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:

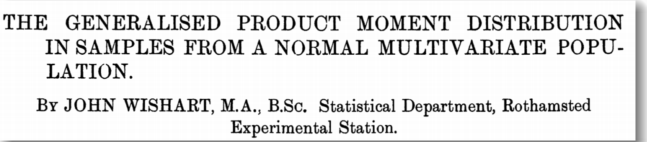
$$Z = \begin{bmatrix} 1.066 & 0.908 & 1.026 & -0.294 & 0.879 \\ 0.908 & -1.794 & 0.596 & -1.014 & -0.103 \\ 1.026 & 0.596 & -0.246 & 0.968 & 0.750 \\ -0.294 & -1.014 & 0.968 & 0.184 & 0.812 \\ 0.879 & -0.103 & 0.750 & 0.812 & 0.210 \end{bmatrix}$$

Goal of Random Matrix Theory is to understand their

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

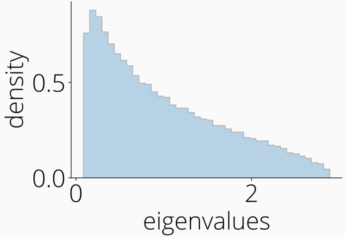
Where do Random Matrices Come From?

1928: Eigenvalues of Normal Covariance Matrices

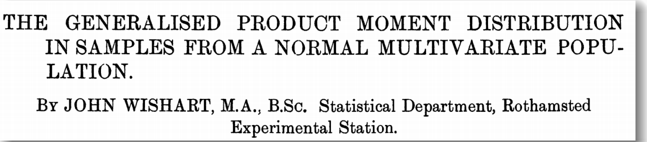

THE GENERALISED PRODUCT MOMENT DISTRIBUTION IN SAMPLES FROM A NORMAL MULTIVARIATE POPULATION.

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


John Wishart



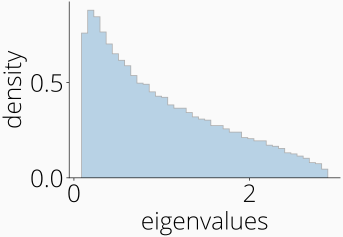
1928: Eigenvalues of Normal Covariance Matrices



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

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

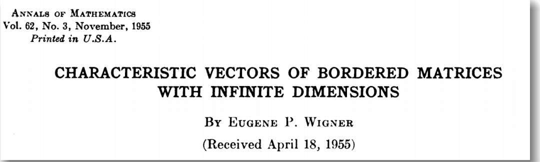
John Wishart



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

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

1955: Random Symmetric Matrices

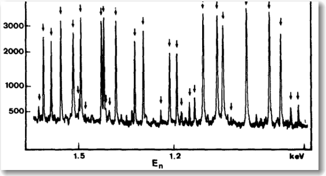


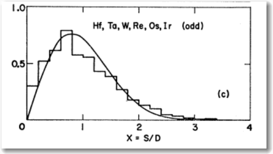
ANNALS OF MATHEMATICS
Vol. 62, No. 3, November, 1955
Printed in U.S.A.

CHARACTERISTIC VECTORS OF BORDERED MATRICES WITH INFINITE DIMENSIONS

BY EUGENE P. WIGNER

(Received April 18, 1955)





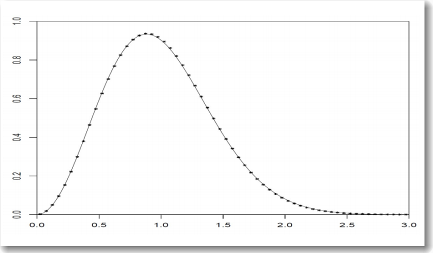
Eugene Wigner

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 ...

