

Complexity bounds for Dirichlet process slice samplers

Beatrice Franzolini,
DEMS Department, University of Milano-Bicocca
Francesco Gaffi
Department of Economics, University of Bergamo

Abstract

Slice sampling is a standard Monte Carlo technique for Dirichlet process (DP)–based models, widely used in posterior simulation. However, formal assessments of the scalability of posterior slice samplers have remained largely unexplored, primarily because the computational cost of a slice-sampling iteration is random and potentially unbounded. In this work, we obtain high-probability bounds on the computational complexity of DP slice samplers. Our main results show that, uniformly across posterior cluster-growth regimes, the overhead induced by slice variables, relatively to the number of clusters supported by the posterior, is $O_{\mathbb{P}}(\log n)$. As a consequence, even in worst-case configurations, superlinear blow-ups in per-iteration computational cost occur with vanishing probability. Our analysis applies broadly to DP–based models without any likelihood-specific assumptions, still providing complexity guarantees for posterior sampling on arbitrary datasets. These results establish a theoretical foundation for assessing the practical scalability of slice sampling in DP-based models.

Keywords: Bayesian nonparametrics, Dirichlet process, Markov Chain Monte Carlo, Mixture models, Slice sampling

1 Introduction

Dirichlet process (DP)-based models are widely used to define flexible latent-component structures in problems where the number of clusters, groups, or latent effects is unknown. Canonical examples include mixture models for density estimation and clustering (Neal, 2000), stochastic block models for network data (Kemp et al., 2006; Xu et al., 2006), regression and random-effects models (Wade and Inácio, 2025), and species sampling problems (Balocchi et al., 2024). Beyond classical applications, DP-based constructions continue to be an active modeling tool in novel machine learning methodologies and applications, including domain adaptation (Ling et al., 2024), open-set classification (Snell et al., 2024), covariate-informed clustering (Chakrabarti et al., 2025), and online anomaly detection (Mei and Yan, 2026). In all these settings, inference involves exploring distributions over partitions whose complexity grows with the sample size, offering substantial modeling flexibility while posing significant computational challenges.

Markov chain Monte Carlo (MCMC) algorithms for DP-based models broadly fall into two main classes: marginal and conditional methods. Marginal algorithms integrate out the random probability measure, yielding exact inference for the induced partition structure, but typically rely on sequential, one-at-a-time updates and nontrivial bookkeeping (Neal, 2000). While often effective for simple mixture models and small sample sizes, these features become increasingly restrictive in large datasets and in models with complex likelihoods, such as regression settings with covariate-dependent structure (see, for instance, Dunson and Park, 2008) or relational data with non-exchangeable observations (see, for instance, Cai et al., 2016; Zhou et al., 2024). Conditional algorithms instead retain the random probability measure explicitly, commonly via the stick-breaking representation of the DP (Sethuraman, 1994). This enables joint updates of latent variables and often leads to simpler implementations. A widely used approach in this class is truncation of the stick-breaking representation at a fixed level, resulting in blocked Gibbs samplers that are computationally efficient and, for typical likelihood specifications, perform joint updates (Ishwaran and James, 2001). However, truncation fundamentally alters the model: unless the truncation level grows with the sample size, truncated algorithms introduce a hard-threshold non-vanishing bias in the posterior distribution of the partition by assigning zero probability to configurations with more clusters than the truncation allows. Although average-case error bounds under the generative model are available (Ishwaran and James, 2002; Ishwaran and Zarepour, 2002; Campbell et al., 2019; Li and Campbell, 2021), they do not provide guarantees for a fixed observed dataset.

Slice sampling offers a principled alternative that avoids both marginalization and truncation while preserving convergence to the exact posterior (Neal, 2003; Walker, 2007; Ge et al., 2015). By introducing auxiliary slice variables, slice samplers restrict attention to a finite subset of components at each MCMC iteration without imposing a fixed truncation level. Originally developed for DP mixture models in Walker (2007), slice sampling extends directly to a broad class of DP-based models, generalizations of DPs, and discrete random measures whenever a stick-breaking or weight-based representation is available (Kalli et al., 2011; Zhu et al., 2020). In practice, slice samplers combine the advantages of marginal and blocked methods: they target the true posterior distribution, typically

allow joint updates, and require minimal bookkeeping.

Despite these advantages, slice samplers for the DP (as defined for instance in [Walker, 2007](#); [Kalli et al., 2011](#); [Ge et al., 2015](#)) suffer from a fundamental theoretical limitation: their per-iteration computational cost is unbounded. The number of components that must be instantiated at a given iteration depends on the minimum slice variable, which can be arbitrarily close to zero with positive probability, forcing the number of steps in the algorithm to be arbitrarily large. Formally, let $(X_s)_{s \geq 1}$ denote the Markov chain induced by the slice sampler, $C(X_s)$ the computational cost of iteration s (in some unit of measure), and π the stationary distribution. For any finite threshold $C < \infty$, define the high-cost set $A_C := \{x : C(x) > C\}$. At stationarity, the marginal probability $\mathbb{P}_\pi(C(X_s) > C) =: \pi(A_C)$ is constant across iterations. Defining the hitting time $\tau_C = \inf\{s \geq 1 : X_s \in A_C\}$, since $\pi(A_C) > 0$, $\tau_C < \infty$ almost surely and

$$\mathbb{P}_\pi(\exists s \leq T : C(X_s) > C) = \mathbb{P}_\pi(\tau_C \leq T) \xrightarrow[T \rightarrow \infty]{} 1.$$

Consequently, no deterministic upper bound on the computational cost of DP slice samplers exists. This lack of formal complexity guarantees has made it difficult to assess the scalability of slice-based inference.

In this work, we provide a formal analysis of the computational complexity of slice sampling for a broad class of DP-based models. Rather than seeking deterministic worst-case bounds, which are unattainable for slice samplers, we adopt a high-probability perspective and derive probabilistic bounds on the number of components instantiated at each iteration. Our main results show that, with arbitrarily high probability, the computational overhead introduced by the slice variables over the number of clusters supported by the posterior grows at most logarithmically with the sample size. Thus, even in unfavorable configurations, superlinear blow-ups in per-iteration cost occur with vanishing probability. Our analysis applies broadly to general DP-based models without relying on model-specific likelihood assumptions, while providing guarantees for any input data. By establishing explicit high-probability complexity guarantees, this work provides a theoretical foundation for the practical scalability of slice-based inference in DP-based models.

2 Preliminaries on Dirichlet process and slice samplers

We define a general class of models based on the DP as follows.

Definition 2.1 (DP-based generative model). Let $\mathbf{z}_n = (z_1, \dots, z_n)$. We say that a random object Y is generated from a DP-based model with sample size n if

$$\begin{aligned} Y \mid \eta, \mathbf{z}_n &\sim \mathcal{L}(\mathbf{z}_n, \eta), \\ z_i \mid G &\stackrel{\text{iid}}{\sim} G, \quad i \in [n], \\ G &\sim \text{DP}(\alpha, P_0), \end{aligned}$$

where \mathcal{L} denotes a likelihood depending on the latent variables \mathbf{z}_n and parameters η , $[n] = \{1, \dots, n\}$, and $\text{DP}(\alpha, P_0)$ is the law of a DP (Ferguson, 1973) with concentration parameter $\alpha > 0$ and non-atomic base measure P_0 .

For different choices of the likelihood \mathcal{L} , Definition 2.1 encompasses a wide range of models, including species sampling models, DP mixture models, DP stochastic block models, and related constructions.

A fundamental challenge in sampling (both *a priori* and *a posteriori*) from models in Definition 2.1 arises from the infinite-dimensional nature of the random probability measure G . Almost surely, G admits the stick-breaking representation (Sethuraman, 1994)

$$G(dx) = \sum_{k=1}^{\infty} \pi_k \delta_{\phi_k}(dx), \quad (1)$$

where the atoms $(\phi_k)_{k \geq 1}$ and the weights $(\pi_k)_{k \geq 1}$ are independent, with

$$\phi_k \stackrel{\text{iid}}{\sim} P_0, \quad \pi_k = V_k \prod_{\ell=1}^{k-1} (1 - V_\ell), \quad V_k \stackrel{\text{iid}}{\sim} \text{Beta}(1, \alpha).$$

This representation highlights the intrinsic infinite-dimensional structure of G , which is the main source of both the flexibility of DP-based models and the computational challenges associated with performing inference under them. The latent variables \mathbf{z}_n in Definition 2.1 naturally induce a random partition of the index set $[n]$. Specifically, define an equivalence relation on $[n]$ by declaring $i \sim j$ if and only if $z_i = z_j$. The resulting equivalence classes correspond to clusters of latent variables sharing the same value and define a partition ρ_n of $[n]$. Under the DP prior law on G , this random partition is exchangeable, and its *a priori* distribution depends solely on α (Pitman, 1996). In the following, we encode such a partition also with a *cluster-label vector* $\mathbf{c}_n = (c_1, \dots, c_n)$, such that $c_i \in [n]$ and $c_i = c$ means that i belongs to cluster c (in some order, *e.g.*, order of appearance) according to ρ_n .

