Compressive Learning with Random Moments

Antoine Chatalic

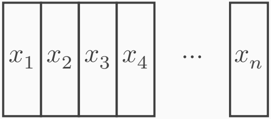
Ph.D. supervisor: Rémi Gribonval.

September 4, 2018

Université de Rennes 1, IRISA, Rennes – PANAMA research group

Large-Scale Machine Learning

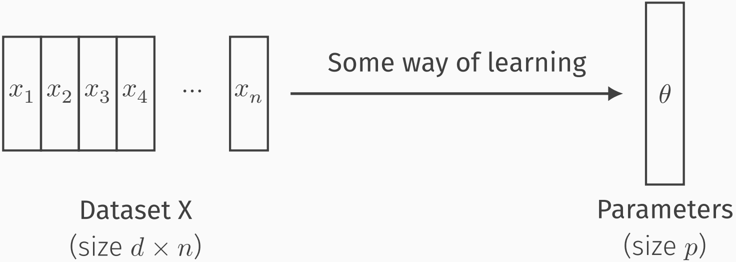
Goal: learn from the dataset about the underlying **distribution**!



Dataset X
(size $d \times n$)

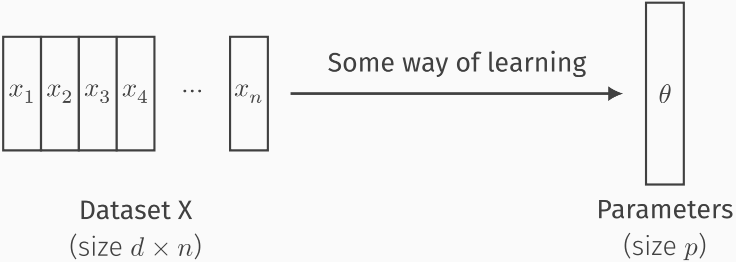
Large-Scale Machine Learning

Goal: learn from the dataset about the underlying **distribution**!



Large-Scale Machine Learning

Goal: learn from the dataset about the underlying **distribution**!



Example of learning task used in this talk: **k-means** clustering.

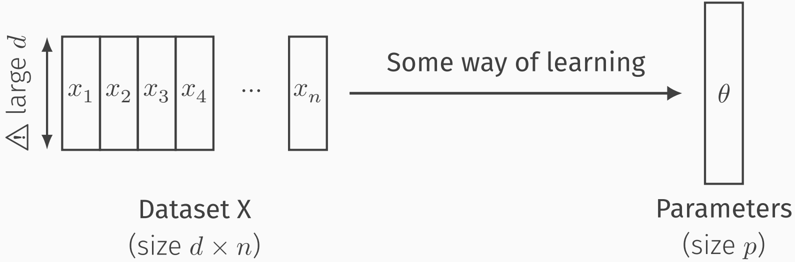
Goal: find θ = locations of k centroids $(c_j)_{1 \leq j \leq k}$ in $\mathbb{R}^d$ minimizing:

$$\text{SSE}((c_j)_{1\leq j\leq k}, X) = \sum_{i=1}^{n} \min_j \|x_i - c_j\|^2.$$

Each sample x_i is then assigned to the closest centroid.

Large-Scale Machine Learning

Goal: learn from the dataset about the underlying **distribution**!



Example of learning task used in this talk: k-means clustering.

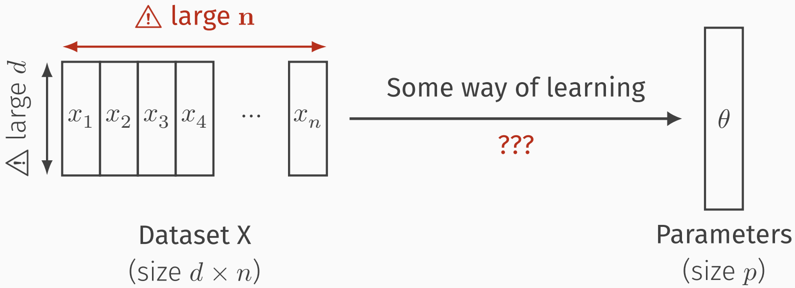
Goal: find θ = locations of k centroids $(c_j)_{1 \le j \le k}$ in $\mathbb{R}^d$ minimizing:

$$\text{SSE}((c_j)_{1 \le j \le k}, X) = \sum_{i=1}^{n} \min_j \|x_i - c_j\|^2.$$

Each sample x_i is then assigned to the closest centroid.