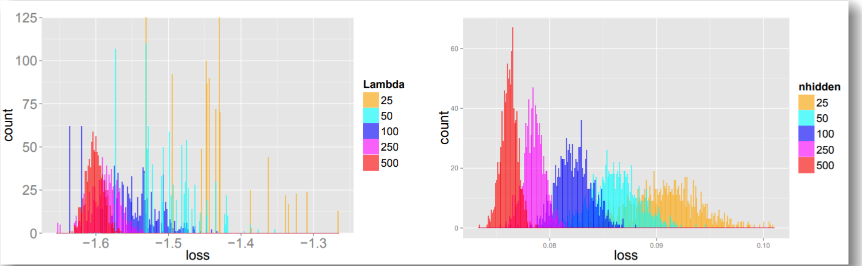


Distribution function of gaps between eigenvalues compared with histogram of gaps between ζ zeros. Source: [Odlyzko, 1987]

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]

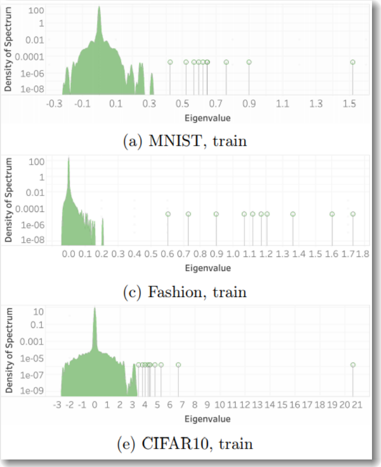


Loss study through spin-glass model. Scaled test losses for the spin-glass (left) and the neural network (right). Source: Choromanska et al. [2015] *The Loss Surfaces of Multilayer Networks.*

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*



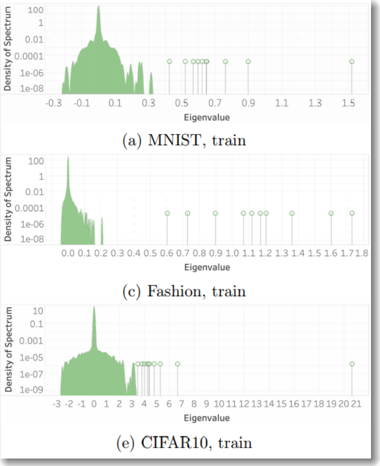
Source: [Papyan, 2020]

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] ...



(a) MNIST, train

(c) Fashion, train

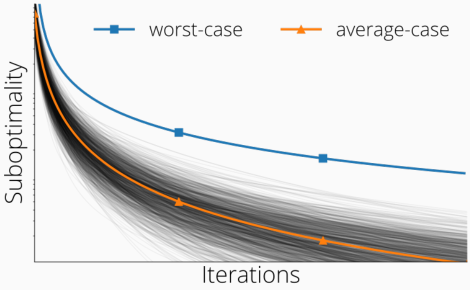
(e) CIFAR10, train

Source: [Papyan, 2020]

Random Matrices in Machine Learning: Numerical Algorithms

Analyze algorithms with **random** data.

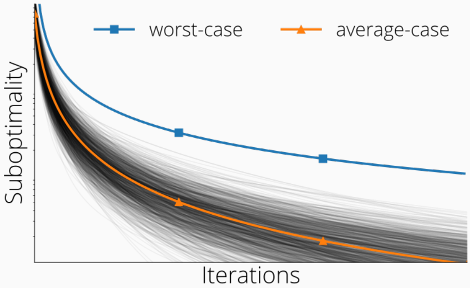
- Simplex [Borgwardt, 1987, Smale, 1983, Spielman and Teng, 2004, Vershynin, 2009] etc.
- Conjugate Gradient [Deift and Trogdon, 2017, Paquette and Trogdon, 2020]
- Acceleration [Pedregosa and Scieur, 2020, Lacotte and Pilanci, 2020]



Random Matrices in Machine Learning: Numerical Algorithms

Analyze algorithms with **random** data.

