:

$$m, d, n \to \infty \quad \text{such that} \quad \phi \equiv \frac{d}{n}, \psi \equiv \frac{d}{m} \quad \text{are constant.}$$

Random feature regression: high-dimensional asymptotics

To develop an analytical model of double descent, we study the high-dimensional asymptotics:

$$m, d, n \to \infty \quad \text{such that} \quad \phi \equiv \frac{d}{n}, \psi \equiv \frac{d}{m} \quad \text{are constant.}$$

In this limit, only *linear* functions of the data can be learned.

- Intuition: only enough constraints to disambiguate linear combinations of features.
- Nonlinear target function behaves like linear function plus noise

Random feature regression: high-dimensional asymptotics

To develop an analytical model of double descent, we study the high-dimensional asymptotics:

$$m, d, n \to \infty \quad \text{such that} \quad \phi \equiv \frac{d}{n}\,, \psi \equiv \frac{d}{m} \quad \text{are constant.}$$

In this limit, only *linear* functions of the data can be learned.

- Intuition: only enough constraints to disambiguate linear combinations of features.
- Nonlinear target function behaves like linear function plus noise

Therefore it suffices to consider labels given by

$$Y = \frac{1}{\sqrt{d}} \boldsymbol{\beta}^{*\top} \boldsymbol{X} + \boldsymbol{\varepsilon}\,, \quad \varepsilon_i \sim \mathcal{N}(0, \sigma_\varepsilon^2)\,.$$

Random feature regression: high-dimensional asymptotics

To develop an analytical model of double descent, we study the high-dimensional asymptotics:

$$m, d, n \to \infty \quad \text{such that} \quad \phi \equiv \frac{d}{n}\,, \psi \equiv \frac{d}{m} \quad \text{are constant.}$$

In this limit, only *linear* functions of the data can be learned.

- Intuition: only enough constraints to disambiguate linear combinations of features.
- Nonlinear target function behaves like linear function plus noise

Therefore it suffices to consider labels given by

$$Y = \frac{1}{\sqrt{d}} \boldsymbol{\beta}^{*\top} \boldsymbol{X} + \boldsymbol{\varepsilon}\,, \quad \varepsilon_i \sim \mathcal{N}(0, \sigma_\varepsilon^2)\,.$$

For simplicity, we focus on the specific setting in which,

$$X_{ij} \sim \mathcal{N}(0,1) \quad \text{and} \quad \boldsymbol{\beta}^* \sim \mathcal{N}(0, I_d)\,.$$

Random feature regression: test error

In the high-dimensional asymptotic setup from above, the random feature test error can be written as,

$$\begin{aligned} E_{\text{test}}(\lambda) &= E_{\text{GCV}}(\lambda) \\ &= \lim_{n\to\infty} \tfrac{1}{n}\|YQ\|_2^2/(\tfrac{1}{n}\text{tr}(Q))^2 \\ &= \lim_{n\to\infty} \tfrac{1}{n}\|(\tfrac{1}{\sqrt{d}}\beta^{*\top}X+\varepsilon)Q\|_2^2/(\tfrac{1}{n}\text{tr}(Q))^2 \\ &= \lim_{n\to\infty} \tfrac{1}{n}\text{tr}[(\sigma_\varepsilon^2 I_n + \tfrac{1}{d}X^\top X)Q^2]/(\tfrac{1}{n}\text{tr}(Q))^2 \\ &\equiv -\frac{\sigma_\varepsilon^2\tau_1'(\lambda)+\tau_2'(\lambda)}{\tau_1(\lambda)^2}\,, \end{aligned}$$

where we used that $\frac{\partial}{\partial\lambda}Q = -Q^2$, and we defined

$$\tau_1 = \lim_{n\to\infty}\tfrac{1}{n}\text{tr}(Q) \quad \text{and} \quad \tau_2 = \lim_{n\to\infty}\tfrac{1}{n}\text{tr}(\tfrac{1}{d}X^\top XQ)\,.$$

