

Random Matrix Advances in Machine Learning

(Imaging and Machine Learning)

Mathematics Workshop #3

Institut Henri Poincaré

Romain COUILLET

CentraleSupélec, L2S, University of ParisSaclay, France
GSTATS IDEX DataScience Chair, GIPSA-lab, University Grenoble-Alpes, France.

April 1st, 2019



CentraleSupélec



Outline

Basics of Random Matrix Theory

- Motivation: Large Sample Covariance Matrices
- Spiked Models

Application to Machine Learning

Outline

Basics of Random Matrix Theory

Motivation: Large Sample Covariance Matrices
Spiked Models

Application to Machine Learning

Outline

Basics of Random Matrix Theory

Motivation: Large Sample Covariance Matrices

Spiked Models

Application to Machine Learning

Context

Baseline scenario: $y_1, \dots, y_n \in \mathbb{C}^p$ (or \mathbb{R}^p) i.i.d. with $E[y_1] = 0$, $E[y_1 y_1^*] = C_p$:

Context

Baseline scenario: $y_1, \dots, y_n \in \mathbb{C}^p$ (or \mathbb{R}^p) i.i.d. with $E[y_1] = 0$, $E[y_1 y_1^*] = C_p$:

- If $y_1 \sim \mathcal{N}(0, C_p)$, ML estimator for C_p is the sample covariance matrix (SCM)

$$\hat{C}_p = \frac{1}{n} Y_p Y_p^* = \frac{1}{n} \sum_{i=1}^n y_i y_i^*$$

$(Y_p = [y_1, \dots, y_n] \in \mathbb{C}^{p \times n})$.

Context

Baseline scenario: $y_1, \dots, y_n \in \mathbb{C}^p$ (or \mathbb{R}^p) i.i.d. with $E[y_1] = 0$, $E[y_1 y_1^*] = C_p$:

- If $y_1 \sim \mathcal{N}(0, C_p)$, ML estimator for C_p is the sample covariance matrix (SCM)

$$\hat{C}_p = \frac{1}{n} Y_p Y_p^* = \frac{1}{n} \sum_{i=1}^n y_i y_i^*$$

($Y_p = [y_1, \dots, y_n] \in \mathbb{C}^{p \times n}$).

- If $n \rightarrow \infty$, then, **strong law of large numbers**

$$\hat{C}_p \xrightarrow{\text{a.s.}} C_p.$$

or equivalently, **in spectral norm**

$$\|\hat{C}_p - C_p\| \xrightarrow{\text{a.s.}} 0.$$

Context

Baseline scenario: $y_1, \dots, y_n \in \mathbb{C}^p$ (or \mathbb{R}^p) i.i.d. with $E[y_1] = 0$, $E[y_1 y_1^*] = C_p$:

- If $y_1 \sim \mathcal{N}(0, C_p)$, ML estimator for C_p is the sample covariance matrix (SCM)

$$\hat{C}_p = \frac{1}{n} Y_p Y_p^* = \frac{1}{n} \sum_{i=1}^n y_i y_i^*$$

($Y_p = [y_1, \dots, y_n] \in \mathbb{C}^{p \times n}$).

- If $n \rightarrow \infty$, then, **strong law of large numbers**

$$\hat{C}_p \xrightarrow{\text{a.s.}} C_p.$$

or equivalently, **in spectral norm**

$$\|\hat{C}_p - C_p\| \xrightarrow{\text{a.s.}} 0.$$

Random Matrix Regime

- No longer valid if $p, n \rightarrow \infty$ with $p/n \rightarrow c \in (0, \infty)$,

$$\|\hat{C}_p - C_p\| \not\rightarrow 0.$$

Context

Baseline scenario: $y_1, \dots, y_n \in \mathbb{C}^p$ (or \mathbb{R}^p) i.i.d. with $E[y_1] = 0$, $E[y_1 y_1^*] = C_p$:

- If $y_1 \sim \mathcal{N}(0, C_p)$, ML estimator for C_p is the sample covariance matrix (SCM)

$$\hat{C}_p = \frac{1}{n} Y_p Y_p^* = \frac{1}{n} \sum_{i=1}^n y_i y_i^*$$

$(Y_p = [y_1, \dots, y_n] \in \mathbb{C}^{p \times n})$.

- If $n \rightarrow \infty$, then, **strong law of large numbers**

$$\hat{C}_p \xrightarrow{\text{a.s.}} C_p.$$

or equivalently, **in spectral norm**

$$\|\hat{C}_p - C_p\| \xrightarrow{\text{a.s.}} 0.$$

Random Matrix Regime

- No longer valid if $p, n \rightarrow \infty$ with $p/n \rightarrow c \in (0, \infty)$,

$$\|\hat{C}_p - C_p\| \not\rightarrow 0.$$

- For practical p, n with $p \simeq n$, leads to dramatically wrong conclusions

Context

Baseline scenario: $y_1, \dots, y_n \in \mathbb{C}^p$ (or \mathbb{R}^p) i.i.d. with $E[y_1] = 0$, $E[y_1 y_1^*] = C_p$:

- If $y_1 \sim \mathcal{N}(0, C_p)$, ML estimator for C_p is the sample covariance matrix (SCM)

$$\hat{C}_p = \frac{1}{n} Y_p Y_p^* = \frac{1}{n} \sum_{i=1}^n y_i y_i^*$$

($Y_p = [y_1, \dots, y_n] \in \mathbb{C}^{p \times n}$).

- If $n \rightarrow \infty$, then, **strong law of large numbers**

$$\hat{C}_p \xrightarrow{\text{a.s.}} C_p.$$

or equivalently, **in spectral norm**

$$\|\hat{C}_p - C_p\| \xrightarrow{\text{a.s.}} 0.$$

Random Matrix Regime

- No longer valid if $p, n \rightarrow \infty$ with $p/n \rightarrow c \in (0, \infty)$,

$$\|\hat{C}_p - C_p\| \not\rightarrow 0.$$

- For practical p, n with $p \simeq n$, leads to dramatically wrong conclusions
- **Even for $p = n/100$.**

The Marčenko–Pastur law

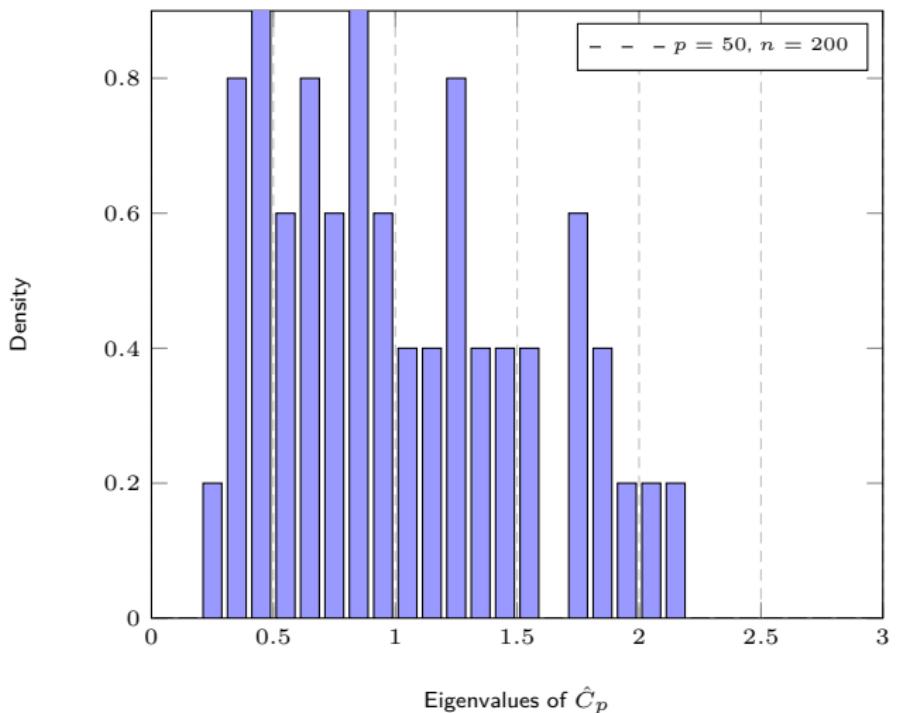


Figure: Histogram of the eigenvalues of \hat{C}_p for $c = 1/4$, $C_p = I_p$.

The Marčenko–Pastur law

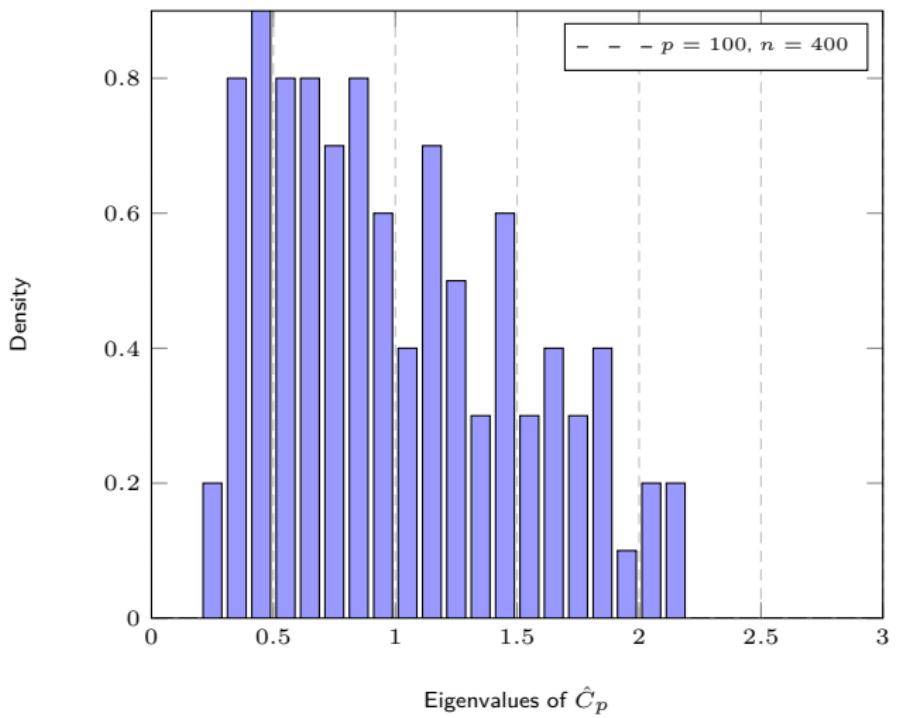


Figure: Histogram of the eigenvalues of \hat{C}_p for $c = 1/4$, $C_p = I_p$.

The Marčenko–Pastur law

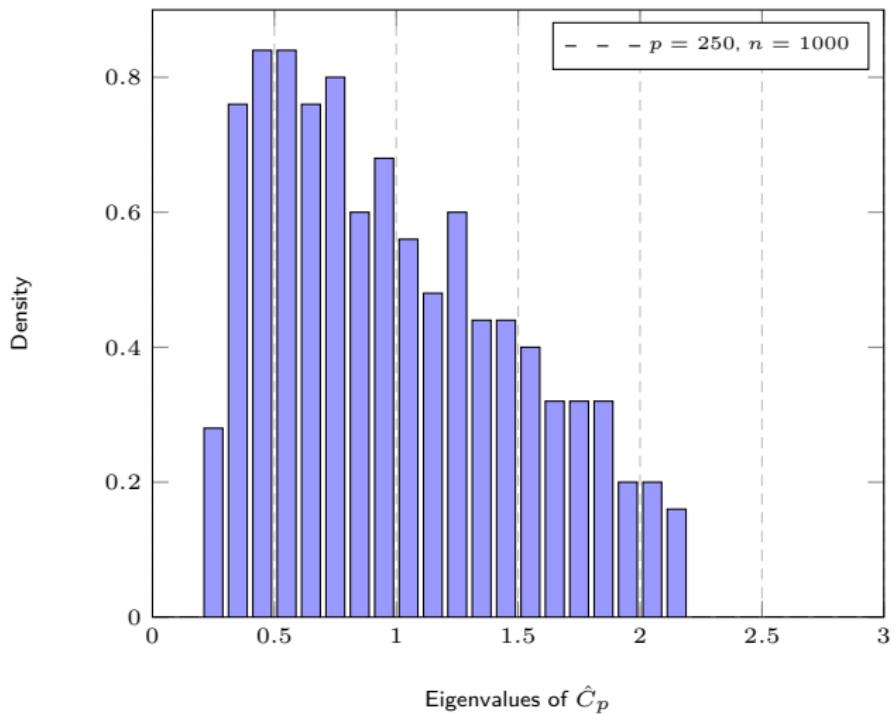


Figure: Histogram of the eigenvalues of \hat{C}_p for $c = 1/4$, $C_p = I_p$.

The Marčenko–Pastur law

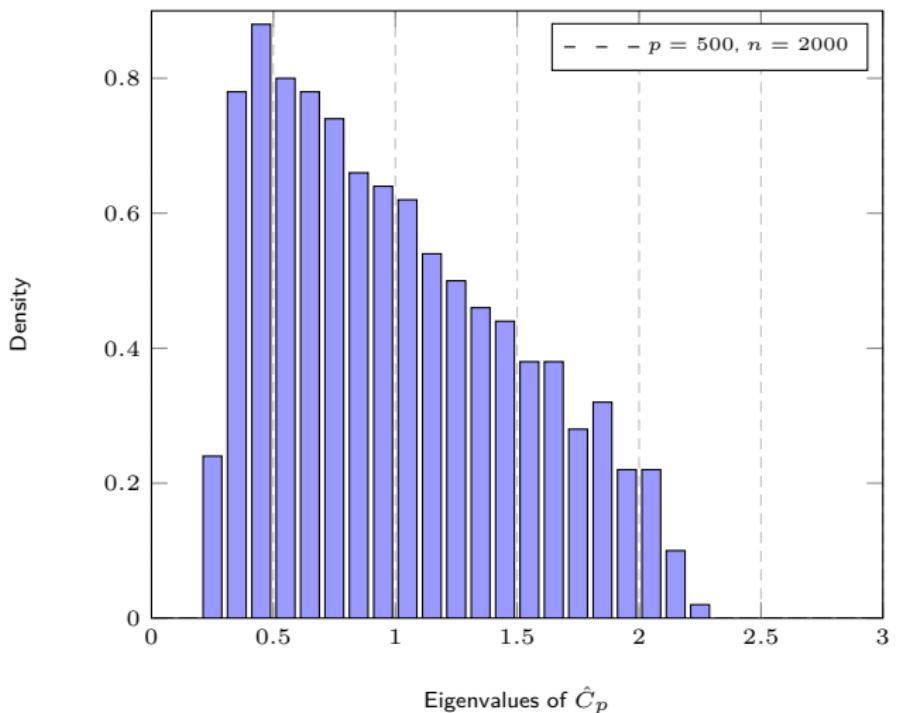


Figure: Histogram of the eigenvalues of \hat{C}_p for $c = 1/4$, $C_p = I_p$.

The Marčenko–Pastur law

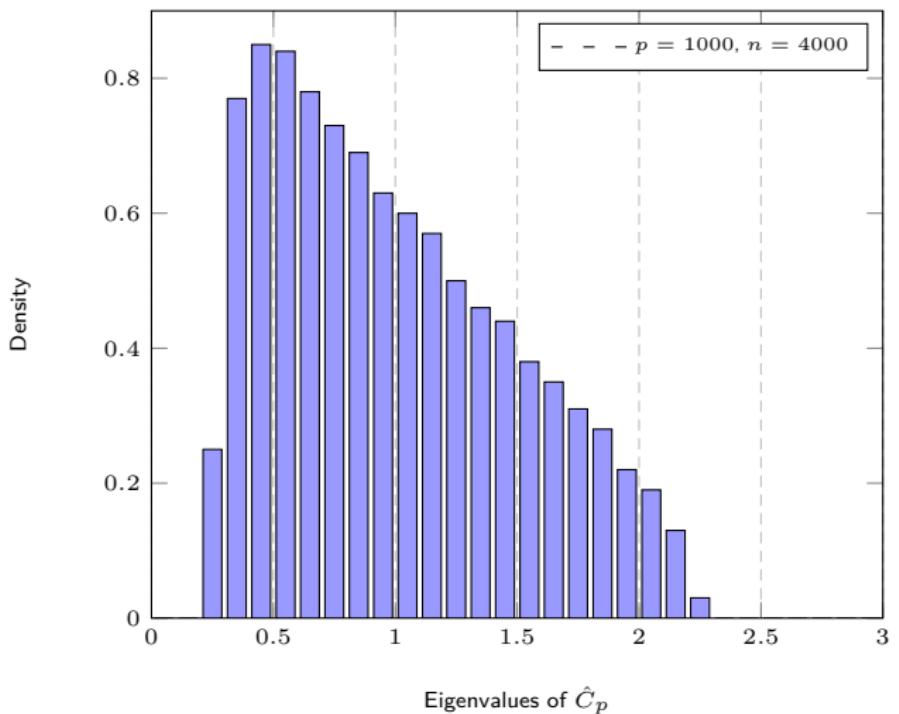


Figure: Histogram of the eigenvalues of \hat{C}_p for $c = 1/4$, $C_p = I_p$.