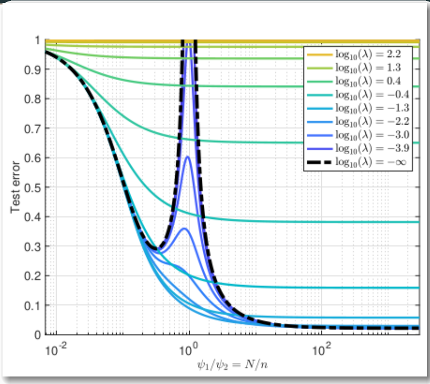
- Simplex [Borgwardt, 1987, Smale, 1983, Spielman and Teng, 2004, Vershynin, 2009] etc.
- Conjugate Gradient [Deift and Trogdon, 2017, Paquette and Trogdon, 2020]
- Acceleration [Pedregosa and Scieur, 2020, Lacotte and Pilanci, 2020]



Topic of Part 3 of this tutorial

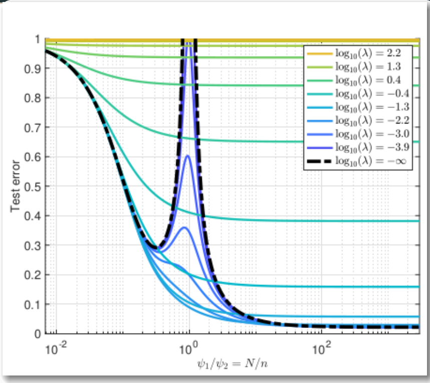
Random Matrices in Machine Learning: Generalization

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

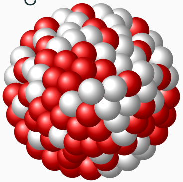
Building Blocks

Classical Random Matrix Ensembles

Gaussian Orthogonal Ensemble (GOE)

Motivation: Model Hamiltonian
heavy nuclei

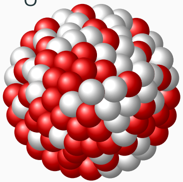
^{238}U



Gaussian Orthogonal Ensemble (GOE)

Motivation: Model Hamiltonian heavy nuclei

^{238}U



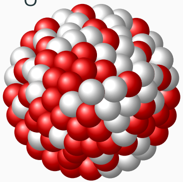
- **Rotational invariant**
  for any fixed orthogonal matrix O,

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

Gaussian Orthogonal Ensemble (GOE)

Motivation: Model Hamiltonian heavy nuclei

^{238}U



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

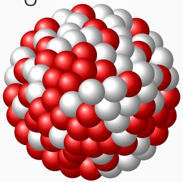
$$A \overset{\text{law}}{=} O^T A O \,.$$

- **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)

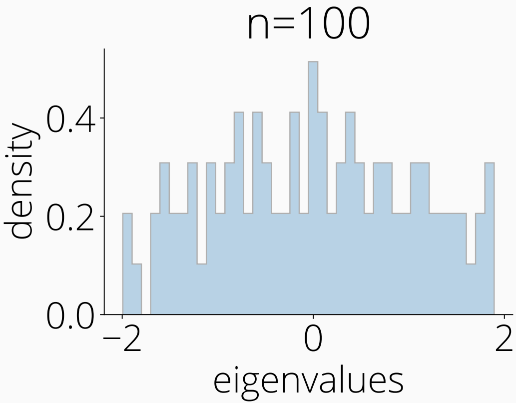
- real $n \times n$ matrix
- $\mathcal{N}(0,1)$ above diagonal
- Symmetric
- $\mathcal{N}(0,2)$ diagonal

