Sketched Clustering via Hybrid Approximate Message Passing

Evan Byrne, Rémi Gribonval, and Philip Schniter

The Ohio State University

Inria — inventors for the digital world

(Supported by NSF Grant 1716388 and MIT Lincoln Labs)

Traditional Clustering Problem Statement

- Given a dataset of T N-dimensional feature vectors $\boldsymbol{X} = [\boldsymbol{x}_1, ..., \boldsymbol{x}_T] \in \mathbb{R}^{N\times T}$, estimate K N-dimensional cluster centers $\boldsymbol{C} = [\boldsymbol{c}_1, ..., \boldsymbol{c}_K] \in \mathbb{R}^{N\times K}$ that minimize sum of squared errors (SSE):

$$\text{SSE} = \sum_{t=1}^{T} \min_k \|\boldsymbol{x}_t - \boldsymbol{c}_k\|_2^2. \tag{1}$$

- However, finding $\boldsymbol{C}$ to minimize the SSE in (1) is NP-hard.
- K-means is a commonly applied heuristic approach.
- K-means generally works well wrt minimizing the SSE, except its complexity is $\mathcal{O}(NKTl)$, where l is the number of iterations, which is prohibitive for large T.

Sketched Clustering

- Sketched clustering [Kerivan 16] is an alternate approach possibly more efficient than K-means.
- Let $\boldsymbol{y} \in \mathbb{C}^M$ be the "sketch" of $\boldsymbol{X}$, where

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

for some set of N-dimensional frequency vectors $\boldsymbol{W} = [\boldsymbol{w}_1, ... \boldsymbol{w}_M]$.
- The sketch in (2) can be interpreted as the empirical characteristic function of the dataset $\boldsymbol{X}$.
- CLOMPR [Kerivan 17] is the state-of-the-art Sketched Clustering algorithm, which solves

$$\{\widehat{\boldsymbol{C}}, \widehat{\boldsymbol{\alpha}}\} = \underset{\boldsymbol{C},\boldsymbol{\alpha}}{\arg\min} \sum_{m=1}^{M} \left| y_m - \sum_{k=1}^{K} \alpha_k \exp(\mathrm{j}\boldsymbol{w}_m^{\mathsf{T}}\boldsymbol{c}_k)\right|^2 \tag{3}$$

via a greedy optimization approach.
- In practice, $\widehat{\boldsymbol{C}}_{\text{CLOMPR}}$ works well wrt SSE compared to $\widehat{\boldsymbol{C}}_{\text{K-means}}$, despite no link between (3) and (1).
- CLOMPR's complexity is $\mathcal{O}(MNK^2l + MNT)$, which includes the cost of computing $\boldsymbol{y}$.
- Note that once $\boldsymbol{y}$ is computed, $\boldsymbol{X}$ is not stored during CLOMPR, so the memory requirement is significantly reduced.
- CLOMPR's authors have developed several approaches for randomly generating the frequencies $\boldsymbol{w}_m$ and have observed around $M \approx 10KN$ frequencies necessary for accurate performance.

Sketched Clustering via Approximate Message Passing

- We choose to model the feature vectors $\boldsymbol{x}_t$ with a Gaussian Mixture where the mixture centers are the "true" cluster centers, i.e. ,

$$\boldsymbol{x}_t \sim \sum_{k=1}^{K} \alpha_k \mathcal{N}(\boldsymbol{c}_k, \boldsymbol{\Sigma}_k). \tag{4}$$

- Then, for large T,

$$y_m = \frac{1}{T}\sum_{t=1}^{T}\exp(\mathrm{j}\boldsymbol{w}_m^{\mathsf{T}}\boldsymbol{x}_t) \approx \mathrm{E}\{\exp(\mathrm{j}\boldsymbol{w}_m^{\mathsf{T}}\boldsymbol{x}_t)\} = \sum_{k=1}^{K}\alpha_k \exp\big(\mathrm{j}\underbrace{\boldsymbol{w}_m^{\mathsf{T}}\boldsymbol{c}_k}_{\triangleq z_{mk}} - \underbrace{\boldsymbol{w}_m^{\mathsf{T}}\boldsymbol{\Sigma}_k\boldsymbol{w}_m}_{\triangleq \tau_{mk}}/2\big), \tag{5}$$

