Bubble Clustering: Set Covering via Union of Ellipses

Matt Kraning, Arezou Keshavarz, Lei Zhao
CS229, Stanford University
Email: {mkraning,arezou,leiz}@stanford.edu

Abstract

We develop an algorithm called bubble clustering which attempts to cover a given set of points with the union of k ellipses of minimum total volume. This algorithm operates by splitting and merging ellipses according to an annealing schedule and does not assume any prior distribution on the data. We compare our algorithm with k-means and the EM algorithm for mixture of Gaussians. Numerical results suggest that our algorithm achieves superior performance to k-means for prior-less data and comparable performance to algorithms that have access to prior information for statistically generated data.

I. Project Goal

We want to cover a set of N points $\mathbf{X} = \{x_1, x_2, \ldots, x_N\}$, $x_i \in \mathbf{R}^d$, by the union of k ellipses $(\mathcal{E}_1, \mathcal{E}_2, \ldots, \mathcal{E}_k)$ such that the total volume of all ellipses is minimized. Unlike mixture of Gaussians, we do not assume a prior distribution on data, nor do we even assume the data *has* a prior distribution. Consequently, the optimization problem we are attempting to solve is

$$\begin{aligned} \text{minimize} \quad & \textstyle\sum_{i=1}^{k} \log\det A_i^{-1} \\ \text{subject to} \quad & \|A_{I_j} x_j + b_{I_j}\|_2 \leq 1 \\ & I_j \in \{1, \ldots, k\}, \quad j = 1 \ldots N \\ & A_i \in S_{++}^n, \quad i = 1 \ldots k, \end{aligned}$$

which is an NP-Hard problem due to the presence of the constraint $I_j \in \{1, \ldots, k\}$; this constraint specifics that we find the optimal assignment of every given data point x_j to a specific ellipse $\mathcal{E}_{I_j}$.

II. Algorithm

Because solving our problem exactly is NP-Hard, we instead focus on developing an approximate algorithm to solve this problem. Rather than use EM, whose performance can depend heavily on initial starting conditions that are usually randomly generated, we sought an algorithm that is both deterministic and as free of tunable parameters as possible. This was done to maximize the algorithm's universality: it should not depend on random numbers, nor should it have to be highly 'tuned' to work well on specific datasets.

Bubble clustering works by iteratively splitting minimum volume ellipses that were fit to parts of the data on earlier iterations. The intuition behind bubble clustering is that good clustering performance can be obtained by looking at a global summary of part of the dataset (the points within a given ellipse) and then taking a lower volume covering of that same set by splitting the ellipse containing that set into two smaller ellipses. After splitting down to k ellipses, bubble clustering then performs a variant of simulated annealing to escape from weak local minima. The annealing schedule controls if ellipses are split or merged during each iteration and guarantees that the number of ellipses converges to the pre-specified value k. This is described by the pseudocode

below and a more detailed description of how splitting and merging of ellipses is performed is given in the following sections.

Given data $\mathbf{X}$, number of final clusters k, and an annealing schedule, $anneal$, fit a single ellipse to the data. Then,
for $t = 1$ to k **do**
  split ellipse
end for
for $i = 1$ to $length(anneal)$ **do**
  if $anneal(i) == split$ **then**
    split ellipse
  else if $anneal(i) == merge$ **then**
    if one ellipse contained inside another **then**
      eliminate smaller ellipse
    else
      merge two ellipses
    end if
  end if
end for

A. Splitting

The density of the ith ellipse, ρ_i, is the ratio of the volume of the ellipse to the number of data points inside the ellipse, i.e.

$$\rho_i = \frac{vol(\mathcal{E}_i)}{|\{i : 1 \leq i \leq \kappa, x_i \in \mathcal{E}_i\}|}.$$

In the splitting phase of a given iteration with κ ellipses, bubble clustering greedily chooses the ellipse with lowest density, $\mathcal{E}_I$, where $I = \underset{i \in \{1,\ldots,\kappa\}}{\operatorname{argmin}} \rho_i$, and splits it into two ellipses $\mathcal{E}_{I_1}, \mathcal{E}_{I_2}$. It splits $\mathcal{E}_I$ along the hyperplane perpendicular to the direction which gives maximum data variance for data points within $\mathcal{E}_I$. This direction is the eigenvector corrosponding to the maximum eigenvalue for the *empirical* covariance matrix of the data points inside $\mathcal{E}_I$. After splitting, it fits new minimum-volume ellipses around both sets of points by solving the convex optimization problem

$$\begin{array}{ll} \text{minimize} & \log\det A_{I_j}^{-1} \\ \text{subject to} & \|A_{I_j} x_i + b_{I_j}\|_2 \leq x_i, \quad x_i \in \mathcal{E}_{I_j} \end{array}$$

for $j = 1, 2$.

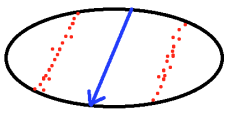

Fig. 1. Splitting along the eigenvector of maximum eigenvalue for the empirical covariance matrix of the data point inside the chosen ellipse.