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

Eigenvalues of GOE

Python pseudocode

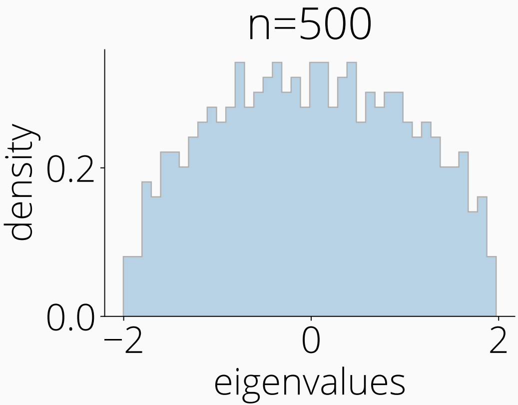
```python
import numpy as np
import matplotlib.pyplot as plt

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

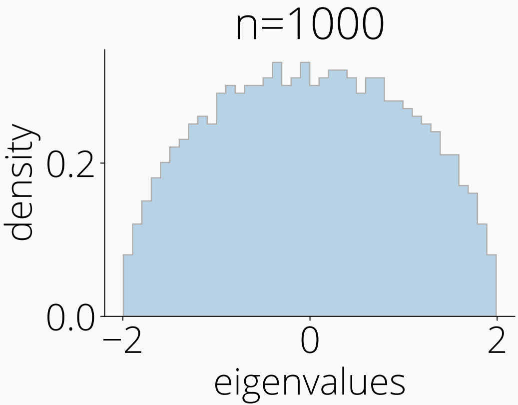
```
import numpy as np
import matplotlib.pyplot as plt

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

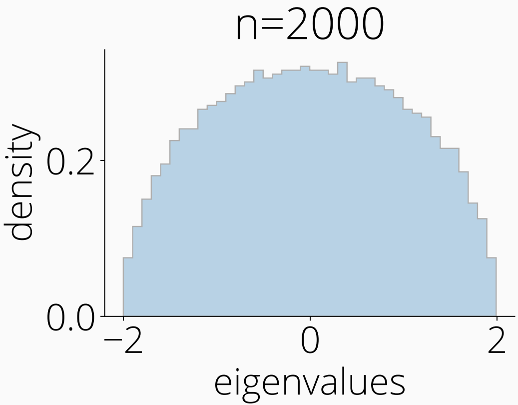
```
import numpy as np
import matplotlib.pyplot as plt

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

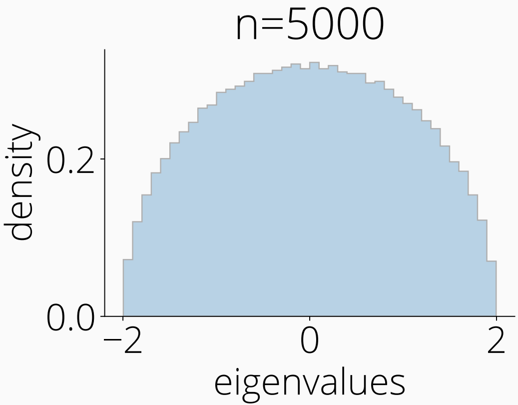
```
import numpy as np
import matplotlib.pyplot as plt

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

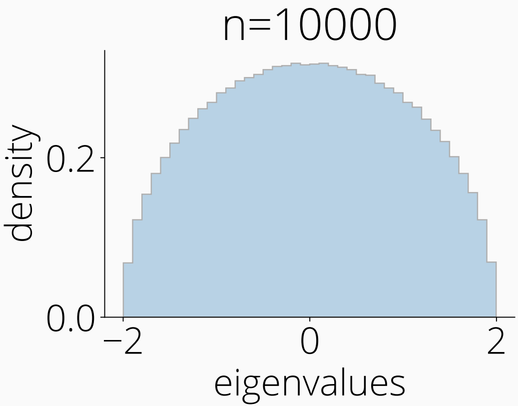
```
import numpy as np
import matplotlib.pyplot as plt

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

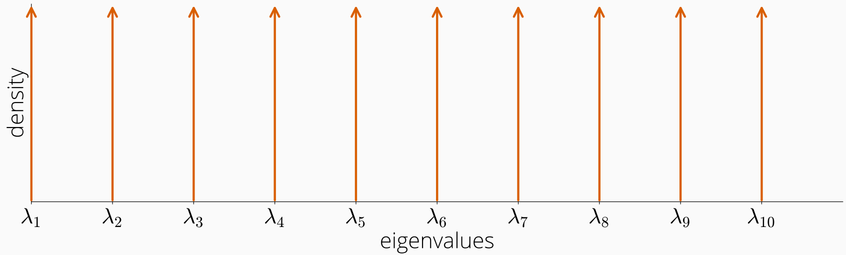
```
import numpy as np
import matplotlib.pyplot as plt

