Sketching Datasets for Large-Scale Learning (long version)

Rémi Gribonval, Antoine Chatalic, Nicolas Keriven,
Vincent Schellekens, Laurent Jacques, and Philip Schniter

Abstract

This article considers "compressive learning," an approach to large-scale machine learning where datasets are massively compressed before learning (*e.g.*, clustering, classification, or regression) is performed. In particular, a "sketch" is first constructed by computing carefully chosen nonlinear random features (*e.g.*, random Fourier features) and averaging them over the whole dataset. Parameters are then learned from the sketch, without access to the original dataset. This article surveys the current state-of-the-art in compressive learning, including the main concepts and algorithms, their connections with established signal-processing methods, existing theoretical guarantees—on both information preservation and privacy preservation, and important open problems.

Big data can be a blessing: with very large training datasets it becomes possible to perform complex learning tasks with unprecedented accuracy. Yet, this improved performance comes at the price of enormous computational challenges. Thus, one may wonder: Is it possible to leverage the information content of huge datasets while keeping computational resources under control? Can this also help solve some of the privacy issues raised by large-scale learning? This is the ambition of *compressive learning,* where the dataset is massively compressed *before* learning. Here, a "sketch" is first constructed by computing carefully chosen nonlinear random features (*e.g.*, random Fourier features) and averaging them over the whole dataset. Parameters are then learned from the sketch, without access to the original dataset. This article surveys the current state-of-the-art in compressive learning, including the main concepts and algorithms; their connections with established signal-processing methods; existing theoretical guarantees, on both information preservation and privacy preservation; and important open problems.

1 Introduction to Compressive Learning

The overall principle of compressive learning is summarized in Fig. 1. During the *sketching phase*, a potentially huge collection of d-dimensional data vectors $\{\boldsymbol{x}_i\}_{i=1}^n$ is summarized into a single m-dimensional vector $\widetilde{\boldsymbol{z}}$, called the "sketch". The sketch is constructed by transforming each data vector and then averaging the results:

$$\widetilde{\boldsymbol{z}} := \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{\Phi}(\boldsymbol{x}_i). \tag{1}$$

Next, during the *learning phase*, an estimate $\widetilde{\boldsymbol{\theta}}$ of some essential statistical parameters $\boldsymbol{\theta}$ of the dataset are extracted from the sketch $\widetilde{\boldsymbol{z}}$. These parameters are application dependent. For example, as illustrated in Fig. 2, they could represent principal data subspaces (in subspace fitting problems), prediction weights (in

R. Gribonval is with Univ Lyon, Inria, CNRS, ENS de Lyon, UCB Lyon 1, LIP UMR 5668, F-69342, Lyon, France; remi.gribonval@inria.fr. A. Chatalic is with Univ Rennes, Inria, CNRS, IRISA, F-35042 Rennes Cedex, France; antoine.chatalic@irisa.fr. N. Keriven is with CNRS, GIPSA-lab, Grenoble INP, UGA, Grenoble, France; nicolas.keriven@gipsa-lab.grenoble-inp.fr. V. Schellekens and L. Jacques are with ISPGroup/ICTEAM, UCLouvain, Belgium; {vincent.schellekens,laurent.jacques}@uclouvain.be. P. Schniter is with The Ohio State University, Columbus, OH, USA; schniter.1@osu.edu

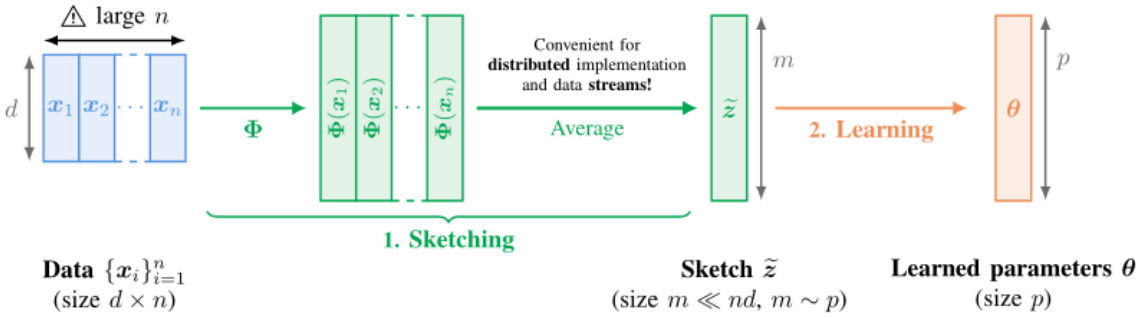


Fig. 1: Overview of sketching and parameter learning.

estimation and regression problems), distributional parameters (in density estimation problems), or centroids (in clustering problems); we give detailed examples below. Python notebooks that illustrate examples from the paper are available at https://gitlab.inria.fr/SketchedLearning/spm-notebook.

The transformation $\boldsymbol{\Phi}(\cdot)$, known as the "feature map," is generally non-linear and randomized. Although the use of $\boldsymbol{\Phi}(\cdot)$ is related to "kernel methods" in machine learning (*e.g.*, [1, 2][3–5]), as will be discussed later, the act of sketching (1) also includes averaging over the n data samples. The advantages of compressive learning are the following.

1) By choosing a sketch of dimension $m \ll nd$, the data gets massively compressed. This has obvious advantages for storage and transfer.
2) Sketching can speed up the learning phase, whose complexity becomes independent of the cardinality, n, of the original dataset. This enables one to handle massive datasets while keeping computational resources under control.
3) Sketching can preserve privacy: the transformation $\boldsymbol{\Phi}(\cdot)$ can be chosen so that individual-user information is lost while aggregate-user information is preserved.
4) The sketching mechanism in (1) is well matched to distributed implementations and streaming scenarios: the sketch of a concatenation of datasets is a simple mean of the sketches of those datasets.

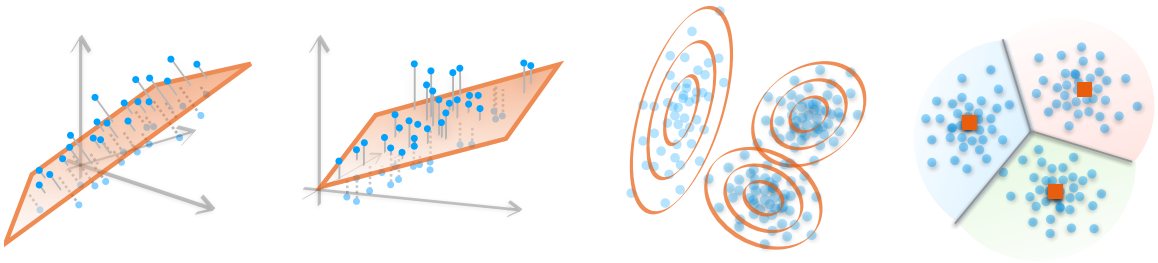
2 Illustration using Four Worked Examples

To illustrate the compressive-learning framework and discuss the essential aspects of it, we now outline four canonical examples of machine learning tasks to which sketching can be readily applied. See Fig. 2 for an illustration.

a) Principal component analysis (PCA)

PCA seeks to find the linear subspace of a fixed dimension $k < d$ that best fits the d-dimensional data $\{\boldsymbol{x}_i\}_{i=1}^n$ in the least-squares (LS) sense. In this case, the target parameters $\boldsymbol{\theta}$ can be described by a k-dimensional orthonormal basis $\{\boldsymbol{u}_\ell\}_{\ell=1}^k$ that maximizes $\sum_{\ell=1}^k \sum_{i=1}^n |\boldsymbol{u}_\ell^\top \boldsymbol{x}_i|^2$.

It is well known that one solution is given by the k principal eigenvectors of the empirical autocorrelation matrix $\widehat{\boldsymbol{R}} = \frac{1}{n}\sum_{i=1}^n \boldsymbol{x}_i \boldsymbol{x}_i^\top$. This $\widehat{\boldsymbol{R}}$ can be interpreted as a sketch of the form (1) that uses the feature map $\boldsymbol{\Phi}(\boldsymbol{x}) = \mathrm{vec}(\boldsymbol{x}\boldsymbol{x}^\top) = (x_1 x_1, x_2 x_1, \cdots, x_d x_d)^\top$ of dimension $m = d^2$. Here and in the sequel, the $\mathrm{vec}(\cdot)$ operator vectorizes a matrix by stacking its columns. As soon as the cardinality n of the dataset exceeds the dimension d, the dimension of this sketch is smaller than nd, the total size of the dataset. Using



(a) PCA (b) Linear regression (c) Mixture modeling (d) Clustering

Fig. 2: Schematic representation of the four running examples covered by this paper. In the four figures, each $\boldsymbol{x}_i$ is associated with a blue colored disk, hence the training collection correspond to a point cloud, and the orange color geometrically represents the learned parameters. (a) PCA learns the principal k-dimensional subspace of the dataset for some $k \leq d$. (b) Linear regression fits observed data (the blue dot heights) as a linear model of the inputs (here, the 2-D horizontal coordinates). For least squares, this amounts to minimizing the square of the differences between these data and the linear predictions. (c) In Gaussian mixture modeling, we learn the set of parameters (mixture weight, mean, and covariance) characterizing each Gaussian term of the mixture, which probability level sets are displayed here as orange ellipses. (d) Clustering methods (such as k-means) learn a set of centroids (the orange squares) defining a Voronoi partition grouping together similar data samples.

techniques from matrix completion and compressive matrix sensing, the sketch dimension can be further reduced [6][7]. For example, one can sketch via (1) using $\boldsymbol{\Phi}(\boldsymbol{x}) = \left((\boldsymbol{w}_1^\top \boldsymbol{x})^2, \ldots, (\boldsymbol{w}_m^\top \boldsymbol{x})^2\right)^\top$, where the d-dimensional vectors $\boldsymbol{w}_j$, $1 \leq j \leq m$ are the rows of a tall random matrix $\boldsymbol{W}$ of size $m \times d$. For m on the order of $kd < d^2$, low-rank matrix reconstruction techniques can estimate the k-dimensional principal subspace of $\widehat{\boldsymbol{R}}$ with provable accuracy [8].

b) LS linear regression

Suppose that the data vectors take the form $\boldsymbol{x}_i = [\boldsymbol{x}_{1i}^\top, \boldsymbol{x}_{2i}^\top]^\top$, and our goal is to linearly predict the d_1-dimensional vector $\boldsymbol{x}_{1i}$ from the d_2-dimensional vector $\boldsymbol{x}_{2i}$ (with $d = d_1 + d_2$). That is, we want to design a $d_1 \times d_2$ weight matrix $\boldsymbol{\theta}$ such that $\boldsymbol{x}_{1i} \approx \boldsymbol{\theta}\boldsymbol{x}_{2i}$ for all samples i. The LS approach to this supervised learning problem chooses $\widehat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} \sum_{i=1}^n \|\boldsymbol{x}_{1i} - \boldsymbol{\theta}\boldsymbol{x}_{2i}\|^2$.

Although it is possible to compute the LS solution using gradient descent, each iteration would involve the full dataset $\{\boldsymbol{x}_i\}_{i=1}^n$. A well-known alternative is to first build the empirical auto-correlation matrix $\widehat{\boldsymbol{R}} := \frac{1}{n}\sum_{i=1}^n \boldsymbol{x}_i\boldsymbol{x}_i^\top$ and then compute $\widehat{\boldsymbol{\theta}}$ in closed-form as

$$\widehat{\boldsymbol{\theta}} = \widehat{\boldsymbol{R}}_{12}\widehat{\boldsymbol{R}}_{22}^{-1}, \text{ with } \begin{pmatrix} \widehat{\boldsymbol{R}}_{11} & \widehat{\boldsymbol{R}}_{12} \\ \widehat{\boldsymbol{R}}_{21} & \widehat{\boldsymbol{R}}_{22} \end{pmatrix} = \widehat{\boldsymbol{R}}, \tag{2}$$

where each sub-matrix $\widehat{\boldsymbol{R}}_{ij}$ has dimension $d_i \times d_j$. Here we again use the sketch (1) with the feature map $\boldsymbol{\Phi}(\boldsymbol{x}) = \mathrm{vec}(\boldsymbol{x}\boldsymbol{x}^\top)$ and extract the target parameter $\widehat{\boldsymbol{\theta}}$ from (2).

c) Gaussian-mixture modeling

Here the objective is to find the parameters $\boldsymbol{\theta} = \{\alpha_\ell, \boldsymbol{\mu}_\ell, \boldsymbol{\Sigma}_\ell\}_{\ell=1}^k \subset \boldsymbol{\Theta}$ that best fit a k-term Gaussian-mixture model $p(\boldsymbol{x}|\boldsymbol{\theta}) = \sum_{\ell=1}^k \alpha_\ell \mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}_\ell, \boldsymbol{\Sigma}_\ell)$ to the data $\{\boldsymbol{x}_i\}_{i=1}^n$. The parameter space $\boldsymbol{\Theta}$ demands that, for each ℓ: $\alpha_\ell > 0$, $\boldsymbol{\mu}_\ell$ is a d-dimensional centroid, $\boldsymbol{\Sigma}_\ell$ is a $d \times d$ positive definite matrix, and $\sum_\ell \alpha_\ell = 1$. Traditionally, expectation-maximization (EM) [9] is used to approximate the maximum likelihood (ML) estimate $\widehat{\boldsymbol{\theta}} = \arg\max_{\boldsymbol{\theta}\in\boldsymbol{\Theta}} \sum_{i=1}^n \log p(\boldsymbol{x}_i|\boldsymbol{\theta})$, but the EM algorithm processes all n data samples $\{\boldsymbol{x}_i\}_{i=1}^n$ at each iteration, which can be computationally burdensome when n is very large.

[Box 1] Compressive Mixture Modeling for Speaker Verification

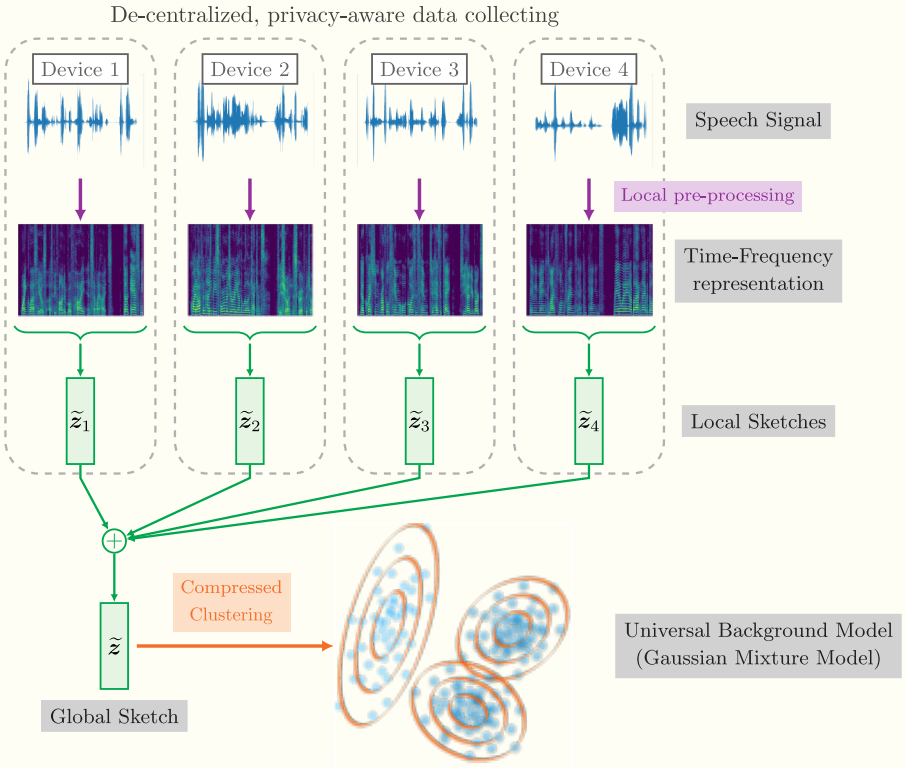


Fig. 3: Speaker verification via compressive learning. Unlabeled training speech data is collected in a decentralized manner, pre-processed, and locally sketched. The local sketches are then merged into a global sketch, from which a Universal Background Model is learned.

Given a fragment of speech and a candidate speaker, the goal of speaker verification is to assess whether the fragment was indeed spoken by that person. A classic approach to speaker verification is the GMM-UBM [10] (Gaussian Mixture Model–Universal Background Model). There, the idea is to train a model of a "universal" speaker from unlabeled training data and then compare it to a specialized model for the candidate speaker. Using the speech fragment, the likelihood ratio between the specialized and universal models is computed, and a positive decision is made if the ratio exceeds a certain threshold. In GMM-UBM, the data is modeled using a GMM. That is, $p(\boldsymbol{x}|\boldsymbol{\theta})$ are multivariate Gaussian distributions applied to a suitable time-frequency transform of the raw audio signal [10] (see Fig. 3).

The training of the UBM is computationally demanding, since it must be done on a large corpus of speech data. Moreover, the latter must be collected in a wide range of situations, so that the final model may be as universal as possible. For this reason, the data collecting process is best performed in a decentralized manner. Finally, speech data collected in real-life situations is known to be sensitive information. For all of these reasons, compressive learning is well suited to speaker verification.

In [11], using an RF feature map $\boldsymbol{\Phi}(\cdot)$, the authors compressed 1000 hours of speech data (50 gigabytes) into a sketch of a few kilobytes on a single laptop. Subsequently, they performed GMM estimation, i.e., learning, using a greedy algorithm, thereby tackling the optimization problem (18). They compared this compressive-learning approach to the EM algorithm, which, on the same laptop, could only be trained using 5 hours of speech given the available RAM. They observed that, by enabling the use of more data within a fixed memory budget, sketching produced better results, despite the tremendous compression factor (see Fig. 4).

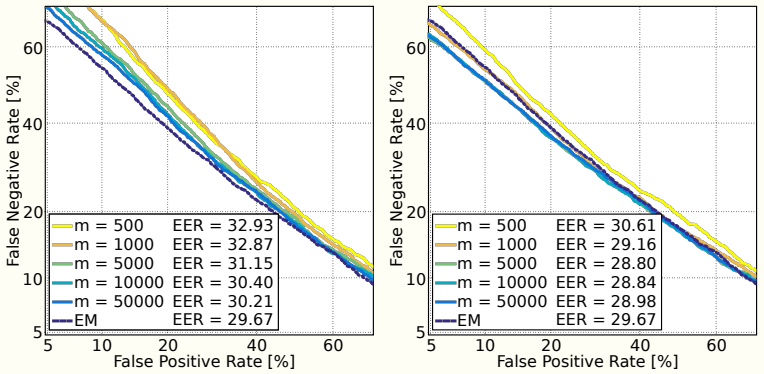


Fig. 4: Detection Error Tradeoff (DET) curve (False-Positive/False-Negative tradeoff obtained by varying the decision threshold) and Equal Error Rate (EER). On a single laptop: EM was trained on $5h$ of speech, compressive GMM estimation used $5h$ in the left figure and $1000h$ in the right, for different sketch dimensions m.

An alternative [11] is to compute a sketch $\widetilde{\boldsymbol{z}}$ of the form (1) using random Fourier (RF) features [12]:

$$\boldsymbol{\Phi}(\boldsymbol{x}) = \exp(-\mathrm{j}2\pi \boldsymbol{W}\boldsymbol{x}), \tag{3}$$

where $\mathrm{j} = \sqrt{-1}$ and $\boldsymbol{W} \in \mathbb{R}^{m\times d}$ is a realization of a random matrix (*e.g.*, i.i.d. Gaussian). Here, $\exp(\cdot)$ is the componentwise exponential, not the matrix exponential. We will have a lot to say about the RF feature map later in this article.[1] The parameters can then be extracted by optimizing a cost function as explained later. This approach has been applied to audio source-separation [13] as well as speaker verification [11], where it was shown that 1000 hours of speech can be compressed down to a few kilobytes without loss of verification performance (see Box 1).

d) k-means clustering

The goal of clustering is to group together "similar" data samples from $\{\boldsymbol{x}_i\}_{i=1}^n$. In the k-means approach to clustering, one aims to find the set of k centroids $\{\boldsymbol{c}_\ell\}_{\ell=1}^k$ that minimizes the average squared distance from each sample to its nearest centroid, *i.e.*, $\sum_{i=1}^n \min_\ell \|\boldsymbol{x}_i - \boldsymbol{c}_\ell\|^2$. The famous Lloyd algorithm [14][15] is typically used in an attempt to solve this problem. When n is very large, however, Lloyd's algorithm becomes computationally demanding. Instead, one could sketch the dataset using (1) and extract the centroids from the sketch [16, 17]. For this purpose, one could use RF features (3) and (as explained in the sequel) solve for the centroids $\{\boldsymbol{c}_\ell\}_{\ell=1}^k$ and the (non-negative, sum-to-one) weights $\{\alpha_\ell\}_{\ell=1}^k$ that minimize $\|\widetilde{\boldsymbol{z}} - \sum_{\ell=1}^k \alpha_\ell \boldsymbol{\Phi}(\boldsymbol{c}_\ell)\|$. On large datasets, this approach can be orders-of-magnitude better than Lloyd's algorithm in memory and runtime, provided that the sketch dimension m is large enough, *i.e.*, that m is on the order of kd, where kd is the number of free parameters in $\{\boldsymbol{c}_\ell\}_{\ell=1}^k$ [16, 17]. For example, this method allows us to cluster the MNIST digit dataset, of dimension $d = 784$ and cardinality $n = 70\,000$, using a complex-valued sketch of dimension $m = 400$ (see Box 2).