Slice sampling for DP-based models introduces auxiliary variables to restrict inference to a finite, data-dependent subset of latent components while preserving exactness of the target distribution (*i.e.*, the posterior of ρ_n). The original formulation of slice sampling procedures to sample x from a distribution with density f on $\mathbb{X} \subset \mathbb{R}^m$ (Neal, 2003) introduces a latent slice variable u with joint density $p(x, u) = \mathbb{1}(0 < u < f(x))$ that leaves the marginal distribution of x unchanged. Conditional on u , sampling x reduces to sampling uniformly over the slice set $\{x : f(x) > u\}$. This construction specializes naturally to discrete distributions (Walker, 2007; Kalli et al., 2011; Ge et al., 2015). In particular, for a DP realization as in (1), conditionally on the latent slice variable u , a finite active set is defined as $A(u) = \{k : \pi_k > u\}$, and x is sampled uniformly from $A(u)$. Figure 1 exemplifies the distributions and the sampling mechanism that define a slice sampler for a discrete distribution G . Algorithms 1 and 2 contain the slice sampling algorithms to sample respectively from the generative model and from the posterior distribution for any model following Definition 2.1. In contrast with the procedure proposed in Walker (2007) and Kalli et al. (2011), Algorithm 2 relies on the posterior representation of the DP described in Pitman (1996) and employed in the improved slice sampler

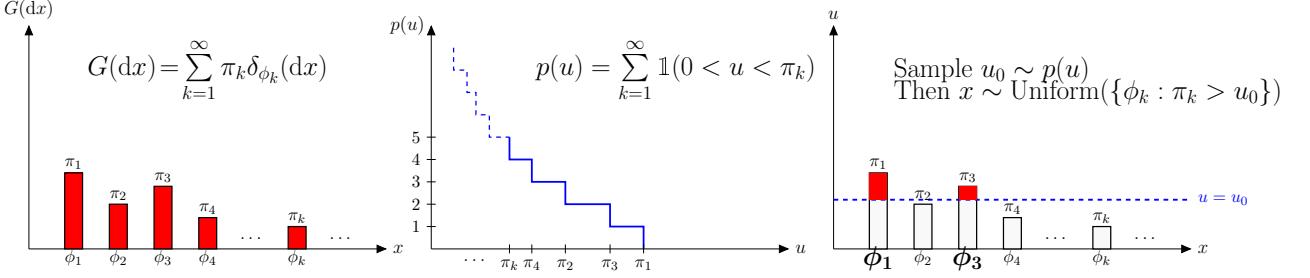


Figure 1: Distributions involved in the slice sampler. Sampling $x \sim G(dx)$ is equivalent to sampling $u \sim p(u)$ and $x \mid u \sim p(x \mid u)$. *Left panel:* A realization of a discrete probability measure $G(dx) = \sum_{k=1}^{\infty} \pi_k \delta_{\phi_k}(dx)$, represented as point masses located at atoms ϕ_k with weights π_k . Sampling $x \sim G$ corresponds to selecting one atom ϕ_k with probability π_k . *Center panel:* The marginal distribution of the auxiliary slice variable u , given by $p(u) = \sum_{k \geq 1} \mathbb{1}(0 < u < \pi_k)$, which is a decreasing step function supported on $(0, \max_k \pi_k)$. *Right panel:* Slice-sampling mechanism. First, a slice value u_0 is drawn from $p(u)$ (horizontal dashed line). Conditionally on u_0 , the variable x is sampled uniformly from the finite set $\{\phi_k : \pi_k > u_0\}$, corresponding to atoms whose weights exceed the slice level.

of [Ge et al. \(2015\)](#), which allows to recover exchangeability of components, improving the mixing of the slice sampler and leading to the update of the weights of the allocated components according to $(\pi_1, \dots, \pi_H, \pi^*) \sim \text{Dirichlet}(n_1, \dots, n_H, \alpha)$. See Section 3 for a more detailed account of the implications of this choice.

While all steps involve sampling from relatively simple distributions, the core of the algorithm arguably lies in determining how many weights and atoms must be instantiated. To guarantee that the set $A(u_i)$ can be identified exactly for each i , a sufficient number of components must be instantiated so that all weights satisfying $\pi_k > u_i$ are available. In Neal's original slice sampler ([Neal, 2003](#)), the slice set is located via generic expansion procedures that are well-suited to continuous targets but possibly inefficient when G is discrete. The practical difficulty of controlling the number of instantiated components under this approach for discrete distributions is evident, for instance, in the exact slice sampler for hierarchical DP proposed by [Amini et al. \(2019\)](#), which relies on additional mechanisms to manage the growth of the active set.

In this regard, [Walker \(2007\)](#) observes that a finite number K of components sufficient to allocate the latent variables in \mathbf{z} at each iteration is

$$K = \min \left\{ k \in \mathbb{N} : \sum_{h=k+1}^{\infty} \pi_h < u_{\min} \right\} \quad (2)$$

with $u_{\min} := \min_i u_i$. By this definition, K is finite and any weight beyond the K -th component necessarily satisfies $\pi_j \leq u_i$ for every i and $j > K$. These components can be safely ignored at the current iteration for sampling \mathbf{z} . To implement this approach, one may dynamically determine the truncation level K during the sampling of the stick-breaking weights by checking the condition $\sum_{k=1}^K \pi_k > 1 - u_{\min}$ and stopping as soon as it is met, as detailed in steps 16–22 of Algorithm 1 and steps 11–17 of Algorithm 2. In this way, the algorithm avoids accept-reject steps, retrospective sampling ([Papaspiliopoulos and Roberts, 2008](#)), or any approximated truncated representation.

Algorithm 1 Slice-Sampler a priori (generative mechanism)

Require: Hyperparameters α, P_0, η , num. of iterations T

```

1: Initialize allocations  $c_i$  for  $i \in [n]$ 
2: for  $t = 1, \dots, T$  do
3:    $H \leftarrow$  number of clusters
4:   Remap  $(c_i)_{i=1}^n$  into  $[H]^n$ 
5:    $\pi^* \leftarrow 1$ 
6:   for  $h = 1, \dots, H$  do
7:     Sample  $\phi_h \sim P_0$ 
8:     Sample  $v_h \sim \text{Beta}(1, \alpha)$ 
9:      $\pi_h \leftarrow v_h \times \pi^*$ 
10:     $\pi^* \leftarrow \pi^* - \pi_h$ 
11:   end for
12:   for  $i = 1$  to  $n$  do
13:     Sample  $u_i \sim \text{Uniform}(0, \pi_{c_i})$ 
14:   end for
15:    $u^* \leftarrow \min_i u_i, K \leftarrow H$ 
16:   while  $\pi^* > u^*$  do
17:      $K \leftarrow K + 1$ 
18:     Sample  $\phi_K \sim P_0$ 
19:     Sample  $z_K \sim \text{Beta}(1, \alpha)$ 
20:      $\pi_K \leftarrow z_K \times \pi^*$ 
21:      $\pi^* \leftarrow \pi^* - \pi_K$ 
22:   end while
23:   for  $i = 1$  to  $n$  do
24:      $A_i \leftarrow \{k \in [K] : \pi_k > u_i\}$ 
25:     Sample  $c_i \sim \text{Uniform}(A_i)$ 
26:      $z_i \leftarrow \phi_{c_i}$ 
27:   end for
28:   Sample  $Y \sim \mathcal{L}(\mathbf{z}_n, \eta)$ 
29: end for

```

However, u_{\min} can be arbitrarily close to zero with positive probability at each iteration, and therefore, the condition

$$\sum_{j=1}^K \pi_j > 1 - u_{\min}$$

may force K to be arbitrarily large, so that the sampler must update an unbounded number of

Algorithm 2 Slice-Sampler a posteriori

Require: Hyperparameters α , P_0 , η , num. of iterations T

```
1: Initialize allocations  $c_i$  for  $i \in [n]$ 
2: for  $t = 1, \dots, T$  do
3:    $H \leftarrow$  number of clusters
4:   Remap  $(c_i)_{i=1}^n$  into an element in  $[H]^n$ 
5:    $n_h \leftarrow \sum_i \mathbb{1}(c_i = h)$ 
6:    $(\pi_1, \dots, \pi_H, \pi^*) \sim \text{Dirichlet}(n_1, \dots, n_H, \alpha)$ 
7:   for  $h = 1$  to  $H$  do
8:     Sample  $\phi_h$ 
9:   end for
10:  for  $i = 1$  to  $n$  do
11:    Sample  $u_i \sim \text{Uniform}(0, \pi_{c_i})$ 
12:  end for
13:   $u^* \leftarrow \min_i u_i$ ,  $K \leftarrow H$ 
14:  while  $\pi^* > u^*$  do
15:     $K \leftarrow K + 1$ 
16:    Sample  $\phi_K \sim P_0$ 
17:    Sample  $v_K \sim \text{Beta}(1, \alpha)$ 
18:     $\pi_K \leftarrow v_K \times \pi^*$ 
19:     $\pi^* \leftarrow \pi^* - \pi_K$ 
20:  end while
21:  for  $i = 1$  to  $n$  do
22:     $A_i \leftarrow \{k \in [K] : \pi_k > u_i\}$ 
23:    Sample  $c_i \sim \text{Cat}(w_k \propto \text{Lik}(Y; z_i = \phi_k), k \in A_i)^*$ 
24:     $z_i \leftarrow \phi_{c_i}$ 
25:  end for
26: end for
```

* $\text{Lik}(Y; z_i = \phi_k)$ denotes the likelihood evaluated at $z_i = \phi_k$ and current state of the other parameters. Typically, this step does not require full evaluation of the likelihood (see Section 4).

components (an effect that becomes more pronounced as the tails of the Dirichlet process become heavier). Therefore, it is important to assess the probability of actually exceeding a specific computational threshold at each iteration and how such probability varies with the sample size n growing. In Section 3, we derive *a posteriori* high-probability bounds on computational cost, valid in any cluster-growth regimes, providing guarantees that hold *a posteriori* for any dataset. All the proofs are given in Appendix A.

3 High-probability bounds for slice samplers complexity

The computational cost of Algorithm 2 depends on the truncation level K instantiated at each iteration. In particular, computational complexity is driven by the final **for**-loop updating the allocations. It requires, for each observation $i \in [n]$, evaluating the likelihood for all components $k \in A_i \subseteq [K]$. In the worst case, this step entails up to $n \times K$ likelihood evaluations per iteration. Consequently, while the slice sampler avoids fixed truncation, its practical scalability hinges on probabilistic control of the magnitude of K , which directly governs the dominant per-iteration computational cost. In the following, we show that the performance of the slice sampler for DP-based models is safeguarded by high-probability bounds. To derive the result the main quantity under study is the dynamically-determined truncation level K and its behaviour as $n \rightarrow \infty$, therefore from now on we replace the notation K with K_n to highlight dependence on the sample size.

The truncation level K_n , as defined by Walker (2007) and reported in equation (2), is the smallest index such that the remaining stick-breaking mass falls below the minimum slice variable u_{\min} . When the stick-breaking weights entering this definition are sampled from the Dirichlet process prior and one conditions on a fixed value of u_{\min} , the distribution of K_n admits an explicit characterization: by a classical result of Muliere and Tardella (1998), one has $K_n - 1 \mid u_{\min} \sim \text{Poisson}(\alpha \log(1/u_{\min}))$. This result serves as a useful reference point for understanding how the slice level controls the number of instantiated components under prior sampling. However, its scope is inherently limited: it is based on prior sampling for the weights, does not account for the configuration of the allocated components, and, crucially, does not describe how u_{\min} itself behaves as the sample size grows.

