Report on a New Split-Merge MCMC Algorithm

Adrian Cao, Allen Huang, Wei Li, Jingcheng Meng

Dec 21, 2022

1 Motivation and Goal of the Project

Gibbs sampler is the most common choice for Bayesian mixture model which takes advantage for a nicely restaurant process. But it may become trapped in isolated modes corresponding to an inappropriate clustering of data points, especially when there are similar clusters. This is because incremental updates are unlikely to move a single observation to a new mixture component because such an intermediate state has low probability and may therefore lead to bad mixing and convergence of the chain. A **random split-merge scheme** proposed by **Neal** helps with this case.

A worthwhile problem might be how to properly design the proposal in such a manner that split and merge is decided conditional on the data. The restricted Gibbs sampling proposed by Neal et.al describes a proposal distri-

bution in which properties of the observed data are used to decide how to split mixture components. The split-merge process is done via a restricted Gibbs sampling scan with respect to only a subset of data. This yields a method in which reasonable splits of components are more frequently proposed. However, it's still limited in a random split manner under a restricted data scan.

Even though there are some other variants of the split-merge scheme such as **smart-dumb algorithm**, which is smart in a sense to construct "smart" proposals with higher probability according to the target distribution a smart design can only be used when coupled with a dumb design as to preserve the detailed balance condition which might sacrifice its overall performance.

All of the above methods outperform the original Gibbs sampler for the mixture model with similar components but overall they still have a fairly low acceptance probability, this may be due to those random/dumb configurations in the algorithms.

Given that **Try multiple Metropolis** is a more flexible framework, which proposes a few independent samples simultaneously to be compared and decided in a probabilistic manner, we can avoid performing the dumb step but designing split and merge proposal in a more flexible way. Also, the decision for split or merge can be improved into a more sensible way that conditional on data.

A direct adjustment of the split-merge algorithm based on the Multiple Try framework is our goal for this project. It will include proposing a new

Multiple Try Split Merge algorithm for mixture model, suggesting several possible options for proposal distributions as well as conditional move type structure, and possible implementation of the algorithm on simulated data if time permits.

2 Problem Statement and Setup

The basic set-up of Dirichlet process mixture model is that for the data $y_1, \ldots, y_n$, which are independent drawn from a mixture of distribution $F(\theta)$. And there is no restrictions for y_i: they could be multivariate with real-valued or categorical components. For the parameters θ_i of the distribution where we draw the observation y_i, they are also independently draw from a mixture model G. For this mixing distribution G, we acquire the Dirichlet process prior, with concentration parameter α and base distribution G_0.

$$\begin{aligned} y_i \,|\, \theta_i &\sim F(\theta_i) \\ \theta_i \,|\, G &\sim G \\ G &\sim DP(G_0, \alpha) \end{aligned} \tag{1}$$

As for the Dirichlet process, the realization is discrete with the probability of one. Thus, we could consider the prior of distribution of θ_i holding the

following conditional distribution:

$$\theta_i \,|\, \theta_1, \ldots, \theta_{i-1} \sim \frac{1}{i-1+\alpha} \sum_{j=1}^{i-1} \delta(\theta_j) + \frac{\alpha}{i-1+\alpha} G_0 \tag{2}$$

Here, $\delta(\theta_j)$ is the distribution centered around θ_j. And θ_j could also be represented by using the "latent class" c_i and let $\theta_j = \phi_{c_j}$. Thus the we could represent the Dirichlet mixture model into a finite mixture model of K components as K goes to infinity.

$$\begin{aligned} y_i \,|\, c_i \phi &\sim F(\phi_{c_i}) \\ c_i \,|\, \boldsymbol{p} &\sim Discrete(p_0, \ldots, p_K) \\ \phi_c &\sim G_0 \\ \boldsymbol{p} &\sim Dirichlet(K, \alpha) \end{aligned} \tag{3}$$

The mixing proportion of the latent classes c_i is $\boldsymbol{p} = (p_1, \ldots, p_K)$, given a symmetric Dirichlet prior with concentration parameter α/K, so that it approaches zero as K goes to infinity. Then the prior of the latent classes c_i could be calculated by integrating out the mixing proportions $\boldsymbol{p}$. Thus,

$$P(c_i = c \,|\, c_i, \ldots, c_{i-1}) = \frac{n_{i,c} + \alpha/K}{i-1+\alpha} \tag{4}$$