[1]For example, the connection between the RF map and the Gaussian kernel is discussed in Box 8.

[Box 2] Compressive Clustering of MNIST digits

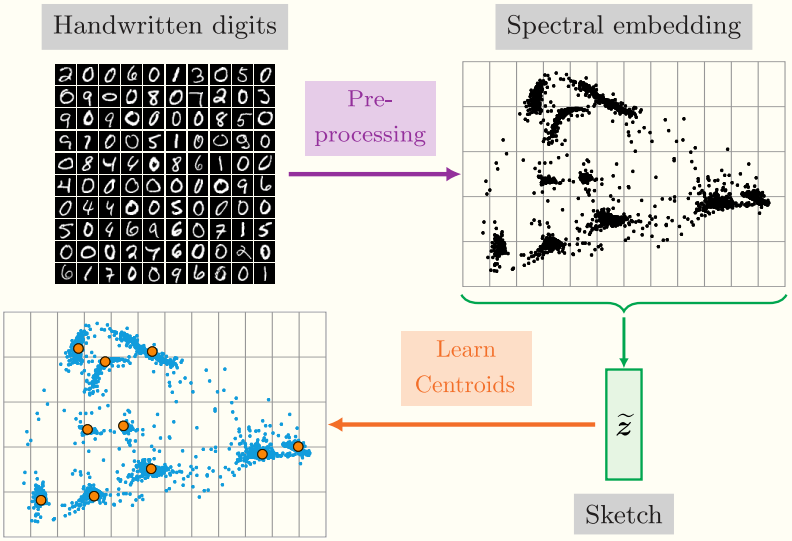


Fig. 5: Clustering of handwritten digits via compressive k-means. First, SIFT descriptors are extracted from each image. Then a similarity graph is constructed to obtain a so-called *spectral embedding* of the dataset (using the first eigenvectors of the Laplacian of the graph). The spectral features are aggregated into a sketch, from which centroids are extracted.

Clustering is a classic machine-learning task and a component of many machine-learning pipelines. The most popular method is Lloyd's algorithm [15][14], which aims to solve the k-means problem. In many cases, the raw features are converted to a spectral embedding before clustering. As a proof of concept, the authors in [16] applied this technique to handwritten digits from the MNIST dataset, but used compressive k-means clustering in place of Lloyd's algorithm (see Fig. 5). Using a $n = 10^7$-sample augmentation of MNIST, they found that compressive k-means clustering gave approximately the same accuracy as Lloyd's algorithm but reduced the time- and memory-complexity by 1.5 and 4 orders-of-magnitude, respectively (see Fig. 6).

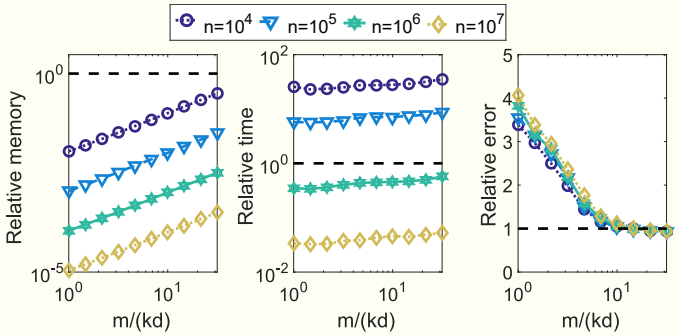


Fig. 6: Memory, runtime, and error of compressive k-means clustering relative to Lloyd's algorithm for various sample cardinalities n and sketch lengths m, using $k = 10$ centroids and $d = 10$ dimensional spectral features. Figure from [16].

Although this paper focuses on these examples, the general compressive-learning framework extends beyond them. It has been applied to, for example, Independent Component Analysis [18] (by sketching the cumulant tensor) and subspace clustering [19] (using a polynomial feature map). Extending the framework to a new task raises essential questions such as: How do we choose the feature map $\boldsymbol{\Phi}(\cdot)$? How do we learn the essential parameters $\boldsymbol{\theta}$ from the sketch $\widetilde{\boldsymbol{z}}$? Are there statistical learning guarantees? Can sketching and

learning be made computationally efficient? Can we sketch while respecting privacy? This paper sketches answers to these questions for the worked examples introduced above.

[Box 3] Notations and Conventions

Notation	Description
$\mathcal{X} = \{\boldsymbol{x}_i\}_{i=1}^n$	dataset of n training samples $\boldsymbol{x}_i \in \mathbb{R}^d$
$\widetilde{\boldsymbol{z}}$	(empirical) sketch of the dataset, a vector of dimension m
$\boldsymbol{\Phi}(\cdot)$	sketching feature map, a function mapping $\mathbb{R}^d$ to either $\mathbb{R}^m$ or $\mathbb{C}^m$
$\boldsymbol{\theta} \in \boldsymbol{\Theta}$	target parameters to learn, of dimension p (*e.g.*, PCA matrix, centroids, mixture model)
$\boldsymbol{W}$	random matrix associated with certain feature maps $\boldsymbol{\Phi}(\cdot)$, usually drawn i.i.d. Gaussian
$\varrho(\cdot)$	componentwise nonlinearity associated with certain feature maps $\boldsymbol{\Phi}(\cdot)$
w.h.p.	with high probability, *i.e.*, with exponentially decaying failure probability
i.i.d.	independent and identically distributed
w.p.1	with probability 1
$\|\cdot\|, \langle\cdot,\cdot\rangle$	Euclidean norm of a vector, inner product between vectors
$\|\cdot\|_0$	ℓ^0 norm of a vector, *i.e.*, its number of nonzero entries
$\|\cdot\|_F$	Frobenius norm of a matrix
$\|\cdot\|_{L^2}, \langle\cdot,\cdot\rangle_{L^2}$	L^2 norm of a function, L^2 inner product between functions

3 HISTORICAL BACKGROUND

The term "sketch" has different meanings, depending on the field. Our use of the term comes from the literature on relational databases, and more particularly from the subfield of *Approximate Query Processing* (AQP) [20]. The goal of AQP is to build a short description of the content of a massive dataset, called a *synopsis*, by analogy with the synopsis of a movie or a book, such that certain *queries* can be efficiently performed to return answers with controlled error and/or probability of failure. Well-known queries include the frequency of occurrence of a particular element in a stream of data (taken from a discrete collection), and the minimum of several of these frequencies, yielding the celebrated count-min sketch [20, Section 5.3.1][21] synopsis. In this context, the statistical parameters $\boldsymbol{\theta}$ learned from a sketch are interpreted as the result of a particular query on the dataset.

Despite the apparent similarities between AQP and compressive learning, both datatypes and learning tasks differ significantly between the two fields. Typically, AQP focuses on (multi)sets of elements taken from a discrete collection of objects and considers database queries and related operations, while compressive learning focuses on continuous-valued signals (*e.g.*, images or audio signals) and considers machine-learning tasks such as density estimation or regression.

3.1 SKETCHES FOR STREAMING AND DISTRIBUTED METHODS

In AQP, a popular class of synopses is that of *linear sketches* [20, Chap. 5]. A linear sketch, which maps a dataset to a vector, must satisfy the single condition that *the sketch of the concatenation of two datasets equals the sum of their sketches*. In mathematical terms, if we denote by $\widetilde{\boldsymbol{z}}(\mathcal{X})$ the sketch of a dataset $\mathcal{X} = \{\boldsymbol{x}_i\}_{i=1}^n$, then it is required that $\widetilde{\boldsymbol{z}}(\mathcal{X}) = \widetilde{\boldsymbol{z}}(\mathcal{X}_1) + \widetilde{\boldsymbol{z}}(\mathcal{X}_2)$ whenever $\mathcal{X}$ is the concatenation of $\mathcal{X}_1$ and $\mathcal{X}_2$. It thus follows that a linear sketch must be of the form $\widetilde{\boldsymbol{z}}(\mathcal{X}) = \sum_{i=1}^n \boldsymbol{\Phi}(\boldsymbol{x}_i)$ for some possibly nonlinear feature map $\boldsymbol{\Phi}(\cdot)$. That is the same definition that we adopt in (1), except that we normalize by the number of elements n. Linear sketches are popular in AQP mainly because they are well suited to *streaming* scenarios. That is, inserting or deleting an element $\boldsymbol{x}_i$ from the dataset corresponds to adding or subtracting $\boldsymbol{\Phi}(\boldsymbol{x}_i)$ from the sketch, respectively (see Fig. 7). Note that some very simple sketches are

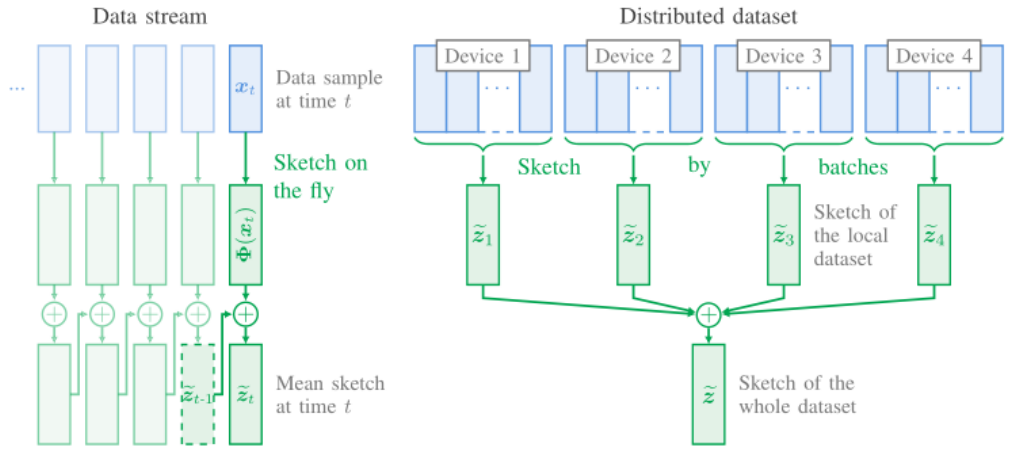


Fig. 7: Left: A streaming scenario, where the data samples are sketched one-by-one, and the mean sketch is updated at each time. Right: A distributed scenario, where each device computes a local sketch, and a centralized entity further averages these local sketches.

not linear: for instance, it is easy to sketch the maximum running value in a stream of scalar data by computing the maximum of the current sketch and each new data point, but this sketching procedure is not linear [20, Chap. 5.2.1]. In particular, this "max" sketch facilitates the insertion of a new element in the database, but not the deletion of an existing one.

Remark. *In the subsections that follow, we describe linear sketches from other points of view. Before doing that, however, we want to clarify that the terms "sketching" and "linear sketching" appear in many other fields, although with meanings that differ considerably from ours. For instance, sketching is often used to indicate* ***linear*** *dimensionality reduction in n or d, as attained by multiplying the data matrix $[\boldsymbol{x}_1, \dots, \boldsymbol{x}_n]^\top \in \mathbb{R}^{n \times d}$ by a random matrix on the left or the right [22–25]. Sketching may also refer to the use of sampling-based approaches [26], such as coresets [27] or the Nyström method [28]. These latter methods differ from linear sketches in AQP, which is our notion of sketching, in that these latter methods generally do not respect the concatenation condition described earlier and are therefore less directly amenable to streaming scenarios. Moreover, these methods typically do not involve a* ***nonlinear*** *feature map $\boldsymbol{\Phi}(\cdot)$, which is a key component of our sketch.*

3.2 Sketches as (randomized) generalized moments

In signal processing and machine learning, the data samples $\boldsymbol{x}_i$ generally live in the vector space $\mathbb{R}^d$ and are often modeled as independent and identically distributed (i.i.d.) random vectors having a *probability distribution* with density p_X. Consider what happens when the number of samples n goes to infinity. The strong law of large numbers says that

$$\lim_{n\to\infty} \frac{1}{n} \sum_{i=1}^{n} \boldsymbol{\Phi}(\boldsymbol{x}_i) \overset{\text{w.p.1}}{=} \mathbb{E}[\boldsymbol{\Phi}(X)] = \int p_X(\boldsymbol{x}) \boldsymbol{\Phi}(\boldsymbol{x}) \, \mathrm{d}\boldsymbol{x}, \tag{4}$$

where $\mathbb{E}[\cdot]$ denotes expectation with respect to the probability density p_X. If we consider the simple case of dimension $d = 1$ and the scalar transformation $\boldsymbol{\Phi}(x) = x^k$ (so that $m = 1$), then $\mathbb{E}[\boldsymbol{\Phi}(X)]$ is the (uncentered) k-th *moment* of the random variable X, a quantity that has a long history in statistics. By

analogy, with a generic vector-valued feature map $\boldsymbol{\Phi}(\cdot)$ and in dimension $d > 1$, quantities of the form $\mathbb{E}[\boldsymbol{\Phi}(X)]$ are known as *generalized moments* of the random vector $X \in \mathbb{R}^d$.

Performing inference from generalized moments is often referred to as the *Generalized Method of Moments* (GeMM) [29]. This method to "learn from a sketch" is very popular in, *e.g.*, the field of econometrics [29, Chap. 1]. The GeMM can be seen as an alternative to maximum likelihood estimation that avoids the need to work with the full likelihood function, which can have computational benefits. Indeed, for many classes of probability distributions, such as heavy-tailed α-stable distributions, the likelihood function is not given in closed form, but generalized moments are given in closed form for appropriately chosen feature maps $\boldsymbol{\Phi}(\cdot)$ (see Box 1).

GeMM differs from compressive learning in several aspects. In GeMM, the feature map $\boldsymbol{\Phi}(\cdot)$ is typically constructed to make the parameter estimates computable in closed form. This narrows the range of learning tasks that GeMM can handle. In compressive learning, $\boldsymbol{\Phi}(\cdot)$ is designed with information-preservation in mind. Consequently, the range of learning tasks is much broader in compressive learning, although the estimation procedure (i.e. *learning from a sketch*) may be algorithmic in nature. Another difference is that, in compressive learning, $\boldsymbol{\Phi}(\cdot)$ is typically randomized. This results in *randomized generalized moments*, which are rarely seen in GeMM.

3.3 COMPRESSIVE LEARNING AND COMPRESSIVE SENSING

As we will show in this section, the sketching mechanism (1) can be interpreted as a dimensionality-reducing linear "projection" *of the probability distribution underlying the data set* $\{\boldsymbol{x}_i\}_{i=1}^n$. This differs from the traditional approach in signal processing, where dimensionality reduction is performed on the features $\boldsymbol{x}_i \in \mathbb{R}^d$ themselves, rather than the distribution that generates them. Still, many of the intuitions, analyses, and tools designed for feature-based dimensionality reduction can be extended to distribution-based dimensionality reduction. Some details are now provided.

In the field of *inverse problems* and *compressive sensing* (CS) [6, 30], a signal-of-interest is modeled as a high-dimensional vector $\boldsymbol{x} \in \mathbb{R}^d$, and a physical measurement of that signal is approximated by a linear transformation plus additive noise: $\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{e} \in \mathbb{R}^m$ (see Box 4). Here, the linear measurement operator is represented by the $m \times d$ matrix $\boldsymbol{A}$. In many applications, there is a great motivation to make the measurement dimension much less than the signal dimension, *i.e.*, $m \ll d$. In this case, the recovery of $\boldsymbol{x}$ from $\boldsymbol{y}$ is generally ill-posed. Still, it is possible to accurately recover $\boldsymbol{x}$ from $\boldsymbol{y}$ when both $\boldsymbol{x}$ and $\boldsymbol{A}$ obey certain properties and the noise is small enough.

The celebrated Johnson Lindenstrauss (JL) lemma and its variations [22, 31] specifies, for instance, that with high probability one can nearly preserve the pairwise distances between N features in $\mathbb{R}^d$ by linearly projecting them in a $O(\log N)$-dimensional domain. This is used, for instance, to accelerate nearest-neighbor searches in large databases. An important extension is to compressive sensing: if $\boldsymbol{x}$ is *sparse*, in that relatively few of its coefficients deviate significantly from zero, and if $\boldsymbol{A}$ satisfies the restricted isometry property (RIP)—an extension of the JL lemma to the continuous set of sparse signals—then one can pose a *regularized* inverse problem whose solution is close to the true $\boldsymbol{x}$ (see Box 4).

[Box 4] Compressive Sensing and the Restricted Isometry Property

In compressive sensing, one observes a linear measurement $\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x} \in \mathbb{R}^m$ of signal $\boldsymbol{x} \in \mathbb{R}^d$ with $m \ll d$, *i.e.*, with significantly reduced dimension. (For simplicity, we focus on the noiseless case for now.) To distinguish between different signals $\boldsymbol{x}$ in a given signal class, one desires that the distances

between all signals in that class are preserved by the measurement operator $\boldsymbol{A}$. For the class of k-sparse signals Σ_k, *i.e.*, signals with at most k non-zero entries, this property is satisfied up to a tolerance $\delta \in [0, 1)$ when $\boldsymbol{A}$ obeys the restricted isometry property (RIP) [6, 30][32]

$$(1-\delta)\|\boldsymbol{x}-\boldsymbol{x}'\|^2 \leq \|\boldsymbol{A}\boldsymbol{x}-\boldsymbol{A}\boldsymbol{x}'\|^2 \leq (1+\delta)\|\boldsymbol{x}-\boldsymbol{x}'\|^2, \ \forall \boldsymbol{x}, \boldsymbol{x}' \in \Sigma_k. \tag{5}$$

One way to create a RIP-satisfying $\boldsymbol{A}$ is to draw it randomly. For example, if the coefficients of $\boldsymbol{A}$ are drawn i.i.d. zero-mean Gaussian, then $\boldsymbol{A}$ will satisfy the RIP *with high probability* (w.h.p.) when the sketch dimension m is at least on the order of $k \log(d/k)$ [6].

To recover k-sparse $\boldsymbol{x}$ from $\boldsymbol{y}$, one might attempt to search for the sparsest signal among all of those that agree with the measurements, *i.e.*, within the set $\mathcal{C}_{\boldsymbol{y}} := \{\boldsymbol{u} : \boldsymbol{A}\boldsymbol{u} = \boldsymbol{y}\}$. The complexity of this search, however, grows exponentially in k. Fortunately, when $\boldsymbol{A}$ satisfies the RIP, one can provably recover the true $\boldsymbol{x}$ using polynomial-complexity methods [6]. One approach is to solve the convex problem of finding the signal with the smallest ℓ_1-norm within $\mathcal{C}_{\boldsymbol{y}}$. Another is to use a greedy algorithm, like orthogonal matching pursuit (OMP), which estimates $\boldsymbol{x}$ by progressively removing from $\boldsymbol{y}$ the k columns of $\boldsymbol{A}$ that best "align" with it, using a least-squares fit.

The RIP also provides guarantees on robust recovery. Suppose that we have noisy measurements $\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{e}$, with noise $\boldsymbol{e}$ of bounded norm $\|\boldsymbol{e}\| \leq \varepsilon$. In addition, suppose that $\boldsymbol{x}$ is only *approximately* k-sparse, and use $\boldsymbol{x}_k$ to denote the best k-sparse approximation of $\boldsymbol{x}$. Finally, consider recovering an estimate $\widehat{\boldsymbol{x}}$ of $\boldsymbol{x}$ by searching for the signal with smallest ℓ_1-norm that agrees with the measurements up to a tolerance of ε, *i.e.*, within the set $\mathcal{C}_{\boldsymbol{y},\varepsilon} := \{\boldsymbol{u} : \|\boldsymbol{A}\boldsymbol{u} - \boldsymbol{y}\| \leq \varepsilon\}$. Then, if the RIP (5) holds, the estimation error $\widehat{\boldsymbol{x}} - \boldsymbol{x}$ satisfies [6][33]

$$\|\widehat{\boldsymbol{x}} - \boldsymbol{x}\| \leq C \frac{\|\boldsymbol{x} - \boldsymbol{x}_k\|_1}{\sqrt{k}} + D\varepsilon, \tag{6}$$

where constants $C, D > 0$ depend only on the value of δ that appears in (5). Thus, the estimation error increases linearly with the noise level ε and the deviation $\|\boldsymbol{x} - \boldsymbol{x}_k\|_1$ from perfect sparsity.

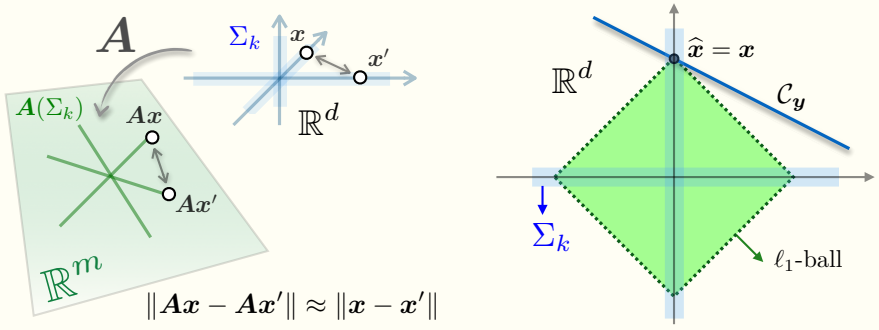


Fig. 8: (left) Geometrical interpretation of the RIP. (right) 2-D illustration of the recoverability of $\boldsymbol{x}$ from $\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x}$ by finding the vector $\widehat{\boldsymbol{x}}$ in $\mathcal{C}_{\boldsymbol{y}}$ with smallest ℓ_1-norm.

Now that CS has been described, we can clearly connect it to compressive learning. Recall that the sketch (1) $\widetilde{\boldsymbol{z}}$ converges to the generalized moment $\boldsymbol{z} := \mathbb{E}[\boldsymbol{\Phi}(X)] = \int p_X(\boldsymbol{x})\boldsymbol{\Phi}(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x}$ as the number of data samples n tends to infinity, as per (4).

