Bayesian Possibilistic C-Means Clustering

https://solour-lfq.github.io/

Contents

1 Introduction **4**

2 Review of Classical Model-Based Clustering Algorithms **8**

2.1 Hard C-Means 8

2.2 Fuzzy C-Means 9

2.3 Possibilistic C-Means 10

2.4 Gaussian Mixture Model 11

2.5 Summary of Mentioned Models 12

3 Bayesian Possibilistic C-Means Clustering **13**

3.1 Proposed Model 13

3.2 Degenerated Cases 17

3.3 Analytic Properties (Partial) 18

4 Illustrations and Experimentations **23**

4.1 Butterfly Illustration: E-step, a Study of Robustness 23

4.2 Butterfly Illustration: M-step, a Study of Stability 26

4.3 Typical Mixture Model Data 28

5 Conclusion **31**

5.1 Conclusion 31

5.2 Works to be Done 31

6 Reference(unassorted) **32**

Abstract

A clustering algorithm as a Bayesian generalization of possibilistic c-means clustering is proposed. By taking a Bayesian approach the overfitting problem and the fuzzy extra constraints are eliminated. This method is motivated by the Gaussian mixture model's design ideas. We prove some desired properties of the proposed method. This properties are further supported by many illustrations on some synthetic datasets.

1 Introduction

As one the the problems with the longest history of being studied, clustering has gone through consistent research of years as long as the history of its hypnernym, unsupervised learning itself's. The popularity of clustering stems from its nature as an optimization problem with various possible formulations and its broad applications in physical contexts. As a typical example to start an introduction to unsupervised learning, clustering represents a primitive approach to recognise pattern from scratch, which is ubiquitous throughout closely all branches in engineering as well as science.

Studies of clustering have bloomed into numerous classes. The variation between different classes derives from the set of hypotheses they make as well as the modeling tools they perfer. This variation is not likely to be eliminated taking considertaion of the variety of clustering datasets and the loosely statement of the problem itself. At least four mainstream classes of clustering algorithms have been widely applied in engineering: model-based, density-based, graph-based and kernel-based. Believing in the appearance of new classes, these four classical ones have went through validations of time as well as data. And they form the foundation of almost all later improvements.

Model-based clustering is a generative algorithm, it assumes that the local pattern, i.e.a subset of the whole dataset is induced by some model. Then the mixture, or the superposition of all local patterns with label dispelled form the observed dataset. The model assumed at a local pattern is usually a constraint on the expectation of some measure:

$$\mathbb{E}[f(\mathbf{x})] \leq c$$

Where $\mathbf{x}$ represents the data entry and c is a local constant, the expectation is taken with respect to a local distribution. Under this modeling, the local distribution is necessarily an exponential family distribution and the whole dataset is a mixture of exponential family distributions. This conclusion require f to be a convex function of $\mathbf{x}$. Although it neatly demonstrate an intuitively generation process of unlabelled data with latent pattern, its tuning of parameters remains uneasy. The origin of this uneasiness is that the parameters of the local distributions and the parameters that encode the membership of data to different locality, i.e.clusters demand conjugate optimization. This conjugation results in a non-diagonal Hessian which rejects direct optimization with unexamined convexity. However, previous studies have come up with various approaches toward this problem and have obtained fruitful outcomes. This formulation made model-based clustering known also as cost-minimization algorithms. We will go through some crucial examples in the next section.

The elegance of the generative breaks when the data appear to be far uglier than what they are assumed. Therefore people began to develop discriminative methods that no longer require modelling local pattern.

Graph-based clustering is a discriminative model that handles the clustering problem as a graph problem. Aftering representing the dataset as a similarity matrix $\mathbf{W}$, where $\mathbf{W}_{ij}$ represents the similarity between the ith and the jth data entries, it seeks a min-cut of the graph described by $\mathbf{W}$. Algebraically, this task is done by compactly minimize:

$$\mathrm{Tr}\left\{\mathbf{H}^{\mathrm{T}}\mathbf{L}\mathbf{H}\right\}$$

Where $\mathbf{L}$ is $\mathbf{W}$'s Laplacian and $\mathbf{H}$ is a normalized matrix that encoded the potential assignment of a data entry as a cluster. Then the clustering task is transformed into an integer programming, i.e, a combinatorial optimization problem. This is usually solved by dropping some constrains and applying eigen-formulations.

Dropping constrains and convert the original combinatorial optimization into an easier convex optimization is a practical trick in graph-based as well as model-based clustering. Since both of them demand a representation of the partition of data into clusters in an orthogonal matrix, where the encoding of one data entry is usually one-hot. It is of mathematical convenience and of philosophical significance to apply a ”fuzzy” version by replace the one-hot constraint. We will come to this topic again in the next section.

Taking generative method as a top-to-bottom approach, then the density-based clustering and kernel-based clustering are unambiguous bottom-to-top approaches. Compared with graph-based methods, they two abandon the very concept of partition representation and parameters(not hyperparameter, through.)

Density-based clustering forms a cluster by ensembling observed data, in its context, a cluster is like a Doodle Snake that consumes one new adjacent data in each iteration. If it finds all data outside are too far to fetch, it stops growing as well as iteration. The adjacency between an entry and a cluster can be measured by the distance between the entry and the cluster's center of mass or the minimal distance between the entry and an entry inside the cluster. There are also methods to regularize the growing speed of cluster to avoid pathology. Typical density-based clustering algorithms include DBSCAN and its variants.

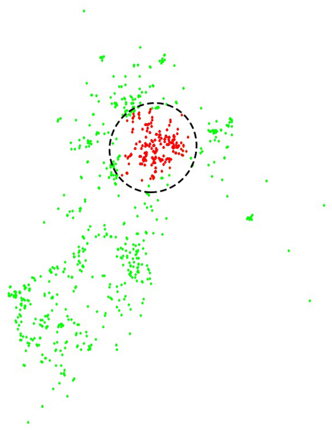


Figure 1: Illustration of Density-Based Spatial Clustering of Applications with Noise(DBSCAN), a density-based clustering algorithm.

Kernel-based clustering is a newly developed method, instead of clustering by fusing entries, it seeks an envelope that contains clusters. This is done by mapping the dataset into another space, minimizing the radius of the sphere that contains all mapped entries in this space(w.r.t the hyperparameters that determined the mapping), and mapping the sphere back into the data space. By taking advantage of the kernel representative power, kernel-based clustering algorithms can discover highly non-linear boundaries in data space.

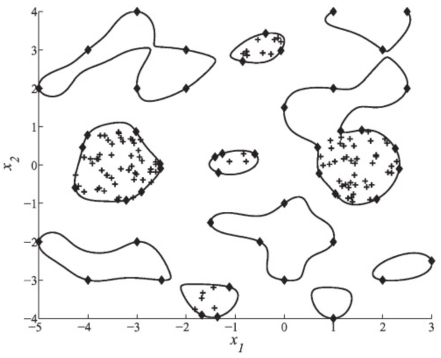


Figure 2: Illustration of support vector clustering(SVC), a kernel-based clustering algorithm.

The newly-developed algorithms combined with kernel skills or neural networks have showed promising outcome in complex datasets, which should not surprise too much since the tools combined are designed to outperform traditional formulations in dealing with non-linear relationships. However, the classical models still have privileges for their profound analytic properties and illustrative significance.

The rest of this papaer organized as follows: Section 2 briefly reviews some representative algorithms in model-based clustering, spefically Fuzzy C-Means(FCM), Possibilistic C-Means(PCM) and Gaussian Mixture Model(GMM) and elaborate the similarities and relationships within these models. Section 3 proposes a hypernym, i.e.a generalized model that covers those typical model-based algorithms as special instances. Section IV derives some further illustrative cases and shows some experiments on datasets. Section V concludes the previous works.

2 Review of Classical Model-Based Clustering Algorithms

In this section we review some classical model-based clustering algorithms, namely FCM, PCM and GMM. In all three context, we denote the dataset by $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^{N}$, $\mathbf{x} \in \mathcal{X}$. Denote the membership by matrix $\mathbf{U}$, where $\mathbf{U}_{i,j}$ represents the membership of $\mathbf{x}_i$ to the jth cluster. And the centers of C clusters are $\mathbf{M} = \{\mathbf{m}_j\}_{j=1}^{C}$.