and so

$$p_{\mathsf{y}|\mathbf{z}}(y_m \,|\, \boldsymbol{z}_m) = \delta\Big(y_m - \sum_{k=1}^{K}\alpha_k \exp\left(\mathrm{j}z_{mk} - \tau_{mk}/2\right)\Big), \tag{6}$$

where $\{\tau_{mk}\}$ and $\{\alpha_k\}$ are treated as hyperparameters.
- If we assume $p_{\mathsf{y}|\mathbf{z}}(y_m \,|\, \boldsymbol{z}_m)$ are independent across m and assume $p_{\mathbf{C}}(\boldsymbol{C}) = \prod_{n=1}^{N} p_{\mathbf{c}}(\boldsymbol{c}_n)$, we obtain

$$p_{\mathbf{y},\mathbf{C}}(\boldsymbol{y}, \boldsymbol{C}) = \prod_{m=1}^{M} p_{\mathsf{y}|\mathbf{z}}(y_m \,|\, \boldsymbol{w}_m^{\mathsf{T}}\boldsymbol{C}) \prod_{n=1}^{N} p_{\mathbf{c}}(\boldsymbol{c}_n). \tag{7}$$

- With (7), we treat sketched clustering as an inference problem rather than an optimization problem.
- In particular, we approximate

$$\widehat{\boldsymbol{C}} = \mathrm{E}\{p_{\mathbf{C}\,|\,\mathbf{y}}(\boldsymbol{C} \,|\, \boldsymbol{y})\}, \tag{8}$$

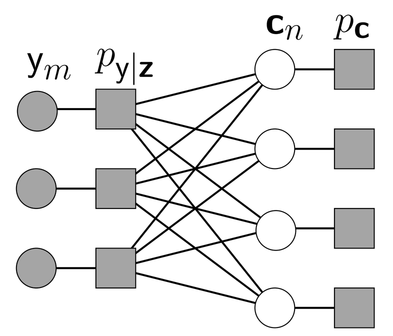
using the Simplified-Hybrid-GAMP (SHyGAMP) algorithm [Byrne 16].
- The SHyGAMP algorithm is based on the more general HyGAMP algorithm [Rangan 17]. The only difference between the two is SHyGAMP restricts the messages that are passed to have diagonal covariance matrices, which drastically reduces computational complexity.

References

- N. Keriven, A. Bourrier, R. Gribonval, and P. Pérèz, "Sketching for Large-Scale Learning of Mixture Models," in *Proc. IEEE Intl. Conf. on Acoustic, Speech and Signal Processing (ICASSP)*, March 2016.
- N. Keriven, N. Tremblay, Y. Traonmilin, and R. Gribonval, "Compressive K-means," in *Proc. IEEE Intl. Conf. on Acoustic, Speech and Signal Processing (ICASSP)*, March 2017.
- E. Byrne and P. Schniter, "Sparse Multinomial Logistic Regression via Approximate Message Passing," in *IEEE Trans. on Signal Processing*, vol. 64, no. 21, pp. 5485-5498, Nov. 2016.
- S. Rangan, A. K. Fletcher, V. K. Goyal, E. Byrne, and P. Schniter, "Hybrid Approximate Message Passing," in *IEEE Trans. on Signal Processing*, vol. 65, no. 17, pp. 4577-4592, Sep. 2017.
- P. Schniter and S. Rangan, "Compressive Phase Retrieval via Generalized Approximate Message Passing," *IEEE Trans. on Signal Processing*, vol. 63, no. 4, pp. 1043-1055, Feb. 2015.

