Many Processors, Little Time: MCMC for Partitions via Optimal Transport Couplings

Tin D. Nguyen **Brian L. Trippe** **Tamara Broderick**
MIT LIDS

Abstract

Markov chain Monte Carlo (MCMC) methods are often used in clustering since they guarantee asymptotically exact expectations in the infinite-time limit. In finite time, though, slow mixing often leads to poor performance. Modern computing environments offer massive parallelism, but naive implementations of parallel MCMC can exhibit substantial bias. In MCMC samplers of continuous random variables, Markov chain couplings can overcome bias. But these approaches depend crucially on paired chains meetings after a small number of transitions. We show that straightforward applications of existing coupling ideas to discrete clustering variables fail to meet quickly. This failure arises from the "label-switching problem": semantically equivalent cluster relabelings impede fast meeting of coupled chains. We instead consider chains as exploring the space of partitions rather than partitions' (arbitrary) labelings. Using a metric on the partition space, we formulate a practical algorithm using optimal transport couplings. Our theory confirms our method is accurate and efficient. In experiments ranging from clustering of genes or seeds to graph colorings, we show the benefits of our coupling in the highly parallel, time-limited regime.

1 INTRODUCTION

Markov chain Monte Carlo (MCMC) is widely used in applications for exploring distributions over clusterings, or partitions, of data. For instance, Prabhakaran et al. [2016] use MCMC to approximate a Bayesian posterior over clusters of gene expression data for "discovery and characterization of cell types"; Chen et al. [2019] use MCMC to approximate the number of k-colorings of a graph; and DeFord et al. [2021] use MCMC to identify partisan gerrymandering via partitioning of geographical units into districts. An appealing feature of MCMC for many applications is that it yields asymptotically exact expectations in the infinite-time limit. However, real-life samplers must always be run in finite time, and MCMC mixing is often prohibitively slow in practice. While this slow mixing has led some practitioners to turn to other approximations such as variational Bayes [Blei and Jordan, 2006], these alternative methods can yield arbitrarily poor approximations of the expectation of interest [Huggins et al., 2020].

A different approach is to speed up MCMC, e.g. by taking advantage of recent computational advantages. While wall-clock time is often at a premium, modern computing environments increasingly offer massive parallel processing. For example, institute-level compute clusters commonly make hundreds of processors available to their users simultaneously [Reuther et al., 2018]. Recent efforts to enable parallel MCMC on graphics processing units [Lao et al., 2020] offer to expand parallelism further, with modern commodity GPUs providing over ten thousand cores. A naive approach to exploiting parallelism is to run MCMC separately on each processor; we illustrate this approach on a genetics dataset (GENE) in Figure 1 with full experimental details in Section 5. One might either directly average the resulting estimates across processors (red solid line in Figure 1) or use a robust averaging procedure (red dashed line in Figure 1). Massive parallelism can be used to reduce variance of the final estimate but does not mitigate the problem of bias, so the final estimate does not improve substantially as the number of processes increases.

Recently, Jacob et al. [2020] built on the work of Glynn and Rhee [2014] to eliminate bias in MCMC with a *coupling*. The basic idea is to cleverly set up dependence between two MCMC chains so that they are still practical to run and also meet exactly at a random

Proceedings of the 25th International Conference on Artificial Intelligence and Statistics (AISTATS) 2022, Valencia, Spain. PMLR: Volume 151.

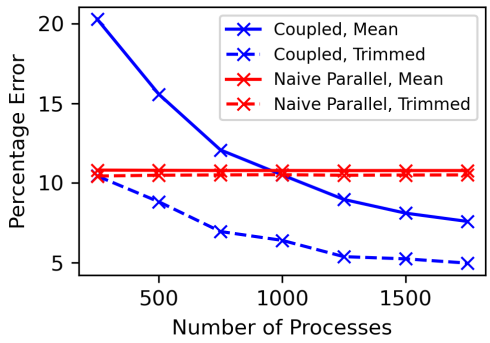


Figure 1: Lower error at high process count using our estimator (blue) versus using naive parallelism (red). For details, see Section 5.2.

but finite time. After meeting, these coupled chains can be used to compute an unbiased estimate of the expectation of interest. So arbitrarily large reductions in the estimate's variance due to massive parallelism translate directly into arbitrarily large reductions in total error. Since a processor's computation concludes after the chains meet, a useful coupling relies heavily on setting up coupled chains that meet quickly.

Jacob et al. [2020] did not consider MCMC over partitions in particular and Glynn and Rhee [2014] did not work on MCMC. But there is existing work on couplings applied to partitions in other contexts that can be adapted into the Jacob et al. [2020] framework. For instance, Jerrum [1998] uses maximal couplings on partition labelings to prove convergence rates for graph coloring, and Gibbs [2004] uses a common random number coupling for two-state Ising models. Though Jerrum [1998] was theoretical rather than practical and Gibbs [2004] did not apply to general partition models, we can adapt the Jacob et al. [2020] setup in a straightforward manner to use either coupling scheme. While this adaptation ensures asymptotically-unbiased MCMC samples, we will see (Section 5.3) that both schemes exhibit slow meeting times in practice. We attribute this issue to the *label-switching problem*, which is well-known for plaguing MCMC over partitions [Jasra et al., 2005]. In particular, many different labelings correspond to the same partition. In the case of couplings, two chains may nearly agree on the partition but require many iterations to change label assignments, so the coupling is unnecessarily slow to meet.

Our main contribution, then, is to propose and analyze a practical coupling that uses the unbiasedness of the [Jacob et al., 2020] framework but operates directly in the true space of interest – i.e., the space of partitions – to thereby exhibit fast meeting times. In particular, we define an optimal transport (OT) coupling in the partition space (Section 3). For clustering models, we prove that our coupling produces unbiased estimates (Section 4.1). We provide a big-O analysis to support the fast meeting times of our coupling (Section 4.2).We empirically demonstrate the benefits of our coupling on a simulated analysis; on Dirichlet process mixture models applied to real genetic, agricultural, and marine life data; and on a graph coloring problem. We show that, for a fixed wall time, our coupling provides much more accurate estimates and confidence intervals than naive parallelism (Section 5.2). And we show that our coupling meets much more quickly than standard label-based couplings for partitions (Section 5.3). Our code is available at `https://github.com/tinnguyen96/partition-coupling`.