In practical implementations of slice-based Gibbs samplers, both *a priori* and *a posteriori*, the truncation level K_n is subject to an additional structural constraint: it must always be at least as large as the maximum label appearing in the current cluster allocation. Otherwise, the slice variables u_i cannot be sampled (see line 13 of Algorithm 1 and line 8 of Algorithm 2). The improved slice sampler of Ge et al. (2015) exploits the exchangeability of the posterior representation of the DP described in Pitman (1996) to enforce this constraint optimally, by remapping cluster labels so that the maximum label coincides with the number of occupied clusters. This relabeling step leads to a more efficient Gibbs sampler without altering the target posterior distribution. Differently from an approach relying solely on a fixed stick-breaking ordering, the improved slice sampler of Ge et al. (2015) avoids the forced instantiation and sampling of weights corresponding to empty components before entering the **while**-loop. Denoting by H_n the number of clusters at the current iteration, the effective truncation level used by the algorithm can therefore be written as

$$K_n = \min \left\{ k \geq H_n : \sum_{h=k+1}^{\infty} \pi_h < u_{\min} \right\}. \quad (3)$$

In posterior inference, while the formal definition of K_n in (3) remains unchanged, the probabilistic structure governing it differs substantially from the prior-based setting. Both the weights and the slice variables are sampled from their posterior distributions, conditionally on the currently visited partition configuration, which itself depends on the observed data. As a consequence, the tail mass appearing

in the definition of K_n is governed by posterior rather than prior-distributed stick-breaking factors, and the minimum slice variable u_{\min} has a data-dependent distribution.

Understanding the computational complexity of posterior slice sampling therefore requires controlling the joint behavior of the sampled weights and the minimum slice variable u_{\min} as the sample size increases, without relying on prior laws or conditioning on a fixed value of u_{\min} . We start investigating the law of the minimum slicing variable conditioning on the partition configuration visited by the chain and marginalizing the weights. The next Proposition highlights how merging two clusters always reduces the conditional probability of observing minimum slicing variables below any given threshold.

Proposition 3.1 (Merging two clusters increases the survival probability of u_{\min}). *For any ρ_n partition of $[n]$ with cluster sizes (n_1, \dots, n_H) , $\sum_{h=1}^H n_h = n$, let $(\pi_1, \dots, \pi_H, \pi^*) \mid \rho_n \sim \text{Dirichlet}(n_1, \dots, n_H, \alpha)$, let $u_i \mid \pi_{c_i} \sim \text{Uniform}(0, \pi_{c_i})$ independently, and $u_{\min} = \min_{1 \leq i \leq n} u_i$. If $\rho_n^{(r \oplus s)}$ is the partition obtained from ρ_n by merging two (distinct) clusters r and s into a single cluster of size $n_r + n_s$ and leaving the other clusters unchanged, then, for any $x \in (0, 1)$*

$$\mathbb{P}(u_{\min} \leq x \mid \rho_n^{(r \oplus s)}) \leq \mathbb{P}(u_{\min} \leq x \mid \rho_n).$$

As a direct consequence of Proposition 3.1, we have that the partition with n clusters of size 1 induces the lowest survival probability for u_{\min} .

Corollary 3.2 (Singleton partition yields the lowest survival probability of u_{\min}). *Let ρ_n^{sing} denote the singleton partition of $[n]$, i.e., the partition with n clusters of size 1. Then, for every $x \in (0, 1)$ and every partition ρ_n ,*

$$\mathbb{P}(u_{\min} > x \mid \rho_n^{\text{sing}}) \leq \mathbb{P}(u_{\min} > x \mid \rho_n).$$

Once it is formally established that the singleton partition represents the worst-case scenario in terms of controlling the mass around zero of the minimum slice variable, we can state our main result on the tail of K_n , which holds uniformly over all partitions visited by the posterior algorithm.

Theorem 3.3 (High-probability bound on dynamic truncation level). *Let K_n and H_n be respectively the truncation level in (3) and the number of occupied clusters at a given iteration of Algorithm 2 when run for n input data. Then, for every $\delta \in (0, 1)$ there exist constants $B_\alpha^{(1)}, B_\alpha^{(2)}$ (independent of δ and n) such that for any $n \geq 2$*

$$\mathbb{P}(K_n - H_n \leq C_\delta \log n \mid \rho_n) \geq 1 - \delta, \quad \forall \rho_n$$

with $C_\delta = B_\alpha^{(1)} + B_\alpha^{(2)} \log(1/\delta)$, and $B_\alpha^{(1)}, B_\alpha^{(2)} \asymp \alpha$.

In particular, $K_n - H_n = O_{\mathbb{P}}(\log n)$ and, in the worst-case scenario of $H_n = O(n)$, $K_n = O_{\mathbb{P}}(n)$.

The strength of providing: (i) an explicit constant C_δ with logarithmic growth in δ and (ii) uniformity in n for the high-probability bound is exemplified in the following Corollary, which establishes exponential tails and, consequently, uniformly bounded moments for the slice overhead.

Corollary 3.4. Let (K_n, H_n) be defined as in Theorem 3.3. There exist constants $B_\alpha^{(1)}, B_\alpha^{(2)} > 0$ such that for all $n \geq 2$ and all $t \geq 0$,

$$\mathbb{P} \left(\frac{K_n - H_n}{\log n} > B_\alpha^{(1)} + B_\alpha^{(2)} t \mid \rho_n \right) \leq e^{-t}, \quad \forall \rho_n.$$

In particular, for any $p \geq 1$, there exists a constant $C_{p,\alpha} > 0$ such that

$$\sup_{n \geq 2} \mathbb{E} \left[\left(\frac{K_n - H_n}{\log n} \right)^p \mid \rho_n \right] \leq C_{p,\alpha}, \quad \forall \rho_n.$$

Moreover, the uniform-in- ρ_n nature of the high-probability bound allows for further derivation of an almost-sure bound under infinite-data coupling, implying that super-logarithmic slice overheads cannot occur infinitely often as the sample size grows.

Corollary 3.5. Let (K_n, H_n) be defined as in Theorem 3.3. There exists a constant $D_\alpha > 0$ such that

$$\mathbb{P} \left(\limsup_{n \rightarrow \infty} \left\{ K_n - H_n > D_\alpha \log \frac{1}{\delta_n} \log n \right\} \right) = 0$$

for any summable sequence $(\delta_n)_{n \geq 1} \subset (0, 1/2)$.

4 Comparison with alternative algorithms for mixture models

To translate the results of the previous section into explicit high-probability statements on the scalability of slice sampling for DP models, we focus, for concreteness, on DP mixtures of univariate Normals with fixed variance. Specifically, we consider models of the form

$$Y_i \mid z_i \stackrel{\text{ind}}{\sim} \mathcal{N}(z_i, \sigma^2), \quad z_i \mid G \stackrel{\text{iid}}{\sim} G, \quad G \sim \text{DP}(\alpha, P_0),$$

for $i \in [n]$, where $\mathcal{N}(\mu, \sigma^2)$ denotes a Normal distribution with mean μ and variance σ^2 . This setting serves as a representative working example in which the cost of likelihood evaluation at step 23 of Algorithm 2 is constant per component. However, the arguments developed below extend directly to the broader class of DP-based models described in Definition 2.1, with the understanding that model-specific likelihoods may introduce different per-evaluation computational costs as a function of n . Table 1 summarizes the computational scalability and qualitative properties of several MCMC posterior sampling strategies for DP mixture models, highlighting fundamental trade-offs between exactness, scalability, and implementation complexity.

Marginal samplers based on the Chinese restaurant process (CRP) representation target the exact posterior distribution over partitions and avoid truncation bias. Their dominant per-iteration cost scales as $n \times H_n$, corresponding to the reassignment of each observation across all currently occupied clusters. While this scaling captures the leading-order behavior, the associated constant depends on implementation details. In particular, when cluster-specific parameters are analytically integrated out

	CRP	BGS- L	BGS- n	Slice
Scalability by posterior cluster growth				
$H_n = O(n)$	$O(n^2)$	$\Theta(n)$	$\Theta(n^2)$	$O_{\mathbb{P}}(n^2)$
$H_n = O(\log n)$	$O(n \log n)$	$\Theta(n)$	$\Theta(n^2)$	$O_{\mathbb{P}}(n \log n)$
$H_n = O(1)$	$O(n)$	$\Theta(n)$	$\Theta(n^2)$	$O_{\mathbb{P}}(n \log n)$
Exact posterior partition target	✓	✗	✗	✓
No hard-threshold bias	✓	✗	✓	✓
No bookkeeping	✗	✓	✓	✓
Joint Updates	✗	✓	✓	✓

Table 1: Comparison of posterior sampling algorithms for DP mixture models. The table reports per-iteration computational complexity as a function of the sample size n and the posterior number of clusters H_n , together with key qualitative properties of each method. Since the posterior number of clusters H_n can grow at most linearly in n , the first row represents the worst-case cluster growth regime and thus provides worst-case computational guarantees.

from the full-conditionals of the latent cluster-label vector \mathbf{c}_n , each reassignment requires evaluating a predictive likelihood that typically involves an integral with respect to the base measure of the DP. Moreover, bookkeeping operations needed to maintain sufficient statistics and cluster counts, while not altering the $n \times H_n$ scaling, can substantially increase the constant factor and limit practical scalability to large datasets, while assignments of observations to different clusters, *i.e.*, sampling the elements in \mathbf{c}_n , can only be performed one-at-a-time.

Blocked Gibbs samplers with fixed truncation levels (BGS- L) enable simpler implementations and joint updates, *i.e.*, the assignments to different clusters can be performed in parallel, but at the cost of introducing a hard truncation. This is a model misspecification, not just an approximation error, and leads to a non-vanishing bias. In this regard, [Ishwaran and James \(2002\)](#) establishes in their Theorem 1 and Corollary 1 that the marginal density error of a L -truncated approximation of the DP mixture of Normals is exponentially accurate for the posterior of the partition. Formally, denote with π_L the posterior of the truncated model and with π_∞ and m_∞ the posterior and the marginal likelihood for the DP mixture of Normals. [Ishwaran and James \(2002\)](#) proves that

$$\int_{\mathbb{R}^n} \sum_{\rho_n} |\pi_L(\rho_n \mid \mathbf{Y}) - \pi_\infty(\rho_n \mid \mathbf{Y})| m_\infty(\mathbf{Y}) d\mathbf{Y} = O(4n e^{-(L-1)/\alpha}).$$

While this result gives a useful average-case rate under the prior predictive, it clearly does not guarantee small error for any fixed dataset (and, in particular, for the worst-case scenario) since large discrepancies may occur on regions of the data space with small m_∞ -mass. The truncated model simply cannot represent more than L clusters: $\pi_L(\rho_n) = 0$ on all partitions with more than L clusters, yet these partitions may have substantial π_∞ -mass. Since the truncation level is fixed by design, their per-iteration cost is deterministic, and scales as nL . One way to avoid the systematic underestimation of the probability of partitions with $H_n > L$ in truncated blocked algorithms is to let the truncation

level to be exactly n (BGS- n). This strategy, however, leads to a $\Theta(n^2)$ per-iteration computational cost in all scenarios, while the chain keeps not targeting the exact posterior.