The Marčenko–Pastur law

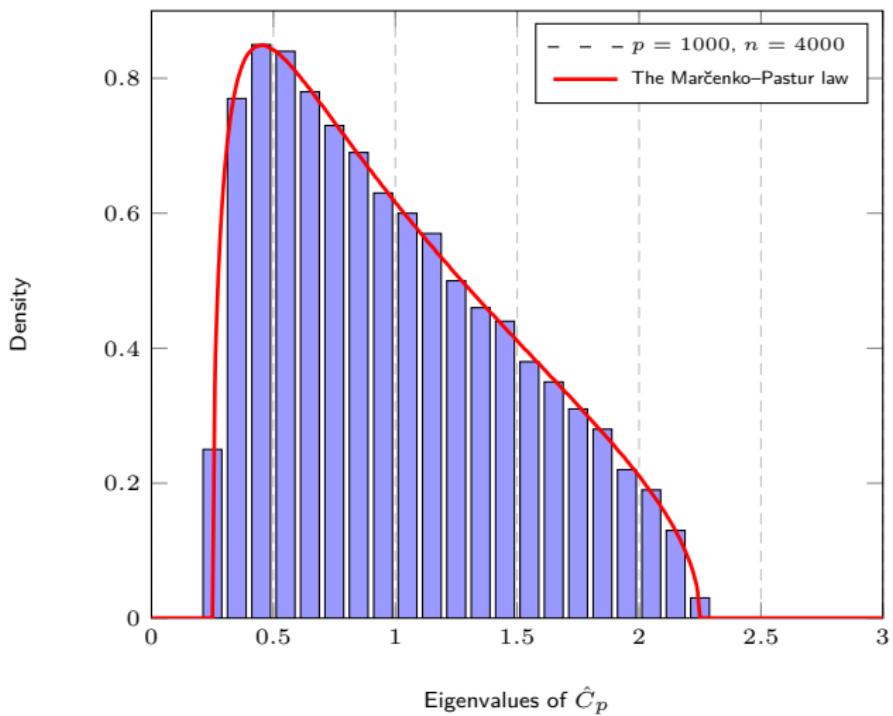


Figure: Histogram of the eigenvalues of \hat{C}_p for $c = 1/4$, $C_p = I_p$.

The Marčenko–Pastur law

Definition (Empirical Spectral Density)

Empirical spectral density (e.s.d.) μ_p of Hermitian matrix $A_p \in \mathbb{C}^{p \times p}$ is

$$\mu_p = \frac{1}{p} \sum_{i=1}^p \delta_{\lambda_i(A_p)}.$$

The Marčenko–Pastur law

Definition (Empirical Spectral Density)

Empirical spectral density (e.s.d.) μ_p of Hermitian matrix $A_p \in \mathbb{C}^{p \times p}$ is

$$\mu_p = \frac{1}{p} \sum_{i=1}^p \delta_{\lambda_i(A_p)}.$$

Theorem (Marčenko–Pastur Law [Marčenko, Pastur'67])

$X_p \in \mathbb{C}^{p \times n}$ with i.i.d. zero mean, unit variance entries.

As $p, n \rightarrow \infty$ with $p/n \rightarrow c \in (0, \infty)$, e.s.d. μ_p of $\frac{1}{n} X_p X_p^*$ satisfies

$$\mu_p \xrightarrow{\text{a.s.}} \mu_c$$

weakly, where

- $\mu_c(\{0\}) = \max\{0, 1 - c^{-1}\}$

The Marčenko–Pastur law

Definition (Empirical Spectral Density)

Empirical spectral density (e.s.d.) μ_p of Hermitian matrix $A_p \in \mathbb{C}^{p \times p}$ is

$$\mu_p = \frac{1}{p} \sum_{i=1}^p \delta_{\lambda_i(A_p)}.$$

Theorem (Marčenko–Pastur Law [Marčenko, Pastur'67])

$X_p \in \mathbb{C}^{p \times n}$ with i.i.d. zero mean, unit variance entries.

As $p, n \rightarrow \infty$ with $p/n \rightarrow c \in (0, \infty)$, e.s.d. μ_p of $\frac{1}{n} X_p X_p^*$ satisfies

$$\mu_p \xrightarrow{\text{a.s.}} \mu_c$$

weakly, where

- ▶ $\mu_c(\{0\}) = \max\{0, 1 - c^{-1}\}$
- ▶ on $(0, \infty)$, μ_c has continuous density f_c supported on $[(1 - \sqrt{c})^2, (1 + \sqrt{c})^2]$

$$f_c(x) = \frac{1}{2\pi cx} \sqrt{(x - (1 - \sqrt{c})^2)((1 + \sqrt{c})^2 - x)}.$$

The Marčenko–Pastur law

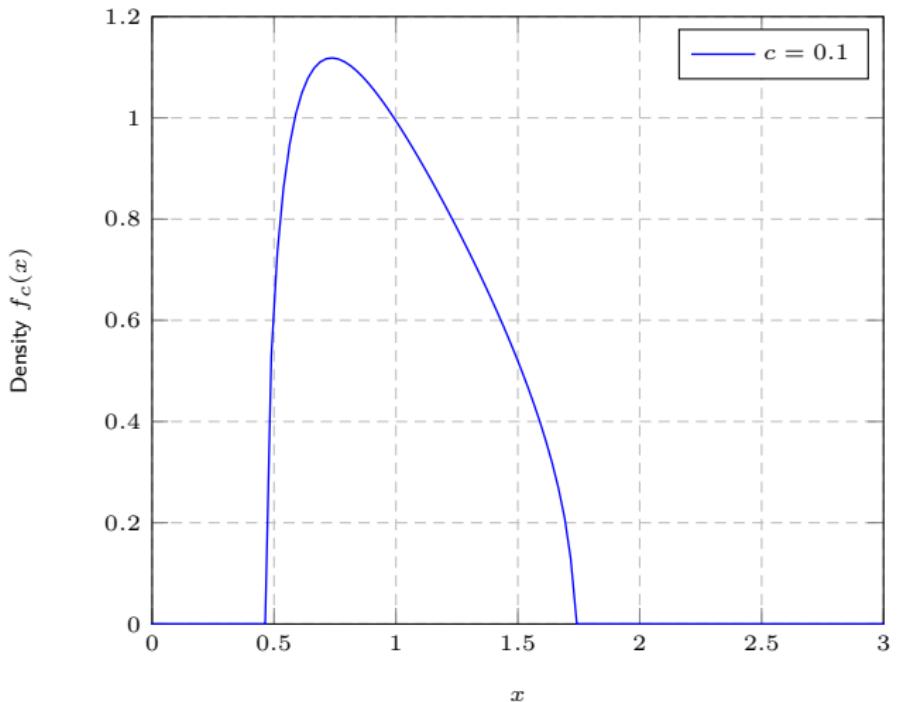


Figure: Marčenko–Pastur law for different limit ratios $c = \lim_{p \rightarrow \infty} p/n$.

The Marčenko–Pastur law

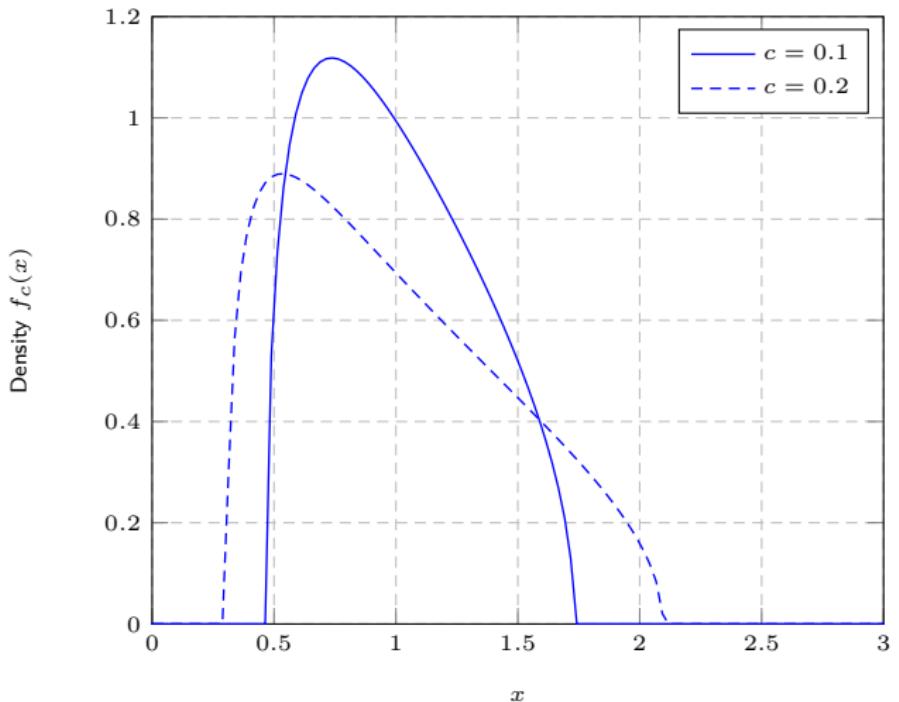


Figure: Marčenko–Pastur law for different limit ratios $c = \lim_{p \rightarrow \infty} p/n$.

The Marčenko–Pastur law

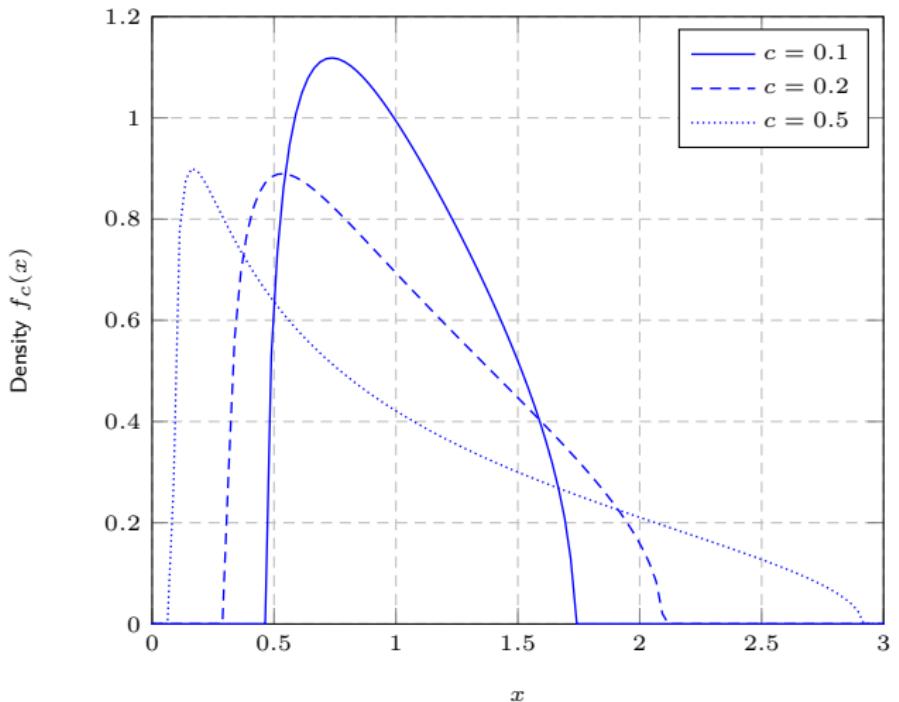


Figure: Marčenko–Pastur law for different limit ratios $c = \lim_{p \rightarrow \infty} p/n$.

Outline

Basics of Random Matrix Theory

Motivation: Large Sample Covariance Matrices

Spiked Models

Application to Machine Learning

Spiked Models

Small rank perturbation: $C_p = I_p + P$, P of low rank.

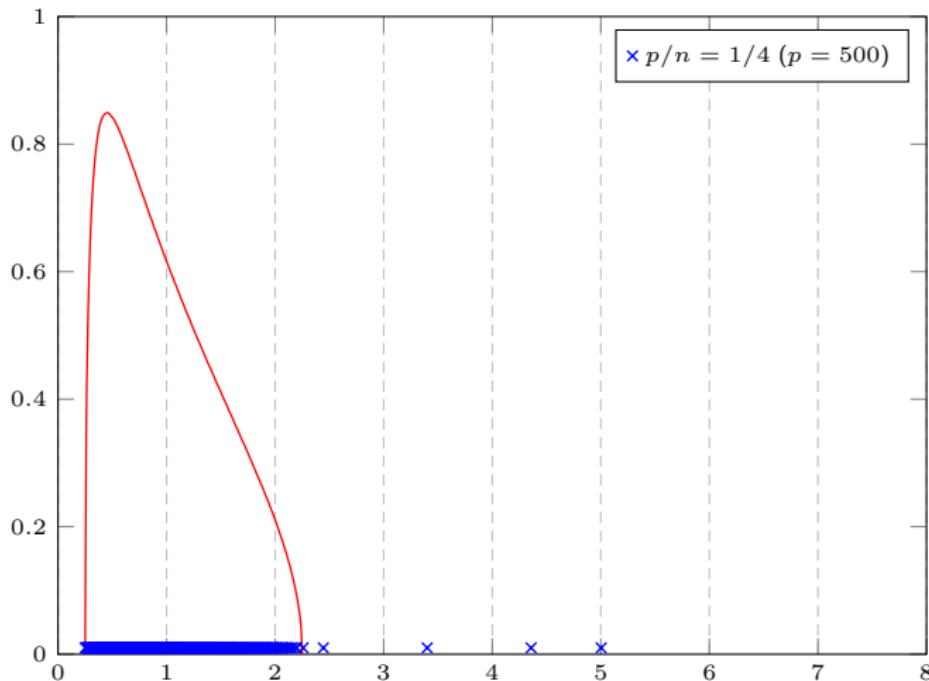


Figure: Eigenvalues of $\frac{1}{n} Y_p Y_p^\top$, $\text{eig}(C_p) = \underbrace{\{1, \dots, 1\}}_{p-4}, 2, 3, 4, 5\}$.

Spiked Models

Small rank perturbation: $C_p = I_p + P$, P of low rank.

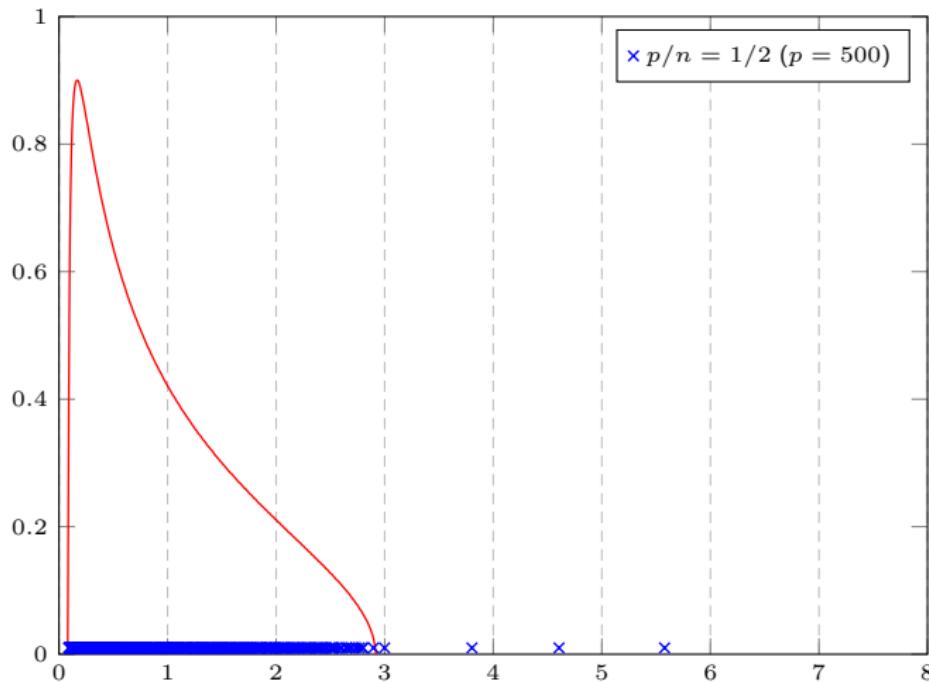


Figure: Eigenvalues of $\frac{1}{n} Y_p Y_p^\top$, $\text{eig}(C_p) = \underbrace{\{1, \dots, 1\}}_{p-4}, 2, 3, 4, 5\}$.

Spiked Models

Small rank perturbation: $C_p = I_p + P$, P of low rank.