Related work. Couplings of Markov chains have a long history in MCMC. But they either have primarily been a theoretical tool, do not provide guarantees of consistency in the limit of many processes, or are not generally applicable to Markov chains over partitions (Appendix A). Likewise, much previous work has sought to utilize parallelism in MCMC. But this work has focused on splitting large datasets into small subsets and running MCMC separately on each subset. But here our distribution of interest is over partitions of the data; combining partitions learned separately on multiple processors seems to face much the same difficulties as the original problem (Appendix A). Xu et al. [2021] have also used OT techniques within the Jacob et al. [2020] framework, but their focus was continuous-valued random variables. For partitions, OT techniques might most straightforwardly be applied to the label space – and we expect would fare poorly, like the other label-space couplings in Section 5.3. Our key insight is to work directly in the space of partitions.

2 SETUP

Before describing our method, we first review random partitions, set up Markov chain Monte Carlo for partitions – with an emphasis on Gibbs sampling, and review the Jacob et al. [2020] coupling framework.

2.1 Random Partitions

For a natural number N, a *partition* of $[N] := \{1, 2, \ldots, N\}$ is a collection of $K \leq N$ non-empty disjoint sets $\{A^1, A^2, \ldots, A^K\}$, whose union is $[N]$. In a clustering problem, we can think of A^k as containing the data indices in a particular cluster. Let $\mathcal{P}_N$ denote the set of all partitions of $[N]$. Let π denote an element of $\mathcal{P}_N$, and let Π be a random partition (i.e. a $\mathcal{P}_N$-valued random variable) with probability mass function (p.m.f.) p_Π. We report a summary that takes the form of an expectation: $H^* := \int h(\Pi) p_\Pi(\Pi) d\Pi$.

As an example, consider a Bayesian cluster analysis for N data points $\{W_n\}_{n=1}^N$, with $W_n \in \mathbb{R}^D$. A common generative procedure uses a Dirichlet process mixture model (DPMM) and conjugate Gaussian cluster likelihoods – with hyperparameters $\alpha > 0$, $\mu_0 \in \mathbb{R}^D$, and Σ_0, Σ_1 positive definite $D \times D$ matrices. First draw $\Pi = \pi$ with probability $\alpha^{|\pi|} \prod_{A \in \pi} (|A| - 1)! / [\alpha(\alpha+1)\cdots(\alpha+N-1)]$. Then draw cluster centers $\mu_A \overset{\text{i.i.d.}}{\sim} \mathcal{N}(\mu_0, \Sigma_0)$ for $A \in \Pi$ and observed data $W_j \,|\, \mu_A \overset{\text{i.i.d.}}{\sim} \mathcal{N}(\mu_A, \Sigma_1)$ for $j \in A$. The distribution of interest is the Bayesian posterior over Π: $p_\Pi(\pi) := \Pr(\Pi = \pi \,|\, W)$. A summary H^* of interest might be the posterior mean of the number of clusters for N data points or of the proportion of data in the largest cluster; see Appendix B for more discussion.

An assignment of data points to partitions is often encoded in a vector of labels. E.g., one might represent $\pi = \{\{1,2\},\{3\}\}$ with the vector $z = [1,1,2]$; z indicates that data points 1 and 2 are in the same cluster (arbitrarily labeled 1 here) while point 3 is in a different cluster (arbitrarily labeled 2). The partition can be recovered from the labeling, but the labels themselves are ancillary to the partition and, as we will see, can introduce unnecessary hurdles for fast MCMC mixing.

2.2 Markov Chain Monte Carlo

In the DPMM example and many others, the exact computation of the summary H^* is intractable, so Markov chain Monte Carlo provides an approximation. In particular, let X_t (for any t) denote a random partition; suppose we have access to a Markov chain $\{X_t\}_{t=0}^\infty$ with starting value X_0 drawn according to some initial distribution and evolving according to a transition kernel $X_t \sim T(X_{t-1}, \cdot)$ stationary with respect to p_Π. Then we approximate H^* with the empirical average of samples: $T^{-1} \sum_{t=1}^T h(X_t)$.

We focus on Gibbs samplers in what follows – since they are a convenient and popular choice for partitions [MacEachern, 1994, Neal, 2000, de Valpine et al., 2017]. We also extend our methods to more sophisticated samplers, such as split-merge samplers [Jain and Neal, 2004], that use Gibbs samplers as a sub-routine; see Section 3.3. To form a Gibbs sampler on the partition itself rather than the labeling, we first introduce some notation. Namely, let $\pi(-n)$ and $\Pi(-n)$ denote π and Π, respectively, with data point n removed. For example, if $\pi = \{\{1,3\},\{2\}\}$, then $\pi(-1) = \{\{3\},\{2\}\}$.

With this notation, we can write the *leave-out* conditional distributions of the Gibbs sampler as $p_{\Pi|\Pi(-n)}$. In particular, take a random partition X. Suppose $X(-n)$ has $K-1$ elements. Then the nth data point can either be added to an existing element or form a new element in the partition. Each of these K options forms a new partition; call the new partitions $\{\pi^k\}_{k=1}^K$. It follows that there exist $a_k \geq 0$ such that

$$p_{\Pi|\Pi(-n)}(\cdot \,|\, X(-n)) = \sum_{k=1}^K a_k \delta_{\pi^k}(\cdot), \tag{1}$$

where δ_{π^k} denotes a Dirac measure on π^k. When p_Π is available up to a proportionality constant, it is tractable to compute or sample from $p_{\Pi|\Pi(-n)}$.

Algorithm 1 shows one sweep of the resulting Gibbs sampler. For any X, the transition kernel $T(X, \cdot)$ for this sampler's Markov chain is the distribution of the output, $\widetilde{X}$, of Algorithm 1.

Input: Target p_Π. Current partition X.
$\widetilde{X} \leftarrow X$
for $n \leftarrow 1$ **to** N **do**
  $\widetilde{X} \sim p_{\Pi|\Pi(-n)}(\cdot \,|\, \widetilde{X}(-n))$
end
Return $\widetilde{X}$

Algorithm 1: Single Gibbs Sweep

2.3 An Unbiased Estimator

[Jacob et al., 2020] show how to construct an unbiased estimator of H^* for some Markov chain $\{X_t\}$ when an additional Markov chain $\{Y_t\}$ with two properties is available. First, $Y_t \,|\, Y_{t-1}$ must also evolve using the same transition $T(\cdot,\cdot)$ as $\{X_t\}$, so that $\{Y_t\}$ is equal in distribution to $\{X_t\}$. Second, there must exist a random *meeting time* $\tau < \infty$ with sub-geometric tails such that the two chains meet exactly at time τ ($X_\tau = Y_{\tau-1}$) and remain faithful afterwards (for all $t \geq \tau$, $X_t = Y_{t-1}$). When these properties hold, the following provides an unbiased estimate of H^*:

$$H_{\ell:m}(X,Y) := \underbrace{\frac{1}{m-\ell+1} \sum_{t=\ell}^m h(X_t)}_{\text{Usual MCMC average}} + \underbrace{\sum_{t=\ell+1}^{\tau-1} \min\left(1, \frac{t-\ell}{m-\ell+1}\right) \{h(X_t) - h(Y_{t-1})\}}_{\text{Bias correction}}, \tag{2}$$

where ℓ is the burn-in length and m sets a minimum number of iterations [Jacob et al., 2020, Equation 2]. ℓ and m are hyperparameters that impact the runtime and variance of $H_{\ell:m}$; for instance, smaller m is typically associated with smaller runtimes but larger variance. Jacob et al. [2020, Section 6] recommend setting ℓ to be a large quantile of the meeting time and m as a multiple of ℓ. We follow these recommendations in our work.