Large-Scale Machine Learning

Goal: learn from the dataset about the underlying **distribution**!



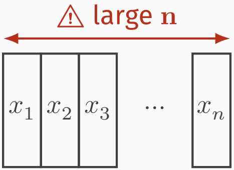
Example of learning task used in this talk: **k-means** clustering.

Goal: find θ = locations of k centroids $(c_j)_{1 \leq j \leq k}$ in $\mathbb{R}^d$ minimizing:

$$\text{SSE}((c_j)_{1 \leq j \leq k}, X) = \sum_{i=1}^{n} \min_j \|x_i - c_j\|^2.$$

Each sample x_i is then assigned to the closest centroid.

Handling Large Datasets: Several Approaches



Dataset (size $d \times n$).

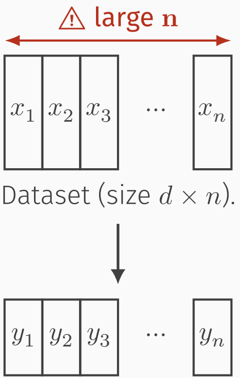
Use the whole dataset

e.g. run k-means, train a neural network.

⚠ Requires storage, RAM, time, GPUs.

"Compressive" approaches?

Handling Large Datasets: Several Approaches



Use the whole dataset

e.g. run k-means, train a neural network.

⚠ Requires storage, RAM, time, GPUs.

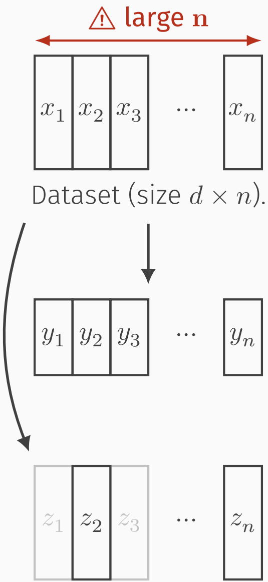
"Compressive" approaches?

Dimensionality reduction

New dimension $d' \ll d$, but still n samples!

Johnson-Lindenstrauss lemma: distances can be preserved with factor ε using $d' = \Theta(\frac{\log(n)}{\varepsilon^2})$.

Handling Large Datasets: Several Approaches



Use the whole dataset

e.g. run k-means, train a neural network.

⚠ Requires storage, RAM, time, GPUs.

"Compressive" approaches?

Dimensionality reduction

New dimension $d' \ll d$, but still n samples!

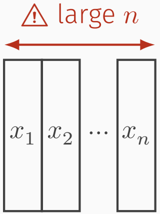
Johnson-Lindenstrauss lemma: distances can be preserved with factor ε using $d' = \Theta(\frac{\log(n)}{\varepsilon^2})$.

Subsampling

$n' \ll n$ samples, but still in dimension d!

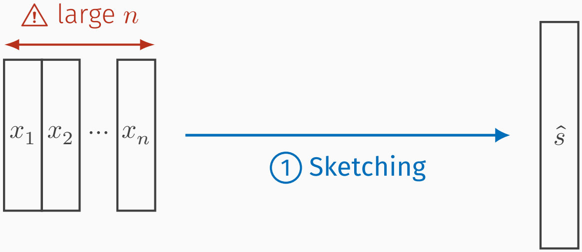
Coresets, Nyström methods.

