Sketched Clustering via Hybrid Approximate Message Passing

Evan Byrne, Antoine Chatalic, Rémi Gribonval, Philip Schniter

Sketched Clustering via Hybrid Approximate Message Passing

Evan Byrne,* Antoine Chatalic,† Rémi Gribonval†, *IEEE Fellow*, and Philip Schniter*, *IEEE Fellow*

***Abstract*—In sketched clustering, a dataset of T samples is first sketched down to a vector of modest size, from which the centroids are subsequently extracted. Advantages include i) reduced storage complexity and ii) centroid extraction complexity independent of T. For the sketching methodology recently proposed by Keriven, et al., which can be interpreted as a random sampling of the empirical characteristic function, we propose a sketched clustering algorithm based on approximate message passing. Numerical experiments suggest that our approach is more efficient than the state-of-the-art sketched clustering algorithm "CL-OMPR" (in both computational and sample complexity) and more efficient than k-means++ when T is large.**

***Index Terms*—clustering algorithms, data compression, compressed sensing, approximate message passing**

I. Introduction

Given a dataset $\boldsymbol{X} \triangleq [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_T] \in \mathbb{R}^{N \times T}$ comprising T samples of dimension N, the standard clustering problem is to find K centroids $\boldsymbol{C} \triangleq [\boldsymbol{c}_1, \ldots, \boldsymbol{c}_K] \in \mathbb{R}^{N \times K}$ that minimize the sum of squared errors (SSE)

$$\text{SSE}(\boldsymbol{X}, \boldsymbol{C}) \triangleq \frac{1}{T} \sum_{t=1}^{T} \min_{k} \|\boldsymbol{x}_t - \boldsymbol{c}_k\|_2^2. \tag{1}$$

Finding the optimal $\boldsymbol{C}$ is an NP-hard problem [1]. Thus, many heuristic approaches have been proposed, such as the *k-means* algorithm [2,3]. Because k-means can get trapped in bad local minima, robust variants have been proposed, such as *k-means++* [4], which uses a careful random initialization procedure to yield solutions with SSE that have on average $\leq 8(\ln K + 2)$ times the minimal SSE. The computational complexity of k-means++ scales as $O(TKNI)$, with I the number of iterations, which is impractical when T is large.

A. Sketched Clustering

In *sketched clustering* [5,6,7], the dataset $\boldsymbol{X}$ is first sketched down to a vector $\boldsymbol{y}$ with $M = O(KN)$ components, from which the centroids $\boldsymbol{C}$ are subsequently extracted. In the typical case that $K \ll T$, the sketch consumes much less memory than the original dataset. If the sketch can be performed efficiently, then—since the complexity of centroid-extraction is invariant to T—sketched clustering may be more efficient than direct clustering methods when T is large. Note, for example, that k-means++ processes the T data samples in $\boldsymbol{X}$ at every iteration, whereas sketched clustering processes the T data samples in $\boldsymbol{X}$ only once, during the sketching step.

In this work, we focus on sketches of the type proposed by Keriven et al. in [5,6], which use $\boldsymbol{y} = [y_1, \ldots, y_M]^\mathsf{T}$ with

$$y_m = \frac{1}{T} \sum_{t=1}^{T} \exp(\mathrm{j}\boldsymbol{w}_m^\mathsf{T} \boldsymbol{x}_t) \tag{2}$$

and randomly generated $\boldsymbol{W} \triangleq [\boldsymbol{w}_1, \ldots, \boldsymbol{w}_M]^\mathsf{T} \in \mathbb{R}^{M \times N}$. Note that y_m in (2) can be interpreted as a sample of the empirical characteristic function [8], i.e.,

$$\phi(\boldsymbol{w}_m) = \int_{\mathbb{R}^N} p(\boldsymbol{x}) \exp(\mathrm{j}\boldsymbol{w}_m^\mathsf{T} \boldsymbol{x}) \, \mathrm{d}\boldsymbol{x} \tag{3}$$

under the empirical distribution $p(\boldsymbol{x}) = \frac{1}{T} \sum_{t=1}^{T} \delta(\boldsymbol{x} - \boldsymbol{x}_t)$, with Dirac $\delta(\cdot)$. Here, each $\boldsymbol{w}_m$ can be interpreted as a multidimensional frequency sample. The process of sketching $\boldsymbol{X}$ down to $\boldsymbol{y}$ via (2) costs $O(TMN)$ operations, but it can be performed efficiently in an online and/or distributed manner.

To recover the centroids $\boldsymbol{C}$ from $\boldsymbol{y}$, the state-of-the-art algorithm is *compressed learning via orthogonal matching pursuit with replacement* (CL-OMPR) [5,6]. It aims to solve

$$\arg\min_{\boldsymbol{C}} \min_{\boldsymbol{\alpha}: \mathbf{1}^\mathsf{T}\boldsymbol{\alpha}=1} \sum_{m=1}^{M} \left| y_m - \sum_{k=1}^{K} \alpha_k \exp(\mathrm{j}\boldsymbol{w}_m^\mathsf{T} \boldsymbol{c}_k) \right|^2 \tag{4}$$

using a greedy heuristic inspired by the *orthogonal matching pursuit* (OMP) algorithm [9] popular in compressed sensing. With sketch length $M \geq 10KN$, CL-OMPR typically recovers centroids of similar or better quality to those attained with k-means++. One may wonder, however, whether it is possible to recover accurate centroids with sketch lengths closer to the counting bound $M = KN$. Also, since CL-OMPR's computational complexity is $O(MNK^2)$, one may wonder whether it is possible to recover accurate centroids with computational complexity $O(MNK)$.

B. Contributions

To recover the centroids $\boldsymbol{C}$ from a sketch $\boldsymbol{y}$ of the form in (2), we propose the *compressive learning via approximate message passing* (CL-AMP) algorithm, with computational

*E. Byrne (byrne.133@osu.edu) and P. Schniter (schniter.1@osu.edu) are with the Department of Electrical and Computer Engineering at The Ohio State University, Columbus, OH, USA.

†A. Chatalic (antoine.chatalic@irisa.fr) and R. Gribonval (remi.gribonval@inria.fr) are with Univ. Rennes, Inria, CNRS, IRISA, France. A. Chatalic received a travel grant from the French research network GdR MIA to visit The Ohio State University.

Please direct all correspondence to Philip Schniter, Dept. ECE, 2015 Neil Ave., Columbus OH 43210, phone 614.247.6488, fax 614.292.7596.

*E. Byrne and P. Schniter acknowledge support from NSF grant 1716388 and MIT Lincoln Labs.

Portions of this work were presented at the 2017 Asilomar Conference of Signals, Systems, and Computers.

complexity $O(MNK)$. Numerical experiments show that CL-AMP accurately recovers centroids from sketches of length $M = 2KN$ in most cases, which is an improvement over CL-OMPR. Experiments also show that CL-AMP recovers centroids faster and more accurately than k-means++ in certain operating regimes, such as when T is large.

We proposed a simple incarnation of the CL-AMP algorithm in the conference paper [10], with derivation details omitted due to space limitations. In this paper, we present the full derivation of CL-AMP with an improved initialization and hyperparameter tuning scheme, and a much more comprehensive set of numerical experiments.

The remainder of the paper is organized as follows. In Section II, we derive CL-AMP after reviewing relevant background on approximate message passing (AMP) algorithms. In Section III, we present numerical experiments using synthetic and MNIST data, and we apply CL-AMP to multidimensional frequency estimation. In Section IV, we conclude.

II. Compressive Learning via AMP

A. High-Dimensional Inference Framework

CL-AMP treats centroid recovery as a high-dimensional inference problem rather than an optimization problem like minimizing (1) or (4). In particular, it models the data using a Gaussian mixture model (GMM)

$$\boldsymbol{x}_t \sim \sum_{k=1}^{K} \alpha_k \mathcal{N}(\boldsymbol{c}_k, \boldsymbol{\Phi}_k), \tag{5}$$

where the centroids $\boldsymbol{c}_k$ act as the GMM means, and the GMM weights α_k and covariances $\boldsymbol{\Phi}_k$ are treated as unknown parameters. To recover the centroids $\boldsymbol{C} \triangleq [\boldsymbol{c}_1, \ldots, \boldsymbol{c}_K]$ from $\boldsymbol{y}$, CL-AMP computes an approximation to the MMSE estimate

$$\widehat{\boldsymbol{C}} = \mathbb{E}\{\boldsymbol{C} \,|\, \boldsymbol{y}\}, \tag{6}$$

where the expectation is taken over the posterior density

$$p(\boldsymbol{C}|\boldsymbol{y}) \propto p(\boldsymbol{y}|\boldsymbol{C}) p(\boldsymbol{C}). \tag{7}$$

In (7), $p(\boldsymbol{y}|\boldsymbol{C})$ is the likelihood function of $\boldsymbol{C}$, and $p(\boldsymbol{C})$ is the prior density on $\boldsymbol{C}$. The dependence of $p(\boldsymbol{y}|\boldsymbol{C})$ on $\{\alpha_k\}$ and $\{\boldsymbol{\Phi}_k\}$ will be detailed in the sequel.

The form of the sketch in (2) implies that, when conditioning on the centroids $\boldsymbol{C}$ (and the frequencies $\boldsymbol{W}$), the elements of $\boldsymbol{y}$ can be modeled as i.i.d. In other words, the sketch $\boldsymbol{y}$ follows a generalized linear model (GLM) [11]. To make this precise, let us define the normalized frequency vectors

$$\boldsymbol{a}_m \triangleq \boldsymbol{w}_m / g_m \text{ with } g_m \triangleq \|\boldsymbol{w}_m\| \tag{8}$$

and the (normalized) transform outputs

$$\boldsymbol{z}_m^{\mathsf{T}} \triangleq \boldsymbol{a}_m^{\mathsf{T}} \boldsymbol{C} \in \mathbb{R}^K. \tag{9}$$

Then $p(\boldsymbol{y}|\boldsymbol{C})$ takes the form of a GLM, i.e.,

$$p(\boldsymbol{y}|\boldsymbol{C}) = \prod_{m=1}^{M} p_{\mathsf{y}|\mathsf{z}}(y_m | \boldsymbol{a}_m^{\mathsf{T}} \boldsymbol{C}), \tag{10}$$

for a conditional pdf $p_{\mathsf{y}|\mathsf{z}}$ that will be detailed in the sequel.

From (2) and the definitions of $\boldsymbol{a}_m$ and g_m in (8), we have

$$y_m = \frac{1}{T} \sum_{t=1}^{T} \exp(\mathrm{j} \boldsymbol{w}_m^{\mathsf{T}} \boldsymbol{x}_t) \tag{11}$$

$$\approx \mathbb{E}\{\exp(\mathrm{j} \boldsymbol{w}_m^{\mathsf{T}} \boldsymbol{x}_t)\} \tag{12}$$

$$= \sum_{k=1}^{K} \alpha_k \exp\Bigg(\mathrm{j} g_m \underbrace{\boldsymbol{a}_m^{\mathsf{T}} \boldsymbol{c}_k}_{\triangleq\, z_{mk}} - \frac{g_m^2}{2} \underbrace{\boldsymbol{a}_m^{\mathsf{T}} \boldsymbol{\Phi}_k \boldsymbol{a}_m}_{\triangleq\, \tau_{mk}} \Bigg), \tag{13}$$

where (12) holds under large T and (13) follows from the facts

$$\boldsymbol{w}_m^{\mathsf{T}} \boldsymbol{x}_t \sim \sum_{k=1}^{K} \alpha_k \mathcal{N}(g_m z_{mk}, g_m^2 \tau_{mk}) \tag{14}$$

under (5) and the following well-known result [12, p.153]

$$\mathbb{E}\{e^{\mathrm{j}x}\} = \exp\left(\mathrm{j}\mu - \sigma^2/2\right) \text{ when } x \sim \mathcal{N}(\mu, \sigma^2). \tag{15}$$

For $\boldsymbol{a}_m$ distributed uniformly on the sphere, the elements $\{\tau_{mk}\}_{m=1}^{M}$ in (13) concentrate as $N \to \infty$ [13], in that

$$\tau_{mk} \xrightarrow{p} \mathbb{E}\{\tau_{mk}\} = \mathrm{tr}(\boldsymbol{\Phi}_k)/N \triangleq \tau_k, \tag{16}$$

as long as the peak-to-average eigenvalue ratio of $\boldsymbol{\Phi}_k$ remains bounded. Thus, for large T and N, (13) and (16) imply that

$$y_m = \sum_{k=1}^{K} \alpha_k \exp\left(\mathrm{j} g_m z_{mk} - \frac{g_m^2 \tau_k}{2} \right), \tag{17}$$

which can be rephrased as

$$p_{\mathsf{y}|\mathsf{z}}(y_m | \boldsymbol{z}_m; \boldsymbol{\alpha}, \boldsymbol{\tau}) = \delta\Bigg(y_m - \sum_{k=1}^{K} \alpha_k \exp\left(\mathrm{j} g_m z_{mk} - \frac{g_m^2 \tau_k}{2} \right) \Bigg) \tag{18}$$

where $\boldsymbol{\tau} \triangleq [\tau_1, \ldots, \tau_K]^{\mathsf{T}}$ and $\boldsymbol{\alpha} \triangleq [\alpha_1, \ldots, \alpha_K]^{\mathsf{T}}$ are hyperparameters of the GLM that will be estimated from $\boldsymbol{y}$.

For the CL-AMP framework, any prior of the form

$$p(\boldsymbol{C}) = \prod_{n=1}^{N} p_{\mathsf{c}}(\boldsymbol{c}_n^{\mathsf{T}}) \tag{19}$$

is admissible, where (with some abuse of notation) $\boldsymbol{c}_n^{\mathsf{T}}$ denotes the nth row of $\boldsymbol{C}$. For all experiments in Section III, we used the trivial prior $p(\boldsymbol{C}) \propto 1$.