The task of clustering is to find $\mathbf{U}$ and $\mathbf{M}$ given $\mathbf{X}$, a (parameterized) measure $d : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}^{+}$, and C. Studies have showed that C can be somehow inferred from $\mathbf{X}$, which requires an extra evidence framework, of which we temporarily skipped.

Finding $\mathbf{U}$ and $\mathbf{M}$ is uniformly done by minimizing a loss function with they as differentiable parameters:

$$\mathcal{L}(\mathbf{X}, \mathbf{M}, \mathbf{U}) = \sum_{i=1}^{N} \sum_{j=1}^{C} u_{i,j} \cdot d(\mathbf{x}_i, \mathbf{m}_j) \tag{1}$$

The partial derivative of $\mathcal{L}$ w.r.t $\mathbf{U}$ or $\mathbf{M}$ involves each other, which makes the global minima intractable. The alternative approach is to optimize two parameters iteratively, when $\mathbf{U}$ is optimized, $\mathbf{M}$ is taken for granted, vice versa.

The variety of model-based clustering algorithms can be attributed to the difference in: the constraint exerted on $\mathbf{U}$ and the detailed optimization techniques.

2.1 Hard C-Means

HCM assumes $\mathbf{U}$'s rows to be a one-hot vector, i.e:

$$\forall i = 1, 2, ..., N\text{: } u_{i,j} = 0 \text{ or } 1, \text{ and } \sum_{j=1}^{C} u_{i,j} = 1$$

In optimization, HCM looks for local optima at each step. To optimize $\mathbf{U}$ w.r.t. fixed

M:

$$\forall i = 1, 2, ..., N\colon\ u_{i,j} = 1 \text{ for } j' = 1, 2, ..., C, j' \neq j, d(\mathbf{x}_i, \mathbf{m}_j) \leq d(\mathbf{x}_i, \mathbf{m}_{j'}) \tag{2}$$

To optimize **M** w.r.t. fixed **U**, $\mathbf{m}_j$ is set so that:

$$\forall j = 1, 2, ..., C\colon\ \sum_{i=1}^{N} u_{i,j} \cdot d'(\mathbf{x}_i, \mathbf{m}_j) = 0 \tag{3}$$

Both formula can be obtained from finding partial derivative of (1), it should be noticed that both step is locally optimized if we assume that d is a convex function w.r.t $\mathbf{m}$. Semantically, (2) states that we should assign an entry to the cluster whose centroid is closest with all confidence, and (3) states that the centroid of a cluster is some sort of average of entries assigned to it. HCM repeats (2) and (3) until some convergence condition is met. The convergence is obvious since during each iteration, $\mathcal{L}$ is reduced, yet we must have $\mathcal{L} \geq 0$, i.e, $\mathcal{L}$ is the Lyapunov function of HCM that guarentees its dynamics converge.

2.2 Fuzzy C-Means

The ordinary formulation of FCM is:

$$\mathcal{L}(\mathbf{X}, \mathbf{W}, \mathbf{M}) = \sum_{i=1}^{N} \sum_{j=1}^{C} (w_{i,j})^m \cdot d(\mathbf{x}_i, \mathbf{m}_j),\ m \geq 1$$

s.t.

$$\forall i, j, w_{i,j} \geq 0, \text{ and } \sum_{j=1}^{C} w_{i,j} = 1$$

The origin of FCM is a report that by squaring the weight, the clustering procedure yields a better result, but the physical significance of squaring a weight factor is intractable.

But it can be easily altered into (1) s.t.

$$\forall i, j, u_{i,j} \geq 0, \text{ and } \sum_{j=1}^{C} (u_{i,j})^p = 1, 0 \leq p \leq 1$$

So that $u_{i,j} = (w_{i,j})^m$, $p = \frac{1}{m}$. Now the seemlingly arbitrary exponential m is altered as a norm constraint, which should be more comfortable. Moreover, using the perspective of norm constraint helps to interpret the robustness of FCM.

Taking a special case where $C = 2, p_1 = \frac{1}{2}, p_2 = 1$, Figure 3 plots the possible weight vector for an entry $(w_{i,1}, w_{i,2})$ for p_1(orange line) and p_2(blue line). The remarkable issues is at the ambiguous cases, for p_2, $(\frac{1}{2}, \frac{1}{2})$ has weight $\frac{1}{2}$ for both centroids, but for p_1, the influence is cut in half. In other words, we expect FCM to have some degree of robustness when increasing m. However, let $m \to \infty$ $(p \to 0)$ reduces FCM to HCM, which is a pathology.

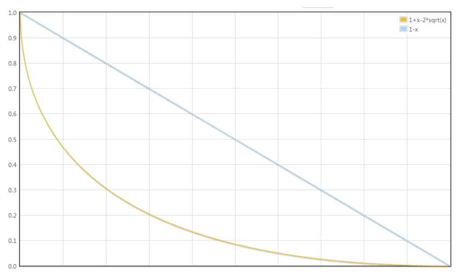


Figure 3: Illustration of robustness under norn constraint

The optimization of parameters in FCM is again straightforward by applying Lagrangian multiplier. The condition $m > 1$ guarentees that local optima is feasible.

[E and M step for FCM omitted here.]

2.3 Possibilistic C-Means

Although the motivations of PCM has failed after our formulation in the previous subsection, we briefly restated them for reference. The presenters of PCM claimed that the weight vector (**W**'s row) for FCM is not representative enough since we might want a vector as

$(0.25, 0.25)$ instead of $(0.5, 0.5)$. So they suggest a loss function:

$$\mathcal{L}(\mathbf{X}, \mathbf{W}, \mathbf{M}) = \sum_{i=1}^{N}\sum_{j=1}^{C}(w_{i,j})^m \cdot d(\mathbf{x}_i, \mathbf{m}_j) + \sum_{j=1}^{C}\eta_j \sum_{i=1}^{N}(1 - w_{i,j})^m \tag{4}$$

The extra term on the r.h.s of (4) is a regularizer to avoid the weight vector being absorbed by the null case $(0, 0, ..., 0)$, which is obviously a minimizer of (1). The optimization procedure is straightforward since η_j is predetermined. The only difference is the normalization constraint on $\mathbf{w}$ is dropped, the innate additional regularizer roles as the multiplzer. The optimization w.r.t $\mathbf{M}$ remains the same as in FCM.

Using our formulation, this is tantamount to replacing the norm constraint:

$$\forall j = 1, 2, ..., C, \ \sum_{i=1}^{N}(u_{i,j})^p = 1$$

by

$$\forall j = 1, 2, ..., C, \ \sum_{i=1}^{N}\{1 - (u_{i,j})^p\}^{\frac{1}{p}} = 1$$

Semantically, this constraint does encode the intuitive conditions: null weight vector should be rejected. After all, PCM is still an variant of (1) with an extra constraint, plus a two-staged iterative optimization procedure.

2.4 Gaussian Mixture Model

The philosophy behind GMM is to model the conditional distribution of $\mathbf{x}$ w.r.t given $\mathbf{M}$ and $\mathbf{u}$:

$$p(\mathbf{x}|\mathbf{M}, \mathbf{u}) \propto \prod_{j=1}^{C} \exp\{-d(\mathbf{x}, \mathbf{m}_j)\}^{u_j}$$

Where we deliberately ignore the mixture coefficient π and the covariance matrix Σ for conciseness. In GMM the weight vector is constrained to be one-hot code. However, In optimizing w.r.t $\mathbf{M}$, the one-hot vectors are taken expectation and converted into simplex vectors. The expectation of the one-hot vector is innately normalized since its jth component equals $p(\mathbf{u} = (0, 0, ..., 1, ...0)|\mathbf{x})$,with the jth component of $\mathbf{u}$ be 1, and the latent space consistes of all C one-hot vectors.

[E and M step for GMM omitted here.]

2.5 Summary of Mentioned Models

We summary the mentioned models in the Table 1, the significant aspects are how they model the constraint, or the properties of the latent weight space, plus the technique they used in optimization.

Table 1: A summary of mentioned models.

	Weight space	E-step	M-step
HCM	One-hot	Local optima	Local optima
FCM	Normed space	Local optima	Local optima
PCM	Regularized space	Local optima	Local optima
GMM	One-hot	Distributed estimation	Local optima

It has been reported that HKM and FCM has the problem of biased weight space, PCM sufferer from overfitting of initialization parameters, while GMM is somehow sound, it rejects a fuzzy interpretation. Naturally, we try to find a compromise that combines GMM's robustness and fuzzy philosophy. Observing Table 1, it is intuitively to try a clustering algorithm that has a latent weight space more representative than one-hot, while using a distributed estimation instead of a point one in optimization.