Sketched Learning: Yet Another Idea



Dataset X
(size $d \times n$)

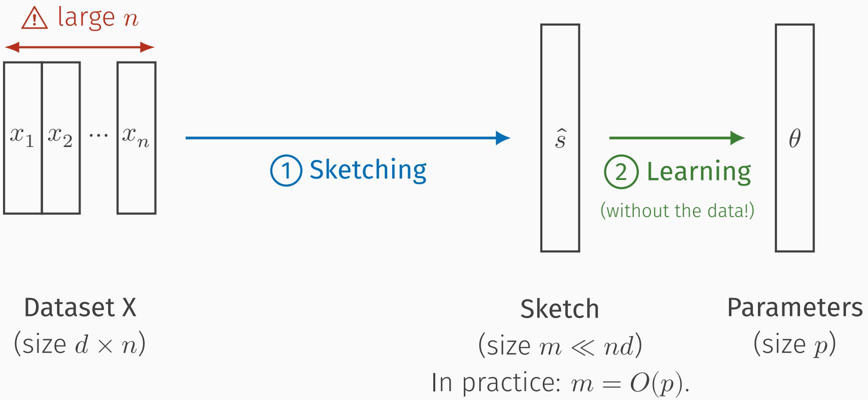
Sketched Learning: Yet Another Idea



Dataset X
(size $d \times n$)

Sketch
(size $m \ll nd$)

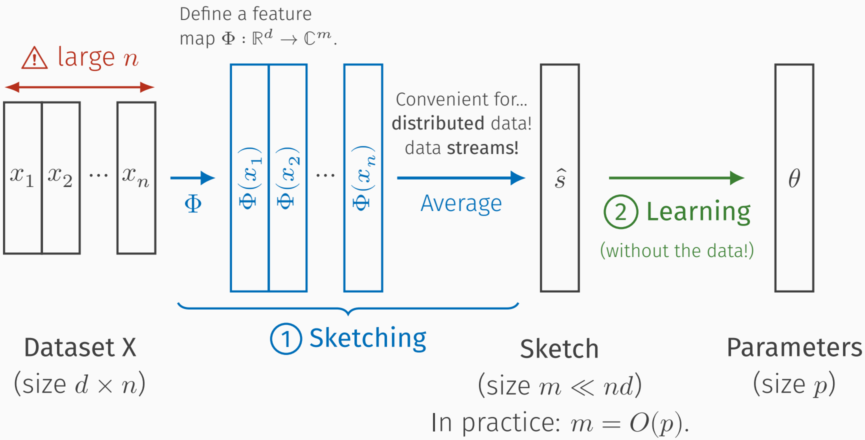
Sketched Learning: Yet Another Idea



Bourrier, Gribonval, and Pérez, 2015. **"Compressive Gaussian Mixture Estimation"**

Roots in **compressive sensing** (on distributions) Foucart and Rauhut (2013) .

Sketched Learning: Yet Another Idea



Bourrier, Gribonval, and Pérez, 2015. **“Compressive Gaussian Mixture Estimation”**

Roots in **compressive sensing** (on distributions) Foucart and Rauhut (2013) .

Choice of the Feature Function

Which feature map Φ can we use?

For k-means clustering and GMM fitting, **random Fourier features:**

$$\Phi(x) = \begin{bmatrix} e^{-i\omega_1^T x} \\ \vdots \\ e^{-i\omega_m^T x} \end{bmatrix} \in \mathbb{C}^m, \text{ with random i.i.d. } (\omega_j)_{1 \leq j \leq m}.$$

Rahimi and Recht, 2007. **"Random Features for Large-Scale Kernel Machines"**

Choice of the Feature Function

Which feature map Φ can we use?

For k-means clustering and GMM fitting, **random Fourier features:**

$$\Phi(x) = \begin{bmatrix} e^{-i\omega_1^T x} \\ \vdots \\ e^{-i\omega_m^T x} \end{bmatrix} \in \mathbb{C}^m, \text{ with random i.i.d. } (\omega_j)_{1 \leq j \leq m}.$$

Rahimi and Recht, 2007. **"Random Features for Large-Scale Kernel Machines"**

Hence, the j-th element of the sketch is: $\hat{s}_j = \frac{1}{n} \sum_{i=1}^{n} e^{-i\omega_j^T x_i}$.