One interpretation of Equation (2) is as the usual MCMC estimate plus a bias correction. Since $H_{\ell:m}$ is unbiased, a direct average of many copies of $H_{\ell:m}$ computed in parallel can be made to have arbitrarily small error (for estimating H^*). It remains to apply the idea from Equation (2) to partition-valued chains.

2.4 Couplings

To create two chains of partitions that evolve together, we will need a joint distribution over partitions from both chains that respects the marginals of each chain. To that end, we define a coupling.

Definition 1 *A* coupling γ *of two discrete distributions,* $\sum_{k=1}^{K} a_k \delta_{\pi^k}(\cdot)$ *and* $\sum_{k'=1}^{K'} b_{k'} \delta_{\nu^{k'}}(\cdot)$*, is a distribution on the product space,*

$$\gamma(\cdot) = \sum_{k} \sum_{k'} u^{k,k'} \delta_{(\pi^k, \nu^{k'})}(\cdot), \tag{3}$$

that satisfies the marginal constraints $\sum_k u^{k,k'} = b_{k'}, \quad \sum_{k'} u^{k,k'} = a_k, \quad 0 \le u^{k,k'} \le 1.$

3 OUR METHOD

We have just described how to achieve unbiased estimates when two chains with a particular relationship are available. It remains to show that we can construct these chains so that they meet quickly in practice. First, we describe a general setup for a coupling of two Gibbs samplers over partitions in Section 3.1. Our method is a special case where we choose a coupling function that encourages the two chains to meet quickly (Section 3.2). We extend our coupling to split-merge samplers in Section 3.3. We employ a variance reduction procedure to further improve our estimates (Section 3.4).

3.1 Coupling For Gibbs On Partitions

Let X, Y be two partitions of $[N]$. By Equation (1), we can write $p_{\Pi|\Pi(-n)}(\cdot \,|\, X(-n)) = \sum_{k=1}^{K} a_k \delta_{\pi^k}(\cdot)$ for some K and tuples (a_k, π^k). And we can write $p_{\Pi|\Pi(-n)}(\cdot \,|\, Y(-n)) = \sum_{k'=1}^{K'} b_{k'} \delta_{\nu^{k'}}(\cdot)$ for some K' and tuples $(b_{k'}, \nu^{k'})$. We say that a coupling function is any function that returns a coupling for these distributions.

Definition 2 *A* coupling function ψ *takes as input a target* p_Π*, a leave-out index* n*, and partitions* X, Y*. It returns a coupling* $\gamma = \psi(p_\Pi, n, X, Y)$ *of* $p_{\Pi|\Pi(-n)}(\cdot \,|\, X(-n))$ *and* $p_{\Pi|\Pi(-n)}(\cdot \,|\, Y(-n))$*.*

Given a coupling function ψ, Algorithm 2 gives the coupled transition from the current pair of partitions (X, Y) to another pair $(\widetilde{X}, \widetilde{Y})$. Repeating this algorithm guarantees the first required property from the Jacob et al. [2020] construction in Section 2.3: co-evolution of the two chains with correct marginal distributions. It remains to show that we can construct an appropriate coupling function and that the chains meet (quickly).

Input: Target p_Π. Coupling function ψ. Current partitions X and Y.

$\widetilde{X} \leftarrow X, \widetilde{Y} \leftarrow Y$
for $n \leftarrow 1$ **to** N **do**
  $\gamma \leftarrow \psi(p_\Pi, n, \widetilde{X}, \widetilde{Y})$
  $(\widetilde{X}, \widetilde{Y}) \sim \gamma$
end
Return $\widetilde{X}, \widetilde{Y}$

Algorithm 2: Coupled Gibbs Sweep

3.2 An Optimal Transport Coupling

We next detail our choice of coupling function; namely, we start from an optimal transport (OT) coupling and add a nugget term for regularity. For a distance d between partitions, the OT coupling function $\psi^{\mathrm{OT}} = \psi^{\mathrm{OT}}(p_\Pi, n, X, Y)$ minimizes the expected distance between partitions after one coupled Gibbs step given partitions X, Y and leave-out index n. Using the notation of Sections 2.4 and 3.1, we define

$$\psi^{\mathrm{OT}} := \underset{\text{couplings } \gamma}{\arg\min} \sum_{k=1}^{K} \sum_{k'=1}^{K'} u^{k,k'} \mathrm{d}(\pi^k, \nu^{k'}). \tag{4}$$

To complete the specification of ψ^{OT}, we choose a metric d on partitions that was introduced by Mirkin and Chernyi [1970] and Rand [1971]:

$$\mathrm{d}(\pi, \nu) = \sum_{A \in \pi} |A|^2 + \sum_{B \in \nu} |B|^2 - 2 \sum_{A \in \pi, B \in \nu} |A \cap B|^2. \tag{5}$$

Observe that $\mathrm{d}(\pi, \nu)$ is zero when $\pi = \nu$. More generally, we can construct a graph from a partition by treating the indices in $[N]$ as vertex labels and assigning any two indices in the same partition element to share an edge; then $\mathrm{d}/2$ is equal to the Hamming distance between the adjacency matrices implied by π and ν [Mirkin and Chernyi, 1970, Theorems 2–3]. The principal trait of d for our purposes is that d steadily increases as π and ν become more dissimilar. In Appendix I, we discuss other potential metrics and show that an alternative with similar qualitative behavior yields essentially equivalent empirical results.