A crucial observation is the following: *due to the linearity of integration, the sketch $\boldsymbol{z}$ depends linearly on the probability density p_X*. To see it another way, consider a mixture of two densities, $p_X = \alpha p_{X_1} +$

$(1-\alpha)p_{X_2}$. The corresponding expectation satisfies

$$\mathbb{E}_{X\sim p_X}[\boldsymbol{\Phi}(X)] = \alpha\mathbb{E}_{X_1\sim p_{X_1}}[\boldsymbol{\Phi}(X_1)] + (1-\alpha)\mathbb{E}_{X_2\sim p_{X_2}}[\boldsymbol{\Phi}(X_2)], \tag{7}$$

which implies that the generalized moment $\boldsymbol{z}$ is linear in p_X. With this understanding, we can write

$$\boldsymbol{z} = \mathcal{A}(p_X) := \mathbb{E}_{X\sim p_X}[\boldsymbol{\Phi}(X)], \tag{8}$$

where $\mathcal{A}$ is a linear operator mapping the probability distribution p_X to the m-dimensional sketch vector $\boldsymbol{z}$.

Remark. *Although $\mathcal{A}$ is a linear function of p_X, we emphasize that the feature map $\boldsymbol{\Phi}(\boldsymbol{x})$ is generally* not *a linear function of $\boldsymbol{x}$. This makes compressive learning concretely very different from the vast majority of existing "sketching" mechanisms, which use random* linear *projections of the data for dimensionality reduction.*

With a small modification of the above arguments, we can handle the finite-sample case. Consider the difference between the true generalized moment $\boldsymbol{z}$ and the empirical moment $\widetilde{\boldsymbol{z}}$, *i.e.*,

$$\frac{1}{n}\sum_{i=1}^{n}\boldsymbol{\Phi}(\boldsymbol{x}_i) - \mathbb{E}[\boldsymbol{\Phi}(X)] =: \boldsymbol{e}. \tag{9}$$

As we discussed earlier, $\boldsymbol{e}$ converges to zero as n tends to infinity by the law of large numbers. Combining (1), (8), and (9), we obtain

$$\widetilde{\boldsymbol{z}} = \mathcal{A}(p_X) + \boldsymbol{e}, \tag{10}$$

which shows that the sketch (1) can be interpreted as *a "noisy" observation of the data distribution p_X through the linear measurement operator $\mathcal{A}$.* Under mild conditions[2], the central limit theorem can be used to show that $\|\boldsymbol{e}\|$ decays as $1/\sqrt{n}$ w.h.p. [6, Chapter 8]. With the above interpretation of compressive learning, one recovers all of the traditional ingredients of CS:

- The measurement operator $\mathcal{A}$ is linear.
- The measurements $\widetilde{\boldsymbol{z}}$ are drastically dimension-reduced. In mathematical terms, probability distributions p_X belong to the infinite-dimensional vector space of so-called *finite* measures [34][35]. The operator $\mathcal{A}$ maps these infinite-dimensional objects to vectors of finite dimension m.
- The measurement operator $\mathcal{A}$ is typically designed using randomness. This is accomplished by choosing an appropriate randomized feature map $\boldsymbol{\Phi}(\cdot)$, such as one based on RF features, as in (3).
- The measurements $\widetilde{\boldsymbol{z}}$ are noisy, as per (10).

The analogy between compressive learning and CS is illustrated in Fig. 9. The CS analog to the *sketching phase* (1) is the *signal sensing phase*, where the signal $\boldsymbol{x}$ is *linearly* mapped to the observation vector $\boldsymbol{y}$. The analog to *sparse recovery*, where an estimate of $\boldsymbol{x}$ is computed from the observation $\boldsymbol{y}$ by solving an optimization problem, is to *learn from a sketch*, where an estimate of the data distribution p_X (or of distributional parameters $\boldsymbol{\theta}$ of interest) is computed from the sketch $\widetilde{\boldsymbol{z}}$. Its expression as an optimization problem will be further discussed in Section 4.

3.4 Random Fourier sampling and super-resolution recovery

In this section, we consider the specific case of compressive clustering, which allows us to forge a concrete connection between CS and compressive learning using random Fourier sampling and super-

[2]From the assumed independence of the data samples, this happens for instance if $\mathbb{P}[\|\boldsymbol{\Phi}(X)\| \geq t]$ decays exponentially fast when t increases.

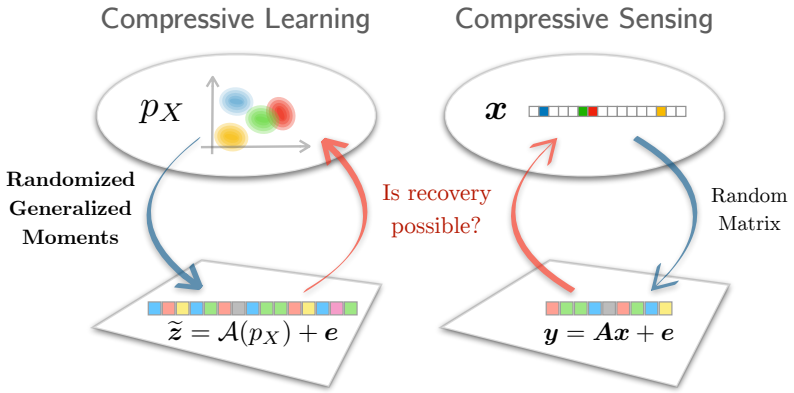


Fig. 9: The analogy between compressive learning, which uses the dimensionality reducing linear measurement $\mathcal{A}(p_X) = \mathbb{E}[\boldsymbol{\Phi}(X)]$ of a distribution p_X, and compressive sensing, which uses the dimensionality reducing linear measurement $\boldsymbol{A}\boldsymbol{x}$ of a signal $\boldsymbol{x}$.

resolution recovery.

We begin by considering the goal of recovering k centroids $\{\boldsymbol{c}_\ell\}_{\ell=1}^k$ in $\mathbb{R}^d$ with norm $\leq r$ that are separated from each other by $\geq \varepsilon$. A naive approach would be to discretize the d-dimensional cube of side-length $2r$ with a grid spacing of ϵ, leading to $N = (2r/\epsilon)^d$ bins. A valid sketch of the data $\mathcal{X}$ is obtained by simply computing the histogram $\widehat{\boldsymbol{p}} \in \mathbb{R}_+^N$ over these bins (*i.e.*, by using the "binning" feature map). However, the dimension of this sketch, N, grows exponentially in the feature dimension, d. To construct a smaller sketch, one might reason that, if the data clusters tightly around k points, then $\widehat{\boldsymbol{p}}$ is close to a k-sparse vector. In this case, ideas from CS can be directly exploited. In particular, one could use a sketched histogram [20][36] of the form $\widetilde{\boldsymbol{z}} = \boldsymbol{A}\widehat{\boldsymbol{p}}$, where matrix $\boldsymbol{A} \in \mathbb{R}^{m \times N}$ is randomly drawn with i.i.d. Gaussian components. Here, "learning from a sketch" means recovering the centroids. For this, one would first search for the best non-negative, k-sparse, sum-to-one vector using

$$\widetilde{\boldsymbol{p}} = \underset{\boldsymbol{p} \in \Sigma_k^+}{\arg\min} \|\widetilde{\boldsymbol{z}} - \boldsymbol{A}\boldsymbol{p}\|^2, \tag{11}$$

(or a convex or greedy relaxation of this problem) where Σ_k^+ here denotes the set of k-sparse, non-negative, sum-to-one vectors. Then, one would identify the k grid locations in $\mathbb{R}^d$ corresponding to the non-zero indices of $\widetilde{\boldsymbol{p}}$. CS theory [6] says that the support of $\widetilde{\boldsymbol{p}}$ will be accurate for sketch dimensions m at least on the order of $k \log N$, *i.e.*, at least on the order of $kd \log(r/\epsilon)$ Although this latter approach substantially reduces the dimension of the sketch, practical challenges remain when the feature dimension d is large. For example, the number of columns, N, in the compression matrix $\boldsymbol{A}$ grows exponentially in d, making storage and multiplication by $\boldsymbol{A}$ impractical.

An alternative approach could be to construct $\boldsymbol{A}$ using m rows of the (d-dimensional, in this case) discrete Fourier transform (DFT) matrix, *i.e.*, by sampling the DFT at m (d-dimensional) frequencies. In this case, multiplication-by-$\boldsymbol{A}$ could be implemented by the fast Fourier transform (FFT) algorithm, and thus the matrix $\boldsymbol{A}$ would not need to be explicitly stored. Fourier-domain sampling is a familiar operation in the context of signal processing, as it forms the cornerstone for radar, medical imaging, and radio interferometry, see *e.g.*, [37, 38] [39–41]. When the m frequencies are drawn uniformly at random, CS theory [6, 30][33] has established that accurate recovery of an N-length k-sparse signal can be accomplished (with high probability) when m is on the order of $k \log^3(k) \log N$ [6, Corollary 12.38]. Since $N = (2r/\epsilon)^d$

here, this would mandate sketch dimensions m at least on the order of $kd\log^3(k)\log(r/\epsilon)$. Although the FFT avoids the need to store $\boldsymbol{A}$ as an explicit matrix and allows efficient computation of the sketch, the cost of solving the optimization problem (11) using existing convex relaxations or greedy approaches is impractical due to the need to manipulate N-dimensional vectors, where N grows exponentially in d.

Until now, we considered discretizing the d-dimensional feature space on an ϵ-spaced hypergrid within a $2r$-sidelength hypercube, but found that this requires manipulating vectors (*e.g.*, a histogram $\widehat{\boldsymbol{p}}$) whose dimension grows exponentially in d. We can avoid this discretization (*i.e.*, take $\epsilon \to 0$ and $r \to \infty$) by replacing the DFT with the *continuous* Fourier transform, in which case we are Fourier-transforming the empirical distribution $\widehat{p}_{\mathcal{X}}(\boldsymbol{x}) := \frac{1}{n}\sum_{i=1}^{n}\delta(\boldsymbol{x}-\boldsymbol{x}_i)$ rather than its N-bin histogram $\widehat{\boldsymbol{p}}$. The sketch $\widetilde{\boldsymbol{z}}$ then has components

$$\widetilde{z}_j = \int_{\mathbb{R}^d} \widehat{p}_{\mathcal{X}}(\boldsymbol{x}) \exp(-\mathrm{j}2\pi \boldsymbol{w}_j^\top \boldsymbol{x})\,\mathrm{d}\boldsymbol{x} \tag{12}$$

$$= \frac{1}{n}\sum_{i=1}^{n} \exp(-\mathrm{j}2\pi \boldsymbol{w}_j^\top \boldsymbol{x}_i) \qquad j = 1,\ldots,m, \tag{13}$$

where $\{\boldsymbol{w}_j\}_{j=1}^m$ are d-dimensional frequencies that are drawn at random. Typically, $\boldsymbol{w}_j$ are drawn i.i.d. Gaussian, but other distributions can be used, as discussed in Sections 5.6 and 6.1. Note that (13) corresponds precisely to the sketch (1) with the random Fourier (RF) feature map $\boldsymbol{\Phi}(\cdot)$ from (3). Taking a statistical perspective, the components $\widetilde{z}_j$ in (12) can also be recognized as samples of the *characteristic function* of $\widehat{p}_{\mathcal{X}}$. Recall that, for a density p, the characteristic function Ψ_p is defined as (the complex conjugate of) its Fourier transform, *i.e.*,

$$\Psi_p(\boldsymbol{w}) := \int_{\mathbb{R}^d} p(\boldsymbol{x}) \exp(\mathrm{j}2\pi \boldsymbol{w}^\top \boldsymbol{x})\,\mathrm{d}\boldsymbol{x} = \mathbb{E}_{X\sim p}[\exp(\mathrm{j}2\pi \boldsymbol{w}^\top X)]. \tag{14}$$

[Box 5] Super-Resolution Recovery

Super-resolution (SR) is a general class of techniques to enhance the resolution of a sensing system, *e.g.*, to observe sub-wavelength features in astronomy or medical imaging [34]. The problem addressed by SR is to recover a continuous-time (when dimension $d = 1$), or continuous-space (when dimension $d \geq 2$) sparse signal $s(\boldsymbol{t})$ from a few, possibly noisy Fourier measurements $\{y_j\}_{j=1}^m$. This amounts to recovering a weighted sum $s(\boldsymbol{t}) = \sum_{\ell=1}^{k}\alpha_\ell \delta(\boldsymbol{t}-\boldsymbol{t}_\ell)$ of k Diracs with amplitudes α_ℓ and locations $\boldsymbol{t}_\ell \in \mathbb{R}^d$ from

$$y_j = \int_{\mathbb{R}^d} s(\boldsymbol{t}) \exp(-\mathrm{j}2\pi \boldsymbol{w}_j^\top \boldsymbol{t})\,\mathrm{d}\boldsymbol{t} + e_j, \qquad j = 1,\ldots,m, \tag{15}$$

with measurement noise e_j and frequency vectors $\{\boldsymbol{w}_j\}_{j=1}^m$ in $\mathbb{R}^d$. Recovery of $s(\cdot)$ can be posed as an infinite-dimensional convex problem on measures [34] [35]. However, most reconstruction algorithms involve non-convex steps [42]. When the frequencies $\boldsymbol{w}_j$ are drawn randomly, the signal can be accurately recovered with high probability when m is of the order of at least kd^3, up to log factors. Proving this usually requires additional assumptions, such as a minimal separation between the locations $\boldsymbol{t}_\ell$ [43] or positivity of the amplitudes α_ℓ [42].

The link between super-resolution and compressive clustering follows from rewriting (12) as

$$\widetilde{z}_j = \int_{\mathbb{R}^d} p_X(\boldsymbol{x}) \exp(-\mathrm{j}2\pi \boldsymbol{w}_j^\top \boldsymbol{x})\,\mathrm{d}\boldsymbol{x} + e_j, \qquad j = 1,\ldots,m, \tag{16}$$

where e_j captures the "noise" due to finite-sample effects (recall (9)). Comparing (16) to (15), we see

that they are mathematically equivalent when $p_X(\boldsymbol{x}) = \sum_{\ell=1}^{k} \alpha_\ell \delta(\boldsymbol{x} - \boldsymbol{c}_\ell)$, except for the fact that, in the case of compressive clustering, α_ℓ are non-negative and sum to one.

Intuitively, when a probability distribution p has a "simple" structure, one can recover it (with high probability) from enough randomly chosen samples of its Fourier transform. Centroid recovery from the sketch (12) is premised on the empirical distribution $\widehat{p}_{\mathcal{X}}$ being well approximated by a mixture of k Diracs, *i.e.*, $\widehat{p}_{\mathcal{X}}(\boldsymbol{x}) \approx \sum_{\ell=1}^{k} \alpha_\ell \delta(\boldsymbol{x} - \boldsymbol{c}_\ell)$. In this case, centroid recovery parallels the "super-resolution" recovery problem (see Box 5) through an optimization problem that is the continuous analog to (11) (or a convex or greedy relaxation for the continuous case) and will be further elaborated in the next section.

In both problems, recovery guarantees are possible when the frequencies $\boldsymbol{w}_j$ are randomly drawn. For example, when the centroids $\{\boldsymbol{c}_\ell\}_{\ell=1}^{k}$ are ϵ-separated and r-bounded, centroid recovery guarantees have been established provided the sketch dimension m is on the order of $k^2 d \log(r/\epsilon)$, omitting—for simplicity—some log factors involving k and d [8]. Similar guarantees hold when $\widehat{p}_{\mathcal{X}}$ is approximately a sum of spatially localized components (*e.g.*, in compressive GMM) [8].

4 LEARNING FROM A SKETCH

Until now, we have primarily focused on the first stage of the compressive-learning pipeline (see Fig. 1), where the dataset $\mathcal{X}$ is sketched down to $\widetilde{\boldsymbol{z}}$, a compressed and noisy representation of the underlying data-generating distribution p_X. We now discuss the second stage of the pipeline, where the distributional parameters of interest, $\boldsymbol{\theta}$, are recovered from the sketch $\widetilde{\boldsymbol{z}}$. The close analogy between sketching and CS allows us to cast this "parameter learning" stage as an optimization problem, *i.e.*,

$$\widetilde{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} C(\boldsymbol{\theta} | \, \widetilde{\boldsymbol{z}}), \tag{17}$$

where the cost function $C(\cdot | \, \widetilde{\boldsymbol{z}})$ is adapted to the considered learning task. As in CS, many candidate distributions p_X (and hence many candidate parameters $\boldsymbol{\theta}$) can yield the same sketch $\widetilde{\boldsymbol{z}}$. Thus, to make the inverse problem well-posed, one needs to employ *concrete modeling assumptions* and *regularization*, both of which can take several forms. As in CS, we will assume that the sketched quantity p_X is of low intrinsic complexity, *i.e.*, close to some family of "simple" probability distributions.

As a first example, we consider the problem of learning a mixture model from a sketch. Similar to a sparse vector $\boldsymbol{x}$, which is a linear combination of a few elements of the standard basis, a *mixture model* p_X is a linear combination of a few "simple" densities $\{p_{\boldsymbol{\theta}_\ell}\}_{\ell=1}^{k}$. Concretely, $p_X = \sum_{\ell=1}^{k} \alpha_\ell p_{\boldsymbol{\theta}_\ell}$, where the mixture weights $\{\alpha_\ell\}_{\ell=1}^{k}$ are non-negative and sum to one. For example, with a Gaussian Mixture Model (GMM), we have that $p_{\boldsymbol{\theta}_\ell} = \mathcal{N}(\boldsymbol{\mu}_\ell, \boldsymbol{\Sigma}_\ell)$, where $\boldsymbol{\theta}_\ell = \{\boldsymbol{\mu}_\ell, \boldsymbol{\Sigma}_\ell\}$ contains the mean $\boldsymbol{\mu}_\ell$ and covariance $\boldsymbol{\Sigma}_\ell$. If p_X is well approximated by the mixture model $\sum_{\ell=1}^{k} \alpha_\ell p_{\boldsymbol{\theta}_\ell}$, then, according to (10), the sketch $\widetilde{\boldsymbol{z}}$ is well approximated by the linear combination $\sum_{\ell=1}^{k} \alpha_\ell \mathcal{A}(p_{\boldsymbol{\theta}_\ell})$. Hence, one could try to extract the mixture parameters, $\boldsymbol{\theta} = \{\alpha_\ell, \boldsymbol{\theta}_\ell\}_{\ell=1}^{k}$, from the sketch $\widetilde{\boldsymbol{z}}$ by solving the (nonconvex) optimization problem (17) with

$$C(\boldsymbol{\theta} | \, \widetilde{\boldsymbol{z}}) := \left\| \widetilde{\boldsymbol{z}} - \sum_{\ell=1}^{k} \alpha_\ell \mathcal{A}(p_{\boldsymbol{\theta}_\ell}) \right\|^2. \tag{18}$$

The cost $C(\boldsymbol{\theta} | \, \widetilde{\boldsymbol{z}})$ can be interpreted as the negative log-likelihood (up to a shift and scale) of $\boldsymbol{\theta}$ given the sketch $\widetilde{\boldsymbol{z}}$, under the classic modeling assumption of i.i.d. Gaussian measurement noise $\boldsymbol{e}$ in (10).

When the RF feature map $\boldsymbol{\Phi}(\cdot)$ is used to compute the sketch $\widetilde{\boldsymbol{z}}$ and the component densities $p_{\boldsymbol{\theta}_\ell}$ are Gaussian or α-stable, there exist analytic expressions for $\mathcal{A}(p_{\boldsymbol{\theta}_\ell})$ and for the gradient of $\mathcal{A}(p_{\boldsymbol{\theta}_\ell})$ with respect to the mixture parameters in $\boldsymbol{\theta}_\ell$ [11]. These expressions are convenient when numerically optimizing (18). For instance, greedy approaches, similar to the OMP algorithm for CS (recall Box 4), can be used [11] to estimate the parameters $\{\alpha_\ell, \boldsymbol{\theta}_\ell\}_{\ell=1}^k$. These approaches sequentially estimate and subtract, from $\widetilde{\boldsymbol{z}}$, each of the k components $\alpha_\ell \mathcal{A}(p_{\boldsymbol{\theta}_\ell})$ that best align with it, where "best" is measured via the correlation between $\widetilde{\boldsymbol{z}}$ and $\mathcal{A}(p_{\boldsymbol{\theta}_\ell})$. As another example, an iterative approach [17] was proposed that exploits the log-likelihood interpretation of $C(\boldsymbol{\theta}|\,\widetilde{\boldsymbol{z}})$ in (18) and the i.i.d. random nature of the linear transform $\boldsymbol{W}$ in the RF map (3). In [11] and [13], applications of compressive mixture-modeling are demonstrated on speaker verification (see Box 1) and source separation.