Sketching = sampling the (empirical) **characteristic function** at the random frequency vectors $(\omega_j)_{1 \leq j \leq m}$.

Sketching and neural networks

Let $\Omega = \begin{bmatrix} \omega_1 & \omega_2 & \omega_3 & \cdots & \omega_n \end{bmatrix}$ be the matrix of frequencies.

Sketching is performed as follows:

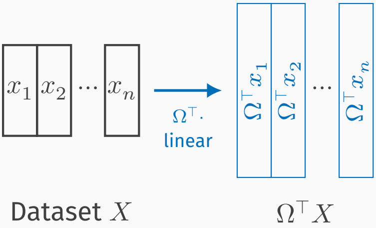
$$\begin{bmatrix} x_1 & x_2 & \cdots & x_n \end{bmatrix}$$

Dataset X

Sketching and neural networks

Let $\Omega = \begin{bmatrix} \omega_1 & \omega_2 & \omega_3 & \cdots & \omega_n \end{bmatrix}$ be the matrix of frequencies.

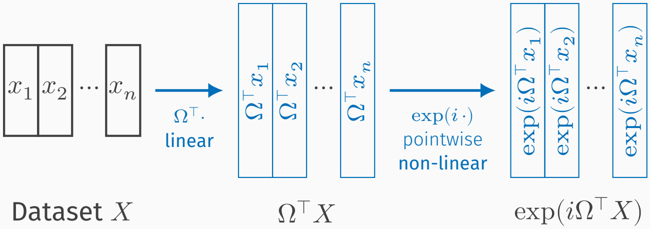
Sketching is performed as follows:



Sketching and neural networks

Let $\Omega = \begin{bmatrix} \omega_1 & \omega_2 & \omega_3 & \cdots & \omega_n \end{bmatrix}$ be the matrix of frequencies.

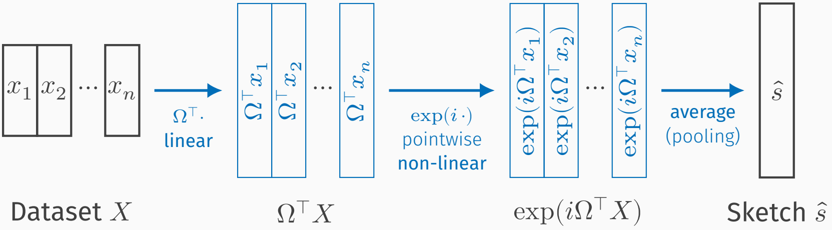
Sketching is performed as follows:



Sketching and neural networks

Let $\Omega = \begin{bmatrix} \omega_1 & \omega_2 & \omega_3 & \cdots & \omega_n \end{bmatrix}$ be the matrix of frequencies.

Sketching is performed as follows:



Sketching ≈ single-layer neural network with random weights + pooling.

cf. invertibility of CNNs Gilbert et al. (2017).

Multi-layer sketching → DNNs?

Solving the Inverse Problem

Learn from the empirical sketch = **moment-matching** problem.

cf. Generalized method of moments Hall (2005).

Example (k-means clustering): looking for centroids $(c_i)_{1\leq i\leq k}$ in $\mathbb{R}^d$:

$$(C,\alpha) = \arg\min_{C,\alpha} \Big\| \underbrace{\sum_{i=1}^{k} \alpha_i \Phi(c_i)}_{\text{sketch of the centroids } (c_i)_{1\leq i\leq k}} - \underbrace{\hat{s}}_{\text{empirical sketch}} \Big\|_2 .$$

⚠ This is non-convex!

Solving the Inverse Problem

Learn from the empirical sketch = **moment-matching** problem.

cf. Generalized method of moments Hall (2005).