In practice, any standard optimal transport[1] solver can

[1]We note that the optimization problem defining Equation (4) is an *exact* transport problem, not an entropically-regularized transport problem [Cuturi, 2013]. Hence the marginal distributions defined by ψ^{OT} automatically match the inputs $p_{\Pi|\Pi(-n)}(\cdot \,|\, X(-n))$ and $p_{\Pi|\Pi(-n)}(\cdot \,|\, Y(-n))$, without need of post-processing.

be used in ψ^{OT}, and we discuss our particular choice in more detail in Section 4.2. To prove unbiasedness of a coupling (Theorem 1), it is convenient to ensure that every joint setting of (X, Y) is reachable from every other joint setting in the sampler. As we discuss after Theorem 1 and in Appendix C, adding a small nugget term to the coupling function accomplishes this goal. To that end, define the independent coupling ψ^{ind} to have atom size $u^{k,k'} = a_k b_{k'}$ at $(\pi^k, \nu^{k'})$. Let $\eta \in (0, 1)$. Then our final coupling function $\psi_\eta^{\mathrm{OT}} = \psi_\eta^{\mathrm{OT}}(p_\Pi, n, X, Y)$ equals

$$\begin{cases} \psi^{\mathrm{OT}}(X, X) & \text{if } X = Y \\ (1-\eta)\psi^{\mathrm{OT}}(X, Y) + \eta\psi^{\mathrm{ind}}(X, Y) & \text{else,} \end{cases} \tag{6}$$

where we elide the dependence on p_Π, n for readability. In practice, we set η to 10^{-5}, so the behavior of ψ_η^{OT} is dominated by ψ^{OT}.

As a check, notice that when two chains first meet, the behavior of ψ_η^{OT} reverts to that of ψ^{OT}. Since there is a coupling with expected distance zero, that coupling is chosen as the minimizer in ψ^{OT}. Therefore, the two chains remain faithful going forward.

3.3 Extension To Other Samplers

With ψ_η^{OT}, we can also couple samplers that use Gibbs sampling as a sub-routine; to illustrate, we next describe a coupling for a split-merge sampler [Jain and Neal, 2004]. Split-merge samplers pair a basic Gibbs sweep with a Metropolis-Hastings (MH) move designed to facilitate larger-scale changes across the clustering. In particular, the MH move starts from partition X by selecting a pair of distinct data indices (i, j) uniformly at random. If i and j belong to the same cluster, the sampler proposes to split this cluster. Otherwise, the sampler proposes to merge together the two clusters containing i and j. The proposal is accepted or rejected in the MH move. For our purposes, we summarize the full move, including proposal and acceptance but conditional on the choice of i and j, as $\widetilde{X} \sim \mathrm{SplitMerge}(i, j, X)$. One iteration of the split-merge sampler is identical to Algorithm 1, except that between lines 1 and 2 of Algorithm 1, we sample (i, j) and perform $\mathrm{SplitMerge}(i, j, \widetilde{X})$.

Algorithm 3 shows our coupling of a split-merge sampler. We use the same pair of indices (i, j) in the split-merge moves across both the X and Y chains. We use ψ_η^{OT} to couple at the level of the Gibbs sweeps.

Gibbs samplers and split-merge samplers offer differing strengths and weaknesses. For instance, the MH move may take long to finish; Algorithm 1 might run for more iterations in the same time, potentially producing better estimates sooner. The MH move is also more complex and thus potentially more prone to errors in implementation. In what follows, we consider both samplers; we compare our coupling to naive parallelism for Gibbs sampling in Section 5, and we make the analogous comparison for split-merge samplers in Appendix J.

Input: Target probability mass function (p.m.f.) p_Π. Current partitions X and Y.
Output: $\widetilde{X}, \widetilde{Y}$
1 $\widetilde{X} \leftarrow X, \widetilde{Y} \leftarrow Y$
2 $(i, j) \leftarrow$ Uniformly random pair of data indices
3 $\widetilde{X} \sim \mathrm{SplitMerge}(i, j, \widetilde{X})$
4 $\widetilde{Y} \sim \mathrm{SplitMerge}(i, j, \widetilde{Y})$
5 **for** $n \leftarrow 1$ **to** N **do**
6 $\quad \gamma \leftarrow \psi_\eta^{\mathrm{OT}}(p_\Pi, n, \widetilde{X}, \widetilde{Y})$
7 $\quad (\widetilde{X}, \widetilde{Y}) \sim \gamma$
8 **end**
9 Return $\widetilde{X}, \widetilde{Y}$
Algorithm 3: Coupled Gibbs Sweep with Split–Merge Move

3.4 Variance Reduction Via Trimming

We have described how to generate a single estimate of H^* from Equation (2); in practice, on the jth processor, we run chains X^j and Y^j to compute $H_{\ell:m}(X^j, Y^j)$. It remains to decide how to aggregate the observations $\{H_{\ell:m}(X^j, Y^j)\}_{j=1}^J$ across J processors.

A natural option is to report the sample mean, $\frac{1}{J}\sum_{j=1}^J H_{\ell:m}(X^j, Y^j)$. If each individual estimate is unbiased, the squared error of the sample mean decreases to zero at rate $1/J$. And standard confidence intervals have asymptotically correct coverage.

For finite J, though, there may be outliers that drive the sample mean far from H^*. To counteract the effect of outliers and achieve a lower squared error, we also report a classical robust estimator: the trimmed mean [Tukey and McLaughlin, 1963]. Recall that for $\alpha \in (0, 0.5)$, the *α-trimmed mean* is the average of the observations between (inclusive) the 100α quantile and the $100(1-\alpha)$ quantile of the observed data. The trimmed mean is asymptotically normally distributed [Bickel, 1965, Stigler, 1973] and provides sub-Gaussian confidence intervals [Lugosi and Mendelson, 2019]. See Appendix F for more discussion on the trimmed mean.

4 THEORETICAL RESULTS

To verify that our coupling is useful, we need to check that it efficiently returns accurate estimates. We first check that the coupled estimate $H_{\ell:m}(X, Y)$ at a single

processor is unbiased – so that aggregated estimates across processors can exhibit arbitrarily small squared loss. Second, we check that there is no undue computational cost of coupling relative to a single chain.

4.1 Unbiasedness

Jacob et al. [2020, Assumptions 1–3] give sufficient conditions for unbiasedness of Equation (2). We next use these to establish sufficient conditions that $H_{\ell:m}(X, Y)$ is unbiased when targeting a DPMM posterior.