Here, $n_{i,c}$ is the number of c_j for $j < i$ that are equal to c. And if K goes to infinity, we could get the prior for c_i for the Dirichlet process mixture model

we talked in equation (1)

$$\begin{aligned} P(c_i = c \,|\, c_i, \ldots, c_{i-1}) &\to \frac{n_{i,c}}{i-1+\alpha} \\ P(c_i \neq c_j \forall j < i \,|\, c_i, \ldots, c_{i-1}) &\to \frac{\alpha}{i-1+\alpha} \end{aligned} \tag{5}$$

3 Formulation of the Algorithm

3.1 The Main Algorithm Step

Let x denote the current state of cluster configuration. We first define the proposal function to be:

$$Q(x, y) = p(\text{split}|x)Q_s(x, y) + p(\text{merge}|x)Q_m(x, y)$$

where Q_s is the proposal function for split and Q_m is the proposal function for merge. $\pi(x)$ is the likelihood function derived from Dirichlet Mixture Process. $\lambda(x, y)$ is a non-negative symmetric function in x and y.

Then **Multiple Try Split-Merge Algorithm** with $k = 2$ is stated as following:

1. Draw two independent proposals y_1, y_2 from $Q(x, .)$. Compute the weight $w(y_i, x)$ for y_1 and y_2 where $w(x, y) = \pi(x)Q(x, y)\lambda(x, y)$.

2. Select y from y_1 and y_2 with probability proportional to the weights.

3. Now produce a reference set by drawing x_1 from the distribution $Q(y, .)$. Set $x_2 = x$(the current point).

4. Accept y with the probability $r = min(1, \frac{w(y_1,x)+w(y_2,x)}{w(x_1,y)+w(x_2,y)})$.

The multiple try metropolis algorithm satisfies the detailed balance property and thus **Multiple Try Split-Merge Algorithm** satisfies the detailed balance property. Hence, **Multiple Try Split-Merge Algorithm** will always produce a reversible Markov chain with $\pi(x)$ as the stationary distribution no matter what proposal function $Q_s(x, y)$ and $Q_m(x, y)$ we choose.

3.2 Choice of Q_s and Q_m

Before we conduct our algorithm, we need to determine Q_s and Q_m. In [3], they use

$$Q_s(x, y) \propto \frac{1}{P(x^{(i)}|x)} \tag{6}$$

as smart split, where the current cluster configuration of data x is $x = (x^{(1)}, .., x^{(i-1)}, x^{(i)}, x^{(i+1)}, ..., x^{(I)})$, $y = (x^{(1)}, .., x^{(i-1)}, x^{(i,1)}, x^{(i,2)}, x^{(i+1)}, ..., x^{(I)})$ is the cluster configuration of next step, and P is the probability from 5. And

they use the following probability as Q_m:

$$Q_m(x, y) \propto P(x^{(i)}, x^{(j)}|x)$$

where $y = \{x \backslash \{x_i, x_j\}, x^{(i)} \cup x^{(j)}\}$ is the cluster configuration for the next step. In this project, we will also use the same Q_s and Q_m as our proposal distribution for the split and merge.

For $p(\text{split}|x)$ and $p(\text{merge}|x)$, we will let them both equal to $1/2$ for simplicity.

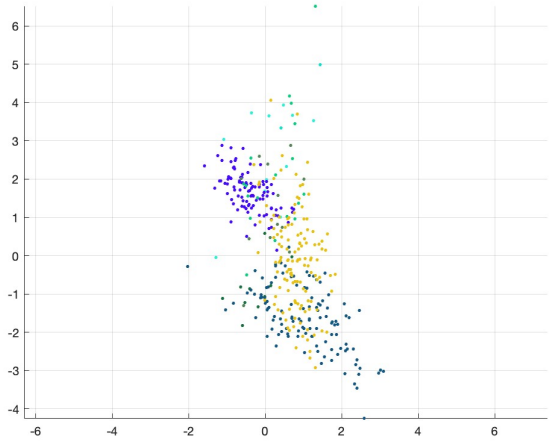
4 Simulation Study

4.1 Reproduction of Restricted Gibbs Scan Algorithm

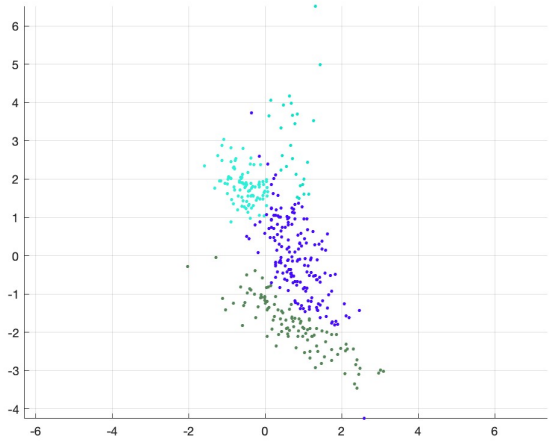
One of our purposes of doing simulation is to get a sense of the performance of some current competing methods, such as the classic Restricted Gibbs Scan Algorithm.