On the other hand, the scalability of slice-based samplers is effectively safeguarded by our high-probability bound: the excess number of sampled components beyond those occupied by the data grows at most logarithmically with high probability. Consequently, as summarized in Table 1, slice samplers retain exactness, easy or no bookkeeping, and joint updates while achieving favorable scalability in high-probability, especially in regimes where the number of clusters grows slowly with the sample size.

5 Numerical experiments

First, in this section, we present a numerical study aimed at empirically comparing the computational performance of six posterior sampling algorithms for DP mixtures of univariate Normals. The methods considered are: the slice sampler described in Algorithm 2; a variant of the slice sampler in which the atoms ϕ_k are analytically marginalized out; two blocked Gibbs samplers based on finite dimensional Dirichlet distributions with truncation levels $L = 10$ and $L = n$, respectively; and two marginal samplers based on the CRP predictive scheme, one sampling the cluster allocations \mathbf{c}_n conditionally on the atoms ϕ_k and one integrating the atoms out analytically.

Data are generated from three equally sized clusters, with observations in each cluster drawn independently from a Normal distribution with unit variance and means equal to -3 , 0 , and 3 . Experiments are repeated for sample sizes $n \in \{150, 300, 600, 1500, 3000, 7500, 12000\}$. For all methods, the base measure of the DP is taken to be the standard Normal distribution, and the concentration parameter α is assigned a $\text{Gamma}(3, 3 \log n)$ prior, and, thus $\mathbb{E}[\alpha] = 1/\log(n)$ a priori. For each algorithm and sample size, we run 10,000 MCMC iterations. All chains are initialized using a k -means clustering solution with five clusters, obtained via the R function `kmeans`. All algorithms are implemented in R and are made available at github.com/beatricefranzolini/DPalg. Figures 2 and 3 report, respectively, the wall-clock time required for 1,000 iterations (in seconds) and the effective sample size (ESS) per second as functions of the input size. For some methods, results at larger sample sizes are unavailable due to computational constraints; we declare an algorithm to be infeasible at a given sample size if it requires more than one second to complete the first ten iterations on a laptop with 13th Gen Intel(R) Core(TM) i7-1370P.

Marginal CRP-based samplers exhibit the steepest growth in computational time, becoming infeasible already at moderate sample sizes, while blocked Gibbs samplers display predictable scaling behavior. In contrast, slice-based samplers achieve a more favorable trade-off: although their per-iteration cost increases with n , the growth is substantially slower than that of marginal samplers and comparable to that of blocked Gibbs samplers with $L = 10$ components, which, however, do not target the correct posterior. This behavior is also reflected in high ESS per second. Marginalizing the atoms within the slice sampler appears to increase the computational time per iteration but does not improve mixing sufficiently, particularly for small datasets, leading to lower ESS per second for small sample sizes compared to the original version. Overall, these empirical results align with the high-probability

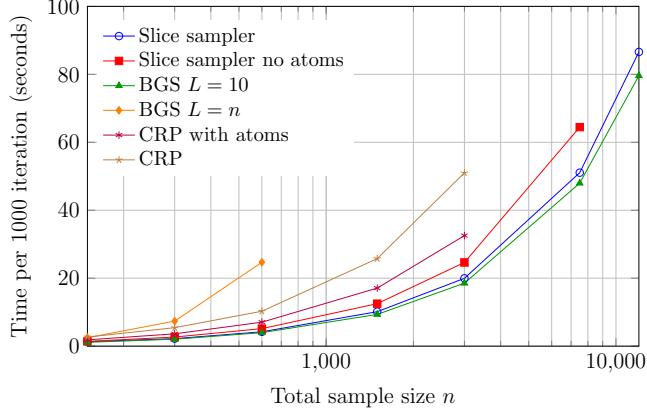


Figure 2: Wall-clock time per 1000 iterations in seconds computed as average over 10,000 iterations (including burn-in) as a function of input size (x-axis on log scale). Codes are in R and run on a laptop with 13th Gen Intel(R) Core(TM) i7-1370P.

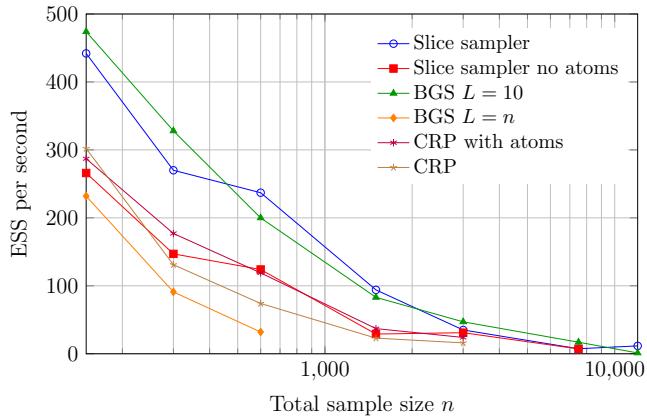


Figure 3: ESS (w.r.t. to the log likelihood) per second computed as average over the last 5,000 iterations (excluding a burn-in of 5,000) as a function of input size (x-axis on log scale). Codes are in R and run on a laptop with 13th Gen Intel(R) Core(TM) i7-1370P.

complexity bounds established in Section 3, supporting the scalability of slice-based approaches for posterior inference in DP-based models.

To conclude, we empirically assess the limitations of blocked Gibbs samplers compared to slice samplers with a second numerical experiment. Here, the synthetic data-generating mechanism follows a *perturbed Zipf* model, which entails a steadily increasing cluster-growth regime, up to elevated sample sizes, while still having finite support over the cluster labels. Specifically, for each sample size $n \in \{150, 300, 600, 1500, 3000\}$, latent cluster labels are generated independently from a discrete distribution on $\{1, \dots, 500\}$, with the probability of cluster c proportional to c^{-2} . This induces a heavy-tailed cluster-size distribution with a growing number of small but non-negligible clusters, providing a challenging setting for truncated algorithms. Conditional on the latent labels, observations are drawn independently from Normal distributions with unit variance and cluster-specific means proportional to the label index. With the exception of the data-generating process, all other settings coincide with

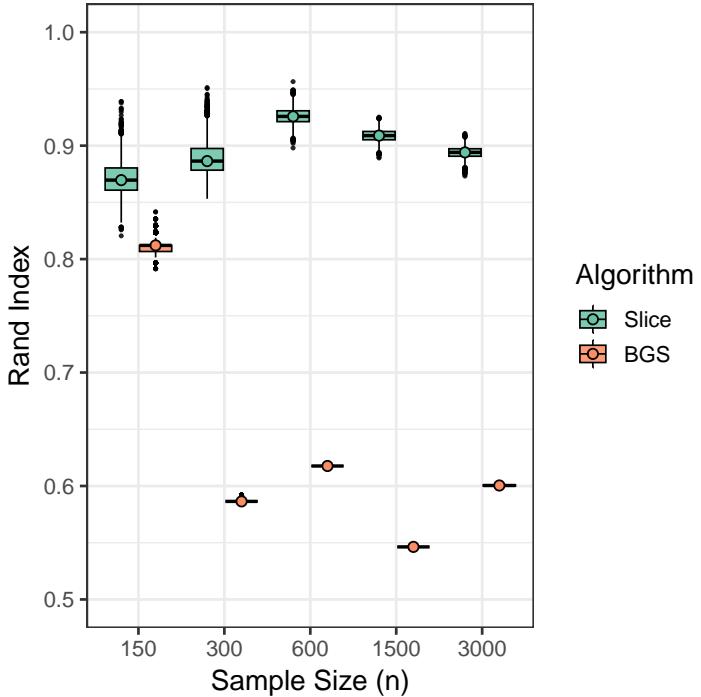


Figure 4: Boxplots of the Rand index between the true clustering and the partitions visited by the MCMC algorithms, for each algorithm and sample size. Results are computed after a burn-in period of 5,000 iterations.

those of the previous numerical study.

The results are reported in Figure 4. As expected, the inferential performance of the blocked Gibbs sampler deteriorates rapidly as the sample size increases, with the chain exhibiting very limited mixing and remaining almost constant on a single partition. In contrast, the slice sampler consistently achieves Rand index values around 0.9 across all sample sizes, indicating excellent recovery of the true clustering even in this challenging scenario.

Additional results, regarding posterior inference performances and computational aspects for both numerical studies, can be found in Appendix B.

6 Conclusion

Slice samplers provide an appealing strategy for posterior simulation in DP-based models, combining joint updates, minimal bookkeeping, and the avoidance of fixed truncation. At the same time, their practical scalability has long remained theoretically unclear due to the random and unbounded nature of their per-iteration computational cost. This work addresses this gap by providing the first high-probability complexity guarantees for DP slice samplers that hold *a posteriori*, uniformly over all partitions visited by the Markov chain and for any observed dataset. Our main results establish that the dynamically instantiated truncation level exceeds the number of occupied clusters by at most a

logarithmic factor in the sample size, with arbitrarily high probability. As a consequence, even in worst-case cluster-growth regimes, super-linear increases in per-iteration cost occur with vanishing probability. These guarantees formalize and explain the favorable empirical scalability of slice-based algorithms observed in practice, while preserving exact targeting of the posterior distribution over partitions.

An additional feature of our results is that all constants appearing in the high-probability bounds are explicit in the DP concentration parameter α . This opens the door to extensions to adaptive and data-driven DP formulations, such as those considered by [Tsiligkaridis and Forsythe \(2015\)](#) and [Ohn and Lin \(2023\)](#), where α is learned from the data or evolves over time. More broadly, the techniques developed here suggest a promising avenue for extending high-probability complexity guarantees to slice-based algorithms for general completely random measures, Pitman–Yor processes, and structured DP-based models, including hierarchical Dirichlet processes, sticky constructions, and temporally evolving partition models. In many of these settings, the conditional formulation underlying slice samplers leads to substantially simpler full conditional distributions than those arising in marginal approaches, hence the computational gains relative to alternative inference schemes are expected to be even more pronounced.

References

Amini, A. A., Paez, M., Lin, L., and Razaee, Z. S. (2019). Exact slice sampler for hierarchical Dirichlet processes. *arXiv preprint arXiv:1903.08829*.

Balocchi, C., Favaro, S., and Naulet, Z. (2024). Bayesian nonparametric inference for species-sampling problems. *Statistical Science, in press*.

Boucheron, S., Lugosi, G., and Massart, P. (2013). *Concentration Inequalities: A Nonasymptotic Theory of Independence*. Oxford University Press.

Cai, D., Campbell, T., and Broderick, T. (2016). Edge-exchangeable graphs and sparsity. *Advances in Neural Information Processing Systems*.