In summary, CL-AMP aims to compute the MMSE estimate of $\boldsymbol{C} \in \mathbb{R}^{N \times K}$ from the sketch $\boldsymbol{y} \in \mathbb{C}^M$ under the prior $\boldsymbol{C} \sim \prod_{n=1}^{N} p_{\mathsf{c}}(\boldsymbol{c}_n)$ from (19) and the likelihood $\boldsymbol{y} \sim \prod_{m=1}^{M} p_{\mathsf{y}|\mathsf{z}}(y_m | \boldsymbol{z}_m; \boldsymbol{\alpha}, \boldsymbol{\tau})$ from (18), where $\boldsymbol{z}_m^{\mathsf{T}}$ is the mth row of $\boldsymbol{Z} = \boldsymbol{A}\boldsymbol{C} \in \mathbb{R}^{M \times K}$ and $\boldsymbol{A} \in \mathbb{R}^{M \times N}$ is a large random matrix with rows $\{\boldsymbol{a}_m^{\mathsf{T}}\}$ distributed uniformly on the unit sphere. CL-AMP estimates the values of $\boldsymbol{\alpha}$ and $\boldsymbol{\tau}$ from the sketch prior to estimating $\boldsymbol{C}$, as detailed in the sequel.

B. Approximate Message Passing

Exactly computing the MMSE estimate of $\boldsymbol{C}$ from $\boldsymbol{y}$ is impractical due to the form of $p_{\mathsf{y}|\mathsf{z}}$. Instead, one might consider approximate inference via the sum-product algorithm (SPA), but even the SPA is intractable due to the form of $p_{\mathsf{y}|\mathsf{z}}$. Given the presence of a large random matrix $\boldsymbol{A}$ in the problem

formulation, we instead leverage *approximate message passing* (AMP) methods. In particular, we propose to apply the *simplified hybrid generalized AMP* (SHyGAMP) methodology from [14], while simultaneously estimating $\boldsymbol{\alpha}$ and $\boldsymbol{\tau}$ through expectation maximization (EM). A brief background on AMP methods will now be provided to justify our approach.

The original AMP algorithm of Donoho, Maleki, and Montanari [15] was designed to estimate i.i.d. $\boldsymbol{c}$ under the standard linear model (i.e., $\boldsymbol{y} = \boldsymbol{A}\boldsymbol{c} + \boldsymbol{n}$ with known $\boldsymbol{A} \in \mathbb{R}^{M\times N}$ and additive white Gaussian noise $\boldsymbol{n}$). The generalized AMP (GAMP) algorithm of Rangan [16] extended AMP to the generalized linear model (i.e., $\boldsymbol{y} \sim p(\boldsymbol{y}|\boldsymbol{z})$ for $\boldsymbol{z} = \boldsymbol{A}\boldsymbol{c}$ and separable $p(\boldsymbol{y}|\boldsymbol{z}) = \prod_{m=1}^{M} p(y_m|z_m)$). Both AMP and GAMP give accurate approximations of the SPA under large i.i.d. sub-Gaussian $\boldsymbol{A}$, while maintaining a computational complexity of only $O(MN)$. Furthermore, both can be rigorously analyzed via the state-evolution framework, which proves that they compute MMSE optimal estimates of $\boldsymbol{c}$ in certain regimes [17].

A limitation of AMP [15] and GAMP [16] is that they treat only problems with i.i.d. estimand $\boldsymbol{c}$ and separable likelihood $p(\boldsymbol{y}|\boldsymbol{z}) = \prod_{m=1}^{M} p(y_m|z_m)$. Thus, *Hybrid GAMP* (HyGAMP) [18] was developed to tackle problems with a structured prior and/or likelihood. HyGAMP could be applied to the compressive learning problem described in Section II-A, but it would require computing and inverting $O(N{+}M)$ covariance matrices of dimension K at each iteration. For this reason, we instead apply the *simplified HyGAMP* (SHyGAMP) algorithm from [14], which uses diagonal covariance matrices in HyGAMP to reduce its computational complexity. As described in [14], SHyGAMP can be readily combined with the EM algorithm to learn the hyperparameters $\boldsymbol{\alpha}$ and $\boldsymbol{\tau}$.

C. SHyGAMP

The SHyGAMP algorithm was proposed and described in detail in [14]; we provide only a brief review here. Algorithm 1 summarizes the SHyGAMP algorithm using the language of Section II-A. In lines 10-11, with some abuse of notation, we use $\boldsymbol{c}_n^{\mathsf{T}}$ to denote the nth row of the centroid matrix $\boldsymbol{C}$ (where in (5) we used $\boldsymbol{c}_k$ to denote the kth column of $\boldsymbol{C}$). We also use $\widehat{\boldsymbol{P}} \triangleq [\widehat{\boldsymbol{p}}_1, \dots, \widehat{\boldsymbol{p}}_M]^{\mathsf{T}}$, $\widehat{\boldsymbol{Z}} \triangleq [\widehat{\boldsymbol{z}}_1, \dots, \widehat{\boldsymbol{z}}_M]^{\mathsf{T}}$, $\widehat{\boldsymbol{R}} \triangleq [\widehat{\boldsymbol{r}}_1, \dots, \widehat{\boldsymbol{r}}_N]^{\mathsf{T}}$, $\oslash$ for componentwise division, and $\odot$ for componentwise multiplication. In the sequel, covariance matrices will be denoted by (superscripted) $\boldsymbol{Q}$ and vectors of their diagonal elements denoted by (superscripted) $\boldsymbol{q}$. A brief interpretation of SHyGAMP is now provided.

At each iteration, lines 4-5 of Algorithm 1 generate the posterior mean and covariance of the transform outputs $\boldsymbol{z}_m$ from (9) under a likelihood $p_{\mathsf{y}|\mathsf{z}}$ like (18) and the "pseudo" prior $\boldsymbol{z}_m \sim \mathcal{N}(\widehat{\boldsymbol{p}}_m, \boldsymbol{Q}^{\mathsf{p}})$, where $\widehat{\boldsymbol{p}}_m$ and $\boldsymbol{Q}^{\mathsf{p}} = \mathrm{Diag}(\boldsymbol{q}^{\mathsf{p}})$ are updated at each SHyGAMP iteration. Thus, the pdf used for the covariance and expectation in lines 4-5 is

$$p_{\mathsf{z}|\mathsf{y},\mathsf{p}}(\boldsymbol{z}_m|y_m, \widehat{\boldsymbol{p}}_m; \boldsymbol{Q}^{\mathsf{p}}, \boldsymbol{\alpha}, \boldsymbol{\tau}) = \frac{p_{\mathsf{y}|\mathsf{z}}(y_m|\boldsymbol{z}_m; \boldsymbol{\alpha}, \boldsymbol{\tau})\mathcal{N}(\boldsymbol{z}_m; \widehat{\boldsymbol{p}}_m, \boldsymbol{Q}^{\mathsf{p}})}{\int p_{\mathsf{y}|\mathsf{z}}(y_m|\boldsymbol{z}'_m; \boldsymbol{\alpha}, \boldsymbol{\tau})\mathcal{N}(\boldsymbol{z}'_m; \widehat{\boldsymbol{p}}_m, \boldsymbol{Q}^{\mathsf{p}})\,\mathrm{d}\boldsymbol{z}'_m}. \tag{20}$$

Algorithm 1 SHyGAMP

Require: Measurements $\boldsymbol{y} \in \mathbb{C}^M$, matrix $\boldsymbol{A} \in \mathbb{R}^{M\times N}$ with $\|\boldsymbol{A}\|_F^2 = M$, pdfs $p_{\mathsf{c}|\mathsf{r}}(\cdot|\cdot)$ and $p_{\mathsf{z}|\mathsf{y},\mathsf{p}}(\cdot|\cdot,\cdot;\boldsymbol{\alpha},\boldsymbol{\tau})$ from (20) and (22), initial $\widehat{\boldsymbol{C}}_0 \in \mathbb{R}^{N\times K}$ and $\boldsymbol{q}^{\mathsf{p}} = \boldsymbol{q}_0^{\mathsf{p}} \in \mathbb{R}_+^K$.

1: $\widehat{\boldsymbol{S}} \leftarrow \boldsymbol{0}$, $\widehat{\boldsymbol{C}} \leftarrow \widehat{\boldsymbol{C}}_0$.
2: **repeat**
3: $\quad\widehat{\boldsymbol{P}} \leftarrow \boldsymbol{A}\widehat{\boldsymbol{C}} - \widehat{\boldsymbol{S}}\,\mathrm{Diag}(\boldsymbol{q}^{\mathsf{p}})$
4: $\quad\boldsymbol{q}_m^{\mathsf{z}} \leftarrow \mathrm{diag}\left(\mathrm{Cov}\left\{\boldsymbol{z}_m \middle| y_m, \widehat{\boldsymbol{p}}_m; \mathrm{Diag}(\boldsymbol{q}^{\mathsf{p}}), \boldsymbol{\alpha}, \boldsymbol{\tau}\right\}\right),\ m = 1...M$
5: $\quad\widehat{\boldsymbol{z}}_m \leftarrow \mathbb{E}\left\{\boldsymbol{z}_m \middle| y_m, \widehat{\boldsymbol{p}}_m; \mathrm{Diag}(\boldsymbol{q}^{\mathsf{p}}), \boldsymbol{\alpha}, \boldsymbol{\tau}\right\},\ m = 1...M$
6: $\quad\boldsymbol{q}^{\mathsf{s}} \leftarrow \boldsymbol{1} \oslash \boldsymbol{q}^{\mathsf{p}} - \left(\frac{1}{M}\sum_{m=1}^{M} \boldsymbol{q}_m^{\mathsf{z}}\right) \oslash (\boldsymbol{q}^{\mathsf{p}} \odot \boldsymbol{q}^{\mathsf{p}})$
7: $\quad\widehat{\boldsymbol{S}} \leftarrow (\widehat{\boldsymbol{Z}} - \widehat{\boldsymbol{P}})\,\mathrm{Diag}(\boldsymbol{q}^{\mathsf{p}})^{-1}$
8: $\quad\boldsymbol{q}^{\mathsf{r}} \leftarrow \frac{N}{M}\boldsymbol{1} \oslash \boldsymbol{q}^{\mathsf{s}}$
9: $\quad\widehat{\boldsymbol{R}} \leftarrow \widehat{\boldsymbol{C}} + \boldsymbol{A}^{\mathsf{T}}\widehat{\boldsymbol{S}}\,\mathrm{Diag}(\boldsymbol{q}^{\mathsf{r}})$
10: $\quad\boldsymbol{q}_n^{\mathsf{c}} \leftarrow \mathrm{diag}\left(\mathrm{Cov}\left\{\boldsymbol{c}_n \middle| \widehat{\boldsymbol{r}}_n; \mathrm{Diag}(\boldsymbol{q}^{\mathsf{r}})\right\}\right),\ n = 1...N$
11: $\quad\widehat{\boldsymbol{c}}_n \leftarrow \mathbb{E}\left\{\boldsymbol{c}_n \middle| \widehat{\boldsymbol{r}}_n; \mathrm{Diag}(\boldsymbol{q}^{\mathsf{r}})\right\},\ n = 1...N$
12: $\quad\boldsymbol{q}^{\mathsf{p}} \leftarrow \frac{1}{N}\sum_{n=1}^{N} \boldsymbol{q}_n^{\mathsf{c}}$
13: **until** convergence
14: **return** $\widehat{\boldsymbol{C}}$

Similarly, lines 10-11 compute the posterior mean and covariance of $\boldsymbol{c}_n$ under a prior p_{c} of the form (19) and "pseudo" measurements $\widehat{\boldsymbol{r}}_n$ that follow the statistical model

$$\widehat{\boldsymbol{r}}_n = \boldsymbol{c}_n + \boldsymbol{v}_n, \quad \boldsymbol{v}_n \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{Q}^{\mathsf{r}}), \tag{21}$$

where $\widehat{\boldsymbol{r}}_n$ and $\boldsymbol{Q}^{\mathsf{r}} = \mathrm{Diag}(\boldsymbol{q}^{\mathsf{r}})$ are updated at each SHyGAMP iteration. Thus, the pdf used for the covariance and expectation in lines 10-11 is

$$p_{\mathsf{c}|\mathsf{r}}(\boldsymbol{c}_n|\widehat{\boldsymbol{r}}_n; \boldsymbol{Q}^{\mathsf{r}}) = \frac{p_{\mathsf{c}}(\boldsymbol{c}_n)\mathcal{N}(\boldsymbol{c}_n; \widehat{\boldsymbol{r}}_n, \boldsymbol{Q}^{\mathsf{r}})}{\int p_{\mathsf{c}}(\boldsymbol{c}'_n)\mathcal{N}(\boldsymbol{c}'_n; \widehat{\boldsymbol{r}}_n, \boldsymbol{Q}^{\mathsf{r}})\,\mathrm{d}\boldsymbol{c}'_n}. \tag{22}$$

As the SHyGAMP iterations progress, the output $[\widehat{\boldsymbol{c}}_1, \dots, \widehat{\boldsymbol{c}}_N]^{\mathsf{T}}$ of line 11 converges to an approximation of the MMSE estimate $\mathbb{E}\{\boldsymbol{C}|\boldsymbol{y}\}$, and the output $[\widehat{\boldsymbol{z}}_1, \dots, \widehat{\boldsymbol{z}}_M]^{\mathsf{T}}$ of line 5 converges to an approximation of the MMSE estimate $\mathbb{E}\{\boldsymbol{Z}|\boldsymbol{y}\}$. Essentially, the SHyGAMP algorithm breaks an inference problem of dimension NK into $M+N$ inference problems of dimension K, each involving an independent-Gaussian pseudo-prior or pseudo-likelihood, evaluated iteratively. The computational complexity of SHyGAMP is $O(MNK)$.

D. From SHyGAMP to CL-AMP

The SHyGAMP algorithm can be applied to many different problems via appropriate choice of $p_{\mathsf{y}|\mathsf{z}}$ and p_{c}. To apply SHyGAMP to sketched clustering, we choose $p_{\mathsf{y}|\mathsf{z}}$ and p_{c} as described in Section II-A. As we will see, the main challenge is evaluating lines 4-5 of Algorithm 1 for the $p_{\mathsf{y}|\mathsf{z}}$ in (18).