Description of SHyGAMP

- SHyGAMP approximates sum-product loopy belief propagation on factor graphs of the form:



- SHyGAMP iteratively passes messages back and forth between the $p_{\mathbf{c}}$ and $p_{\mathsf{y}|\mathbf{z}}$ nodes until convergence.
- Messages are approximated as K-dimensional Gaussian pdfs with diagonal covariance structure.
- This iterative message passing allows an NK-dimensional inference problem is broken into many K-dimensional inference problems.
- SHyGAMP's complexity for sketched clustering is $\mathcal{O}\big(K(M+N)l + MNT\big)$.
- The SHyGAMP algorithm can be divided into "linear" and "non-linear" steps.
- At each iteration the non-linear steps require computing the mean and covariance of the estimands using the following approximate posterior distributions:

$$p_{\mathbf{c}|\mathbf{r}}(\boldsymbol{c}_n|\widehat{\boldsymbol{r}}_n; \boldsymbol{Q}_n^{\mathbf{r}}) \propto p_{\mathbf{c}}(\boldsymbol{c}_n)\mathcal{N}(\boldsymbol{c}_n; \widehat{\boldsymbol{r}}_n, \boldsymbol{Q}_n^{\mathbf{r}}) \tag{9}$$

and

$$p_{\mathbf{z}|\mathsf{y},\mathbf{p}}(\boldsymbol{z}_m|y_m, \widehat{\boldsymbol{p}}_m; \boldsymbol{Q}_m^{\mathbf{p}}) \propto p_{\mathsf{y}|\mathbf{z}}(y_m|\boldsymbol{z}_m)\mathcal{N}(\boldsymbol{z}_m; \widehat{\boldsymbol{p}}_m, \boldsymbol{Q}_m^{\mathbf{p}}), \tag{10}$$

where the quantities $\widehat{\boldsymbol{p}}_m$, $\boldsymbol{Q}_m^{\mathbf{p}}$, $\widehat{\boldsymbol{r}}_n$, and $\boldsymbol{Q}_n^{\mathbf{r}}$ are computed during the linear steps.

The SHyGAMP Algorithm

Require: frequency matrix $\boldsymbol{W}$, sketch $\boldsymbol{y}$, pdfs $p_{\mathbf{c}|\mathbf{r}}$ and $p_{\mathbf{z}|\mathsf{y},\mathbf{p}}$ from (9)-(10), initializations $\widehat{\boldsymbol{r}}_n(0)$, $\boldsymbol{Q}_n^{\mathbf{r}}(0)$.
Ensure: $t \leftarrow 0$; $\widehat{\boldsymbol{s}}_m(0) \leftarrow \boldsymbol{0}$.

1: **repeat**
2: $\forall n: \widehat{\boldsymbol{c}}_n(t) \leftarrow \mathrm{E}\left\{\mathbf{c}_n \,\middle|\, \mathbf{r}_n = \widehat{\boldsymbol{r}}_n(t-1); \boldsymbol{Q}_n^{\mathbf{r}}(t-1)\right\}$
3: $\forall n: \boldsymbol{Q}_n^{\mathbf{c}}(t) \leftarrow \mathrm{cov}\left\{\mathbf{c}_n \,\middle|\, \mathbf{r}_n = \widehat{\boldsymbol{r}}_n(t-1); \boldsymbol{Q}_n^{\mathbf{r}}(t-1)\right\}$
4: $\forall m: \boldsymbol{Q}_m^{\mathbf{p}}(t) \leftarrow \sum_{n=1}^{N} W_{nm}^2 \boldsymbol{Q}_n^{\mathbf{c}}(t)$
5: $\forall m: \widehat{\boldsymbol{p}}_m(t) \leftarrow \sum_{n=1}^{N} W_{nm}\widehat{\boldsymbol{c}}_n(t) - \boldsymbol{Q}_m^{\mathbf{p}}(t)\widehat{\boldsymbol{s}}_m(t-1)$
6: $\forall m: \widehat{\boldsymbol{z}}_m(t) \leftarrow \mathrm{E}\left\{\mathbf{z}_m \,\middle|\, y_m, \mathbf{p}_m = \widehat{\boldsymbol{p}}_m(t); \boldsymbol{Q}_m^{\mathbf{p}}(t)\right\}$
7: $\forall m: \boldsymbol{Q}_m^{\mathbf{z}}(t) \leftarrow \mathrm{cov}\left\{\mathbf{z}_m \,\middle|\, y_m, \mathbf{p}_m = \widehat{\boldsymbol{p}}_m(t); \boldsymbol{Q}_m^{\mathbf{p}}(t)\right\}$
8: $\forall m: \boldsymbol{Q}_m^{\mathbf{s}}(t) \leftarrow [\boldsymbol{Q}_m^{\mathbf{p}}(t)]^{-1} - [\boldsymbol{Q}_m^{\mathbf{p}}(t)]^{-1}\boldsymbol{Q}_m^{\mathbf{z}}(t)[\boldsymbol{Q}_m^{\mathbf{p}}(t)]^{-1}$
9: $\forall m: \widehat{\boldsymbol{s}}_m(t) \leftarrow [\boldsymbol{Q}_m^{\mathbf{p}}(t)]^{-1}\big(\widehat{\boldsymbol{z}}_m(t) - \widehat{\boldsymbol{p}}_m(t)\big)$
10: $\forall n: \boldsymbol{Q}_n^{\mathbf{r}}(t) \leftarrow \left[\sum_{m=1}^{M} W_{nm}^2 \boldsymbol{Q}_m^{\mathbf{s}}(t)\right]^{-1}$
11: $\forall n: \widehat{\boldsymbol{r}}_n(t) \leftarrow \widehat{\boldsymbol{c}}_n(t) + \boldsymbol{Q}_n^{\mathbf{r}}(t)\sum_{m=1}^{M} W_{nm}\widehat{\boldsymbol{s}}_m(t)$
12: $t \leftarrow t+1$
13: **until** Terminated