Theorem 1 (Sufficient Conditions for Unbiased Estimation) *Let p_Π be the DPMM posterior in Section 2.1. Assume the following two conditions on ψ.*

(1) There exists $\epsilon > 0$ such that for all $n \in [N]$ and for all $X, Y \in \mathcal{P}_N$ such that $X \neq Y$, the output γ of the coupling function ψ satisfies

$$\forall k \in [K] \text{ and } k' \in [K'], \quad u^{k,k'} \geq \epsilon. \tag{7}$$

(2) If $X = Y$, then the output coupling γ of ψ satisfies $\gamma(\widetilde{X} = \widetilde{Y}) = 1$; i.e. the coupling is faithful.
Then, the estimator in Equation (2) constructed from Algorithm 2 is an unbiased estimator for H^*. Furthermore, Equation (2) has a finite variance and a finite expected computing time.

We prove Theorem 1 in Appendix C. Our proof exploits the discreteness of the sample space to ensure chains meet. Condition (1) roughly ensures that any joint state in the product space is reachable from any other joint state under the Gibbs sweep; we use it to establish that the meeting time τ has sub-geometric tails. Condition (2) implies that the Markov chains are faithful once they meet.

Corollary 1 *Let p_Π be the DPMM posterior. The Equation (2) estimator using Algorithm 2 with coupling function $\psi_\eta^{OT}(p_\Pi, n, X, Y)$ is unbiased for H^*.*

Proof It suffices to check Theorem 1's conditions. We show ψ_η^{OT} is faithful at the end of Section 3.2. For a partition, the associated leave-out distributions place positive mass on all K accessible atoms, so marginal transition probabilities are lower bounded by some $\omega > 0$. The nugget guarantees each $u^{k,k'} \geq \eta\omega^2 > 0$. ■

Note that the introduction of the nugget allows us to verify the first condition of Theorem 1 is met without relying on properties specific to the optimal transport coupling. We conjecture that one could analogously show unbiased estimates may be obtained using couplings of Markov chains defined in the label space by introducing a similar nugget to transitions on this alternative state space. Crucially, though, we will see in Section 5.3 that our coupling in the partition space exhibits much faster meeting times in practice than these couplings in the label space.

4.2 Time Complexity

The accuracy improvements of our method can be achieved only if the compute expense of coupling is not too high relative to single-chain Gibbs. In Section 5.2, we show empirically that our method outperforms naive parallel samplers run for the same wall time. Here we use theory to describe why we expect this behavior.

There are two key computations that must happen in any coupling Gibbs step within a sweep:
(1) computing the atom sizes $a_k, b_{k'}$ and atom locations $\pi^k, \nu^{k'}$ in the sense of Definition 1 and Definition 2;
(2) computing the pairwise distances $\mathrm{d}(\pi^k, \nu^{k'})$; and solving the optimal transport problem (Equation (4)).

Let $\beta(N, K)$ represent the time it takes to compute the Gibbs conditional $p_{\Pi|\Pi(-n)}$ for a partition of size K, and let $\widetilde{K}$ represent the size of the largest partition visited in any chain, across all processors, while the algorithm runs. Then part (1) takes $O(\beta(N, \widetilde{K}))$ time to run. For single chains, computing atom sizes and locations dominates the compute time; the computation required is of the same order, but is done for *one* chain, rather than two, on each processor. We show in Proposition 1 in Appendix D that part (2) can be computed in $O(\widetilde{K}^3 \log \widetilde{K})$ time. Proposition 1 follows from efficient use of data structures; naive implementations are more computationally costly. Note that the total running time for a full Gibbs sweep (Algorithm 1 or Algorithm 2) will be N times the single-step cost.

The extra cost of a coupling Gibbs step will be small relative to the cost of a single-chain Gibbs step, then, if $O(\widetilde{K}^3 \log \widetilde{K})$ is small relative to $O(\beta(N, \widetilde{K}))$.[2] As an illustrative example, consider again the DPMM application from Section 2.1. We start with a comparison that we suspect captures typical operating procedure, but we also consider a worst-case comparison.
Standard comparison: The direct cost of a standard Gibbs step is $\beta(N, K) = O(ND + KD^3)$ (see Proposition 2 in Appendix D). By Equation 3.24 in Pitman [2006], the number of clusters in a DPMM grows a.s. as $O(\log N)$ as $N \to \infty$.[3] If we take

[2] We show in Appendix D that, while there are also initial setup costs before running any Gibbs sweep, these costs do not impact the amortized complexity.

[3] Two caveats: (1) If a Markov chain is run long enough, it will eventually visit all possible cluster configurations. But if we run in finite time, it will not have time to explore every collection of clusters. So we assume $O(\log N)$ is a reasonable approximation of finite time. (2) Also note that the $\log N$ growth is for data generated from a DPMM whereas in real life we cannot expect data are perfectly simulated from the model.

$\widetilde{K} = O(\log N)$, $O(\widetilde{K}^3 \log \widetilde{K})$ will generally be smaller than $\beta(N, K) = O(ND + KD^3)$ for sufficiently large N.

Worst-case comparison: The complexity of a DPMM Gibbs step can be reduced to $\beta(N, K) = O(KD + D^3)$ through careful use of data structures and conditional conjugacy (see Proposition 2 in Appendix D). Still, the coupling cost $O(\widetilde{K}^3 \log \widetilde{K})$ is not much larger than the cost of this step whenever $\widetilde{K}$ is not much larger than D.

For our experiments, we run the standard rather than optimized Gibbs step due to its simplicity and use in existing work [e.g. de Valpine et al., 2017]. In e.g. our gene expression experiment with $D = 50$, we expect this choice has little impact on our results. Our Proposition 1 establishing $O(\widetilde{K}^3 \log \widetilde{K})$ for the optimal transport solver applies to Orlin's algorithm [Orlin, 1993]. However, convenient public implementations are not available. So instead we use the simpler network simplex algorithm [Kelly and O'Neill, 1991] as implemented by Flamary et al. [2021]. Although Kelly and O'Neill [1991, Section 3.6] upper bound the worst-case complexity of the network simplex as $O(\widetilde{K}^5)$, the algorithm's average-case performance may be as good as $O(\widetilde{K}^2)$ [Bonneel et al., 2011, Figure 6].

5 EMPIRICAL RESULTS

We now demonstrate empirically that our OT coupling (1) gives more accurate estimates and confidence intervals for the same wall time and processor budget as naive parallelism and (2) meets much faster than label-based couplings.

5.1 Models, Datasets, And Implementation

We run samplers for both clustering and graph coloring problems, which we describe next. We detail our construction of ground truth, sampler initialization, and algorithm hyperparameters (ℓ and m) in Appendix G.2.