Campbell, T., Huggins, J. H., How, J. P., and Broderick, T. (2019). Truncated random measures. *Bernoulli*, 25(2):1256–1288.

Chakrabarti, A., Ni, Y., Pati, D., and Mallick, B. (2025). Global-local Dirichlet processes for clustering grouped data in the presence of group-specific idiosyncratic variables. *International Conference on Machine Learning*.

Dunson, D. B. and Park, J.-H. (2008). Kernel stick-breaking processes. *Biometrika*, 95(2):307–323.

Ferguson, T. S. (1973). A Bayesian analysis of some nonparametric problems. *The Annals of Statistics*, 1(2):209–230.

Ge, H., Chen, Y., Wan, M., and Ghahramani, Z. (2015). Distributed inference for Dirichlet process mixture models. *International Conference on Machine Learning*.

Ishwaran, H. and James, L. F. (2001). Gibbs sampling methods for stick-breaking priors. *Journal of the American Statistical Association*, 96(453):161–173.

Ishwaran, H. and James, L. F. (2002). Approximate Dirichlet process computing in finite normal mixtures: smoothing and prior information. *Journal of Computational and Graphical Statistics*, 11(3):508–532.

Ishwaran, H. and Zarepour, M. (2002). Exact and approximate sum representations for the Dirichlet process. *Canadian Journal of Statistics*, 30(2):269–283.

Kalli, M., Griffin, J. E., and Walker, S. G. (2011). Slice sampling mixture models. *Statistics and Computing*, 21(1):93–105.

Kemp, C., Tenenbaum, J. B., Griffiths, T. L., Yamada, T., and Ueda, N. (2006). Learning systems of concepts with an infinite relational model. In *Proceedings of the 21st National Conference on Artificial Intelligence*, pages 381–388.

Li, X. and Campbell, T. (2021). Truncated simulation and inference in edge-exchangeable networks. *Electronic Journal of Statistics*, 15(2):5117–5157.

Ling, Y., Li, J., Li, L., and Liang, S. (2024). Bayesian domain adaptation with Gaussian mixture domain-indexing. *Advances in Neural Information Processing Systems*.

Mei, L.-F. and Yan, W.-J. (2026). DPGIIL: Dirichlet process-deep generative model-integrated incremental learning for clustering in transmissibility-based online structural anomaly detection. *Advanced Engineering Informatics*, 69(B):103926.

Muliere, P. and Tardella, L. (1998). Approximating distributions of random functionals of Ferguson-Dirichlet priors. *Canadian Journal of Statistics*, 26(2):283–297.

Neal, R. M. (2000). Markov chain sampling methods for Dirichlet process mixture models. *Journal of Computational and Graphical Statistics*, 9(2):249–265.

Neal, R. M. (2003). Slice sampling. *The Annals of Statistics*, 31(3):705–767.

Ohn, I. and Lin, L. (2023). Optimal Bayesian estimation of Gaussian mixtures with growing number of components. *Bernoulli*, 29(2):1195 – 1218.

Papaspiliopoulos, O. and Roberts, G. O. (2008). Retrospective Markov chain Monte Carlo methods for Dirichlet process hierarchical models. *Biometrika*, 95(1):169–186.

Pitman, J. (1996). Some developments of the Blackwell–MacQueen urn scheme. In *Lecture Notes - Monograph Series*, Statistics, Probability and Game Theory: Papers in Honor of David Blackwell, pages 245–267.

Sethuraman, J. (1994). A constructive definition of Dirichlet priors. *Statistica Sinica*, 4(2):639–650.

Snell, J., Bencomo, G., and Griffiths, T. (2024). A metalearned neural circuit for nonparametric Bayesian inference. *Advances in Neural Information Processing Systems*.

Tsiligkaridis, T. and Forsythe, K. W. (2015). Adaptive low-complexity sequential inference for Dirichlet process mixture models. *Advances in Neural Information Processing Systems*.

Wade, S. and Inácio, V. (2025). Bayesian dependent mixture models: a predictive comparison and survey. *Statistical Science*, 40(1):81–108.

Walker, S. G. (2007). Sampling the Dirichlet mixture model with slices. *Communications in Statistics–Simulation and Computation*, 36(1):45–54.

Xu, Z., Tresp, V., Yu, K., and Kriegel, H.-P. (2006). Infinite hidden relational models. *Uncertainty in Artificial Intelligence*.

Zhou, Y., Gu, Y., and Dunson, D. B. (2024). Bayesian deep generative models for multiplex networks with multiscale overlapping clusters. *arXiv preprint arXiv:2405.20936*.

Zhu, P., Bouchard-Côté, A., and Campbell, T. (2020). Slice sampling for general completely random measures. *Uncertainty in Artificial Intelligence*.

A Proofs

Proof of Proposition 3.1. Fix $h \in [H]$ and let $I_h := \{i : c_i = h\}$ with $|I_h| = n_h$. Conditionally on π_h , the slice variables $(u_i)_{i \in I_h} \stackrel{\text{iid}}{\sim} \text{Uniform}(0, \pi_h)$. Hence

$$\mathbb{P}\left(\min_{i \in I_h} u_i > x \mid \pi_h\right) = \mathbb{P}\left(\bigcap_{i \in I_h} \{u_i > x\} \mid \pi_h\right) = \prod_{i \in I_h} \mathbb{P}(u_i > x \mid \pi_h).$$

For $u \sim \text{Uniform}(0, \pi_h)$ and $x \in (0, 1)$,

$$\mathbb{P}(u > x \mid \pi_h) = \max\left(1 - \frac{x}{\pi_h}, 0\right),$$

so that

$$\mathbb{P}\left(\min_{i \in I_h} u_i > x \mid \pi_h\right) = \left[\max\left(1 - \frac{x}{\pi_h}, 0\right)\right]^{n_h} = \max\left(\left(1 - \frac{x}{\pi_h}\right)^{n_h}, 0\right).$$

Now, consider the survival probability conditional on weights. Grouping indices by cluster, we have

$$\mathbb{P}(u_{\min} > x \mid \pi_{1:H}, \rho_n) = \prod_{h=1}^H \mathbb{P}\left(\min_{i:c_i=h} u_i > x \mid \pi_h\right) = \prod_{h=1}^H \max\left(\left(1 - \frac{x}{\pi_h}\right)^{n_h}, 0\right),$$

Denoting $(\cdot)_+ = \max(\cdot, 0)$ and taking expectation w.r.t. the Dirichlet posterior law for the weights, *i.e.*, $(\pi_1, \dots, \pi_{H_n}, \pi^*) \mid \rho_n \sim \text{Dirichlet}(n_1, \dots, n_H, \alpha)$, yields

$$\mathbb{P}(u_{\min} > x \mid \rho_n) = \mathbb{E}\left[\prod_{h=1}^H \left(1 - \frac{x}{\pi_h}\right)_+^{n_h} \mid \rho_n\right]. \quad (4)$$

Now merge clusters r and s . Let $\tilde{\pi} := \pi_r + \pi_s$ and $S := \pi_r / \tilde{\pi}$. By the aggregation and neutrality properties of the Dirichlet distribution,

$$(\tilde{\pi}, \pi_{-rs}, \pi^*) \mid \rho_n \sim \text{Dirichlet}(n_r + n_s, (n_h)_{h \neq r,s}, \alpha), \quad (5)$$

$S \mid \tilde{\pi}, \rho_n \sim \text{Beta}(n_r, n_s)$, and S is independent of $(\tilde{\pi}, \pi_{-rs}, \pi^*)$. Note that the distribution in (5) coincides with the Dirichlet posterior law for the weights when conditioning on the partition $\rho_n^{(r \oplus s)}$. Now, conditioning on $(\tilde{\pi}, \pi_{-rs}, \pi^*)$ and integrating S ,

$$\mathbb{P}(u_{\min} > x \mid \rho_n) = \mathbb{E}\left[\prod_{h \neq r,s} \left(1 - \frac{x}{\pi_h}\right)_+^{n_h} \mathbb{E}\left[\left(1 - \frac{x}{S\tilde{\pi}}\right)_+^{n_r} \left(1 - \frac{x}{(1-S)\tilde{\pi}}\right)_+^{n_s} \mid \tilde{\pi}, \rho_n\right] \mid \rho_n\right], \quad (6)$$

where the outer expectation is w.r.t. $(\tilde{\pi}, \pi_{-rs}, \pi^*)$.

Under the merged partition $\rho_n^{(r \oplus s)}$, the two clusters are replaced by a single cluster of size $n_r + n_s$ with weight $\tilde{\pi}$, while all other weights have the same joint law as in the aggregated representation above, and, in particular, weights follow the distribution in (5). Hence, analogously to (4),

$$\mathbb{P}(u_{\min} > x \mid \rho_n^{(r \oplus s)}) = \mathbb{E}\left[\left\{\prod_{h \neq r,s} \left(1 - \frac{x}{\pi_h}\right)_+^{n_h}\right\} \left(1 - \frac{x}{\tilde{\pi}}\right)_+^{n_r + n_s} \mid \rho_n^{(r \oplus s)}\right]. \quad (7)$$

It therefore suffices to show that

$$\mathbb{E} \left[\left(1 - \frac{x}{S\tilde{\pi}} \right)_+^{n_r} \left(1 - \frac{x}{(1-S)\tilde{\pi}} \right)_+^{n_s} \mid \tilde{\pi}, \rho_n \right] \leq \left(1 - \frac{x}{\tilde{\pi}} \right)_+^{n_r+n_s} \quad \text{a.s.} \quad (8)$$

Indeed, plugging (8) into (6) and comparing with (7) yields $\mathbb{P}(u_{\min} > x \mid \rho_n) \leq \mathbb{P}(u_{\min} > x \mid \rho_n^{(r \oplus s)})$, which is equivalent to the desired CDF inequality.