3 Bayesian Possibilistic C-Means Clustering

Follow the context we set at the beginning of the previous section, we firstly propose the improved model that answering the problem raised at the end of Section 2. Then we show the proposed model does have some desired properties.

3.1 Proposed Model

Assumed that we are given $\mathbf{X}$, C and the latent space $\mathcal{U}$, the model has parameters $\mathbf{U}$ and $\mathbf{M}$, we show that the target loss function (1) is tantamount to maximum likelihood estimation under a more general $\mathcal{U}$ than one-hot space. First of all, in the broadest sense of fuzziness, $\mathcal{U} = [0,1]^C$, we assume:

$$p(\mathbf{x}_i|\mathbf{u}_i, \mathbf{M}) = \frac{1}{c} \cdot \prod_{j=1}^{C} \exp\left\{-d(\mathbf{x}_i, \mathbf{m}_j)\right\}^{u_{i,j}}, \; 0 \leq u_{i,j} \leq 1 \tag{5}$$

$$c = \int p(\mathbf{x}|\mathbf{u}, \mathbf{M}) \mathrm{d}\mathbf{x} \tag{6}$$

This reduce to the ordinary case in GMM since one-hot space is a subset of $[0,1]^C$. Equation (5) is a non-trivial generalization, when $\mathbf{u} = \mathbf{0}$, we have

$$\forall \mathbf{x}_1,\, \mathbf{x}_2 \in \mathcal{X}, p(\mathbf{x}_1|\mathbf{u}, \mathbf{M}) = p(\mathbf{x}_1|\mathbf{u}, \mathbf{M})$$

It says by saying an entry belongs to no cluster contributes nothing for the total likelihood. When $\mathbf{u}$ is a one-hot vector, c is necessarily the normalization coefficient of the only effective distribution. In other cases, (6)'s integral needs sampling methods to compute.

For the maximum likelihood estimation of the dataset $p(\mathbf{X}|\mathbf{U}, \mathbf{M})$, we take the negative logarithm as the loss function:

$$\mathcal{L} = -\ln \prod_{i=1}^{N} p(\mathbf{x}_i|\mathbf{u}_i, \mathbf{M}) = \sum_{i=1}^{N} \sum_{j=1}^{C} u_{i,j} d(\mathbf{x}_i, \mathbf{m}_j)$$

Which is no more than (1). However, we do not directly optimize w.r.t $\mathbf{U}$ for two reasons:

1. The constraint of the latent weight space might reject introducing traditional optimization procedures.

2. We seek a fuzzy estimation instead of a point estimation, which can not be offered by introducing multipliers.

For the fuzzy, or possibilistic estimation of the weight vector, we repeat the procedure of a general EM framework. We will later concluded that the model we proposed is a Bayesian version of PCM since we treat $\mathbf{U}$ as latent variables in probabilistic models and apply a Bayesian approach to integrate it out, leaving $p(\mathbf{X}|\mathbf{M})$ to be studied.

With out loss of generality, in a general EM framework, we always have:

$$\begin{aligned}
\ln p(\mathbf{X}|\mathbf{M}) =& \ln \left\{ \frac{p(\mathbf{X}, \mathbf{U}|\mathbf{M})}{q(\mathbf{U})} \frac{q(\mathbf{U})}{p(\mathbf{U}|\mathbf{X}, \mathbf{M})} \right\} \\
=& \sum_{\mathbf{U}} q(\mathbf{U}) \ln \left\{ \frac{p(\mathbf{X}, \mathbf{U}|\mathbf{M})}{q(\mathbf{U})} \frac{q(\mathbf{U})}{p(\mathbf{U}|\mathbf{X}, \mathbf{M})} \right\} \\
=& \sum_{\mathbf{U}} q(\mathbf{U}) \ln \left\{ \frac{p(\mathbf{X}, \mathbf{U}|\mathbf{M})}{q(\mathbf{U})} \right\} + \sum_{\mathbf{U}} q(\mathbf{U}) \ln \left\{ \frac{q(\mathbf{U})}{p(\mathbf{U}|\mathbf{X}, \mathbf{M})} \right\} \\
=& \mathcal{L}(q(\mathbf{U}), \mathbf{M}) + \mathbb{KL}(q||p)
\end{aligned} \tag{7}$$

Where $q(\mathbf{U})$ is an arbitrary distribution on $\mathcal{U}$. To optimize $\ln p(\mathbf{X}|\mathbf{M})$, we firstly set $q = p$, then optimize $\mathbf{M}$ with fixed q. In GMM, these are called E-step and M-step, things are analogic in HCM, FCM or PCM with finding a distribution on $\mathbf{U}$ replaced by finding a local optima of $\mathbf{U}$.

To find the posterior distribution $p(\mathbf{U}|\mathbf{X}, \mathbf{M})$, we making use of the assumption made at the beginning of this section:

$$\begin{aligned} p(\mathbf{u}_i|\mathbf{x}_i,\mathbf{M}) =& \frac{p(\mathbf{x}_i,\mathbf{u}_i|\mathbf{M})}{p(\mathbf{x}_i|\mathbf{M})} \\ =& \frac{p(\mathbf{x}_i|\mathbf{u}_i,\mathbf{M})p(\mathbf{u}_i|\mathbf{M})}{p(\mathbf{x}_i|\mathbf{M})} \\ =& \frac{\frac{1}{c}\prod_{j=1}^{C}\exp\{-d(\mathbf{x}_i,\mathbf{m}_j)\}^{u_{i,j}}\cdot p(\mathbf{u}_i)}{\int\frac{1}{c}\prod_{j=1}^{C}\exp\{-d(\mathbf{x}_i,\mathbf{m}_j)\}^{u_{i,j}}\cdot p(\mathbf{u}_i)\mathrm{d}\mathbf{u}_i} \end{aligned} \tag{8}$$

The major computation here is that c is depent on $\mathbf{u}_i$ as shown in (6). And the integral on the denominator of (8) requires another sampling process. But this sampling w.r.t different possible $\mathbf{u}_i$ needs to be done any way since to compute the loss function in (7), we need to compute an amount's expectation w.r.t $q = p$, after all. The $p(\mathbf{u}_i)$ in (8) encodes the constraints exerted onto the weight space, we can modify this prior distribution to select a subspace of $[0,1]^C$ as area of interest. To derive GMM from our proposal, we simply concentrate all possibility mass to one-hot subspaces and we are done.

After finishing computing $q(\mathbf{U})$ by sampling in $\mathcal{X}$ as well as $\mathcal{U}$, the term $\mathcal{L}(q(\mathbf{U}),\mathbf{M})$ requires maximization, with fixed q:

$$\begin{aligned} \mathcal{L}(q(\mathbf{U}),\mathbf{M}) =& H(q) + \mathbb{E}_q[\ln p(\mathbf{X},\mathbf{U}|\mathbf{M})] \\ =& \left\{-\sum_{i=1}^{N}\mathbb{E}_{q(\mathbf{u}_i)}[\sum_{j=1}^{C}u_{i,j}\cdot d(\mathbf{x}_i,\mathbf{m}_j)]\right\} + c \end{aligned} \tag{9}$$

Terms independent of $\mathbf{M}$ are collected into c.

After sampling procedure, (9) becomes a linear combination of all possible $d(\mathbf{x}_i,\mathbf{m}_j)$, which left the optimization w.r.t $\mathbf{M}$ trivial algebra.

The preceding computation is summaried below in Algo.(1):

Algo.(1) looks very much like an optimization in PCM/FCM/GMM, with slight modification. The two fundamental difference are:

1. Using a general space constraint $\mathcal{U}$ and $p(\mathbf{u})$ to allow more representation ability, so that we can incorporate fuzzy vectors naturally.

2. Using a distributed estimation instead of a point estimation in evaluating the weight vector $\mathbf{U}$, the motivation behind this approach is to counteract the overfitting and oversensitivity of point estimation-based approaches, i.e,FCM, PCM, HCM. To do so, a probabilistic framework together with a sampling method has to be applied. Since we no longer work on a simplex subspace, we have non-trivial normalizers than call for update as well.

Much effort is saved if we fixed a sampled collection from $\mathcal{U}$ and $\mathcal{X}$ throughout the computation, believing the law of large numbers (using average to approximate the expectation) does not fail us.