1) Inference of $\boldsymbol{z}_m$: For lines 4-5 of Algorithm 1, we would like to compute the mean and variance

$$\widehat{z}_{mk} = \frac{\int_{\mathbb{R}^K} z_{mk} p_{\mathsf{y}|\mathsf{z}}(y_m|\boldsymbol{z}_m)\mathcal{N}\big(\boldsymbol{z}_m; \widehat{\boldsymbol{p}}_m, \boldsymbol{Q}^{\mathsf{p}}\big)\,\mathrm{d}\boldsymbol{z}_m}{C_m} \tag{23}$$

$$q_{mk}^{\mathsf{z}} = \frac{\int_{\mathbb{R}^K} (z_{mk} - \widehat{z}_{mk})^2 p_{\mathsf{y}|\mathsf{z}}(y_m|\boldsymbol{z}_m)\mathcal{N}\big(\boldsymbol{z}_m; \widehat{\boldsymbol{p}}_m, \boldsymbol{Q}^{\mathsf{p}}\big)\,\mathrm{d}\boldsymbol{z}_m}{C_m}, \tag{24}$$

where q_{mk}^{z} is the kth element of $\boldsymbol{q}_m^{\mathsf{z}}$ and

$$C_m = \int_{\mathbb{R}^K} p_{\mathsf{y}|\mathsf{z}}(y_m|\boldsymbol{z}_m)\mathcal{N}\big(\boldsymbol{z}_m; \widehat{\boldsymbol{p}}_m, \boldsymbol{Q}^{\mathsf{p}}\big)\,\mathrm{d}\boldsymbol{z}_m. \tag{25}$$

However, due to the form of $p_{\mathsf{y}|\mathsf{z}}$ in (18), we are not able to find closed-form expressions for $\widehat{z}_{mk}$ or q^{z}_{mk}. Thus, we propose to approximate $\widehat{z}_{mk}$ and q^{z}_{mk} as follows. The main idea behind our approximation is to write (17) as

$$y_m = \alpha_k \exp(-g_m^2\tau_k/2)\exp\left(\mathrm{j}g_m z_{mk}\right) + \sum_{l\neq k}\alpha_l \exp(-g_m^2\tau_l/2)\exp\left(\mathrm{j}g_m(z_{ml})\right) \tag{26}$$

and treat the sum over l as complex Gaussian. For the remainder of this section, we suppress the subscripts "m" and "$\mathsf{y}|\mathsf{z}$" to simplify the notation.

We begin by writing (26) as

$$y = \underbrace{\alpha_k \exp(-g^2\tau_k/2)}_{\triangleq\,\beta_k}\exp\Big(\mathrm{j}\underbrace{g(z_k+n_k)}_{\triangleq\,\theta_k}\Big) + \sum_{l\neq k}\underbrace{\alpha_l \exp(-g^2\tau_l/2)}_{=\beta_l}\underbrace{\exp\left(\mathrm{j}g(z_l+n_l)\right)}_{\triangleq\,v_l}, \tag{27}$$

where we introduced i.i.d. $n_k \sim \mathcal{N}(0,q^{\mathsf{n}})$ to facilitate the derivation in the sequel. Eventually we will take $q^{\mathsf{n}}\to 0$, in which case (27) exactly matches (26).

Next we derive an expression for the marginal posterior $p(z_k|y)$ under the pseudo-prior $z_k \sim \mathcal{N}(\widehat{p}_k, q^{\mathsf{p}}_k)\ \forall k$. First,

$$p(z_k|y) = \int_{\mathbb{R}^K} p(\boldsymbol{z},\theta_k|y)\,\mathrm{d}\theta_k\,\mathrm{d}\boldsymbol{z}_{\setminus k} \tag{28}$$

$$= \frac{1}{p(y)}\int_{\mathbb{R}^K} p(y|\boldsymbol{z},\theta_k)p(\theta_k|\boldsymbol{z})p(\boldsymbol{z})\,\mathrm{d}\theta_k\,\mathrm{d}\boldsymbol{z}_{\setminus k} \tag{29}$$

$$= \frac{1}{p(y)}\int_{\mathbb{R}^K} p(y|\boldsymbol{z}_{\setminus k},\theta_k)\mathcal{N}(\theta_k; gz_k, g^2q^{\mathsf{n}}) \times \prod_{l=1}^{K}\mathcal{N}(z_l;\widehat{p}_l,q^{\mathsf{p}}_l)\,\mathrm{d}\theta_k\,\mathrm{d}\boldsymbol{z}_{\setminus k}, \tag{30}$$

where $\boldsymbol{z}_{\setminus k} \triangleq [z_1,\ldots,z_{k-1},z_{k+1},\ldots,z_K]^{\mathsf{T}}$. A change-of-variables from z_l to $\widetilde{z}_l \triangleq z_l - \widehat{p}_l$ for all $l\neq k$ gives

$$p(z_k|y) = \frac{\mathcal{N}(z_k;\widehat{p}_k,q^{\mathsf{p}}_k)}{p(y)}\int_{\mathbb{R}}\mathcal{N}(\theta_k;gz_k,g^2q^{\mathsf{n}}) \times\left[\int_{\mathbb{R}^{K-1}} p(y|\widetilde{\boldsymbol{z}}_{\setminus k},\theta_k)\prod_{l\neq k}\mathcal{N}(\widetilde{z}_l;0,q^{\mathsf{p}}_l)\,\mathrm{d}\widetilde{\boldsymbol{z}}_{\setminus k}\right]\mathrm{d}\theta_k, \tag{31}$$

where $p(y|\widetilde{\boldsymbol{z}}_{\setminus k},\theta_k)$ is associated with the generative model

$$y = \beta_k\exp(\mathrm{j}\theta_k) + \sum_{l\neq k}\beta_l\exp\left(\mathrm{j}g(\widehat{p}_l+\widetilde{z}_l+n_l)\right) \tag{32}$$

with i.i.d. $n_l\sim\mathcal{N}(0,q^{\mathsf{n}})$. Now, because $\widetilde{z}_l$ and n_l are (apriori) mutually independent zero-mean Gaussian variables, we can work directly with the sum $\widetilde{n}_l \triangleq \widetilde{z}_l + n_l \sim \mathcal{N}(0,q^{\mathsf{p}}_l+q^{\mathsf{n}})$ and thus bypass the inner integral in (31). This allows us to write

$$p(z_k|y) = \frac{\mathcal{N}(z_k;\widehat{p}_k,q^{\mathsf{p}}_k)}{p(y)}\int_{\mathbb{R}}\mathcal{N}(\theta_k;gz_k,g^2q^{\mathsf{n}})p(y|\theta_k)\,\mathrm{d}\theta_k, \tag{33}$$

where $p(y|\theta_k)$ is associated with the generative model

$$y = \beta_k\exp(\mathrm{j}\theta_k) + \sum_{l\neq k}\beta_l\underbrace{\exp(\mathrm{j}g(\widehat{p}_l+\widetilde{n}_l))}_{=\,v_l} \tag{34}$$

with i.i.d. $\widetilde{n}_l\sim\mathcal{N}(0,q^{\mathsf{p}}_l+q^{\mathsf{n}})$. Recalling that $y\in\mathbb{C}$, it will sometimes be useful to write (34) as

$$\begin{bmatrix}\mathrm{Re}\{y\}\\ \mathrm{Im}\{y\}\end{bmatrix} \sim \mathcal{N}\Bigg(\beta_k\begin{bmatrix}\cos(\theta_k)\\ \sin(\theta_k)\end{bmatrix} + \sum_{l\neq k}\beta_l\,\mathbb{E}\left\{\begin{bmatrix}\mathrm{Re}\{v_l\}\\ \mathrm{Im}\{v_l\}\end{bmatrix}\right\}, \sum_{l\neq k}\beta_l^2\,\mathrm{Cov}\left\{\begin{bmatrix}\mathrm{Re}\{v_l\}\\ \mathrm{Im}\{v_l\}\end{bmatrix}\right\}\Bigg). \tag{35}$$

To compute the posterior mean of z_k, (33) implies

$$\widehat{z}_k \triangleq \mathbb{E}\{z_k|y\} = \int_{\mathbb{R}} z_k\,p(z_k|y)\,\mathrm{d}z_k \tag{36}$$

$$= \frac{1}{p(y)}\int_{\mathbb{R}}\left[\int_{\mathbb{R}} z_k\,\mathcal{N}(gz_k;\theta_k,g^2q^{\mathsf{n}})\mathcal{N}(z_k;\widehat{p}_k,q^{\mathsf{p}}_k)\,\mathrm{d}z_k\right] \times p(y|\theta_k)\,\mathrm{d}\theta_k \tag{37}$$

$$= \int_{\mathbb{R}}\left[\int_{\mathbb{R}} z_k\,\mathcal{N}\left(z_k;\frac{\frac{\theta_k/g}{q^{\mathsf{n}}}+\frac{\widehat{p}_k}{q^{\mathsf{p}}_k}}{\frac{1}{q^{\mathsf{n}}}+\frac{1}{q^{\mathsf{p}}_k}},\frac{1}{\frac{1}{q^{\mathsf{n}}}+\frac{1}{q^{\mathsf{p}}_k}}\right)\mathrm{d}z_k\right] \times \underbrace{\frac{\mathcal{N}\Big(\theta_k;g\widehat{p}_k,g^2(q^{\mathsf{n}}+q^{\mathsf{p}}_k)\Big)\,p(y|\theta_k)}{p(y)}}_{=\,p(\theta_k|y)}\mathrm{d}\theta_k \tag{38}$$

$$= \int_{\mathbb{R}}\frac{\frac{\theta_k/g}{q^{\mathsf{n}}}+\frac{\widehat{p}_k}{q^{\mathsf{p}}_k}}{\frac{1}{q^{\mathsf{n}}}+\frac{1}{q^{\mathsf{p}}_k}}\,p(\theta_k|y)\,\mathrm{d}\theta_k \tag{39}$$

$$= \frac{\widehat{p}_k}{q^{\mathsf{p}}_k/q^{\mathsf{n}}+1} + \frac{\widehat{\theta}_k/g}{1+q^{\mathsf{n}}/q^{\mathsf{p}}_k} \quad \text{for } \widehat{\theta}_k \triangleq \int_{\mathbb{R}}\theta_k\,p(\theta_k|y)\,\mathrm{d}\theta_k, \tag{40}$$

where the Gaussian pdf multiplication rule[1] was used in (38) and where $\widehat{\theta}_k$ denotes the posterior mean of θ_k.

For the posterior variance of z_k, a similar approach gives

$$q^{\mathsf{z}}_k \triangleq \mathrm{var}\{z_k|y\} = \int_{\mathbb{R}}\left(z_k-\widehat{z}_k\right)^2 p(z_k|y)\,\mathrm{d}z_k \tag{41}$$

$$= \frac{1}{p(y)}\int_{\mathbb{R}}\Bigg[\int_{\mathbb{R}}(z_k-\widehat{z}_k)^2\,\mathcal{N}(gz_k;\theta_k,g^2q^{\mathsf{n}}) \times\mathcal{N}(z_k;\widehat{p}_k,q^{\mathsf{p}}_k)\,\mathrm{d}z_k\Bigg]p(y|\theta_k)\,\mathrm{d}\theta_k \tag{42}$$

$$= \int_{\mathbb{R}}\left[\int_{\mathbb{R}}(z_k-\widehat{z}_k)^2\,\mathcal{N}\left(z_k;\frac{\frac{\theta_k/g}{q^{\mathsf{n}}}+\frac{\widehat{p}_k}{q^{\mathsf{p}}_k}}{\frac{1}{q^{\mathsf{n}}}+\frac{1}{q^{\mathsf{p}}_k}},\frac{1}{\frac{1}{q^{\mathsf{n}}}+\frac{1}{q^{\mathsf{p}}_k}}\right)\mathrm{d}z_k\right] \times p(\theta_k|y)\,\mathrm{d}\theta_k. \tag{43}$$

Using a change-of-variables from z_k to $\widetilde{z}_k \triangleq z_k - \widehat{z}_k$, we get

$$q^{\mathsf{z}}_k = \int_{\mathbb{R}}\left[\int_{\mathbb{R}}\widetilde{z}_k^2\,\mathcal{N}\left(\widetilde{z}_k;\frac{\frac{\theta_k/g}{q^{\mathsf{n}}}-\frac{\widehat{\theta}_k/g}{q^{\mathsf{n}}}}{\frac{1}{q^{\mathsf{n}}}+\frac{1}{q^{\mathsf{p}}_k}},\frac{1}{\frac{1}{q^{\mathsf{n}}}+\frac{1}{q^{\mathsf{p}}_k}}\right)\mathrm{d}\widetilde{z}_k\right] \times p(\theta_k|y)\,\mathrm{d}\theta_k \tag{44}$$

[1] $\mathcal{N}(\boldsymbol{x};\boldsymbol{a},\boldsymbol{A})\mathcal{N}(\boldsymbol{x};\boldsymbol{b},\boldsymbol{B}) = \mathcal{N}(\boldsymbol{0};\boldsymbol{a}-\boldsymbol{b},\boldsymbol{A}+\boldsymbol{B})\mathcal{N}\big(\boldsymbol{x};(\boldsymbol{A}^{-1}+\boldsymbol{B}^{-1})^{-1}(\boldsymbol{A}^{-1}\boldsymbol{a}+\boldsymbol{B}^{-1}\boldsymbol{b}),(\boldsymbol{A}^{-1}+\boldsymbol{B}^{-1})^{-1}\big)$.

$$= \int_{\mathbb{R}} \left[\left(\frac{(\theta_k - \widehat{\theta}_k)/g}{1 + q^{\mathsf{n}}/q_k^{\mathsf{p}}} \right)^2 + \frac{q^{\mathsf{n}}}{1 + q^{\mathsf{n}}/q_k^{\mathsf{p}}} \right] p(\theta_k|y)\, \mathrm{d}\theta_k \quad (45)$$

$$= \frac{q^{\mathsf{n}}}{1+q^{\mathsf{n}}/q_k^{\mathsf{p}}} + \frac{1}{g^2} \left(\frac{1}{1+q^{\mathsf{n}}/q_k^{\mathsf{p}}} \right)^2 \underbrace{\int_{\mathbb{R}} (\theta_k - \widehat{\theta}_k)^2\, p(\theta_k|y)\, \mathrm{d}\theta_k}_{\triangleq\, q_k^{\theta} = \mathrm{var}\{\theta_k|y\}}. \quad (46)$$