To verify (8), note first that if $\tilde{\pi} \leq x$ then both sides are zero. Assume $\tilde{\pi} > x$ and define $y := x/\tilde{\pi} \in (0, 1)$. The left-hand side integrand vanishes unless jointly $S > y$ and $1 - S > y$, *i.e.*, $y \leq S \leq 1 - y$. On this event, $\frac{y}{S} \geq y$ and $\frac{y}{1-S} \geq y$, which implies

$$1 - \frac{y}{S} \leq 1 - y \quad \text{and} \quad 1 - \frac{y}{1-S} \leq 1 - y.$$

Therefore, for all $S \in (y, 1 - y)$,

$$F(S) = \left(1 - \frac{y}{S} \right)^{n_r} \left(1 - \frac{y}{1-S} \right)^{n_s} \leq (1-y)^{n_r} (1-y)^{n_s} = (1-y)^{n_r+n_s},$$

and since $F(S) = 0$ for $S \notin (y, 1 - y)$, the bound holds for all $S \in (0, 1)$. Taking expectation with respect to $S \sim \text{Beta}(n_r, n_s)$ yields

$$\mathbb{E}[F(S)] \leq (1-y)^{n_r+n_s} = \left(1 - \frac{x}{\tilde{\pi}} \right)^{n_r+n_s}.$$

which is exactly (8). \square

Proof of Corollary 3.2. Fix a partition ρ_n with H clusters. Starting from the singleton partition ρ_n^{sing} , one can obtain ρ_n by a finite sequence of merges of two clusters: at each step, merge two blocks until exactly the blocks of ρ_n are formed. Denote the resulting sequence of partitions by

$$\rho_n^{(0)} = \rho_n^{\text{sing}}, \rho_n^{(1)}, \dots, \rho_n^{(m)} = \rho_n.$$

By Proposition 3.1, each merge weakly increases the survival probability of u_{\min} , *i.e.*, for every $j = 1, \dots, m$,

$$\mathbb{P}(u_{\min} > x \mid \rho_n^{(j-1)}) \leq \mathbb{P}(u_{\min} > x \mid \rho_n^{(j)}).$$

Chaining these inequalities yields

$$\mathbb{P}(u_{\min} > x \mid \rho_n^{\text{sing}}) \leq \mathbb{P}(u_{\min} > x \mid \rho_n).$$

\square

Proof of Theorem 3.3. Let ρ_n be the random partition of $[n]$ induced by (z_1, \dots, z_n) at a given iteration of Algorithm 2, and let H_n denote the number of clusters. Let $(c_1, \dots, c_n) \in [n]^n$ be any cluster-label vector compatible with ρ_n . Conditionally on ρ_n , draw weights $(\pi_h)_{h \geq 1}$ from the Dirichlet process posterior (as detailed in steps 4–6 of Algorithm 2) and let

$$u_i \sim \text{Uniform}(0, \pi_{c_i}), \quad u_{\min} = \min_{1 \leq i \leq n} u_i.$$

For each integer $k \geq 1$, define

$$R_k = \sum_{h \geq k+1} \pi_h, \quad \text{and} \quad K_n = \min\{k \geq H_n : R_k < u_{\min}\}.$$

Recalling the *a posteriori* representation of allocated components, *i.e.*,

$$(\pi_1, \dots, \pi_H, \pi^*) \sim \text{Dirichlet}(n_1, \dots, n_H, \alpha)$$

, with α being the concentration parameter, and the stick-breaking construction of the weights $(\pi_h)_{h \geq 1}$, for any $k \geq H_n$ we have

$$R_k = \pi^* \times \prod_{i=H_n+1}^k (1 - V_i)$$

where we use the convention that $\prod_{i=h+1}^h y_i := 1$ for any h and $(y_i)_i$. The distribution of each stick-breaking weight V_i , for $i > H_n$, conditionally on ρ_n , is $V_i \sim \text{Beta}(1, \alpha)$. So that

$$K_n = \min \left\{ k \geq H_n : \pi^* \times \prod_{i=H_n+1}^k (1 - V_i) < u_{\min} \right\}.$$

Let us define for any $x \in (0, 1)$

$$K_n(x) := \min \left\{ k > H_n : \pi^* \times \prod_{i=H_n+1}^k (1 - V_i) < x \right\} \quad (9)$$

and

$$K_n^0(x) := \min \left\{ k > H_n : \prod_{i=H_n+1}^k (1 - V_i) < x \right\} \quad (10)$$

where, since the products are decreasing in k , both the sets in (9) and (10) are non-empty for any x .

Similarly, since $\pi^* < 1$ a.s., we have

$$\left\{ k > H_n : \pi^* \times \prod_{i=H_n+1}^k (1 - V_i) < x \right\} \supseteq \left\{ k > H_n : \prod_{i=H_n+1}^k (1 - V_i) < x \right\}$$

for any $x \in (0, 1)$, taking the minimum

$$K_n(x) \leq K_n^0(x) \quad \text{a.s.} \quad \forall x \in (0, 1). \quad (11)$$

It is easy to see that $u_{\min} > x$ implies

$$\left\{ k > H_n : \pi^* \times \prod_{i=H_n+1}^k (1 - V_i) < x \right\} \subseteq \left\{ k \geq H_n : \pi^* \times \prod_{i=H_n+1}^k (1 - V_i) < u_{\min} \right\}.$$

Hence, using again a minimum argument, $K_n(x) \geq K_n$. By negation then, $K_n(x) < K_n$ implies $u_{\min} \leq x$, which means

$$\mathbb{P}(K_n(x) < K_n \mid \rho_n) \leq \mathbb{P}(u_{\min} \leq x \mid \rho_n) \quad (12)$$

for any $x \in (0, 1)$.

Now, for any non-negative quantity $A_{n,\delta}$, possibly dependent on n and δ , we have

$$\begin{aligned}\mathbb{P}(K_n > A_{n,\delta} \mid \rho_n) &\leq \mathbb{P}(K_n^0(x) > A_{n,\delta} \mid \rho_n) + \mathbb{P}(K_n > K_n^0(x) \mid \rho_n) \\ &\leq \mathbb{P}(K_n^0(x) \geq A_{n,\delta} \mid \rho_n) + \mathbb{P}(K_n > K_n(x) \mid \rho_n) \\ &\leq \mathbb{P}(K_n^0(x) \geq A_{n,\delta} \mid \rho_n) + \mathbb{P}(u_{\min} \leq x \mid \rho_n)\end{aligned}$$

where we used in the first inequality the law of total probability disintegrating with respect to the event $\{K_n^0(x) \geq K_n\}$ and its complement, in the second (11) and in the third (12). Hence, choosing $A_{n,\delta} = H_n + C_\delta \log n$, we just need to find $x = x(n, \delta)$ such that, for any $n \geq 2$

$$\mathbb{P}(K_n^0(x(n, \delta)) \geq H_n + C_\delta \log n \mid \rho_n) \leq \frac{\delta}{2} \quad \text{and} \quad \mathbb{P}(u_{\min} \leq x(n, \delta) \mid \rho_n) \leq \frac{\delta}{2}. \quad (13)$$

For the first inequality, we notice, as in [Muliere and Tardella \(1998\)](#) and [Walker \(2007\)](#), that, conditionally on ρ_n , we have $-\log(1 - V_i) \stackrel{\text{iid}}{\sim} \exp(\alpha)$ for $i > H_n$, which makes $\sum_{i=H_n+1}^k -\log(1 - V_i)$ the arrival times of a homogeneous Poisson process of rate α . Hence, by its definition in (10), we have

$$K_n^0(x) - H_n - 1 \mid \rho_n \sim \text{Poisson}(\alpha \log(1/x)) \quad (14)$$

for any $x \in (0, 1)$. We leverage the following Chernoff-style result for Poisson random variables: if $Z \sim \text{Poisson}(\lambda)$ then

$$\mathbb{P}(Z \geq \lambda + t) \leq \exp\left\{-\frac{t^2}{2(\lambda + t/3)}\right\} \quad (15)$$

for any $t > 0$. This can be easily proved, *e.g.* by applying the Bernstein inequality in Equation 2.10 of [Boucheron et al. \(2013\)](#) to a sequence of n independent Bernoulli(λ/n) and then passing to the limit. Hence, by (14) and (15), for any $t > 0$ and any $x \in (0, 1)$, we have

$$\mathbb{P}(K_n^0(x) \geq H_n + 1 + \alpha \log(1/x) + t \mid \rho_n) \leq \exp\left\{-\frac{t^2}{2(\alpha \log(1/x) + t/3)}\right\}. \quad (16)$$

We note that the bound is independent of ρ_n .

The r.h.s. of (16) is bounded by $\delta/2$ if and only if

$$t^2 - \frac{2}{3} \log(2/\delta)t - 2 \log(2/\delta) \alpha \log(1/x) \geq 0 \quad (17)$$

for any $x \in (0, 1)$. The positive root of (17) is

$$\begin{aligned}t_+ &= \frac{1}{3} \log(2/\delta) + \frac{1}{2} \sqrt{\frac{4}{9} \log^2(2/\delta) + 8\alpha \log(2/\delta) \log(1/x)} \\ &\leq \frac{2}{3} \log(2/\delta) + \sqrt{2 \log(2/\delta) \alpha \log(1/x)} \leq \log(2/\delta) + \frac{3\alpha}{2} \log(1/x)\end{aligned} \quad (18)$$

where we used that $\sqrt{y_1 + y_2} \leq \sqrt{y_1} + \sqrt{y_2}$ and that $\sqrt{2y_1 y_2} \leq \eta y_1/2 + y_2/\eta$ for any $\eta > 0$, and in particular, we set $\eta = 2/3$. Since the l.h.s of (17) is increasing around t_+ , from (16) and (18) we have that

$$\mathbb{P}(K_n^0(x) - H_n \geq 1 + 3\alpha \log(1/x) + \log(2/\delta) \mid \rho_n) \leq \frac{\delta}{2} \quad (19)$$

for any $x \in (0, 1)$.

Now, for the second inequality in (13), we note that by Proposition 3.1 and Corollary 3.2

$$\mathbb{P}(u_{\min} \leq x \mid \rho_n) \leq \mathbb{P}(u_{\min} \leq x \mid \rho^{\text{sing}}) \quad (20)$$

where ρ^{sing} is the singleton partition and $(c_1^{\text{sing}}, \dots, c_n^{\text{sing}})$ a correspondent cluster-label vector, *i.e.*, $c_i^{\text{sing}} \neq c_j^{\text{sing}}$ for any $i \neq j \in [n]$. Then, we use the following minimum argument and union bound

$$\mathbb{P}(u_{\min} \leq x \mid \rho^{\text{sing}}) \leq \mathbb{P}\left(\bigcup_{i=1}^n \{u_i \leq x\} \mid \rho^{\text{sing}}\right) \leq \sum_{i=1}^n \mathbb{P}(u_i \leq x \mid \rho^{\text{sing}}). \quad (21)$$

Focusing on each summand on the r.h.s. of (21), let $w_i := \pi_{c_i^{\text{sing}}}$, *i.e.*, the weight associated to the allocation of the i -th observation, so that the slice variable u_i^{sing} , conditionally on w_i , is uniform on $(0, w_i)$. In particular, for any $x \in (0, 1)$ we have

$$\mathbb{P}(u_i \leq x \mid w_i, \rho^{\text{sing}}) = \min\left(1, \frac{x}{w_i}\right).$$

Moreover,

$$\left(w_1, \dots, w_n, 1 - \sum_{i=1}^n w_i\right) \mid \rho^{\text{sing}} \sim \text{Dirichlet}(1, \dots, 1, \alpha)$$

and, in particular, $w_i \mid \rho^{\text{sing}} \sim \text{Beta}(1, \alpha + n - 1)$. Thus, for any $x \in (0, 1)$

$$\begin{aligned} \mathbb{P}(u_i \leq x \mid \rho^{\text{sing}}) &= (\alpha + n - 1) \left[\int_0^x (1-w)^{\alpha+n-2} dw + x \int_x^1 \frac{(1-w)^{\alpha+n-2}}{w} dw \right] \\ &\leq (\alpha + n - 1)x[1 + \log(1/x)] \end{aligned} \quad (22)$$

where we use that $1 - w < 1$. Combining (20), (21) and (22) yields

$$\mathbb{P}(u_{\min} \leq x \mid \rho_n) \leq n(\alpha + n - 1)x[1 + \log(1/x)]. \quad (23)$$

We note that the bound is again independent of ρ_n .