Computation of SHyGAMP Non-linear Steps

- The key technical challenge in applying SHyGAMP to sketched clustering is computing Lines 6-7 of the SHyGAMP algorithm when $p_{\mathsf{y}|\mathbf{z}}$ has the form in (6).
- We have developed a method based on approximating $p_{\mathsf{y}|\mathbf{z}}(y_m|\boldsymbol{z}_m)$ with a Generalized von Mises distribution and evaluating the necessary integrals with the Laplace Approximation.

Parameter Tuning

- Our Gaussian Mixture model in (4) requires properly selecting α_k and τ_{mk} in (6).
- Currently, we assume τ_{mk} is invariant to m.
- Allowing τ_{mk} to vary with m increases the generalizability of the model, but is more difficult to learn. Exploring this is one avenue for future work.
- One approach to tuning α_k and τ_k is via approximate EM:

$$\{\widehat{\boldsymbol{\alpha}}, \widehat{\boldsymbol{\tau}}\} = \underset{\boldsymbol{\alpha}\geq\boldsymbol{0}, \boldsymbol{\alpha}^{\mathsf{T}}\boldsymbol{1}=1, \boldsymbol{\tau}^{w}\geq\boldsymbol{0}}{\arg\max} \sum_{m=1}^{M}\int_{\mathbb{R}^K}\mathcal{N}(\boldsymbol{z}_m; \widehat{\boldsymbol{z}}_m, \boldsymbol{Q}_m^{\mathbf{z}})\log p(y_m|\boldsymbol{z}_m)\,\mathrm{d}\boldsymbol{z}_m, \tag{11}$$

which can be optimized at every SHyGAMP iteration (immediately after Line 7) using gradient-projection.
- An alternate approach based on Bethe Free Energy Minimization [Schniter 15] is currently in development.