Motivating examples and target models. For clustering, we use single-cell RNA sequencing data [Prabhakaran et al., 2016], X-ray data of agricultural seed kernels [Charytanowicz et al., 2010, Dua and Graff, 2017], physical measurements of abalone [Nash et al., 1994, Dua and Graff, 2017], and synthetic data from a Gaussian mixture model. In each case, our target model is the Bayesian posterior over partitions from the DPMM. For graph colorings, sampling from the uniform distribution on k-colorings of graphs is a key sub-routine in fully polynomial randomized approximation algorithms. And it suffices to sample from the partition distribution induced by the uniform distribution on k-colorings, which serves as our target model; see Appendix G.1 for details.

Summaries of interest. Our first summary is the mean proportion of data points in the largest cluster; we write LCP for "largest component proportion." See, e.g., Liverani et al. [2015] for its use in Bayesian analysis. Our second summary is the co-clustering probability; we write $\mathrm{CC}(a, b)$ for the probability that data points indexed by a and b belong to the same cluster. See, e.g., DeFord et al. [2021] for its use in redistricting. In Appendix M, we also report a more complex summary: the posterior predictive distribution, which is a quantity of interest in density estimation [Görür and Rasmussen, 2010, Escobar and West, 1995].

Dataset details. Our SYNTHETIC dataset has 300 observations and 2 covariates. Our GENE dataset originates from Zeisel et al. [2015] and was previously used by Prabhakaran et al. [2016] in a DPMM-based analysis. We use a subset with 200 observations and 50 covariates to allow us to quickly iterate on experiments. We use the unlabeled version of the SEED dataset from Charytanowicz et al. [2010], Dua and Graff [2017] with 210 observations and 7 covariates. For the ABALONE dataset from Nash et al. [1994], Dua and Graff [2017], we remove the labels and binary features, which yields 4177 observations and 7 covariates. For graph data (K-REGULAR), we use a 4-regular graph with 6 vertices; we target the partition distribution induced by the uniform distribution on 4-colorings.

5.2 Improved Accuracy With Coupling

In Figure 2, we first show that our coupling estimates and confidence intervals offer improved accuracy over naive parallelism. To the best of our knowledge, no previous coupling paper as of this writing has compared coupling estimates or confidence intervals to those that arise from naively parallel chains.

Processor setup. We give both coupling and naively parallel approaches the same number of processors J. We ensure equal wall time across processors as we describe next; this setup represents a computing system where, e.g., the user pays for total wall time, in which case we ensure equal cost between approaches. For the coupling on the jth processor, we run until the chains meet and record the total time ξ^j. In the naively parallel case, then, we run a single chain on the jth processor for time ξ^j. In either case, each processor returns an estimate of H^*. We can aggregate these estimates with a sample mean or trimmed estimator. Let $H_{c,J}$ represent the coupled estimate after aggregation across J processors and $H_{u,J}$ represent the naive parallel (uncoupled) estimate after aggregation across J processors. To understand the variability of these estimates, we replicate them I times: $\{H_{c,J}^{(i)}\}_{i=1}^I$ and $\{H_{u,J}^{(i)}\}_{i=1}^I$. In

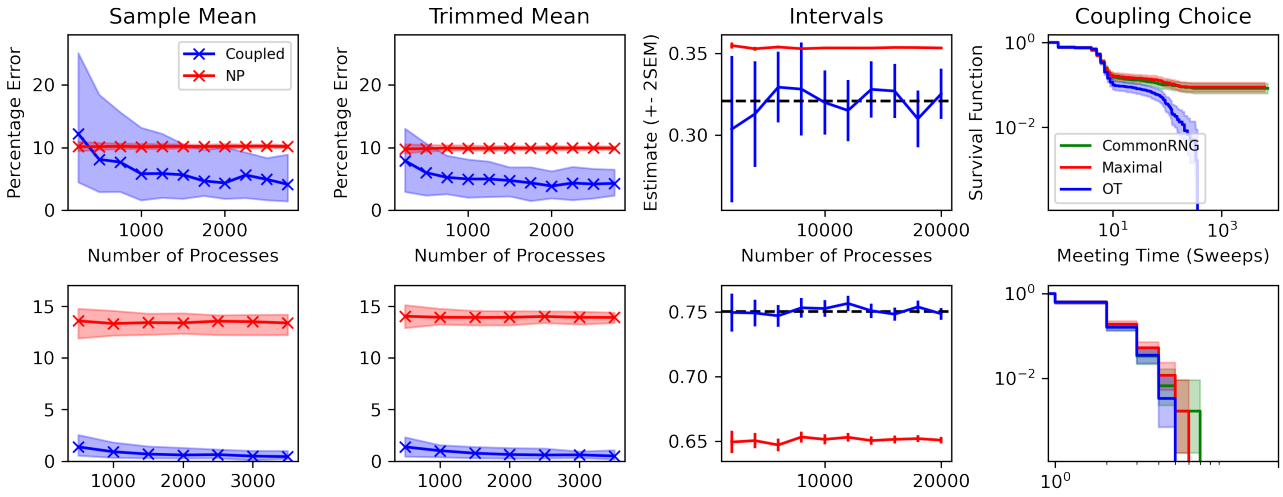


Figure 2: Top row and bottom row give results for GENE and K-REGULAR, respectively. The first two columns show that coupled chains provide better point estimates than naive parallelism. The third column shows that confidence intervals based on coupled chains are better than those from naive parallelism. The fourth column shows that OT coupling meets in less time than label-based couplings.

particular, we simulate running on 180,000 processors, so for each J, we let $I = 180{,}000/J$; see Appendix G.2 for details. For the ith replicate, we compute squared error $e_{c,i} := (H_{c,J}^{(i)} - H^*)^2$; similarly in the uncoupled case.

Better point estimates. The upper left panel of Figure 2 shows the behavior of LCP estimates for GENE. The horizontal axis gives the number of processes J. The vertical value of any solid line is found by taking the square root of the median (across I replicates) of the squared error and then dividing by the (positive) ground truth. Blue shows the performance of the aggregated standard-mean coupling estimate; red shows the naive parallel estimate. The blue regions show the 20% to 80% quantile range. We can see that, at higher numbers of processors, the coupling estimates consistently yield a lower percentage error than the naive parallel estimates for a shared wall time. The difference is even more pronounced for the trimmed estimates (first row, second column of Figure 2); here we see that, even at smaller numbers of processors, the coupling estimates consistently outperform the naive parallel estimates for a shared wall time. We see the same patterns for estimating CC(2,4) in K-REGULAR (second row, first two columns of Figure 2) and also for SYNTHETIC, SEED, and ABALONE in Figures 7a, 8a and 9a in Appendix H. We see similar patterns in the root mean squared error across replicates in Figure 1 (which pertains to GENE) and the left panel of Figures 7b, 8b, 9b and 11b for the remaining datasets.