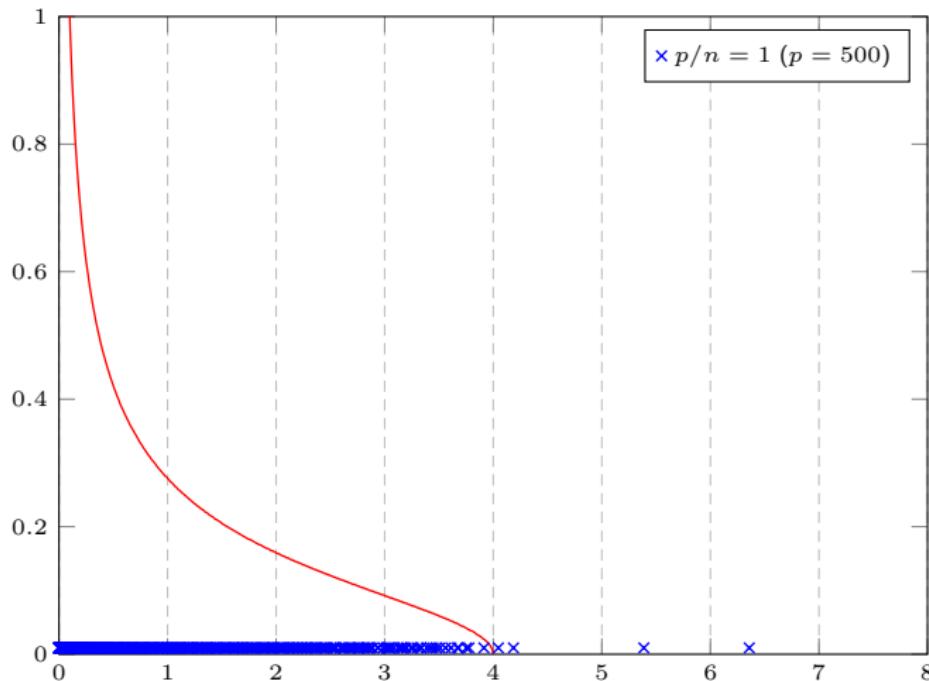


Figure: Eigenvalues of $\frac{1}{n} Y_p Y_p^\top$, $\text{eig}(C_p) = \underbrace{\{1, \dots, 1\}}_{p-4}, 2, 3, 4, 5\}$.

Spiked Models

Small rank perturbation: $C_p = I_p + P$, P of low rank.

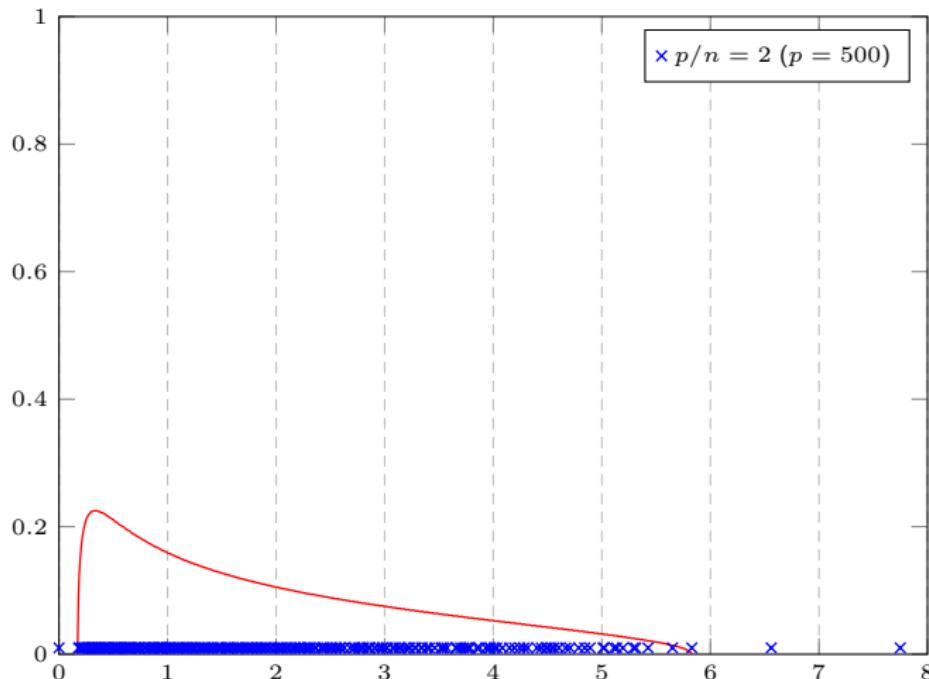


Figure: Eigenvalues of $\frac{1}{n} Y_p Y_p^\top$, $\text{eig}(C_p) = \underbrace{\{1, \dots, 1\}}_{p-4}, 2, 3, 4, 5\}$.

Spiked Models

Theorem (Eigenvalues [Baik, Silverstein'06])

Let $Y_p = C_p^{\frac{1}{2}} X_p$, with

- ▶ X_p with i.i.d. zero mean, unit variance, $E[|X_p|_{ij}^4] < \infty$.
- ▶ $C_p = I_p + P$, $P = U\Omega U^*$, where, for K fixed,

$$\Omega = \text{diag}(\omega_1, \dots, \omega_K) \in \mathbb{R}^{K \times K}, \text{ with } \omega_1 \geq \dots \geq \omega_K > 0.$$

Spiked Models

Theorem (Eigenvalues [Baik, Silverstein'06])

Let $Y_p = C_p^{\frac{1}{2}} X_p$, with

- ▶ X_p with i.i.d. zero mean, unit variance, $E[|X_p|_{ij}^4] < \infty$.
- ▶ $C_p = I_p + P$, $P = U\Omega U^*$, where, for K fixed,

$$\Omega = \text{diag}(\omega_1, \dots, \omega_K) \in \mathbb{R}^{K \times K}, \text{ with } \omega_1 \geq \dots \geq \omega_K > 0.$$

Then, as $p, n \rightarrow \infty$, $p/n \rightarrow c \in (0, \infty)$, denoting $\lambda_m = \lambda_m(\frac{1}{n} Y_p Y_p^*)$ ($\lambda_m > \lambda_{m+1}$),

$$\lambda_m \xrightarrow{\text{a.s.}} \begin{cases} 1 + \omega_m + c \frac{1+\omega_m}{\omega_m} > (1 + \sqrt{c})^2 & , \omega_m > \sqrt{c} \\ (1 + \sqrt{c})^2 & , \omega_m \in (0, \sqrt{c}] \end{cases}$$

Spiked Models

Theorem (Eigenvectors [Paul'07])

Let $Y_p = C_p^{\frac{1}{2}} X_p$, with

- ▶ X_p with i.i.d. zero mean, unit variance, $E[|X_p|_{ij}^4] < \infty$.
- ▶ $C_p = I_p + P$, $P = U\Omega U^* = \sum_{i=1}^K \omega_i u_i u_i^*$, $\omega_1 > \dots > \omega_M > 0$.

Spiked Models

Theorem (Eigenvectors [Paul'07])

Let $Y_p = C_p^{\frac{1}{2}} X_p$, with

- ▶ X_p with i.i.d. zero mean, unit variance, $E[|X_p|_{ij}^4] < \infty$.
- ▶ $C_p = I_p + P$, $P = U\Omega U^* = \sum_{i=1}^K \omega_i u_i u_i^*$, $\omega_1 > \dots > \omega_M > 0$.

Then, as $p, n \rightarrow \infty$, $p/n \rightarrow c \in (0, \infty)$, for $a, b \in \mathbb{C}^p$ deterministic and \hat{u}_i eigenvector of $\lambda_i(\frac{1}{n} Y_p Y_p^*)$,

$$a^* \hat{u}_i \hat{u}_i^* b - \frac{1 - c\omega_i^{-2}}{1 + c\omega_i^{-1}} a^* u_i u_i^* b \cdot \mathbf{1}_{\omega_i > \sqrt{c}} \xrightarrow{\text{a.s.}} 0$$

In particular,

$$|\hat{u}_i^* u_i|^2 \xrightarrow{\text{a.s.}} \frac{1 - c\omega_i^{-2}}{1 + c\omega_i^{-1}} \cdot \mathbf{1}_{\omega_i > \sqrt{c}}.$$

Spiked Models

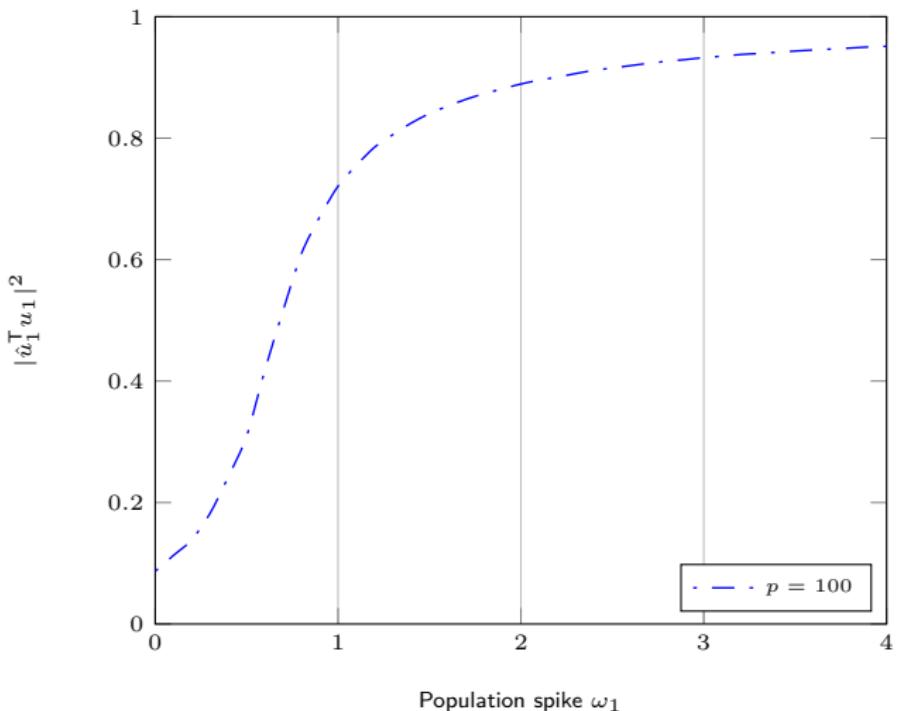


Figure: Simulated versus limiting $|\hat{u}_1^T u_1|^2$ for $Y_p = C_p^{\frac{1}{2}} X_p$, $C_p = I_p + \omega_1 u_1 u_1^T$, $p/n = 1/3$, varying ω_1 .

Spiked Models

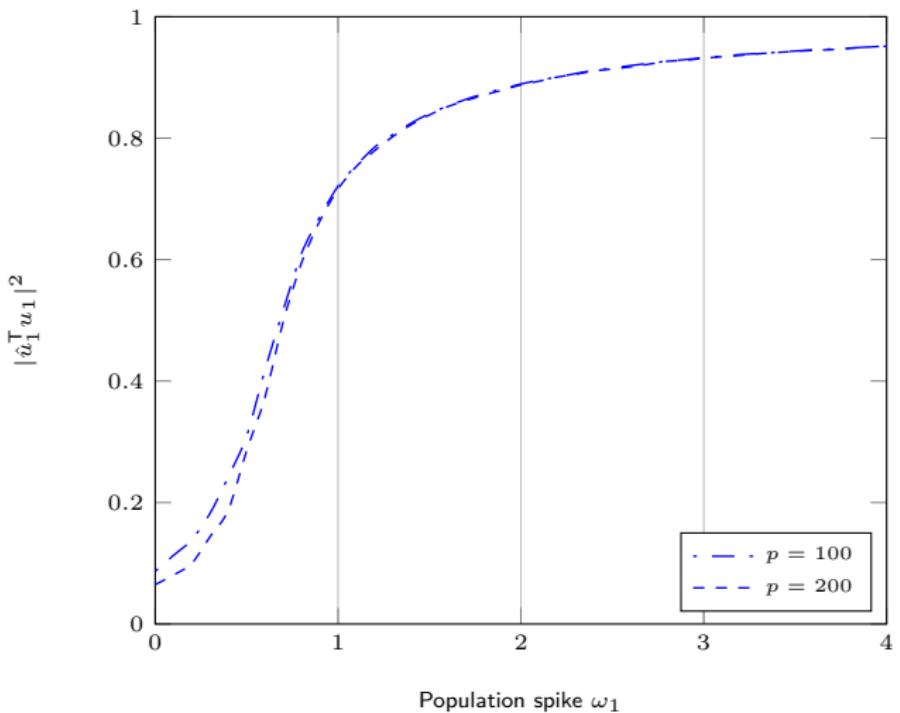


Figure: Simulated versus limiting $|\hat{u}_1^T u_1|^2$ for $Y_p = C_p^{\frac{1}{2}} X_p$, $C_p = I_p + \omega_1 u_1 u_1^T$, $p/n = 1/3$, varying ω_1 .

Spiked Models

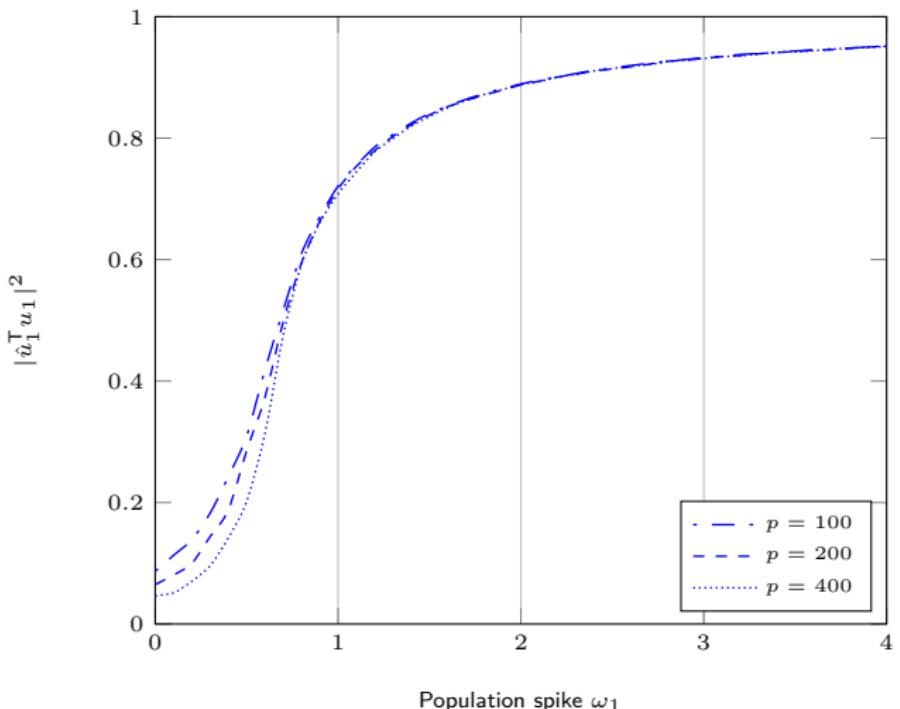


Figure: Simulated versus limiting $|\hat{u}_1^T u_1|^2$ for $Y_p = C_p^{\frac{1}{2}} X_p$, $C_p = I_p + \omega_1 u_1 u_1^T$, $p/n = 1/3$, varying ω_1 .

Spiked Models

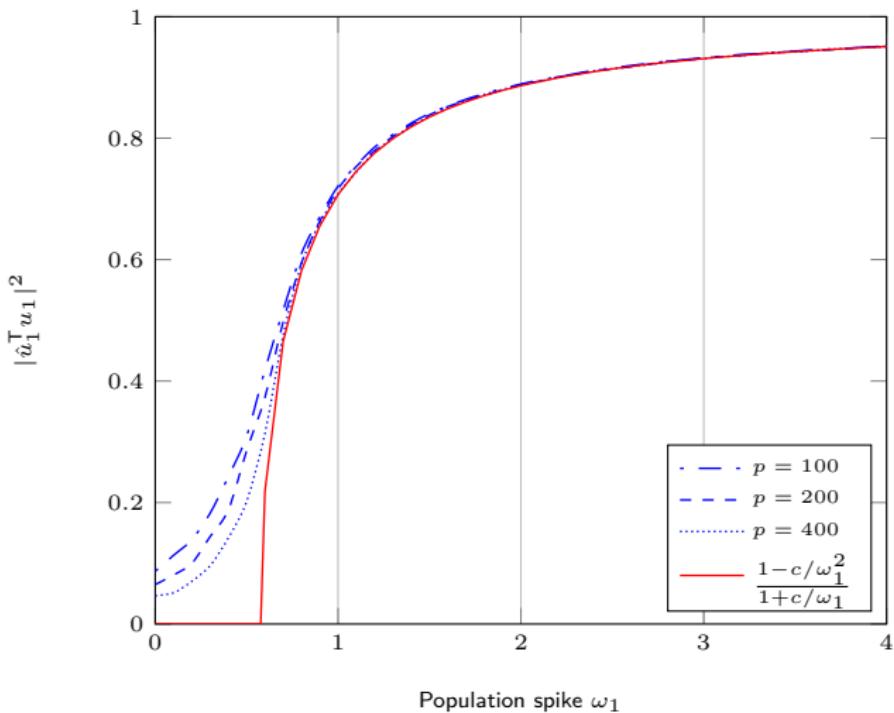


Figure: Simulated versus limiting $|\hat{u}_1^T u_1|^2$ for $Y_p = C_p^{\frac{1}{2}} X_p$, $C_p = I_p + \omega_1 u_1 u_1^T$, $p/n = 1/3$, varying ω_1 .