Algorithm 1 Bayesian Possibilistic C-Means Clustering

```
Input: $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^N \in \mathcal{X}^N$, $C$, $\mathcal{U}$, $p(\mathbf{u})$, $\epsilon$
Output: $p(\mathbf{U})$ and $\mathbf{M}$
 1: Sample a sequence of $\{\mathbf{u}_k\}_{k=1}^K$ according to $p(\mathcal{U})$ in $\mathcal{U} \in [0,1]^C$;
 2: Sample a sequence of $\{\mathbf{x}_l\}_{l=1}^L$ in $\mathcal{X}$;
 3: Random initialize $\mathbf{M}^{(0)}$, $t = 0$;
 4: while NOT ($t \geq 1$ AND $||\mathbf{M}^{(t)} - \mathbf{M}^{(t-1)}|| \leq \epsilon$) do
 5:     for all $\mathbf{x}_i$ in $\mathbf{X}$ do
 6:         Compute the posterior $p(\mathbf{u}_i|\mathbf{x}_i, \mathbf{M}^{(t)})$ by computing it on $\{\mathbf{u}_k\}_{k=1}^K$,
            during which $c(\mathbf{u}_i)$ has to be clarified for each $k$,
            which can be obtained by computing an average on $\{\mathbf{x}_l\}_{l=1}^L \cup \mathbf{X}$
            //Finish computing the amount in (8).
 7:         Take coefficient of all $N \cdot C$ possible $d(\mathbf{x}_i, \mathbf{m}_j)$,
            do a linear optimization w.r.t. $\mathbf{M}$, save as $\mathbf{M}^{(t+1)}$
            //Finish computing the amount in (9).
 8:         t++;
 9:     end for
10: end while
```

Remark: We are actually generalize the EM in a broad sense as a clustering procedure by using two sampling process to compute the integrals. In essence we maximize the likelihood $p(\mathbf{X}|\mathbf{M})$ with $\mathbf{U}$ eliminated during integral. This process represent a Bayesian treatment of latent weight labels, what makes it different from GMM is that we apply the integral in a broader space under the enlightment of fuzzy mathematics. So we name this algorithm after

Bayesian Possibilistic C-Means Clustering to represent covered elements.

3.2 Degenerated Cases

To show the model we proposed merge FCM/PCM and GMM into a union, we show how it degenerated into them by taking specific parameters. Its degeneration to GMM has been briefly introduced in the previous subsection, here we have a formal repetation:

To implement GMM using BPCMC, some modifications to Algo.(1) have to be made:

1. Fix $\mathcal{U}$ to be $[0,1]^C$'s simplex subset, or hold $\mathcal{U} = [0,1]^C$ while altering $p(\mathcal{U})$ such that all possibility mass are concentrated on vertices.

2. No modification in E or M step has to be done, since in this subset the normalizer is necessarily 1.

To implement FCM, PCM, or HCM using BPCMC, we analogically render analyse the two aspect above:

1. Alter $\mathcal{U}$ or $p(\mathbf{u})$ as indicated in Table.1.

2. Hope that the Bayesian approach finally peaks all possibility mass onto its model, which is only possible in idealistic case.

It is possible to arbitrarily modify Equation.(8) and force it to output its maximum. But by doing so we lose the robustness and the counteract against overfitting and stability in a Bayesian approach. The belief behind this modification is that a Bayesian model can outperform a non-Bayesian/point-estimation-based model in stability, though relatively higher in computation cost. After all, if some complex constraints are exerted onto $\mathcal{U}$, a local optima's positioning will be painstaking as well.

3.3 Analytic Properties (Partial)

In this subsection we derive some properties of proposed model to show that some desired conditions have be met.

Property One: When a data entry $\mathbf{x}_i$ is close enough to a cluster centroid $\mathbf{m}_j$ rather than $\mathbf{m}_{j'}$, $j' \neq j$ then the weight assigned to $\mathbf{u}_i$ is higher than that to $\mathbf{u}_i^{'}$, where $\mathbf{u}_i$ and $\mathbf{u}_i^{'}$ share the same values on components corresponding to $j' \neq j$ and $u_{i,j} > u'_{i,j}$.

To be illustrative, we demonstrate a simplified case where $C = 2$, thus $\mathcal{U}$ is a subset of $[0,1]^2$, we demonstrate it in Figure.4:

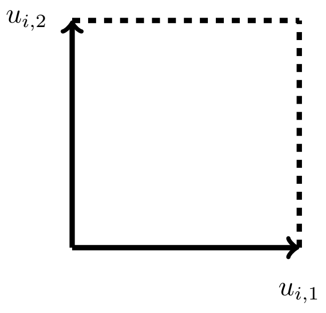


Figure 4: The universe contains $\mathcal{U}$ when $C = 2$.

While the precise picking of $\mathcal{U}$, or tantamount $\mathcal{U}$ has numerous possibilities, we list some past examples:

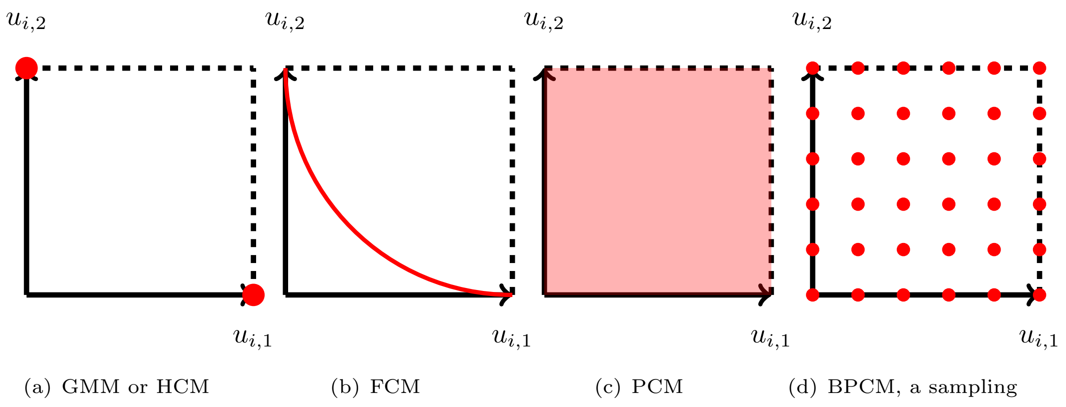


Figure 5: $\mathcal{U}$ introduced in different models, red marks the valid $\mathcal{U}$.

BPCM's $\mathcal{U}$ is as powerful as that of PCM, which offers fuzzy interpretation. The weight assigned to a certain $\mathbf{u}_i$ is:

$$p(\mathbf{u}_i|\mathbf{x}_i) = \frac{p(\mathbf{x}_i|\mathbf{u}_i)p(\mathbf{u}_i)}{p(\mathbf{x}_i)} \propto p(\mathbf{x}_i|\mathbf{u}_i)$$

Where we have dropped $\mathbf{M}$ from the condition w.l.o.g, also used the propostion of sampling.

To evaluate $p(\mathbf{x}_i|\mathbf{u}_i)$, recall Equation.(5), but we now make the dependency of c on $\mathbf{u}_i$ explicit:

$$p(\mathbf{x}_i|\mathbf{u}_i) = \frac{1}{c(\mathbf{u}_i)} \prod_{j''=1}^{C} \exp\{-d(\mathbf{x}_i, \mathbf{m}_{j''})\}^{u_{i,j''}}$$

Where:

$$c(\mathbf{u}_i) = \int \prod_{j''=1}^{C} \exp\{-d(\mathbf{x}_i, \mathbf{m}_{j''})\}^{u_{i,j''}} \,\mathrm{d}\mathbf{x}_i$$

Plug in $\mathbf{u}_i$ and $\mathbf{u}_i^{'}$ as assumed in the statement of Property One, we have:

$$\forall \mathbf{x}_i \in \mathcal{X},\ \exp\{-d(\mathbf{x}_i, \mathbf{m}_j)\}^{u_{i,j}} \le \exp\{-d(\mathbf{x}_i, \mathbf{m}_j)\}^{u_{i',j}}$$

Which implies:

$$c(\mathbf{u}_i) \le c(\mathbf{u}_i^{'})$$

And we also have:

$$\frac{p(\mathbf{x}_i|\mathbf{u}_i)}{p(\mathbf{x}_i|\mathbf{u}_i^{'})} = \frac{c(\mathbf{u}_i')}{c(\mathbf{u}_i)} \cdot \exp\{-d(\mathbf{x}_i, \mathbf{m}_j)\}^{\epsilon}$$

Where $\epsilon = u_{i,j} - u_{i',j} > 0$.