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

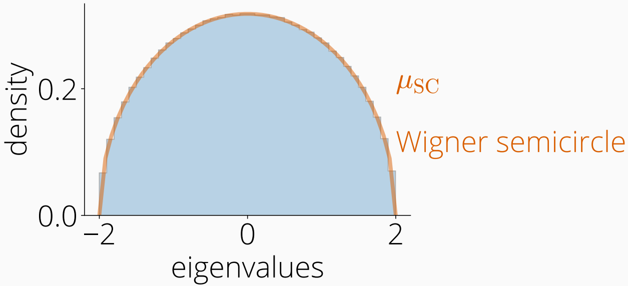
$$\mu_{\mathrm{ESD}} = \mu_n \stackrel{\text{def}}{=} \frac{1}{n} \sum_{i=1}^{n} \delta_{\lambda_i(A_n)}.$$



Wigner Semicircle Law

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

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



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 $\frac{1}{n}X^T X$

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 $\frac{1}{2n}\|Xw - \mathbf{y}\|^2$

Wishart ensemble

Wishart

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

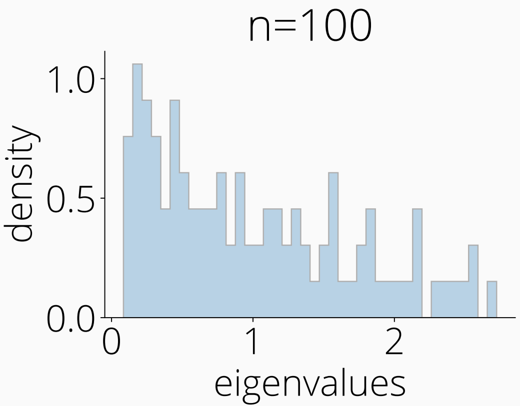
Remarks

- 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 $\frac{1}{2n}\|Xw - \mathbf{y}\|^2$
- Parameter $\mathbf{r} = \frac{d}{n}$

Wishart ensemble

Python pseudocode

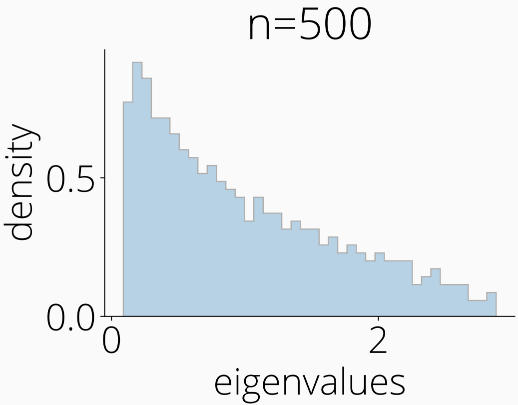
```
import numpy as np
import matplotlib.pyplot as plt

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

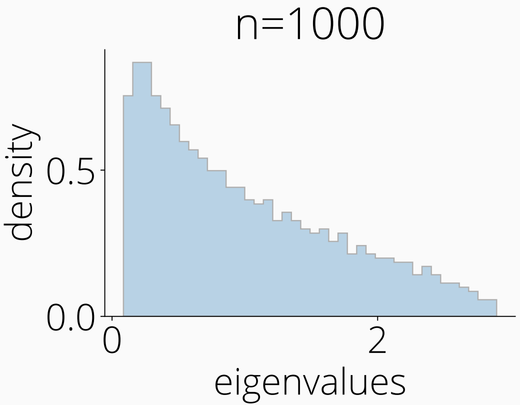
```python
import numpy as np
import matplotlib.pyplot as plt