As a second example, we consider compressive k-means clustering. Here, the goal is to recover, from the sketch $\widetilde{\boldsymbol{z}}$, the centroids $\boldsymbol{\theta} = \{\boldsymbol{c}_\ell\}_{\ell=1}^k$ that minimize the average squared Euclidean distance from each sample to its nearest centroid, *i.e.*, $\frac{1}{n}\sum_{i=1}^n \min_\ell \|\boldsymbol{x}_i - \boldsymbol{c}_\ell\|^2$. To tackle this k-means problem, we view it as an approximation of a particular GMM fitting problem. In particular, suppose that the probability distribution p_X is a GMM with weights α_ℓ, mean vectors $\boldsymbol{c}_\ell$, and covariance matrices $\boldsymbol{\Sigma}_\ell$. Then, in the special case that $\alpha_\ell = 1/k$ and $\boldsymbol{\Sigma}_\ell = \sigma^2 \boldsymbol{I}$ for all components $1 \leq \ell \leq k$, we can write the likelihood as $\prod_{i=1}^n p(\boldsymbol{x}_i|\boldsymbol{\theta}) \propto \prod_{i=1}^n \sum_{\ell=1}^k \exp(-\frac{1}{2\sigma^2}\|\boldsymbol{x}_i - \boldsymbol{c}_i\|^2)$, and so the negative log-likelihood becomes $-\sum_{i=1}^n \log \sum_{\ell=1}^k \exp(-\frac{1}{2\sigma^2}\|\boldsymbol{x}_i - \boldsymbol{c}_i\|^2)$ up to an additive constant. We can then use the log-sum-exp approximation $\log \sum_\ell \exp(f_\ell(\boldsymbol{x})) \approx \max_\ell f_\ell(\boldsymbol{x})$ to approximate this latter expression as $\frac{1}{2\sigma^2}\sum_{i=1}^n \min_\ell \|\boldsymbol{x}_i - \boldsymbol{c}_\ell\|^2$, which agrees with the k-means cost up to a scaling. If we furthermore consider the case of a vanishing variance $\sigma^2 \to 0$, then the component density $p_{\boldsymbol{\theta}_\ell}$ reduces to a point mass, *i.e.*, $p_{\boldsymbol{\theta}_\ell}(\boldsymbol{x}) \to \delta(\boldsymbol{x} - \boldsymbol{c}_\ell)$. In this limiting case, the linear measurement of the point-mass $p_{\boldsymbol{\theta}_\ell}$ is $\mathcal{A}(p_{\boldsymbol{\theta}_\ell}) = \mathbb{E}_{X\sim p_{\boldsymbol{\theta}_\ell}}[\boldsymbol{\Phi}(X)] = \int \boldsymbol{\Phi}(\boldsymbol{x}) p_{\boldsymbol{\theta}_\ell}(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} = \boldsymbol{\Phi}(\boldsymbol{c}_\ell)$ according to (8).

Thus, with these justifications, the cost function (18) for compressive GMM would change to

$$C(\boldsymbol{\theta}|\,\widetilde{\boldsymbol{z}}) := \left\| \widetilde{\boldsymbol{z}} - \frac{1}{k}\sum_{\ell=1}^k \boldsymbol{\Phi}(\boldsymbol{c}_\ell) \right\|^2 \tag{19}$$

for compressive k-means. If we do not want to assume that $\alpha_\ell = 1/k$ for each ℓ, we could instead estimate $\{\alpha_\ell\}_{\ell=1}^k$ from the sketch, leading to the cost function suggested in [16]:

$$C(\boldsymbol{\theta}|\,\widetilde{\boldsymbol{z}}) := \min_{\boldsymbol{\alpha}} \left\| \widetilde{\boldsymbol{z}} - \sum_{\ell=1}^k \alpha_\ell \boldsymbol{\Phi}(\boldsymbol{c}_\ell) \right\|^2. \tag{20}$$

Similar to the compressive GMM problem described earlier, minimization of the cost function (20) for compressive k-means can be tackled by greedy approaches, as described in [16]. Despite the fact that (20) does not directly minimize the k-means cost, it has been shown empirically [16] that the centroids estimated by such greedy algorithms nearly minimize this cost, see, *e.g.*, Box 2 for an example on MNIST data. This claim is also supported by theoretical results guaranteeing that the minimizer of (20) is endowed with statistical-learning guarantees with respect to the original k-means cost [8]. Such guarantees will be discussed shortly.

Depending on the choice of parameters $\boldsymbol{\theta}$ and the feature map $\boldsymbol{\Phi}(\cdot)$, the form of the optimization problem posed to learn-from-a-sketch can differ considerably from that for GMMs in (18) and that for k-means in (19)-(20). Consider, for example, learning-from-a-sketch for PCA. As described earlier, the parameter $\boldsymbol{\theta}$ of interest is the k-dimensional subspace that best fits the d-dimensional data $\{\boldsymbol{x}_i\}_{i=1}^n$ in a least-

squares sense. It is well known that this subspace is spanned by k principal eigenvectors of the empirical autocorrelation matrix $\widehat{\boldsymbol{R}} = \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{x}_i \boldsymbol{x}_i^\top$, or—equivalently—the column space of the (symmetric) matrix $\widehat{\boldsymbol{R}}_k$ that is closest to $\widehat{\boldsymbol{R}}$ in the Frobenius norm:

$$\widehat{\boldsymbol{R}}_k := \underset{\boldsymbol{R}:\, \mathrm{rank}(\boldsymbol{R}) \le k}{\arg\min} \|\widehat{\boldsymbol{R}} - \boldsymbol{R}\|_F = \underset{\boldsymbol{R}:\, \mathrm{rank}(\boldsymbol{R}) \le k}{\arg\min} \|\,\mathrm{vec}(\widehat{\boldsymbol{R}}) - \mathrm{vec}(\boldsymbol{R})\|^2. \tag{21}$$

When sketching using quadratic features $\boldsymbol{\Phi}(\boldsymbol{x}) = ((\boldsymbol{w}_1^\top \boldsymbol{x})^2, \ldots, (\boldsymbol{w}_m^\top \boldsymbol{x})^2)$ with random $\boldsymbol{w}_j \in \mathbb{R}^d$, the jth component of the sketch becomes $\widetilde{z}_j = \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{w}_j^\top \boldsymbol{x}_i \boldsymbol{x}_i^\top \boldsymbol{w}_j = \boldsymbol{w}_j^\top \widehat{\boldsymbol{R}} \boldsymbol{w}_j$. Importantly, this $\widetilde{z}_j$ is a linear function of $\widehat{\boldsymbol{R}}$, and so there exists an $m \times d^2$ matrix $\boldsymbol{A}$ such that $\widetilde{\boldsymbol{z}} = \boldsymbol{A}\,\mathrm{vec}(\widehat{\boldsymbol{R}})$. Thus, by analogy with (21), one could first fit a low-rank matrix to the sketch $\widetilde{\boldsymbol{z}}$ via

$$\widetilde{\boldsymbol{R}} = \underset{\boldsymbol{R}:\, \mathrm{rank}(\boldsymbol{R}) \le k,\, \boldsymbol{R}^\top = \boldsymbol{R}}{\arg\min} \|\widetilde{\boldsymbol{z}} - \boldsymbol{A}\,\mathrm{vec}(\boldsymbol{R})\|^2, \tag{22}$$

and then set the parameter estimate $\widetilde{\boldsymbol{\theta}}$ equal to the column space of $\widetilde{\boldsymbol{R}}$.

The low-rank matrix recovery problem (22) has been thoroughly investigated by the signal processing and machine learning communities (*e.g.*, [44]). It arises, for example, in applications such as collaborative filtering for recommender systems and signal reconstruction from phaseless measurements. Early approaches to solving the non-convex problem (22) involved convex relaxation via nuclear-norm regularization [6, Chapter 4]. More recent approaches exploit the non-convex geometry of (22) [45].

5 Compressive learning with theoretical guarantees

In the previous section, learning-from-a-sketch was posed as the optimization problem (17), repeated here for convenience:

$$\widetilde{\boldsymbol{\theta}} = \underset{\boldsymbol{\theta}}{\arg\min}\, C(\boldsymbol{\theta} |\, \widetilde{\boldsymbol{z}}). \tag{23}$$

The quantity $C(\cdot |\, \widetilde{\boldsymbol{z}})$ is a real-valued *cost function* whose minimizer $\widetilde{\boldsymbol{\theta}}$ is the "best" (in some sense) estimate of $\boldsymbol{\theta}$ from the sketch $\widetilde{\boldsymbol{z}}$. This approach contrasts with traditional statistical learning [46, 47], which computes the parameter estimate

$$\widehat{\boldsymbol{\theta}} := \underset{\boldsymbol{\theta}}{\arg\min}\, R(\boldsymbol{\theta} |\, \mathcal{X}) \qquad \text{with } R(\boldsymbol{\theta} |\, \mathcal{X}) := \frac{1}{n}\sum_{i=1}^{n} L(\boldsymbol{\theta} |\, \boldsymbol{x}_i), \tag{24}$$

where $R(\cdot |\, \mathcal{X})$ is the "empirical risk" and $L(\cdot |\, \boldsymbol{x}_i)$ is the "loss function" for the ith sample. Table I presents typical loss functions for our four running examples. In most cases, performing the minimization in (24) involves querying the full training dataset $\mathcal{X}$ many times, *e.g.*, for stochastic gradient descent. In compressive learning, the cost $C(\cdot |\, \widetilde{\boldsymbol{z}})$, which depends only on the low-dimensional sketch $\widetilde{\boldsymbol{z}}$, is used as a *surrogate* for the empirical risk. Because the dimension of the sketch is so much smaller than the cardinality of the full dataset $\mathcal{X}$, the minimization in (17) can be made much more efficient than that in (24).

Of course, the estimate $\widetilde{\boldsymbol{\theta}}$, which minimizes the cost $C(\cdot |\, \widetilde{\boldsymbol{z}})$ is not, in general, the same as $\widehat{\boldsymbol{\theta}}$, which minimizes the empirical risk $R(\cdot |\, \mathcal{X})$. Both, however, are approximations of the *ideal* estimate

$$\boldsymbol{\theta}^\star := \underset{\boldsymbol{\theta}}{\arg\min}\, R^\star(\boldsymbol{\theta}) \qquad \text{with } R^\star(\boldsymbol{\theta}) := \mathbb{E}[L(\boldsymbol{\theta} |\, X)], \tag{25}$$

where $R^\star(\boldsymbol{\theta})$ is known as the "true risk." Indeed, the feature map $\boldsymbol{\Phi}(\cdot)$ and cost $C(\cdot |\, \widetilde{\boldsymbol{z}})$ are designed precisely to ensure that the surrogate minimization (23) approximates the true-risk minimization (25).

TABLE I: Loss functions for our four running examples

Running example	Parameters $\boldsymbol{\theta}$	Loss function $L(\boldsymbol{\theta}\vert\, \boldsymbol{x}_i)$
Principal component analysis	Orthonormal basis $\boldsymbol{\theta} = \{\boldsymbol{u}_\ell\}_{\ell=1}^k$	$\sum_{\ell=1}^k \vert\boldsymbol{x}_i^\top \boldsymbol{u}_\ell\vert^2$
Least-squares linear regression	A weight matrix $\boldsymbol{\theta}$	$\Vert\boldsymbol{x}_{1i} - \boldsymbol{\theta}\boldsymbol{x}_{2i}\Vert^2$, $\boldsymbol{x}_i := (\boldsymbol{x}_{1i}^\top, \boldsymbol{x}_{2i}^\top)^\top$
Gaussian mixture modeling	GMM parameters $\boldsymbol{\theta} = \{\alpha_\ell, \boldsymbol{\mu}_\ell, \boldsymbol{\Sigma}_\ell\}_{\ell=1}^k$	$-\log \sum_{\ell=1}^k \alpha_\ell \mathcal{N}(\boldsymbol{x}_i; \boldsymbol{\mu}_\ell, \boldsymbol{\Sigma}_\ell)$
k-means clustering	A set of centroids $\boldsymbol{\theta} = \{\boldsymbol{c}_\ell\}_{\ell=1}^k$	$\min_\ell \Vert\boldsymbol{x}_i - \boldsymbol{c}_\ell\Vert^2$

5.1 Excess risk guarantees

To establish a guarantee on the goodness of an estimate $\boldsymbol{\theta}$, one must prove that the "excess risk" $R^\star(\boldsymbol{\theta}) - R^\star(\boldsymbol{\theta}^\star) \geq 0$ is small. Indeed, the true risk can be interpreted as the *expected* loss on test samples that have the same generating distribution p_X as the training samples $\mathcal{X}$, but that are drawn independently and not accessible at training time. In statistical learning tasks, the true risk $R^\star(\cdot)$ is the primary metric by which one judges the quality of an arbitrary estimate $\boldsymbol{\theta}$. Note that controlling the excess risk is different from proving that $\boldsymbol{\theta}$ is close to the ideal estimate $\boldsymbol{\theta}^\star$ in, *e.g.*, the Euclidean distance $\Vert\boldsymbol{\theta} - \boldsymbol{\theta}^\star\Vert$. Consider, for example, the problem of fitting a one-dimensional linear subspace to a dataset (*i.e.*, PCA with $k = 1$). When the two largest eigenvalues of the autocorrelation matrix $\boldsymbol{R}$ are equal, any nontrivial linear combination of the corresponding eigenvectors generates a one-dimensional subspace $\boldsymbol{\theta}$ with minimum true risk. The problem of estimating a subspace "close to the optimal one" is thus ill-defined, yet finding a subspace with close-to-optimal performance is well-defined and achievable, both by classic PCA and compressive PCA.

[Box 6] Traditional Statistical Learning versus Compressive Learning

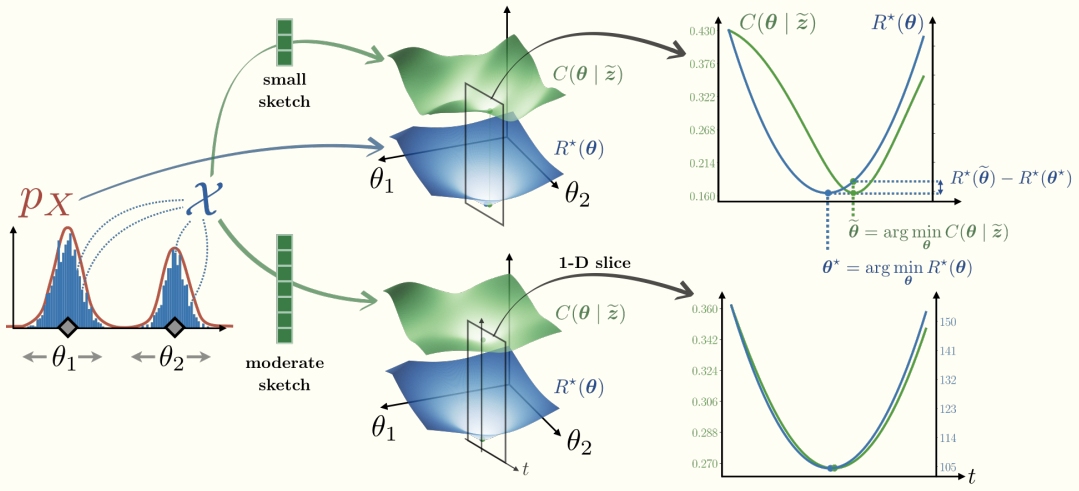


Fig. 10: k-means clustering of one-dimensional data $\mathcal{X}$, showing cost function $C(\boldsymbol{\theta}\vert\, \widetilde{\boldsymbol{z}})$, risk $R^\star(\boldsymbol{\theta})$, ideal estimate $\boldsymbol{\theta}^\star$, compressive learning estimate $\widetilde{\boldsymbol{\theta}}$, and excess risk $R^\star(\widetilde{\boldsymbol{\theta}}) - R^\star(\boldsymbol{\theta}^\star)$, for a sketch of small dimension m (top) and for a sketch of moderate size m (bottom).

In statistical learning, the ideal parameter $\boldsymbol{\theta}^\star$ minimizes the true risk $R^\star(\boldsymbol{\theta}) = \mathbb{E}_{X\sim p_X}[L(\boldsymbol{\theta}\vert\, X)]$. The performance of any estimate $\boldsymbol{\theta}$ is measured according to its excess risk, *i.e.*, $R^\star(\boldsymbol{\theta}) - R^\star(\boldsymbol{\theta}^\star)$. In compressive learning, one obtains $\widetilde{\boldsymbol{\theta}}$ by minimizing a cost function $C(\boldsymbol{\theta}\vert\, \widetilde{\boldsymbol{z}})$, where the sketch $\widetilde{\boldsymbol{z}}$ is a compressed version of a finite-size dataset $\mathcal{X}$ with samples drawn from distribution p_X.

To gain intuition into how these quantities manifest in compressive learning, Fig. 10 shows a simple example: compressive k-means clustering of one-dimensional data $\mathcal{X} = \{x_i\}_{i=1}^n$ with two centroids, θ_1 and θ_2. The true risk $R^\star(\cdot)$ and the cost $C(\cdot\,|\,\widetilde{\boldsymbol{z}})$ are plotted versus $\boldsymbol{\theta} = (\theta_1, \theta_2)$ and versus a 1-D slice in the $\boldsymbol{\theta}$ plane. As the sketch dimension m increases, it can be seen that the excess risk $R^\star(\widetilde{\boldsymbol{\theta}}) - R^\star(\boldsymbol{\theta}^\star)$ decreases. Moreover, although the cost function $C(\cdot\,|\,\widetilde{\boldsymbol{z}})$ is nonconvex, it is approximately quadratic in a large basin of attraction around its global minimizer, suggesting that gradient-descent algorithms will behave well when properly initialized.

Despite the fact that the estimates $\widetilde{\boldsymbol{\theta}}$, $\widehat{\boldsymbol{\theta}}$, and $\boldsymbol{\theta}^\star$ minimize different objective functions, they often yield similar true-risks, *i.e.*, the excess risks of $\widetilde{\boldsymbol{\theta}}$ and $\widehat{\boldsymbol{\theta}}$ are provably small. For $\widehat{\boldsymbol{\theta}}$, the proofs use classic results from statistical learning [46, 47], under assumptions that we will briefly discuss in the sequel. For $\widetilde{\boldsymbol{\theta}}$, with an appropriately designed feature map $\boldsymbol{\Phi}(\cdot)$ and cost function $C(\cdot\,|\,\widetilde{\boldsymbol{z}})$, the proofs informally follow from the fact that $C(\cdot\,|\,\widetilde{\boldsymbol{z}})$ and $R^\star(\cdot)$ have a similar shape. This fact is illustrated in Box 6 for a toy example of compressive clustering. There it can be seen that, with a properly designed feature map $\boldsymbol{\Phi}(\cdot)$, a well-chosen cost function $C(\cdot\,|\,\widetilde{\boldsymbol{z}})$, and a sufficiently large sketch dimension m, the minimizer $\widetilde{\boldsymbol{\theta}}$ of the cost yields a nearly minimal true risk $R^\star(\widetilde{\boldsymbol{\theta}})$ and lives in a large basin-of-attraction in $C(\cdot\,|\,\widetilde{\boldsymbol{z}})$. If the sketch dimension m is chosen too small, however, then the excess risk $R^\star(\widetilde{\boldsymbol{\theta}}) - R^\star(\boldsymbol{\theta}^\star)$ increases.

A theory of compressive learning [8] has been developed to better understand how (random) feature maps $\boldsymbol{\Phi}(\cdot)$ and cost functions $C(\cdot\,|\,\widetilde{\boldsymbol{z}})$ can be designed to ensure that the sketch $\widetilde{\boldsymbol{z}}$ captures sufficient information to control the excess risk on $\widetilde{\boldsymbol{\theta}}$. **In the following four subsections, we aim to summarize the key aspects of this theory using broadly accessible language.** For those interested in the full technical details, they can be found in [8].

In a nutshell, the theory relies on interpreting $\widetilde{\boldsymbol{\theta}}$ as the minimizer of the risk attributed to a *surrogate* probability distribution $\widetilde{p}$ with the following properties:

- $\widetilde{p}$ is "close" to the distribution p_X, as measured by a task-driven distance $d(\widetilde{p}, p_X)$;
- $\widetilde{p}$ is a "simple" distribution (*e.g.*, for compressive GMM, $\widetilde{p}$ is a Gaussian mixture);
- $\widetilde{p}$ minimizes $\|\widetilde{\boldsymbol{z}} - \mathcal{A}(p)\|$ among all simple distributions p.

For example, in compressive GMM, this holds when $\widetilde{p} = p_{\widetilde{\boldsymbol{\theta}}}$, *i.e.*, the Gaussian mixture distribution parameterized by $\widetilde{\boldsymbol{\theta}}$. Given such a $\widetilde{p}$, the theory can be explained in the following step-by-step manner:

- Just as in traditional statistical learning, excess risk bounds can be established if the task-driven distance $d(\widetilde{p}, p_X)$ is controlled (see Section 5.2).
- Given a family of "simple" probability distributions (see Section 5.3), this distance can indeed be controlled provided a certain "lower restricted isometry property" (LRIP) holds (see Section 5.4). This is analogous to traditional CS, where the RIP allows one to control the performance of sparse signal recovery.
- To establish the LRIP for a random feature maps $\boldsymbol{\Phi}(\cdot)$, one can build on connections between kernel methods and the JL Lemma (see Section 5.5).
- Finally, to design task-specific feature maps $\boldsymbol{\Phi}(\cdot)$, one can appeal to the problem of kernel design (see Section 5.6).