Formally, Property One states that:

Property One Restated: For every $\epsilon > 0$, $\mathbf{u}_i = \mathbf{u}_i^{'} + \epsilon \cdot \mathbf{1}_j$, there exists an δ, such $d(\mathbf{x}_i, \mathbf{m}_j) \le \delta$ implies $p(\mathbf{x}_i|\mathbf{u}_i) \ge p(\mathbf{x}_i|\mathbf{u}_i^{'})$. $\mathbf{1}_j$ denotes a one-hot vector with the jth component non-zero.

And we see it easily satisfied by let:

$$\delta = \frac{1}{\epsilon} \ln \frac{c(\mathbf{u}_i')}{c(\mathbf{u}_i)}$$

In other words, Property One says that the distribution on $\mathcal{U}$ introduced by a single $\mathbf{x}_i$ will have a local peak at components corresponding to a centroid close to $\mathbf{x}_i$. This statement is of significance since it counteracts the problem that confronted the founders of PCM: the null assignment $\mathbf{u} = \mathbf{0}$ always tends to minimize the loss function. By taking a Bayesian procedure, this overfitting is naturally evaded.

Property Two: BPCMC provides a measure of noise/substantial entry as well as the fuzziness of a dataset.

First of all, we have a "fuzzy" definition as well as a "probabilistic" definition of exception or noise:

1. (Probabilistic version) It is relatively straightforward to assign a degree of reliability to a data entry $\mathbf{x}_i$ once we compute $p(\mathbf{x}_i|\mathbf{M})$. After iterations terminate and we have a final $\mathbf{M}_f$, we can compute $p(\mathbf{x}_i|\mathbf{M})$ for each $\mathbf{x}_i$, those with a relatively low probability can be seen as exceptions or noise.

2. (Fuzzy version) Apart from $p(\mathbf{x}_i|\mathbf{M})$ where we have $\mathbf{u}_i$ integrated out, it is also appliable to observe the weight assigned to the null-adjacent part in of $\mathbf{u}_i$ in $\mathbf{U}$. For example, we may find an entry $\mathbf{x}_i$ whose $p(\mathbf{x}_i|\mathbf{M})$'s contribution is given mainly by $\mathbf{u}_i \approx \mathbf{0}$. Then we can classify such an entry as trivial entry intuitively as well.

To combine the previous two intuitive definition of a noised or exceptional entry, we show that if an entry $\mathbf{x}_i$ is relatively far from every centroid in $\mathbf{M}$, then its Bayesian belief $p(\mathbf{x}_i|\mathbf{M})$ and null-contribution $\frac{1}{p(\mathbf{0}|\mathbf{x}_i)}$ are both relatively low.

For formal convenience, assume another $\mathbf{x}_i^{'}$ that is close only to $\mathbf{m}_1$ w.l.o.g.

From $p(\mathbf{u}_i = \mathbf{0}|\mathbf{x}) = \frac{p(\mathbf{u}_i=\mathbf{0})\cdot p(\mathbf{x}|\mathbf{u}_i=\mathbf{0})}{p(\mathbf{x})}$, we see the two types of definition of noise imply each other, we only have to show one situation, and it is easy to see:

$$\begin{aligned} p(\mathbf{x}_i) &= \int p(\mathbf{x}_i|\mathbf{u}_i)p(\mathbf{u}_i)\mathrm{d}\mathbf{u}_i \\ &\leq \int p(\mathbf{x}_i|\mathbf{u}_i)p(\mathbf{u}_i)\mathrm{d}\mathbf{u}_i \\ &= p(\mathbf{x}_i') \end{aligned}$$

The inequality holds since we assume both $textbfx_i$ and $\mathbf{x}_i^{'}$ are relatively far from $\mathbf{m}_2$ to $\mathbf{m}_3$, thus the terms involve $j = 2, 3, ..., C$ in the product makes no significance. For all $\mathbf{u}_i$, consider only the term involve $\mathbf{m}_1$ yields this relationship.

In other words, we can use the definition of exception or noise as an entry with a low generative probability as well as an entry with an assignment vector close to null, in this context two definitions are unified.

A simplified illustration in Figure.6 might help to support the conclusion drawn.

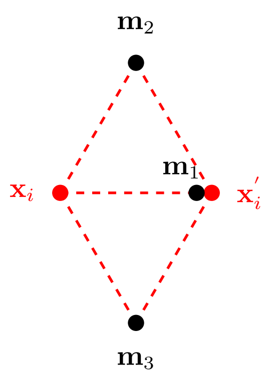


Figure 6: The universe contains $\mathcal{U}$ when $C = 2$.

Property Three:(not examined yet) It is possible to solve the problem of determing the number of clusters in the context of BPCMC. In semantics, reducing the number of clusters equals reducing the universe of weight vectors from $[0, 1]^{C_1}$ to $[0, 1]^{C_2}$ where $C_1 > C_2$.

Two possible ways to check whether a reduce from C_1 to C_2 is good are:

1. An evidence benchmark: we mark C as a hyperparameter, since computing $p(\mathbf{X}|C)$ (recalling that $\mathbf{M}$ has been dropped w.l.o.g.) is straightforward. We then compare $p(\mathbf{X}|C_1)$ and $p(\mathbf{X}|C_2)$.

2. We check a principal component analysis(PCA) to the $\mathbf{U}$, and see whether the first C_2 eigenvalues of the covariance matrix of $\mathbf{U}$ are significant enough. The point here is that we draw a connection between the determination of number of clusters in a clustering problem and the lower rank approximation of a matrix. But the physical significance of the principal vectors remains ambiguous.

4 Illustrations and Experimentations

4.1 Butterfly Illustration: E-step, a Study of Robustness

The butterfly case is a commonly used one to illustrate some basic properties of a clustering algorithm, with original unlablelled data organized as wings of a butterfly as shown in Figure.7, with two centroids initialized as the red points:

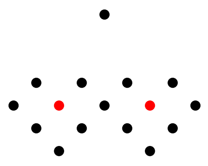


Figure 7: Butterfly data.

For further illustration we index the data as Figure.8:

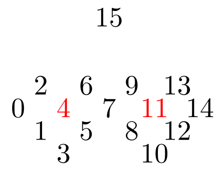
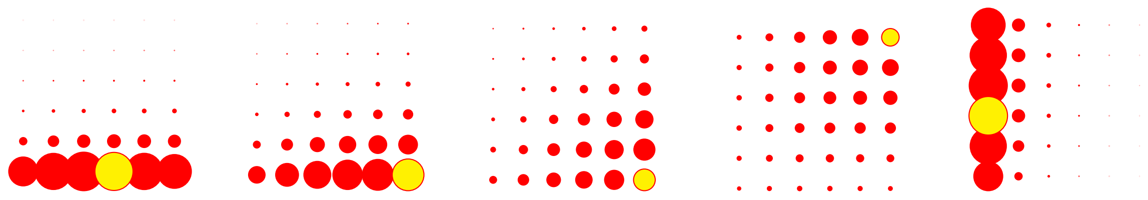


Figure 8: Indexed butterfly data.

Now we apply the E-step counterpart in BPCMC as introduced in the previous section, taking the sampling in $\mathcal{U}$ as a grid with step = 0.2, and $l = 0$. We illustrate the state of weight space corresponding to some data in Figure.9, the radius of the red circle shows the value of the corresponding $\mathbf{u}$ and we mark the maxima grid index with yellow.



(a) $i = 0$ (b) $i = 1$ (c) $i = 4$ (d) $i = 7$ (e) $i = 14$

Figure 9: Weight assigned to sampled $\mathcal{U}$ for some data in the butterfly dataset.

It is straightforward to read some intuitive conclusions from Figure.9:

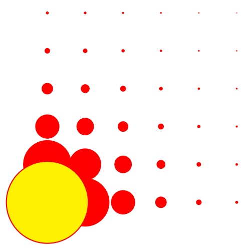


Figure 10: $i = 15$

Take 9.(a) for an instance. The picture indicates that BPCMC prefers to assign $\mathbf{x}_0$ to vectors as $(u_1, 0)$, i.e, $\mathbf{x}_0$ is not likely to be classified into the second cluster with centroid at $\mathbf{x}_{11}$.