Other Spiked Models

Similar results for multiple matrix models:

- ▶ $Y_p = \frac{1}{n}(I + P)^{\frac{1}{2}} X_p X_p^* (I + P)^{\frac{1}{2}}$
- ▶ $Y_p = \frac{1}{n} X_p X_p^* + P$
- ▶ $Y_p = \frac{1}{n} X_p^* (I + P) X$
- ▶ $Y_p = \frac{1}{n} (X_p + P)^* (X_p + P)$
- ▶ etc.

Outline

Basics of Random Matrix Theory

Motivation: Large Sample Covariance Matrices
Spiked Models

Application to Machine Learning

Machine Learning is not “Simple Linear Statistics”:

Machine Learning is not “Simple Linear Statistics”:

- ▶ data are data... and are **not easily modeled**

Machine Learning is not “Simple Linear Statistics”:

- ▶ data are data... and are **not easily modeled**
- ▶ machine learning **algorithms involve non-linear functions**, difficult to analyze

Machine Learning is not “Simple Linear Statistics”:

- ▶ data are data... and are **not easily modeled**
- ▶ machine learning **algorithms involve non-linear functions**, difficult to analyze
- ▶ recent trends go towards highly complex computer-science oriented methods: **deep neural nets**.

Machine Learning is not “Simple Linear Statistics”:

- ▶ data are data... and are **not easily modeled**
- ▶ machine learning **algorithms involve non-linear functions**, difficult to analyze
- ▶ recent trends go towards highly complex computer-science oriented methods: **deep neural nets.**

What can we say about those?:

Machine Learning is not “Simple Linear Statistics”:

- ▶ data are data... and are **not easily modeled**
- ▶ machine learning **algorithms involve non-linear functions**, difficult to analyze
- ▶ recent trends go towards highly complex computer-science oriented methods: **deep neural nets.**

What can we say about those?:

- ▶ Much **more than we think**, and actually much more than has been said so far!

Machine Learning is not “Simple Linear Statistics”:

- ▶ data are data... and are **not easily modeled**
- ▶ machine learning **algorithms involve non-linear functions**, difficult to analyze
- ▶ recent trends go towards highly complex computer-science oriented methods: **deep neural nets.**

What can we say about those?:

- ▶ Much **more than we think**, and actually much more than has been said so far!
- ▶ **Key observation 1:** In “non-trivial” (not so) large dimensional settings, **machine learning intuitions break down!**

Machine Learning is not “Simple Linear Statistics”:

- ▶ data are data... and are **not easily modeled**
- ▶ machine learning **algorithms involve non-linear functions**, difficult to analyze
- ▶ recent trends go towards highly complex computer-science oriented methods: **deep neural nets.**

What can we say about those?:

- ▶ Much **more than we think**, and actually much more than has been said so far!
- ▶ **Key observation 1:** In “non-trivial” (not so) large dimensional settings, **machine learning intuitions break down!**
- ▶ **Key observation 2:** In these “non-trivial” settings, **RMT explains a lot of things and can improve algorithms!**

Machine Learning is not “Simple Linear Statistics”:

- ▶ data are data... and are **not easily modeled**
- ▶ machine learning **algorithms involve non-linear functions**, difficult to analyze
- ▶ recent trends go towards highly complex computer-science oriented methods: **deep neural nets.**

What can we say about those?:

- ▶ Much **more than we think**, and actually much more than has been said so far!
- ▶ **Key observation 1:** In “non-trivial” (not so) large dimensional settings, **machine learning intuitions break down!**
- ▶ **Key observation 2:** In these “non-trivial” settings, **RMT explains a lot of things and can improve algorithms!**
- ▶ **Key observation 3:** Universality goes a long way...: **RMT findings are compliant with real data observations!**

Takeaway Message 1

“RMT Explains Why Machine Learning Intuitions Collapse in Large Dimensions”

The curse of dimensionality and its consequences

Clustering setting in (not so) large n, p :

The curse of dimensionality and its consequences

Clustering setting in (not so) large n, p :

- ▶ GMM setting: $x_1^{(a)}, \dots, x_{n_a}^{(a)} \sim \mathcal{N}(\mu_a, C_a), a = 1, \dots, k$

The curse of dimensionality and its consequences

Clustering setting in (not so) large n, p :

- ▶ GMM setting: $x_1^{(a)}, \dots, x_{n_a}^{(a)} \sim \mathcal{N}(\mu_a, C_a)$, $a = 1, \dots, k$
- ▶ Non-trivial task:

$$\|\mu_a - \mu_b\| = O(1), \quad \text{tr}(C_a - C_b) = O(\sqrt{p}), \quad \text{tr}[(C_a - C_b)^2] = O(p)$$

The curse of dimensionality and its consequences

Clustering setting in (not so) large n, p :

- ▶ GMM setting: $x_1^{(a)}, \dots, x_{n_a}^{(a)} \sim \mathcal{N}(\mu_a, C_a)$, $a = 1, \dots, k$
- ▶ Non-trivial task:

$$\|\mu_a - \mu_b\| = O(1), \quad \text{tr}(C_a - C_b) = O(\sqrt{p}), \quad \text{tr}[(C_a - C_b)^2] = O(p)$$

(non-trivial because otherwise too easy or too hard)

The curse of dimensionality and its consequences

Clustering setting in (not so) large n, p :

- ▶ GMM setting: $x_1^{(a)}, \dots, x_{n_a}^{(a)} \sim \mathcal{N}(\mu_a, C_a)$, $a = 1, \dots, k$
- ▶ Non-trivial task:

$$\|\mu_a - \mu_b\| = O(1), \quad \text{tr}(C_a - C_b) = O(\sqrt{p}), \quad \text{tr}[(C_a - C_b)^2] = O(p)$$

(non-trivial because otherwise too easy or too hard)

Classical method: spectral clustering

The curse of dimensionality and its consequences

Clustering setting in (not so) large n, p :

- ▶ GMM setting: $x_1^{(a)}, \dots, x_{n_a}^{(a)} \sim \mathcal{N}(\mu_a, C_a)$, $a = 1, \dots, k$
- ▶ Non-trivial task:

$$\|\mu_a - \mu_b\| = O(1), \quad \text{tr}(C_a - C_b) = O(\sqrt{p}), \quad \text{tr}[(C_a - C_b)^2] = O(p)$$

(non-trivial because otherwise too easy or too hard)

Classical method: spectral clustering

- ▶ Extract and cluster the dominant eigenvectors of

$$K = \{\kappa(x_i, x_j)\}_{i,j=1}^n$$

The curse of dimensionality and its consequences

Clustering setting in (not so) large n, p :

- ▶ GMM setting: $x_1^{(a)}, \dots, x_{n_a}^{(a)} \sim \mathcal{N}(\mu_a, C_a)$, $a = 1, \dots, k$
- ▶ Non-trivial task:

$$\|\mu_a - \mu_b\| = O(1), \quad \text{tr}(C_a - C_b) = O(\sqrt{p}), \quad \text{tr}[(C_a - C_b)^2] = O(p)$$

(non-trivial because otherwise too easy or too hard)

Classical method: spectral clustering

- ▶ Extract and cluster the dominant eigenvectors of

$$K = \{\kappa(x_i, x_j)\}_{i,j=1}^n, \quad \kappa(x_i, x_j) = f\left(\frac{1}{p}\|x_i - x_j\|^2\right).$$

The curse of dimensionality and its consequences

Clustering setting in (not so) large n, p :

- ▶ GMM setting: $x_1^{(a)}, \dots, x_{n_a}^{(a)} \sim \mathcal{N}(\mu_a, C_a)$, $a = 1, \dots, k$
- ▶ Non-trivial task:

$$\|\mu_a - \mu_b\| = O(1), \quad \text{tr}(C_a - C_b) = O(\sqrt{p}), \quad \text{tr}[(C_a - C_b)^2] = O(p)$$

(non-trivial because otherwise too easy or too hard)

Classical method: spectral clustering

- ▶ Extract and cluster the dominant eigenvectors of

$$K = \{\kappa(x_i, x_j)\}_{i,j=1}^n, \quad \kappa(x_i, x_j) = f\left(\frac{1}{p}\|x_i - x_j\|^2\right).$$

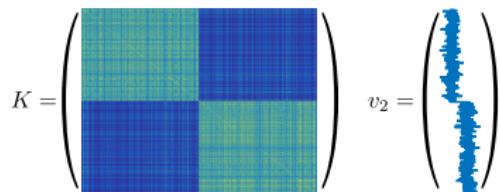
- ▶ Why? **Finite-dimensional intuition**

$$K = \left(\begin{array}{ccc|ccc} \kappa(x_i, x_j) & \kappa(x_i, x_j) \\ \hline & \gg 1 & \ll 1 & \ll 1 & \gg 1 & \ll 1 \\ \kappa(x_i, x_j) & \kappa(x_i, x_j) \\ \hline & \ll 1 & \gg 1 & \ll 1 & \ll 1 & \gg 1 \\ \kappa(x_i, x_j) & \gg 1 \end{array} \right) \begin{array}{c} \uparrow \mathcal{C}_1 \\ \downarrow \mathcal{C}_2 \\ \uparrow \mathcal{C}_3 \end{array}$$

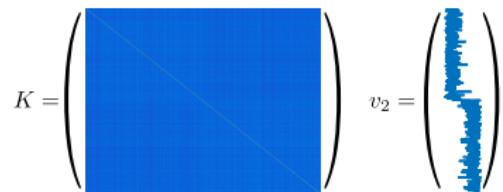
The curse of dimensionality and its consequences (2)

In reality, here is what happens...

Kernel $K_{ij} = \exp(-\frac{1}{2p}\|x_i - x_j\|^2)$ and second eigenvector v_2
 $(x_i \sim \mathcal{N}(\pm\mu, I_p), \mu = (2, 0, \dots, 0)^\top \in \mathbb{R}^p)$.



$$p = 4, n = 1000$$

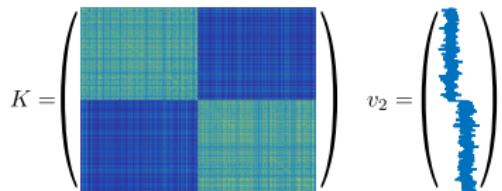


$$p = 400, n = 1000$$

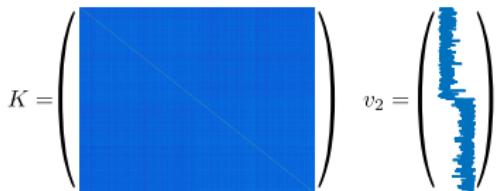
The curse of dimensionality and its consequences (2)

In reality, here is what happens...

Kernel $K_{ij} = \exp(-\frac{1}{2p}\|x_i - x_j\|^2)$ and second eigenvector v_2 ($x_i \sim \mathcal{N}(\pm\mu, I_p)$, $\mu = (2, 0, \dots, 0)^\top \in \mathbb{R}^p$).



$p = 4, n = 1000$



$p = 400, n = 1000$

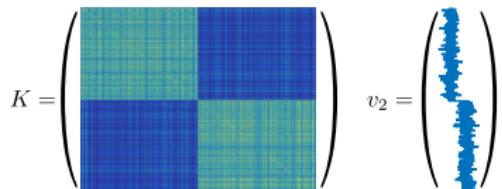
Key observation: Under growth rate assumptions,

$$\max_{1 \leq i \neq j \leq n} \left\{ \left| \frac{1}{p} \|x_i - x_j\|^2 - \tau \right| \right\} \xrightarrow{\text{a.s.}} 0, \quad \tau = \frac{2}{p} \sum_{i=1}^k \text{tr} \frac{n_a}{n} C_a.$$

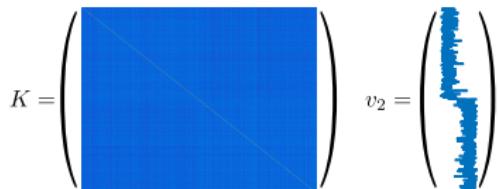
The curse of dimensionality and its consequences (2)

In reality, here is what happens...

Kernel $K_{ij} = \exp(-\frac{1}{2p}\|x_i - x_j\|^2)$ and second eigenvector v_2 ($x_i \sim \mathcal{N}(\pm\mu, I_p)$, $\mu = (2, 0, \dots, 0)^\top \in \mathbb{R}^p$).



$p = 4, n = 1000$



$p = 400, n = 1000$

Key observation: Under growth rate assumptions,

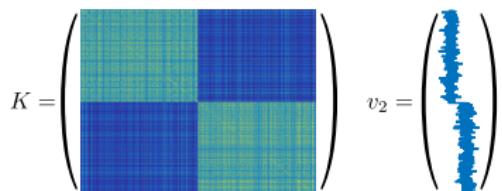
$$\max_{1 \leq i \neq j \leq n} \left\{ \left| \frac{1}{p} \|x_i - x_j\|^2 - \tau \right| \right\} \xrightarrow{\text{a.s.}} 0, \quad \tau = \frac{2}{p} \sum_{i=1}^k \text{tr} \frac{n_a}{n} C_a.$$

- ▶ this suggests $K \cong f(\tau)1_n 1_n^\top$!

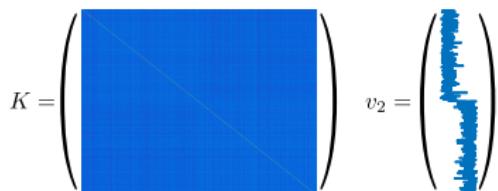
The curse of dimensionality and its consequences (2)

In reality, here is what happens...

Kernel $K_{ij} = \exp(-\frac{1}{2p}\|x_i - x_j\|^2)$ and second eigenvector v_2 ($x_i \sim \mathcal{N}(\pm\mu, I_p)$, $\mu = (2, 0, \dots, 0)^\top \in \mathbb{R}^p$).



$p = 4, n = 1000$



$p = 400, n = 1000$

Key observation: Under growth rate assumptions,

$$\max_{1 \leq i \neq j \leq n} \left\{ \left| \frac{1}{p} \|x_i - x_j\|^2 - \tau \right| \right\} \xrightarrow{\text{a.s.}} 0, \quad \tau = \frac{2}{p} \sum_{i=1}^k \text{tr} \frac{n_a}{n} C_a.$$

- ▶ this suggests $K \simeq f(\tau)1_n 1_n^\top$
 - ▶ more importantly, **in non-trivial settings, data are neither close, nor far!**

The curse of dimensionality and its consequences (3)

(Major) consequences:

The curse of dimensionality and its consequences (3)

(Major) consequences:

- ▶ Most machine learning intuitions collapse

The curse of dimensionality and its consequences (3)

(Major) consequences:

- ▶ Most **machine learning intuitions collapse**
- ▶ **But luckily**, concentration of distances allows for Taylor expansion, linearization...

The curse of dimensionality and its consequences (3)

(Major) consequences:

- ▶ Most **machine learning intuitions collapse**
- ▶ **But luckily**, concentration of distances allows for Taylor expansion, linearization...
- ▶ This is where **RMT kicks back in!**

The curse of dimensionality and its consequences (3)

(Major) consequences:

- ▶ Most machine learning intuitions collapse
- ▶ But luckily, concentration of distances allows for Taylor expansion, linearization...
- ▶ This is where RMT kicks back in!

Theorem ([C-Benaych'16] Asymptotic Kernel Behavior)

Under growth rate assumptions, as $p, n \rightarrow \infty$,

$$\|K - \hat{K}\| \xrightarrow{\text{a.s.}} 0, \quad \hat{K} \simeq \frac{1}{p} ZZ^T + \textcolor{red}{JAJ^T} + *$$

The curse of dimensionality and its consequences (3)

(Major) consequences:

- ▶ Most machine learning intuitions collapse
- ▶ But luckily, concentration of distances allows for Taylor expansion, linearization...
- ▶ This is where RMT kicks back in!

Theorem ([C-Benaych'16] Asymptotic Kernel Behavior)

Under growth rate assumptions, as $p, n \rightarrow \infty$,

$$\|K - \hat{K}\| \xrightarrow{\text{a.s.}} 0, \quad \hat{K} \simeq \frac{1}{p} ZZ^T + \textcolor{red}{J} \textcolor{blue}{A} \textcolor{red}{J}^T + *$$

with $\textcolor{red}{J} = [j_1, \dots, j_k] \in \mathbb{R}^{n \times k}$, $j_a = (0, 1_{n_a}, 0)^T$ (*the clusters!*)

The curse of dimensionality and its consequences (3)

(Major) consequences:

- ▶ Most machine learning intuitions collapse
- ▶ But luckily, concentration of distances allows for Taylor expansion, linearization...
- ▶ This is where RMT kicks back in!