The computation of $\widehat{z}_k$ and q_k^{z} is still complicated by the form of the posterior $p(\theta_k|y)$ implied by (34). To circumvent this problem, we propose to apply a Gaussian approximation to the sum in (34). Because $\{\widetilde{n}_l\}_{\forall l \neq k}$ are mutually independent, the mean and covariance of the sum in (34) are simply the sum of the means and covariances (respectively) of the $K-1$ terms making up the sum. Recalling (35), this implies that

$$p\left(\begin{bmatrix} \mathrm{Re}\{y\} \\ \mathrm{Im}\{y\} \end{bmatrix} \middle| \theta_k \right) \approx \mathcal{N}\left(\begin{bmatrix} \mathrm{Re}\{y\} \\ \mathrm{Im}\{y\} \end{bmatrix}; \beta_k \begin{bmatrix} \cos(\theta_k) \\ \sin(\theta_k) \end{bmatrix} + \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k \right) \quad (47)$$

with

$$\boldsymbol{\mu}_k = \sum_{l \neq k} \alpha_l e^{-g^2(\tau_k + q_k^{\mathsf{p}})/2} \begin{bmatrix} \cos(g\widehat{p}_l) \\ \sin(g\widehat{p}_l) \end{bmatrix} \quad (48)$$

$$\boldsymbol{\Sigma}_k = \frac{1}{2} \sum_{l \neq k} \beta_l^2 \big(1 - e^{-g^2 q_l^{\mathsf{p}}}\big) \times \left(\boldsymbol{I} - e^{-g^2 q_l^{\mathsf{p}}} \begin{bmatrix} \cos(2g\widehat{p}_l) & \sin(2g\widehat{p}_l) \\ \sin(2g\widehat{p}_l) & -\cos(2g\widehat{p}_l) \end{bmatrix} \right). \quad (49)$$

We note that (48) and (49) were obtained using

$$\mathbb{E}\left\{ \mathrm{Re}\{v_l\} \right\} = \exp\big(-g^2 q_l^{\mathsf{p}}/2 \big) \cos(g\widehat{p}_l) \quad (50)$$

$$\mathbb{E}\left\{ \mathrm{Im}\{v_l\} \right\} = \exp\big(-g^2 q_l^{\mathsf{p}}/2 \big) \sin(g\widehat{p}_l) \quad (51)$$

$$2\,\mathbb{E}\left\{ \mathrm{Re}\{v_l\}^2 \right\} = 1 + \exp\big(-g^2 q_l^{\mathsf{p}} \big) \cos(2g\widehat{p}_l) \quad (52)$$

$$2\,\mathbb{E}\left\{ \mathrm{Im}\{v_l\}^2 \right\} = 1 - \exp\big(-g^2 q_l^{\mathsf{p}} \big) \cos(2g\widehat{p}_l) \quad (53)$$

$$2\,\mathbb{E}\left\{ \mathrm{Re}\{v_l\}\,\mathrm{Im}\{v_l\} \right\} = \exp\big(-g^2 q_l^{\mathsf{p}} \big) \sin(2g\widehat{p}_l), \quad (54)$$

which use the fact that, after letting $q^{\mathsf{n}} \to 0$,

$$\mathbb{E}\{v_l\} = \int_{\mathbb{R}} \mathcal{N}(z_l; \widehat{p}_l, q_l^{\mathsf{p}}) \exp(\mathrm{j}g z_l)\, \mathrm{d}z_l \quad (55)$$

$$= \exp\big(\mathrm{j}g\widehat{p}_l - g^2 q_l^{\mathsf{p}}/2 \big). \quad (56)$$

Rewriting (47) as

$$p\left(\beta_k^{-1} \begin{bmatrix} \mathrm{Re}\{y\} \\ \mathrm{Im}\{y\} \end{bmatrix} \middle| \theta_k \right) \quad (57)$$

$$\approx \mathcal{N}\left(\begin{bmatrix} \cos(\theta_k) \\ \sin(\theta_k) \end{bmatrix}; \beta_k^{-1} \begin{bmatrix} \mathrm{Re}\{y\} \\ \mathrm{Im}\{y\} \end{bmatrix} - \beta_k^{-1} \boldsymbol{\mu}_k, \beta_k^{-2} \boldsymbol{\Sigma}_k \right),$$

the right side of (57) can be recognized as being proportional to the generalized von Mises (GvM) density over $\theta_k \in [0, 2\pi)$ from [19]. Under this GvM approximation, we have [19] that

$$p(y|\theta_k) \propto \exp\big(\kappa_k \cos(\theta_k - \zeta_k) + \overline{\kappa}_k \cos[2(\theta_k - \overline{\zeta}_k)] \big) \quad (58)$$

for parameters $\kappa_k, \overline{\kappa}_k > 0$ and $\zeta_k, \overline{\zeta}_k \in [0, 2\pi)$ defined from $\beta_k^{-1} y$, $\beta_k^{-1} \boldsymbol{\mu}_k$, and $\beta_k^{-2} \boldsymbol{\Sigma}_k$. In particular,

$$\kappa_k \cos(\zeta_k) = -\frac{1}{1-\rho_k^2} \left(\frac{\rho_k \overline{\nu}_k}{\sigma_k \overline{\sigma}_k} - \frac{\nu_k}{\sigma_k^2} \right) \quad (59)$$

$$\kappa_k \sin(\zeta_k) = -\frac{1}{1-\rho_k^2} \left(\frac{\rho_k \nu_k}{\sigma_k \overline{\sigma}_k} - \frac{\overline{\nu}_k}{\overline{\sigma}_k^2} \right) \quad (60)$$

$$\overline{\kappa}_k \cos(2\overline{\zeta}_k) = -\frac{1}{4(1-\rho_k^2)} \left(\frac{1}{\sigma_k^2} - \frac{1}{\overline{\sigma}_k^2} \right) \quad (61)$$

$$\overline{\kappa}_k \sin(2\overline{\zeta}_k) = \frac{\rho_k}{2(1-\rho_k^2)\sigma_k \overline{\sigma}_k}, \quad (62)$$

where

$$\begin{bmatrix} \nu_k \\ \overline{\nu}_k \end{bmatrix} \triangleq \beta_k^{-1} \left(\begin{bmatrix} \mathrm{Re}\{y\} \\ \mathrm{Im}\{y\} \end{bmatrix} - \boldsymbol{\mu}_k \right) \quad (63)$$

$$\begin{bmatrix} \sigma_k^2 & \rho_k \sigma_k \overline{\sigma}_k \\ \rho_k \sigma_k \overline{\sigma}_k & \overline{\sigma}_k^2 \end{bmatrix} \triangleq \beta_k^{-2} \boldsymbol{\Sigma}_k. \quad (64)$$

From (58) and the SHyGAMP pseudo-prior $z_k \sim \mathcal{N}(\widehat{p}_k, q_k^{\mathsf{p}})$, we see that the posterior on θ_k takes the form

$$p(\theta_k|y) \propto \mathcal{N}\big(\theta_k; g\widehat{p}_k, g^2 q_k^{\mathsf{p}}\big)\, p(y|\theta_k) \quad (65)$$

$$\propto \exp\left[\kappa_k \cos(\theta_k - \zeta_k) + \overline{\kappa}_k \cos[2(\theta_k - \overline{\zeta}_k)] - \frac{(\theta_k - g\widehat{p}_k)^2}{2g^2 q_k^{\mathsf{p}}} \right]. \quad (66)$$

We now face the task of computing $\widehat{\theta}_k = \mathbb{E}\{\theta_k|y\}$ and $q_k^{\theta} = \mathrm{var}\{\theta_k|y\}$ under (66). Since these quantities do not appear to be computable in closed form, we settle for an approximation, such as that based on the Laplace approximation [20] or numerical integration. For the Laplace approximation, we would first compute $\widehat{\theta}_{k,\mathsf{MAP}} \triangleq \arg\max_{\theta_k} \ln p(\theta_k|y)$ and then approximate $\widehat{\theta}_k \approx \widehat{\theta}_{k,\mathsf{MAP}}$ and $q_k^{\theta} \approx -\frac{\mathrm{d}^2}{\mathrm{d}\theta_k{}^2} \ln p(\theta_k|y)\big|_{\theta_k = \widehat{\theta}_{k,\mathsf{MAP}}}$. However, since computing $\arg\max_{\theta_k} \ln p(\theta_k|y)$ is complicated due to the presence of multiple local maxima, we instead use numerical integration. For this, we suggest a grid of $N_{\text{pts}} N_{\text{per}} + 1$ uniformly-spaced points centered at $g\widehat{p}_k$ with width $2\pi N_{\text{per}}$, where $N_{\text{per}} = \left\lceil \frac{N_{\text{std}}}{\pi} \sqrt{g^2 q_k^{\mathsf{p}}} \right\rceil$. This choice of grid ensures that the sampling points cover at least N_{std} standard deviations of the prior on θ_k. We used $N_{\text{std}} = 4$ and $N_{\text{pts}} = 7$ in the numerical experiments in Section III.

Finally, after approximating $\widehat{\theta}_k$ and q_k^{θ} via numerical integration, we set $\widehat{z}_k = \widehat{\theta}_k / g$ and $q_k^{\mathsf{z}} = q_k^{\theta}/g^2$.

2) Inference of $\boldsymbol{c}_n$: Recall that lines 10-11 of Algorithm 1 support an arbitrary prior $p_{\mathbf{c}}$ on $\boldsymbol{c}_n$. For the experiments in Section III, we used the trivial non-informative prior $p_{\mathbf{c}}(\boldsymbol{c}_n) \propto 1$, after which lines 10-11 reduce to

$$\boldsymbol{q}_n^{\mathbf{c}} = \boldsymbol{q}^{\mathbf{r}}\ \forall n \quad \text{and} \quad \widehat{\boldsymbol{c}}_n = \widehat{\boldsymbol{r}}_n\ \forall n. \quad (67)$$

E. Initialization

We recommend initializing CL-AMP with $\widehat{\boldsymbol{C}} = \widehat{\boldsymbol{C}}_0$ and $\boldsymbol{q}^{\mathbf{p}} = \boldsymbol{q}_0^{\mathbf{p}}$, where $\widehat{\boldsymbol{C}}_0$ is drawn i.i.d. $\mathcal{N}(0, \sigma^2)$ and where $\boldsymbol{q}_0^{\mathbf{p}} = \sigma^2 \mathbf{1}$, with σ^2 from (80) (as described in Section II-H).

In some cases, running CL-AMP from $R > 1$ different random initializations can help avoid to spurious solutions. Here, CL-AMP is run from a different random initialization $\widehat{\boldsymbol{C}}_{0,r}$, for $r = 1, \ldots, R$, and then the quality of the recovered solution $\widehat{\boldsymbol{C}}_r$ is evaluated by constructing the "estimated sketch" $\widehat{\boldsymbol{y}}_r$ via

$$\widehat{y}_{mr} = \sum_{k=1}^{K} \alpha_k \exp(-g_m^2 \tau_k) \exp(\mathrm{j}g_m \boldsymbol{a}_m^{\mathsf{T}} \widehat{\boldsymbol{c}}_{kr}) \quad (68)$$

recalling (9) and (17), and then measuring its distance to the true sketch $\boldsymbol{y}$. The initialization index is then selected as

$$r_* = \arg\min_r \|\boldsymbol{y} - \widehat{\boldsymbol{y}}_r\|, \tag{69}$$

and the centroids saved as $\widehat{\boldsymbol{C}} = \widehat{\boldsymbol{C}}_{r_*}$. In Section III, we used $R = 2$ for all experiments.

F. Hyperparameter Tuning

The likelihood model $p_{\mathsf{y}|\mathbf{z}}$ in (18) depends on the unknown hyperparameters $\boldsymbol{\alpha}$ and $\boldsymbol{\tau}$. We propose to estimate these hyperparameters using a combination of *expectation maximization* (EM) and SHyGAMP, as suggested in [14] and detailed—for the simpler case of GAMP—in [21]. The idea is to run SHyGAMP using an estimate of $\boldsymbol{\alpha}$ and $\boldsymbol{\tau}$, update $\boldsymbol{\alpha}$ and $\boldsymbol{\tau}$ from the SHyGAMP outputs, and repeat until convergence. For the first estimate, we suggest to use $\alpha_k = \frac{1}{K}$ and $\tau_k = 0 \;\forall\; k$.