But compare (a) with (b), we see that although $\mathbf{x}_0$ and $\mathbf{x}_1$ are likely to be the first cluster's member, we do not give $\mathbf{x}_0$ a full credit, so its maxima is on $(0, 0, 6)$ rather than $\mathbf{x}_1$'s $(0, 1)$, vice versa for (e).

Even for $\mathbf{x}_4$ which overlaps with the former centroid, (c) shows the mass is not concentrated, this distribution can be taken as a distribution estimation of a point estimation peaked at $(1, 0)$.

For ambiguous cases where data lies with distances to $\mathbf{m}_1 = \mathbf{x}_4$ and $\mathbf{m}_2 = \mathbf{x}_{11}$ equal, we have intuitively two cases: either it is where two models overlap, and the data should be assigned to $(1, 1)$, or the data is a noise of no significance that should be assigned to $(0, 0)$, we have (d) illustrate the first case. For the second one, we run BPCMC's E-step on the 15th data, then Figure.10 shows that these two cases can be clarified automatically. The first case includes data that is close to both centroids, while the second one includes those who are far from both centroids.

Comparison with other models: We list the privileges of BPCMC over other mentioned mainstream models below.

The advantage over HCM is avoiding overfitting, and in many situations a soft or fuzzy assignment is better since it provides uncertainty and a bridge to Bayesian decision.

The advantage over FCM is that the proposed method allows a broader weight space. To be practical, in FCM, both $\mathbf{x}_7$ and $\mathbf{x}_{15}$ in Figure.8 are actually assigned $(\frac{1}{4}, \frac{1}{4})$, or in the original notation $(\frac{1}{2}, \frac{1}{2})$. But considering that $\mathbf{x}_{15}$ is a obvious exception, it should be assigned a weight vector close to $\mathbf{0}$. And $\mathbf{x}_7$ should have a weight vector like $(0.8, 0.8)$ to indicate two clusters overlap at here, which are both obtained in our procedure.

With respect to PCM, the crucial improvement is that trivial null assignment is avoided without including explicit constraints. As far as we concern null assignment(assign all $w_{i,j}$ to zero in Equation.(4) without the constraint term) as a sort of unavoidable overfitting in PCM by its point estimation nature, a Bayesian improvement is a nature solution. It is observed from Figure.9 and Figure.10 that in most plausible cases the null state is not even the local maxima, not mentioning the concentration of all weight mass. The null weight privilege only appears when it does make a physical sense, i.e. the corresponding data entry is likely to be a noise.

For comparison with GMM, we see the whole picture as a more general case, since sampling only $(0, 1)$ and $(1, 0)$ from $\mathcal{U}$ will yield the exact result of GMM. However, introducing fuzzy weight vector(namely the vectors with their components summed up as a number other than one) brings a more fruitful conclusion. Using GMM we could discriminate $\mathbf{x}_7$ and $\mathbf{x}_{15}$ as a informative and a noisy entry only after computing $p(\mathbf{x}_i)$ by integrating out $\mathbf{U}$(theoretically $\mathbf{M}$ should be integrated out as well, but it would be difficult so the point estimation, or maximum liklihood estimation is remained.) But we can draw the same conclusion at the E-step, moreover, we explicitly bring that conclusion into the M-step, so the centroids will not be attracted to $\mathbf{x}_0$ and $\mathbf{x}_{14}$. From a perspective of symmetry, the idealistic centroids on $\mathbf{x}_4$ and $\mathbf{x}_{11}$ can be stable ones only if $\mathbf{x}_7$ is assigned a weight vector close to $(1, 1)$, which can only be yielded by BPCMC in all mentioned methods.

To summary, we include the conclusions drawn from the butterfly dataset in Table.2:

Table 2: A comparison of models from the butterfly case.

	Comparative advantage
HCM	Soft assignment, avoid overfitting.
FCM	Allow overlapped assignments(close to $(1,1)$,) avoid overfitting.
PCM	Avoid overfitting, specifically null assignment, without including extra arbitrary constraints.
GMM	Allow a more representative weight space, better physical interpretation, early detection of exceptional entries.

Most of those advantages are intuitively straightforward corrolaries from the essence difference w.r.t Table.1.

4.2 Butterfly Illustration: M-step, a Study of Stability

So far we only examine the weight space introduced by a fixed centroids' location, now we apply M-step and study the convergence of BPCMC.

We use a randomized initialization of $\mathbf{M}^0$ and operate Algorithm.1 until convergence, we mark the initialization points red in Figure.11.

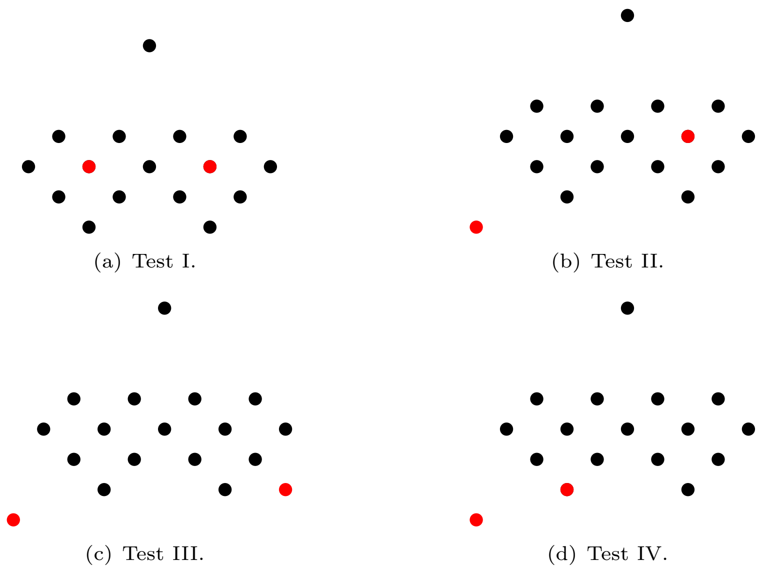

(a) Test I. (b) Test II. (c) Test III. (d) Test IV.

Figure 11: Randomized initialization of $\mathbf{M}^0$

In all four cases the computations converge in less than twenty iterations with result showed in Table.3.

Table 3: The result in four random initialized cases.

Case index	$\mathbf{m}_1$	$\mathbf{m}_2$
I	(1.98178,1.93054)	(3.01822,1.93054)
II	(1.98086,1.93041)	(3.01729,1.93066)
III	(1.98178,1.93054)	(3.01822,1.93054)
IV	(1.3453,1.81954)	(2.05675,1.95276)

In all cases I, II and III we have the final computed centroids at Figure.12.

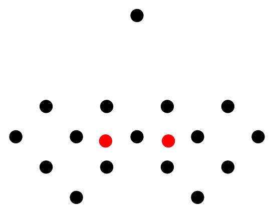


Figure 12: Result of case I, II and III.

In case IV we have Figure.13.

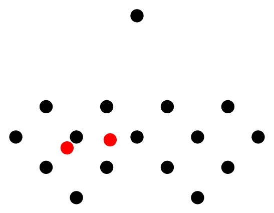


Figure 13: Result of case I, II and III.

Some extra attemps show that the final **M** converges to Figure.13 if they are initialized at different "wings". Or the convergence is weired. Altough the result might seem less convincing, but butterfly itself is not a typical model-based dataset after all. But the stabiltiy is proved to be realiable.

Remark: Although BPCMC is relatively stable in this toy dataset, as all model-based clustering algorithms, BPCMC is still sensitive to the scale since it might lead to float inaccuracy in computing the exponential. We suggest that different models, or rather the exact form of d be exhaustively test in order to find a best physical plausible result. Throughout this two section we have used the form:

$$d(\mathbf{x},\mathbf{m}) = \lambda \cdot (\mathbf{x}-\mathbf{m})^{(\mathrm{T})}(\mathbf{x}-\mathbf{m})$$

Which gives a relatively straightforward form of updating in M-step:

$$\mathbf{m}_1^{(t+1)} = \frac{\sum_{i=0}^{15}\sum_{\mathbf{u}\in\mathrm{grid}} p(\mathbf{u}|\mathbf{x}_i,\mathbf{M}^{(t)})u_1\mathbf{x}_i}{\sum_{i=0}^{15}\sum_{\mathbf{u}\in\mathrm{grid}} p(\mathbf{u}|\mathbf{x}_i,\mathbf{M}^{(t)})u_1}$$

$$\mathbf{m}_2^{(t+1)} = \frac{\sum_{i=0}^{15}\sum_{\mathbf{u}\in\mathrm{grid}} p(\mathbf{u}|\mathbf{x}_i,\mathbf{M}^{(t)})u_2\mathbf{x}_i}{\sum_{i=0}^{15}\sum_{\mathbf{u}\in\mathrm{grid}} p(\mathbf{u}|\mathbf{x}_i,\mathbf{M}^{(t)})u_2}$$

The conjugation between $\mathbf{m}_1$ and $\mathbf{m}_2$ lies in their affecting the term $p(\mathbf{u}|\mathbf{x},\mathbf{m}_1,\mathbf{m}_2)$.