Therefore, we need to choose $x = x(n, \delta) \in (0, 1)$ such that, for any $n \geq N$, the lower-bound in the probability in (19) is upper-bounded by $C_\delta \log n$, and the r.h.s. of (23) is upper-bounded by $\delta/2$. We prove that

$$x(n, \delta) := \frac{\delta}{4n(\alpha + n - 1) \log(2n(\alpha + n - 1)e/\delta)} \quad (24)$$

fulfills both the requirements for $n \geq 2$. Let $r = \frac{\delta}{2n(\alpha + n - 1)}$. For any $n \geq 2$, we have $0 < r < 1$, since $2n(\alpha + n - 1) > 1$, hence $x(n, \delta) < 1$. Moreover

$$\begin{aligned} x(n, \delta) \left[1 + \log \frac{1}{x(n, \delta)}\right] &= \frac{r}{2 \log(e/r)} \left[1 + \log \left(\frac{2 \log(e/r)}{r}\right)\right] \\ &= \frac{r}{2 \log(e/r)} [\log(e/r) + \log(2 \log(e/r))] \leq r \end{aligned} \quad (25)$$

since $\log(2y) \leq y$ for any $y > 0$. Combining (23) and (25) we have that $x(n, \delta)$ in (24) satisfies the second bound in (13). Moreover, first observe that

$$\log \frac{1}{x(n, \delta)} = \log(4/\delta) + \log n + \log(\alpha + n - 1) + \log \log \left(\frac{2n(\alpha + n - 1)e}{\delta} \right). \quad (26)$$

Since $\alpha + n - 1 \leq (1 + \alpha)n$ for $n \geq 2$, we have $\log(\alpha + n - 1) \leq \log(1 + \alpha) + \log n$, hence

$$\log \log \left(\frac{2n(\alpha + n - 1)e}{\delta} \right) \leq \log \left(\frac{2n(\alpha + n - 1)e}{\delta} \right) \leq 2 \log n + \log \frac{2e(1 + \alpha)}{\delta}. \quad (27)$$

Combining (26) and (27) yields, for all $n \geq 2$,

$$\log \frac{1}{f(n, \delta)} \leq 4 \log n + \log \left(\frac{8e(1 + \alpha)^2}{\delta^2} \right).$$

Therefore

$$1 + 3\alpha \log \frac{1}{x(n, \delta)} + \log \frac{2}{\delta} \leq 12\alpha \log n + \left(1 + 3\alpha \log \left(\frac{8e(1 + \alpha)^2}{\delta^2} \right) + \log \frac{2}{\delta} \right) \leq C_\delta \log n \quad (28)$$

where

$$C_\delta = B_\alpha^{(1)} + B_\alpha^{(2)} \log \frac{1}{\delta} \quad \text{with} \quad B_\alpha^{(2)} = \frac{6\alpha + 1}{\log 2}, \quad B_\alpha^{(1)} = 12\alpha + \frac{1 + 3\alpha \log(8e(1 + \alpha)^2) + \log 2}{\log 2} \quad (29)$$

and we used $\log 2 \leq \log n$ to collect all the terms in one constant.

Going back to (19), from (28), we get

$$\mathbb{P}(K_n^0(x(n, \delta)) - H_n > C_\delta \log n \mid \rho_n) \leq \mathbb{P}\left(K_n^0(x(n, \delta)) - H_n > 1 + 3\alpha \log \frac{1}{x(n, \delta)} + \log \frac{2}{\delta} \mid \rho_n\right) \leq \frac{\delta}{2}$$

which is the first bound in (13).

To sum up, we proved that, uniformly over any partition ρ_n visited by the posterior algorithm at any considered iteration, for any $\delta \in (0, 1)$ there exists C_δ such that for any $n \geq 2$

$$\mathbb{P}(K_n - H_n > C_\delta \log n \mid \rho_n) \leq \delta$$

where we give an explicit expression for the constant C_δ in (29). In particular, we have $K_n - H_n = O_{\mathbb{P}}(\log n)$ and in the worst-case scenario of $H_n = O(n)$, $K_n = O_{\mathbb{P}}(n)$. \square

Proof of Corollary 3.4. Let $X_n := \frac{K_n - H_n}{\log n}$. In the bound obtained in Theorem 3.3 one can simply take $\delta = e^{-t}$ for $t > 0$ and obtain, for any $n \geq 2$,

$$\mathbb{P}\left(X_n > B_\alpha^{(1)} + B_\alpha^{(2)}t \mid \rho_n\right) \leq e^{-t} \quad (30)$$

for any visited partition ρ_n , *i.e.*, an exponential bound on the tails of the normalized overhead, uniform in n and in ρ_n . To obtain the uniform bound on the p -th moment of X_n , we leverage the tail moment identity: for any nonnegative random variable Z

$$\mathbb{E}[Z^p] = \int_0^\infty ps^{p-1} \mathbb{P}(Z > s) ds$$

which we apply for $Z = (X_n - B_\alpha^{(1)})_+$, obtaining

$$\begin{aligned}\mathbb{E} \left[\left(X_n - B_\alpha^{(1)} \right)_+^p \mid \rho_n \right] &= \int_0^\infty ps^{p-1} \mathbb{P} \left(X_n > B_\alpha^{(1)} + s \mid \rho_n \right) ds \\ &\leq \int_0^\infty ps^{p-1} e^{-s/B_\alpha^{(2)}} ds = p \Gamma(p) (B_\alpha^{(2)})^p\end{aligned}\tag{31}$$

where we applied the bound in (30) with $s = tB_\alpha^{(2)}$. Now we simply observe that

$$X_n = (X_n - B_\alpha^{(1)})_+ + \min(X_n, B_\alpha^{(1)}),$$

and use the inequality $(a+b)^p \leq 2^{p-1}(a^p + b^p)$ valid for all $a, b \geq 0$ and $p \geq 1$. This yields

$$X_n^p \leq 2^{p-1} \left\{ (X_n - B_\alpha^{(1)})_+^p + (B_\alpha^{(1)})^p \right\}$$

Hence, taking conditional expectation and using (31), we have

$$\mathbb{E} [X_n^p \mid \rho_n] \leq 2^{p-1} \left\{ (B_\alpha^{(1)})^p + p \Gamma(p) (B_\alpha^{(2)})^p \right\} := C_{\alpha,p}$$

which, taking the supremum over $n \geq 2$, gives for any ρ_n

$$\sup_{n \geq 2} \mathbb{E} [X_n^p \mid \rho_n] \leq C_{\alpha,p}.$$

□

Proof of Corollary 3.5. Given the result in Theorem 3.3, taking expectation w.r.t. the partition ρ_n we have

$$\mathbb{P} \left(K_n - H_n > \left[B_\alpha^{(1)} + B_\alpha^{(2)} \log(1/\delta_n) \right] \log n \right) \leq \delta_n \tag{32}$$

for any vanishing sequence $(\delta_n)_{n \geq 1} \subset (0, 1/2)$. Taking $D_\alpha := \frac{B_\alpha^{(1)}}{\log 2} + B_\alpha^{(2)}$ we have

$$\left[B_\alpha^{(1)} + B_\alpha^{(2)} \log(1/\delta_n) \right] \log n \leq D_\alpha \log(1/\delta_n) \log n$$

since $\log 2 < \log(1/\delta_n)$. Hence (32) becomes

$$\mathbb{P} (K_n - H_n > D_\alpha \log(1/\delta_n) \log n) \leq \delta_n.$$

Finally, if $\sum_{n \geq 1} \delta_n < \infty$, then by Borel-Cantelli

$$\mathbb{P} \left(\limsup_{n \rightarrow \infty} \{K_n - H_n > D_\alpha \log(1/\delta_n) \log n\} \right) = 0.$$

□

B Additional details on numerical experiments

B.1 Three equally-balanced clusters

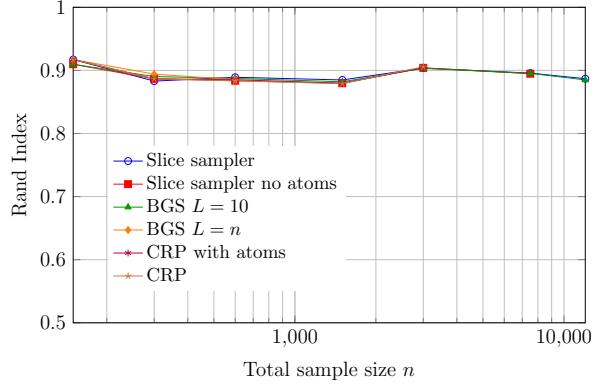


Figure 5: **Three equally-balanced clusters:** Rand index between the partition's point estimate and the true clustering configuration. The point estimate is obtained by minimizing the Binder loss function.