Extrapolating [21, eq. (23)] to the SHyGAMP case, the EM update of $(\boldsymbol{\alpha}, \boldsymbol{\tau})$ takes the form

$$(\widehat{\boldsymbol{\alpha}}, \widehat{\boldsymbol{\tau}}) = \underset{\boldsymbol{\alpha}\geq\mathbf{0}, \boldsymbol{\alpha}^{\mathsf{T}}\mathbf{1}=1, \boldsymbol{\tau}>\mathbf{0}}{\arg\max} \sum_{m=1}^{M} \int_{\mathbb{R}^K} \mathcal{N}(\boldsymbol{z}_m; \widehat{\boldsymbol{z}}_m, \boldsymbol{Q}_m^{\mathbf{z}}) \times \ln p_{\mathsf{y}|\mathbf{z}}(y_m|\boldsymbol{z}_m; \boldsymbol{\alpha}, \boldsymbol{\tau}) \,\mathrm{d}\boldsymbol{z}_m, \tag{70}$$

where $\widehat{\boldsymbol{z}}_m$ and $\boldsymbol{Q}_m^{\mathbf{z}} = \mathrm{Diag}\{\boldsymbol{q}_m^{\mathbf{z}}\}$ are obtained by running SHyGAMP to convergence under $(\boldsymbol{\alpha}, \boldsymbol{\tau})$. To proceed, we model the Dirac delta in (18) using a circular Gaussian pdf with vanishingly small variance $\epsilon > 0$, in which case

$$\ln p_{\mathsf{y}|\mathbf{z}}(y_m|\boldsymbol{z}_m; \boldsymbol{\alpha}, \boldsymbol{\tau}) = -\frac{1}{\epsilon}\left| y_m - \sum_{k=1}^{K} \alpha_k \exp\left(\mathrm{j}g_m z_{mk} - \frac{g_m^2 \tau_k}{2}\right)\right|^2 + \text{const.} \tag{71}$$

Plugging (73) back into (70), we see that the constant and the $1/\epsilon$-scaling play no role in the optimization, and so we can discard them to obtain

$$(\widehat{\boldsymbol{\alpha}}, \widehat{\boldsymbol{\tau}}) = \underset{\boldsymbol{\alpha}\geq\mathbf{0}, \boldsymbol{\alpha}^{\mathsf{T}}\mathbf{1}=1, \boldsymbol{\tau}>\mathbf{0}}{\arg\min} \sum_{m=1}^{M} \int_{\mathbb{R}^K} \mathcal{N}(\boldsymbol{z}_m; \widehat{\boldsymbol{z}}_m, \boldsymbol{Q}_m^{\mathbf{z}}) \times \left| y_m - \sum_{k=1}^{K} \alpha_k \exp\left(\mathrm{j}g_m z_{mk} - \frac{g_m^2 \tau_k}{2}\right)\right|^2 \mathrm{d}\boldsymbol{z}_m. \tag{72}$$

A closed-form solution to the optimization problem in (72) seems out of reach. Also, the optimization objective is convex in $\boldsymbol{\alpha}$ for fixed $\boldsymbol{\tau}$, and convex in $\boldsymbol{\tau}$ for fixed $\boldsymbol{\alpha}$, but not jointly convex in $[\boldsymbol{\alpha}^{\mathsf{T}}, \boldsymbol{\tau}^{\mathsf{T}}]$. Although the optimization problem (72) is difficult to solve, the solutions obtained by gradient projection (GP) [22] seem to work well in practice. Also, GP is made practical by closed-form gradient expressions. In particular, let

$$q_{mk} \triangleq \exp\left(-\frac{g_m^2 \tau_k}{2}\right) \tag{73}$$

$$\rho_{mk} \triangleq \exp\left(\mathrm{j}g_m \widehat{z}_{mk} - \frac{q_{mk}^{\mathbf{z}} g_m^2}{2}\right), \tag{74}$$

and recall that $v_{mk} = \exp(\mathrm{j}g_m z_{mk})$ from (27). Then the mth term of the sum in the objective in (72) becomes

$$\begin{aligned}\int_{\mathbb{R}^K} \mathcal{N}(\boldsymbol{z}_m; \widehat{\boldsymbol{z}}_m, \boldsymbol{Q}_m^{\mathbf{z}}) \left| y_m - \sum_{k=1}^{K} \alpha_k q_{mk} v_{mk}\right|^2 \mathrm{d}\boldsymbol{z}_m \\ = |y_m|^2 - 2\sum_{k=1}^{K} \alpha_k q_{mk} \operatorname{Re}\left\{y_m^* \rho_{mk}\right\} \\ + \sum_{k=1}^{K} \alpha_k q_{mk} \rho_{mk}^* \sum_{l\neq k}^{K} \alpha_l q_{ml} \rho_{ml} + \sum_{k=1}^{K} \alpha_k^2 q_{mk}^2,\end{aligned} \tag{75}$$

where we used the fact that $\int_{\mathbb{R}} \mathcal{N}(z_{mk}; \widehat{z}_{mk}, q_{mk}^{\mathbf{z}}) v_{mk} \,\mathrm{d}z_{mk} = \rho_{mk}$. After reapplying the sum over m, we get

$$\frac{\partial}{\partial \alpha_k} \sum_{m=1}^{M} \int_{\mathbb{R}^K} \mathcal{N}(\boldsymbol{z}_m; \widehat{\boldsymbol{z}}_m, \boldsymbol{Q}_m^{\mathbf{z}}) \left| y_m - \sum_{k=1}^{K} \alpha_k q_{mk} v_{mk}\right|^2 \mathrm{d}\boldsymbol{z}_m = -2\sum_{m=1}^{M} q_{mk} \gamma_{mk} \tag{76}$$

$$\frac{\partial}{\partial \tau_k} \sum_{m=1}^{M} \int_{\mathbb{R}^K} \mathcal{N}(\boldsymbol{z}_m; \widehat{\boldsymbol{z}}_m, \boldsymbol{Q}_m^{\mathbf{z}}) \left| y_m - \sum_{k=1}^{K} \alpha_k q_{mk} v_{mk}\right|^2 \mathrm{d}\boldsymbol{z}_m = \alpha_k \sum_{m=1}^{M} g_m^2 q_{mk} \gamma_{mk} \tag{77}$$

for

$$\gamma_{mk} \triangleq \operatorname{Re}\left\{y_m^* \rho_{mk}\right\} - \alpha_k q_{mk} - \sum_{l\neq k}^{K} \alpha_l q_{ml} \operatorname{Re}\left\{\rho_{mk}^* \rho_{ml}\right\}. \tag{78}$$

We found that complexity of hyperparameter tuning can be substantially reduced, without much loss in accuracy, by using only a subset of the terms in the sum in (72), as well as in the corresponding gradient expressions (76)-(77). For the experiments in Section III, we used a fixed random subset of $\min(M, 20K)$ terms.

G. Algorithm Summary

Algorithm 2 summarizes the CL-AMP algorithm with R random initializations and tuning of the hyperparameters $(\boldsymbol{\alpha}, \boldsymbol{\tau})$. Note that the random initializations $\{\widehat{\boldsymbol{C}}_{0,r}\}$ are used only for the first EM iteration, i.e., $i = 0$. Subsequent EM iterations (i.e., $i \geq 1$) are initialized using the output $\widehat{\boldsymbol{C}}_i$ of the previous EM iteration.

H. Frequency Generation

As proposed in [5], $\boldsymbol{a}_m$ were drawn uniformly on the unit sphere and $\{g_m\}$ were drawn i.i.d. from the distribution

$$p(g; \sigma^2) \propto 1_{[0,\infty)}(g) \sqrt{g^2\sigma^2 + \frac{g^4\sigma^4}{4}} \exp\Big(-\frac{1}{2}g^2\sigma^2\Big), \tag{79}$$

which has parameter σ^2. The authors in [5] suggest using $\sigma^2 = \frac{1}{NK}\sum_{k=1}^{K} \mathrm{tr}(\boldsymbol{\Phi}_k)$ and propose a method to estimate σ^2 from $\boldsymbol{y}$. However, our numerical experiments suggest that using

$$\sigma^2 = \mathbb{E}\{\|\boldsymbol{x}\|_2^2\}/N \approx \|\boldsymbol{X}\|_F^2/NT \tag{80}$$

provides significantly improved performance. Note that the right side of (80) can be computed in an online manner, or

Algorithm 2 CL-AMP with hyperparameter tuning and multiple random initializations

Require: Measurements $\boldsymbol{y} \in \mathbb{C}^M$, gains $\{g_m\}_{m=1}^M$, number of initializations $R \geq 1$, initializations $\{\widehat{\boldsymbol{C}}_{0,r}\}_{r=1}^R$, $\boldsymbol{q}_0^{\mathsf{p}}$, $\boldsymbol{\alpha}_0$, $\boldsymbol{\tau}_0$.

$i = 0$
repeat
  if $i = 0$ **then**
    for $r = 1 : R$ **do**
      Run CL-AMP with fixed $(\boldsymbol{\alpha}_0, \boldsymbol{\tau}_0)$ from initialization $(\widehat{\boldsymbol{C}}_{0,r}, \boldsymbol{q}_0^{\mathsf{p}})$, yielding output $\widehat{\boldsymbol{C}}_{1,r}$, $\widehat{\boldsymbol{Z}}_r$, and $\{\boldsymbol{q}_{mr}^{\mathsf{z}}\}_{m=1}^M$.
    end for
    Compute $\widehat{y}_{mr} \triangleq \sum_{k=1}^K \alpha_{0k} \exp(-g_m^2 \tau_{0k}) \exp(\mathrm{j} g_m \widehat{z}_{mkr})\ \forall mr$
    Find $r_* = \arg\min_r \|\boldsymbol{y} - \widehat{\boldsymbol{y}}_r\|$.
    Set $\widehat{\boldsymbol{C}}_1 = \widehat{\boldsymbol{C}}_{1,r_*}$, $\widehat{\boldsymbol{Z}} = \widehat{\boldsymbol{Z}}_{r_*}$ and $\{\boldsymbol{q}_m^{\mathsf{z}}\}_{m=1}^M = \{\boldsymbol{q}_{mr_*}^{\mathsf{z}}\}_{m=1}^M$.
  else
    Run CL-AMP with fixed $(\boldsymbol{\alpha}_i, \boldsymbol{\tau}_i)$ from initialization $(\widehat{\boldsymbol{C}}_i, \boldsymbol{q}_0^{\mathsf{p}})$, yielding output $\widehat{\boldsymbol{C}}_{i+1}$, $\widehat{\boldsymbol{Z}}$, and $\{\boldsymbol{q}_m^{\mathsf{z}}\}_{m=1}^M$.
  end if
  Compute $(\boldsymbol{\alpha}_{i+1}, \boldsymbol{\tau}_{i+1})$ via (72) using $\widehat{\boldsymbol{Z}}$ and $\{\boldsymbol{q}_m^{\mathsf{z}}\}_{m=1}^M$.
  $i \leftarrow i + 1$.
until convergence

approximated using a subset of the data. For the experiments in Section III, we computed σ^2 via the right side of (80) and used it in computing $\boldsymbol{q}_0^{\mathsf{c}} = \sigma^2 \mathbf{1}$.

III. Numerical Experiments

In this section, we present the results of several experiments used to test the performance of the CL-AMP, CL-OMPR, and k-means++ algorithms. For k-means++, we used the implementation provided by MATLAB and, for CL-OMPR, we downloaded the MATLAB implementation from [23]. CL-OMPR and CL-AMP used the same sketch $\boldsymbol{y}$, whose frequency vectors $\boldsymbol{W}$ were drawn using the method described in Section II-H, with the scaling parameter σ^2 set via (80). For CL-OMPR and CL-AMP, the reported runtimes include the time of computing the sketch, unless otherwise noted. All experiments were run on a Dell PowerEdge C6320 two-socket server with Intel Xeon E5-2680 v4 processors (14 cores, 2.40GHz) and 128GB RAM.

A. Experiments with Synthetic Data

1) Performance vs. sketch length M: In the first experiment, we test each algorithm's ability to minimize SSE on a set of training data, i.e., to solve the problem (1). In addition, we test how well the recovered centroids work in minimum-distance classification.

The experiment was conducted as follows. Fixing the number of classes at $K = 10$ and the data dimension at $N = 100$, 10 Monte Carlo trials were performed. In each trial, the true centroids were randomly drawn[2] as $\boldsymbol{c}_k \sim \mathcal{N}(\mathbf{0}_N, 1.5^2 K^{2/N} \boldsymbol{I}_N)$. Then, using these centroids, a training dataset $\{\boldsymbol{x}_t\}_{t=1}^T$ with $T = 10^7$ samples was drawn from the GMM (5) with weights $\alpha_k = 1/K$ and covariances $\boldsymbol{\Phi}_k = \boldsymbol{I}_N \forall k$. Additionally, a test dataset $\{\overline{\boldsymbol{x}}_t\}$ of 10^6 samples was independently generated.

For centroid recovery, k-means++ was invoked on the training dataset, and both CL-AMP and CL-OMPR were invoked after sketching the training data with M samples as in (2). Sketch lengths $M/KN \in \{1, 2, 3, 5, 10, 20\}$ were investigated. CL-AMP used two random initializations, i.e., $R = 2$ as defined in Algorithm 2.

For each algorithm, the SSE of its estimated centroids $\{\widehat{\boldsymbol{c}}_k\}_{k=1}^K$ was calculated using the training data $\{\boldsymbol{x}_t\}_{t=1}^T$ via (1). Additionally, the performance of the estimated centroids in minimum-distance classification was evaluated as follows. First, labels $\{j_k\}_{k=1}^K$ were assigned to the estimated centroids by solving the linear assignment problem [24] without replacement, given by

$$\underset{\{j_1,\ldots,j_K\}=\{1,\ldots,K\}}{\arg\min} \sum_{k=1}^K \|\boldsymbol{c}_k - \widehat{\boldsymbol{c}}_{j_k}\|_2^2. \tag{81}$$

Next, each test sample $\overline{\boldsymbol{x}}_t$ was classified using minimum-distance classification, producing the estimated label

$$\widehat{k}_t = \underset{k \in \{1,\ldots,K\}}{\arg\min} \|\overline{\boldsymbol{x}}_t - \widehat{\boldsymbol{c}}_{j_k}\|. \tag{82}$$

The classification error rate (CER) was then calculated as the proportion of estimated labels $\widehat{k}_t$ that do not equal the true label k_t from which the test sample $\overline{\boldsymbol{x}}_t$ was generated.[3]

Figures 1a, 1b, and 1c show the median SSE, CER, and runtime (including sketching), respectively, for CL-AMP and CL-OMPR versus M/KN. Also shown is the median SSE, CER, and runtime of k-means++, as a baseline, where k-means++ has no dependence on M. Because a low runtime is meaningless if the corresponding SSE is very high, the runtime was not shown for CL-AMP and CL-OMPR whenever its SSE was more than 1.5 times that of k-means++. The error bars show the standard deviation of the estimates.

Figure 1a shows that CL-AMP achieved a low SSE with smaller sketch size M than CL-OMPR. In particular, CL-AMP required $M \approx 2KN$ to yield a low SSE, while CL-OMPR required $M \approx 10KN$. Also, with sufficiently large M, the SSE achieved by CL-AMP and CL-OMPR was lower than that achieved by k-means++.