B. Merging Ellipses

In the merging step, we first check if there is any ellipse that is completely contained in another ellipse. Denote the ellipses by $\mathcal{E}_i = \{x | \|A_i x - b_i\|_2^2 \leq 1\}$ and $\mathcal{E}_j = \{x | \|A_j x - b_j\|_2^2 \leq 1\}$. We want to check if $\mathcal{E}_i \subseteq \mathcal{E}_j$, or equivalently, whether $\|A_i x - b_i\|_2^2 \leq 1 \Rightarrow \|A_j x - b_j\|_2^2 \leq 1$.

The S-Procedure specifies the necessary and sufficient conditions for this to occur: (If $\exists \tau \geq 0$ such that $Q - \tau P \geq 0$) $\Longleftrightarrow$ ($x^T P x \geq 0 \Rightarrow x^T Q x \geq 0$) [1]. Applying the S-procedure to our problem, it is sufficient to check the feasibility of the linear matrix inequalities (LMIs)

$$\begin{bmatrix} -D & Dd \\ (Dd)^T & -d^T Dd + 1 \end{bmatrix} - \tau \begin{bmatrix} -I & 0 \\ 0 & 1 \end{bmatrix} \geq 0, \quad \tau \geq 0,$$

where $d = (A_i^T A_i)^{\frac{1}{2}}(A_i^{-1} b_i - A_j^{-1} b_j)$ and $D = (A_i^T A_i)^{-\frac{T}{2}} A_j^T A_j (A_i^T A_i)^{-\frac{1}{2}}$, for all pairs of ellipses $\mathcal{E}_i, \mathcal{E}_j, i \neq j$.

If we cannot find an ellipse contained inside another ellipse, we merge two ellipses based on their relative orientations and overall closeness. To measure relative orientation, we use the difference between the Mahalanobis distances of each ellipse to the center of the other ellipse, where the Mahalanobis distance to a point x from the ellipse $\mathcal{E}_i$ with center c_i and inverse covariance matrix A_i is defined as $\|x\|_{\mathcal{E}_i}^2 = (x - c_i)^T A_i (x - c_i)$. If these distances are close, then the ellipses are aligned. Consequently, we merge the two ellipses, $\mathcal{E}_i$ and $\mathcal{E}_j$, that minimize $|\|c_j\|_{\mathcal{E}_i} - \|c_i\|_{\mathcal{E}_j}| + \lambda \|c_i - c_j\|_2$, where λ is a tradeoff parameter, and c_i and c_j are the centers of $\mathcal{E}_i$ and $\mathcal{E}_j$, respectivly. Then, we merge ellipses $\mathcal{E}_i$ and $\mathcal{E}_j$ by fitting a minimum volume ellipse around the union of points contained in either ellipse (see Figure 2).

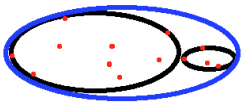

Fig. 2. Merging ellipses: two black ellipses are merged into the blue ellipse.

III. Numerical results

A. Synthetic data, no prior

We initialized k-means randomly 50 times, and took the result with minimum total ellipsoid volume, which was 1.28. The average volume over all iterations was 1.59. Our Bubble clustering algorithm attained a total volume of 1.24. The clusters generated by the K-Means algorithm have some overlapping clusters, whereas the clusters generated by the bubble clustering algorithm are separable (Figure 3).

B. Synthetic data, mixture of Gaussian prior

In this experiment, we assume a Gaussian mixture model with 9 classes and a uniform distribution over those classes.

$$p(x) = \sum_{i=1}^{9} \frac{1}{9} \frac{1}{(2\pi)^{d/2} \sqrt{\det |S_i|}} \exp\{-\frac{1}{2}(x - \mu_i)^T S_i^{-1} (x - \mu_i)\},$$

where the dimension d is set to be 2. The parameters $S_i, \mu_i, i = 1, .., 9$ are randomly generated and fixed for all the data. We use this Gaussian mixture model to generate 300 points, shown in Figure 4.

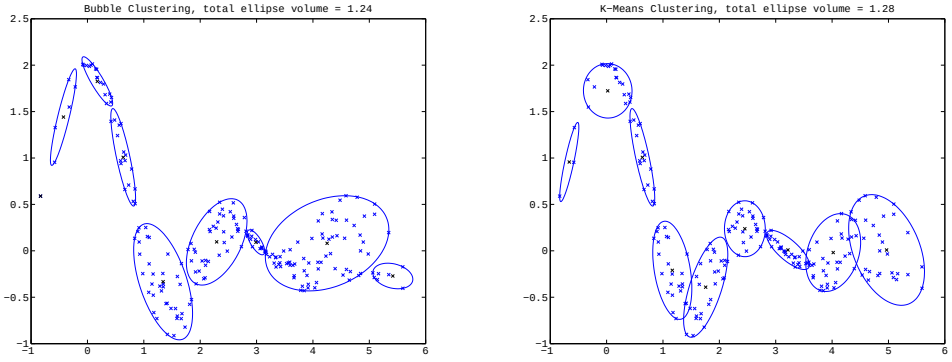

Fig. 3. Comparison between bubble clustering (left) and k-means (right)