Computing the asymptotic test error

To compute the test error, we need:

$$\tau_1 = \lim_{n\to\infty} \tfrac{1}{n}\mathrm{tr}(K + \lambda I_n)^{-1} \quad \text{and} \quad \tau_2 = \lim_{n\to\infty} \tfrac{1}{n}\mathrm{tr}(\tfrac{1}{d}X^\top X(K + \lambda I_n)^{-1}) \,.$$

Computing the asymptotic test error

To compute the test error, we need:

$$\tau_1 = \lim_{n\to\infty} \tfrac{1}{n}\mathrm{tr}(K + \lambda I_n)^{-1} \quad \text{and} \quad \tau_2 = \lim_{n\to\infty} \tfrac{1}{n}\mathrm{tr}(\tfrac{1}{d}X^\top X(K + \lambda I_n)^{-1}) \,.$$

Recalling the definition of K,

$$K = \tfrac{1}{m}F^\top F \,, \quad F = f(\tfrac{1}{\sqrt{d}}WX) \,,$$

it is evident that the entries of F are nonlinearly dependent.

- Cannot simply utilize standard results for Wishart matrices
- Stieltjes transform is insufficient for τ_2

Computing the asymptotic test error

To compute the test error, we need:

$$\tau_1 = \lim_{n\to\infty} \tfrac{1}{n}\mathrm{tr}(K + \lambda I_n)^{-1} \quad \text{and} \quad \tau_2 = \lim_{n\to\infty} \tfrac{1}{n}\mathrm{tr}(\tfrac{1}{d}X^\top X(K + \lambda I_n)^{-1}) \,.$$

Recalling the definition of K,

$$K = \tfrac{1}{m}F^\top F \,, \quad F = f(\tfrac{1}{\sqrt{d}}WX) \,,$$

it is evident that the entries of F are nonlinearly dependent.

- Cannot simply utilize standard results for Wishart matrices
- Stieltjes transform is insufficient for τ_2

These technical challenges can be overcome with two tricks:

1. Constructing an equivalent Gaussian linearized model
2. Analyzing a suitably augmented resolvent

Computing the asymptotic test error: Gaussian equivalents

The nonlinear dependencies in $F = f(\frac{1}{\sqrt{d}}WX)$ complicate the analysis.

Can we identify a simpler matrix in the same universality class?

Computing the asymptotic test error: Gaussian equivalents

The nonlinear dependencies in $F = f(\frac{1}{\sqrt{d}}WX)$ complicate the analysis.

Can we identify a simpler matrix in the same universality class?

There exist constants c_1 and c_2 such that

$$F \cong F_{\text{lin}} \equiv c_1 \tfrac{1}{\sqrt{d}} WX + c_2 \Theta\,, \quad \Theta_{ij} \sim \mathcal{N}(0,1)\,,$$

where $F \cong F_{\text{lin}}$ indicates the two matrices share all statistics relevant for computing the test error:

$$\tau_1 = \lim_{n\to\infty} \tfrac{1}{n}\text{tr}(\tfrac{1}{m}F^\top F + \lambda I_n)^{-1} = \lim_{n\to\infty} \tfrac{1}{n}\text{tr}(\tfrac{1}{m}F_{\text{lin}}^\top F_{\text{lin}} + \lambda I_n)^{-1}$$

$$\tau_2 = \lim_{n\to\infty} \tfrac{1}{n}\text{tr}(\tfrac{1}{d}X^\top X(\tfrac{1}{m}F^\top F + \lambda I_n)^{-1}) = \lim_{n\to\infty} \tfrac{1}{n}\text{tr}(\tfrac{1}{d}X^\top X(\tfrac{1}{m}F_{\text{lin}}^\top F_{\text{lin}} + \lambda I_n)^{-1})$$

Computing the asymptotic test error: Gaussian equivalents

The nonlinear dependencies in $F = f(\frac{1}{\sqrt{d}}WX)$ complicate the analysis.