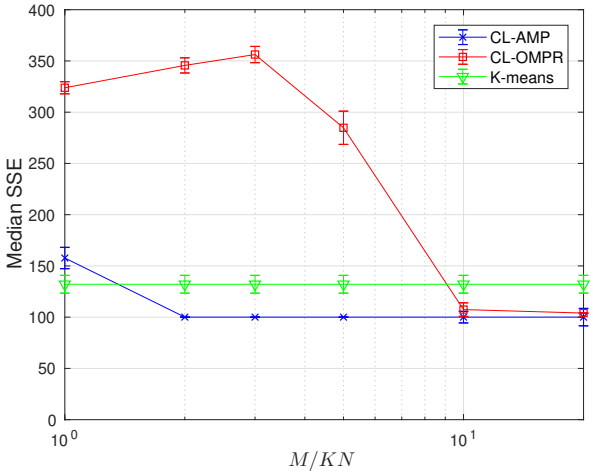
Figure 1b shows that CL-AMP achieved a low CER with sketch size $M \approx KN$, while CL-OMPR required $M \approx 10KN$. Also, with sufficiently large M, CL-AMP and CL-OMPR achieved near-zero CER, whereas k-means++ achieved an error rate of only ≈ 0.2.

Finally, Fig. 1c shows that, for $M/KN \in \{10, 20\}$, k-means++ ran slightly faster than CL-AMP, which ran slightly faster than CL-OMPR. However, for $M/KN \in \{1, 2, 3, 5\}$, CL-AMP ran significantly faster than k-means++. For $M/KN \in \{1, 2, 3, 5\}$, the runtime of CL-OMPR was not shown because it generated centroids of significantly worse SSE than those of k-means++.

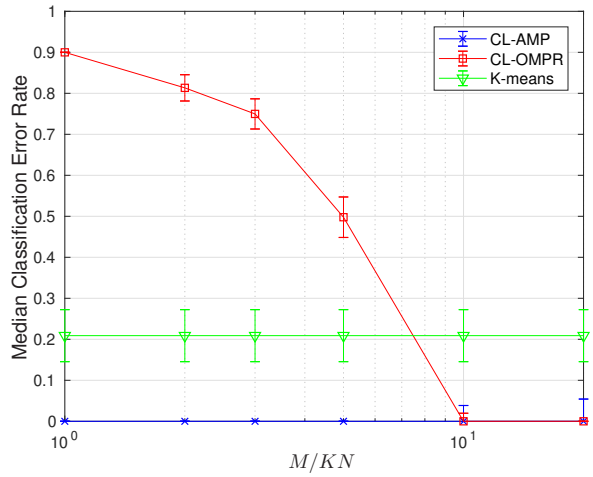
2) Performance vs. number of classes K: In a second experiment, we evaluated each algorithm's performance versus the number of classes $K \in \{5, 10, 15, 20, 25, 30, 40, 50\}$ and sketch sizes $M/KN \in \{2, 5, 10\}$ for fixed data dimension $N = 50$. The data was generated in exactly the same way as

[2]This data-generation model was chosen to match that from [5], and is intended to have a relatively constant Bayes error rate w.r.t. N and K.

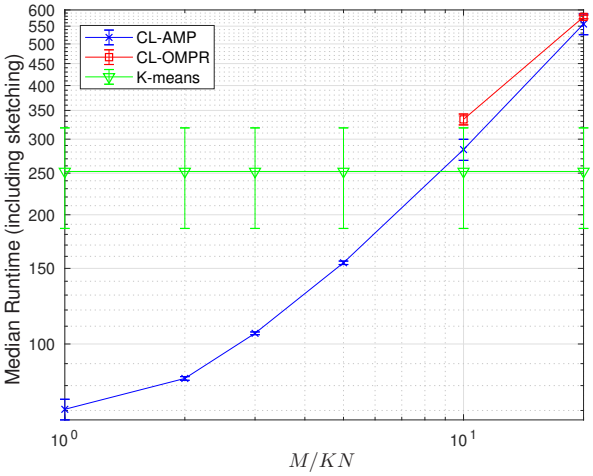
[3]Note that the true label k_t was assigned when the test sample $\overline{\boldsymbol{x}}_t$ was generated. The true label k_t does not necessarily indicate which of the true centroids $\{\boldsymbol{c}_k\}$ is closest to $\overline{\boldsymbol{x}}_t$.



(a) SSE vs. M



(b) Classification Error Rate vs. M



(c) Runtime (including sketching) vs. M

Fig. 1: Performance vs. sketch length M for $K = 10$ clusters, dimension $N = 100$, and $T = 10^7$ training samples.

the previous experiment, and the same performance metrics were evaluated. Figures 2a, 2b, and 2c show the median SSE, CER, and runtime (including sketching) versus K, for CL-AMP, CL-OMPR, and k-means++.

Figure 2a shows that, as K increases, the SSE of k-means++ remained roughly constant, as expected based on the generation of the true centers $\boldsymbol{c}_k$. For $K \leq 20$, CL-AMP yielded the best SSE for all tested values of M. For $K > 20$, CL-AMP yielded the best SSE with sketch sizes $M \in \{5KN, 10KN\}$, but performed poorly with $M = 2KN$. Meanwhile, CL-OMPR performed reasonably well with sketch size $M = 10KN$, but poorly with $M \in \{2KN, 5KN\}$.

Figure 2b shows similar trends. With sketch size $M \in \{5KN, 10KN\}$, CL-AMP had the lowest CER of any algorithm for all tested values of K. With sketch size $M = 10KN$, CL-OMPR gave CER better than k-means++ for all tested K, but with $M \in \{2KN, 5KN\}$ CL-OMPR gave CER worse than k-means++ for all tested K.

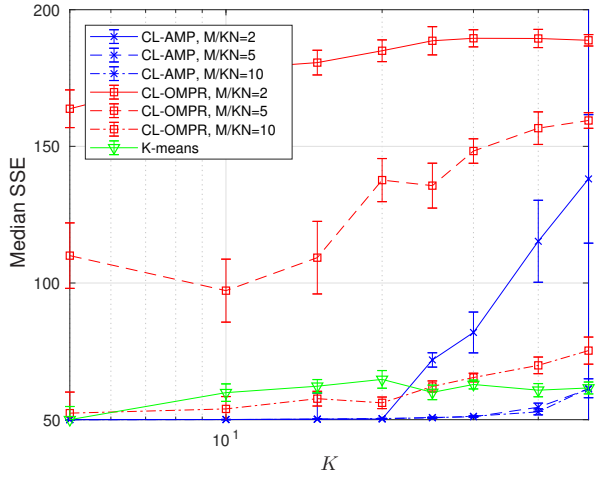
Finally, Fig. 2c shows that CL-AMP ran faster than CL-OMPR at all tested K due to its ability to work with a smaller sketch size M. For large K, Fig. 2c suggests that the runtime of both CL-AMP and CL-OMPR grow as $O(K^2)$. The $O(K^2)$ complexity scaling is expected for CL-AMP, since its complexity is $O(MNK)$ and we set $M = O(K)$. But the $O(K^2)$ complexity scaling is somewhat surprising for CL-OMPR, since its complexity is $O(MNK^2)$ and we set $M = 10NK$. Also, Fig. 2c shows that CL-AMP ran faster than k-means++ for most values of K; for the smallest tested value of K (i.e., $K = 5$), the median runtime of k-means++ was lower than CL-AMP (but the error-bar suggests that the runtime of k-means++ was highly variable at this K). For the largest tested value of K, k-means++ was again faster than CL-AMP, because the runtime of k-means++ is expected to grow linearly with K, whereas that of CL-AMP is expected to grow quadratically with K when M/KN is fixed.

3) Performance vs. dimension N: In a third experiment, we evaluated each algorithm's performance versus the dimension N (logarithmically spaced between 10 and 316) for $K = 10$ classes and sketch size $M \in \{2, 5, 10\} \times KN$. The data was generated in exactly the same way as the previous two experiments, and the same performance metrics were evaluated. Figures 3a, 3b, and 3c show the median SSE/N, the CER, and the runtime (including sketching) versus N, for CL-AMP, CL-OMPR, and k-means++.

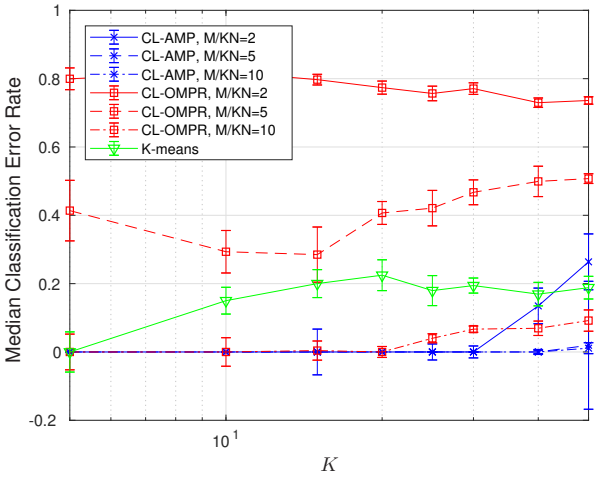
Figure 3a shows that, among all algorithms, CL-AMP achieved the lowest SSE for all tested values of N and M. Meanwhile, both CL-OMPR under sketch size $M = 10KN$ and k-means++ achieved reasonably good SSE, but CL-OMPR under smaller sketches gave much higher SSE.

Figure 3b shows that, among all algorithms, CL-AMP achieved the lowest CER for all tested values of N and M. Meanwhile, CL-OMPR under sketch size $M = 10KN$ gave similar CER to CL-AMP for most N, k-means++ gave significantly worse CER compared to CL-AMP for all N, and CL-OMPR under sketch size $M = 5KN$ or $2KN$ gave even worse CER for all N.

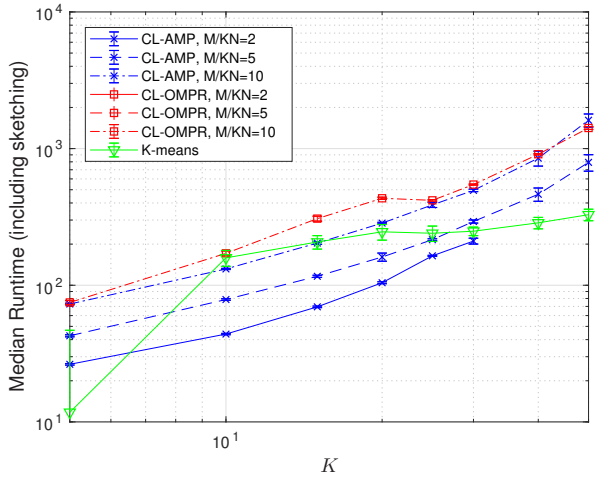
Finally, Fig. 3c shows that, among all algorithms, CL-AMP with sketch size $M = 2KN$ ran the fastest for all tested values of N. Meanwhile, CL-OMPR with sketch size $M = 10KN$ ran at a similar speed to CL-AMP with sketch size $M =$



(a) SSE vs. K



(b) Classification Error Rate vs. K



(c) Runtime (including sketching) vs. K

Fig. 2: Performance vs. number of clusters K for dimension $N = 50$, sketch size $M \in \{2, 5, 10\} \times KN$, and $T = 10^7$ training samples.

$10KN$, for all N. The runtimes for CL-OMPR with smaller sketches are not shown because it achieved significantly worse SSE than k-means++. Figure 3c suggests that, if N is increased beyond 316, then eventually k-means++ will be faster than CL-AMP under fixed M/KN.

4) Performance vs. training size T: In a final synthetic-data experiment, we evaluated each algorithm's performance versus the number of training samples T (logarithmically spaced between 10^5 and 10^8) for $K = 10$ classes, dimension $N = 50$, and sketch size $M \in \{2, 5, 10\}KN$. The data was generated in exactly the same way as the previous three experiments, and the same performance metrics were evaluated.

Figures 4a and 4b show the median SSE and CER versus T, for CL-AMP, CL-OMPR, and k-means++. From these figures, we observe that the SSE and CER for each algorithm (and sketch length M) were approximately invariant to T. CL-AMP (under any tested M) yielded the lowest values of SSE and CER. Both CL-OMPR under sketch size $M = 10KN$ and k-means++ gave reasonably good SSE and CER, but CL-OMPR under smaller sketches gave worse SSE and CER.

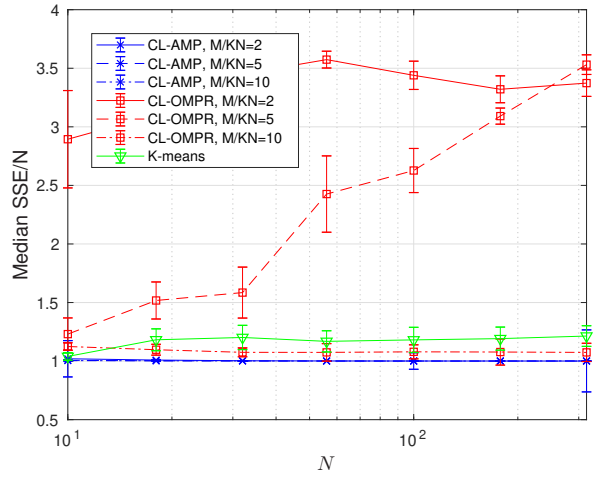
Figures 4c and 4d show the median runtime with and without sketching, respectively, for the algorithms under test. Figure 4c shows that, if sketching time is included in runtime, then all runtimes increased linearly with training size T. However, for large T, CL-AMP ran faster than k-means++ and CL-OMPR (while also achieving lower SSE and CER). Meanwhile, Fig. 4d shows that, if sketching time is not included in runtime, then the runtimes of both CL-AMP and CL-OMPR were relatively invariant to T. Also, Figures 4c and 4d together show that, for $T > 10^6$, the sketching time was the dominant contributer to the overall runtime.