5.2 TASK-DRIVEN DISTANCES AND EXCESS-RISK BOUNDS

In traditional statistical learning, it is common to define a distance between the true distribution p_X and an arbitrary surrogate p'_X as

$$d(p'_X, p_X) = \sup_{\boldsymbol{\theta}} \big| R^\star(\boldsymbol{\theta}|\, p'_X) - R^\star(\boldsymbol{\theta}|\, p_X) \big|, \tag{26}$$

where $R^\star(\boldsymbol{\theta}|\, p) := \mathbb{E}_{X\sim p}[L(\boldsymbol{\theta}|\, X)]$ denotes the risk under an arbitrary distribution p. This distance is task specific, since the loss function $L(\cdot|\,\cdot)$ depends on the estimation task. The utility of $d(p'_X, p_X)$ for assessing the goodness of parameter estimation follows from the inequalities

$$\begin{aligned} R^\star(\boldsymbol{\theta}^\star|\, p_X) \le R^\star(\boldsymbol{\theta}^{\star\prime}|\, p_X) &\le R^\star(\boldsymbol{\theta}^{\star\prime}|\, p'_X) + d(p'_X, p_X) \\ &\le R^\star(\boldsymbol{\theta}^\star|\, p'_X) + d(p'_X, p_X) \\ &\le R^\star(\boldsymbol{\theta}^\star|\, p_X) + 2d(p'_X, p_X), \end{aligned} \tag{27}$$

where $\boldsymbol{\theta}^\star := \arg\min_{\boldsymbol{\theta}} R^\star(\boldsymbol{\theta}|p_X)$ and $\boldsymbol{\theta}^{\star\prime} := \arg\min_{\boldsymbol{\theta}} R^\star(\boldsymbol{\theta}|p'_X)$. In particular, these inequalities yield the following bounds on the excess risk of $\boldsymbol{\theta}^{\star\prime}$:

$$0 \le R^\star(\boldsymbol{\theta}^{\star\prime}|\, p_X) - R^\star(\boldsymbol{\theta}^\star|\, p_X) \le 2d(p'_X, p_X). \tag{28}$$

For example, if p'_X equals the empirical distribution $\widehat{p}_{\mathcal{X}}(\boldsymbol{x}) := \frac{1}{n}\sum_{i=1}^n \delta(\boldsymbol{x} - \boldsymbol{x}_i)$, then (28) bounds the "generalization error" of *empirical risk minimization*, *i.e.*, the error incurred when training with $\mathcal{X} = \{\boldsymbol{x}_i\}_{i=1}^n$ but testing with independent samples drawn from p_X. This is the reasoning behind many classic statistical-learning guarantees, where so-called "uniform convergence" results establish that, under appropriate conditions, $d(\widehat{p}_{\mathcal{X}}, p_X) = O(1/\sqrt{n})$ with high probability on the draw of i.i.d. training samples $\mathcal{X} = \{\boldsymbol{x}_i\}_{i=1}^n$. The term "uniform convergence" stems from the analogy between the distance (26) and the ℓ_∞ norm.

Similarly, excess risk bounds for compressive learning are achieved by interpreting the minimizer $\widetilde{\boldsymbol{\theta}}$ of (23) *as the minimizer of the risk $R^\star(\boldsymbol{\theta}|\widetilde{p})$, for some "simple" probability distribution $\widetilde{p}$ that minimizes $\|\widetilde{\boldsymbol{z}} - \mathcal{A}(\widetilde{p})\|$, and by controlling the distance $d(\widetilde{p}, p_X)$. To control this distance, key geometric intuitions exploit the connections between compressive learning and CS, as illustrated in Fig. 9.*

5.3 EXPLOITING SIMPLIFIED MODELS TO LEARN FROM A SKETCH

Recall that, in CS, the goal is to recover the best k-sparse approximation of an unknown vector $\boldsymbol{x}_0 \in \mathbb{R}^d$ given the noisy linear measurement $\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x}_0 + \boldsymbol{e} \in \mathbb{R}^m$, where $m \ll d$. Ideally, recovery aims to find the k-sparse vector whose (noiseless) measurement is closest to the observed $\boldsymbol{y}$, *i.e.*,

$$\widehat{\boldsymbol{x}} = \arg\min_{\boldsymbol{x}\in\Sigma_k} \|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|^2. \tag{29}$$

In practice, convex or greedy approximations of this combinatorial approach are often used. Recovery guarantees can be established using the *restricted isometry property* (RIP) in (5), which says that the Euclidean distance $\|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{A}\boldsymbol{x}'\|$ between the (noiseless) measurements of two k-sparse vectors is almost the same as the Euclidean distance $\|\boldsymbol{x} - \boldsymbol{x}'\|$ between the vectors themselves (see Box 4). These guarantees require that the sparsity k is sufficiently small for a given m and d.

To connect compressive learning to CS, we first reframe the goal of learning-from-a-sketch as that of recovering a parametric model *distribution* $p_{\boldsymbol{\theta}} \in \Sigma_{\boldsymbol{\Theta}}$ from the sketch $\widetilde{\boldsymbol{z}}$, rather than recovering the model parameters $\boldsymbol{\theta} \in \boldsymbol{\Theta}$ themselves. Here, $\Sigma_{\boldsymbol{\Theta}} := \{p_{\boldsymbol{\theta}} : \boldsymbol{\theta} \in \boldsymbol{\Theta}\}$ denotes some set of admissible

distributions $p_{\boldsymbol{\theta}}$. For example, in compressive GMM, the goal becomes that of recovering a k-term GMM $p_{\boldsymbol{\theta}} = \sum_{\ell=1}^{k} \alpha_\ell \mathcal{N}(\boldsymbol{\mu}_\ell, \boldsymbol{\Sigma}_\ell)$ rather than the GMM parameters $\boldsymbol{\theta} = \{\alpha_\ell, \boldsymbol{\mu}_\ell, \boldsymbol{\Sigma}_\ell\}_{\ell=1}^{k}$. Likewise, in compressive PCA, the goal becomes that of recovering a distribution $p_{\boldsymbol{\theta}}$ whose autocorrelation matrix has rank of at most k, rather than a k-dimensional orthonormal basis $\boldsymbol{\theta}$ for the column space of that autocorrelation matrix. Then, mirroring the CS recovery approach (29), compressive learning recovery aims to find a parametric distribution $p_{\boldsymbol{\theta}} \in \Sigma_{\boldsymbol{\Theta}}$ whose (noiseless) sketch is closest to the observed sketch $\widetilde{\boldsymbol{z}}$, *i.e.*,

$$\widetilde{p} = p_{\widetilde{\boldsymbol{\theta}}} := \underset{p_{\boldsymbol{\theta}} \in \Sigma_{\boldsymbol{\Theta}}}{\arg\min} \|\widetilde{\boldsymbol{z}} - \mathcal{A}(p_{\boldsymbol{\theta}})\|^2. \tag{30}$$

The construction of $\Sigma_{\boldsymbol{\Theta}}$ implies that the parameters $\widetilde{\boldsymbol{\theta}}$ defining $p_{\widetilde{\boldsymbol{\theta}}}$ minimize a certain cost function, *i.e.*,

$$\widetilde{\boldsymbol{\theta}} = \underset{\boldsymbol{\theta} \in \boldsymbol{\Theta}}{\arg\min}\, C(\boldsymbol{\theta} | \, \widetilde{\boldsymbol{z}}) \qquad \text{with } C(\boldsymbol{\theta} | \, \widetilde{\boldsymbol{z}}) = \|\widetilde{\boldsymbol{z}} - \mathcal{A}(p_{\boldsymbol{\theta}})\|^2. \tag{31}$$

Note the similarity between (31) and (17)-(18). Moreover, $\widetilde{\boldsymbol{\theta}}$ also minimizes the risk $R^\star(\boldsymbol{\theta} | \, \widetilde{p})$ [8].

From the description above, we see that a central theme of both CS and compressive learning is that *measurement compression makes it impossible to recover the full object-of-interest* (*i.e.*, $\boldsymbol{x}_0$ in CS or p_X in compressive learning) without additional side-information. For this reason, both seek to recover the parameters of *a simplified model* of the object-of-interest. In CS, this is accomplished by seeking to recover the best k-sparse approximation to $\boldsymbol{x}_0$, rather than $\boldsymbol{x}_0$ itself. In compressive learning, this is accomplished by seeking to recover the best model distribution $p_{\boldsymbol{\theta}} \in \Sigma_{\boldsymbol{\Theta}}$, rather than the true distribution p_X. In both cases, if the linear operator (*i.e.*, $\boldsymbol{A}$ in CS or $\mathcal{A}(\cdot)$ in compressive learning) is well designed, then the model parameters (in Σ_k for CS or in $\Sigma_{\boldsymbol{\Theta}}$ for compressive learning) can be accurately recovered.

5.4 THE LRIP AND EXCESS-RISK CONTROL

Recovery guarantees can be established using a tool analogous to the RIP (5) in CS.

Take, for example, the case of compressive PCA. As discussed around (22), one could use a sketch of the form $\widetilde{\boldsymbol{z}} = \boldsymbol{A}\,\mathrm{vec}(\widehat{\boldsymbol{R}}) \in \mathbb{R}^m$, where $\widehat{\boldsymbol{R}} = \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{x}_i \boldsymbol{x}_i^\top$ is the $d \times d$ empirical autocorrelation matrix and $m \ll d^2$, and then search for $\widetilde{\boldsymbol{R}}$, the symmetric matrix with rank at-most-k that minimizes the cost $C(\boldsymbol{R} | \, \widetilde{\boldsymbol{z}}) = \|\widetilde{\boldsymbol{z}} - \boldsymbol{A}\,\mathrm{vec}(\boldsymbol{R})\|^2$. Recovery guarantees can be established using an RIP of the form

$$(1-\delta)\|\boldsymbol{R} - \boldsymbol{R}'\|_F^2 \le \|\boldsymbol{A}\,\mathrm{vec}(\boldsymbol{R}) - \boldsymbol{A}\,\mathrm{vec}(\boldsymbol{R}')\|^2 \le (1+\delta)\|\boldsymbol{R} - \boldsymbol{R}'\|_F^2, \quad \forall \boldsymbol{R}, \boldsymbol{R}' \in \Sigma_k, \tag{32}$$

where $\delta \in (0,1)$ is a tolerance and Σ_k is now the set of $d \times d$ symmetric rank-k matrices. It is known [6, Chapter 9] that i.i.d. Gaussian $\boldsymbol{A}$ satisfy the RIP (32) with high probability when m is on the order of kd times a constant that depends on the tolerance δ. Furthermore, the cost-minimizing $\widetilde{\boldsymbol{R}}$ is near-optimal in the sense of minimizing the excess risk, even when a convex relaxation of the cost is used [8].

These compressive-PCA guarantees hold even under *model mismatch*: although Σ_k constrains $\widetilde{\boldsymbol{R}}$ to have rank at-most-k, the empirical autocorrelation matrix $\widehat{\boldsymbol{R}}$ used to construct $\widetilde{\boldsymbol{z}}$ tends to have full rank d in practice. To explore this idea in more detail, suppose for the moment that the data $\mathcal{X}$ lies in a rank-k subspace. In this case, $\widehat{\boldsymbol{R}}$ would have rank at-most-k, and there exists a symmetric $\boldsymbol{R}$ with rank at-most-k that drives the cost $C(\boldsymbol{R} | \, \widetilde{\boldsymbol{z}}) = \|\boldsymbol{A}\,\mathrm{vec}(\widehat{\boldsymbol{R}}) - \boldsymbol{A}\,\mathrm{vec}(\boldsymbol{R})\|^2$ to zero. Furthermore, when $\boldsymbol{A}$ satisfies the RIP (32), the left inequality in (32) implies that this cost-minimizing $\widetilde{\boldsymbol{R}}$ must equal $\widehat{\boldsymbol{R}}$. When the data $\mathcal{X}$ does not lie in a rank-k subspace, the minimal cost will be non-zero. In this case, an upper bound on the error $\|\widehat{\boldsymbol{R}} - \widetilde{\boldsymbol{R}}\|_F$ of the cost-minimizing estimate $\widetilde{\boldsymbol{R}}$, as well as an upper bound on the excess risk of the principal subspace $\widetilde{\boldsymbol{\theta}}$ of $\widetilde{\boldsymbol{R}}$, can both be obtained as a consequence of the RIP (32).

In place of the RIP, compressive learning uses the so-called "*lower* RIP" (LRIP),

$$d(p_{\boldsymbol{\theta}}, p_{\boldsymbol{\theta}'}) \leq C_0 \|\mathcal{A}(p_{\boldsymbol{\theta}}) - \mathcal{A}(p_{\boldsymbol{\theta}'})\|, \tag{33}$$

assumed to be valid for each pair $p_{\boldsymbol{\theta}}, p_{\boldsymbol{\theta}'} \in \Sigma_{\boldsymbol{\Theta}}$, where C_0 is a positive constant. The LRIP says that the Euclidean distance $\|\mathcal{A}(p_{\boldsymbol{\theta}}) - \mathcal{A}(p_{\boldsymbol{\theta}'})\|$ between the (noiseless) sketches of two distributions is—up to a scaling—controlling the distance $d(p_{\boldsymbol{\theta}}, p_{\boldsymbol{\theta}'})$ between the distributions themselves (26). Note that the lower bound $d(\cdot, \cdot)$ in (33) is task-specific, and not Euclidean as in the CS case (5).

The first main theoretical guarantee about compressive learning [8] is that, when the LRIP (33) holds, the estimate $\widetilde{\boldsymbol{\theta}}$ obtained by minimizing (31) automatically has controlled excess risk. In particular, using $\widetilde{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} R^\star(\boldsymbol{\theta}|\, p_{\widetilde{\boldsymbol{\theta}}})$ [8], the excess risk bound (28) can be combined with the LRIP (33) and the definition of $\widetilde{\boldsymbol{\theta}}$ (31) as follows:

$$\begin{aligned}
R^\star(\widetilde{\boldsymbol{\theta}}|\, p_X) - R^\star(\boldsymbol{\theta}'|\, p_X) &\leq 2 d(p_{\widetilde{\boldsymbol{\theta}}}, p_X) \\
&\leq 2\Big[d(p_{\widetilde{\boldsymbol{\theta}}}, p_{\boldsymbol{\theta}'}) + d(p_{\boldsymbol{\theta}'}, p_X)\Big] \\
&\leq 2\Big[C_0 \|\mathcal{A}(p_{\widetilde{\boldsymbol{\theta}}}) - \mathcal{A}(p_{\boldsymbol{\theta}'})\| + d(p_{\boldsymbol{\theta}'}, p_X)\Big] \\
&\leq 2\Big[C_0 \|\mathcal{A}(p_{\widetilde{\boldsymbol{\theta}}}) - \widetilde{\boldsymbol{z}}\| + C_0 \|\widetilde{\boldsymbol{z}} - \mathcal{A}(p_{\boldsymbol{\theta}'})\| + d(p_{\boldsymbol{\theta}'}, p_X)\Big] \\
&\leq 2\Big[2C_0 \|\mathcal{A}(p_{\boldsymbol{\theta}'}) - \widetilde{\boldsymbol{z}}\| + d(p_{\boldsymbol{\theta}'}, p_X)\Big] \\
&\leq 2\Big[2C_0 \|\mathcal{A}(p_X) - \widetilde{\boldsymbol{z}}\| + 2C_0 \|\mathcal{A}(p_{\boldsymbol{\theta}'}) - \mathcal{A}(p_X)\| + d(p_{\boldsymbol{\theta}'}, p_X)\Big],
\end{aligned} \tag{34}$$

for any distribution $p_{\boldsymbol{\theta}'}$ in $\Sigma_{\boldsymbol{\Theta}}$ (because $d(\cdot, \cdot)$ is a valid distance metric, *i.e.*, $d(p_1, p_2) \leq d(p_1, p_3) + d(p_3, p_2)\ \forall p_3$). One option is to choose $p_{\boldsymbol{\theta}'} = p_{\boldsymbol{\theta}^\star}$, in which case the term $2C_0 \|\mathcal{A}(p_{\boldsymbol{\theta}^\star}) - \mathcal{A}(p_X)\| + d(p_{\boldsymbol{\theta}^\star}, p_X)$ in the upper bound reflects the excess risk due to modeling and $2C_0 \|\mathcal{A}(p_X) - \widetilde{\boldsymbol{z}}\|$ reflects the excess risk due to sketching from a finite dataset $\mathcal{X}$. For a tighter bound, we could choose $p_{\boldsymbol{\theta}'}$ as the distribution in $\Sigma_{\boldsymbol{\Theta}}$ that minimizes the right side of (34). These approaches, and more refined variants, have been applied to analyze various learning tasks, in *e.g.*, [8, 18, 19].

It should be emphasized that recovery guarantees based on the RIP (5) or LRIP (33) hold *even in the presence of measurement noise* $\boldsymbol{e}$ *and/or modeling errors*. In CS, measurement noise arises due to, *e.g.*, thermal noise or interference, while modeling errors arise when $\boldsymbol{x}_0$ is not truly k-sparse. Many sparse recovery techniques are provably robust to such noise and modeling errors, *cf.* (6) in Box 4. In compressive learning, measurement noise arises due to the finite cardinality of the dataset $\mathcal{X}$ (recall (9)), while modeling errors arise when $p_X \notin \Sigma_{\boldsymbol{\Theta}}$. For example, GMM-recovery guarantees can be established even when p_X is not truly a GMM.

5.5 Establishing the LRIP via kernels and the JL lemma

In light of the fact that the LRIP (33) yields statistical learning guarantees (34) for compressive learning, a key question becomes: How can we choose the sketch dimension m so that the LRIP (33) holds? Similar to how the RIP is proven in CS (see, *e.g.*, [6, Lemma 9.33]), refinements of a mathematical tool called the Johnson-Lindenstrauss (JL) lemma [31] can be used to obtain a value of m sufficient for the LRIP to hold *with high probability on the draw of the random feature map* $\boldsymbol{\Phi}(\cdot)$. We now summarize this approach.

To begin, we establish connections to kernel methods (see Box 7 for a brief review of kernels). Some key observations are that any feature map explicitly defines a positive definite *kernel*, and that the *expectation*

of this kernel defines another useful positive definite kernel. To see why, consider, for example, the RF feature map (3). First, for a fixed set of frequency vectors $\{\boldsymbol{w}_j\}_{j=1}^m$, we have

$$\langle \tfrac{1}{\sqrt{m}}\boldsymbol{\Phi}(\boldsymbol{x}), \tfrac{1}{\sqrt{m}}\boldsymbol{\Phi}(\boldsymbol{x}')\rangle = \frac{1}{m}\sum_{j=1}^{m} \exp(-\mathrm{j}2\pi \boldsymbol{w}_j^\top(\boldsymbol{x}-\boldsymbol{x}')), \qquad \forall \boldsymbol{x}, \boldsymbol{x}' \tag{35}$$

where $\langle \cdot, \cdot \rangle$ is the standard Euclidean inner product in $\mathbb{R}^m$ or $\mathbb{C}^m$. The left-hand side defines a positive definite kernel according to the terminology reviewed in Box 7. This kernel is also "shift-invariant," in that it depends only on the difference $\boldsymbol{x}-\boldsymbol{x}'$. Next, imagine that the d-dimensional frequency vectors $\boldsymbol{w}_j$, which constitute the m rows of the random matrix $\boldsymbol{W}$, are drawn i.i.d. from some probability distribution p_W. In the limit of large sketch dimension m, the law of large numbers combined with (35) says

$$\langle \tfrac{1}{\sqrt{m}}\boldsymbol{\Phi}(\boldsymbol{x}), \tfrac{1}{\sqrt{m}}\boldsymbol{\Phi}(\boldsymbol{x}')\rangle \overset{\text{w.p.1}}{\to} \mathbb{E}_W \exp(-\mathrm{j}2\pi W^\top(\boldsymbol{x}-\boldsymbol{x}')). \tag{36}$$

The right hand side of (36) is the Fourier transform (FT) of the probability density p_W evaluated at $\boldsymbol{x}-\boldsymbol{x}'$, *i.e.*, the characteristic function $\Psi_{p_W}(\boldsymbol{x}'-\boldsymbol{x})$ using the notation from (14). Because $\Psi_{p_W}(\cdot)$ is the FT of a non-negative function, it is a so-called *positive definite function*, which means that, for any $\{\boldsymbol{x}_i\}_{i=1}^n$, the $n \times n$ matrix $\boldsymbol{\Psi}$ defined with elements $\Psi_{ij} = \Psi_{p_W}(\boldsymbol{x}_i - \boldsymbol{x}_j)$ for $1 \le i, j \le n$ will be positive semi-definite. Thus, if we construct a kernel as $\kappa(\boldsymbol{x}, \boldsymbol{x}') := \Psi_{p_W}(\boldsymbol{x}'-\boldsymbol{x})$, then it will be a positive definite kernel. When $p_W = \mathcal{N}(\boldsymbol{0}, \sigma_w^2 \boldsymbol{I}_d)$, this approach yields the familiar Gaussian kernel (a particular type of "radial basis function"), *i.e.*, $\kappa_\sigma(\boldsymbol{x}, \boldsymbol{x}') := \exp(-\|\boldsymbol{x}-\boldsymbol{x}'\|^2/2\sigma^2)$, here of width $\sigma = 1/\sigma_w$.

More generally, by considering any parametric feature map of the form $\boldsymbol{\Phi}(\boldsymbol{x}|\,\boldsymbol{W})$, where the parameter $\boldsymbol{W}$ is drawn at random according to some probability distribution, one can define[3] the "expected kernel"

$$\kappa(\boldsymbol{x}, \boldsymbol{x}') := \mathbb{E}_{\boldsymbol{W}} \left\langle \tfrac{1}{\sqrt{m}}\boldsymbol{\Phi}(\boldsymbol{x}|\,\boldsymbol{W}), \tfrac{1}{\sqrt{m}}\boldsymbol{\Phi}(\boldsymbol{x}'|\,\boldsymbol{W}) \right\rangle, \tag{37}$$

not to be confused with the "mean kernel" defined in (38). This setting includes RF features (3) with i.i.d. frequencies $\boldsymbol{w}_j$ (as above), or with a frequency matrix $\boldsymbol{W}$ that includes structured blocks of rows, as we will soon discuss.