Figure 3 illustrates that the problem with naive parallelism is the bias of the individual chains, whereas only

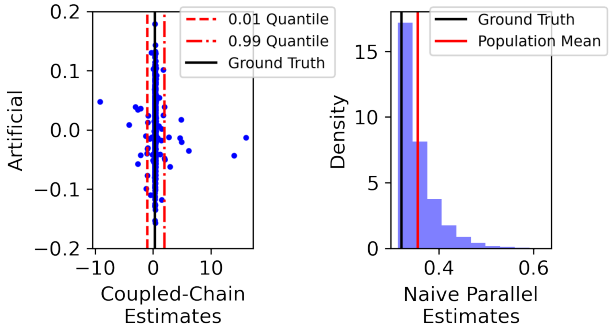


Figure 3: Coupled-chain estimates have large outliers. Meanwhile, naive parallelism estimates have substantial bias that does not go away with replication.

variance is eliminated by parallelism. In particular, the histogram on the right depicts the J estimates returned across each uncoupled chain at each processor j. We see that the population mean across these estimates is substantially different from the ground truth. This observation also clarifies why trimming does not benefit the naive parallel estimator: trimming can eliminate outliers but not systematic bias across processors.

By contrast, we plot the J coupling estimates returned across each processor j as horizontal coordinates of points in the left panel of Figure 3. Vertical coordinates are random noise to aid in visualization. By plotting the 1% and 99% quantiles of the J estimators, we can see that trimming will eliminate a few outliers. But the vast majority of estimates concentrate near the ground truth.

Better confidence intervals. The third column of Figure 2 shows that the confidence intervals returned by coupling are also substantially improved relative to naive parallelism. The setup here is slightly different from that of the first two columns. For the first two columns, we instantiated many replicates of individual users and thereby checked that coupling generally can be counted upon to beat naive parallelism. But, in practice, an actual user would run just a single replicate. Here, we evaluate the quality of a confidence interval that an actual user would construct. We use only the individual estimates s_j that make up *one* $H_{c,J}$, $s_j = H_{\ell:m}(X^j, Y^j)$ (or the equivalent for $H_{u,J}$), to form a point estimate of H^* and a notion of uncertainty.

In the third column of Figure 2, each solid line shows the sample-average estimate aggregated across J processors: $(1/J) \times \sum_{j=1}^{J} s_j$. The error bars show ± 2 standard errors of the mean (SEM), where one SEM equals $\sqrt{\mathrm{Var}(\{s_j\}_{j=1}^{J})/(J-1)}$. Since the individual coupling estimators (blue) from each processor j are unbiased, we expect the error bars to be calibrated, and indeed we see appropriate coverage of the ground truth (dashed black line). By contrast, we again see systematic bias in the naive parallel estimates – and very-overconfident intervals; indeed they are so small as to be largely invisible in the top row of the third column of Figure 2 – i.e., when estimating LCP in the GENE dataset. The ground truth is many standard errors away from the naive parallel estimates. We see the same patterns for estimating CC(2,4) for K-REGULAR (second row, third column of Figure 2). See the right panel of Figures 7b, 8b and 9b in Appendix H for similar behaviors in SYNTHETIC, SEED, and ABALONE.

5.3 Faster Meeting With OT Couplings

Next we show that meeting times with our OT coupling on partitions are faster than with label-based coupling using maximal [Jerrum, 1998] and common random number generator (common RNG) [Gibbs, 2004]. We did not directly add a comparison with label-based couplings to our plots in Section 5.2 since, in many cases, the label-based coupling chains fail to meet altogether even with a substantially larger time budget than Section 5.2 currently uses.

Instead, we now provide a direct comparison of meeting times in the fourth column of Figure 2. To generate each figure, we set a fixed amount of compute time budget: 10 minutes for the top row, and 2 minutes for the bottom row. Each time budget is roughly the amount of time taken to generate the ground truth (i.e., the long, single-chain runs) for each dataset. If during that time a coupling method makes the two chains meet, we record the meeting time τ; otherwise, the meeting time for that replica is right-censored, and we record the number of data sweeps up to that point. Using the censored data, we estimate the survival functions of the meeting times using the classic Kaplan–Maier procedure [Kaplan and Meier, 1958].

In the clustering examples (Figure 2 top row, fourth column and also the left panel of Figures 7c, 8c and 9c in Appendix H), the label-based couplings' survival functions $\Pr(\tau > t)$ do not go to zero for large times t, but instead they plateau around 0.1. In other words, the label-based coupling chains fail to meet on about 10% of attempts. Meanwhile, all replicas with our OT coupling successfully meet in the allotted time. Since so many label-based couplings fail to meet before the time taken to generate the ground truth, these label-based couplings perform worse than essentially standard MCMC. In addition to survival functions, we also plot the distance between coupled chains – which decreases the fastest for our OT couplings – in the right panel of Figures 6c, 7c, 8c, 9c and 11c in Appendix H. As discussed in Appendix E, we believe the improvement of our OT coupling over baselines arises from using a coupling function that incentivizes decreasing the distance between partitions rather than between labelings.

Separate from accurate estimation in little time, our comparison of survival functions in the bottom row, fourth column of Figure 2 and in Figure 19 from Appendix L is potentially of independent interest. While the bottom row of Figure 2 gives results for K-REGULAR, Figure 19 gives results on Erdős-Rényi random graphs. The tightest bounds for mixing time for Gibbs samplers on graph colorings to date [Chen et al., 2019] rely on couplings on labeled representations. Our result suggests better bounds may be attainable by considering convergence of partitions rather than labelings.

6 CONCLUSION

We demonstrated how to efficiently couple partition-valued Gibbs samplers using optimal transport – to take advantage of parallelism for improved estimation. Multiple directions show promise for future work. E.g., while we have used CPUs in our experiments here, we expect that GPU implementations will improve the applicability of our methodology. More extensive theory on the trimmed estimator could clarify its guarantees and best practical settings. Another direction is developing couplings for models with more complicated combinatorial structure – such as topic modeling Pritchard et al. [2000], Blei et al. [2003] or feature allocations [Griffiths and Ghahramani, 2011], in which data indices can belong to more than one latent group at a time.

ACKNOWLEDGMENTS

This work was supported by an NSF CAREER Award and an ONR Early Career Grant. BLT was also supported by NSF GRFP.

References

Peter J. Bickel. On Some Robust Estimates of Location. *The Annals of Mathematical Statistics*, 36(3):847 – 858, 1965.

Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag, Berlin, Heidelberg, 2006.