Motivated by [1], we first run a simulation of 4-component bi-variate Gaussian mixture data, and then perform a restricted gibbs scan algorithm to be compared with a matlab built-in GMM estimation, following in figure 1, there are both the original simulated data, and also clustering performance from our competing algorithms. And Figure 2 is screenshot of process of the Restricted Gibbs Scan Algorithm.

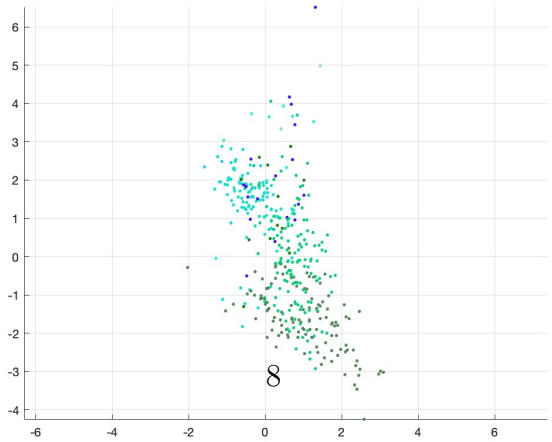
Based on the results, we have several observations:



(a) simulated data



(b) built-in estimation for GMM: EM algorithm



(c) Restricted Gibbs Scan

Figure 1: Simulation stduy for a four-component bivariate Gaussian Data

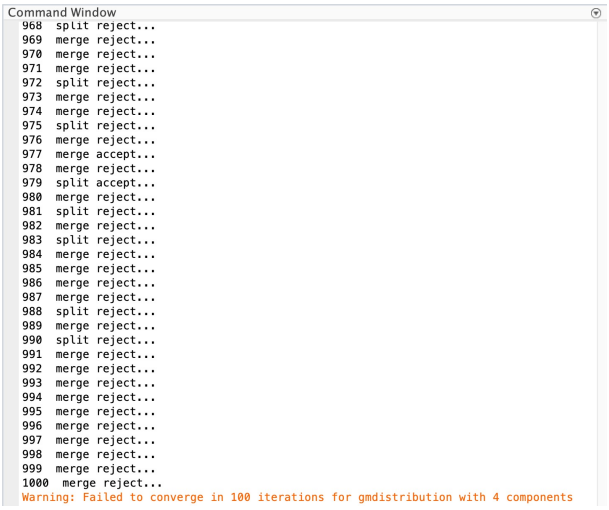


Figure 2: of Restricted Gibbs Scan Algorithm

1. From Figure 1, Restricted Gibbs Scan seems to outperform the deterministic EM algorithm for this case when there are a lot of overlapping among components.

2. Figure 2 shows that the restricted gibbs scan has a obviously low acceptance ratio and the chain seems not converge even after sufficient number of iterations for this case.

4.2 Construction of Multiple Try Split Merge Algorithm

Based on it, We will refer to some of the implementation structures from there such as how a merge and split step is actually performed, and how a likelihood model is parameterized, how a prior elicitation is considered for mixture model, etc. Bring both the likelihood model, split-merge realization

and smart proposals into our main body of a Multiple Try MCMC, we can expect a relization of our **Multiple Try Split Merge Algorithm**.

This will by the main focus of our future study.

5 Further Plans and Extensions

Further finalization of this project would be to implement such a Multiple Try Split Merge Algorithm, conducting comparison study of different choices of proposals. Comparing our algorithm with those current competing methods for a improvement of acceptance rate would be a large interest.

Apart from adjusting the MCMC algorithm to get avoid of random configurations, another way to bring the conditional sampling structure is to incorporation metric of cluster distance into the algorithm itself as to navigate the next split-merge in a more smart manner.

References

[1] *An online implementation of Restricted Gibbs Scan.* `https://github.com/zhouyifan233/Split-Merge-MCMC-for-Gaussian-Mixture-Model.matlab`. Accessed: 2022-12-15.

[2] Sonia Jain and Radford M Neal. "A split-merge Markov chain Monte Carlo procedure for the Dirichlet process mixture model". In: *Journal of computational and Graphical Statistics* 13.1 (2004), pp. 158–182.

[3] Wei Wang and Stuart Russell. “A Smart-Dumb/Dumb-Smart Algorithm for Efficient Split-Merge MCMC.” In: *UAI*. Vol. 15. 2015, pp. 902–911.