[Box 7] Kernel Methods and Kernel Embeddings of Probability Distributions

Sketching shares connections with *kernel methods* [2], a family of machine learning techniques that produce decisions or insights using a kernel function, $\kappa(\boldsymbol{x}, \boldsymbol{x}') \in \mathbb{R}$, which measures the "similarity" between $\boldsymbol{x}$ and $\boldsymbol{x}'$. A kernel is said to be "positive definite" if the $n \times n$ matrix $\boldsymbol{K}$, constructed with entries $\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j)$ for $1 \le i, j \le n$, is positive *semi*-definite for every possible $\{\boldsymbol{x}_i\}_{i=1}^n$. The celebrated "kernel trick" states that any positive definite kernel *implicitly* amounts to an inner product in some higher-dimensional (and potentially infinite-dimensional) feature space $\mathcal{H}$, and vice-versa. That is, $\kappa(\boldsymbol{x}, \boldsymbol{x}') = \langle \overline{\boldsymbol{\Phi}}(\boldsymbol{x}), \overline{\boldsymbol{\Phi}}(\boldsymbol{x}')\rangle$ for some (not necessarily explicitly known) mapping $\overline{\boldsymbol{\Phi}}(\cdot)$ from the signal space to $\mathcal{H}$. Any machine learning method that relies only on the evaluation of inner products—such as ridge regression, support-vector-machine classification, PCA [46], and dictionary learning [48]—can be "kernelized" by using a kernel in place of the inner product. Kernelizing a method is thus tantamount to applying that method in a transformed, higher-dimensional feature space. In this way, more complex estimation and/or decision functions can be implemented.

Given a positive definite kernel $\kappa(\boldsymbol{x}, \boldsymbol{x}')$ operating on signals $\boldsymbol{x}$ and $\boldsymbol{x}'$ in some set, it is possible to

[3] In many papers, the $1/\sqrt{m}$ scaling in (35)-(37) is subsumed in the feature map $\boldsymbol{\Phi}(\cdot)$.

"lift" $\kappa(\cdot,\cdot)$ to a positive definite kernel *operating on probability distributions* over this set by defining the so-called *mean kernel*

$$k(p,q) := \mathbb{E}_{X\sim p, X'\sim q}[\kappa(X,X')]. \tag{38}$$

This defines an embedding of probability distributions into a kernel space, which is analogous to the finite-dimensional embedding $\mathcal{A}(p)$ of probability distribution p from (8). The *maximum mean discrepancy* (MMD) [49, 50]

$$\mathrm{MMD}(p,q) := \sqrt{k(p,p)+k(q,q)-2k(p,q)} \tag{39}$$

is the Euclidean metric naturally induced by the mean kernel. It is analogous to the Euclidean distance between sketches, $\|\mathcal{A}(p)-\mathcal{A}(q)\|$. The MMD, originally introduced in the context of two-sample hypothesis testing [49], is now well-known in machine learning. When the MMD behaves as a true metric, *i.e.*, when $\mathrm{MMD}(p,q)=0 \Leftrightarrow p=q$, the mean kernel $k(\cdot,\cdot)$ is said to be "characteristic". In $\mathbb{R}^d$, many classic kernels $\kappa(\cdot,\cdot)$, such as the Gaussian and Laplace kernels, yield characteristic mean kernels [50].

Now that the kernel connections have been established, we return to our original objective of *understanding when the LRIP* (33) *holds*. Importantly, the law of large numbers shows that, in the limit of large sketch dimension m, the right side of (33) is related to the maximum mean discrepancy (MMD) from (39), which is a kernel-based distance between distributions (see Box 7). Indeed, for arbitrary probability distributions p and q, we have

$$\lim_{m\to\infty} \frac{1}{\sqrt{m}}\|\mathcal{A}(p)-\mathcal{A}(q)\| \overset{\text{w.p.1}}{=} \mathrm{MMD}(p,q). \tag{40}$$

Thus, when the LRIP (33) holds, it must also be true that, for sufficiently large sketch dimension m,

$$d(p_{\boldsymbol{\theta}}, p_{\boldsymbol{\theta}'}) \le C_0' \,\mathrm{MMD}(p_{\boldsymbol{\theta}}, p_{\boldsymbol{\theta}'}), \qquad \text{for all } p_{\boldsymbol{\theta}}, p_{\boldsymbol{\theta}'} \in \Sigma_{\boldsymbol{\Theta}}, \tag{41}$$

where C_0' is a positive constant. Property (41) connects two different metrics on probability distributions: the left hand side of (41) is defined by the learning task (recall (26)), while the right hand side of (41) is defined by the expected kernel κ, from (37), associated with the randomized feature map. Note that (41) is a deterministic property; it does not depend on the draw of the randomized feature map, unlike the LRIP (33). In the literature, (41) is called the "kernel-LRIP" [8] because it is a kernel-based analog to the LRIP. Being deterministic, the expected kernel is often easier to manipulate than the random feature map, and thus eases the proof of the kernel-LRIP. This is important because, if the kernel-LRIP (41) holds, then, using arguments based on the JL lemma, one can also establish [8] the LRIP (33), as we show below.

The JL lemma [31] is a precursor of the restricted isometry property that is specialized to finite sets (5). It states that, given N arbitrary d-dimensional vectors $\boldsymbol{x}_i$, there exists an $m\times d$ matrix $\boldsymbol{A}$, with m on the order of $\log N$, such that $\|\boldsymbol{A}\boldsymbol{x}_i-\boldsymbol{A}\boldsymbol{x}_j\| \approx \|\boldsymbol{x}_i-\boldsymbol{x}_j\|$ for all i,j. Indeed, since there are only N^2 pairs of parametric mixtures $p_{\boldsymbol{\theta}}, p_{\boldsymbol{\theta}'} \in \Sigma_{\boldsymbol{\Theta}}$, the lower bound (42) is valid *uniformly* for all of them, except with probability at most $N^2\exp(-c_0 m)$. This failure probability can be made smaller than any $\epsilon>0$ by choosing m larger than $2c_0^{-1}\log(N/\sqrt{\epsilon})$.

To illustrate how the JL lemma is useful in the context of compressive learning, let us momentarily restrict our attention to a learning problem where the collection $\Sigma_{\boldsymbol{\Theta}}$ of parametric distributions is of *finite cardinality*, noting that we can extend this approach to continuous families of parametric distributions

through discretization arguments involving the notion of covering numbers [6, Appendix C], as described below. As a concrete example, let us consider a discretized variant of compressive GMM using the RF feature map (3). As in [8, 11], we assume that the d-dimensional frequency vectors $\boldsymbol{w}_j$ are drawn i.i.d. from the normal distribution $\mathcal{N}(\mathbf{0}, \sigma_w^2 \boldsymbol{I}_d)$, and we consider learning a mixture of Gaussian components $p_{\boldsymbol{\theta}_\ell} = \mathcal{N}(\boldsymbol{\mu}_\ell, \boldsymbol{I})$ for $\ell = 1, \ldots, k$, with equal weights $\alpha_\ell = 1/k$ for each ℓ, where the means $\boldsymbol{\mu}_\ell$ are assumed to be bounded, separated, and discretized on a regular grid. In this setting, there exists a finite number, N, of possible parametric mixture distributions $p_{\boldsymbol{\theta}} \in \Sigma_{\boldsymbol{\Theta}}$. As long as the MMD is a true metric (as defined in Box 7), the ratio $d(p_{\boldsymbol{\theta}}, p_{\boldsymbol{\theta}'}) / \operatorname{MMD}(p_{\boldsymbol{\theta}}, p_{\boldsymbol{\theta}'})$ is finite for $p_{\boldsymbol{\theta}} \neq p_{\boldsymbol{\theta}'}$, hence whenever $\Sigma_{\boldsymbol{\Theta}}$ is a finite collection, as in this concrete example, the kernel-LRIP (41) holds (for a sufficiently large C_0'). Moreover, specializing to an arbitrary pair of mixtures $p_{\boldsymbol{\theta}}, p_{\boldsymbol{\theta}'} \in \Sigma_{\boldsymbol{\Theta}}$ and a given sketch dimension m, refinements of (40) (using *measure concentration*) enable one to show that the lower bound

$$\frac{1}{\sqrt{2}} \operatorname{MMD}(p_{\boldsymbol{\theta}}, p_{\boldsymbol{\theta}'}) \leq \frac{1}{\sqrt{m}} \|\mathcal{A}(p_{\boldsymbol{\theta}}) - \mathcal{A}(p_{\boldsymbol{\theta}'})\| \tag{42}$$

holds with probability at least $1 - \exp(-c_0 m)$, where c_0 is a positive *concentration constant*. Combining (41) and (42) using a union bound, it can be shown [8] that the LRIP (33) holds with high probability when C_0 is on the order of $C_0' \sqrt{m}$ and the sketch dimension m grows logarithmically with N, the number of parametric distributions in the finite set $\Sigma_{\boldsymbol{\Theta}}$.

For infinite collections $\Sigma_{\boldsymbol{\Theta}}$, proving the kernel-LRIP (41), and eventually the LRIP (33), is more technical and can require some additional assumptions. As an example, for compressive clustering with RF features, it can be proven that there is a constant C_0' such that (41) holds, provided that the centroids are sufficiently separated and bounded [8]. The JL lemma can be extended by refining techniques used to establish the RIP (5). The main idea is to first prove the LRIP on some finite collection $\Sigma' \subset \Sigma_{\boldsymbol{\Theta}}$ of N probability distributions, and then to extrapolate it (with slightly worse constants) to $\Sigma_{\boldsymbol{\Theta}}$. Technically, this involves the notion of covering numbers, and the cardinality $N = |\Sigma'|$ is typically exponential in the number of parameters needed to describe $\Sigma_{\boldsymbol{\Theta}}$. For example, for GMM, one needs kd parameters to describe the means $\boldsymbol{\mu}_\ell \in \mathbb{R}^d$, $1 \leq \ell \leq k$ of the mixture $p_{\boldsymbol{\theta}} = \sum_{\ell=1}^k \frac{1}{k} \mathcal{N}(\boldsymbol{\mu}_\ell, \boldsymbol{I})$, and $\log N$ essentially depends linearly on kd. The dimension m of the sketch for which the LRIP holds with high probability is thus on the order of kd/c_0, up to some additional factors due to the proof technique [8].

Empirical studies of compressive clustering [16, 17] and compressive GMM [11] suggest that a sketch dimension m on the order of kd (the number of parameters in these settings) is sufficient to yield accurate learning performance. The best known bounds on provably good sketch dimensions [8] remain pessimistic compared to these empirically validated sketch dimensions. This is most likely related to sub-optimal bounds for the concentration constant c_0 and/or shortcomings in the techniques used to extend the LRIP from a finite collection Σ' to an infinite collection $\Sigma_{\boldsymbol{\Theta}}$.

5.6 THE CHALLENGE OF DESIGNING A FEATURE MAP GIVEN A LEARNING TASK

In the existing literature, the LRIP has been established [8] for randomized feature maps $\boldsymbol{\Phi}(\cdot)$ (*e.g.*, random Fourier features, random quadratic features) that mimic related constructions from compressive sensing, developed either for sparse-vector recovery or low-rank matrix recovery.

When sketching with random Fourier features (*e.g.*, for compressive clustering and compressive GMM), the main design choice for $\boldsymbol{\Phi}(\cdot)$ is the distribution from which to draw the random frequencies $\boldsymbol{w}_j$ (*i.e.*, the rows of $\boldsymbol{W}$ in (3)). In light of the connections to shift-invariant kernels (recall (35)), this design task is a particular instance of the difficult problem of kernel design [2, Sec. 4.4.5].

For example, when the rows of $\boldsymbol{W}$ are drawn i.i.d. $\mathcal{N}(\boldsymbol{0}, \sigma_w^2 \boldsymbol{I})$, the choice of the variance σ_w^2 determines the choice of the width $\sigma = 1/\sigma_w$ of the corresponding Gaussian kernel $\kappa_\sigma(\cdot, \cdot)$. Indeed, from a signal-processing standpoint, the corresponding mean kernel $k_\sigma(\cdot, \cdot)$ (recall (38)) acts to low-pass filter the underlying data distributions. To see why, observe that $\kappa_\sigma(\boldsymbol{x}, \boldsymbol{x}') = g_\sigma(\boldsymbol{x} - \boldsymbol{x}')$ with $g_\sigma(\boldsymbol{x}) := e^{-\|\boldsymbol{x}\|^2/2\sigma^2}$, and so

$$k_\sigma(p, q) = \mathbb{E}_{X \sim p, X' \sim q}[\kappa_\sigma(X, X')] = \iint e^{-\|X - X'\|^2/2\sigma^2} \, \mathrm{d}p(X) \, \mathrm{d}q(X') \tag{43}$$

$$= \langle g_\sigma \star p, q \rangle_{L^2} = \langle g_{\overline{\sigma}} \star p, g_{\overline{\sigma}} \star q \rangle_{L^2}, \tag{44}$$

where $\star$ denotes convolution and $\overline{\sigma} = \sigma/\sqrt{2}$. Hence, the associated MMD (39) satisfies $\mathrm{MMD}(p, q) = \|g_{\overline{\sigma}} \star p - g_{\overline{\sigma}} \star q\|_{L^2}$. Recall that learning-from-a-sketch is often performed by minimizing a "sketch matching" cost $\|\widetilde{\boldsymbol{z}} - \mathcal{A}(p_{\boldsymbol{\theta}})\|^2 = \|\mathcal{A}(\widehat{p}_{\mathcal{X}}) - \mathcal{A}(p_{\boldsymbol{\theta}})\|^2$, as in (30), where $\widehat{p}_{\mathcal{X}}$ denotes the empirical distribution of the data $\mathcal{X}$. In the limit of large sketch dimension m, this cost compares the smoothed versions of the probability distributions $\widehat{p}_{\mathcal{X}}$ and $p_{\boldsymbol{\theta}}$, since $\frac{1}{m}\|\widetilde{\boldsymbol{z}} - \mathcal{A}(p_{\boldsymbol{\theta}})\|^2 \approx \left\|g_{\overline{\sigma}} \star \widehat{p}_{\mathcal{X}} - g_{\overline{\sigma}} \star p_{\boldsymbol{\theta}}\right\|_{L_2}^2$. Similarly, when using a greedy algorithm to learn a mixture model (or cluster centroids) from a sketch, the normalized inner product $\frac{1}{m}\langle \widetilde{\boldsymbol{z}}, \mathcal{A}(p_{\boldsymbol{\theta}_\ell}) \rangle$ approximates the correlation $\langle g_{\overline{\sigma}} \star \widehat{p}_{\mathcal{X}}, g_{\overline{\sigma}} \star p_{\boldsymbol{\theta}_\ell} \rangle_{L_2}$ between the low-passed versions of the empirical data distribution $\widehat{p}_{\mathcal{X}}$ and the candidate mixture component $p_{\boldsymbol{\theta}_\ell}$, respectively.

This latter idea is illustrated for compressive clustering in Figure 11. There, since the mixture component associated to a candidate centroid $\boldsymbol{c}$ is the Dirac $p_{\boldsymbol{c}}(\boldsymbol{x}) = \delta(\boldsymbol{x} - \boldsymbol{c})$ (recall the discussion before (19)), we have that $\langle \widetilde{\boldsymbol{z}}_\sigma, \mathcal{A}_\sigma(p_{\boldsymbol{c}}) \rangle = \langle \widetilde{\boldsymbol{z}}_\sigma, \boldsymbol{\Phi}_\sigma(\boldsymbol{c}) \rangle$, where the dependence on the kernel width $\sigma = 1/\sigma_w$ has been made explicit. Meanwhile,

$$\langle g_{\overline{\sigma}} \star \widehat{p}_{\mathcal{X}}, g_{\overline{\sigma}} \star p_{\boldsymbol{c}} \rangle_{L_2} = (g_\sigma \star \widehat{p}_{\mathcal{X}})(\boldsymbol{c}) = \frac{1}{n} \sum_{i=1}^{n} g_\sigma(\boldsymbol{c} - \boldsymbol{x}_i). \tag{45}$$

In (45), we recognize a Parzen window density estimator [51], whose computation for a given $\boldsymbol{c}$ requires access to the n training samples $\boldsymbol{x}_i$. In contrast, its surrogate $\langle \widetilde{\boldsymbol{z}}_\sigma, \boldsymbol{\Phi}_\sigma(\boldsymbol{c}) \rangle$ only requires access to the m-dimensional sketch $\widetilde{\boldsymbol{z}}$. In a large-scale setting, this can save huge amounts of memory and computation. As can be seen when comparing the top and bottom rows in Figure 11, $\langle \widetilde{\boldsymbol{z}}_\sigma, \boldsymbol{\Phi}_\sigma(\boldsymbol{c}) \rangle$ well approximates $(g_\sigma \star \widehat{p}_{\mathcal{X}})(\boldsymbol{c})$ for sufficiently large kernel width σ. By comparing the different columns of Figure 11, it can also be seen that the kernel width σ should be chosen compatible with the cluster width and separation. This choice involves a tradeoff between smoothing the unwanted gaps between data samples and over-smoothing the (desired) gaps between clusters. Existing theory [8] identifies sufficient conditions on the choice of σ related to the number k of candidate clusters and their minimum separation.

Although i.i.d. Gaussian frequencies $\boldsymbol{w}_j$ were used with the RF map above, one may also consider the use of i.i.d. non-Gaussian frequencies. Such designs, as proposed in [11], can yield improved empirical behavior. As we will see in the next section, it is also possible to deviate from the RF map with the goal of improving computational efficiency. Such constructions also yield non-Gaussian expected kernels (37). Although they work well in practice, there is currently no proof that these latter kernels satisfy the kernel-LRIP.

While existing theory focuses on proving the LRIP for a given random feature map and learning task, an important open question is: How should one *design the feature map* to best match a given learning task? In particular, can we design a random feature map that satisfies the LRIP (33) for a given learning task defined by a loss function $L(\boldsymbol{\theta} | \, \boldsymbol{x}_i)$ and embodied by a task-driven distance (26)? A promising yet still challenging avenue would be to first identify a positive definite kernel $\kappa_0(\boldsymbol{x}, \boldsymbol{x}')$ for which the corresponding MMD

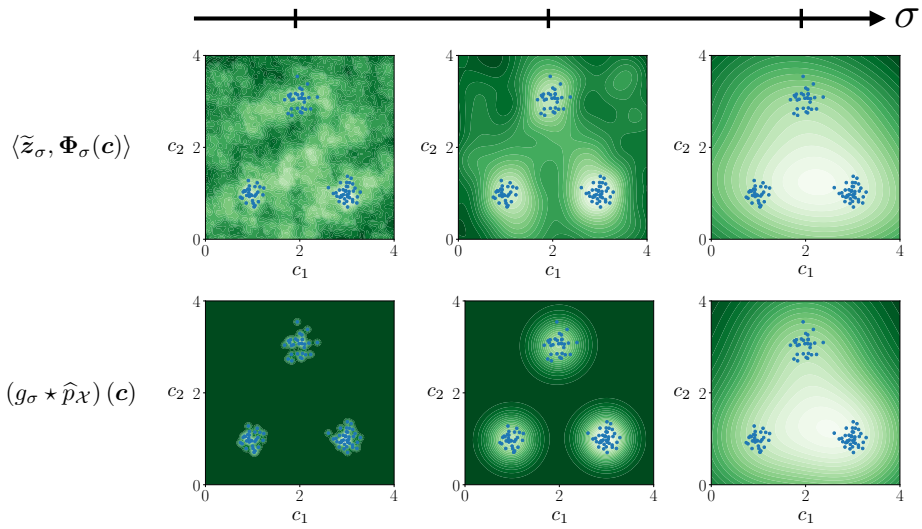


Fig. 11: Criterion $\langle \widetilde{\boldsymbol{z}}_\sigma, \boldsymbol{\Phi}_\sigma(\boldsymbol{c}) \rangle$ used by greedy parameter estimation algorithms in compressive clustering with random Fourier features (top row) versus its expected value $(g_\sigma \star \widehat{p}_{\mathcal{X}})(\boldsymbol{c})$ (bottom row) as a function of the centroid hypothesis location $\boldsymbol{c} = [c_1, c_2]^\top$. The dataset (in blue) consists of $n = 100$ points drawn according to a mixture of $k = 3$ isotropic Gaussians. The frequencies $\boldsymbol{w}_j$ used to define the feature map $\boldsymbol{\Phi}_\sigma(\cdot) = \exp(-\mathrm{j}2\pi \boldsymbol{W} \cdot)$ are drawn according to a standard Gaussian $\mathcal{N}(\boldsymbol{0}, \sigma_w^2 \boldsymbol{I})$ with $\sigma_w = 1/\sigma$. The sketch $\widetilde{\boldsymbol{z}}_\sigma$ is computed with the feature map $\boldsymbol{\Phi}_\sigma(\cdot)$. With $\sigma^2 = \frac{1}{500}$ (left), there is insufficient smoothing, and the criterion displays many spurious local maxima; with $\sigma^2 = \frac{1}{10}$ (middle), there is appropriate smoothing, and local maxima are in good correspondence with the true cluster centers; with $\sigma^2 = 1$ (right), there is over-smoothing, and the criterion displays only a single maximum.

satisfies the kernel-LRIP (41), and then use Bochner's theorem or Mercer's theorem (see Box 8) to design a random feature map $\boldsymbol{\Phi}(\cdot)$ whose expected kernel (37) is precisely κ_0.

[Box 8] Approximating kernel methods with random feature maps

While each random feature map $\boldsymbol{\Phi}(\cdot)$ implicitly defines a positive definite kernel by taking the expectation (37), the converse is also true.

