

EVOLUTIONARY MONTE CARLO METHODS FOR CLUSTERING

Gopika R. Goswami Jun S. Liu
Wing H. Wong

{goswami, liu}@stat.harvard.edu, whwong@stanford.edu *

September 9, 2005

Abstract

We consider the problem of clustering a group of observations according to some objective function (e.g. K-means clustering, variable selection) or according to a posterior density (e.g. posterior from a Dirichlet Process prior) of cluster indicators. We cast both kinds of problems in the framework of sampling for cluster indicators. So far, Gibbs sampling, “split-merge” Metropolis-Hasting algorithm and various modifications of these have been the basic tools used for sampling in this context. We propose a new population based MCMC approach, in the same vein as parallel tempering. We introduce three new “crossover moves” (based on swapping and reshuffling sub-clusters intersections) which make such an algorithm very efficient with respect to Integrated Autocorrelation Time (IAT) of various relevant statistics and also with respect to the ability to escape from local modes. We call this new algorithm Population Based Clustering (PBC) algorithm. We apply PBC algorithm to motif clustering, Beta mixture of Bernoulli clustering and a Bayesian Information Criterion (BIC) based variable selection problem. We also discuss clustering of mixture of Normals and compare the performance PBC algorithm as a stochastic optimizer with K-means clustering.

*Jun S. Liu is professor and Gopika R. Goswami is lecturer and post doc in the Department of Statistics, Harvard University, Cambridge, MA 02138, USA. Wing H. Wong is professor of Statistics and Health Research and Policy, Stanford University, Stanford, CA 94305, USA. GRG and JSL’s work was supported by the National Science Foundation under Grant No. DMS04-38240. WHW’s work was supported by the National Science Foundation under Grant No. DMS0505732

1 Introduction

The problem of clustering a given set of observations arises in many different applications, e.g., text/speech recognition, biological motif clustering, various classification problems etc. K-means algorithm [8, Hastie et. al., 2001] is one of the principal and widely used methods of clustering. The algorithm minimizes an objective function which measures the “goodness” or “fitness” of a proposed cluster. The starting point of our approach here is that “any minimization problem could be cast into a stochastic minimization exercise through sampling”. In other words, we incorporate the objective function mentioned above in the Boltzman distribution format and sample from the resulting density over the space of all possible clusters. In some cases, such a density over the space of possible clusters arises naturally, for example, Bayesian analysis of Dirichlet process mixture models. Thus, the problem of clustering, in general, could be addressed as a problem of sampling from a given density over clusters. For doing the sampling we use Markov Chain Monte Carlo (MCMC) methods.

Most popular MCMC methods are Metropolis-Hastings (MH) algorithm [9, Hastings, 1970], Gibbs sampling [4, Gelfand et. al., 1990], to name a few. In difficult multimodal situations, however, the above sampling methods do not work very well. A class of MCMC sampling methods, known to be effective in such situations, rely on the use of a population and a corresponding temperature ladder. We call such methods *population based methods*. Some of the popular population based methods existing in the literature are Parallel Tempering (PT) [5, Geyer, 1991], Adaptive Directional Sampling (ADS) [15, Gilks et. al., 1994], Conjugate Gradient Monte Carlo (CGMC) [14, Liu et. al., 2000], Evolutionary Monte Carlo (EMC) method [12, Liang et. al., 2000].

We call our sampling recipe the Population Based Clustering (PBC) algorithm. Our method is quite similar in essence to EMC, which is, in turn, built on PT. We introduce three new “moves”, namely, SCSC:TWO-NEW, SCSC:ONE-NEW and SCRC within the EMC framework. These moves of PBC enhance performance with respect to “dependency of samples” and ability to escape “local modes”; as a result the sampler performs much better than Gibbs sampling, the “split-merge” Metropolis-Hastings algorithm and the K-means algorithm. For the main motivation behind the PBC algorithm over EMC see section 2.

The paper is organized as follows. In section 2, we discuss the pros-n-cons of various existing approaches to clustering and motivate the need for considering the PBC algorithm. In the various subsections of section 3 the PBC method and its different “moves” are introduced. In section 4 we discuss implementation of the algorithm. In section 5 we introduce the Dirichlet process mixture model an exposure to which is needed for the examples in this article. In section 6 we discuss a general strategy for computing the full conditionals needed for Gibbs sampling. In various subsections of section 7, we present four different applications of PBC, namely, a Beta mixture of Bernoulli problem, a motif clustering problem, a stochastic optimization problem

with comparison to K-means and an example on variable selection. Lastly, in section 8 we gather our observations from our studies and propose future directions of research.

2 Various Approaches to Clustering

Various approaches to clustering a set of observations can be broadly divided into two classes, namely, methods which rely on sampling over space of possible clusters and methods which do not use sampling.

The principal methods falling in the second category are K-means clustering and hierarchical clustering. K-means performs a greedy search for the best clustering solution by iteratively minimizing an objective function (see section 7.3) and thus very often get stuck in some local mode in the solution space. This problem is solved partially by starting the algorithm with many different random choices for the starting values and then choosing the solution having the smallest value for the objective function. The statistical software R takes this approach. Another route for avoiding the local minima trap is to start K-means from a hierarchical clustering solution to the problem. The statistical software SPLUS takes this approach. But there is no guarantee that the global minimum of the objective function is achieved by the ultimate solution. Moreover, K-means takes the number of clusters the data needs to be categorized into as an input and thus there is no way to objectively determine the “right” cluster size.

The sampling-based approaches to clustering avoid some of the above mentioned problems of K-means. The principal method in this category is Gibbs sampling (**Gibbs**) and Metropolis-Hastings algorithm (**MH**). These samplers sample from the posterior density of the clusters which is induced by a reasonable prior. The most popular prior used in this context is the Dirichlet process prior which gives positive probability to all possible clustering solutions of all possible cluster sizes. One could also consider an uniform prior in this context. See [11, Jensen et. al., 2005] for a discussion on these two types of priors and section 5 for a brief introduction to Dirichlet process priors. Given the choice of prior, one can objectively determine the “right” number of clusters by considering the modal value of the induced posterior mode. Thus, if the either of the samplers, namely, **Gibbs** and **MH** are efficient enough to sample the posterior properly we can avoid the problem of specifying the cluster size beforehand, as is the case with K-means.

Unfortunately, both **Gibbs** and **MH** are known to do a poor job of sampling from a general density in high dimensions. They, like K-means, suffer from the problem of local-minima-trapping. In the MCMC literature, the most popular solution to this problem is to consider Parallel Tempering (**PT**) and similar approaches. The most advanced method in this class is called the Evolutionary Monte Carlo (**EMC**) method. **EMC** has been shown to perform much better than **PT**, Reversible Jump

MCMC, Gibbs and MH. For an intuitive discussion on why EMC is expected to solve the local-minima-trapping problem see the conclusion section of [13, Liang et. al., 2001].

Our main innovation over vanilla EMC is the design of new crossover moves that can take advantage of the special structure of the space of clustering solutions, see sections 3 and 7.4 for further discussions.

3 Population Based Clustering

Suppose we have $d(\in \mathbb{N})$ many objects to be classified into clusters. We denote the possible cluster indicator vectors by \underline{z} . Let $D := \{1, 2, \dots, d\}$; then evidently, for some $\underline{Z} \subset D^d$ we have, $\underline{z} \in \underline{Z}$. So, for example, $\underline{z} = (k, k, \dots, k)$ for any $k \in \{1, 2, \dots, d\}$ means that all our objects belong to only one cluster. On the other extreme of the spectrum, \underline{z} equals any permutation of $(1, 2, \dots, d)$ means that each object forms its own cluster. Now let we have a density for the cluster indicators expressed in the Boltzman distribution form with a suitable value for τ_{min} (which is usually 1 for sampling and less than 1 for stochastic optimization)

$$g(\underline{z}) \propto \exp\{-H(\underline{z})/\tau_{min}\}, \quad \underline{z} \in \underline{Z} \tag{1}$$

Here the function $H(\cdot)$ is called the *fitness* or the *energy* function. Usually the choice for temperature τ_{min} is dictated by the problem. For example, if we are interested in sampling from some unnormalized density $k(\underline{z})$ then one could take $H(\underline{z}) = -\log(k(\underline{z}))$ and $\tau_{min} = 1$ in equation (1). On the other hand, if we are interested in locating the modes of the density $k(\underline{z})$ then we consider $H(\underline{z}) = -\log(k(\underline{z}))$, as before, but now take $\tau_{min} < 1$ in equation (1).

The density $g(\cdot)$ could be the posterior density of cluster indicators arising from some Bayesian set up or it could arise from some cluster formation criterion measure (e.g. K-means clustering) raised to the exponent. We are going to use a *population based* method quite similar in vein to Parallel Tempering (PT) [5, Geyer, 1991] and Evolutionary Monte Carlo (EMC) Algorithm [12, Liang et. al., 2000] for sampling from $g(\cdot)$ above.

Toward that end, we first introduce the set up of population based algorithms like PT. First, we introduce a *suitable* temperature ladder: $t_1 > t_2 > \dots > t_N (= \tau_{min}) > 0$. Then we construct a sequence of densities for the cluster indicators $f_i(\underline{x}_i)$ defined by $f_i(\underline{x}_i) \propto \exp\{-H(\underline{x}_i)/t_i\}$. Note because $t_N = \tau_{min}$ we have $f_N(\cdot) = g(\cdot)$. Now, we expand the sample space from \underline{Z} to \underline{Z}^N and define the modified target density

$$f(\underline{x}) \propto \prod_{i=1}^N f_i(\underline{x}_i) \quad \text{on } \underline{Z}^N \quad (2)$$

We sample from the above density $f(\underline{x})$ and upon convergence, $\{\underline{x}_N^{(t)}, t = 1, 2, \dots\}$ will be the samples of our interest. In this context, we denote t_1 by τ_{max} for later reference. Also, borrowing terminology from [13, Liang et. al., 2001], we will use the notation $(\underline{x}, \underline{t}) := (\underline{x}_1, t_1; \dots; \underline{x}_N, t_N)$ and refer to the previous “tuple” as the state of the chain or the population. We will also call the individual components; \underline{x}_i of \underline{x} the chromosomes.

For a given cluster indicator vector \underline{z} , we would call $\mathcal{H} := \{H_s \mid s = 1, 2, \dots, S\}$ the sub-cluster representation of \underline{z} if H_s 's form a partition of D i.e. they are non-empty, disjoint with $\cup_{s=1}^S H_s = D$ and $\forall s, \{z_i \mid i \in H_s\}$ is a singleton i.e. all the members of H_s have the same cluster indicator.

Now in the following subsections, we describe the different moves involved in the population based clustering method.

3.1 Mutation

This move essentially consists of updating a chosen chromosome using a “split-merge” move. In other words, first we choose $i \in \{1, 2, \dots, N\}$ using some random or deterministic distribution $p(I = i \mid \underline{x})$ and then update it in the following manner. We fix a $q_s \in (0, 1)$, the probability of “split” and call $q_m := 1 - q_s$, the probability of “merge”. Now, let $\mathcal{A} := \{A_k \mid k = 1, 2, \dots, K\}$ represent the sub-clusters formed by the cluster indicator vector \underline{x}_i . First we choose a coordinate, say, j -th either systematically or randomly among the d coordinates of \underline{x}_i . Say, $j \in A_{k_0}$, for some $k_0 \in \{1, 2, \dots, K\}$. Now, depending on the the values of d, K and $|A_{k_0}|$, we decide to do the following.