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

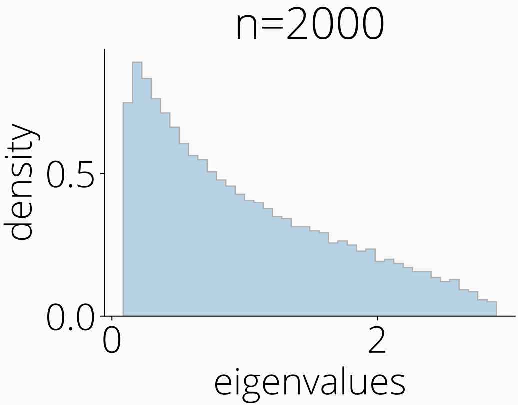
```
import numpy as np
import matplotlib.pyplot as plt

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

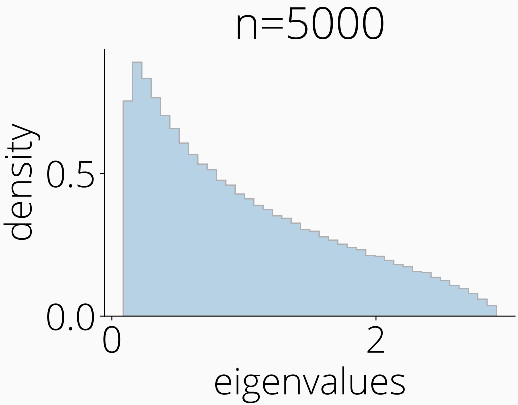
```
import numpy as np
import matplotlib.pyplot as plt

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

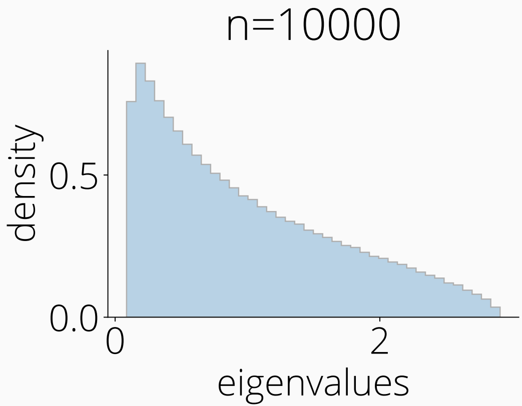
```
import numpy as np
import matplotlib.pyplot as plt

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)
```



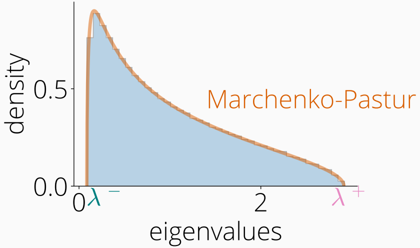
Limit of Wishart matrices

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

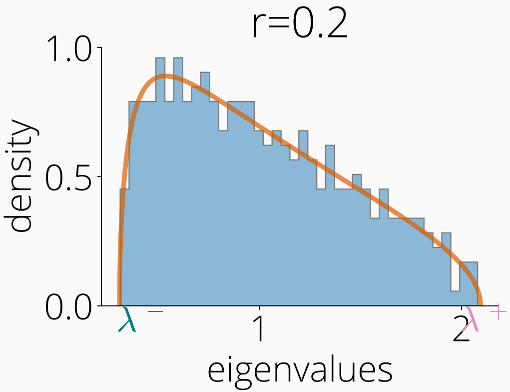
As $n, d \to \infty$, $\frac{d}{n} \to r$, μ_{ESD} converges to the Marchenko-Pastur distribution:

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

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



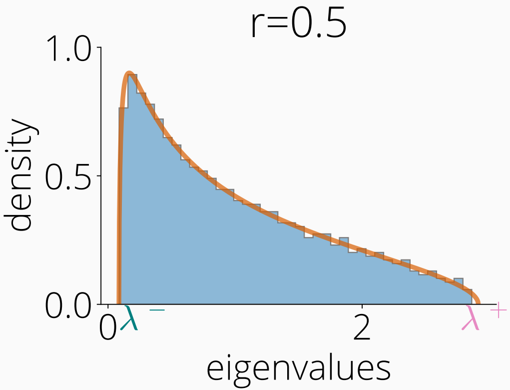
- $r < 1 \implies d < n \implies W$ is product of two **fat** matrices.
- $r > 1 \implies d > n \implies W$ is product of two **thin** matrices (rank-deficient).