For shift-invariant kernels, *i.e.*, kernels for which $\kappa(\boldsymbol{x}, \boldsymbol{x}') = \kappa(\boldsymbol{x} - \boldsymbol{x}', 0)$ only depends on the difference $\boldsymbol{x} - \boldsymbol{x}'$ (such as the Gaussian kernel), this is a consequence of Bochner's theorem [12], which states that if κ is a positive definite kernel such that $\kappa(\boldsymbol{x}, \boldsymbol{x}) = 1$ for all $\boldsymbol{x}$, then its Fourier transform yields a probability distribution $p_W(\boldsymbol{w}) = \int \kappa(\boldsymbol{x}, 0) \exp(-\mathrm{j}2\pi \boldsymbol{w}^\top \boldsymbol{x}) \, \mathrm{d}\boldsymbol{x}$. Conversely, the kernel can be obtained by the inverse Fourier transform, which can also be phrased as an expectation:

$$\kappa(\boldsymbol{x}, \boldsymbol{x}') = \int \exp\left(\mathrm{j}2\pi \boldsymbol{w}^\top (\boldsymbol{x} - \boldsymbol{x}')\right) p_W(\boldsymbol{w}) \, \mathrm{d}\boldsymbol{w} = \mathbb{E}_{W \sim p_W} \exp\left(\mathrm{j}2\pi W^\top (\boldsymbol{x} - \boldsymbol{x}')\right).$$

Hence, drawing i.i.d. frequency vectors $\boldsymbol{w}_j$ according to p_W yields an RF feature map (3) whose expected kernel is precisely κ. For instance, a Laplace kernel can be approximated if the rows of $\boldsymbol{W}$ are drawn i.i.d. from the Cauchy distribution [52].

More generally, under mild assumptions on a positive definite kernel κ, one can invoke Mercer's theorem [53] to similarly show the existence of a random feature map whose expected kernel, in the sense of (37), matches κ.

6 SKETCHING WITH REDUCED COMPUTATIONAL RESOURCES

The computational cost of sketching via (1) is heavily dependent on the feature map $\boldsymbol{\Phi}(\cdot)$. The computational cost of parameter estimation via (17) is also heavily dependent on $\boldsymbol{\Phi}(\cdot)$, since it often involves iterative application of $\boldsymbol{\Phi}(\cdot)$.

Often, the feature map is constructed as a randomized linear operation followed by a componentwise non-linear operation, *i.e.*,

$$\boldsymbol{\Phi}(\boldsymbol{x}) = \varrho(\boldsymbol{W}\boldsymbol{x}), \tag{46}$$

where $\boldsymbol{W}$ is a (randomly drawn) matrix of size $m \times d$ and $\varrho(\cdot)$ applies a scalar non-linear function identically to each element of the vector $\boldsymbol{W}\boldsymbol{x}$. For example, the feature maps described earlier for compressive PCA, GMM, and clustering all have this form. Reducing the computational cost of each stage has been the goal of several studies. For example, using a fast transform for $\boldsymbol{W}$ drastically reduces the memory and computational complexity demands relative to an explicit matrix. Also, quantized versions of the nonlinearity $\varrho(\cdot)$ are much more easily implemented in hardware than, say, the complex exponential nonlinearity $\varrho_{\mathrm{RF}}(\cdot) := \exp(-\mathrm{j}2\pi\,\cdot)$ used in the RF map (3). We discuss such constructions of $\boldsymbol{W}$ and $\varrho(\cdot)$ below.

6.1 SKETCHING WITH STRUCTURED RANDOM MATRICES

In most of our previous examples, we constructed $\boldsymbol{W}$ by drawing its m rows i.i.d. from the normal distribution $\mathcal{N}(\boldsymbol{0}, \sigma_w^2 \boldsymbol{I}_d)$ with some variance $\sigma_w^2 > 0$. Recall that, with the RF map $\varrho_{\mathrm{RF}}(\cdot)$, the rows of $\boldsymbol{W}$ correspond to the (d-dimensional) frequencies used when sampling the Fourier transform of the (empirical) data distribution. In any case, when $\boldsymbol{W}$ is an explicit matrix, the computational complexity of computing $\boldsymbol{\Phi}(\boldsymbol{x})$ of the form (46) is dominated by the matrix-vector product $\boldsymbol{W}\boldsymbol{x}$. Thus, it is on the order of md, which is also the order of the memory needed to store $\boldsymbol{W}$.

As an alternative to these approaches, it has been suggested to construct $\boldsymbol{W}$ as a structured random matrix with a fast implementation that mimics an i.i.d. Gaussian matrix. Multiple ways of accomplishing this goal have been proposed in the literature. We focus on the approach suggested in [54], which was successfully applied to compressive learning in [55]. There, the idea is to construct $\boldsymbol{W}$ as a vertical concatenation of $b = \lceil m/d \rceil$ blocks $\{\boldsymbol{B}_j\}_{j=1}^b$, each of size $d \times d$. These blocks have the form $\boldsymbol{B}_j = \boldsymbol{D}_j^{(0)} \boldsymbol{H} \boldsymbol{D}_j^{(1)} \boldsymbol{H} \boldsymbol{D}_j^{(2)} \boldsymbol{H} \boldsymbol{D}_j^{(3)}$, where $\boldsymbol{H}$ is the Walsh-Hadamard matrix and $\boldsymbol{D}_j^{(k)}$ are random diagonal matrices. In particular, the diagonal elements of $\boldsymbol{D}_j^{(1)}$, $\boldsymbol{D}_j^{(2)}$, and $\boldsymbol{D}_j^{(3)}$ are drawn i.i.d. from the uniform distribution over $\{-1, 1\}$, and the diagonal elements of $\boldsymbol{D}_j^{(0)}$ are drawn i.i.d. from the χ distribution with d degrees-of-freedom, which is the distribution of the norm of a d-variate Gaussian vector. This construction is depicted in Fig. 12. The fast Walsh-Hadamard transform offers an order $d \log d$-complexity implementation of the matrix-vector multiplication $\boldsymbol{H}\boldsymbol{x}$ and prevents the need to explicitly store $\boldsymbol{H}$. With this structured and fast incarnation of $\boldsymbol{W}$, the sketching complexity shrinks from order md to order $m \log d$. Moreover, since only the diagonal matrices need to be stored, the storage cost shrinks from order md to order m.

6.2 SKETCHING WITH QUANTIZED CONTRIBUTIONS

With the RF map (3), which is commonly used in compressive clustering and GMM, the non-linear operation $\varrho(\cdot)$ in (46) becomes $\varrho_{\mathrm{RF}}(\cdot) := \exp(-\mathrm{j}2\pi\,\cdot)$. Since implementing $\varrho_{\mathrm{RF}}(\cdot)$ with high accuracy is somewhat costly and not amenable to easy hardware acceleration, one might consider quantizing it. For

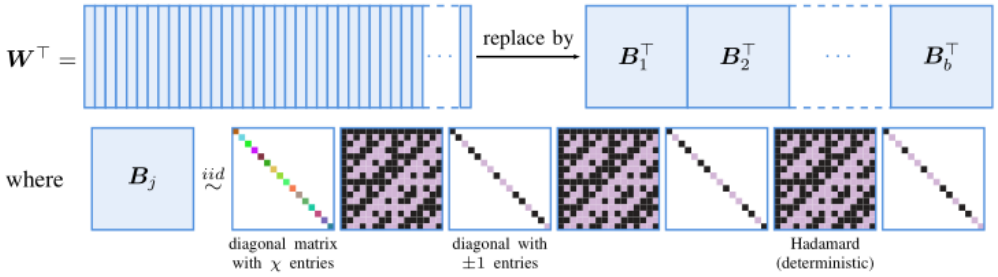


Fig. 12: Structured random matrix design from [54]. Each block is a composition of several Hadamard and diagonal matrices. For convenience, we draw $\boldsymbol{W}^\top$ instead of $\boldsymbol{W}$.

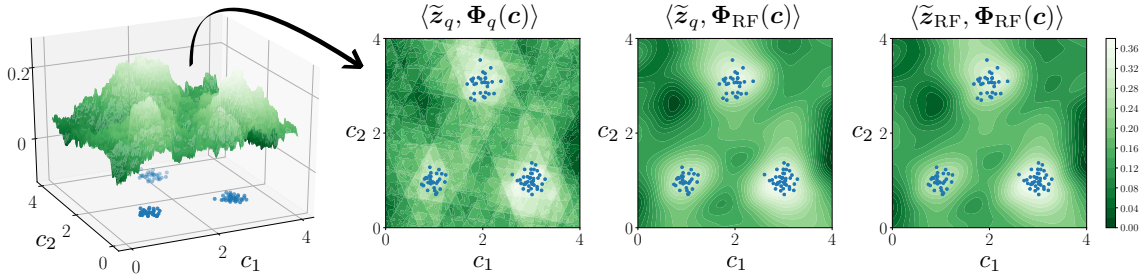


Fig. 13: Criterion $\langle \widetilde{\boldsymbol{z}}, \boldsymbol{\Phi}(\boldsymbol{c}) \rangle$ versus centroid hypothesis $\boldsymbol{c} = [c_1, c_2]^\top$ used by greedy parameter estimation algorithms in compressive clustering with $k = 3$ and a 2D dataset (in blue), with and without quantization of the sketch $\widetilde{\boldsymbol{z}}$ and/or feature map $\boldsymbol{\Phi}$. The surface plot on the left highlights the irregularity of $\boldsymbol{c} \mapsto \langle \widetilde{\boldsymbol{z}}_q, \boldsymbol{\Phi}_q(\boldsymbol{c}) \rangle$.

example, dropping the imaginary part of $\varrho_{\mathrm{RF}}(\cdot)$ and quantizing the real part to one bit of precision yields the 2π-periodic function $\varrho_q(\cdot) := \mathrm{sign}(\cos(2\pi\cdot))$ which is simply a "square wave".

To alleviate the effects of quantization, one can apply dithering [15] to the input. In this case, the feature map becomes $\boldsymbol{\Phi}_q(\boldsymbol{x}) = \varrho_q(\boldsymbol{W}\boldsymbol{x} + \boldsymbol{\xi}) \in \{-1, +1\}^m$, with i.i.d. dither components ξ_j drawn uniformly over $[0, 1)$. The effect of dithering is to make the quantized $\boldsymbol{\Phi}_q$ behave similarly to non-quantized $\boldsymbol{\Phi}_{\mathrm{RF}}$ *on average*. For instance, it was shown in [56] that for each $\boldsymbol{W}$, $\boldsymbol{x}$, $\boldsymbol{x}'$, and $\boldsymbol{\xi}$,

$$\langle \boldsymbol{\Phi}_q(\boldsymbol{x}), \boldsymbol{\Phi}_{\mathrm{RF}}(\boldsymbol{x}') \rangle \approx \mathbb{E}_{\boldsymbol{\xi}} \langle \boldsymbol{\Phi}_q(\boldsymbol{x}), \boldsymbol{\Phi}_{\mathrm{RF}}(\boldsymbol{x}') \rangle = c \langle \boldsymbol{\Phi}_{\mathrm{RF}}(\boldsymbol{x}), \boldsymbol{\Phi}_{\mathrm{RF}}(\boldsymbol{x}') \rangle, \tag{47}$$

where c is a constant. The approximation above is accurate for a typical draw of the m-dimensional dither vector $\boldsymbol{\xi}$ when the sketch dimension m is large enough [57]. Note that, when this dithered quantizer is used to compute a sketch $\widetilde{\boldsymbol{z}}_q := \frac{1}{n} \sum_{i=1}^n \boldsymbol{\Phi}_q(\boldsymbol{x}_i) = \frac{1}{n} \sum_{i=1}^n \varrho_q(\boldsymbol{W}\boldsymbol{x}_i + \boldsymbol{\xi})$, it is important to use the same dither realization $\boldsymbol{\xi}$ for all samples i.

The question then arises: when estimating parameters $\boldsymbol{\theta}$ via (31), how should we account for quantization in the sketch? Simply replacing the $\mathcal{A}(p_{\boldsymbol{\theta}_\ell})$ term with $\mathcal{A}_q(p_{\boldsymbol{\theta}_\ell}) := \mathbb{E}_{X \sim p_{\boldsymbol{\theta}_\ell}}[\boldsymbol{\Phi}_q(X)]$ may sound appealing. For example, with compressive GMM (18) or compressive clustering (20), this would mean minimizing $\left\| \widetilde{\boldsymbol{z}}_q - \sum_{\ell=1}^k \alpha_\ell \mathcal{A}_q(p_{\boldsymbol{\theta}_\ell}) \right\|^2$. However, the optimization problem (31) would become more challenging, as suggested by comparing the left two panels to the right two panels in Fig. 13, which plots the centroid-selection criterion $\langle \widetilde{\boldsymbol{z}}, \boldsymbol{\Phi}(\boldsymbol{c}) \rangle = \langle \widetilde{\boldsymbol{z}}, \mathcal{A}(p_{\boldsymbol{c}}) \rangle$ used by greedy algorithms. Also, this approach would not inherit the theoretical guarantees that were carefully established using the LRIP (33), which does not easily translate to the quantized case.

Instead, we suggest to use $\mathcal{A}_{\mathrm{RF}}(p_{\boldsymbol{\theta}_\ell}) := \mathbb{E}_{X\sim p_{\boldsymbol{\theta}_\ell}}[\boldsymbol{\Phi}_{\mathrm{RF}}(X)]$ for the $\mathcal{A}(p_{\boldsymbol{\theta}_\ell})$ term in (18), and to re-scale $\widetilde{\boldsymbol{z}}'_q = \widetilde{\boldsymbol{z}}_q/c$ with the constant c from (47). Indeed, thanks to (47), the resulting cost function $C(\boldsymbol{\theta}|\widetilde{\boldsymbol{z}}'_q) = \left\|\widetilde{\boldsymbol{z}}'_q - \sum_{\ell=1}^k \alpha_\ell \mathcal{A}_{\mathrm{RF}}(p_{\boldsymbol{\theta}_\ell})\right\|^2$ is, in expectation over $\boldsymbol{\xi}$, *exactly the same* (up a constant additive bias term) as the non-quantized cost function $C(\boldsymbol{\theta}|\widetilde{\boldsymbol{z}}_{\mathrm{RF}})$ [56]. The similarity between $C(\boldsymbol{\theta}|\widetilde{\boldsymbol{z}}'_q)$ and $C(\boldsymbol{\theta}|\widetilde{\boldsymbol{z}}_{\mathrm{RF}})$, even for a single realization of the m-dimensional dither vector $\boldsymbol{\xi}$, is suggested by comparing the right two panels in Fig. 13.

Empirical results [56] suggest that, when the sketch dimension is inflated by about 25%, this quantized compressive learning procedure yields the same performance as the non-quantized procedure. Moreover, accurate probabilistic bounds for approximation (47), established in [57], allow one to extend the theoretical compressive learning guarantees in (34) to this new cost function.

[Box 9] Potential of Sketching for Privacy-Aware Learning and Alternative Approaches

Given a dataset, privacy preservation can be achieved by asking a trusted dataholder to corrupt all queries of the dataset [58, 59] in a controlled manner. There are, however, challenges to this so-called *interactive* approach. For example, because the privacy-preserving effects of this corruption can often be diminished through the mining of multiple query responses (especially if the queries are adaptive), the per-query corruption levels must be designed with the type and total number of queries in mind. These corruption levels are often designed using a so-called "privacy budget," which is expended over multiple queries in order to meet an overall privacy level. Once the entire privacy budget has been used up, the data can no longer be accessed by a given datauser. Also, the dataholder must ensure that responses to *different* datausers cannot be combined in a way that circumvents the intended privacy preservation.

In contrast, the *non-interactive* approach [58, 59] is to publish an intermediate privacy-preserving synopsis of the dataset, to which the public is allowed unlimited access. For example, with a low-dimensional dataset, one could publish a privacy-preserving histogram of the data [60], from which aggregate statistics could be subsequently extracted. The non-interactive approach is attractive for several reasons. For example, there is no need to formulate nor allocate a privacy budget; it is sufficient to set an overall privacy level. Also, there is no need to worry about datausers sharing/combining data.

By adding noise to a sketch of the form (1), one can easily generate a privacy preserving synopsis of a dataset. By construction, such sketches capture the global statistics of the dataset $\mathcal{X} = \{\boldsymbol{x}_i\}_{i=1}^n$ while being relatively insensitive to each individual data sample $\boldsymbol{x}_i$, especially when the sample cardinality n is large. Also, when the original data is distributed across multiple devices, a privacy preserving global sketch can be constructed by first locally sketching at each device and then averaging those local sketches at a fusion center, as illustrated in Box 1. In this scenario, the local sketches will themselves be privacy preserving, which alleviates concerns about privacy leaks during data fusion.

7 PRIVACY PRESERVATION

In addition to its efficient use of computational resources, sketching is a promising tool for privacy-preserving machine learning. In numerous applications, such as when working with medical records, online surveys, or measurements coming from personal devices, data samples contain sensitive personal information and data providers ask that individuals' contributions to the dataset remain private, *i.e.*, not publicly discoverable. Learning from such data collections while protecting the privacy of individual contributors has become a crucial challenge [58, 59, 61].

A common way to preserve privacy is to have a trusted dataholder (or "curator") corrupt the response

to each query of the dataset [58] in a controlled manner. A query may ask for something as simple as counting the number of times a given event occurred, or it may ask for more sophisticated information that requires the dataholder to run an inference algorithm. As the corruption becomes more significant, the privacy guarantee gets stronger, but the quality of the response to the query (called the "utility") degrades. This can be conceptualized by a *privacy-utility tradeoff* [58, 61].

When sketching a dataset via (1), there is a very simple way to preserve privacy in any subsequent learning task: simply add i.i.d. noise (with appropriate distribution) to the sketch. The privacy level can be adjusted by changing the variance of that noise, as described below. This one-time approach to privacy preservation (more generally known as *privacy-preserving data publishing* [58]) has several benefits over the query-based approach to privacy preservation discussed in the previous paragraph. (See also Box 9.)

7.1 SKETCHING WITH DIFFERENTIAL PRIVACY GUARANTEES

Differential privacy [59] is a standard framework for privacy preservation that has a precise mathematical definition and is well-known in machine learning and signal processing (*e.g.*, [61]). When a given (randomized) learning pipeline is differentially private, its output depends negligibly on the presence or absence of any individual sample in the dataset. Differential privacy is robust to many forms of attack, such as when the adversary can access side information that nullifies privacy guarantees based on anonymization or mutual information measures (*e.g.*, when the adversary can control some of the data vectors $\boldsymbol{x}_i$, or can access additional databases that are correlated with the primary database).

For compressive learning methods, enforcing differential privacy guarantees is as simple as adding well-calibrated noise $\boldsymbol{v}$ to the usual sketch $\widetilde{\boldsymbol{z}}$, *i.e.*, constructing

$$\widetilde{\boldsymbol{s}}(\mathcal{X}) := \widetilde{\boldsymbol{z}}(\mathcal{X}) + \boldsymbol{v}, \tag{48}$$

where we find it helpful to explicitly denote the dependence of the dataset $\mathcal{X}$. We will assume that the realization $\boldsymbol{\Phi}(\cdot)$ of the random feature map is fixed and publicly known, in contrast to other approaches like [62, 63] that use linear mixing matrices as encryption keys to ensure privacy preservation. As a result, when we treat $\widetilde{\boldsymbol{s}}(\mathcal{X})$ as random, this is due to the randomness in $\boldsymbol{v}$, not the randomness in $\boldsymbol{\Phi}(\cdot)$ or $\mathcal{X}$.

Formally, the sketching mechanism $\widetilde{\boldsymbol{s}}(\cdot)$ is said to be *ϵ-differentially private* [59] if, for any dataset $\mathcal{X} = \{\boldsymbol{x}_i\}_{i=1}^n$ and "neighboring" dataset $\mathcal{X}' = \mathcal{X} \setminus \{\boldsymbol{x}_j\} \cup \{\boldsymbol{x}'_j\}$ that replaces the individual sample $\boldsymbol{x}_j$ by another sample $\boldsymbol{x}'_j$, and for any possible sketch outcome $\boldsymbol{s}$, we have that

$$\exp(-\epsilon) \leq \frac{p_{\widetilde{\boldsymbol{s}}(\mathcal{X})}(\boldsymbol{s})}{p_{\widetilde{\boldsymbol{s}}(\mathcal{X}')}(\boldsymbol{s})} \leq \exp(\epsilon). \tag{49}$$

Here, $\epsilon > 0$ plays the role of a privacy level: smaller ϵ implies a stronger privacy guarantee. In words, (49) says that, when ϵ is small, the densities of $\widetilde{\boldsymbol{s}}(\mathcal{X})$ and $\widetilde{\boldsymbol{s}}(\mathcal{X}')$ are almost indistinguishable, as depicted in Figure 14.