Theorem ([C-Benaych'16] Asymptotic Kernel Behavior)

Under growth rate assumptions, as $p, n \rightarrow \infty$,

$$\|K - \hat{K}\| \xrightarrow{\text{a.s.}} 0, \quad \hat{K} \simeq \frac{1}{p} ZZ^T + \textcolor{red}{J} \textcolor{blue}{A} \textcolor{red}{J}^T + *$$

with $\textcolor{red}{J} = [j_1, \dots, j_k] \in \mathbb{R}^{n \times k}$, $j_a = (0, 1_{n_a}, 0)^T$ (the clusters!) and $A \in \mathbb{R}^{k \times k}$ function of:

- ▶ $f(\tau), f'(\tau), f''(\tau)$
- ▶ $\|\mu_a - \mu_b\|, \text{tr}(C_a - C_b), \text{tr}((C_a - C_b)^2)$, for $a, b \in \{1, \dots, k\}$.

The curse of dimensionality and its consequences (3)

(Major) consequences:

- ▶ Most machine learning intuitions collapse
- ▶ But luckily, concentration of distances allows for Taylor expansion, linearization...
- ▶ This is where RMT kicks back in!

Theorem ([C-Benaych'16] Asymptotic Kernel Behavior)

Under growth rate assumptions, as $p, n \rightarrow \infty$,

$$\|K - \hat{K}\| \xrightarrow{\text{a.s.}} 0, \quad \hat{K} \simeq \frac{1}{p} ZZ^T + \textcolor{red}{J} \textcolor{blue}{A} \textcolor{red}{J}^T + *$$

with $\textcolor{red}{J} = [j_1, \dots, j_k] \in \mathbb{R}^{n \times k}$, $j_a = (0, 1_{n_a}, 0)^T$ (*the clusters!*) and $A \in \mathbb{R}^{k \times k}$ function of:

- ▶ $f(\tau), f'(\tau), f''(\tau)$
 - ▶ $\|\mu_a - \mu_b\|, \text{tr}(C_a - C_b), \text{tr}((C_a - C_b)^2)$, for $a, b \in \{1, \dots, k\}$.
- ⇒ This is a spiked model! We can study it fully!

The curse of dimensionality and its consequences (3)

(Major) consequences:

- ▶ Most machine learning intuitions collapse
- ▶ But luckily, concentration of distances allows for Taylor expansion, linearization...
- ▶ This is where RMT kicks back in!

Theorem ([C-Benaych'16] Asymptotic Kernel Behavior)

Under growth rate assumptions, as $p, n \rightarrow \infty$,

$$\|K - \hat{K}\| \xrightarrow{\text{a.s.}} 0, \quad \hat{K} \simeq \frac{1}{p} ZZ^T + \textcolor{red}{J} \textcolor{blue}{A} \textcolor{red}{J}^T + *$$

with $\textcolor{red}{J} = [j_1, \dots, j_k] \in \mathbb{R}^{n \times k}$, $j_a = (0, 1_{n_a}, 0)^T$ (*the clusters!*) and $A \in \mathbb{R}^{k \times k}$ function of:

- ▶ $f(\tau), f'(\tau), f''(\tau)$
 - ▶ $\|\mu_a - \mu_b\|, \text{tr}(C_a - C_b), \text{tr}((C_a - C_b)^2)$, for $a, b \in \{1, \dots, k\}$.
- ⇒ This is a spiked model! We can study it fully!

RMT can explain tools ML engineers use everyday.

Theoretical Findings versus MNIST

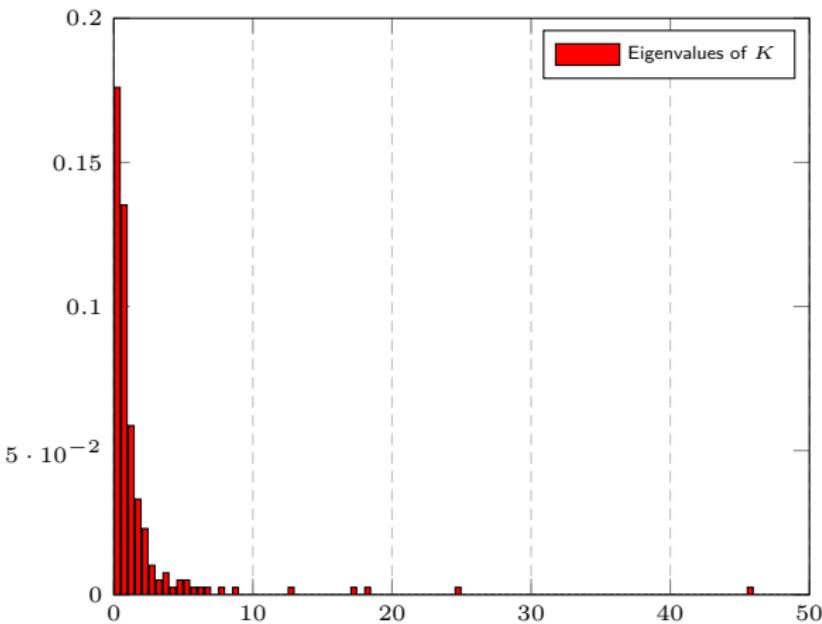


Figure: Eigenvalues of K (red) and (equivalent Gaussian model) \hat{K} (white), MNIST data, $p = 784$, $n = 192$.

Theoretical Findings versus MNIST

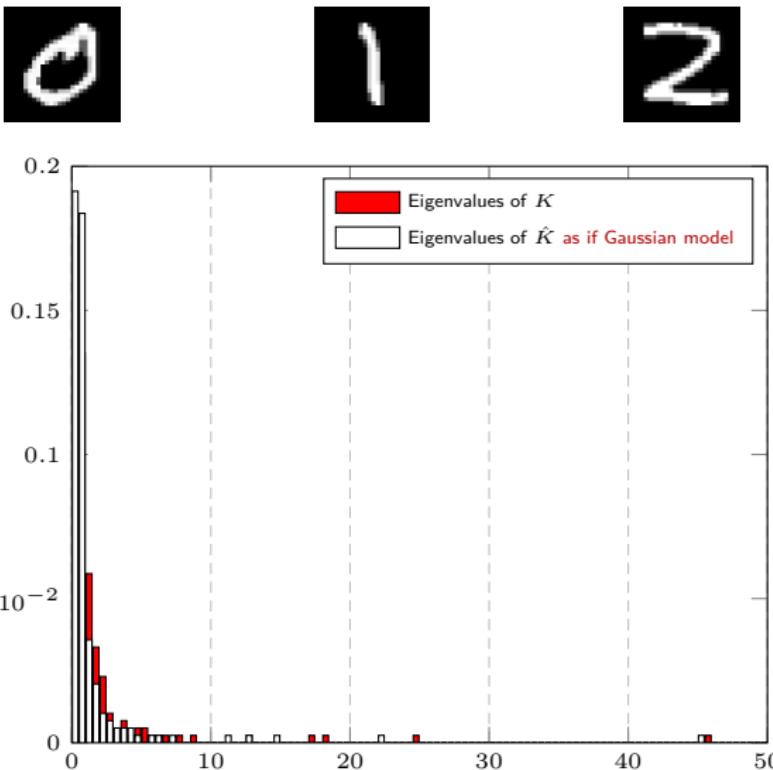


Figure: Eigenvalues of K (red) and (equivalent Gaussian model) \hat{K} (white), MNIST data, $p = 784$, $n = 192$.

Theoretical Findings versus MNIST

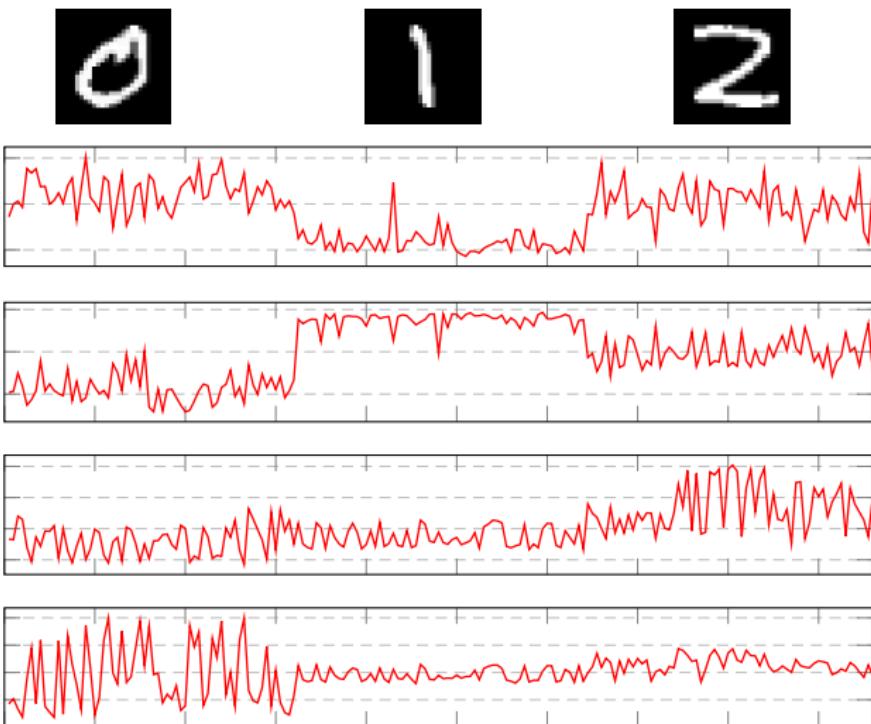


Figure: Leading four eigenvectors of K for MNIST data (**red**) and theoretical findings (**blue**).

Theoretical Findings versus MNIST

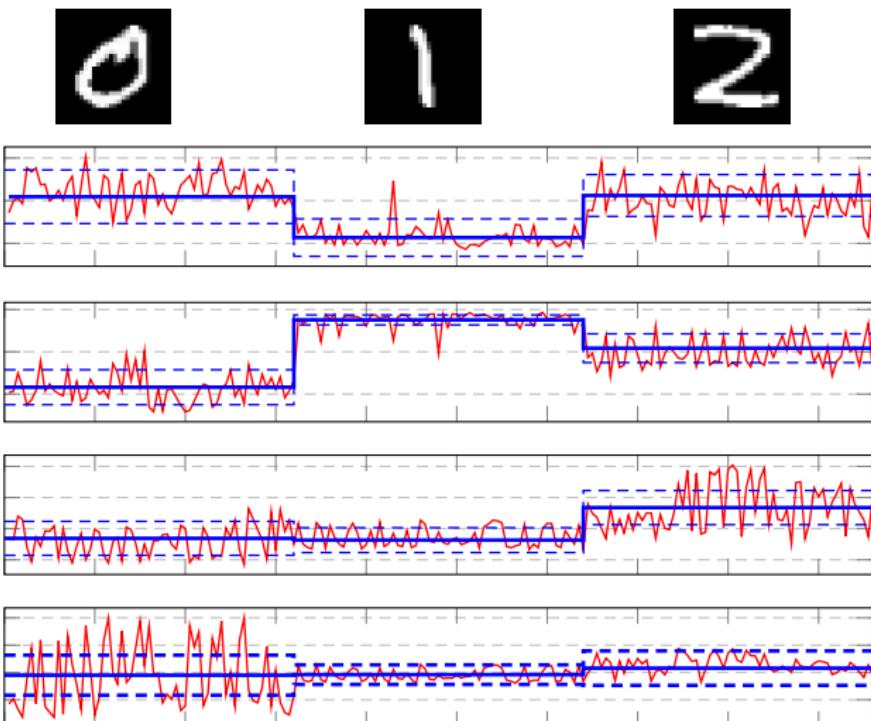
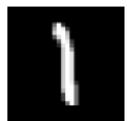


Figure: Leading four eigenvectors of K for MNIST data (red) and theoretical findings (blue).

Theoretical Findings versus MNIST



Eigenvector 2/Eigenvector 1



Eigenvector 3/Eigenvector 2

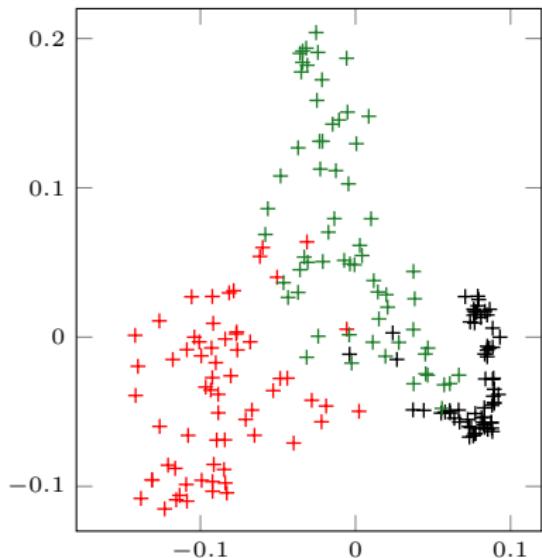
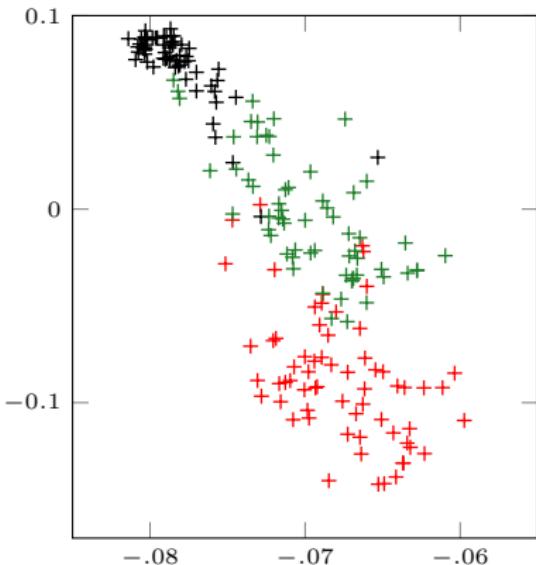


Figure: 2D representation of eigenvectors of K , for the MNIST dataset. Theoretical means and 1- and 2-standard deviations in **blue**. Class 1 in **red**, Class 2 in **black**, Class 3 in **green**.

Theoretical Findings versus MNIST



Eigenvector 2/Eigenvector 1



Eigenvector 3/Eigenvector 2

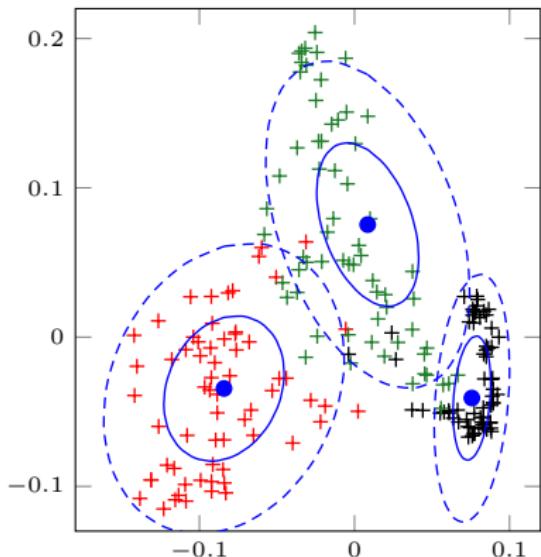
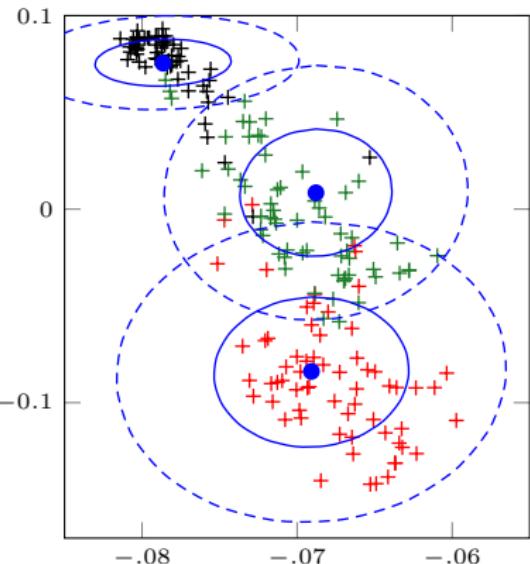


Figure: 2D representation of eigenvectors of K , for the MNIST dataset. Theoretical means and 1- and 2-standard deviations in **blue**. Class 1 in **red**, Class 2 in **black**, Class 3 in **green**.

Takeaway Message 2

“RMT Reassesses and Improves Data Processing”

Improving Kernel Spectral Clustering

Thanks to [C-Benaych'16]: Possibility to improve kernels:

Improving Kernel Spectral Clustering

Thanks to [C-Benaych'16]: Possibility to improve kernels:

- ▶ by “focusing kernels” on best discriminative statistics: tune $f'(\tau), f''(\tau)$

Improving Kernel Spectral Clustering

Thanks to [C-Benaych'16]: Possibility to improve kernels:

- ▶ by “focusing kernels” on best discriminative statistics: tune $f'(\tau), f''(\tau)$
- ▶ by “killing” non discriminative feature directions.