4.3 Typical Mixture Model Data

Now we turn to a typical mixture data, we generate a GMM dataset showed in Figure.14, where we mark the initialized centroids as green and the converged solution as red.

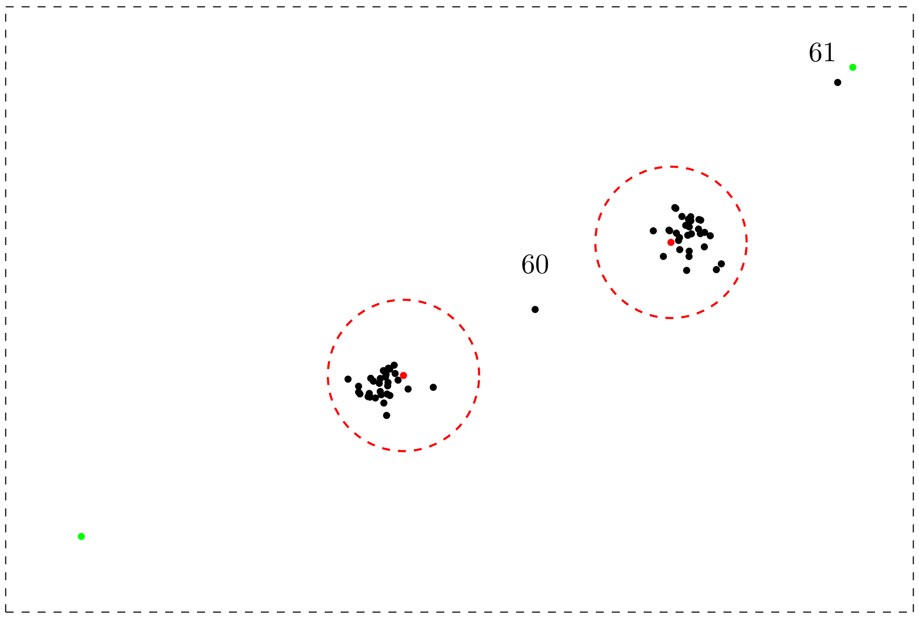


Figure 14: Unlabelled Data, N=60

Despite of a stable result, we are more interested in the induced weight map, and we show them for some typical $\mathbf{x}_i$ in Figure.15 and Figure.16 for the initial case and the converged case. In which we have $\mathbf{x}_0$ and $\mathbf{x}_{10}$ belongs to the left cluster. $\mathbf{x}_7$ and $\mathbf{x}_{23}$ are randomly picked entries from the right one. $\mathbf{x}_{60}$ stands for the entry at the middle of Figure.14, and $\mathbf{x}_{61}$ the far-most one at the top-right corner.

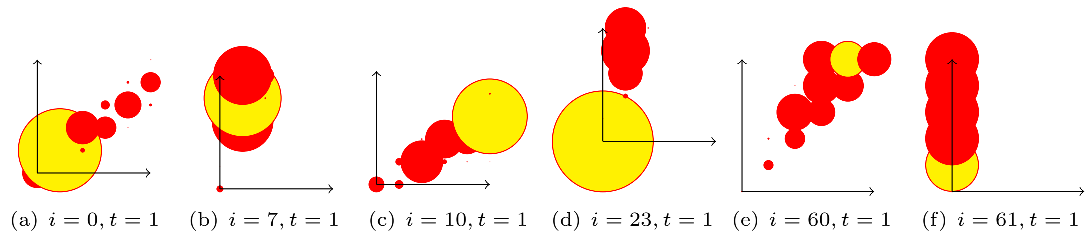

(a) $i = 0, t = 1$ (b) $i = 7, t = 1$ (c) $i = 10, t = 1$ (d) $i = 23, t = 1$ (e) $i = 60, t = 1$ (f) $i = 61, t = 1$

Figure 15: Iteration 1.

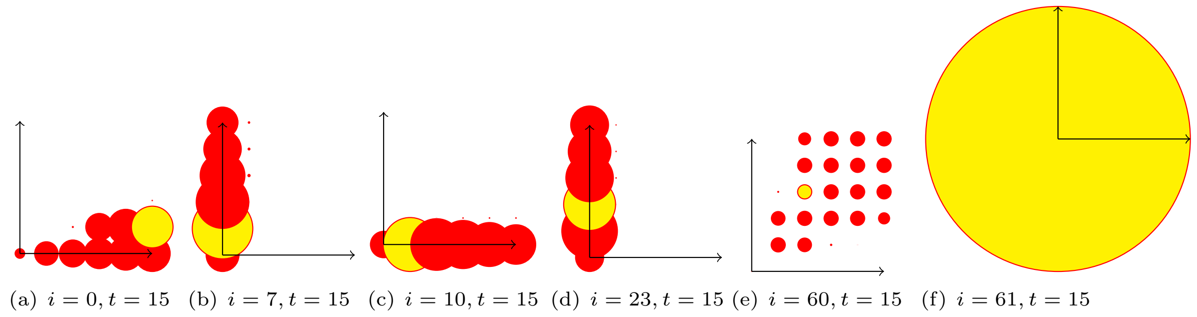

(a) $i = 0, t = 15$ (b) $i = 7, t = 15$ (c) $i = 10, t = 15$ (d) $i = 23, t = 15$ (e) $i = 60, t = 15$ (f) $i = 61, t = 15$

Figure 16: Iteration 15.

It is remarkable to notice the (e) and (f) case in both Figure.15 and Figure.16. Since one centroid is initialized somewhere near $\mathbf{x}_{61}$, so Figure.15(f) appears to be neat, but after iterations it reduced to Figure.15, which strongly suggest that it should be assigned $(0, 0)$, a weight vector that is a natural noise-like given by our computation. For $\mathbf{x}_{60}$, the rational case is to assign (a, a), which is yielded since Figure.16(e) is symmetric w.r.t $y = x$. For other ordinary entries, the mass in weight space is concentrated on the legal edge, which is the desired property.

To be practical, we generate a synthetic database with N =600(generated from aGMM)+375(uniform noise) entries. And the centroids are located with high accuracy(almost exactly) as shown in Figure.17.

One problem here is that when fixing $C = 2$, we sometimes have both $\mathbf{m}_1$ and $\mathbf{m}_2$ converge to a same centroid, yet both ideal centroids could become the shared converged value. In another words, we might run a $C = 2$ BPCMC for several times and only one centroid is found at each time. But both synthetic centroids can be found at length and this could be solved by introducing $C > 2$.

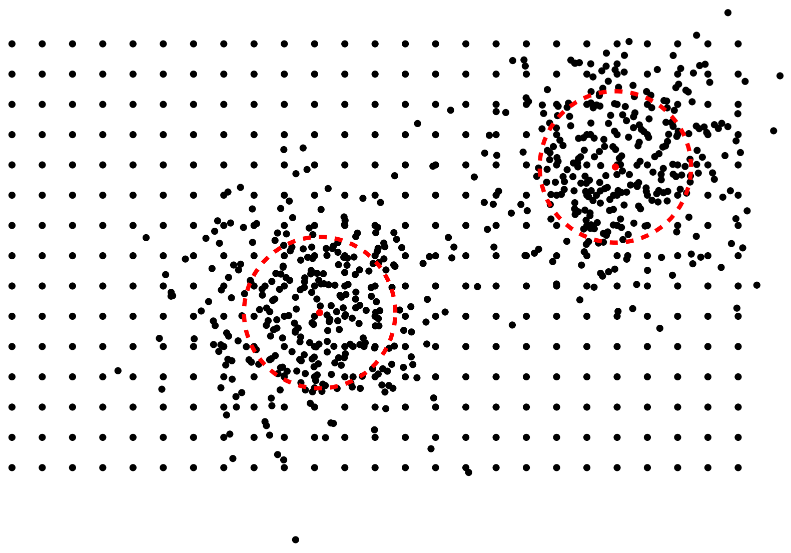


Figure 17: $N = 975$ with additional 375 noise entries

5 Conclusion

5.1 Conclusion

So far we establish a procedure that can do clustering task with some advantages over classical models. Since this proposal is not dataset-motivated, we do not claim that our method is doomed to outperform some dataset-oriented methods.