Comparison Between SHyGAMP, CLOMPR++ and K-means++

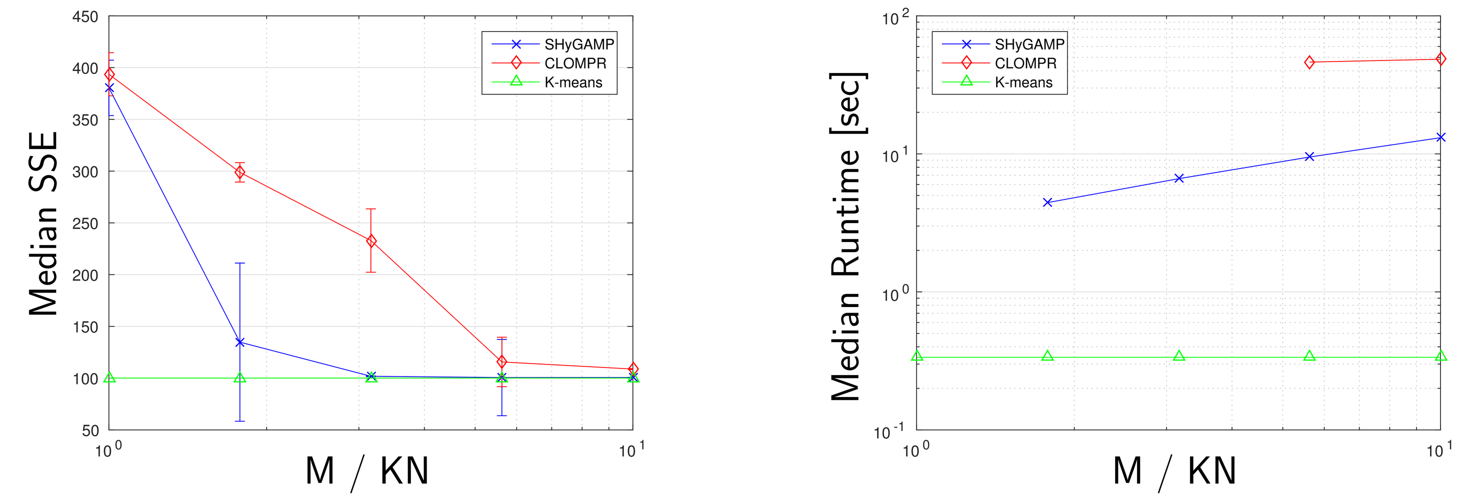
Data generation model

- True $\boldsymbol{x}_t \sim \sum_{k=1}^{K}\alpha_k\mathcal{N}(\boldsymbol{c}_k, \boldsymbol{I}_N)$ for $\boldsymbol{c}_k \sim \mathcal{N}(\boldsymbol{0}_N, (1.5\sqrt[N]{K})^2\boldsymbol{I}_N)$, and $\alpha_k = \frac{1}{K}\ \forall\, k$.