Improving Kernel Spectral Clustering

Thanks to [C-Benaych'16]: Possibility to improve kernels:

- ▶ by “focusing kernels” on best discriminative statistics: tune $f'(\tau), f''(\tau)$
- ▶ by “killing” non discriminative feature directions.

Example: Covariance-based discrimination, kernel $f(t) = \exp(-\frac{1}{2}t)$ versus $f(t) = (t - \tau)^2$ (think about the surprising kernel shape!)

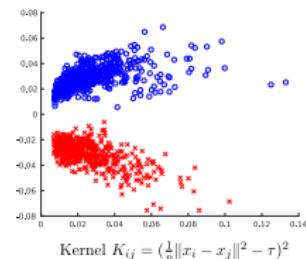
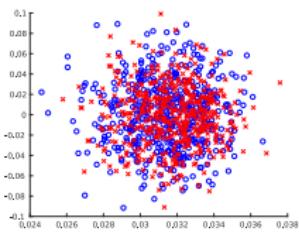
Improving Kernel Spectral Clustering

Thanks to [C-Benaych'16]: Possibility to improve kernels:

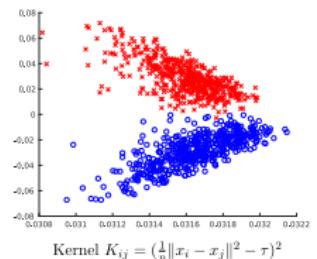
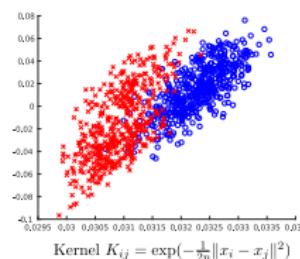
- ▶ by “focusing kernels” on best discriminative statistics: tune $f'(\tau), f''(\tau)$
- ▶ by “killing” non discriminative feature directions.

Example: Covariance-based discrimination, kernel $f(t) = \exp(-\frac{1}{2}t)$ versus $f(t) = (t - \tau)^2$ (think about the surprising kernel shape!)

— Gaussian vectors $\mathcal{N}(0, C_1)$ vs. $\mathcal{N}(0, C_2)$ —————



— Real EEG data —————



Another, more striking, example: Semi-supervised Learning

Semi-supervised learning: a great idea that never worked!

Another, more striking, example: Semi-supervised Learning

Semi-supervised learning: a great idea that never worked!

► **Setting:** assume now

- $x_1^{(a)}, \dots, x_{n_a, [l]}^{(a)}$ already labelled (few),
- $x_{n_a, [l] + 1}^{(a)}, \dots, x_{n_a}^{(a)}$ unlabelled (a lot).

Another, more striking, example: Semi-supervised Learning

Semi-supervised learning: a great idea that never worked!

► **Setting:** assume now

- $x_1^{(a)}, \dots, x_{n_a, [l]}^{(a)}$ already labelled (few),
- $x_{n_a, [l] + 1}^{(a)}, \dots, x_{n_a}^{(a)}$ unlabelled (a lot).

► **Machine Learning original idea:** find “scores” F_{ia} for x_i to belong to class a

$$F = \operatorname{argmin}_{F \in \mathbb{R}^{n \times k}} \sum_{a=1}^k K_{ij} \left(F_{ia} - F_{jb} \right)^2, \quad F_{ia}^{[l]} = \delta_{\{x_i \in \mathcal{C}_a\}}.$$

Another, more striking, example: Semi-supervised Learning

Semi-supervised learning: a great idea that never worked!

- ▶ **Setting:** assume now
 - ▶ $x_1^{(a)}, \dots, x_{n_a, [l]}^{(a)}$ already labelled (few),
 - ▶ $x_{n_a, [l]+1}^{(a)}, \dots, x_{n_a}^{(a)}$ unlabelled (a lot).
- ▶ **Machine Learning original idea:** find “scores” F_{ia} for x_i to belong to class a

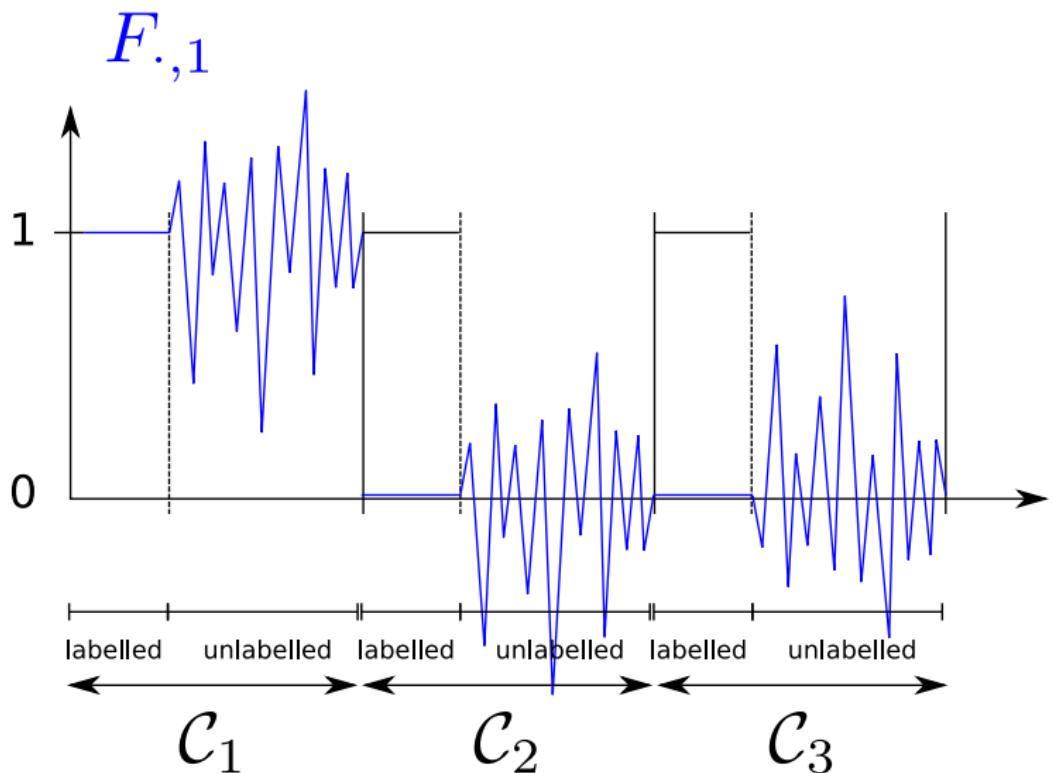
$$F = \operatorname{argmin}_{F \in \mathbb{R}^{n \times k}} \sum_{a=1}^k K_{ij} \left(F_{ia} - F_{jb} \right)^2, \quad F_{ia}^{[l]} = \delta_{\{x_i \in \mathcal{C}_a\}}.$$

- ▶ **Explicit solution:**

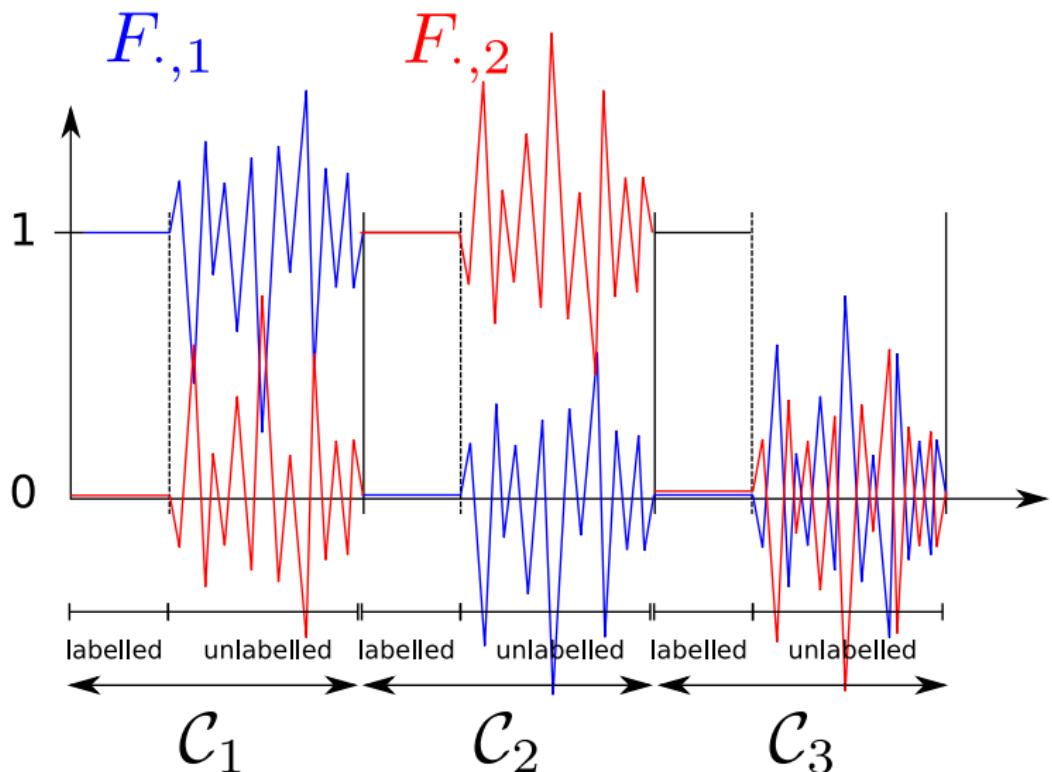
$$F^{[u]} = \left(I_{n_{[u]}} - D_{[u]}^{-1} K_{[uu]} \right)^{-1} D_{[u]}^{-1} K_{[ul]} F^{[l]}$$

where $D = \operatorname{diag}(K1_n)$ (degree matrix) and $[ul]$, $[uu]$, ... blocks of labeled/unlabeled data.

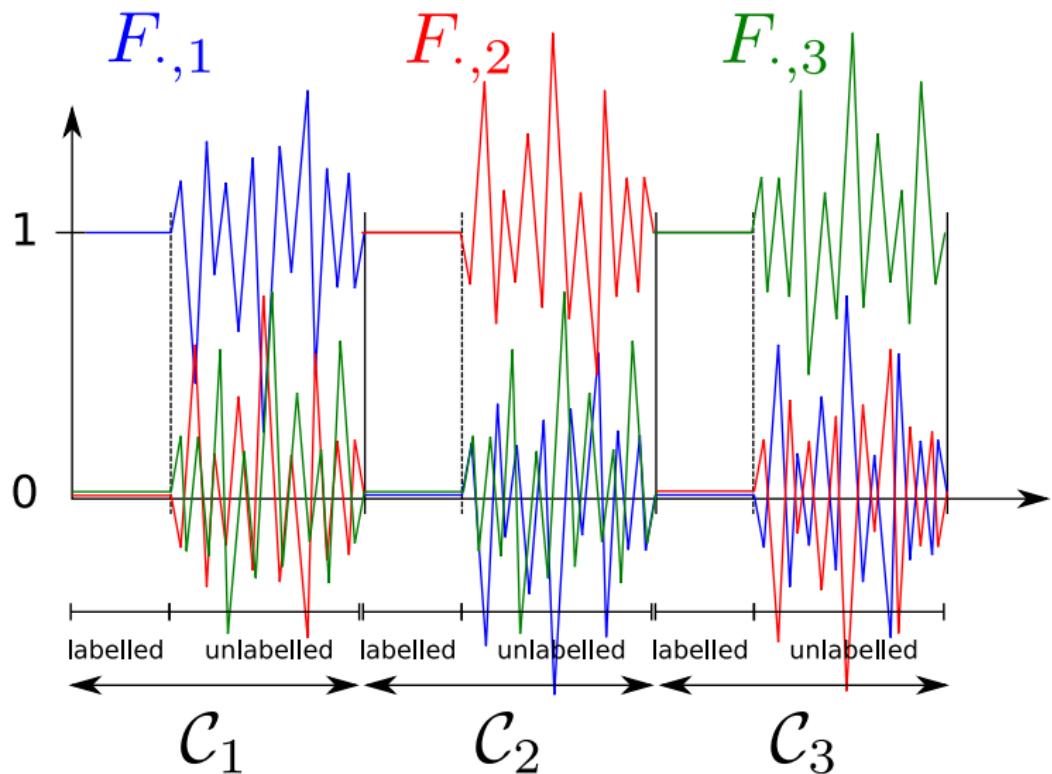
The finite-dimensional intuition: What we expect



The finite-dimensional intuition: What we expect



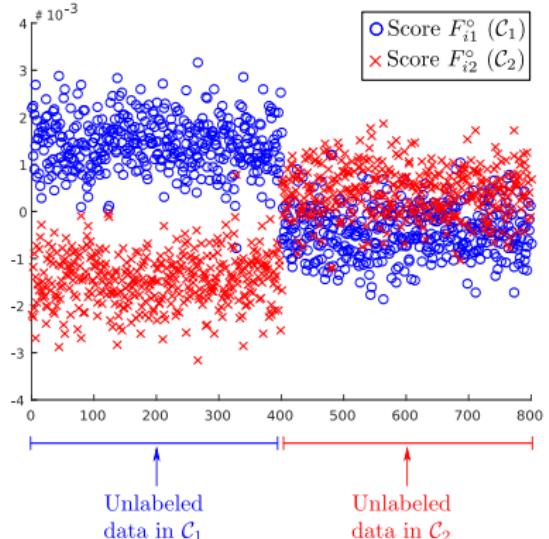
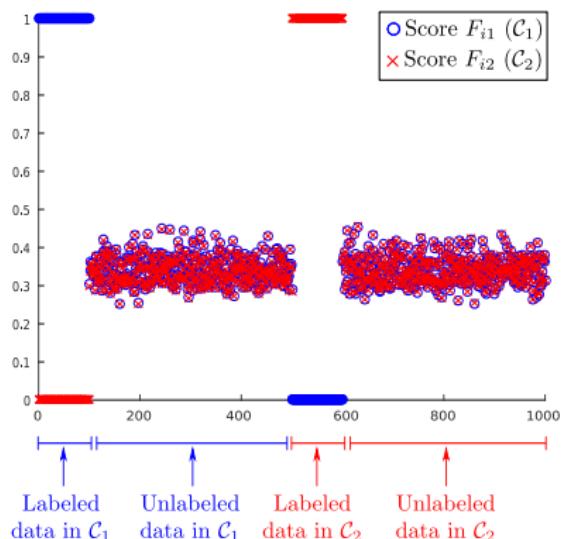
The finite-dimensional intuition: What we expect



The reality: What we see!

Setting. $p = 400$, $n = 1000$, $x_i \sim \mathcal{N}(\pm\mu, I_p)$. Kernel $K_{ij} = \exp(-\frac{1}{2p}\|x_i - x_j\|^2)$.

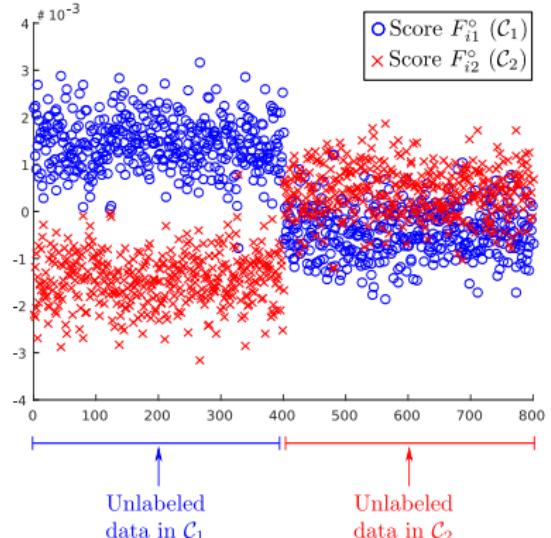
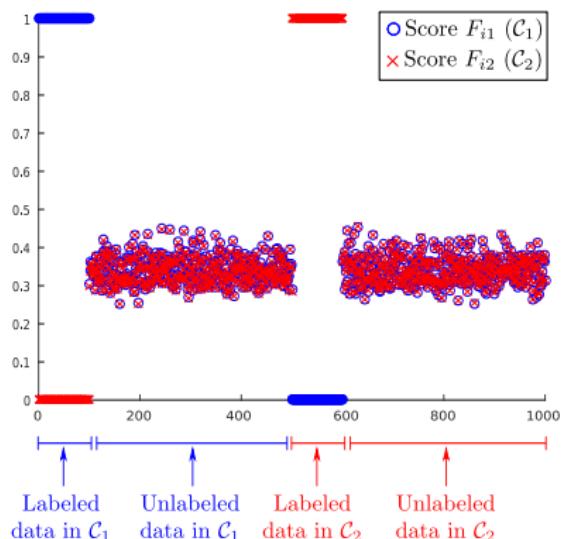
Display. Scores F_{ik} (left) and $F_{ik}^\circ = F_{ik} - \frac{1}{2}(F_{i1} + F_{i2})$ (right).