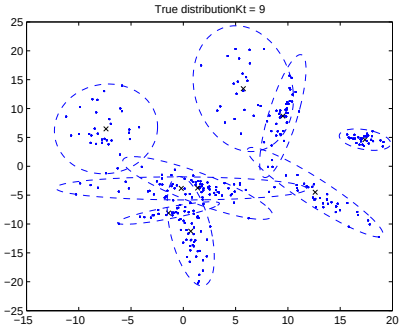

Fig. 4. True distribution of the data, generated by a mixture of Gaussians.

We run Bubble Clustering on this data set and compare it with the result obtained by the EM algorithm for mixture of Gaussian. As Figure 5 indicates, even without a prior assumption, the bubble clustering obtains comparable results.

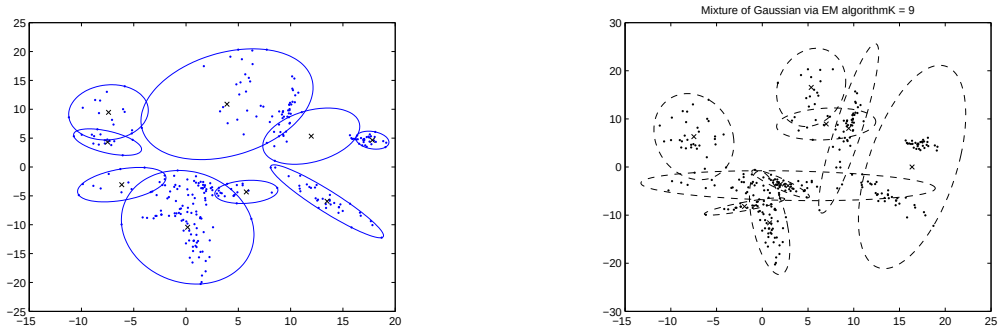

Fig. 5. Comparison between bubble clustering algorithm (left) and the EM algorithm for mixture of Gaussian (right)

C. Real Data: Blogger Population

We also applied the bubble clustering algorithm to a real dataset. We obtained an XML feed from [2], which provides a live feed of blogger entries. We extracted the bloggers residing in California,

and ran the bubble clustering algorithm to find clusters of blogger population (Figure 6).

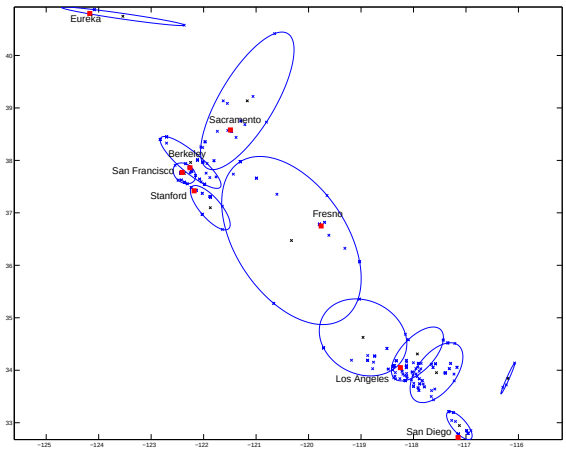

Fig. 6. Blogger population clusters in California

The clusters correspond pretty well to metropolitan areas such as Los Angeles and San Francisco, with smaller ellipses being generally indicative of high population densities over their area.

IV. Conclusions and Future Work

Bubble Clustering is a novel universal clustering method that does not have to rely on the kindness of random number generators or the smoke and mirrors of tunable parameters for good performance. It is effective over off the shelf methods for both clustering and finding small set coverings and can also show data cluster correlations when a prior is either unknown or non-existent.

The main areas for improvement involve using lookahead in both splitting and merging ellipses. In picking which ellipse to split, our ultimate aim is to have the sum of the volumes of both new ellipses be significantly smaller than the volume of the original ellipse before splitting. It is computationally intensive to evaluate the volume reduction gained by splitting each ellipse and only then pick the best ellipse to split; so we instead rely on the heuristic of density as a proxy metric for splitting efficacy. Similarly, in the case of merging, if we check all possible pairs of ellipses to merge and only then choose to merge the two with least volume increase, the algorithm will be drastically slowed down. Finding ways to effectively prune these search spaces (as we did by utilizing S-Procedure to check if one ellipse is inside another) would allow lookahead to be implemented for both splitting and merging, which would likely lead to better performance.

V. Acknowledgements

We would like to thank Eric Chu and Brendan O'Donoghue for their discussions and help on this problem with us.

References

[1] Boyd, S. P. (2009). Linear Matrix Inequalities and the S-Procedure, *EE363 Notes, Winter 2009.*
[2] We feel fine, "http://wefeelfine.org/".