Can we identify a simpler matrix in the same universality class?

There exist constants c_1 and c_2 such that

$$F \cong F_{\text{lin}} \equiv c_1 \frac{1}{\sqrt{d}} WX + c_2 \Theta\,, \quad \Theta_{ij} \sim \mathcal{N}(0,1)\,,$$

where $F \cong F_{\text{lin}}$ indicates the two matrices share all statistics relevant for computing the test error:

$$\tau_1 = \lim_{n\to\infty} \tfrac{1}{n}\text{tr}(\tfrac{1}{m}F^\top F + \lambda I_n)^{-1} = \lim_{n\to\infty} \tfrac{1}{n}\text{tr}(\tfrac{1}{m}F_{\text{lin}}^\top F_{\text{lin}} + \lambda I_n)^{-1}$$
$$\tau_2 = \lim_{n\to\infty} \tfrac{1}{n}\text{tr}(\tfrac{1}{d}X^\top X(\tfrac{1}{m}F^\top F + \lambda I_n)^{-1}) = \lim_{n\to\infty} \tfrac{1}{n}\text{tr}(\tfrac{1}{d}X^\top X(\tfrac{1}{m}F_{\text{lin}}^\top F_{\text{lin}} + \lambda I_n)^{-1})$$

How can we compute these traces? Need to augment the resolvent.

Computing the asymptotic test error: resolvent method

Recall from Part 2 that the resolvent method identifies consistency relations between suitably chosen submatrices of the resolvent.

Here we can undertake a similar analysis as in Part 2, but now on an augmented matrix,

$$H = \begin{bmatrix} \lambda I_n & \frac{1}{\sqrt{m}} F_{\text{lin}}^\top \\ \frac{1}{\sqrt{m}} F_{\text{lin}} & -I_m \end{bmatrix},$$

which encodes the resolvent through $Q = (K + \lambda I_n)^{-1} = [H^{-1}]_{1:n,1:n}$.

Computing the asymptotic test error: resolvent method

Recall from Part 2 that the resolvent method identifies consistency relations between suitably chosen submatrices of the resolvent.

Here we can undertake a similar analysis as in Part 2, but now on an augmented matrix,

$$H = \begin{bmatrix} \lambda I_n & \frac{1}{\sqrt{m}} F_{\text{lin}}^\top \\ \frac{1}{\sqrt{m}} F_{\text{lin}} & -I_m \end{bmatrix},$$

which encodes the resolvent through $Q = (K + \lambda I_n)^{-1} = [H^{-1}]_{1:n,1:n}$.

To derive consistency relations, we consider two submatrices: $H^{(1)}$ (leaving out row/column 1), and $H^{(n+1)}$ (leaving out row/column $n+1$).

As before, we use the Sherman-Morrison formula to compute $[H^{(1)}]^{-1}$ and $[H^{(n+1)}]^{-1}$, and relate them to Q and rows/columns of F_{lin}.

Straightforward concentration arguments eventually lead to coupled self-consistent equations for τ_1 and τ_2 [Adlam et al., 2019].

Computing the asymptotic test error: free probability

An alternative augmentation of the resolvent completely linearizes the dependence on the random matrices:

$$M = \begin{bmatrix} \lambda I_n & \frac{c_2}{m}\Theta^\top & \frac{c_1}{\sqrt{d}m}X^\top & 0 \\ c_2\Theta & -I_m & 0 & \frac{c_1}{\sqrt{d}}W \\ 0 & W^\top & -I_d & 0 \\ X & 0 & 0 & -I_d \end{bmatrix},$$

where the Schur complement formula now gives,

$$\tau_1 = \lim_{n\to\infty} \tfrac{1}{n}\mathrm{tr}([M^{-1}]_{1,1}), \quad \text{and} \quad \tau_2 = \lim_{n\to\infty} \tfrac{1}{n}\mathrm{tr}([M^{-1}]_{4,3}).$$

The asymptotic blockwise traces $\mathrm{tr}([M^{-1}]_{a,b})$ can themselves be computed using free probability [Adlam and Pennington, 2020].