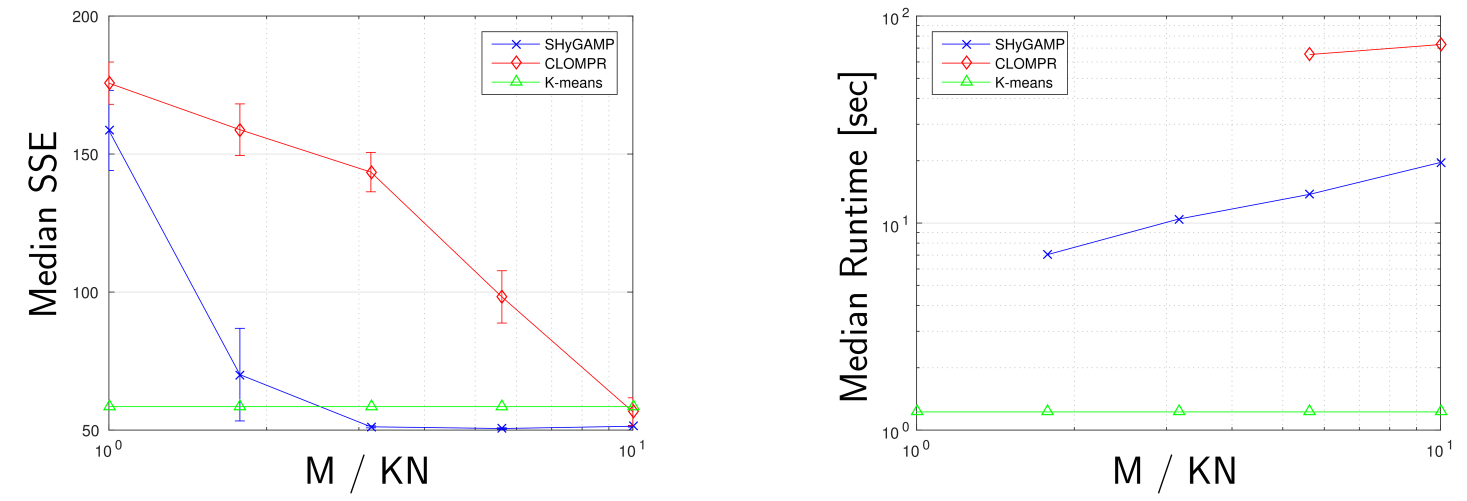
Simulation: SSE vs M

- For each $N \in \{50, 100\}$ and $K \in \{5, 10\}$, we tested several sketch lengths $M \in [KN, 10KN]$.
- We report the Median SSE and Median Runtime for SHyGAMP, CLOMPR++ and K-means++ over 10 trials. For SHyGAMP and CLOMPR++, we report runtime only when SSE $< 2\times$SSE(K-means++).
- Compared to CLOMPR++, SHyGAMP has lower SSE and is faster at all tested M.

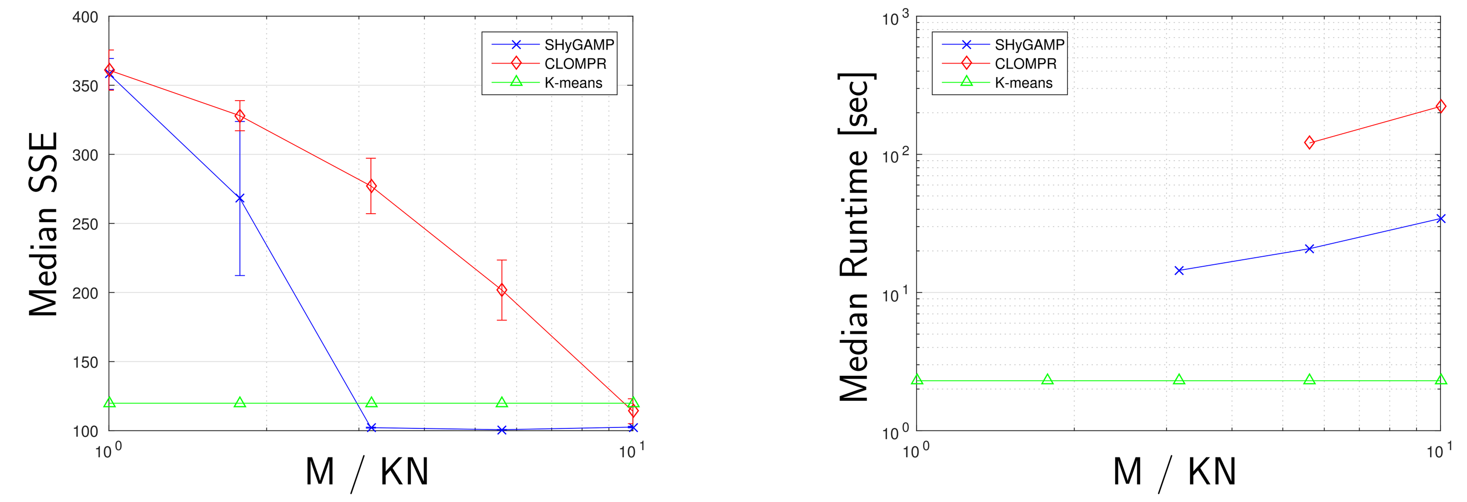
$K = 5$, $N = 100$:



$K = 10$, $N = 50$:



$K = 10$, $N = 100$:



Simulation: Classification Error vs Runtime

- Dimensions $N = 20$ and $K = 30$. Training set with $T = 10^4$ samples.
- Recovered cluster-centers used for classification on a test set with $T = 5 \times 10^6$ samples.
- SHyGAMP and CLOMPR++ traces vary sketch size M logarithmically within $[KN, 100KN]$.
- K-means traces vary training subset size, in $\{\frac{T}{2^6}, \frac{T}{2^5}, ..., T\}$, for a fixed $\#$ replicates in $\{256, ..., 4096\}$.
- Results are the median of 5 trials (each trial used the same true centroids, but random train/test sets).
- SHyGAMP converged to the Bayes' Error Rate (BER) faster than K-means and CLOMPR++.