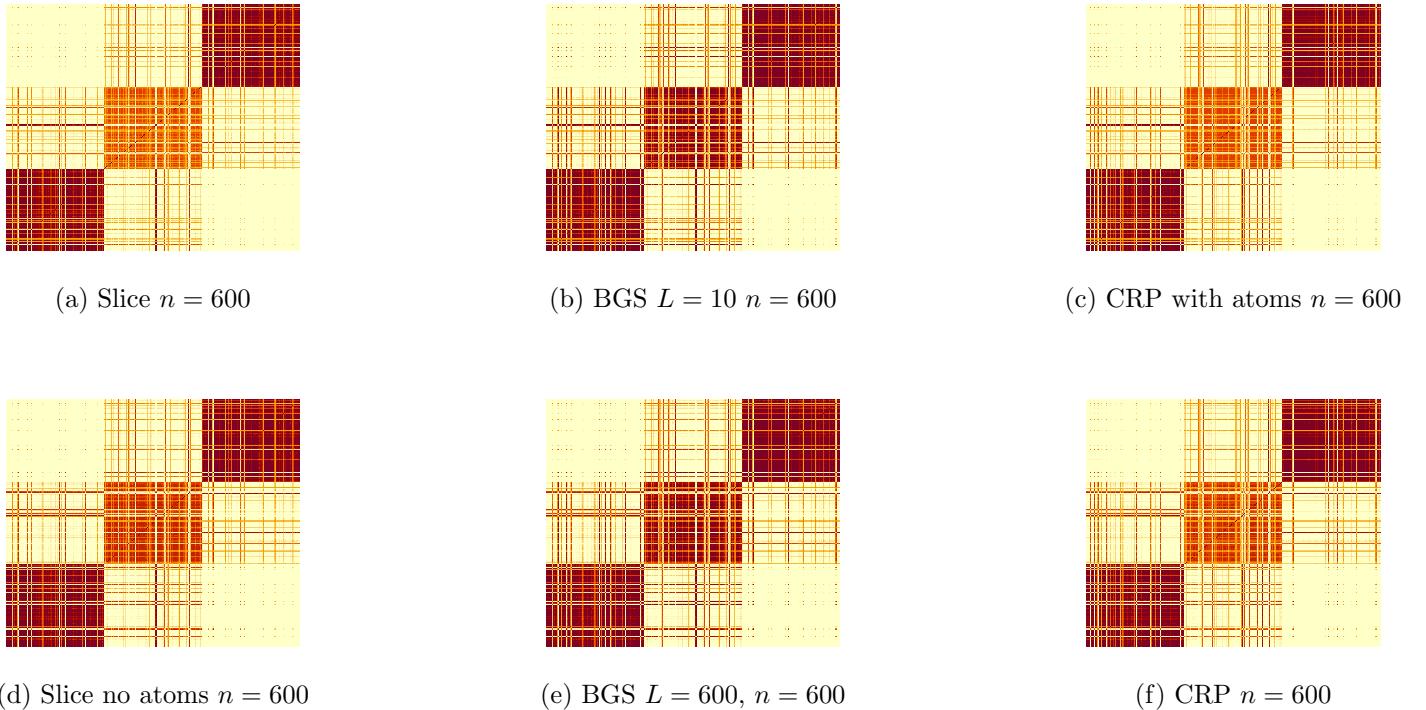


Figure 6: **Three equally-balanced clusters:** Posterior co-clustering matrices, computed discarding 5,000 iterations of burn-in.

B.2 Perturbed Zipf

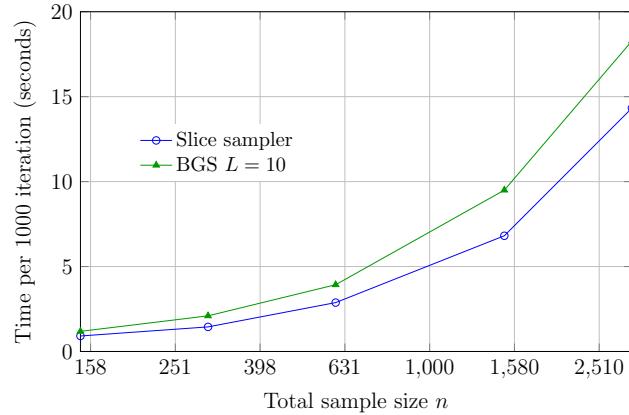


Figure 7: **Perturbed Zipf-generated data:** Time per 1000 iterations in seconds computed as average over 10,000 iterations (including burn-in) as a function of input size (x-axis on log scale). Codes are in R and run on a laptop with 13th Gen Intel(R) Core(TM) i7-1370P.

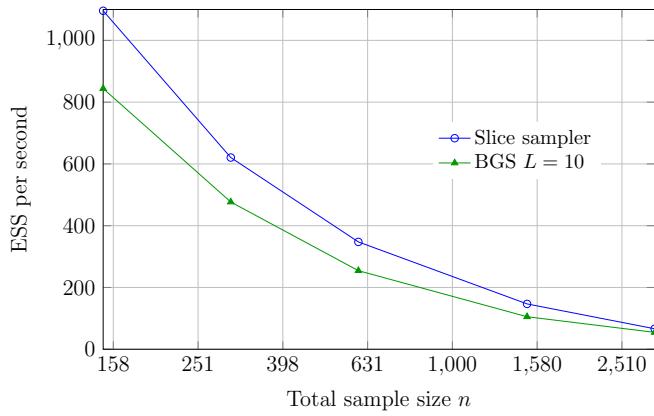


Figure 8: **Perturbed Zipf-generated data:** ESS (w.r.t. to the log likelihood) per second computed as average over the last 5,000 iterations (excluding a burn-in of 5,000) as a function of input size (x-axis on log scale). Codes are in R and run on a laptop with 13th Gen Intel(R) Core(TM) i7-1370P.

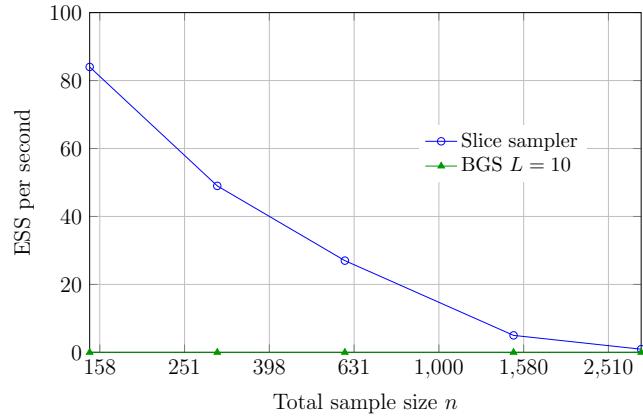


Figure 9: **Perturbed Zipf-generated data:** ESS (w.r.t. to the number of clusters) per second computed as average over the last 5,000 iterations (excluding a burn-in of 5,000) as a function of input size (x-axis on log scale). Codes are in R and run on a laptop with 13th Gen Intel(R) Core(TM) i7-1370P.

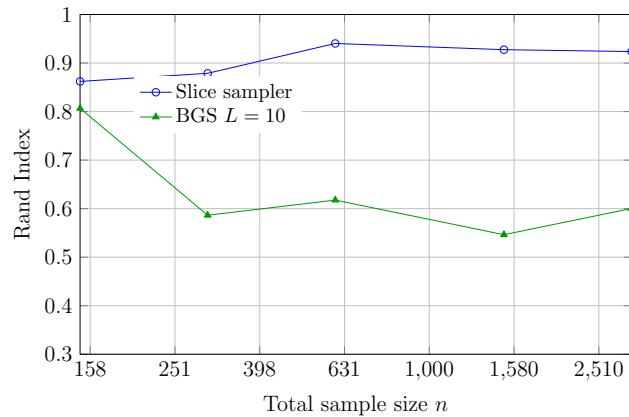
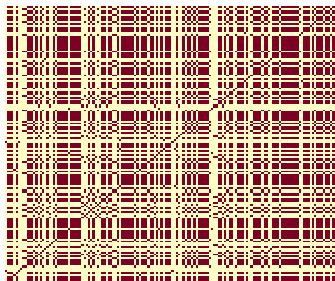
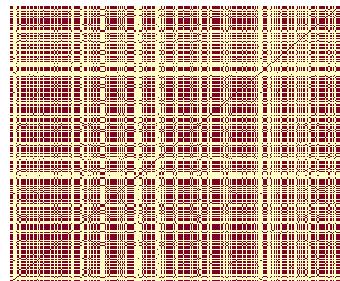


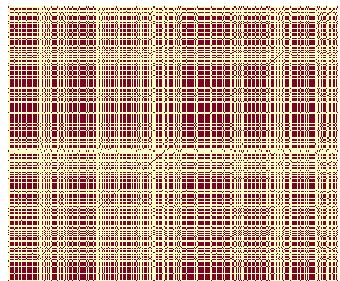
Figure 10: **Perturbed Zipf-generated data:** Rand index between the partition's point estimate and the true clustering configuration. The point estimate is obtained by minimizing the Binder loss function after discarding the first 5,000 iterations as burn-in.



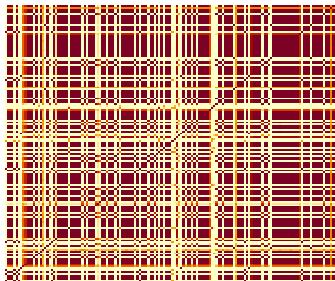
(a) Truth $n = 150$



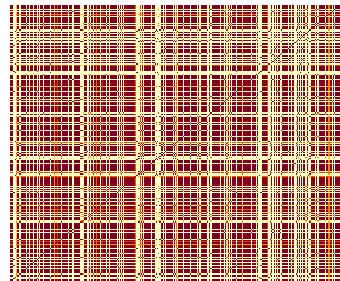
(b) Truth $n = 300$



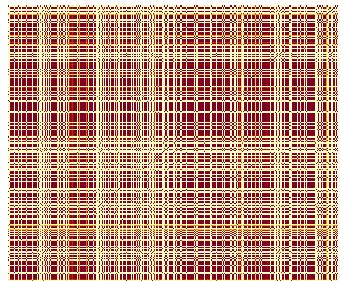
(c) Truth $n = 600$



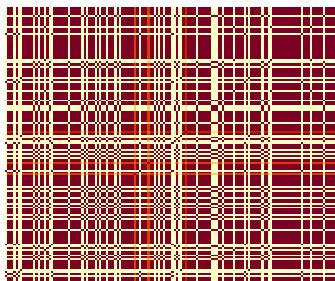
(d) Slice $n = 150$



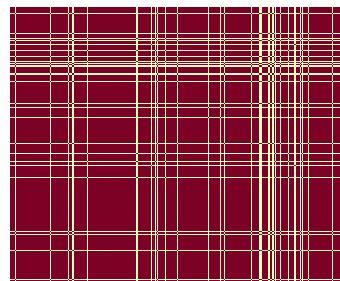
(e) Slice $n = 300$



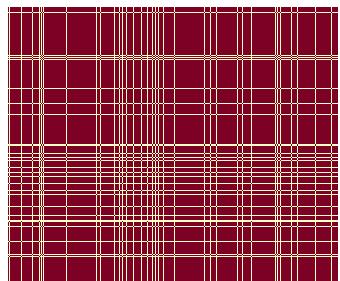
(f) Slice $n = 600$



(g) BGS $n = 150$



(h) BGS $n = 300$



(i) BGS $n = 600$

Figure 11: **Perturbed Zipf-generated data:** Posterior co-clustering matrices, computed discarding 5,000 iterations of burn-in.