B. Spectral Clustering of MNIST

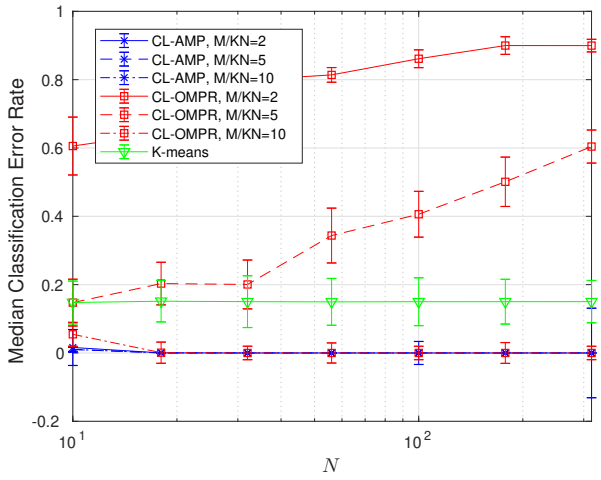
Next we evaluated the algorithms on the task of spectral clustering [25] of the MNIST dataset. This task was previously investigated for CL-OMPR and k-means++ in [6], and we used the same data preprocessing steps: extract SIFT descriptors [26] of each image, compute the K-nearest-neighbors adjacency matrix (for $K = 10$) using FLANN [27], and compute the 10 principal eigenvectors of the associated normalized Laplacian matrix (since we know $K = 10$), yielding features of dimension $N = 10$. We applied this process to the original MNIST dataset, which includes $T = 7 \times 10^4$ samples, as well as an augmented one with $T = 3 \times 10^5$ samples constructed as described in [6].

The experiment was conducted as follows. In each of 10 trials, we randomly partitioned each sub-dataset into equally-sized training and testing portions. Then, we invoked CL-AMP, CL-OMPR, and k-means++ on the training portion of the dataset, using sketch sizes $M \in \{1, 2, 3, 5, 10\} \times KN$ for CL-AMP and CL-OMPR. The algorithm parameters were the same as in Section III-A. Finally, the estimated centroids produced by each algorithm were evaluated using the same two metrics as in Section III-A: SSE on the training data, and classification error rate (CER) when the centroids were used for minimum-distance classification of the test data samples.

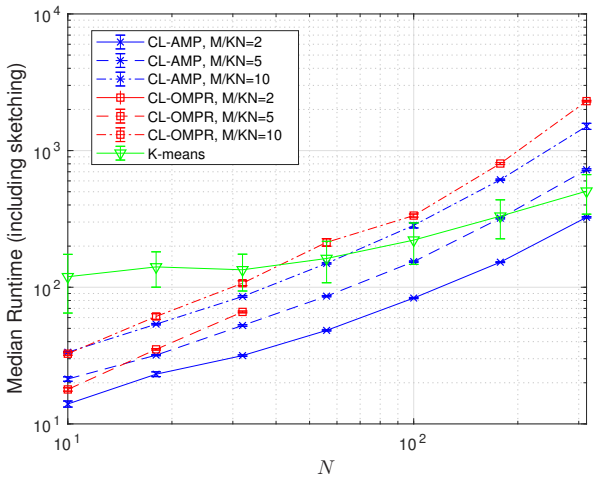
The median SSE, CER, and runtime, versus sketch length M, are shown for CL-AMP and CL-OMPR in Fig. 5 for the



(a) SSE/N vs. N

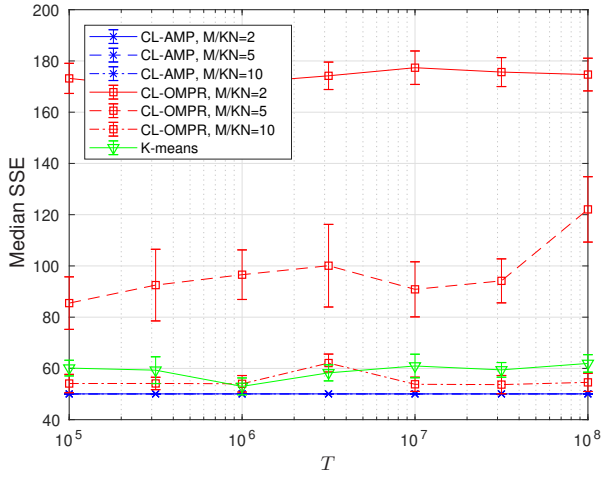


(b) Classification Error Rate vs. N

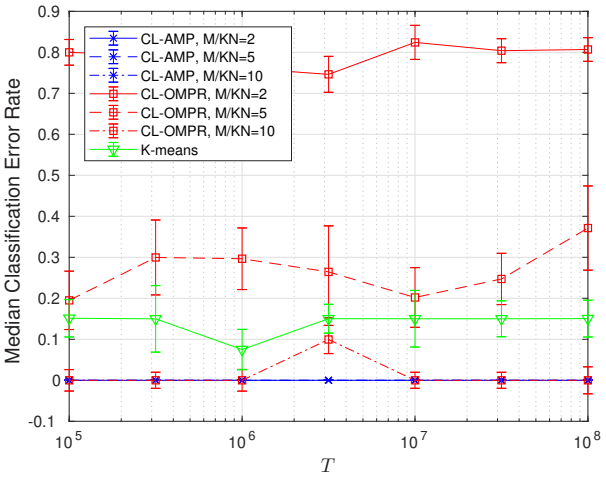


(c) Runtime (including sketching) vs. N

Fig. 3: Performance vs. dimension N for $K = 10$ classes, $T = 10^7$ samples, and sketch size $M \in \{2, 5, 10\} \times KN$.



(a) SSE vs. T



(b) Classification Error Rate vs. T

Fig. 4: Performance vs. training size T for $K = 10$ classes, dimension $N = 50$, and sketch size $M \in \{2, 5, 10\} \times KN$.

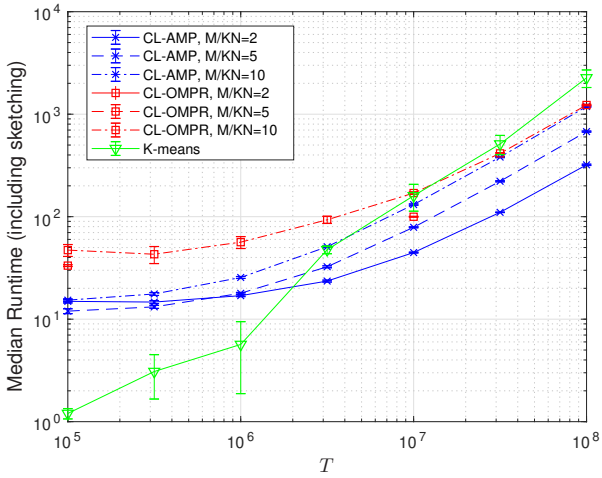
$T = 7 \times 10^4$-sample MNIST sub-dataset, and in Fig. 6 for $T = 3 \times 10^5$-sample MNIST sub-dataset. As before, k-means++ is shown, as a baseline, although it does not use the sketch and thus is performance is invariant to M. From these figures, we observe that CL-AMP and CL-OMPR gave respectable results for sketch lengths $M \geq 2KN$, and SSE nearly identical to kmeans++ for $M \geq 5KN$. For $M \geq 2KN$, however, CL-AMP yielded significantly lower CER than both CL-OMPR and k-means++, at the cost of a slower runtime. We attribute CL-AMP's slower runtime to its use of many iterations i in Algorithm 2 for hyperparameter tuning.

C. Frequency Estimation

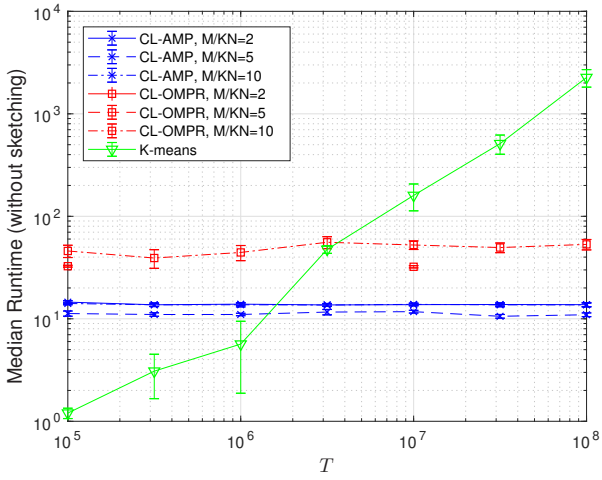
Our final experiment concerns multi-dimensional frequency estimation. Consider a sum-of-sinusoids signal of the form

$$y(\boldsymbol{t}) = \sum_{k=1}^{K} \alpha_k \exp(\mathrm{j}\boldsymbol{t}^{\mathsf{T}}\boldsymbol{c}_k), \tag{83}$$

where $\boldsymbol{c}_k \in \mathbb{R}^N$ is the frequency of the kth sinusoid, $\alpha_k > 0$ is the amplitude of the kth sinusoid, and $\boldsymbol{t} \in \mathbb{R}^N$ denotes

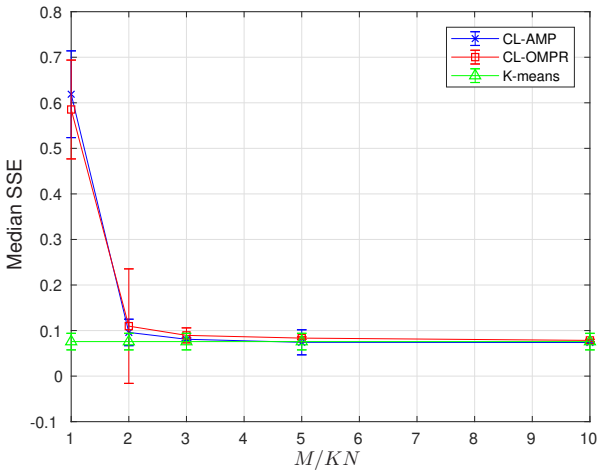


(c) Runtime (including sketching) vs. T

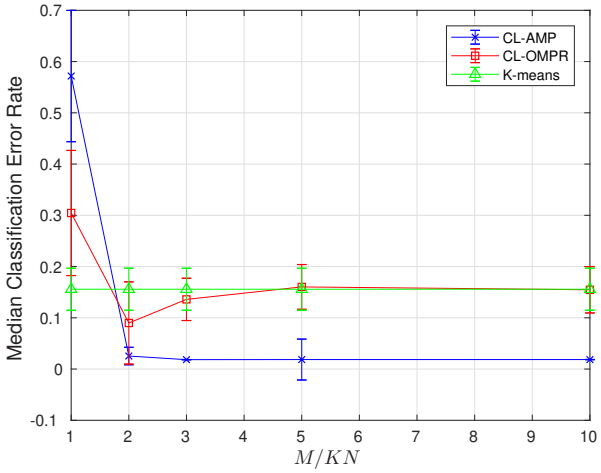


(d) Runtime (without sketching) vs. T

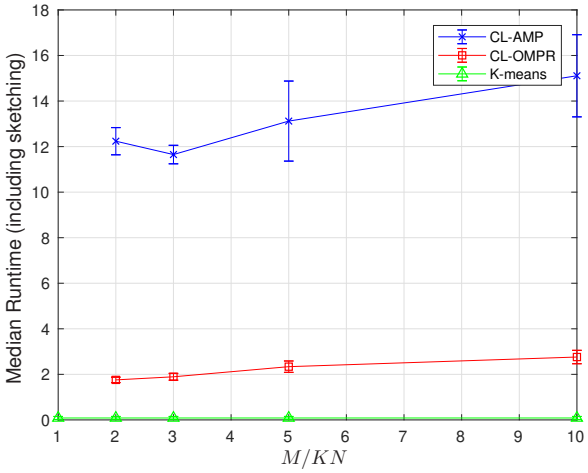
Fig. 4: Performance vs. training size T for $K = 10$ classes, dimension $N = 50$, and sketch size $M \in \{2, 5, 10\} \times KN$.



(a) SSE vs. M



(b) Classification Error Rate vs. M



(c) Runtime (including sketching) vs. M

Fig. 5: Performance vs. M for the $T = 70\,000$-sample spectral MNIST dataset, with $K = 10$ clusters and dimension $N = 10$.

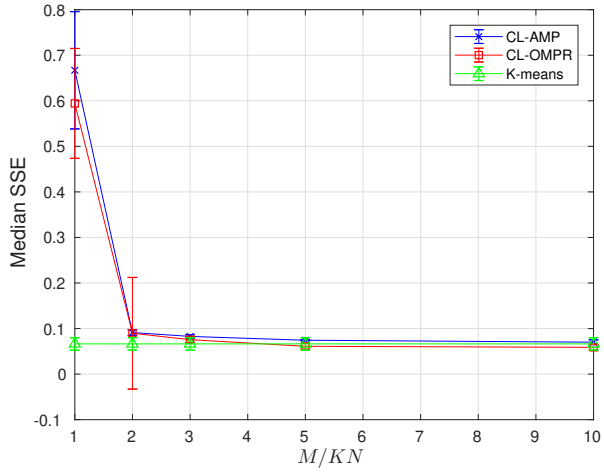
time. Given measurements of the signal $y(\boldsymbol{t})$ at a collection of random times $\boldsymbol{t} \in \{\boldsymbol{t}_m\}_{m=1}^M$, i.e.,

$$y_m = y(\boldsymbol{t}_m) \text{ for } m = 1, \dots, M, \tag{84}$$

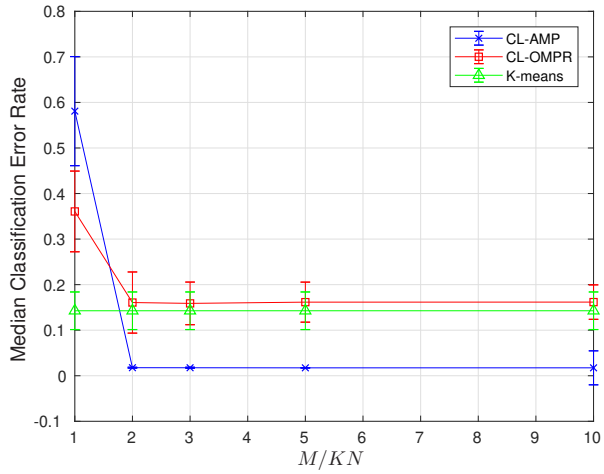
we seek to recover the frequencies $\{\boldsymbol{c}_k\}_{k=1}^K$. We are particularly interested in the case where the frequencies $\{\boldsymbol{c}_k\}$ are closely spaced, i.e., the "super-resolution" problem.

Note that the model in (83) matches that in (13) with $g_m \boldsymbol{a}_m = \boldsymbol{t}_m \ \forall m$ and $\boldsymbol{\Phi}_k = \boldsymbol{0} \forall k$, so that we can apply CL-AMP to this frequency estimation problem. The model in (83) also matches (4) with $\boldsymbol{w}_m = \boldsymbol{t}_m \ \forall m$, and so we can also apply CL-OMPR. But we cannot apply k-means++.