Example (k-means clustering): looking for centroids $(c_i)_{1\leq i\leq k}$ in $\mathbb{R}^d$:

$$(C,\alpha) = \arg\min_{C,\alpha} \Big\| \underbrace{\sum_{i=1}^{k} \alpha_i \Phi(c_i)}_{\text{sketch of the centroids } (c_i)_{1\leq i\leq k}} - \underbrace{\hat{s}}_{\text{empirical sketch}} \Big\|_2 .$$

⚠ This is non-convex! Heuristics can be used in practice:

- **CL-OMP(R):** a greedy approach.
  It is a continuous adaptation of orthogonal matching pursuit with replacement.
  Keriven et al., 2016. **"Sketching for large-scale learning of mixture models"**

Solving the Inverse Problem

Learn from the empirical sketch = **moment-matching** problem.

cf. Generalized method of moments Hall (2005).

Example (k-means clustering): looking for centroids $(c_i)_{1\le i\le k}$ in $\mathbb{R}^d$:

$$(C,\alpha) = \arg\min_{C,\alpha} \Big\| \underbrace{\sum_{i=1}^{k} \alpha_i \Phi(c_i)}_{\text{sketch of the centroids } (c_i)_{1\le i\le k}} - \underbrace{\hat{s}}_{\text{empirical sketch}} \Big\|_2 .$$

⚠ This is non-convex! Heuristics can be used in practice:

- **CL-OMP(R):** a greedy approach.
  It is a continuous adaptation of orthogonal matching pursuit with replacement.
  Keriven et al., 2016. **"Sketching for large-scale learning of mixture models"**
- **CL-AMP:** inference using generalized message passing (GAMP).
  Graphical model linking the centroids (input signal) to the sketch (observation) through an input channel (Gaussian prior), a linear mixing and an output channel (non-linearity + pooling).
  Byrne, Gribonval, and Schniter, 2017. **"Sketched Clustering via Hybrid Approximate Message Passing"**

Sketch size

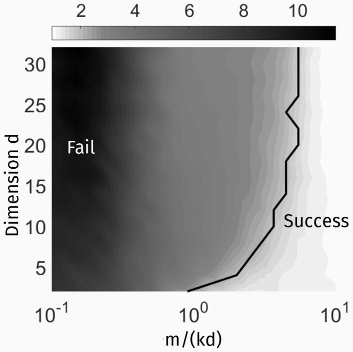
Which sketch size m to learn $p = kd$ parameters?

Statistical learning guarantees (control of the excess risk).
For ε-separated clusters in $\mathcal{B}(0, R)$:

$$m = O(k^2 d \log(R/\varepsilon)).$$

In practice: $m = O(kd)$ is sufficient.

Results for Gaussian mixtures and PCA as well.

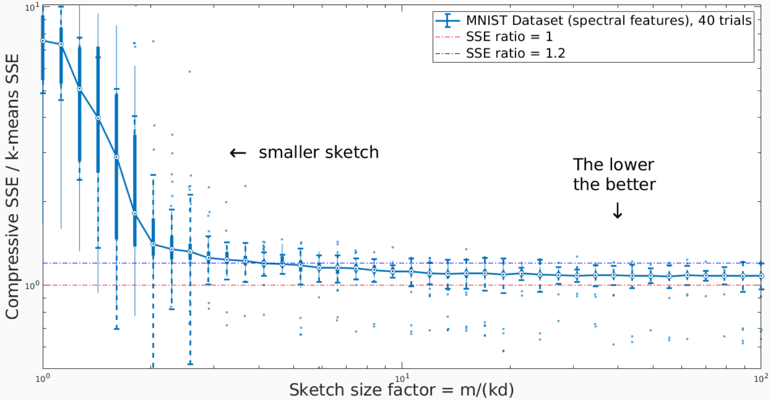


Color = SSE / k-means SSE Steinhaus (1956).
Data $\sim$ GMM, $k = 10$. Figure: Keriven et al. (2017).