If $|A_{k_0}| = 1$ then we randomly choose $k_1 \in \{1, 2, \dots, K\} \setminus \{k_0\}$ and decide to merge j to A_{k_1} i.e. we define the new sub-clusters as $\mathcal{C} := [\mathcal{A} \cup \{A_{k_1} \cup \{j\}\}] \setminus \{A_{k_0}, A_{k_1}\}$.

If on the other hand, $|A_{k_0}| > 1$ then we consider two different scenarios. If $K = 1$ (i.e. if $\mathcal{A} = \{A_{k_0}\}$) then we split j from the rest i.e. we define the new sub-clusters as $\mathcal{C} := \{A_{k_0} \setminus \{j\}, \{j\}\}$. If $K > 1$ then decide to split with probability q_s and merge with the remaining probability q_m . If we decide to split j from A_{k_0} then we define the new sub-clusters as $\mathcal{C} := [\mathcal{A} \cup \{A_{k_0} \setminus \{j\}, \{j\}\}] \setminus \{A_{k_0}\}$. If we decide to merge then we randomly choose $k_1 \in \{1, 2, \dots, K\} \setminus \{k_0\}$ and merge j to A_{k_1} , as described before.

Now, let \underline{y}_i be the cluster indicator vector formed by \mathcal{C} and propose the new population $(\underline{y}, \underline{t}) = (\underline{x}_1, t_1; \dots; \underline{y}_i, t_i; \dots; \underline{x}_N, t_N)$ and accept this new population with probability $\min(1, r_m)$ where,

$$r_m = \frac{f_i(\underline{y}_i)}{f_i(\underline{x}_i)} \times \frac{p(I = i|\underline{y})}{p(I = i|\underline{x})} \times \frac{T(\underline{y}_i, \underline{x}_i)}{T(\underline{x}_i, \underline{y}_i)} \quad (3)$$

Here $T(\underline{x}_i, \underline{y}_i)$ is the probability of generating \underline{y}_i from \underline{x}_i by the “split-merge” move described earlier. Let $\mathcal{C} := \{C_l \mid l = 1, 2, \dots, L\}$ and $j \in C_{l_0}$ for some $l_0 \in 1, 2, \dots, L$. With this notation, $T(\cdot, \cdot)$ takes the following form

$$T(\underline{x}_i, \underline{y}_i) = \begin{cases} 1 & \text{if } |C_{l_0}| = 1, |\mathcal{A}| = 1 \\ q_s & \text{if } |C_{l_0}| = 1, |\mathcal{A}| > 1 \\ 1/(|\mathcal{A}| - 1) & \text{if } |C_{l_0}| > 1, |A_{k_0}| = 1 \\ q_m \cdot 1/(|\mathcal{A}| - 1) & \text{if } |C_{l_0}| > 1, |A_{k_0}| > 1 \end{cases}$$

Note, we are omitting a term $1/d$, the probability of choosing the j -th object at random from the above expression because its going to appear both in $T(\underline{x}_i, \underline{y}_i)$ and $T(\underline{y}_i, \underline{x}_i)$ and hence going to get canceled anyway. This “split-merge” introduced here is different from the method introduced in [10, Jain et. al., 2004] and [2, Dahl, 2003]. This mutation move when applied systematically to each component of the chromosome of at a given temperature level, say, e.g. τ_{min} we call it the Metropolized Gibbs sampler and refer to it as the MH sampler. The rationale behind this choice of terminology is that in this case instead of sampling from the exact conditionals of the components of the chromosome we are carrying out a Metropolis-Hastings step.

3.2 The New Crossover Moves

In general, a crossover move in EMC is a move that takes two configurations (i.e. two parents) in the current population and re-combine them to produce two new configurations (two children) each inheriting some aspects of the parental configurations. We have developed two types of crossover moves for clustering problems, namely SCSC and SCRC. SCSC moves are further divided into two varieties: SCSC:TWO-NEW, and SCSC:ONE-NEW. These naming conventions will be explained at the appropriate places below. In the SCSC(SCRC) moves, we swap(reallocate) the members of sub-cluster intersections formed by two chosen parent chromosomes in certain ways and thus produce child chromosome(s) as proposal and hence we coin the terminology Sub Cluster Swap(Reallocation) Crossover.

3.3 SCSC:TWO-NEW

In the SCSC:TWO-NEW move, we try to exchange several bits of two randomly chosen chromosomes in a clever fashion. Here we first sample $i, j \in \{1, 2, \dots, N\}$, $i \neq j$ using some distributions $p(I_1 = i|x)$ and $p(I_2 = j|x, I_1 = i)$. We let $\mathcal{A} := \{A_k \mid$

$k = 1, 2, \dots, K$ and $\mathcal{B} := \{B_l \mid l = 1, 2, \dots, L\}$ represent the sub-clusters formed by the cluster indicator vectors \underline{x}_i and \underline{x}_j respectively. Here we first choose k_1, k_2 from $U := \{k \mid A_k \cap B_l \neq \emptyset, \text{ for at least two } l\text{'s}, k = 1, 2, \dots, K\}$ randomly without replacement. Note, if $|U| \leq 1$ then this move cannot be performed. Now we choose $l_1, l_2 \in \{1, 2, \dots, L\}$, $l_1 \neq l_2$ in the following way. For k_1 we choose l_1 randomly from $V_1 := \{l \mid A_{k_1} \cap B_l \neq \emptyset\}$. Then, We choose l_2 randomly from $V_2 := \{l \mid A_{k_2} \cap B_l \neq \emptyset \text{ and } l \neq l_1\}$. Note, $V_1, V_2 \neq \emptyset$; this is guaranteed by the way k_1 and k_2 were chosen. So, here we are essentially choosing $k_1 \neq k_2$ and $l_1 \neq l_2$ randomly such that $A_{k_i} \cap B_{l_i} \neq \emptyset$, $i = 1, 2$. Now, we form the disjoint collection of sets $\{S_{k,l} \mid k = 1, 2, \dots, K; l = 1, 2, \dots, L\}$ where,

$$S_{k,l} = \begin{cases} A_{k_1} \cap B_{l_1} & \text{for } k = k_2, l = l_2 \\ A_{k_2} \cap B_{l_2} & \text{for } k = k_1, l = l_1 \\ A_k \cap B_l & \text{otherwise} \end{cases} \quad (4)$$

and define $C_k = \uplus_{l=1}^L S_{k,l}$, $k = 1, 2, \dots, K$ and $D_l = \uplus_{k=1}^K S_{k,l}$, $l = 1, 2, \dots, L$. Let the cluster indicator vector formed by $\mathcal{C} := \{C_k \mid k = 1, 2, \dots, K\}$ and $\mathcal{D} := \{D_l \mid l = 1, 2, \dots, L\}$ be \underline{y}_i and \underline{y}_j respectively. Here we call $\underline{x}_i, \underline{x}_j$ the parents and $\underline{y}_i, \underline{y}_j$ the children. Here two parent chromosomes are being ‘‘crossed’’ to produce two new child chromosomes and hence this move is called the SCSC:TWO-NEW move. Now we propose the new population $(\underline{y}, \underline{t}) = (\underline{x}_1, t_1; \dots; \underline{y}_i, t_i; \dots; \underline{y}_j, t_j; \dots; \underline{x}_N, t_N)$ (where without loss of generality we assume $i < j$) and accept this new population with probability $\min(1, r_{scsc:two-new})$ where,

$$r_{scsc:two-new} = \frac{f_i(\underline{y}_i) f_j(\underline{y}_j)}{f_i(\underline{x}_i) f_j(\underline{x}_j)} \times \frac{T_{i,j}(\underline{y}, \underline{x})}{T_{i,j}(\underline{x}, \underline{y})} \quad (5)$$

Here, $T_{i,j}(\underline{x}, \underline{y}) = p(I_1 = i | \underline{x}) p(I_2 = j | \underline{x}, I_1 = i) + p(I_1 = j | \underline{x}) p(I_2 = i | \underline{x}, I_1 = j)$. Note, the probability for generating \underline{y} from \underline{x} by swapping the labels (k_1, l_1) and (k_2, l_2) is omitted in the above expression because they are the same in $T_{i,j}(\underline{x}, \underline{y})$ and $T_{i,j}(\underline{y}, \underline{x})$ and hence in they cancel out. One can either randomly select I_1 and I_2 without replacement or $p(I_1 = i | \underline{x})$ and $p(I_2 = j | \underline{x}, I_1 = i)$ are taken to be proportional to the Boltzmann distribution w.r.t. some selection temperature s , namely, $p(I_1 = i | \underline{x}) \propto \exp\{-H(\underline{x}_i)/s\}$ and $p(I_2 = j | \underline{x}, I_1 = i) \propto \exp\{-H(\underline{x}_j)/s\}$, $j \neq i$. Usually, s is taken close to the lowest temperature $t_N = \tau_{min}$. The intuition behind this kind of choice for s is that *good* parents would in turn produce *good* children and that a *good* sample w.r.t. a low temperature is also be *good* w.r.t. higher temperatures. A diagrammatic representation of the proposed strategy is shown in figure 1.

3.4 SCSC:ONE-NEW

In the SCSC:ONE-NEW move, we try to exchange several bits of one randomly chosen chromosome “guided” by another randomly chosen chromosome in a clever fashion. Here we first sample $i \in \{1, 2, \dots, N\}$ using some distribution $p(I_1 = i | \underline{x})$. Then we randomly choose a “neighbor” j of i in the following way. We select $j = i \pm 1$ s.t. $p(I_2 = i \pm 1 | \underline{x}, I_1 = i) = 0.5$, $p(I_2 = 1 | \underline{x}, I_1 = 0) = 1$ and $p(I_2 = N - 1 | \underline{x}, I_1 = N) = 1$.

Now, we choose one of I_1, I_2 above with probability $p(I_3 = i_3 | \underline{x}, I_1, I_2)$, $i_3 \in \{I_1, I_2\}$ and call this chosen chromosome \underline{x}_{I_3} the “parent” the other one the “child”. Note, we could choose I_3 randomly or for a given selection temperature s we could take $p(I_3 = i_3 | \underline{x}, I_1, I_2) \propto \exp[-H(\underline{x}_{i_3})/s]$, $i_3 \in \{I_1, I_2\}$