The reality: What we see!

Setting. $p = 400$, $n = 1000$, $x_i \sim \mathcal{N}(\pm\mu, I_p)$. Kernel $K_{ij} = \exp(-\frac{1}{2p}\|x_i - x_j\|^2)$.

Display. Scores F_{ik} (left) and $F_{ik}^\circ = F_{ik} - \frac{1}{2}(F_{i1} + F_{i2})$ (right).



☞ Score are almost all identical... and do not follow the labelled data!

MNIST Data Example

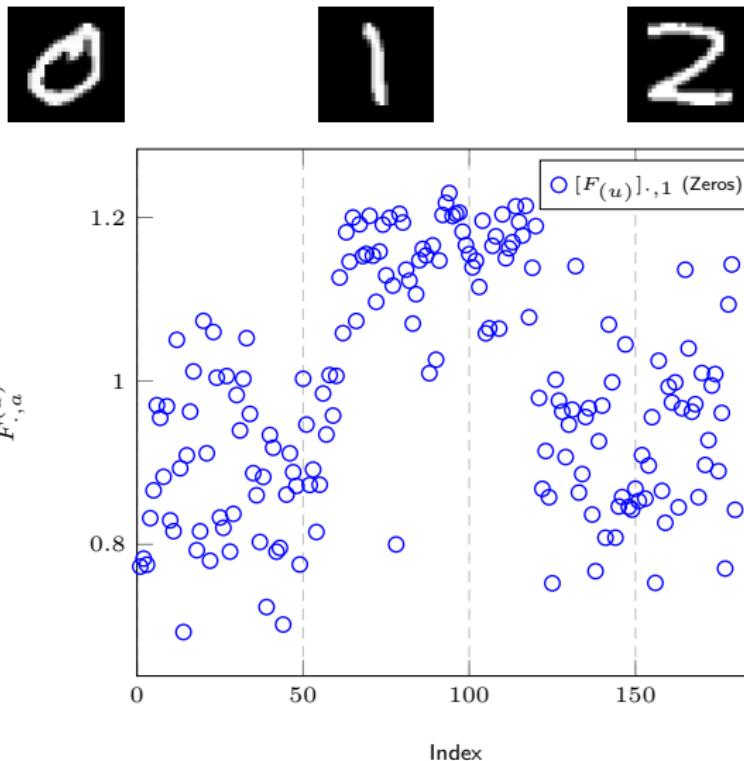


Figure: Vectors $[F^{(u)}]_{\cdot,a}$, $a = 1, 2, 3$, for 3-class MNIST data (zeros, ones, twos), $n = 192$, $p = 784$, $n_l/n = 1/16$, Gaussian kernel.

MNIST Data Example

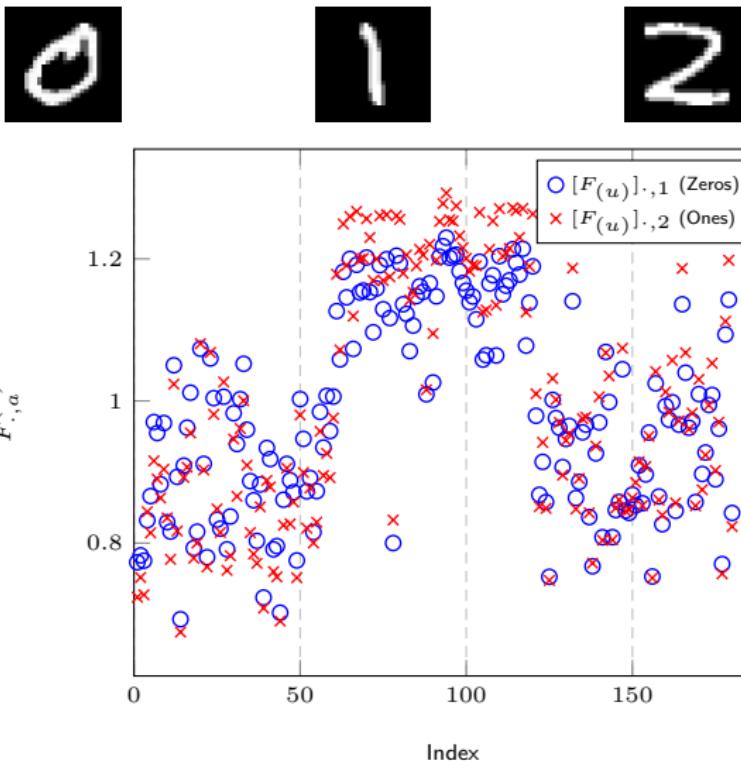


Figure: Vectors $[F^{(u)}]_{\cdot,a}$, $a = 1, 2, 3$, for 3-class MNIST data (zeros, ones, twos), $n = 192$, $p = 784$, $n_l/n = 1/16$, Gaussian kernel.

MNIST Data Example

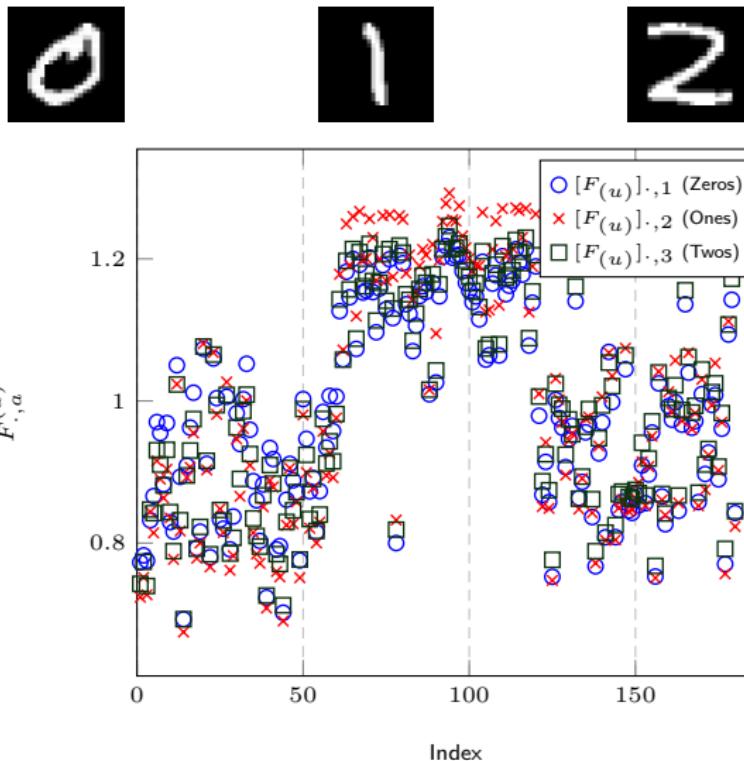


Figure: Vectors $[F^{(u)}]_{\cdot,a}$, $a = 1, 2, 3$, for 3-class MNIST data (zeros, ones, twos), $n = 192$, $p = 784$, $n_l/n = 1/16$, Gaussian kernel.

Exploiting RMT to resurrect SSL

Consequences of the finite-dimensional “mismatch”

Exploiting RMT to resurrect SSL

Consequences of the finite-dimensional “mismatch”

- ▶ A priori, **the algorithm should not work**

Exploiting RMT to resurrect SSL

Consequences of the finite-dimensional “mismatch”

- ▶ A priori, **the algorithm should not work**
- ▶ Indeed “in general” it does not!

Exploiting RMT to resurrect SSL

Consequences of the finite-dimensional “mismatch”

- ▶ A priori, **the algorithm should not work**
- ▶ Indeed “in general” it does not!
- ▶ But, luckily, after some (not clearly motivated) renormalization, it works again...

Exploiting RMT to resurrect SSL

Consequences of the finite-dimensional “mismatch”

- ▶ A priori, **the algorithm should not work**
- ▶ Indeed “in general” it does not!
- ▶ But, luckily, after some (not clearly motivated) renormalization, it works again...
- ▶ **BUT** it does not use efficiently unlabelled data!

Exploiting RMT to resurrect SSL

Consequences of the finite-dimensional “mismatch”

- ▶ A priori, **the algorithm should not work**
- ▶ Indeed “in general” it does not!
- ▶ But, luckily, after some (not clearly motivated) renormalization, it works again...
- ▶ **BUT** it does not use efficiently unlabelled data!

Chapelle, Schölkopf, Zien, “**Semi-Supervised Learning**”, Chapter 4, 2009.

Our concern is this: it is frequently the case that we would be better off just discarding the unlabeled data and employing a supervised method, rather than taking a semi-supervised route. Thus we worry about the embarrassing situation where the addition of unlabeled data degrades the performance of a classifier.

Exploiting RMT to resurrect SSL

Consequences of the finite-dimensional “mismatch”

- ▶ A priori, **the algorithm should not work**
- ▶ Indeed “in general” it does not!
- ▶ But, luckily, after some (not clearly motivated) renormalization, it works again...
- ▶ **BUT** it does not use efficiently unlabelled data!

Chapelle, Schölkopf, Zien, “**Semi-Supervised Learning**”, Chapter 4, 2009.

Our concern is this: it is frequently the case that we would be better off just discarding the unlabeled data and employing a supervised method, rather than taking a semi-supervised route. Thus we worry about the embarrassing situation where the addition of unlabeled data degrades the performance of a classifier.

What RMT can do about it

Exploiting RMT to resurrect SSL

Consequences of the finite-dimensional “mismatch”

- ▶ A priori, **the algorithm should not work**
- ▶ Indeed “in general” it does not!
- ▶ But, luckily, after some (not clearly motivated) renormalization, it works again...
- ▶ **BUT** it does not use efficiently unlabelled data!

Chapelle, Schölkopf, Zien, “**Semi-Supervised Learning**”, Chapter 4, 2009.

Our concern is this: it is frequently the case that we would be better off just discarding the unlabeled data and employing a supervised method, rather than taking a semi-supervised route. Thus we worry about the embarrassing situation where the addition of unlabeled data degrades the performance of a classifier.

What RMT can do about it

- ▶ Asymptotic performance analysis: **clear understanding of what we see!**

Exploiting RMT to resurrect SSL

Consequences of the finite-dimensional “mismatch”

- ▶ A priori, **the algorithm should not work**
- ▶ Indeed “in general” it does not!
- ▶ But, luckily, after some (not clearly motivated) renormalization, it works again...
- ▶ **BUT** it does not use efficiently unlabelled data!

Chapelle, Schölkopf, Zien, “**Semi-Supervised Learning**”, Chapter 4, 2009.

Our concern is this: it is frequently the case that we would be better off just discarding the unlabeled data and employing a supervised method, rather than taking a semi-supervised route. Thus we worry about the embarrassing situation where the addition of unlabeled data degrades the performance of a classifier.

What RMT can do about it

- ▶ Asymptotic performance analysis: **clear understanding of what we see!**
- ▶ Update the algorithm and **provably improve unlabelled data use.**

Asymptotic Performance Analysis

Theorem (**[Mai,C'18]**) Asymptotic Performance of SSL)

For $x_i \in \mathcal{C}_b$ unlabelled, score vector $F_{i,\cdot} \in \mathbb{R}^k$ satisfies:

$$F_{i,\cdot} - G_b \rightarrow 0, \quad G_b \sim \mathcal{N}(m_b, \Sigma_b)$$

with $m_b \in \mathbb{R}^k$, $\Sigma_b \in \mathbb{R}^{k \times k}$ function of $f(\tau), f'(\tau), f''(\tau), \mu_1, \dots, \mu_k, C_1, \dots, C_k$.

Asymptotic Performance Analysis

Theorem ([Mai,C'18] Asymptotic Performance of SSL)

For $x_i \in \mathcal{C}_b$ unlabelled, score vector $F_{i,\cdot} \in \mathbb{R}^k$ satisfies:

$$F_{i,\cdot} - G_b \rightarrow 0, \quad G_b \sim \mathcal{N}(m_b, \Sigma_b)$$

with $m_b \in \mathbb{R}^k$, $\Sigma_b \in \mathbb{R}^{k \times k}$ function of $f(\tau), f'(\tau), f''(\tau), \mu_1, \dots, \mu_k, C_1, \dots, C_k$.

Most importantly: m_b, Σ_b independent of n_u (number of unlabelled data).

Asymptotic Performance Analysis

Theorem ([Mai,C'18] Asymptotic Performance of SSL)

For $x_i \in \mathcal{C}_b$ unlabelled, score vector $F_{i,\cdot} \in \mathbb{R}^k$ satisfies:

$$F_{i,\cdot} - G_b \rightarrow 0, \quad G_b \sim \mathcal{N}(m_b, \Sigma_b)$$

with $m_b \in \mathbb{R}^k$, $\Sigma_b \in \mathbb{R}^{k \times k}$ function of $f(\tau), f'(\tau), f''(\tau), \mu_1, \dots, \mu_k, C_1, \dots, C_k$.

Most importantly: m_b, Σ_b independent of n_u (number of unlabelled data).

Solution: From RMT calculus (but not from ML intuition!), solution is to replace K by

$$\tilde{K} \equiv PKP, \quad P = I_n - \frac{1}{n} \mathbf{1}_n \mathbf{1}_n^\top.$$

Asymptotic Performance Analysis

Theorem ([Mai,C'18] Asymptotic Performance of SSL)

For $x_i \in \mathcal{C}_b$ unlabelled, score vector $F_{i,\cdot} \in \mathbb{R}^k$ satisfies:

$$F_{i,\cdot} - G_b \rightarrow 0, \quad G_b \sim \mathcal{N}(m_b, \Sigma_b)$$

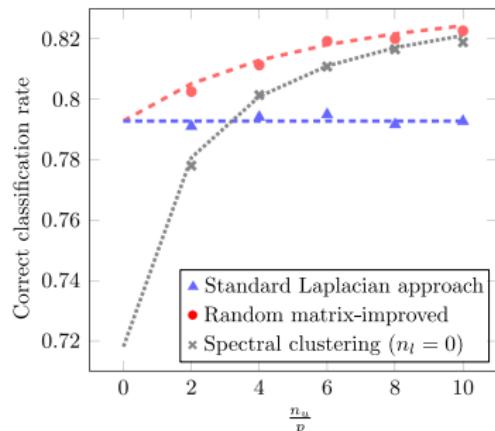
with $m_b \in \mathbb{R}^k$, $\Sigma_b \in \mathbb{R}^{k \times k}$ function of $f(\tau), f'(\tau), f''(\tau), \mu_1, \dots, \mu_k, C_1, \dots, C_k$.

Most importantly: m_b, Σ_b independent of n_u (number of unlabelled data).

Solution: From RMT calculus (but not from ML intuition!), solution is to replace K by

$$\tilde{K} \equiv PKP, \quad P = I_n - \frac{1}{n} \mathbf{1}_n \mathbf{1}_n^\top.$$

Performances:



Experimental evidence: MNIST



Digits	(0,8)	(2,7)	(6,9)
$n_u = 100$			
Centered kernel (RMT)	89.5±3.6	89.5±3.4	85.3±5.9
Iterated centered kernel (RMT)	89.5±3.6	89.5±3.4	85.3±5.9
Laplacian	75.5±5.6	74.2±5.8	70.0±5.5
Iterated Laplacian	87.2±4.7	86.0±5.2	81.4±6.8
Manifold	88.0±4.7	88.4±3.9	82.8±6.5
$n_u = 1000$			
Centered kernel (RMT)	92.2±0.9	92.5±0.8	92.6±1.6
Iterated centered kernel (RMT)	92.3±0.9	92.5± 0.8	92.9±1.4
Laplacian	65.6±4.1	74.4±4.0	69.5±3.7
Iterated Laplacian	92.2±0.9	92.4±0.9	92.0±1.6
Manifold	91.1±1.7	91.4±1.9	91.4±2.0

Table: Comparison of classification accuracy (%) on MNIST datasets with $n_l = 10$. Computed over 1000 random iterations for $n_u = 100$ and 100 for $n_u = 1000$.

Experimental evidence: Traffic signs (HOG features)



Class ID	(2,7)	(9,10)	(11,18)
$n_u = 100$			
Centered kernel (RMT)	79.0±10.4	77.5±9.2	78.5±7.1
Iterated centered kernel (RMT)	85.3±5.9	89.2±5.6	90.1±6.7
Laplacian	73.8±9.8	77.3±9.5	78.6±7.2
Iterated Laplacian	83.7±7.2	88.0±6.8	87.1±8.8
Manifold	77.6±8.9	81.4±10.4	82.3±10.8
$n_u = 1000$			
Centered kernel (RMT)	83.6±2.4	84.6±2.4	88.7±9.4
Iterated centered kernel (RMT)	84.8±3.8	88.0±5.5	96.4±3.0
Laplacian	72.7±4.2	88.9±5.7	95.8±3.2
Iterated Laplacian	83.0±5.5	88.2±6.0	92.7±6.1
Manifold	77.7±5.8	85.0±9.0	90.6±8.1

Table: Comparison of classification accuracy (%) on German Traffic Sign datasets with $n_l = 10$. Computed over 1000 random iterations for $n_u = 100$ and 100 for $n_u = 1000$.