Gribonval et al., 2017. **"Compressive statistical learning with random feature moments"**

Example: MNIST dataset

MNIST dataset of handwritten digits. $k = 10$, $d = 10$, $n = 70000$.



Construction of the Matrix of Frequencies

Replace $\Omega = \begin{bmatrix} \omega_1 & \omega_2 & \omega_3 & \cdots & \omega_n \end{bmatrix}$ by a **structured** matrix Ω^{fast}.

Construction using Walsh-Hadamard + diagonal Rademacher matrices.

cf. Yu et al. (2016), Choromanski, Rowland, and Weller (2017).

Construction of the Matrix of Frequencies

Replace $\Omega = \begin{bmatrix} \omega_1 & \omega_2 & \omega_3 & \cdots & \omega_n \end{bmatrix}$ by a **structured** matrix Ω^{fast}.

Construction using Walsh-Hadamard + diagonal Rademacher matrices.
cf. Yu et al. (2016) , Choromanski, Rowland, and Weller (2017) .

Products can be computed using the **fast Walsh-Hadamard transform!**

- **Time** complexity (both sketching & learning): $d^2 \to d \log(d)$.
- **Space** complexity (RAM + storage of Ω): $d^2 \to d$.
  Example: size of Ω for $d = 1024, k = 512$: 40Go $\to$ 1.8Mo.

$\leadsto$ Same clustering quality in practice.

Chatalic, Gribonval, and Keriven, 2018. **"Large-Scale High-Dimensional Clustering with Fast Sketching"**

Recent Advances: Privacy-Aware Learning

First way to get privacy: compute **less than m observations** per data sample x_i.

Each measurement $\exp(i\omega_j^\top x_i)$ is computed **only with probability $\boldsymbol{\alpha}$.**

Hence, low α = more privacy.

Each $\hat{s}_j$ is computed from $\approx \alpha n$ samples, and:

$$\text{Var}(\hat{s}_j) \approx \frac{1}{\alpha n}$$

Recent Advances: Privacy-Aware Learning

First way to get privacy: compute **less than m observations** per data sample x_i.

Each measurement $\exp(i\omega_j^\top x_i)$ is computed **only with probability α.**

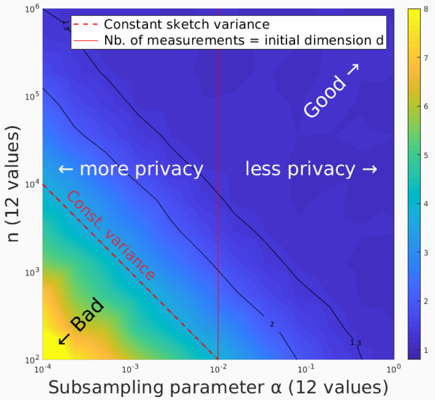
Hence, low α = more privacy.

Each $\hat{s}_j$ is computed from $\approx \alpha n$ samples, and:

$$\text{Var}(\hat{s}_j) \approx \frac{1}{\alpha n}$$

Color = error (SSE) w.r.t. k-means.

$d = k = 10$, $m = 10kd$.



Conclusion: only the **variance** of the sketch matters.
Can also help to reduce clustering time!

Recent Advances: Privacy-Aware Learning II

Noise on the sketches s_{x_i}:

$$s_{x_i}^{\text{sk.n.}} = \exp(i\Omega^\top x_i) + \xi_i$$

Recent Advances: Privacy-Aware Learning II

Noise on the sketches s_{x_i}:

$$s_{x_i}^{\text{sk.n.}} = \exp(i\Omega^\top x_i) + \xi_i$$

Noise on the data samples x_i:

$$s_{x_i}^{\text{d.n.}} = \exp(i\Omega^\top (x_i + \xi_i))$$

Here $x + \xi \sim p_x \star p_\xi$, hence $s^{\text{d.n.}} = s^x \odot s^\xi$. This sketch can be "deconvolved" (but it increases the variance!).