Without loss of generality, say, we happen to choose \underline{x}_i as our parent and \underline{x}_j as the child. Let $\mathcal{A} := \{A_k | k = 1, 2, \dots, K\}$ and $\mathcal{B} := \{B_l | l = 1, 2, \dots, L\}$ represent the sub-clusters formed by the cluster indicator vectors \underline{x}_i and \underline{x}_j respectively. Here we first we choose k_1 from $U := \{k | A_k \cap B_l \neq \emptyset, \text{ for at least two } l\text{'s}, k = 1, 2, \dots, K\}$ randomly. Note, if the set $U = \emptyset$ then this move cannot be performed. Then, we choose l_1, l_2 , $l_1 \neq l_2$ randomly from $\{l | A_{k_1} \cap B_l \neq \emptyset\}$ without replacement. Now by choice of k_1 such choice of l_1 and l_2 is possible. So, here we are essentially choosing k_1 and $l_1 \neq l_2$ randomly such that $A_{k_1} \cap B_{l_i} \neq \emptyset$, $i = 1, 2$. Now, we form the disjoint collection of sets $\{S_{k,l} | k = 1, 2, \dots, K, l = 1, 2, \dots, L\}$ where,

$$S_{k,l} = \begin{cases} A_{k_1} \cap B_{l_1} & \text{for } k = k_1, l = l_2 \\ A_{k_1} \cap B_{l_2} & \text{for } k = k_1, l = l_1 \\ A_k \cap B_l & \text{otherwise} \end{cases} \quad (6)$$

and define $C_k = \uplus_{l=1}^L S_{k,l}$, $k = 1, 2, \dots, K$ and $D_l = \uplus_{k=1}^K S_{k,l}$, $l = 1, 2, \dots, L$. Now, let the cluster indicator vector formed by $\mathcal{C} := \{C_k | k = 1, 2, \dots, K\}$ and $\mathcal{D} := \{D_l | l = 1, 2, \dots, L\}$ be \underline{y}_i and \underline{y}_j respectively. Note, that by construction $\underline{y}_i = \underline{x}_i$. We call \underline{y}_j is a “modified child” from parent \underline{x}_i and child \underline{x}_j . It means that \underline{y}_j could be produced by the set operations described above in equation (6) by “crossing” the parent \underline{x}_i and the child \underline{x}_j . So, here one of the parents gets to stay unperturbed and the other one, namely, the child gets modified with “guidance” from the parent to produce one new child chromosome and hence this move is called the SCSC:ONE-NEW move. A diagrammatic representation of the proposed strategy is shown in figure 2. Now, propose the new population $(\underline{y}, \underline{t}) = (\underline{x}_1, t_1; \dots; \underline{y}_j, t_j; \dots; \dots; \underline{x}_N, t_N)$ and accept this new population with probability $\min(1, r_{scsc:one-new})$ where,

$$r_{scsc:one-new} = \frac{f_j(\underline{y}_j)}{f_j(\underline{x}_j)} \times \frac{T_j(\underline{y}, \underline{x})}{T_j(\underline{x}, \underline{y})} \quad (7)$$

The computation of $T_j(\cdot, \cdot)$ is a little complicated. First, note that \underline{x} and \underline{y} differ

only in the j -th chromosome. So, if j is in the boundary of the temperature ladder i.e. either $j = 1$ or $j = N$ then there could possibly be only one plausible parent of j , namely, $j = 2$ for $j = 1$ and $j = N - 1$ for $j = N$. Otherwise, any of the possible neighbors of j i.e. either $j - 1$ or $j + 1$ could be its parent. With this in mind, we define the indicator function $h(\underline{x}_i, \underline{x}_j, \underline{y}_j)$ which takes the value 1 if \underline{y}_j is a modified child from parent \underline{x}_i and child \underline{x}_j and is 0 otherwise. Now, let $g(i, j, \underline{x}) = p(I_1 = i | \underline{x}) \times p(I_2 = j | \underline{x}, I_1 = i) \times p(I_3 = i | \underline{x}, I_1 = i, I_2 = j) + p(I_1 = j | \underline{x}) \times p(I_2 = i | \underline{x}, I_1 = j) \times p(I_3 = i | \underline{x}, I_1 = j, I_2 = i)$. Thus $g(i, j, \underline{x})$ denotes the probability of choosing i as the modified child where the parents are \underline{x}_i and \underline{x}_j . With these notations we have:

$$T_j(\underline{x}, \underline{y}) = \begin{cases} g(1, 2, \underline{x}) & \text{for } j = 1 \\ g(N, N - 1, \underline{x}) & \text{for } j = N \\ \{g(j - 1, j, \underline{x}) \times h(\underline{x}_{j-1}, \underline{x}_j, \underline{y}_j) + \\ g(j + 1, j, \underline{x}) \times h(\underline{x}_{j+1}, \underline{x}_j, \underline{y}_j)\} & \text{for } 1 < j < N \end{cases} \quad (8)$$

Note the property of leaving one of the parents unchanged of the present move is very desirable. In certain situations, if both the parents get changed to produce two new children (as in the SCSC:TWO-NEW move in section 3.3) and both the children happen to be “bad” samples i.e. have high “fitness” value then such proposal moves may get rejected. For a discussion on this problem in the context of real crossover see the section 2.2 of [7, Goswami et. al.].

3.5 Sub Cluster Reallocation Crossover (SCRC)

Unlike in the SCSC moves as introduced in the previous sections, here we try to reallocate the members of two randomly chosen sub-clusters of one randomly chosen chromosome “guided” by another randomly chosen chromosome in a clever fashion and hence the name of the move. The first few steps involved in this move are exactly the same ones as required for the SCSC:ONE-NEW move as described in section 3.4. For the sake of brevity we avoid repetition and just mention that, following the steps of section 3.4, we choose I_1, I_2, I_3 and also choose k_1 and $l_1 \neq l_2$ randomly such that $A_{k_1} \cap B_{l_i} \neq \emptyset$, $i = 1, 2$. At this point on we divert from section 3.4 and do the following instead.

We consider the set $H := (A_{k_1} \cap B_{l_1}) \uplus (A_{k_1} \cap B_{l_2})$ and randomly divide this set into non-empty sub-sets, say, H_1 and H_2 i.e. we reallocate the members of $(A_{k_1} \cap B_{l_1})$ and $(A_{k_1} \cap B_{l_2})$. Let $m_i := |A_{k_1} \cap B_{l_i}|$ and $h_i := |H_i|$, $i = 1, 2$. Note by definition, $h_1 + h_2 = m_1 + m_2$. Now we form the following disjoint collection of sets $\{S_{k,l} | k = 1, 2, \dots, K, l = 1, 2, \dots, L\}$ where,

$$S_{k,l} = \begin{cases} H_1 & \text{for } k = k_1, l = l_1 \\ H_2 & \text{for } k = k_1, l = l_2 \\ A_k \cap B_l & \text{otherwise} \end{cases} \quad (9)$$

and define $C_k = \uplus_{l=1}^L S_{k,l}$, $k = 1, 2, \dots, K$ and $D_l = \uplus_{k=1}^K S_{k,l}$, $l = 1, 2, \dots, L$. Now let the cluster indicator vector formed by $\mathcal{C} := \{C_k \mid k = 1, 2, \dots, K\}$ and $\mathcal{D} := \{D_l \mid l = 1, 2, \dots, L\}$ be \underline{y}_i and \underline{y}_j respectively. Note that by construction $\underline{y}_i = \underline{x}_i$. As in section section 3.4, we call, \underline{y}_j is a “modified child” from parent \underline{x}_i and child \underline{x}_j to mean that \underline{y}_j could be produced by reallocation and the set operations described above in equation (9) by “crossing” the parent \underline{x}_i and the child \underline{x}_j . A diagrammatic representation of the proposed strategy is shown in figure 3. Now, propose the new population $(\underline{y}, \underline{t}) = (\underline{x}_1, t_1; \dots; \underline{y}_j, t_j; \dots; \dots; \underline{x}_N, t_N)$ and accept this new population with probability $\min(1, r_{scrc})$ where,

$$r_{scrc} = \frac{f_j(\underline{y}_j)}{f_j(\underline{x}_j)} \times \frac{T_j(\underline{y}, \underline{x})}{T_j(\underline{x}, \underline{y})} \times \frac{S_j(\underline{y}, \underline{x})}{S_j(\underline{x}, \underline{y})} \quad (10)$$

The computation of $T_j(\cdot, \cdot)$ is again same as in equation (8) of section 3.4; only caveat being that the word “modified child” has a different meaning for this subsection.

The computation of $S_j(\cdot, \cdot)$ requires we note the following. One could carry out the reallocation operation i.e. dividing H into H_1 and H_2 in many different ways. One strategy could be to make sure that “sizes remain unaltered” i.e. to say we require $\{h_1, h_2\} = \{m_1, m_2\}$. Another strategy could be to pick a random size, say, h_1 from $\{1, 2, \dots, |H| - 1\}$ and then produce H_1 and H_2 with $|H_1| = h_1$ and $|H_2| = |H| - h_1$. We will call the first strategy SCRC:SAME-SIZE and the second one SCRC:RANDOM-SIZE, which we think are quite self-explanatory. In case of SCRC:RANDOM-SIZE we have, $S_j(\underline{y}, \underline{x})/S_j(\underline{x}, \underline{y}) = \left(1/\binom{h_1+h_2}{m_1}\right) / \left(1/\binom{m_1+m_2}{h_1}\right) = \frac{m_1!m_2!}{h_1!h_2!}$, where $\binom{n}{r} := \frac{n!}{r!(n-r)!}$, the usual combination coefficient. Note, terms cancel out because as noted before $h_1 + h_2 = m_1 + m_2$. Now, the function $k(n) := \binom{n}{r}$ as a function of r is maximized for $r = \lfloor n/2 \rfloor$. Thus if $h_1 \approx h_2$ i.e. for any equal reallocation of objects, $\binom{h_1+h_2}{h_1} \geq \binom{h_1+h_2}{m_1}$ and hence the last ratio is going to be ≥ 1 . Thus, SCRC:RANDOM-SIZE prefers equal reallocation of members. In case of SCRC:SAME-SIZE, since $h_1 = m_1$, we have $S_j(\underline{y}, \underline{x})/S_j(\underline{x}, \underline{y}) = \binom{h_1+h_2}{h_1} / \binom{m_1+m_2}{m_1} = 1$.

In essence, this move is very similar to SCSC:ONE-NEW, the main difference being the way we produce the modified child in the crossover operation. Thus we may expect to see all the vice and virtues of the ONE-NEW move shared by this move.

3.6 Random Exchange (RE)

In this step, we propose to exchange a randomly chosen chromosome with one of its neighbors. Here we randomly select $i \in \{1, 2, \dots, N\}$ i.e. we use $p(I_1 = i|\underline{x}) = \frac{1}{N}$. Then we select $j = i \pm 1$ s.t. $p(I_2 = i \pm 1|\underline{x}, I_1 = i) = 0.5$, $p(I_2 = 1|\underline{x}, I_1 = 0) = 1$ and $p(I_2 = N - 1|\underline{x}, I_1 = N) = 1$. We accept the new population $(\underline{y}, \underline{t}) = (\underline{x}_1, t_1; \dots; \underline{x}_j, t_j; \dots; \underline{x}_i, t_i; \dots; \underline{x}_N, t_N)$, with probability $\min(1, r_{re})$ where,

$$r_{re} = \frac{f_i(\underline{x}_j)f_j(\underline{x}_i)}{f_i(\underline{x}_i)f_j(\underline{x}_j)} \times \frac{T_{i,j}(\underline{y}, \underline{x})}{T_{i,j}(\underline{x}, \underline{y})} \quad (11)$$

Here $T_{i,j}(\underline{y}, \underline{x}) = p(I_1 = i|\underline{x})p(I_2 = j|\underline{x}, I_1 = i) + p(I_1 = j|\underline{x})p(I_2 = i|\underline{x}, I_1 = j)$. Due to the structure of the selection mechanism, the second term in the product of the above acceptance probability drops out. Hence, we note that $r_{re} = \exp[(H(\underline{x}_j) - H(\underline{x}_i)) \cdot (1/t_j - 1/t_i)]$. So, if by random chance, we happen to choose i and j such that $i > j$ with $H(\underline{x}_j) \leq H(\underline{x}_i)$ then the due to the above noted form of r_{re} and the fact that the temperature ladder is decreasing we would have $r_{re} \geq 1$. Thus one of the virtues of random exchange is that it tries to bring *good* samples (i.e. samples with lower fitness value or high likelihood value) down the ladder.