The condition (49) can be interpreted as bounding a "likelihood ratio" [64], a familiar quantity in signal processing. Consider two hypotheses: one that the dataset equals $\mathcal{X}$ (*i.e.*, includes $\boldsymbol{x}_j$), and the other that the dataset equals $\mathcal{X}'$ (*i.e.*, includes $\boldsymbol{x}'_j$ instead). Then $p_{\widetilde{\boldsymbol{s}}(\mathcal{X})}(\boldsymbol{s})$ would be the likelihood of observing private sketch $\boldsymbol{s}$ under the first hypothesis, while $p_{\widetilde{\boldsymbol{s}}(\mathcal{X}')}(\boldsymbol{s})$ would be the same for the second hypothesis. Say an adversary wanted to detect whether or not the dataset contains $\boldsymbol{x}_j$. By appropriately thresholding the likelihood ratio $p_{\widetilde{\boldsymbol{s}}(\mathcal{X})}(\boldsymbol{s})/p_{\widetilde{\boldsymbol{s}}(\mathcal{X}')}(\boldsymbol{s})$, one can obtain hypothesis tests that are optimal from various perspectives (*e.g.*, Bayes, minimax, Neyman-Pearson) [64, Ch. 2]. Thus, when (49) holds with small ϵ, it is fundamentally difficult for an adversary to determine whether $\boldsymbol{x}_j$ or $\boldsymbol{x}'_j$ was present in the sketch. Even

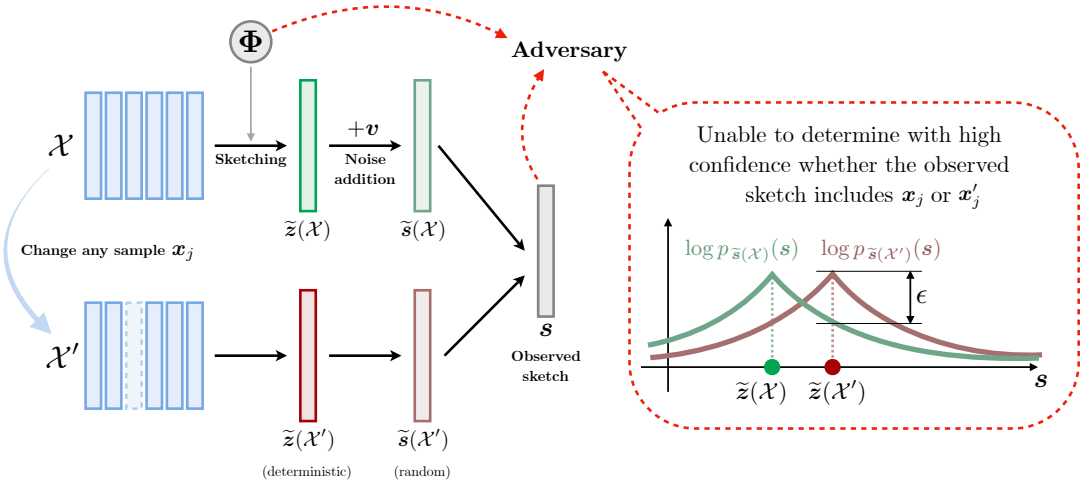


Fig. 14: When sketching with differential privacy, the output log-density of the sketch $\boldsymbol{s}$ remains close when changing one sample in the dataset (since (49) is equivalent to $|\log(p_{\widetilde{\boldsymbol{s}}(\mathcal{X})}(\boldsymbol{s})) - \log(p_{\widetilde{\boldsymbol{s}}(\mathcal{X}')}(\boldsymbol{s}))| \leq \epsilon$ for all possible $\boldsymbol{s}$). An adversary with knowledge of $\boldsymbol{\Phi}(\cdot)$ and $\boldsymbol{s}$—as symbolized by the red arrows—could then hardly decide whether a given sample $\boldsymbol{x}_j$ was used to compute the sketch or not.

if the adversary had non-trivial prior knowledge of the true hypothesis (as in so-called "linkage attacks," which make use of a second public dataset to which the target user contributed), (49) implies that—for any method—the probability of recovering the true hypothesis from the sketch is only slightly higher than that which is achievable *without* observing the sketch.

To ensure that the noisy sketch (48) is differentially private, it is sufficient to draw the noise $\boldsymbol{v}$ as i.i.d. Laplacian with appropriate variance. The variance needed to achieve a given privacy level ϵ can be determined by analyzing the so-called *sensitivity* of the noiseless sketch, *i.e.*, the biggest possible change that can result from removing one sample. When using the random Fourier feature map (3), which generates complex-valued $\widetilde{\boldsymbol{z}}(\mathcal{X})$, it has been established [65, 66] that it is sufficient for the real and imaginary components of $\boldsymbol{v}$ to be i.i.d. Laplacian with standard deviation $\sigma_v \propto \frac{m}{n\epsilon}$.

A weaker form of privacy, known as *approximate differential privacy* or (ϵ, δ)-*differential privacy* [59], can be attained by adding Gaussian noise $\boldsymbol{v}$ with smaller variance. For example, with RF features, it is sufficient for the real and imaginary components of $\boldsymbol{v}$ to be i.i.d. Gaussian with standard deviation $\sigma_v \propto \frac{\sqrt{m}}{n\epsilon}$.

The privacy-utility tradeoff facilitates the comparison of different privacy-preserving learning strategies. For example, given two strategies, one could match the privacy levels ϵ and compare utilities, or one could match utilities and compare privacy levels. For compressive learning, the utility of interest is the risk (recall (25)).

In this section, we focused on differential privacy. Other definitions of privacy exist in the literature, such as information-theoretic ones [67]. Likewise, cryptographic methods, such as fully homomorphic encryption [68], can be used to transmit and manipulate data in a secure and private manner, although this notion of "privacy" is quite different from the former ones. Additional work is needed to understand whether compressive learning is "private" according to definitions other than differential privacy.

8 PERSPECTIVES AND OPEN CHALLENGES

By averaging well-chosen randomized feature transformations over large training collections, sketching significantly compresses data in a way that facilitates provably accurate yet scalable learning from huge and/or streaming datasets, while simultaneously preserving privacy.

In this article, we described several approaches to accelerate the sketching process, including feature quantization and the use of randomized fast transforms. Another approach is to randomly mask each feature vector $\boldsymbol{\Phi}(\boldsymbol{x}_i)$ prior to averaging, *i.e.*, set a random subset of its components to zero. It has been established [66] that such random masking does not increase nor decrease the differential privacy level ϵ. But it does reduce the need to compute all entries of each feature vector, and thus reduces sketching complexity. Another promising approach consists of mixed analog-digital sketches, where, *e.g.*, optical processing units are used to significantly improve the energy efficiency of the linear stage [67].

When discussing methods *to learn* from a sketch, we focused on optimization-based approaches. Although heuristics based on orthogonal matching pursuit [11] [68] and approximate message passing [17] have been proposed that yield promising empirical results, performance guarantees for these approaches have yet to be established. Alternatives, such as total-variation minimization over the space of signed measures [34, 42, 69], principled greedy methods [70][71], and gradient flows on systems of particles [72] could be leveraged to make progress on this front, and black-box optimization could be used to learn from sketches computed by optical processing units.

As for *applications* of compressive learning, most of the current literature, and hence most of our article, has focused on unsupervised learning tasks. Further work is needed to develop compressive learning methods for supervised tasks like regression and classification [47]. For example, one approach to compressive classification was proposed in [73]: for each class $\ell = 1, \ldots, k$, one computes a sketch $\widetilde{\boldsymbol{z}}_\ell$ using only the training examples with label ℓ (*i.e.*, we sketch the k *conditional distributions* of the data). From those sketches, one could estimate the conditional densities of each class (using a mixture model, for example), from which a maximum likelihood (ML) or maximum a posteriori (MAP) classifier could be derived. Another approach is to perform classification directly in the compressed domain: to an unseen example $\boldsymbol{x}'$, we would assign the class ℓ that maximizes the correlation $\langle \widetilde{\boldsymbol{z}}_\ell, \boldsymbol{\Phi}(\boldsymbol{x}') \rangle$. This strategy can be interpreted [73] as compressively evaluating a Parzen-window classifier [51]. Further work is also needed on unsupervised matrix-factorization tasks like dictionary learning, low-rank matrix completion, and non-negative matrix factorization (NMF) [74].

REFERENCES

[1] F. Pérez-Cruz and O. Bousquet. "Kernel methods and their potential use in signal processing". In: *IEEE Signal Process. Mag.* 21.3 (2004).

[2] B. Schölkopf and A. Smola. *Learning with Kernels*. Adaptive Computation and Machine Learning. Cambridge: MIT Press, 2002.

[3] D. S. Rosenberg, V. Sindhwani, P. L. Bartlett, and P. Niyogi. "Multiview point cloud kernels for semisupervised learning [lecture notes]". In: *IEEE Signal Process. Mag.* 26.5 (2009).

[4] R. Jenssen. "Entropy-relevant dimensions in the kernel feature space: Cluster-capturing dimensionality reduction". In: *IEEE Signal Process. Mag.* 30.4 (2013).

[5] J. Arenas-Garcia, K. B. Petersen, G. Camps-Valls, and L. K. Hansen. "Kernel multivariate analysis framework for supervised subspace learning: A tutorial on linear and kernel multivariate methods". In: *IEEE Signal Process. Mag.* 30.4 (2013).

[6] S. Foucart and H. Rauhut. *A Mathematical Introduction to Compressive Sensing*. Springer, May 2012.

[7] N. Halko, P.-G. Martinsson, and J. A. Tropp. "Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions". In: *SIAM Rev.* 53.2 (2011).

[8] R. Gribonval, G. Blanchard, N. Keriven, and Y. Traonmilin. "Compressive Statistical Learning with Random Feature Moments". In: *1706.07180* (2017).

[9] T. K. Moon. “The expectation-maximization algorithm”. In: *IEEE Signal Process. Mag.* 13.6 (1996).

[10] D. A. Reynolds, T. F. Quatieri, and R. B. Dunn. “Speaker Verification Using Adapted Gaussian Mixture Models”. In: *Dig. Sig. Proc.* 10.1-3 (2000).

[11] N. Keriven, A. Bourrier, R. Gribonval, and P. Pérez. “Sketching for large-scale learning of mixture models”. In: *Inform. Inference* 7.3 (2017).

[12] A. Rahimi and B. Recht. “Random Features for Large Scale Kernel Machines”. In: *Proc. Neural Inform. Process. Syst. Conf.* (2007).

[13] N. Keriven, A. Deleforge, and A. Liutkus. “Blind Source Separation Using Mixtures of Alpha-Stable Distributions”. In: *Proc. IEEE Int. Conf. Acoust. Speech & Signal Process.* 2018.

[14] S. P. Lloyd. “Least Squares Quantization in PCM”. In: *IEEE Trans. Inform. Theory* 28.2 (1982).

[15] R. M. Gray and D. L. Neuhoff. “Quantization”. In: *IEEE Trans. Inform. Theory* 44.6 (1998).

[16] N. Keriven, N. Tremblay, Y. Traonmilin, and R. Gribonval. “Compressive K-means”. In: *Proc. IEEE Int. Conf. Acoust. Speech & Signal Process.* 2017.

[17] E. Byrne, A. Chatalic, R. Gribonval, and P. Schniter. “Sketched clustering via hybrid approximate message passing”. In: *IEEE Trans. Signal Process.* 67.17 (2019).

[18] M. P. Sheehan, M. S. Kotzagiannidis, and M. E. Davies. “Compressive Independent Component Analysis”. In: *2019 27th European Signal Processing Conference (EUSIPCO)*. IEEE. 2019.

[19] M. P. Sheehan, A. Gonon, and M. E. Davies. “Compressive Learning for Semi-Parametric Models”. In: *arXiv:1910.10024* (2019).

[20] G. Cormode, M. Garofalakis, P. J. Haas, and C. Jermaine. “Synopses for Massive Data”. In: *Foundations and Trends in Databases* 4.1 (2011).

[21] G. Cormode and S. Muthukrishnan. “An improved data stream summary: the count-min sketch and its applications”. In: *Journal of Algorithms* 55.1 (2005).

[22] D. Achlioptas. “Database-friendly random projections: Johnson-Lindenstrauss with binary coins”. In: *Journal of Computer and System Sciences* 66.4 (2003).

[23] M. W. Mahoney. “Randomized algorithms for matrices and data”. In: *Foundations and Trends in Machine Learning* 3 (2010).

[24] C. Boutsidis, A. Zouzias, and P. Drineas. “Random Projections for k-means Clustering”. In: *Advances in Neural Information Processing Systems*. 2010.

[25] D. P. Woodruff. “Sketching as a tool for numerical linear algebra”. In: *Foundations and Trends in Theoretical Computer Science* 10 (2014).

[26] D. Achlioptas, F. McSherry, and B. Schölkopf. “Sampling techniques for kernel methods”. In: *Advances in Neural Information Processing Systems*. 2002.

[27] D. Feldman, M. Monemizadeh, C. Sohler, and D. P. Woodruff. “Coresets and sketches for high dimensional subspace approximation problems”. In: *Proceedings of the Annual ACM-SIAM Symposium on Discrete Algorithms* 1 (2010).

[28] C. Williams and M. W. Seeger. “Using the Nystrom Method to Speed Up Kernel Machines”. In: *Advances in Neural Information Processing Systems*. Vol. 13. 2001.

[29] A. R. Hall. *Generalized Method of Moments*. Oxford University Press, 2005.

[30] E. J. Candès and M. B. Wakin. “An introduction to compressive sampling”. In: *IEEE Signal Process. Mag.* 25.2 (2008).

[31] W. B. Johnson and J. Lindenstrauss. “Extensions of Lipschitz mappings into a Hilbert space”. In: *Contemporary mathematics* 26.189-206 (1984).

[32] E. J. Candès and T. Tao. “Decoding by linear programming”. In: *IEEE Trans. Inform. Theory* 51.12 (2005).

[33] E. J. Candès, J. Romberg, and T. Tao. “Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information”. In: *IEEE Trans. Inform. Theory* 52.2 (2006).

[34] E. J. Candès and C. Fernandez-Granda. “Towards a mathematical theory of super-resolution”. In: *Commun. Pure & Appl. Math.* 67.6 (2014).

[35] K. Bredies and H. K. Pikkarainen. “Inverse problems in spaces of measures”. In: *ESAIM - Control, Optimisation and Calculus of Variations* 19.1 (2013).

[36] N. Thaper, S. Guha, P. Indyk, and N. Koudas. “Dynamic multidimensional histograms”. In: *SIGMOD ’02: Proceedings of the 2002 ACM SIGMOD international conference on Management of data*. New York, NY, USA: ACM, 2002.

[37] M. Lustig, D. L. Donoho, J. M. Santos, and J. M. Pauly. “Compressed sensing MRI”. In: *IEEE Signal Process. Mag.* 25.2 (2008).

[38] J. A. Fessler. “Optimization Methods for Magnetic Resonance Image Reconstruction: Key Models and Optimization Algorithms”. In: *IEEE Signal Processing Magazine* 37.1 (2020).

[39] Y. Wiaux, L. Jacques, G. Puy, A. M. M. Scaife, and P. Vandergheynst. “Compressed sensing imaging techniques for radio interferometry”. In: *Monthly Notices of the Royal Astronomical Society* 395.3 (May 2009).

[40] M. Cetin, I. Stojanovic, O. Onhon, K. Varshney, S. Samadi, W. C. Karl, and A. S. Willsky. “Sparsity-driven synthetic aperture radar imaging: Reconstruction, autofocusing, moving targets, and compressed sensing”. In: *IEEE Signal Process. Mag.* (2014).

[41] M. S. Greco, J. Li, T. Long, and A. Zoubir. “Advances in Radar Systems for Modern Civilian and Commercial Applications: Part 1 [From the Guest Editors]”. In: *IEEE Signal Process. Mag.* 36.4 (2019).

[42] Q. Denoyelle, V. Duval, G. Peyré, and E. Soubies. “The sliding Frank–Wolfe algorithm and its application to super-resolution microscopy”. English. In: *Inverse problems* 36.1 (Jan. 2020).

[43] C. Poon, N. Keriven, and G. Peyré. “The geometry of off-the-grid compressed sensing”. In: *arXiv:1802.08464* (2020). arXiv: 1802.08464.

[44] Y. Chen and Y. Chi. “Harnessing Structures in Big Data via Guaranteed Low-Rank Matrix Estimation: Recent Theory and Fast Algorithms via Convex and Nonconvex Optimization”. In: *IEEE Signal Processing Magazine* 35.4 (2018).

[45] R. Ge, J. D. Lee, and T. Ma. “Matrix Completion has No Spurious Local Minimum”. In: *Advances in Neural Information Processing Systems 29*. Ed. by D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett. Curran Associates, Inc., 2016.

[46] V. Vapnik. *The Nature of Statistical Learning Theory*. Springer, 2013.

[47] S. Shalev-Shwartz and S. Ben-David. *Understanding Machine Learning*. Cambridge: Cambridge University Press, 2009.

[48] R. Rubinstein, A. M. Bruckstein, and M. Elad. “Dictionaries for Sparse Representation Modeling”. In: *Proceedings of the IEEE* 98.6 (2010).

[49] A. Gretton, K. M. Borgwardt, M. J. Rasch, B. Schölkopf, and A. J. Smola. “A Kernel Method for the Two-Sample Problem”. In: *Advances in Neural Information Processing Systems (NIPS)*. 2007. eprint: 0805.2368.

[50] B. K. Sriperumbudur, A. Gretton, K. Fukumizu, B. Schölkopf, and G. R. Lanckriet. “Hilbert space embeddings and metrics on probability measures”. In: *The Journal of Machine Learning Research* 11 (2010).

[51] Z.-W. Pan, D.-H. Xiang, Q.-W. Xiao, and D.-X. Zhou. “Parzen windows for multi-class classification”. In: *Journal of complexity* 24.5-6 (2008).

[52] P. T. Boufounos, S. Rane, and H. Mansour. “Representation and coding of signal geometry”. In: *Inform. Inference* 6.4 (2017).

[53] F. Bach. “On the Equivalence between Kernel Quadrature Rules and Random Feature Expansions”. In: *Journal of Machine Learning Research* (2017).

[54] F. X. Yu, A. T. Suresh, K. M. Choromanski, D. N. Holtmann-Rice, and S. Kumar. “Orthogonal Random Features”. In: *Proc. Neural Inform. Process. Syst. Conf.* 2016.

[55] A. Chatalic, R. Gribonval, and N. Keriven. “Large-Scale High-Dimensional Clustering with Fast Sketching”. In: *Proc. IEEE Int. Conf. Acoust. Speech & Signal Process.* 2018.

[56] V. Schellekens and L. Jacques. “Quantized Compressive K-Means”. In: *IEEE Signal Process. Lett.* 25.8 (Aug. 2018).

[57] V. Schellekens and L. Jacques. “Breaking the waves: asymmetric random periodic features for low-bitrate kernel machines”. In: (2020). arXiv: 2004.06560.

[58] B. C. M. Fung, K. Wang, R. Chen, and P. S. Yu. “Privacy-preserving Data Publishing: A Survey of Recent Developments”. In: *ACM Comput. Surv.* 42.4 (June 2010).

[59] C. Dwork and A. Roth. “The Algorithmic Foundations of Differential Privacy”. In: *Theoretical Computer Science* 9.3-4 (2014).

[60] W. Qardaji, W. Yang, and N. Li. “Differentially private grids for geospatial data”. In: *2013 IEEE 29th international conference on data engineering (ICDE)*. IEEE. 2013.

[61] A. D. Sarwate and K. Chaudhuri. “Signal processing and machine learning with differential privacy: Algorithms and challenges for continuous data”. In: *IEEE Signal Process. Mag.* 30.5 (2013).

[62] M. Testa, D. Valsesia, T. Bianchi, and E. Magli. *Compressed Sensing for Privacy-Preserving Data Processing*. Springer, 2019.

[63] S. Rane and P. T. Boufounos. “Privacy-preserving nearest neighbor methods: Comparing signals without revealing them”. In: *IEEE Signal Process. Mag.* 30.2 (2013).

[64] H. V. Poor. *An Introduction to Signal Detection and Estimation*. Springer Science & Business Media, 2013.

[65] V. Schellekens, A. Chatalic, F. Houssiau, Y.-A. de Montjoye, L. Jacques, and R. Gribonval. “Differentially Private Compressive k-means”. In: *Proc. IEEE Int. Conf. Acoust. Speech & Signal Process.* 2019.

[66] A. Chatalic, V. Schellekens, F. Houssiau, Y.-A. de Montjoye, L. Jacques, and R. Gribonval. “Compressive Learning with Privacy Guarantees”. In: *Preprint, to appear. hal-02496896* (Mar. 3, 2020).

[67] A. Saade, F. Caltagirone, I. Carron, L. Daudet, A. Drémeau, S. Gigan, and F. Krzakala. “Random projections through multiple optical scattering”. In: *Proc. IEEE Int. Conf. Acoust. Speech & Signal Process.* 2016.

[68] P. Jain, A. Tewari, and I. S. Dhillon. “Orthogonal matching pursuit with replacement”. In: *Proc. Neural Inform. Process. Syst. Conf.* 2011.

[69] N. Boyd, G. Schiebinger, and B. Recht. “The Alternating Descent Conditional Gradient Method for Sparse Inverse Problems”. In: *SIAM J. Optim.* 27.2 (2017).

[70] Y. Traonmilin and J.-F. Aujol. “The basins of attraction of the global minimizers of the non-convex sparse spikes estimation problem”. In: *arXiv:1811.12000* (2018).

[71] C. Elvira, R. Gribonval, C. Herzet, and C. Soussen. “A case of exact recovery using OMP with continuous dictionaries”. In: *Proc. Intl. Conf. on Curves and Surfaces*. 2018.

[72] L. Chizat and F. Bach. “On the Global Convergence of Gradient Descent for Over-parameterized Models using Optimal Transport”. In: *Advances in Neural Information Processing System*. 2018.

[73] V. Schellekens and L. Jacques. “Compressive Classification (Machine Learning without learning)”. In: *arXiv:1812.01410* (2018).

[74] M. Udell, C. Horn, R. Zadeh, and S. Boyd. “Generalized Low Rank Models”. In: *Found. Trends Mach. Learn.* 9.1 (2016).