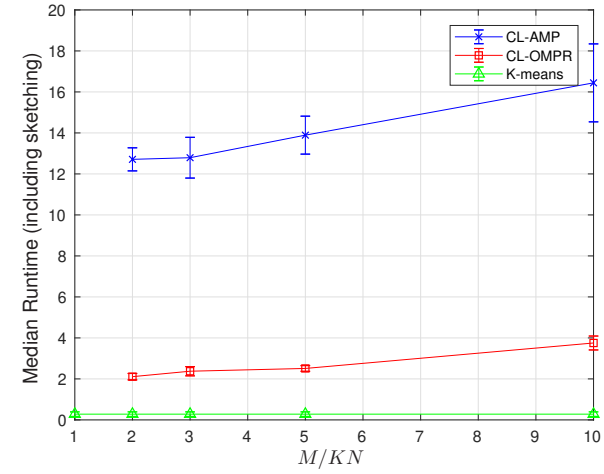
For frequency pairs $\{\boldsymbol{c}_1, \boldsymbol{c}_2\}$ with $\|\boldsymbol{c}_1 - \boldsymbol{c}_2\|_2 \geq \epsilon$, [28] claims that, with $\{\boldsymbol{w}_m\}$ drawn randomly from an appropriate distribution, one can resolve the frequencies with $M \geq O\big(\ln(1/\epsilon)\big)$ measurements. However, choosing $\boldsymbol{w}_m$ uniformly spaced on a grid would require $M \geq O(1/\epsilon)$ measurements. Thus, for a final experiment, similar to those performed in [28], we did the following. For a particular N and K (where K is even for simplicity), we generated $K/2$ pairs of frequencies $\{\boldsymbol{c}_{2k-1}, \boldsymbol{c}_{2k}\}$, where $\|\boldsymbol{c}_{2k-1} - \boldsymbol{c}_{2k}\|_2 = \epsilon$ for $k = 1, ..., K/2$.



(a) SSE vs. M/KN



(b) Classification Error Rate vs. M/KN



(c) Runtime (including sketching) vs. M/KN

Fig. 6: Performance vs. M for the $T = 300\,000$-sample spectral MNIST dataset, with $K = 10$ clusters and dimension $N = 10$.

Then, for a particular realization of $\{\boldsymbol{c}_k\}_{k=1}^K$ and $\{\boldsymbol{w}_m\}_{m=1}^M$, CL-AMP and CL-OMPR were invoked to estimate $\{\widehat{\boldsymbol{c}}_k\}_{k=1}^K$. Recovery was declared successful if

$$\max_k \|\boldsymbol{c}_{j_k} - \widehat{\boldsymbol{c}}_k\|_2 < \epsilon/2, \tag{85}$$

where $\{j_k\}_{k=1}^K$ solves the linear assignment problem (81).

For our experiment, we tested $K=4$ frequency components of dimension $N = 2$ and varied M from $3KN$ to $100KN$ while also varying ϵ from 10^{-1} to 10^{-3}. For each combination, 10 trials were performed. The empirical probability of successful recovery is shown in Figures 7-8. In Fig. 7, $\boldsymbol{a}_m$ were drawn uniformly on the unit sphere and $g_m = |g'_m|$ with $g'_m \sim \mathcal{N}\big(0, 4\epsilon^2 \log_{10}^2(\epsilon)\big)$. Superimposed on the figures are curves showing $M/KN = 0.1/\epsilon$ and $M/KN = \ln(1/\epsilon)$. From the figures, we see that CL-AMP had a higher empirical probability of recovery than CL-OMPR, especially for small ϵ. We also see that the empirical phase transition of CL-AMP is close to the $\ln(1/\epsilon)$ curve with random frequency samples and the $0.1/\epsilon$ curve with uniform frequency samples.

IV. Conclusion

In sketched clustering, the original dataset is sketched down to a relatively short vector, from which the centroids are extracted. For the sketch proposed by [5,6], we proposed the "CL-AMP" centroid-extraction method. Our method assumes that the original data follows a GMM, and exploits the recently proposed simplified hybrid generalized approximate message passing (SHyGAMP) algorithm [14]. Numerical experiments suggest that CL-AMP exhibits better sample complexity (i.e., extracts accurate clusters with fewer compressed samples) than the state-of-the-art sketched-clustering algorithm, CL-OMPR, from [5,6]. In many cases, CL-AMP also exhibits better computational complexity than CL-OMPR. Furthermore, for datasets with many samples, CL-AMP exhibits lower computational complexity than the widely used k-means++ algorithm. As future work, it would be interesting to consider the use of fast deterministic sketching with CL-AMP.

References

[1] P. Drineas, A. Frieze, R. Kannan, S. Vempala, and V. Vinay, "Clustering large graphs via the singular value decomposition," *Mach. Learn.*, vol. 56, no. 1-3, pp. 9–33, 2004.

[2] H. Steinhaus, "Sur la division des corps matériels en parties," *Bull. Acad. Polon. Sci.*, vol. 4, no. 12, pp. 801–804, 1956.

[3] A. K. Jain, "Data clustering: 50 years beyond K-means," *Pattern Recognition Letters*, vol. 31, no. 8, pp. 651–666, Jun. 2010.

[4] D. Arthur and S. Vassilvitskii, "k-means++: The advantages of careful seeding," in *Proc. Symp. Discrete Alg. (SODA)*, 2007, pp. 1027–1035.

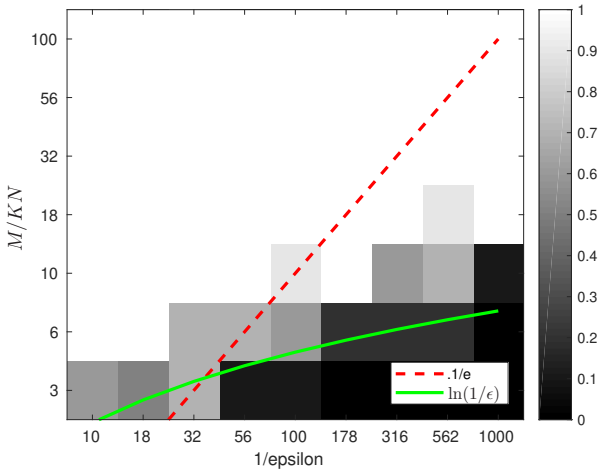
[5] N. Keriven, A. Bourrier, R. Gribonval, and P. Pérez, "Sketching for large-scale learning of mixture models," *Inform. Inference*, vol. 7, no. 3, pp. 447–508, 2017.

[6] N. Keriven, N. Tremblay, Y. Traonmilin, and R. Gribonval, "Compressive K-means," in *Proc. IEEE Int. Conf. Acoust. Speech & Signal Process.*, 2017, pp. 6369–6373.

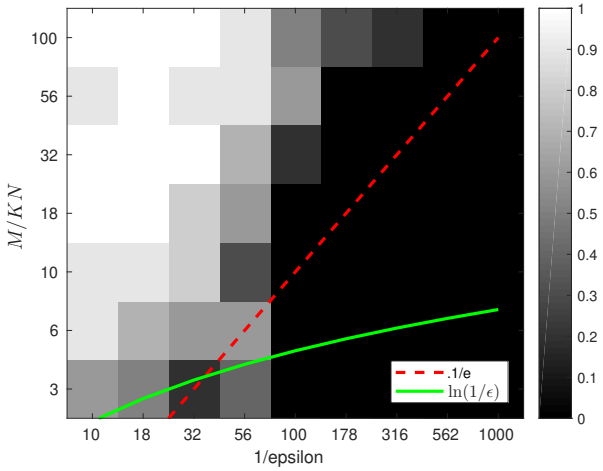
[7] R. Gribonval, G. Blanchard, N. Keriven, and Y. Traonmilin, "Compressive statistical learning with random feature moments," *arXiv:1706.07180*, 2017.

[8] A. Feuerverger and R. A. Mureika, "The empirical characteristic function and its applications," *Ann. Statist.*, vol. 5, no. 1, pp. 88–97, 1977.

[9] Y. C. Pati, R. Rezaiifar, and P. S. Krishnaprasad, "Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition," in *Proc. Asilomar Conf. Signals Syst. Comput.*, Pacific Grove, CA, 1993, pp. 40–44.

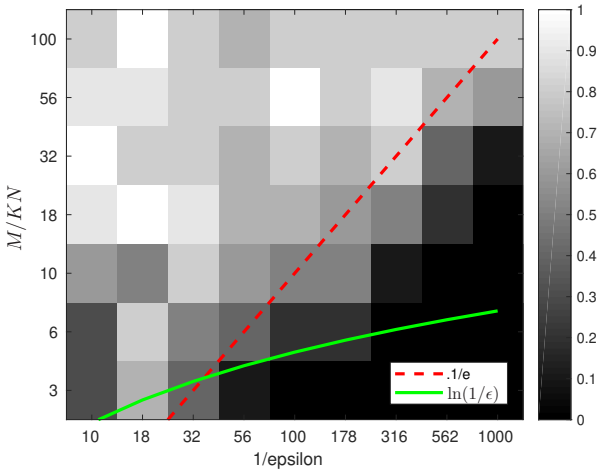


(a) CL-AMP

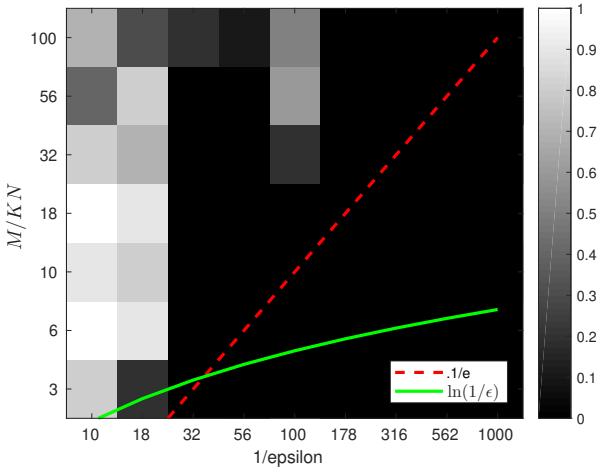


(b) CL-OMPR

Fig. 7: Frequency estimation for $K = 4$ and $N = 2$ with random time samples.



(a) CL-AMP



(b) CL-OMPR

Fig. 8: Frequency estimation for $K = 4$ and $N = 2$ with uniformly spaced time samples.

[10] E. M. Byrne, R. Gribonval, and P. Schniter, "Sketched clustering via hybrid approximate message passing," in *Proc. Asilomar Conf. Signals Syst. Comput.*, 2017, pp. 410–414.

[11] P. McCullagh and J. A. Nelder, *Generalized Linear Models*, 2nd ed. London: Chapman & Hall/CRC, 1989.

[12] A. Papoulis, *Probability, Random Variables, and Stochastic Processes*, 3rd ed. New York: McGraw-Hill, 1991.

[13] M. Rudelson and R. Vershynin, "Hanson-Wright inequality and sub-Gaussian concentration," *Electron. Commun. Probab.*, vol. 18, no. 82, pp. 1–9, 2013.

[14] E. M. Byrne and P. Schniter, "Sparse multinomial logistic regression via approximate message passing," *IEEE Trans. Signal Process.*, vol. 64, no. 21, pp. 5485–5498, 2016.

[15] D. L. Donoho, A. Maleki, and A. Montanari, "Message passing algorithms for compressed sensing," *Proc. Nat. Acad. Sci.*, vol. 106, no. 45, pp. 18 914–18 919, Nov. 2009.

[16] S. Rangan, "Generalized approximate message passing for estimation with random linear mixing," in *Proc. IEEE Int. Symp. Inform. Thy.*, Aug. 2011, pp. 2168–2172, (full version at *arXiv:1010.5141*).

[17] M. Bayati and A. Montanari, "The dynamics of message passing on dense graphs, with applications to compressed sensing," *IEEE Trans. Inform. Theory*, vol. 57, no. 2, pp. 764–785, Feb. 2011.

[18] S. Rangan, A. K. Fletcher, V. K. Goyal, E. Byrne, and P. Schniter, "Hybrid approximate message passing," *IEEE Trans. Signal Process.*, vol. 65, no. 17, pp. 4577–4592, 2017.

[19] R. Gatto and S. R. Jammalamadaka, "The generalized von Mises distribution," *Stat. Method.*, vol. 4, pp. 341–353, 2007.

[20] C. M. Bishop, *Pattern Recognition and Machine Learning*. New York: Springer, 2007.

[21] J. P. Vila and P. Schniter, "Expectation-maximization Gaussian-mixture approximate message passing," *IEEE Trans. Signal Process.*, vol. 61, no. 19, pp. 4658–4672, Oct. 2013.

[22] D. Bertsekas, *Nonlinear Programming*, 2nd ed. Athena Scientific, 1999.

[23] N. Keriven, N. Tremblay, and R. Gribonval, "SketchMLbox : a Matlab toolbox for large-scale learning of mixture models," 2016.

[24] H. W. Kuhn, "The Hungarian Method for the Assignment Problem," *Naval Research Logistics Quarterly*, pp. 83–97, 1955.

[25] A. Y. Ng, M. I. Jordan, and Y. Weiss, "On spectral clustering: Analysis and an algorithm," in *Proc. Neural Inform. Process. Syst. Conf.*, 2001, pp. 849–856.

[26] A. Vedaldi and B. Fulkerson, "VLFeat: An open and portable library of computer vision algorithms," in *Proc. ACM Intl. Conf. Multimedia*, 2010, pp. 1469–1472.

[27] M. Muja and D. G. Lowe, "Fast approximate nearest neighbors with automatic algorithm configuration," in *Proc. Intl. Conf. Comp. Vision Thy. Appl. (VISAPP)*, 2009, pp. 331–340.

[28] Y. Traonmilin, N. Keriven, R. Gribonval, and G. Blanchard, "Spikes super-resolution with random Fourier sampling," in *Proc. Workshop Signal Process. Adapt. Sparse Struct. Repr. (SPARS)*, 2017, pp. 1–2.