4 The PBC Algorithm

With the above different kinds of moves the PBC algorithm is defined as follows. First we initialize the population to $\{\underline{x}_i^{(0)}, i = 1, 2, \dots, N\}$ with a carefully crafted temperature ladder $\{t_i, i = 1, 2, \dots, N\}$. We also fix a proportion $p_m \in (0, 1)$ which is called the mutation rate. Then one iteration of PBC consists of the following sequence of moves:

Algorithm 4.1 (PBC).

1. We apply the mutation move to all the chromosomes and N -many (one of the TWO-NEW, ONE-NEW, SCRC crossover moves) to the population with probability p_m and $1 - p_m$ respectively.
2. We apply N -many random exchange moves on the resultant population.

With probability p_m , we use the mutation move, where all of the chromosomes of the population are chosen systematically and updated once. With probability $1 - p_m$, we apply one of SCSC:TWO-NEW, SCSC:ONE-NEW and SCRC move N times, i.e., we do not mix these three different kinds of moves in one run. At the end of the iterations the we get required samples form the target distribution: $\{x_N^{(t)}; t = 1, 2, \dots\}$.

There are many tunable aspects of this algorithm. We discuss their choice and effect on the performance below. We suggest taking $p_m \in (25\%, 40\%)$. Mutation operator updates the chromosome “locally”, whereas the SCSC, SCRC and RE operators facilitate “global” exploration and by choosing $p_m < 1/2$ we are stressing on global moves more than local moves.

Lastly, for a general discussion on the choice of the temperature ladder and related issues see section 3 of [13, Liang et. al., 2001]. For a specific recipe, for constructing the temperature ladder see the section 5 of [7, Goswami et. al.]. In all the examples considered below we constructed the temperature ladders using the above mentioned strategy. Briefly, this recipe consists of the two preliminary runs. First preliminary run samples help us determine $\tau_{max}(= \tau_1)$; note τ_{min} is dictated by the problem. The second preliminary run samples are used to determine N , the number intermediate temperatures to place $\tau_2, \tau_3, \dots, \tau_{N-1}$ between τ_{max} and τ_{min} . The criterion used for placing the intermediate temperatures is $\hat{A}_{i,i+1} \in (60\%, 70\%), i = 1, 2, \dots, N-1$. Here $\hat{A}_{i,i+1}$ is the estimated average acceptance probability of the RE moves between the distributions $f_i(\cdot)$ and $f_{i+1}(\cdot)$. In other words $\hat{A}_{i,i+1}$ estimates (compare the following equation with equation (11)):

$$A_{i,i+1} = \int \min \left\{ 1, \frac{f_i(x_{i+1}) f_{i+1}(x_i)}{f_i(x_i) f_{i+1}(x_{i+1})} \right\} \cdot f_i(x_i) f_{i+1}(x_{i+1}) dx_i dx_{i+1} \quad (12)$$

Some notational conventions are in order now. By **TWO-NEW-r-r** we denote a run where we randomly select I_1 and I_2 for SCSC:TWO-NEW crossover. On the other hand, **TWO-NEW-b-b** denotes a run where we take $p(I_1 = i|x) \propto \exp\{-H(x_i)/s\}$ and $p(I_2 = j|x, I_1 = i) \propto \exp\{-H(x_j)/s\}$ for some selection temperature $s > 0$. Here **-b-b** is used to reflect the fact that we are probabilistically choosing the two “best” chromosomes x_i and x_j with respect to temperature s (see section 3.3). Unless mentioned otherwise we took $s = 1$ in the following examples. Similar explanations hold for **ONE-NEW-r-r** and **ONE-NEW-b-b** with regards to choice of I_1 and I_3 (see section 3.4).

Also, we use **RANDOM-SIZE** and **SAME-SIZE** for the two flavors of SCRC, namely, SCRC:RANDOM-SIZE and SCRC:SAME-SIZE respectively. In the light of some simulation studies (not included here), we choose I_1 and I_3 for both **RANDOM-SIZE** and **SAME-SIZE** the same way we do in **ONE-NEW-b-b** (see section 3.5). Lastly, by the term family of PBC algorithms we will refer to all the six algorithms introduced above, namely, ***-NEW-*-***, ***-SIZE**.

The algorithm was implemented in **C** and the output was analyzed in **R**. Only the first author is responsible for any bug in the **C** code despite the fact that it has gone through intensive debugging a number of times.

5 Dirichlet Process Mixture Model

Dirichlet process mixture models are used to introduce hierarchical structure in a mixture of distributions set up. For this discussion, we follow the notation of [10, Jain et. al., 2004]. Let our data $\underline{y} := (y_1, y_2, \dots, y_n)$ come from a mixture of distributions of the form $F(\cdot)$ where the mixing distribution is $G(\cdot)$, i.e., we have $y_i | \theta_i \stackrel{i.i.d.}{\sim} F(\theta_i)$ and $\theta_i | G \stackrel{i.i.d.}{\sim} G$. On top of this, we assume that the mixing distribution $G(\cdot)$ itself has a Dirichlet process prior with the baseline measure G_0 and the total mass parameter $\alpha (> 0)$ i.e. we assume $G \sim DP(G_0, \alpha)$. For an introduction to Dirichlet process prior see [3, Ferguson, 1974]. Here G_0 and α are the hyper parameters to be set by the practitioner. Note, lower values α correspond to smaller number of sub-clusters preferred to by the prior.

A different formulation of the above set up due to [1, Blackwell et. al, 1973] is given below. We will be using this formulation in the examples to follow. Here we first generate a cluster indicator vector \underline{z} from the following density for $i > 1$

$$P(z_i = z | z_j, j = 1, 2, \dots, i-1) = \frac{n_{i,z}}{i-1 + \alpha} \quad \text{for } z \in \{z_j | j = 1, 2, \dots, i-1\}$$

$$P(z_i = z | z_j, j = 1, 2, \dots, i-1) = \frac{\alpha}{i-1 + \alpha} \quad \text{for } z \notin \{z_j | j = 1, 2, \dots, i-1\}$$

Note, here z_1 is some value in D and $n_{i,z} = |\{z_j | z_j = z, j = 1, 2, \dots, i-1\}|$. Given this cluster indicator \underline{z} we form the sub-clusters, say, $\mathcal{H} = \{H_s | s = 1, 2, \dots, S\}$ and draw one θ value for each sub cluster i.e. draw $\theta_{H_s} \stackrel{i.i.d.}{\sim} G_0, s = 1, 2, \dots, S$. Then we take $\forall s, \{y_i | i \in H_s\} \stackrel{i.i.d.}{\sim} F(\theta_{H_s})$. Thus, the Dirichlet process induced prior, the likelihood (integrating out the θ 's) and the resulting posterior on the \underline{z} takes the following form:

$$p(\underline{z}) = \alpha^S \frac{\prod_{s=1}^S (|H_s| - 1)!}{\prod_{i=1}^D (\alpha + i - 1)} \quad (13)$$

$$p(\underline{y} | \underline{z}) = \int p(\underline{y}, \underline{\theta} | \underline{z}) d\underline{\theta}$$

$$= \prod_{s=1}^S \int \left[\left\{ \prod_{i \in H_s} F(y_i | \theta_{H_s}) \right\} G_0(\theta_{H_s}) d\theta_{H_s} \right] \quad (14)$$

$$p(\underline{z} | \underline{y}) \propto p(\underline{z}) \times p(\underline{y} | \underline{z}) \quad (15)$$

6 Gibbs Sampling

We implement the Gibbs sampler for the posterior $p(\underline{z}|\underline{y})$ we proceed in the following way. We first fix $i \in D$. We want $p(z_i | \underline{z}_{-i}, \underline{y})$ where $\underline{z}_{-i} := \{z_j \mid j \in D, j \neq i\}$. Let the sub-cluster representation of \underline{z}_{-i} be $\tilde{\mathcal{H}} := \{\tilde{H}_s \mid s = 1, 2, \dots, S\}$. Also, let $p(\underline{y}|\tilde{H}_s)$ be the likelihood for the sub-cluster \tilde{H}_s . Now we note, for $s_0 \notin \{z_j \mid j \neq i\}$, $\tilde{H}_{s_0} = \{i\}$. So we have,

$$\begin{aligned} p(z_i = s_0 \mid \underline{z}_{-i}, \underline{y}) &\propto p(z_i = s_0, \underline{z}_{-i}, \underline{y}) \\ &\propto p(\underline{y} \mid z_i = s_0, \underline{z}_{-i})p(z_i = s_0, \underline{z}_{-i}) \\ &\propto p(\underline{y} \mid z_i = s_0, \underline{z}_{-i})p(z_i = s_0 \mid \underline{z}_{-i}) \\ &\propto p(\underline{y}_i | \tilde{H}_{s_0})\alpha \end{aligned}$$

Now we take $s \in \{z_j \mid j \neq i\}$. Note $s \neq s_0$. Here, we first compute the ratio

$$q(s, s_0) := \frac{p(z_i=s|\underline{z}_{-i},\underline{y})}{p(z_i=s_0|\underline{z}_{-i},\underline{y})} = \frac{p(\underline{y}|z_i=s,\underline{z}_{-i})p(z_i=s|\underline{z}_{-i})}{p(\underline{y}|z_i=s_0,\underline{z}_{-i})p(z_i=s_0|\underline{z}_{-i})} = \frac{p(\underline{y}|\tilde{H}_s \cup \{i\})|\tilde{H}_s|}{p(\underline{y}|\tilde{H}_s)p(\underline{y}_i|\tilde{H}_{s_0})\alpha}$$

In the examples to follow the above ratios turns out be very easy to compute. With this ratios in hand, we finally consider:

$$p(z_i = s \mid \underline{z}_{-i}, \underline{y}) = q(s, s_0) \cdot p(z_i = s_0 \mid \underline{z}_{-i}, \underline{y}) \propto \frac{p(\underline{y}|\tilde{H}_s \cup \{i\})|\tilde{H}_s|}{p(\underline{y}|\tilde{H}_s)}$$

7 Examples

In the following subsections we will consider several examples which illustrate the improvement of the PBC algorithm over the existing methods. We will compare the performance of various flavors of the PBC algorithm with some baseline methods, namely, Gibbs sampler (section 6) and the Metropolized Gibbs sampler (section 3.1). We will use `Gibbs` and `MH` to refer to these baseline schemes.

The simulation parameters for the different examples below are the same as the ones in the paper(s) where they first appeared, unless mentioned otherwise. We also used the same set of starting values for corresponding runs and so, e.g. run 5, say, of all the algorithms were started using the same set of values. We kept track of the computational cost by running all the algorithms for the same amount of CPU time

on a cluster of LINUX boxes. In some simulations, however, we used fixed number of draws instead of fixed amount of computational cost, the reasons for doing so will be indicated in the relevant context.

We took the split probability $q_s = 1/2$ (see section 3.1) for the mutation step. The burn-in period for a chain producing T draws was taken to be $\min(\lfloor T/4 \rfloor, 20000)$ in all the following examples. In the following, we are going to use T for the number of draws from an algorithm *after the mentioned burn-in* period. Geyer in [6, Geyer, 1992] argues that less than 5% of the total number of draws are usually enough for burn-in. We decided to throw away a lot more number of draws than 5% just to be on the safe side.