Computing the asymptotic test error: free probability

M is linear in the random matrices X, W, and Θ:

$$M = \begin{bmatrix} \lambda I_n & 0 & 0 & 0 \\ 0 & -I_m & 0 & 0 \\ 0 & 0 & -I_d & 0 \\ 0 & 0 & 0 & -I_d \end{bmatrix} + \begin{bmatrix} 0 & 0 & \frac{c_1}{\sqrt{dm}} X^\top & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ X & 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{c_1}{\sqrt{d}} W \\ 0 & W^\top & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & \frac{c_2}{m} \Theta^\top & 0 & 0 \\ c_2 \Theta & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Can we compute the blockwise traces with free probability via the R-transform?

Computing the asymptotic test error: free probability

M is linear in the random matrices X, W, and Θ:

$$M = \begin{bmatrix} \lambda I_n & 0 & 0 & 0 \\ 0 & -I_m & 0 & 0 \\ 0 & 0 & -I_d & 0 \\ 0 & 0 & 0 & -I_d \end{bmatrix} + \begin{bmatrix} 0 & 0 & \frac{c_1}{\sqrt{dm}} X^\top & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ X & 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{c_1}{\sqrt{d}} W \\ 0 & W^\top & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & \frac{c_2}{m} \Theta^\top & 0 & 0 \\ c_2 \Theta & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Can we compute the blockwise traces with free probability via the R-transform?

Not naively: the additive terms are independent, but not free over $\mathbb{C}$.

However, they are free over $M_4(\mathbb{C})$, and there exists a suitable *operator-valued* generalization of the R-transform that enables the necessary computations [Mingo and Speicher, 2017].

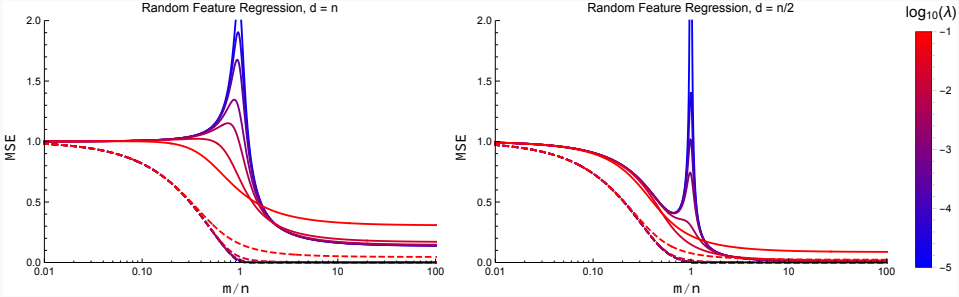
Asymptotic test error

Theorem

Let $\eta = \mathbb{E}[f(g)^2]$ and $\zeta = (\mathbb{E}[gf(g)])^2$ for $g \sim \mathcal{N}(0,1)$. Then, the asymptotic traces $\tau_1(\lambda)$ and $\tau_2(\lambda)$ are given by solutions to the polynomial system,

$$\zeta\tau_1\tau_2\left(1-\lambda\tau_1\right) = \phi/\psi\left(\zeta\tau_1\tau_2 + \phi(\tau_2-\tau_1)\right) = (\tau_1-\tau_2)\,\phi\left((\eta-\zeta)\tau_1 + \zeta\tau_2\right),$$

and, $E_{\text{train}} = -\lambda^2(\sigma_\varepsilon^2\tau_1' + \tau_2')$ and $E_{\text{test}} = -(\sigma_\varepsilon^2\tau_1' + \tau_2')/\tau_1^2$.