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

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

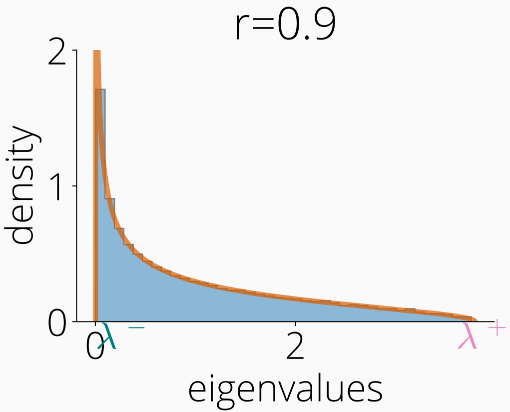
- $r < 1 \implies d < n \implies W$ is product of two **fat** matrices.
- $r > 1 \implies d > n \implies W$ is product of two **thin** matrices (rank-deficient).



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

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

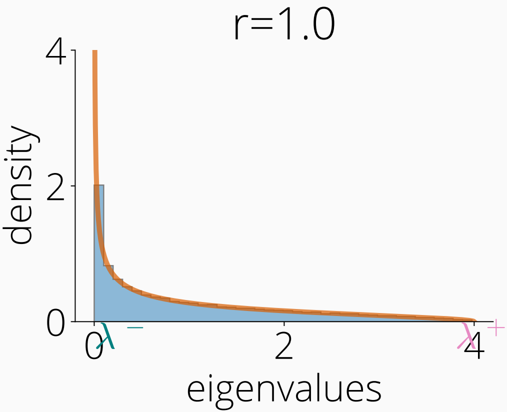
- $r < 1 \implies d < n \implies W$ is product of two **fat** matrices.
- $r > 1 \implies d > n \implies W$ is product of two **thin** matrices (rank-deficient).



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

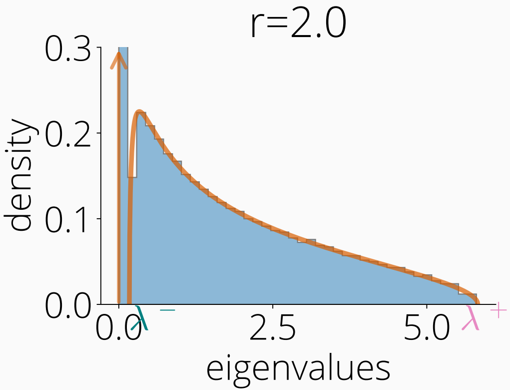
The $r = \frac{d}{n}$ parameter

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



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

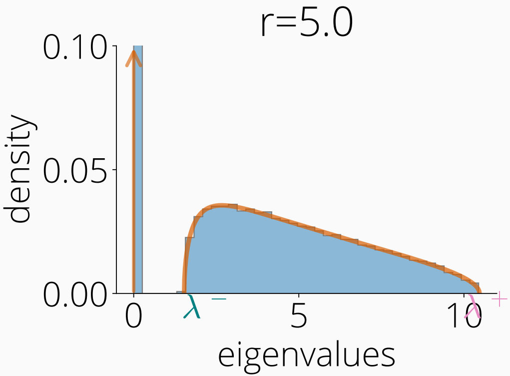
- $r < 1 \implies d < n \implies W$ is product of two **fat** matrices.
- $r > 1 \implies d > n \implies W$ is product of two **thin** matrices (rank-deficient).