Building on [10, Jain et. al., 2004] and [2, Dahl, 2003], we look at the *Average Integrated Autocorrelation Time (AIAT)* of several statistics computed from the MCMC samples to compare the performance of the various algorithms. For $R(\in \mathbb{N})$ different independent runs, the *AIAT*, for a given statistic, is defined as $AIAT_R := \frac{1}{R} \sum_{i=1}^R \hat{\tau}_{T_i}$ where $\hat{\tau}_{T_i}$ is the Integrated Autocorrelation Time (*IAT*) computed from the T_i many values of the statistic (which in turn is computed from the T_i many cluster indicator samples) from the i -th run of the chain. For a detailed discussion on computation of *IAT* refer to the section 5 of [7, Goswami et. al.]. Briefly, we computed $\hat{\tau}_T$ following way. This definition of *IAT* comes from [6, Geyer, 1992]. For a one-dimensional series $\{u_t \mid t = 1, 2, \dots, T\}$ of draws from a reversible Markov chain with sample auto-covariances $\{\hat{\gamma}_j \mid j = 0, 1, \dots, (T-1)\}$, the monotone estimator of the variance of the sample mean is defined as

$$\hat{\sigma}_{mono,T}^2 = -\hat{\gamma}_0 + 2 \times \sum_{j=0}^m \hat{\Gamma}_j^*$$

where $\hat{\Gamma}_j^* = \min \{ \hat{\Gamma}_0, \hat{\Gamma}_1, \dots, \hat{\Gamma}_j \}$ with $\hat{\Gamma}_j = \hat{\gamma}_{2j} + \hat{\gamma}_{2j+1}, j = 0, 1, \dots$ and m is such that $\hat{\Gamma}_{m+1} \leq 0$ for the first time. This definition is dictated by some interesting properties of the population auto-covariances γ_j 's. We refer the interested reader to the source cited above for details. Now, we take, $\hat{\tau}_T := \hat{\sigma}_{mono,T}^2 / \hat{\gamma}_0$. For $AIAT_R$, we will look at some of several one-dimensional statistics of a cluster indicator vector \mathbf{z} (with sub-cluster representation, $\mathcal{H} := \{H_s \mid s = 1, 2, \dots, S\}$) from the following list.

- Number of distinct sub-clusters: $n(\mathbf{z}) := S(= |\mathcal{H}|)$
- Proportion of observations the largest sub-cluster: $p_{max}(\mathbf{z}) := d_{max}(\mathbf{z})/d$, where $d_{max}(\mathbf{z}) := \max\{|H_s| \mid s = 1, 2, \dots, S\}$
- The log density: $l(\mathbf{z}) := -H(\mathbf{z})$, where $H(\cdot)$ the “fitness” function is as introduced in equation (1).

- Entropy of the sub-clusters: $e(\mathbf{z}) := -\sum_{s=1}^S \frac{|H_s|}{d} \log\left(\frac{|H_s|}{d}\right)$, see section 5.1 of [2, Dahl, 2003]

We also define the *Average Maximum Log Density* (AMLD) achieved over several independent runs as $AMLD_R := \frac{1}{R} \sum_{i=1}^R \max_{j=1,2,\dots,T_i} l(z_{ij})$ where $l(z_{ij})$ is the log density value for the j -th of the T_i many samples from the i -th run. We consider this statistic because it gives us a comparative idea of which algorithm(s) are more capable of escaping “local” modes.

7.1 A Simulation Study with Bernoulli-Beta Clustering

This example is based on the example considered in section 4 of [10, Jain et. al., 2004]. Here we consider clustering of Bernoulli data with a conjugate Beta prior. We have independent data points $\mathbf{y} = (\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_d)$ such that each observation $\mathbf{y}_i = (y_{i1}, \dots, y_{im})$, $\forall i$ with $y_{ih} \mid \theta_{ih} \stackrel{i.i.d.}{\sim} \text{Bernoulli}(\theta_{ih})$, $\forall i, h$. We take $\theta_{ih} \stackrel{i.i.d.}{\sim} \text{Beta}(\beta_{1h}, \beta_{0h})$, $\forall i, h$. So, we have,

$$p(\mathbf{y}_i \mid \theta_i) = \prod_{h=1}^m \theta_{ih}^{y_{ih}} (1 - \theta_{ih})^{1-y_{ih}} \quad (16)$$

$$p(\theta_i) = \prod_{h=1}^m \frac{\Gamma(\beta_{1h} + \beta_{0h})}{\Gamma(\beta_{1h})\Gamma(\beta_{0h})} \theta_{ih}^{\beta_{1h}} (1 - \theta_{ih})^{\beta_{0h}} \quad (17)$$

Here $\beta_{1h}, \beta_{0h} > 0, \forall h$. So, in the notation of section 5, $F(\cdot)$ and $G_0(\cdot)$ are given by equations (16) and (17) respectively. With this structure our likelihood of the clusters indicators given the data take the following form (see equation (15)):

$$p(\mathbf{y} \mid \mathbf{z}) = \prod_{s=1}^S \prod_{h=1}^m \frac{\Gamma(x_{sh} + \beta_{1h})\Gamma(n_s - x_{sh} + \beta_{0h})\Gamma(\beta_{1h} + \beta_{0h})}{\Gamma(\beta_{1h})\Gamma(\beta_{0h})\Gamma(n_s + \beta_{1h} + \beta_{0h})} \quad (18)$$

Here $x_{sh} := \sum_{i \in H_s} y_{ih}$ and $n_s := |H_s|$ where $\mathcal{H} := \{H_s \mid s = 1, 2, \dots, S\}$ is the sub-cluster representation of \mathbf{z} . Combining the above with the prior from equation (13) above we get the required posterior $p(\mathbf{z} \mid \mathbf{y}) \propto p(\mathbf{y} \mid \mathbf{z})p(\mathbf{z})$ which becomes the density $g(\cdot)$ of equation (1) in the PBC context. For Gibbs sampling we note for $i \in D$ we have (see section 6),

$$p(z_i = s \mid \mathbf{z}_{-i}, \mathbf{y}) \propto \begin{cases} \frac{n_s - 1}{d - 1 + \alpha} \prod_{h=1}^m \frac{\sum_{k \in H_s, k \neq i} \delta(y_{kh}, y_{ih}) + \beta_{y_{ih}, h}}{n_s - 1 + \beta_{1h} + \beta_{0h}} & \text{if } s \in \{z_j \mid j \neq i\} \\ \frac{\alpha}{d - 1 + \alpha} \prod_{h=1}^m \frac{\beta_{y_{ih}, h}}{\beta_{1h} + \beta_{0h}} & \text{otherwise} \end{cases} \quad (19)$$

We choose $m = 15$ and use 5 distinct θ_i 's as shown in table 1 to simulate 20-many Y_i 's from each of these 5 $F(\cdot)$'s resulting in $d = 100$ many data points. See section 5 for validity of the simulation procedure. For the PBC family of algorithms the temperature ladder was of length 20 with $\tau_{max} = t_1 = 20$ as given in table 2. With this ladder structure the acceptance rates for the various methods mentioned above were as tabulated in table 3.

In all these cases the acceptance rate for the RE move was between 40% – 60%. The acceptance rates for the SCSC and SCRC family of moves are much smaller compared to the RE move. This could be explained by the fact that the temperature placement was done to make sure estimated average acceptance rate for the RE move was “reasonable” and the resulting temperature ladder turns out to be not so favorable to SCSC and SCRC family of moves. This phenomenon is problem specific. Moreover, the TWO-NEW-**-* moves have higher acceptance rates than the ONE-NEW-**-* and *-SIZE moves but again this is problem (and temperature ladder) specific as we will find out in section 7.2. Here, higher acceptance rates for TWO-NEW-**-* produces much lower $AIAT_{20}$ values as compared to those produced by ONE-NEW-**-* and *-SIZE.

We considered $R = 20$ independent runs. We summarize the results from the runs in table 4. Since they are cheap, in the same amount of CPU time, Gibbs and MH produced around 10 times more number of samples than the PBC family of schemes. From the mentioned table we observe that MH and Gibbs perform more or less comparably, MH doing a little better with respect to the ability to escape local modes. Even though these two schemes produce many more samples, all the schemes in the PBC family of algorithms perform better than them both in terms of $AIAT$ s and $AMLD$ s.

7.2 Motif Clustering

We consider a simplified version of a motif clustering problem originally introduced in [11, Jensen et. al., 2005]. We have a bunch of motif matrices $(\mathbf{Y}_1, \mathbf{Y}_2, \dots, \mathbf{Y}_d)$ of fixed width ω ; i.e. all of these are matrices of dimension $\omega \times 4$. Here Y_{ijk} is the count for the nucleotide $k (\in \{A, G, T, C\})$ for the column $j (\in \{1, 2, \dots, \omega\})$ in the motif matrix $i (\in \{1, 2, \dots, d\})$. Let $d_i := \sum_k Y_{ijk}$, $i = 1, 2, \dots, d$. Note, d_i as defined before should not depend on j because all the column sums within a motif matrix \mathbf{Y}_i , $\forall i$ are the same. We assume a product multinomial model for the columns of \mathbf{Y}_i 's i.e. we assume that $p(\mathbf{Y}_i | \Theta_i) = \prod_{j=1}^{\omega} p(\mathbf{Y}_{ij} | \theta_{ij})$ with $\mathbf{Y}_{ij} | \theta_{ij} \stackrel{ind}{\sim} \text{Multinomial}(d_i, \theta_{ij})$. We take a four dimensional Dirichlet distribution for the parameters, namely, we take $\forall i, j$, $\theta_{ij} \stackrel{i.i.d.}{\sim} \text{Dirichlet}(c, c, c, c)$ with $c > 0$. So, we have,

$$p(\mathbf{Y}_{ij} | \theta_{ij}) = \frac{d_i!}{\prod_k Y_{ijk}!} \theta_{ijk}^{Y_{ijk}} \quad (20)$$

$$p(\boldsymbol{\theta}_{ij}) = \frac{\Gamma(4c)}{\Gamma^4(c)} \prod_k \theta_{ijk}^{c-1} \quad (21)$$

So, in the notation of section 5, we take equations (20) and (21) for $F(\cdot)$ and $G_0(\cdot)$ respectively. With this structure our likelihood of the clusters indicators given the data take the following form (see equation (15)):

$$p(\mathbf{Y} | \mathbf{z}) = \prod_{s=1}^S \prod_{j=1}^{\omega} \left[\left\{ \prod_{i \in H_s} \frac{d_i!}{\prod_k Y_{ijk}!} \right\} \times \left\{ \prod_k \frac{\Gamma(X_{sjk} + c)}{\Gamma(c)} \right\} \times \frac{\Gamma(4c)}{\Gamma(\sum_k X_{sjk} + 4c)} \right] \quad (22)$$

Here $X_{sjk} := \sum_{i \in H_s} Y_{ijk}$ where $\mathcal{H} := \{H_s \mid s = 1, 2, \dots, S\}$ is the sub-cluster representation of \mathbf{z} . Combining the above with the prior from equation (13) above we get the required posterior $p(\mathbf{z} | \mathbf{Y})$ which becomes the density $g(\cdot)$ of equation (1) in the PBC context. For Gibbs sampling we note for $i \in D$ we have (see section 6),

$$p(z_i = s \mid \mathbf{z}_{-i}, \mathbf{Y}) \propto \begin{cases} \frac{n_s - 1}{d - 1 + \alpha} \prod_{j=1}^{\omega} \frac{\prod_k \Gamma((Y_{ijk} + \tilde{X}_{sjk}) + c) \Gamma(\sum_k \tilde{X}_{sjk} + 4c)}{\prod_k \Gamma(\tilde{X}_{sjk} + c) \Gamma(\sum_k (Y_{ijk} + \tilde{X}_{sjk}) + 4c)} & \text{if } s \in \{z_j \mid j \neq i\} \\ \frac{\alpha}{d - 1 + \alpha} \prod_{j=1}^{\omega} \frac{\prod_k \Gamma(Y_{ijk} + c)}{\Gamma(\sum_k Y_{ijk} + 4c)} \cdot \frac{\Gamma(4c)}{[\Gamma(c)]^4} & \text{otherwise} \end{cases} \quad (23)$$

Here, we use $\tilde{X}_{sjk} := \sum_{i \in \tilde{H}_s} Y_{ijk}$ where $\tilde{\mathcal{H}} := \{\tilde{H}_s \mid s = 1, 2, \dots, S\}$ the sub-cluster representation of $\mathbf{z}_{-i} := \{z_j \mid j \neq i\}$. We use a data set with $d = 90$ aligned motif matrices \mathbf{Y}_i 's each of width $\omega = 8$. These data were provided by Dr. Shane Jensen (see [11, Jensen et. al] for details) who used his clustering algorithm to align the matrices. For the PBC family of algorithms the temperature ladder was of length 33 with $\tau_{max} = t_1 = 60$. With this ladder the acceptance rates for the various methods mentioned above were as tabulated in table 5.