References

Ben Adlam and Jeffrey Pennington. The neural tangent kernel in high dimensions: Triple descent and a multi-scale theory of generalization. In *International Conference on Machine Learning*. PMLR, 2020. URL *https://arxiv.org/pdf/2008.06786.pdf*.

Ben Adlam, Jake Levinson, and Jeffrey Pennington. A random matrix perspective on mixtures of nonlinearities for deep learning. *arXiv preprint arXiv:1912.00827*, 2019.

Madhu S Advani, Andrew M Saxe, and Haim Sompolinsky. High-dimensional dynamics of generalization error in neural networks. *Neural Networks*, 132:428–446, 2020.

Peter L Bartlett, Philip M Long, Gábor Lugosi, and Alexander Tsigler. Benign overfitting in linear regression. *Proceedings of the National Academy of Sciences*, 117(48):30063–30070, 2020.

Mikhail Belkin, Siyuan Ma, and Soumik Mandal. To understand deep learning we need to understand kernel learning. In *International Conference on Machine Learning*, pages 541–549. PMLR, 2018.

Mikhail Belkin, Daniel Hsu, Siyuan Ma, and Soumik Mandal. Reconciling modern machine-learning practice and the classical bias–variance trade-off. *Proceedings of the National Academy of Sciences*, 116(32):15849–15854, 2019.

References ii

Mikhail Belkin, Daniel Hsu, and Ji Xu. Two models of double descent for weak features. *SIAM Journal on Mathematics of Data Science*, 2(4):1167–1180, 2020.

Alfredo Canziani, Adam Paszke, and Eugenio Culurciello. An analysis of deep neural network models for practical applications. *arXiv preprint arXiv:1605.07678*, 2016.

Trevor Hastie, Andrea Montanari, Saharon Rosset, and Ryan J Tibshirani. Surprises in high-dimensional ridgeless least squares interpolation. *arXiv preprint arXiv:1903.08560*, 2019. URL https://arxiv.org/pdf/1903.08560.pdf.

Arthur Jacot, Berfin Şimşek, Francesco Spadaro, Clément Hongler, and Franck Gabriel. Kernel alignment risk estimator: risk prediction from training data. *arXiv preprint arXiv:2006.09796*, 2020.

Tengyuan Liang and Alexander Rakhlin. Just interpolate: Kernel "ridgeless" regression can generalize. *The Annals of Statistics*, 48 (3):1329–1347, 2020.

Zhenyu Liao, Romain Couillet, and Michael W Mahoney. A random matrix analysis of random fourier features: beyond the gaussian kernel, a precise phase transition, and the corresponding double descent. *arXiv preprint arXiv:2006.05013*, 2020. URL https://arxiv.org/pdf/2006.05013.pdf.

Song Mei and Andrea Montanari. The generalization error of random features regression: Precise asymptotics and double descent curve. *arXiv preprint arXiv:1908.05355*, 2019. URL https://arxiv.org/pdf/1908.05355.pdf.

James A Mingo and Roland Speicher. *Free probability and random matrices*, volume 35. Springer, 2017.

Partha P Mitra. Understanding overfitting peaks in generalization error: Analytical risk curves for *l*_2 and *l*_1 penalized interpolation. *arXiv preprint arXiv:1906.03667*, 2019.

References iii

Preetum Nakkiran, Gal Kaplun, Yamini Bansal, Tristan Yang, Boaz Barak, and Ilya Sutskever. Deep double descent: Where bigger models and more data hurt. *arXiv preprint arXiv:1912.02292*, 2019.

Alexander Rakhlin and Xiyu Zhai. Consistency of interpolation with laplace kernels is a high-dimensional phenomenon. In *Conference on Learning Theory*, pages 2595–2623. PMLR, 2019.

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning (still) requires rethinking generalization. *Communications of the ACM*, 64(3):107–115, 2021.