Takeaway Message 3

“RMT Also Grasps ‘Real Data’ Processing”

From i.i.d. to concentrated random vectors

Current Problem. Data models based on **vectors of i.i.d. entries** (or even only Gaussian).

From i.i.d. to concentrated random vectors

Current Problem. Data models based on **vectors of i.i.d. entries** (or even only Gaussian).

Good news. In RMT, exploitation of time **and** feature dimensions brings **universality!**, i.e., only first moments matter irrespective of distribution.

From i.i.d. to concentrated random vectors

Current Problem. Data models based on **vectors of i.i.d. entries** (or even only Gaussian).

Good news. In RMT, exploitation of time **and** feature dimensions brings **universality!**, i.e., only first moments matter irrespective of distribution.

The Solution?. **Concentrated random vectors go a long way beyond!**

From i.i.d. to concentrated random vectors

Current Problem. Data models based on **vectors of i.i.d. entries** (or even only Gaussian).

Good news. In RMT, exploitation of time **and** feature dimensions brings **universality!**, i.e., only first moments matter irrespective of distribution.

The Solution?. **Concentrated random vectors go a long way beyond!**

Definition (Concentrated Random Vector)

$x \in \mathbb{R}^p$ is a concentrated random vector if, for all Lipschitz $f : \mathbb{R}^p \rightarrow \mathbb{R}$, there exists $m_f \in \mathbb{R}$, such that

$$P(|f(x) - m_f| > \varepsilon) \leq e^{-g(\varepsilon)}, \quad g \text{ increasing function.}$$

From i.i.d. to concentrated random vectors

Current Problem. Data models based on **vectors of i.i.d. entries** (or even only Gaussian).

Good news. In RMT, exploitation of time **and** feature dimensions brings **universality!**, i.e., only first moments matter irrespective of distribution.

The Solution?. **Concentrated random vectors go a long way beyond!**

Definition (Concentrated Random Vector)

$x \in \mathbb{R}^p$ is a concentrated random vector if, for all Lipschitz $f : \mathbb{R}^p \rightarrow \mathbb{R}$, there exists $m_f \in \mathbb{R}$, such that

$$P(|f(x) - m_f| > \varepsilon) \leq e^{-g(\varepsilon)}, \quad g \text{ increasing function.}$$

Theorem ([Louart, C'18] [Seddik, C'19] Kernel Universality)

For $x_i \sim \mathcal{L}(\mu_a, C_a)$ concentrated random vector, under the conditions of [C-Benaych'16],

$$\|K - \hat{K}\| \xrightarrow{\text{a.s.}} 0, \quad K\hat{K} \simeq \frac{1}{p} ZZ^\top + JAJ^\top + *$$

with A only dependent on $f(\tau), f'(\tau), f''(\tau), \mu_1, \dots, \mu_k, C_1, \dots, C_k$.

From i.i.d. to concentrated random vectors

Current Problem. Data models based on **vectors of i.i.d. entries** (or even only Gaussian).

Good news. In RMT, exploitation of time **and** feature dimensions brings **universality!**, i.e., only first moments matter irrespective of distribution.

The Solution?. **Concentrated random vectors go a long way beyond!**

Definition (Concentrated Random Vector)

$x \in \mathbb{R}^p$ is a concentrated random vector if, for all Lipschitz $f : \mathbb{R}^p \rightarrow \mathbb{R}$, there exists $m_f \in \mathbb{R}$, such that

$$P(|f(x) - m_f| > \varepsilon) \leq e^{-g(\varepsilon)}, \quad g \text{ increasing function.}$$

Theorem ([Louart, C'18] [Seddik, C'19] Kernel Universality)

For $x_i \sim \mathcal{L}(\mu_a, C_a)$ concentrated random vector, under the conditions of [C-Benaych'16],

$$\|K - \hat{K}\| \xrightarrow{\text{a.s.}} 0, \quad K\hat{K} \simeq \frac{1}{p} ZZ^\top + JAJ^\top + *$$

with A only dependent on $f(\tau), f'(\tau), f''(\tau), \mu_1, \dots, \mu_k, C_1, \dots, C_k$.

☞ Same result as [C-Benaych'16]... Universality of first two moments!

Ok...so what?

Ok...so what?

Key Finding. Real images are “almost” concentrated random vectors!

Ok...so what?

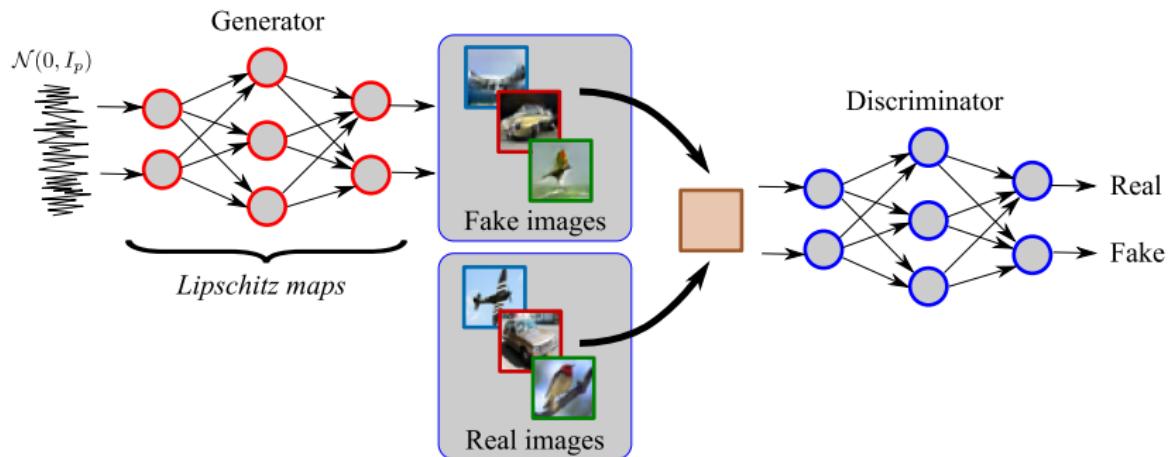
Key Finding. Real images are “almost” concentrated random vectors!

Example: GAN-generated images are concentrated random vectors!

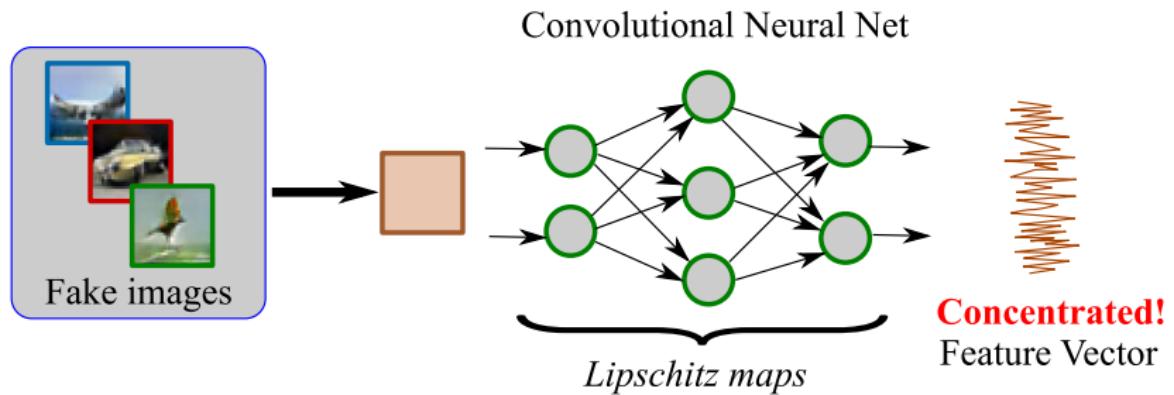
Ok... so what?

Key Finding. Real images are “almost” concentrated random vectors!

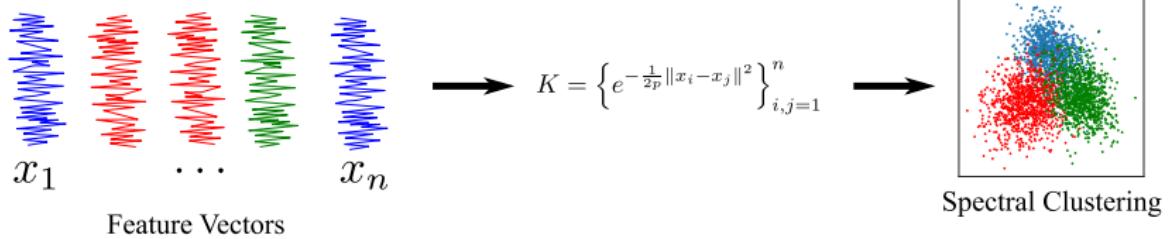
Example: GAN-generated images are concentrated random vectors!



Ok... so what?

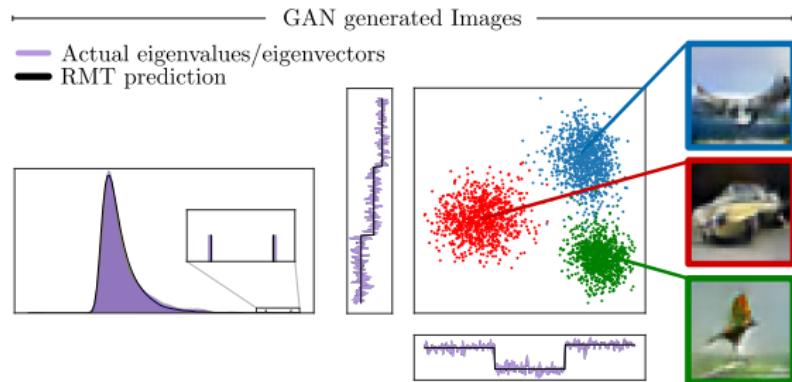


Ok... so what?



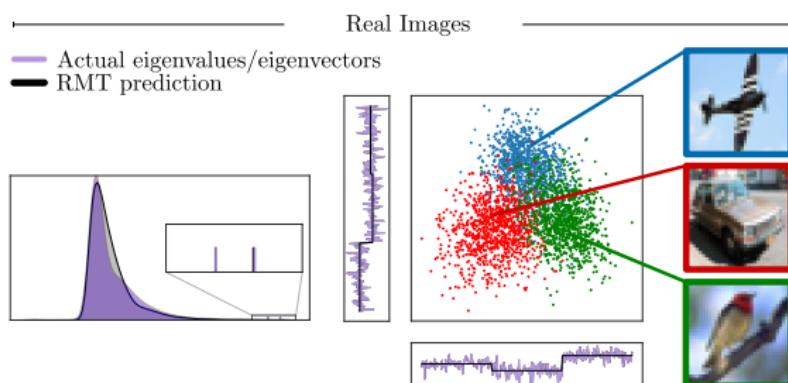
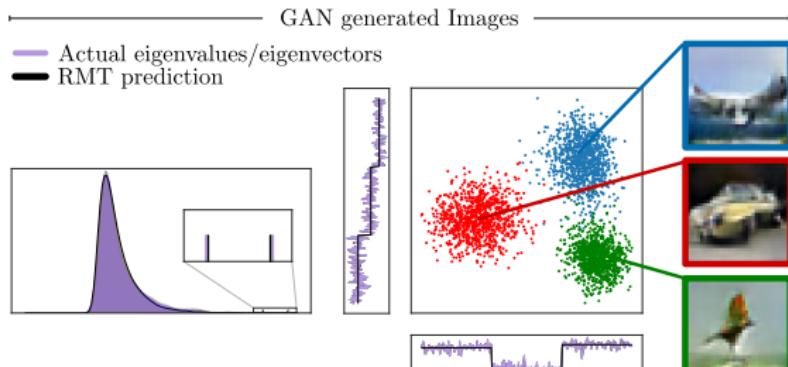
Ok... so what? (2)

Results. [Seddik,C'19]



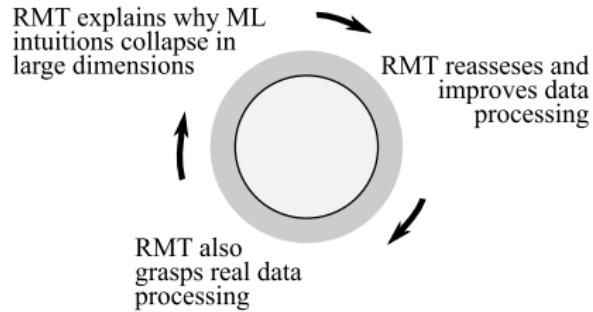
Ok... so what? (2)

Results. [Seddik,C'19]



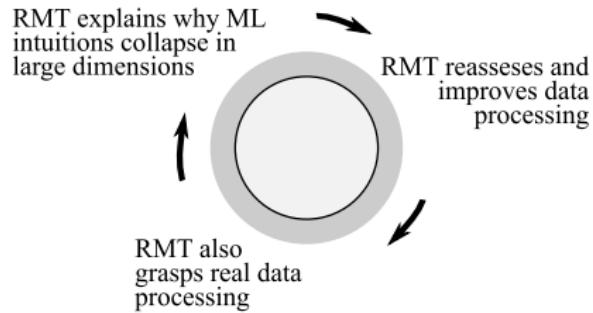
Conclusion

Reminder of Takeaway messages:



Conclusion

Reminder of Takeaway messages:

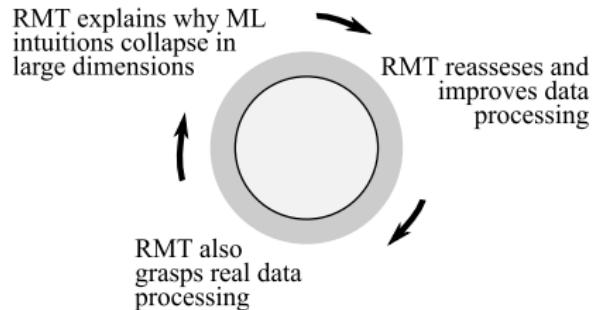


The road ahead:

- ▶ getting away from GMM models and show universality results (**concentration of measure arguments**)

Conclusion

Reminder of Takeaway messages:

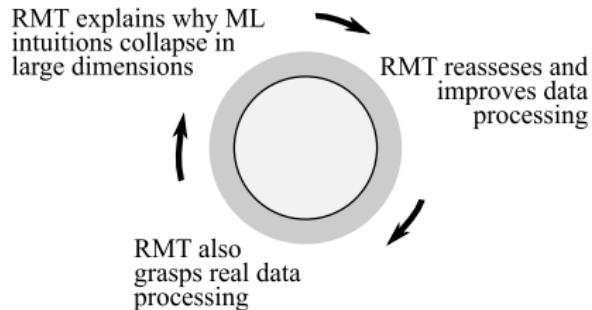


The road ahead:

- ▶ getting away from GMM models and show universality results (**concentration of measure arguments**)
- ▶ generalize the approach to problems having non-explicit solutions (such as convex optim problems)

Conclusion

Reminder of Takeaway messages:



The road ahead:

- ▶ getting away from GMM models and show universality results (**concentration of measure arguments**)
- ▶ generalize the approach to problems having non-explicit solutions (such as convex optim problems)
- ▶ deep learning, recurrent neural nets... are a very different story!

Thank you!

-  **[C-Benaych'16]** R. Couillet, Benaych-Georges, "Kernel Spectral Clustering of Large Dimensional Data", Electronic Journal of Statistics, vol. 10, no. 1, pp. 1393-1454, 2016. [article]
-  **[Mai,C'18]** X. Mai, R. Couillet, "A random matrix analysis and improvement of semi-supervised learning for large dimensional data", (in Press) Journal of Machine Learning Research, 2017. [article]
-  **[Louart,C'18]** C. Louart, Z. Liao, R. Couillet, "A Random Matrix Approach to Neural Networks", The Annals of Applied Probability, vol. 28, no. 2, pp. 1190-1248, 2018. [article]
-  **[Seddik,C'19]** M. Seddik, M. Tamaazousti, R. Couillet, "Kernel Random Matrices of Large Concentrated Data: The Example of GAN-Generated Image", IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP'19), Brighton, UK, 2019. [article]
-  H. Tiomoko Ali, R. Couillet, "Improved spectral community detection in large heterogeneous networks", Journal of Machine Learning Research, vol. 18, no. 225, pp. 1-49, 2018. [article]
-  R. Couillet, M. Tiomoko, S. Zozor, E. Moisan, "Random matrix-improved estimation of covariance matrix distances", (submitted to) Journal of Multivariate Analysis, 2018. [preprint]
-  Z. Liao, R. Couillet, "A Large Dimensional Analysis of Least Squares Support Vector Machines", IEEE Transactions on Signal Processing, vol. 67, no.4, pp. 1065-1074, 2018. [article]