Recent Advances: Privacy-Aware Learning II

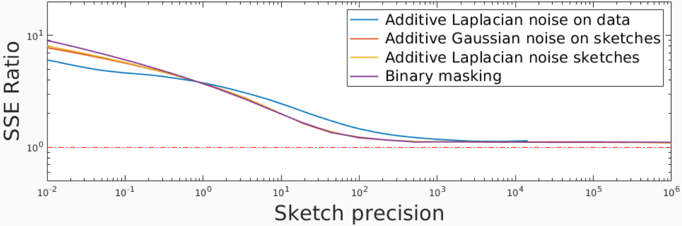
Noise on the sketches s_{x_i}:

$$s_{x_i}^{\text{sk.n.}} = \exp(i\Omega^\top x_i) + \xi_i$$

Noise on the data samples x_i:

$$s_{x_i}^{\text{d.n.}} = \exp(i\Omega^\top (x_i + \xi_i))$$

Here $x + \xi \sim p_x \star p_\xi$, hence $s^{\text{d.n.}} = s^x \odot s^\xi$. This sketch can be "deconvolved" (but it increases the variance!).



The 3 scenarios can be compared by looking at the variance.
Guarantees have yet to be established (e.g. **local differential privacy**).

Conclusion

- A framework for learning efficiently from **large distributed collections** or **data streams**.
- Similar to a single-layer random neural network with pooling.
- Learning = moment-matching; **heuristics** have been proposed to solve the optimization problem.
- **Theoretical guarantees** on the sketch size have been obtained.
- Fast transforms allow to deal with **large dimensions**.
- **Privacy** by construction (pooling), noise can be added.
- More learning tasks: union of subspaces, regression, classification...

Questions?

Bibliography I

Bourrier, Anthony, Rémi Gribonval, and Patrick Pérez (2015). “Compressive Gaussian Mixture Estimation”. In: *Compressed Sensing and its Applications*. Ed. by Holger Boche et al. Applied and Numerical Harmonic Analysis. Springer International Publishing, pp. 239–258.

Byrne, Evan, Rémi Gribonval, and Philip Schniter (2017). “Sketched Clustering via Hybrid Approximate Message Passing”. In: *Asilomar Conference on Signals, Systems, and Computers*.

Chatalic, Antoine, Rémi Gribonval, and Nicolas Keriven (2018). “Large-Scale High-Dimensional Clustering with Fast Sketching”. In: *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*.

Choromanski, Krzysztof, Mark Rowland, and Adrian Weller (2017). “The Unreasonable Effectiveness of Random Orthogonal Embeddings”. In: *arXiv preprint arXiv:1703.00864*.

Bibliography II

Foucart, Simon and Holger Rauhut (2013). *A Mathematical Introduction to Compressive Sensing*. Applied and Numerical Harmonic Analysis. Birkhäuser.

Gilbert, Anna C. et al. (2017). “Towards Understanding the Invertibility of Convolutional Neural Networks”. In: *arXiv:1705.08664 [cs, stat]*. arXiv: `1705.08664`.

Gribonval, Rémi et al. (2017). “Compressive statistical learning with random feature moments”. In: *arXiv preprint arXiv:1706.07180*.

Hall, Alastair R. (2005). *Generalized method of moments*. Oxford university press.

Keriven, N. et al. (2016). “Sketching for large-scale learning of mixture models”. In: *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 6190–6194.

Bibliography III

Keriven, Nicolas et al. (2017). "Compressive K-means". In: Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP).

Rahimi, Ali and Benjamin Recht (2007). "Random Features for Large-Scale Kernel Machines". In: *Advances in Neural Information Processing Systems (NIPS)*. Curran Associates, Inc., pp. 1177–1184.

Steinhaus, Hugo (1956). "Sur la division des corp materiels en parties". In: *Bull. Acad. Polon. Sci* 1.804, p. 801.

Yu, Felix X. et al. (2016). "Orthogonal Random Features". In: *Advances in Neural Information Processing Systems (NIPS)*, pp. 1975–1983.