In all these cases the acceptance rate for the RE move was between 50% – 60%. Here the acceptance rates for all the PBC family of moves are pretty low as compared to that of RE move. Although, as can be observed it is no longer the case (compare with section 7.1) that **TWO-NEW-***** moves have better acceptance rates than **ONE-NEW-***** or ***-SIZE** moves. Notably, here the ***-SIZE** moves have the best acceptance rates and thus producing the the least set of $AIAT_{10}$ values. So, whether letting one of the parents stay as parent i.e. not allowing “too much change in the population”, as is induced by ***-SIZE** (and **ONE-NEW-*****) moves, is a good strategy or not depends entirely on the problem and possibly on the temperature ladder.

We summarize the results from $R = 10$ independent runs in table 6. In the same amount of CPU time, **Gibbs** and **MH** produced around 8 times more number of samples

than the PBC family of schemes all of which produced around 3000 draws. From the mentioned table we observe that `Gibbs` performs much worse than `MH` with respect to the ability to escape local modes and in turn all the PBC family of the schemes perform much better than `MH`. For a clear picture we produce figure 4. There we observe that in all the $R = 10$ independent runs all the PBC family of schemes reach a mode with log density value -6431.527 where as `MH` fails to achieve the same maximum almost half the times even with 8 times more number of draws. `Gibbs` turns out to be really worse in this case failing to reach this mode even once. Moreover, with respect to `AIAT` we see all the PBC schemes outperform both `Gibbs` and `MH`.

Now the performance of `Gibbs` in the above mentioned figure 4 might raise eyebrows. One might tend to think that either there is something wrong in the “normalizing constant” computation somewhere in the Gibbs conditionals or it might be completely due to a bug in the code. To test these hypotheses we ran all the algorithms on a smaller subset of the above discussed real data set consisting of only 20 motifs (as opposed to 90 motifs). In that case, in the same amount of CPU time, `Gibbs` produced around 70000 draws where as PBC family schemes gave us around 4000 samples in each run. In this case, all the methods in $R = 10$ different runs produced the same MAP (Maximum A-Posteriori) estimate i.e. the same posterior mode with the same cluster indicator vector. So, we could say that the figure 4 is a scenario where for a larger data set, in a given amount of time, Gibbs gets trapped in a local mode in all runs whereas in the same amount of time `MH`, in some runs, is able to escape the trap but PBC family of methods are successful in doing so in every run. Thus, doing a lot of `Gibbs` or for that matter `MH` iterations may not be worth the effort, one might want to consider smarter ways (e.g. PBC methods) of exploring the space.

7.3 Normal Clustering

In this section, we investigate the stochastic optimization capabilities of the PBC algorithm in a multivariate Normal set up. Let we have m -dimensional independent data points $\mathbf{Y} = (\mathbf{Y}_1, \mathbf{Y}_2, \dots, \mathbf{Y}_d)$ where each data Y_i comes from a multivariate Normal distribution, namely, $\mathbf{Y}_i \mid \boldsymbol{\theta}_i \stackrel{i.i.d.}{\sim} \mathbf{Normal}_m(\boldsymbol{\theta}_i, \sigma^2 \mathbf{I}_m)$, where \mathbf{I}_m is the m -dimensional identity matrix. We assume that $\boldsymbol{\theta}_i \stackrel{i.i.d.}{\sim} \mathbf{Normal}_m(\boldsymbol{\mu}, \tau^2 \mathbf{I}_m)$. So, in the notation of section 5, these two Normal distributions correspond to $F(\cdot)$ and $G_0(\cdot)$ respectively.

We took $m = 2$, $\boldsymbol{\mu} = \mathbf{0}$, $\sigma^2 = 1$ and $\tau^2 = 30.0$. We generated 5 distinct $\boldsymbol{\theta}_i$'s from the above $G_0(\cdot)$ and simulated 40-many Y_i 's from each of these 5 $F(\cdot)$'s. Thus we have $d = 200$ data points. A sample data set simulated this way is shown in figure 5. See section 5 for validity of the simulation procedure.

7.3.1 Data Generation: Unknown

Here we assume that we don't know the details of the data generating process but the fact that there are 5 clusters in the data set, which is a more realistic situation. In such a scenario, the method of K-means clustering is widely used to cluster observations. So, we choose to consider K-means method as a competing method to PBC. In K-means clustering the objective is to minimize the “within sub-cluster sum of squares” for a given number of sub-clusters. We take the same objective function and formulate this minimization problem as a stochastic optimization (through sampling) problem by considering the following density for the cluster indicator vector z :

$$p(z | \mathbf{Y}) \propto \exp \left[-\frac{1}{\tau_{min}} \sum_{s=1}^S \sum_{i \in H_s} \|\mathbf{Y}_i - \bar{\mathbf{Y}}_s\|^2 \right] \cdot 1_{\{S=5\}}(z), \text{ with } \bar{\mathbf{Y}}_s := \frac{1}{|H_s|} \sum_{i \in H_s} \mathbf{Y}_i \quad (24)$$

In the above, the indicator function $1_{\{S=5\}}(z)$ makes sure that we only “walk” in the space of cluster indicators which have 5 sub-clusters. Now that in the above density the term $\sum_{s=1}^S \sum_{i \in H_s} \|\mathbf{Y}_i - \bar{\mathbf{Y}}_s\|^2$ in the exponent is the mentioned objective function for the K-means clustering problem. Here we take $\tau_{min} = 0.5$. The above density is our $g(\cdot)$ (see equation 1) in the PBC context.

Here the temperature ladder was of length 30 with $\tau_{max} = t_1 = 60$ (and $\tau_{min} = 0.5$). We ran all the PBC family of schemes $R = 50$ times independently for 1000 iterations each. For comparison we also ran the K-means algorithm with random starting values with maximum number of iterations set to 10000 (we used the `kmeans(x, centers, iter.max = 10000)` function in **R** for this). The summary comparison of these methods is given in table 7. The number 339.7051 has been subtracted from all the entries of this table for easy reading. This value corresponds to the minimum of $R = 50$ minimized value of the K-means objective function; it was achieved by at least one run for all the algorithms. We have the (“best”) clustering result corresponding to this value of the objective function in figure 6.

We can see the above mentioned table that the K-means results are very right skewed in the sense that even after (at the most 10000) many iterations it may get stuck in a local mode producing values of the objective function 1404.7676. We have the result of this the (“worst”) clustering result from this local mode in figure 7. Thus we have shown that even in this very simple example K-means clustering can fall into the local mode trap pretty often (at least more than 50% of the times, considering the value of the median in the row for K-means is table 7) whereas stochastic optimization through the PBC family of methods gives us much better results.

7.3.2 Data Generation: Known

Here we assume that we know the exact data generating process described in the beginning of section 7.3. The likelihood of the clusters indicators given the data integrating out the $\boldsymbol{\theta}$'s, as in equation (15), takes the following form:

$$p(\mathbf{Y} \mid \mathbf{z}) = \prod_{s=1}^S \frac{\sqrt{1/(|H_s|/\sigma^2 + 1/\tau^2)}}{(\sqrt{2\pi})^{|H_s|m} \sqrt{\sigma^2|H_s|m} \tau^{2m}} \times \exp \left[-\frac{1}{2} \left\{ \sum_{i \in H_s} \frac{\|\mathbf{Y}_i\|^2}{\sigma^2} + \frac{\|\boldsymbol{\mu}\|^2}{\tau^2} - \frac{\|\sum_{i \in H_s} \mathbf{Y}_i/\sigma^2 + \boldsymbol{\mu}/\tau^2\|^2}{|H_s|/\sigma^2 + 1/\tau^2} \right\} \right] \quad (25)$$

Now we take $\tau_{min} = 0.5$ and define $p(\mathbf{z} \mid \mathbf{Y}) \propto \{p(\mathbf{Y} \mid \mathbf{z})\}^{\tau_{min}} \cdot 1_{\{S=5\}}(\mathbf{z})$. This forms the target density $g(\cdot)$ as in equation (1). Note, as before, the indicator function above $1_{\{S=5\}}(\mathbf{z})$ makes sure that we only “walk” in the space of cluster indicators which have 5 sub-clusters. Here each algorithm was run $R = 10$ times for equal amount CPU time. In table 8 we have the results from these runs which show that except for ONE-NEW-r-r all the other PBC family of schemes perform better than MH as samplers (as measured by the $AIAT_{10s}$) but all the PBC schemes are better stochastic optimizers (as measured by $AMLD_{10}$) and hence in a sense better samplers.

7.4 Variable Selection

Here we consider a modified version of a problem which appeared in section 4.2 of [12, Liang et. al., 2000]. In the original example, the authors used 50 observations on 30 variables. We make the example harder by considering $p = 60$ variables and $n = 100$ observations instead. The original authors used Mallows's C_p as their model selection criterion. They chose to deal with $p = 30$ variables because they wanted to use the command `leaps(.)` in S-PLUS to find “the truth” or the global minimum C_p for the problem for comparison. As an aside, the `leaps(.)` command supports no more than 31 variables. However, in this example, we consider the Bayesian Information Criterion or BIC, in short, as our model selection criterion. We also forgo the opportunity of finding “the truth”, i.e., global minimum value of BIC by considering $p = 60$ because exhaustive or smart search in the space of $2^{60} - 1$ models is almost impossible. We restrict ourselves to the comparative performance of various algorithms in locating minimums.

Let $\mathbf{R}; (\mathbf{Y}_1^*, \mathbf{Y}_2^*, \dots, \mathbf{Y}_p^*) \stackrel{i.i.d.}{\sim} \text{Normal}_n(\mathbf{0}, \mathbf{I}_n)$, the n -dimensional standard Normal distribution. Let $\mathbf{Y}_i := \mathbf{Y}_i^* + 2 \cdot \mathbf{R}$, $i = 1, 2, \dots, p$ be our explanatory variables. This generation procedure induced a lot of correlation among the explanatory variables, a summary of the correlations appears in table 9. The dependent variable

was generated according to the model $\mathbf{Z} = \mathbf{Y} \boldsymbol{\beta} + \boldsymbol{\epsilon}$. Here $\mathbf{Y} := [\mathbf{Y}_1 : \mathbf{Y}_2 : \dots : \mathbf{Y}_p]$, is the matrix of explanatory variables. The coefficient vector $\boldsymbol{\beta}$ is given by $\beta_i = 0$, $i = 1, 2, \dots, 20$; $\beta_i = 1$, $i = 21, 22, \dots, 40$; $\beta_i = 2$, $i = 41, 42, \dots, 60$ and $\boldsymbol{\epsilon} \sim \mathbf{Normal}_n(\mathbf{0}, \sigma^2 \mathbf{I}_n)$, where $\sigma = 2.0$.