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

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).



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

Universality

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

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

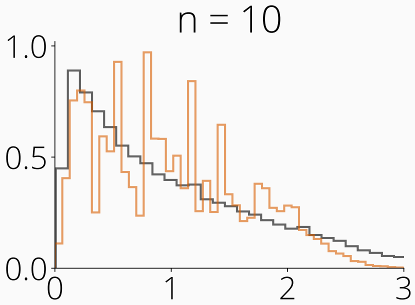
- $X_{ij} \sim \mathcal{N}(0,1)$
- $X_{ij} \sim$ Rademacher $\Pr(X_{ij} = -1) = \Pr(X_{ij} = 1) = \frac{1}{2}$

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 $\Pr(X_{ij} = -1) = \Pr(X_{ij} = 1) = \frac{1}{2}$

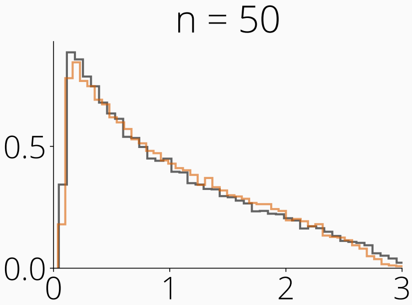


Universality

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

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 $\Pr(X_{ij} = -1) = \Pr(X_{ij} = 1) = \frac{1}{2}$

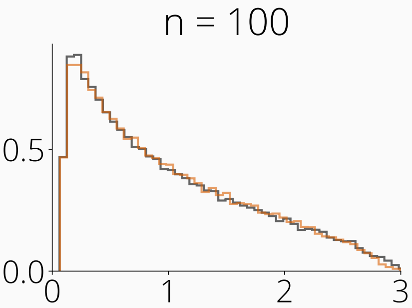


Universality

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

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

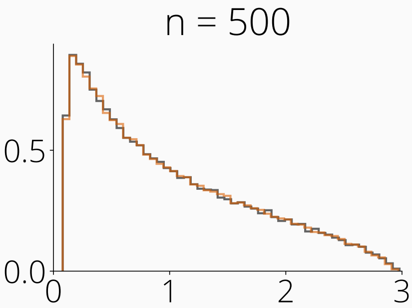
- $X_{ij} \sim \mathcal{N}(0,1)$
- $X_{ij} \sim$ Rademacher $\Pr(X_{ij} = -1) = \Pr(X_{ij} = 1) = \frac{1}{2}$



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 $\Pr(X_{ij} = -1) = \Pr(X_{ij} = 1) = \frac{1}{2}$

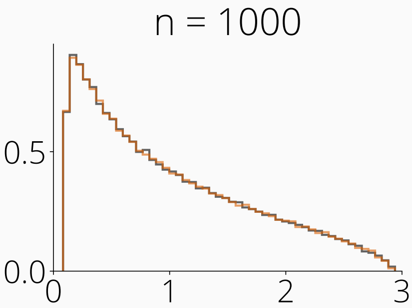


Universality

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

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

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 ...

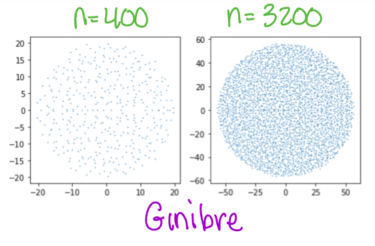
Other matrix ensembles

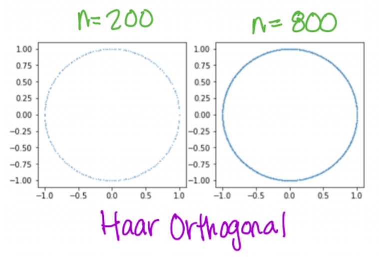
- **Ginibre.** Let G_n be $n \times n$ matrix of i.i.d. $N(0,1)$, (bilinear games [Domingo-Enrich et al., 2020])

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

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

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





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*.

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.

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.