Since our proposal is stemmed from a retrospective reasoning of the ordinary models, we reiterate their principles and qualitive properties as the loss of concise in Section.2. Which should make it easier to follow the reasoning in constructing the model is Section.3. As a Bayesian generalization of PCM, we show that BPCMC can avoid PCM's overfitting, or sensitivity to initializations by integrating out the weight space. Together with a series of relatively informative and inspiring properties are proved.

Some propositions drawn at the design phase of the model is verified by some detailed illustrations in Section.4. As far as FCM, PCM and HKM are concerned, our proposal's advantage is robustness and stability. As for GMM, HKM and FCM, the privilege lies in more informative physical interpretation from sampling from a universe $\mathcal{U}$, spefically the ability to detect noise.

5.2 Works to be Done

We have three possible works to be done within plan:

1. To test the proposed property three, i.e.a natural including of determining the number of clusters in the whole procedure. Especially the practical interpretation of the eigenvectors from the PCA.

2. To test some other d to handle more sophiscated models, e.g.Gaussian rings.

3. To test some modifications on $p(\mathbf{u})$ or $\mathcal{U}$ to fit some practical datasets.

6 Reference(unassorted)

1. Lee, Sw , et al. "Iterative Bayesian fuzzy clustering toward flexible icon-based assistive software for the disabled." Information Sciences 180.3(2010):325-340.

2. Quost, Benjamin , and T. Denoeux . "Clustering Fuzzy Data Using the Fuzzy EM Algorithm." Scalable Uncertainty Management-international Conference DBLP, 2010.

3. Glenn, Taylor C , A. Zare , and P. D. Gader . "Bayesian Fuzzy Clustering." Fuzzy Systems IEEE Transactions on 23.5(2015):1545-1561.

4. Theis, and J. F. . Bayesian fuzzy clustering of colored graphs. Latent Variable Analysis and Signal Separation. Springer Berlin Heidelberg, 2012.

5. Bezdek, and JamesC. Pattern recognition with fuzzy objective function algorithms. Pattern recognition with fuzzy objective function algorithms /. Plenum Press, 1981.

6. Dunn, J. C . "A fuzzy relative of the ISODATA process and its use in detecting compact well-separated clusters." Journal of Cybernetics 3.3(1973):32-57.

7. Cai, Weiling , S. Chen , and D. Zhang . "Fast and robust fuzzyc-means clustering algorithms incorporating local information for image segmentation." Pattern Recognition 40.3(2007):825-838.

8. Wang, Shi Lin , et al. "Robust lip region segmentation for lip images with complex background." Pattern Recognition 40.12(2007):3481-3491.

9. Thierry Denœux. "Maximum likelihood estimation from fuzzy data using the EM algorithm." Fuzzy Sets & Systems 183.1(2011):72-91.

10. Lin, Horng Horng , J. H. Chuang , and T. L. Liu . "Regularized background adaptation: a novel learning rate control scheme for gaussian mixture modeling. " IEEE Trans Image

Process 20.3(2011):822-836.

11. Zivkovic, Zoran , and F. V. D. Heijden . "Efficient adaptive density estimation per image pixel for the task of background subtraction." Pattern Recognition Letters 27.7(2006):773-780.

12. Cai, Weiling , S. Chen , and D. Zhang . "Robust fuzzy relational classifier incorporating the soft class labels." Pattern Recognition Letters 28.16(2007):2250-2263.

13. Liang, Zhehui , P. Zhang , and J. Zhao . "Optimization of the number of clusters in fuzzy clustering." International Conference on Computer Design & Applications IEEE, 2010.

14. Yang, et al. "Improved k-means algorithm to quickly locate optimum initial clustering number K." Control Conference IEEE, 2011.

15. Zhaoshui, He , et al. "Detecting the number of clusters in n-way probabilistic clustering." IEEE Transactions on Pattern Analysis & Machine Intelligence 32.11(2010):2006-21.

16. Wang, Jeen Shing , and J. C. Chiang . "A Cluster Validity Measure With Outlier Detection for Support Vector Clustering." IEEE Transactions on Systems Man & Cybernetics Part B Cybernetics A Publication of the IEEE Systems Man & Cybernetics Society 38.1(2008):78-89.

17. Nie, Feiping , et al. "Spectral Embedded Clustering: A Framework for In-Sample and Out-of-Sample Spectral Clustering." IEEE Transactions on Neural Networks 22.11(2011):1796-808.

18. Law, Martin H C , Mário A T Figueiredo, and A. K. Jain . "Simultaneous feature selection and clustering using mixture models." IEEE Trans Pattern Anal Mach Intell 26.9(2004):1154-1166.

19. Huang, Dong , C. D. Wang , and C. D. Wang . Robust Ensemble Clustering Using Probability Trajectories. IEEE Educational Activities Department, 2016.

20. Wang, Xiao Feng , and D. S. Huang . "A Novel Density-Based Clustering Framework by Using Level Set Method." IEEE Transactions on Knowledge & Data Engineering 21.11(2009):1515-1531.

21. Yu, Zhiwen , et al. "Adaptive Fuzzy Consensus Clustering Framework for Clustering Analysis of Cancer Data." IEEE/ACM Transactions on Computational Biology & Bioinformatics 12.4(2015):887-901.

22. Wong, A. K. , and E. S. Lee . "Aligning and Clustering Patterns to Reveal the Protein Functionality of Sequences." IEEE/ACM Transactions on Computational Biology & Bioinformatics 11.3(2014):548-560.

23. Sun, Ying Sun Ying , et al. "Clustering methods based on rough estimate of cluster core." International Congress on Image & Signal Processing IEEE, 2010.

24. Bhagawati, Rupam , S. R. Laskar , and B. Swain . "Documents clustering using quantum clustering algorithm." International Conference on Microelectronics IEEE, 2016.

25. Li, Jia , D. Li , and Y. Zhang . "Efficient Distributed Data Clustering on Spark." IEEE International Conference on Cluster Computing IEEE Computer Society, 2015.

26. Huang, Dong , C. D. Wang , and J. H. Lai . "Locally Weighted Ensemble Clustering." IEEE Transactions on Cybernetics 48.5(2016):1460-1473.

27. Nock, R , and F. Nielsen . "On weighting clustering." IEEE Trans Pattern Anal Mach Intell 28.8(2006):1223-1235.

28. Krishnapuram, Raghu , and J. M. Keller . "A possibilistic approach to clustering." IEEE Transactions on Fuzzy Systems 1.2(2002):98-110.

29. Krishnapuram, R. , and J. M. Keller . "The possibilistic C-means algorithm: insights and recommendations." IEEE Transactions on Fuzzy Systems 4.3(2002):385-393.

30. Schneider, A. . "Weighted possibilistic c-means clustering algorithms." IEEE IEEE International Conference on Fuzzy Systems IEEE, 2002.

31. Zhang, Jiang She , and Y. W. Leung . "Improved possibilistic C-means clustering algorithms." IEEE Transactions on Fuzzy Systems 12.2(2004):209-217.

32. Pal, N. R. , et al. "A possibilistic fuzzy c-means clustering algorithm." IEEE Transactions on Fuzzy Systems 13.4(2005):517-530.

33. "Robust Clustering Using Outlier-Sparsity Regularization." IEEE Transactions on Signal Processing 60.8(2011).

34. Jafar, O. A. Mohamed , and R. Sivakumar . "A study on possibilistic and fuzzy possibilistic C-means clustering algorithms for data clustering." International Conference on Emerging Trends in Science IEEE, 2013.

35. Ayed, Abdelkarim Ben , M. B. Halima , and A. M. Alimi . "Adaptive fuzzy exponent cluster ensemble system based feature selection and spectral clustering." IEEE International Conference on Fuzzy Systems IEEE, 2017.

36. Wan, Xin , et al. "Ensemble clustering via Fuzzy c-Means." International Conference on Service Systems & Service Management IEEE, 2017.