We are interested in finding the minimum BIC model. Toward that end, we use our cluster indicator vector \mathbf{z} of length p to represent a model as follows. $z_i = 1$ indicates that the explanatory variable \mathbf{Y}_i is in the model and $z_i = 0$ says that the explanatory variable \mathbf{Y}_i has been excluded from it. Let $\mathcal{A} := \{A_0, A_1\}$ represent the sub-clusters formed by the cluster indicator vector \mathbf{z} . Note in this set up, we will always have two sub-clusters A_0 and A_1 and also have $A_1 \neq \emptyset$ for we do not consider $\mathbf{z} = (0, 0, \dots, 0)$ to be a meaningful model. This is so because we do not include an extra ‘‘intercept’’ term in the models and hence a model with no explanatory variables does not deserve consideration. Here we take the following as our target density $g(\cdot)$, see equation (1):

$$p(\mathbf{z} | \mathbf{Y}, \mathbf{Z}) \propto \exp \left[-\frac{1}{\tau_{min}} \left\{ n \cdot \log \left(\frac{\|\mathbf{Z} - \widehat{\mathbf{Z}}_{A_1}\|^2}{n} \right) + |A_1| \cdot \log(n) \right\} \right] \cdot \mathbf{1}_{\{A_1 \neq \emptyset\}}$$

Here $\tau_{min} = 0.5$, $|A_1|$ is the number of explanatory variables in the model and $\widehat{\mathbf{Z}}_{A_1}$ is the (Ordinary Least Square) prediction vector of \mathbf{Z} based on all the explanatory variables in the model, i.e., $\{\mathbf{Y}_i, i \in A_1\}$. In the original article mentioned above the authors used, what they called, the Evolutionary Monte Carlo algorithm to solve their problem. The essential difference between their algorithm and ours is that instead of the PBC family of moves they had the so called Crossover family of moves (see section 3.2 of [12, Liang et. al., 2000]). They noted that the EMC algorithm was a better stochastic optimizer than the well known algorithms like Parallel Tempering and Reversible Jump MCMC.

To compare the relative performance of our algorithm to the EMC algorithm we implemented one of the moves from the Crossover family of moves, namely, the 1-point crossover move, chose $p_m = 0.3$ in algorithm 4.1 above and replaced the N -many SCSC or SCRC moves by the same number of 1-point crossover moves. The highest temperature was found to be $\tau_{max} = 25$ (and of course $\tau_{min} = 0.5$) and it was of length 18. Both the EMC and the three flavors of the PBC algorithm were run 10 times each. We considered only three representatives from the PBC family of methods as opposed to all six of them because in previous examples we have seen that their performance was more or less comparable.

The comparative performance of the methods could be found in table 10. All the PBC family of methods listed in there and EMC produced the same minimum, namely, 251.376 in all the 10 independent runs. The main observation from this table is that EMC and the PBC family of methods perform equally well both in terms of $AIAT$ and $AMLD$, which is not surprising because EMC is a powerful method to begin with.

As an important aside, the minimum BIC achieved by all the algorithms above, namely, 251.376, is quite close to or in fact the global minimum. We believe so because the sampled underlying model which achieves BIC 251.376 is quite close to the “truth”, i.e., close to the model used to generate the data. This sampled “mode” consists of $\{\mathbf{Y}_i \mid i = 21, 22, \dots, 60\} \cup \{\mathbf{Y}_2\}$ which is exactly the data generating model with an extra variable $\{\mathbf{Y}_2\}$.

Our main objective in this example is not to show that “PBC is better than EMC”. We just want to stress the fact that both of these methods are much very powerful methods and they both are better than vanilla MH or Gibbs sampling. Firstly, the moves of EMC are general purpose and thus they do not take advantage of the special structure of the clustering indicator vectors. Secondly, EMC uses the 1-point crossover move and as was alluded at the end of section 3.4, the 1-point crossover (the general real-parameter version of this is known as the Real Crossover) move has some inherent problems. So, for some problems e.g. the one considered in this subsection, EMC may do its charm but in others most of the proposed moves generated by it may be rejected. For a detailed discussion on this aspect in a slightly different context of Real Crossover see the end of section 2.2 of [7, Goswami et. al.].

8 Conclusion

We have demonstrated in this article that the PBC family of schemes have indeed much to offer although seemingly they demand more computational enterprise. Barring the fact that, implementing PBC is a challenging programming exercise, the same amount of computational cost buys us “better” samples. Note that, both Gibbs and MH are much simpler and cheaper algorithms and that’s why in the same amount of computing time they produce roughly 8 and 10 times the amount of draws the PBC family of algorithms output respectively. But even with a lot of draws Gibbs and MH do not seem to produce high quality samples and hence better results. So, if we are dealing with a very high dimensional clustering problem where saving a lot of draws is not an option PBC family can prove to be a very effective alternative.

We have shown in the various examples considered that PBC family of schemes not only have the ability to produce samples with less *AIAT* but also they have the ability to escape local modes as a sampler (see section 7.2) hence they are very efficient stochastic optimizers (see sections 7.3 and 7.4). In the same vein, we should point out that the PBC family of algorithms could be applied to any general problem which could be cast into a clustering problem with a density over the space of cluster indicators. The variable selection problem considered in section 7.4 is a good example of such a scenario.

The SCSC and the SCRC family of moves introduced in the PBC set up are a completely new set of moves and the intuition behind their design is very appealing. We

choose two parent chromosomes either randomly or with probability proportional to their “goodness” or “fitness”. Now we take their “vote” in the sense that we consider the intersections of the sub-clusters formed by the two parents. Then we randomly swap (in SCSC) or reshuffle (in SCRC) two of these intersections and thus form the child(ren). This way of producing child(ren) only perturbs the internal structure of the parents with respect to only two sub-clusters and hence this process respects what the parents jointly have to say about the structure of the other (unperturbed) sub-clusters.

Note, if a SCSC / SCRC move is accepted then we achieve the goal of changing more than one coordinate of the parent chromosome(s) at once which is not possible in Gibbs or Metropolis-Hastings “split-merge” (see section 3.1) sampling where only one coordinate of z is proposed to be updated at a time. The flavor of “split-merge” which could be found, for example, in [10, Jain et. al., 2004] is a bit different from what we mentioned before. In their method, which we call the Jain-Neal “split-merge” method, they first choose two coordinates of z . If these happen to lie in the same mother cluster then the mother cluster is split (in a special way) into two children clusters (containing those two coordinates) and thus a new proposal is generated. In case where the two coordinates lie in different clusters to begin with then those two clusters are merged to produce a new proposal. This procedure of proposing is rather drastic and in some sense the other extreme of a Gibbs move. All the SCSC / SCRC moves in this context take somewhat of a middle ground between Gibbs on one end and the Jain-Neal “split-merge” sampler on the other.

On the intra-comparison of the PBC family of moves the following is worth the mention. The intuition behind introducing the SCRC moves is that it is not as drastic as the SCSC family of moves. Instead of completely swapping two intersection of sub-clusters as required in the SCSC:ONE-NEW move, in SCRC, we reshuffle their members randomly which understandably does not perturb the structure of the parent which gets changed to produce one new child very much. In fact, the SCSC:ONE-NEW move can be considered as a special case of the SCRC move where the reshuffling is nothing but a deterministic swapping. In the same vein, we might say SCSC:TWO-NEW is the most drastic in that it perturbs both the parents. But as noted in the examples (see, in particular, the comments at the end of sections 7.1 and 7.2) whether to perturb both the parents (as in SCSC:TWO-NEW) or just one (as in SCSC:ONE-NEW and SCRC) i.e. which one is a better strategy in that which proposal is going to be accepted more often is an important question and thus far we have no intuition but to say that it depends on the problem and the temperature ladder.

Note the SCSC moves do not alter the number of sub-clusters present in the parents in the process of producing child(ren) and thus are local moves. One might argue that this is an issue with these kinds of moves, namely, the SCSC moves are local in the sense that discovery of new modes may not be possible in difficult scenarios where very different subsets (e.g. subsets in which all cluster indicators do not have

the same number of sub-clusters) of the domain needs to be searched quite rapidly. This is a valid criticism of our new moves. This drawback of the SCSC moves could also prove to be a plus point in certain scenarios. In problems where the number of sub-clusters is always fixed beforehand, see sections 7.3 and 7.4, SCSC moves will be far more effective in producing proposals (and hence getting them accepted) than, say, Metropolis-Hastings “split-merge” move where most of the proposals won’t even be valid because any split move would violate the “number of sub-clusters” fixed condition.

We end this section by looking at some important questions which are to be addressed in our future research. With a very few data points at hand, say, around 100, we have seen in a simulation study in the setting of section 7.1, that its not possible to get back the original or “true” cluster indicator vector which was used to generate a data set as the MAP (Maximum A-Posteriori) estimator using any method of sampling. This is so, because due to sampling variability the generated data along with the prior might give rise to a MAP estimator which has definitely a much higher posterior value than the “true” cluster indicator. Such a phenomenon ceases to persist if number of data points grow. So, how could we provide a meaningful “confidence interval” for the unknown “true” cluster indicator using the samples from our algorithms or for that matter any clustering algorithm? We do not have a meaningful answer to this question at the present moment. Another issue here is that in the literature, e.g. see [10, Jain et. al., 2004] and [2, Dahl, 2003], it seems customary to use Integrated Autocorrelation Time or IAT as a measure of how well the samples move around the space. We have followed the same custom in this article but some questions still remain. Unless we know from an different source that indeed a sampler is covering almost all the major modes (which is almost always impossible) in a certain problem, the IAT’s might be misleading. This is so because a sampler may be moving freely within a local mode producing a low IAT whereas a “better” sampler which actually jumps around modes but very infrequently does so might produce a abnormally very high IAT. So, IAT’s may not be a fair ground of comparison in this context.

References

- [1] David Blackwell and James B. MacQueen. Ferguson distributions via Plya urn schemes. *The Annals of Statistics*, 1:353–355, 1973.
- [2] David B. Dahl. An improved merge-split sampler for conjugate dirichlet process mixture models. Technical report, Department of Statistics, University of Wisconsin, 2003.
- [3] Thomas S. Ferguson. Prior distributions on spaces of probability measures. *The Annals of Statistics*, 2:615–629, 1974.

- [4] Alan E. Gelfand and Adrian F. M. Smith. Sampling-based approaches to calculating marginal densities. *Journal of the American Statistical Association*, 85:398–409, 1990.
- [5] Charles J. Geyer. Markov chain Monte Carlo maximum likelihood. In *Computing Science and Statistics. Proceedings of the 23rd Symposium on the Interface*, pages 156–163. Interface Foundation of North America (Fairfax Station, VA), 1991.
- [6] Charles J. Geyer. Practical Markov chain Monte Carlo (Disc: p483-503). *Statistical Science*, 7:473–483, 1992.
- [7] G. R. Goswami and Jun S. Liu. On real-parameter evolutionary monte carlo algorithm. Technical report, Department of Statistics, Harvard University, 2004.
- [8] Trevor Hastie, Robert Tibshirani, and J. H. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction: with 200 Full-color Illustrations*. Springer-Verlag Inc, 2001.
- [9] W. K. Hastings. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57:97–109, 1970.
- [10] Sonia Jain and Radford M. Neal. A split-merge markov chain monte carlo procedure for the dirichlet process mixture model. *Journal of Computational and Graphical Statistics*, 2004.
- [11] Shane T. Jensen and Jun S. Liu. Bayesian clustering of transcription factor binding motifs.
- [12] Faming Liang and Wing Hung Wong. Evolutionary Monte Carlo: Applications to C_P model sampling and change point problem. *Statistica Sinica*, 10(2):317–342, 2000.
- [13] Faming Liang and Wing Hung Wong. Real-parameter evolutionary monte carlo with applications to bayesian mixture models. *Journal of the American Statistical Association*, 96:653–666, 2001.
- [14] Jun S. Liu, Faming Liang, and Wing Hung Wong. The multiple-try method and local optimization in Metropolis sampling. *Journal of the American Statistical Association*, 95(449):121–134, 2000.
- [15] G. O. Roberts and W. R. Gilks. Convergence of adaptive direction sampling. *Journal of Multivariate Analysis*, 49:287–298, 1994.

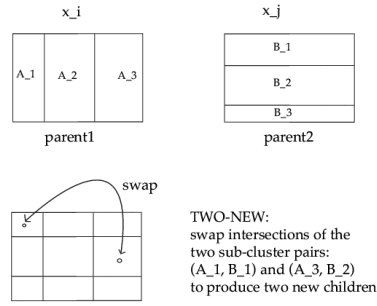


Figure 1: Pictorial representation of SCSC:TWO-NEW clustering.

mixture no.	proportion	$\theta_{ih}, h = 1, 2, \dots, 15$						
		$h = 1$	$h = 2$	$h = 3$	$h = 4$	$h = 5$	\dots	$h = 15$
1	0.2	0.95	0.95	0.95	0.95	0.95	\dots	0.95
2	0.2	0.05	0.05	0.05	0.05	0.95	\dots	0.95
3	0.2	0.95	0.05	0.05	0.95	0.95	\dots	0.95
4	0.2	0.05	0.05	0.05	0.05	0.05	\dots	0.05
5	0.2	0.95	0.95	0.95	0.95	0.05	\dots	0.05

Table 1: TRUE MIXTURE DISTRIBUTION FOR THE BERNOULLI-BETA CLUSTERING (NOTE: THE COLUMNS ENTRIES CORRESPONDING TO $h = 5$ THROUGH $h = 15$ ARE THE SAME IN EACH ROW IF THIS TABLE)

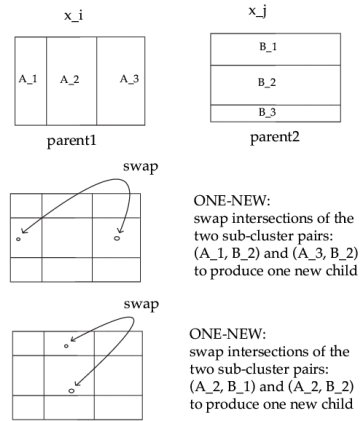


Figure 2: Pictorial representation of SCSC:ONE-NEW clustering.

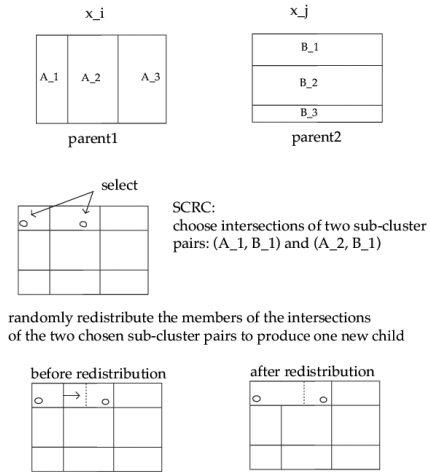


Figure 3: Pictorial representation of SCRC clustering.

t_1 to t_5	20.000000	13.603093	10.934207	8.962745	7.958762
t_6 to t_{10}	6.857107	5.049388	4.508420	4.000087	3.647210
t_{11} to t_{15}	3.218148	2.887211	2.529059	1.675123	1.556680
t_{16} to t_{20}	1.440912	1.334983	1.244001	1.114253	1.000000

Table 2: TEMPERATURE LADDER OF LENGTH 20 FOR THE BERNOULLI BETA EXAMPLE

Method	Acceptance rate range
TWO-NEW-b-b	10% - 20%
TWO-NEW-r-r	15% - 20%
ONE-NEW-b-b	0.1% - 2%
ONE-NEW-r-r	5% - 10%
RANDOM-SIZE	0.1% - 1%
SAME-SIZE	0.1% - 1%

Table 3: ACCEPTANCE RATES FOR VARIOUS METHODS FOR THE BERNOULLI BETA EXAMPLE

Method	$AIAT_{20}$ for statistic				
	$n(\underline{z})$	$e(\underline{z})$	$l(\underline{z})$	$p_{max}(\underline{z})$	$AMLD_{20}$
Gibbs	102.489	98.73	35.174	81.975	-606.481
MH	96.327	161.879	163.48	82.721	-589.482
TWO-NEW-b-b	3.388	6.33	7.889	3.971	-589.782
TWO-NEW-r-r	2.653	4.506	6.622	4.703	-589.73
ONE-NEW-b-b	13.159	29.827	29.91	7.045	-589.44
ONE-NEW-r-r	21.324	21.683	17.336	12.166	-589.44
RANDOM-SIZE	20.072	50.111	49.102	7.359	-589.098
SAME-SIZE	23.284	55.469	54.159	6.869	-589.226

Table 4: COMPARATIVE PERFORMANCE OF VARIOUS ALGORITHMS FOR THE BERNOULLI BETA EXAMPLE

Method	Acceptance rate range
TWO-NEW-b-b	0.1% - 0.5%
TWO-NEW-r-r	1% - 5%
ONE-NEW-b-b	5% - 10%
ONE-NEW-r-r	0.5% - 1%
SAME-SIZE	15% - 25%
RANDOM-SIZE	15% - 25%

Table 5: ACCEPTANCE RATES FOR VARIOUS METHODS FOR THE MOTIF CLUSTERING EXAMPLE

Method	$AIAT_{10}$ for statistic			$AMLD_{10}$
	$n(\underline{z})$	$e(\underline{z})$	$l(\underline{z})$	
Gibbs	222.136	310.25	247.105	-6467.902
MH	21.259	21.359	34.037	-6440.006
TWO-NEW-b-b	11.425	12.136	3.212	-6431.652
TWO-NEW-r-r	10.902	13.459	3.95	-6431.527
ONE-NEW-b-b	10.295	9.619	1.233	-6431.527
ONE-NEW-r-r	24.644	26.716	18.298	-6431.527
RANDOM-SIZE	5.749	5.392	1.065	-6431.527
SAME-SIZE	8.878	8.123	1.018	-6431.527

Table 6: COMPARATIVE PERFORMANCE OF VARIOUS ALGORITHMS FOR THE MOTIF CLUSTERING EXAMPLE

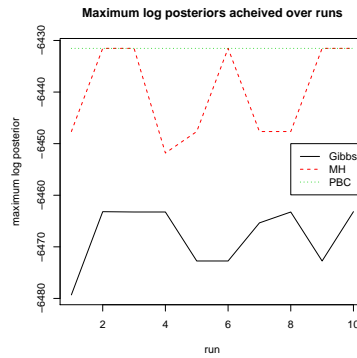


Figure 4: MAXIMUM LOG POSTERIOR COMPARISON

Method	Summary statistics					
	Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
K-means	0	0	0	56.4	40.2	1065.3
TWO-NEW-b-b	0	0	0.2	0.1	0.3	0.5
TWO-NEW-r-r	0	0	0	0.2	0.3	5.8
ONE-NEW-b-b	0	0	0	0.1	0.2	0.6
ONE-NEW-r-r	0	0	0.2	0.2	0.3	0.8
RANDOM-SIZE	0	0	0	0.1	0.2	0.4
SAME-SIZE	0	0	0	0.1	0.2	0.4

Table 7: COMPARISON OF KMEANS AND PBC FAMILY OF METHODS AS MINIMIZERS FOR THE NORMAL CLUSTERING EXAMPLE (THE MINIMUM ACHIEVED I.E. 339.7 HAS BEEN SUBTRACTED FROM THE NUMBERS IN THE TABLE AND THEN THEY HAVE BEEN ROUNDED FOR EASY READING)

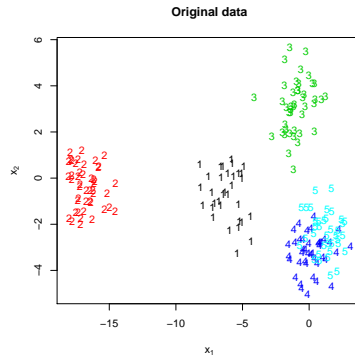


Figure 5: ORIGINAL DATA FOR NORMAL CLUSTERING

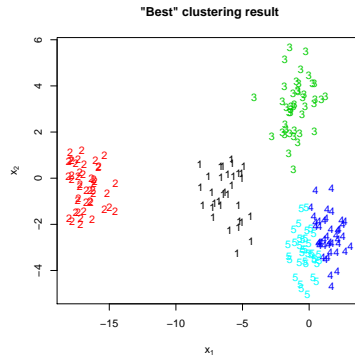


Figure 6: “BEST” CLUSTERING RESULT OF DATA FOR NORMAL CLUSTERING

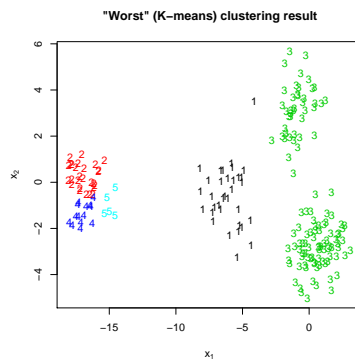


Figure 7: “WORST” (K-MEANS) CLUSTERING RESULT OF DATA FOR NORMAL CLUSTERING

Method	$AIAT_{10}$ for statistic		
	$e(\underline{z})$	$l(\underline{z})$	$AMLD_{10}$
MH	2.782	4.294	-588.289
TWO-NEW-b-b	2.797	2.499	-579.631
TWO-NEW-r-r	4.331	2.688	-579.738
ONE-NEW-b-b	2.395	3.29	-579.449
ONE-NEW-r-r	7.87	3.835	-580.215
RANDOM-SIZE	2.364	3.418	-579.388
SAME-SIZE	2.856	3.664	-579.567

Table 8: COMPARATIVE PERFORMANCE OF VARIOUS ALGORITHMS FOR THE FOR THE NORMAL CLUSTERING EXAMPLE

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
0.6804	0.7739	0.7924	0.7946	0.8122	1

Table 9: SUMMARY OF CORRELATION AMONG THE EXPLANATORY VARIABLES IN THE VARIABLE SELECTION EXAMPLE

Method	$AIAT_{10}$ for statistic			
	$e(\underline{z})$	$ A_1 $	$l(\underline{z})$	$AMLD_{10}$
EMC	2.614	2.606	0.921	-251.376
TWO-NEW-b-b	1.608	1.179	0.942	-251.376
ONE-NEW-b-b	3.251	3.489	0.845	-251.376
RANDOM-SIZE	1.422	1.454	0.828	-251.376

Table 10: COMPARATIVE PERFORMANCE OF VARIOUS ALGORITHMS FOR THE FOR THE VARIABLE SELECTION EXAMPLE