David M. Blei and Michael I. Jordan. Variational inference for Dirichlet process mixtures. *Bayesian Analysis*, 1(1):121 – 143, 2006.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent dirichlet allocation. *Journal of Machine Learning Resesearch*, 3(null):993–1022, mar 2003. ISSN 1532-4435.

Nicolas Bonneel, Michiel Van De Panne, Sylvain Paris, and Wolfgang Heidrich. Displacement interpolation using Lagrangian mass transport. In *Proceedings of the 2011 SIGGRAPH Asia Conference*, 2011.

Małgorzata Charytanowicz, Jerzy Niewczas, Piotr Kulczycki, Piotr A Kowalski, Szymon Łukasik, and Sławomir Żak. Complete gradient clustering algorithm for features analysis of X-ray images. In *Information Technologies in Biomedicine*, pages 15–24. Springer, 2010.

Sitan Chen, Michelle Delcourt, Ankur Moitra, Guillem Perarnau, and Luke Postle. Improved bounds for randomly sampling colorings via linear programming. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM, 2019.

Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 26, 2013.

Perry de Valpine, Daniel Turek, Christopher J. Paciorek, Clifford Anderson-Bergman, Duncan Temple Lang, and Rastislav Bodik. Programming With Models: Writing Statistical Algorithms for General Model Structures With NIMBLE. *Journal of Computational and Graphical Statistics*, 26(2):403–413, 2017.

Daryl DeFord, Moon Duchin, and Justin Solomon. Recombination: a family of Markov chains for redistricting. *Harvard Data Science Review*, 3 2021. https://hdsr.mitpress.mit.edu/pub/1ds8ptxu.

Dheeru Dua and Casey Graff. UCI machine learning repository, 2017. URL `http://archive.ics.uci.edu/ml`.

Michael D. Escobar and Mike West. Bayesian density estimation and inference using mixtures. *Journal of the American Statistical Association*, 90(430):577–588, 1995.

Rémi Flamary, Nicolas Courty, Alexandre Gramfort, Mokhtar Z. Alaya, Aurélie Boisbunon, Stanislas Chambon, Laetitia Chapel, Adrien Corenflos, Kilian Fatras, Nemo Fournier, Léo Gautheron, Nathalie T.H. Gayraud, Hicham Janati, Alain Rakotomamonjy, Ievgen Redko, Antoine Rolet, Antony Schutz, Vivien Seguy, Danica J. Sutherland, Romain Tavenard, Alexander Tong, and Titouan Vayer. POT: Python Optimal Transport. *Journal of Machine Learning Research*, 22(78):1–8, 2021.

S. Ghosal, J. K. Ghosh, and R. V. Ramamoorthi. Posterior consistency of Dirichlet mixtures in density estimation. *Annals of Statistics*, 27(1):143–158, 1999.

J. K. Ghosh and R. V. Ramamoorthi. *Bayesian Nonparametrics*. Springer Series in Statistics, 2003.

Alison L. Gibbs. Convergence in the Wasserstein metric for Markov chain Monte Carlo algorithms with applications to image restoration. *Stochastic Models*, 20(4):473–492, 2004.

Peter W. Glynn and Chang-han Rhee. Exact estimation for Markov chain equilibrium expectations. *Journal of Applied Probability*, 51(A):377–389, 2014.

Dilan Görür and Carl E. Rasmussen. Dirichlet process Gaussian mixture models: choice of the base distribution. *Journal of Computer Science and Technology*, 25(4):653–664, 2010.

Thomas L. Griffiths and Zoubin Ghahramani. The Indian Buffet Process: An Introduction and Review. *Journal of Machine Learning Research*, 12(32):1185–1224, 2011.

Jonathan Huggins, Mikolaj Kasprzak, Trevor Campbell, and Tamara Broderick. Validated variational inference via practical posterior error bounds. In *International Conference on Artificial Intelligence and Statistics*, pages 1792–1802. PMLR, 2020.

Pierre E. Jacob. Couplings and Monte Carlo. Course Lecture Notes, 2020.

Pierre E. Jacob, John O'Leary, and Yves F. Atchadé. Unbiased Markov chain Monte Carlo methods with couplings. *Journal of the Royal Statistical Society Series B*, 82(3):543–600, 2020.

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*, 13(1):158–182, 2004.

Ajay Jasra, Chris C. Holmes, and David A. Stephens. Markov chain Monte Carlo methods and the label switching problem in Bayesian mixture modeling. *Statistical Science*, pages 50–67, 2005.

Mark Jerrum. Mathematical foundations of the Markov chain Monte Carlo method. In *Probabilistic Methods for Algorithmic Discrete Mathematics*, pages 116–165. Springer, 1998.

E. L. Kaplan and Paul Meier. Nonparametric estimation from incomplete observations. *Journal of the American Statistical Association*, 53(282):457–481, 1958.

Damian J. Kelly and Garrett M. O'Neill. *The minimum cost flow problem and the network simplex solution method*. PhD thesis, Citeseer, 1991.

Junpeng Lao, Christopher Suter, Ian Langmore, Cyril Chimisov, Ashish Saxena, Pavel Sountsov, Dave Moore, Rif A. Saurous, Matthew D. Hoffman, and Joshua V. Dillon. tfp. mcmc: Modern Markov Chain Monte Carlo Tools Built For Modern Hardware. *arXiv preprint arXiv:2002.01184*, 2020.

David A. Levin and Yuval Peres. *Markov chains and mixing times*, volume 107. American Mathematical Society, 2017.

Antonio Lijoi, Igor Prünster, and Stephen G. Walker. On consistency of nonparametric normal mixtures for Bayesian density estimation. *Journal of the American Statistical Association*, 100(472):1292–1296, 2005.

Torgny Lindvall. *Lectures on the coupling method*. Courier Corporation, 2002.

Silvia Liverani, David I. Hastie, Lamiae Azizi, Michail Papathomas, and Sylvia Richardson. PReMiuM: An R package for profile regression mixture models using Dirichlet processes. *Journal of Statistical Software*, 64(7):1, 2015.

Gábor Lugosi and Shahar Mendelson. Mean estimation and regression under heavy-tailed distributions: A survey. *Foundations of Computational Mathematics*, 19(5):1145–1190, 2019.

Steven N. MacEachern. Estimating normal means with a conjugate style Dirichlet process prior. *Communications in Statistics - Simulation and Computation*, 23(3):727–741, 1994.

Marina Meilă. Comparing clusterings—an information based distance. *Journal of Multivariate Analysis*, 98 (5):873–895, 2007.

Jeffrey W Miller and Matthew T Harrison. Mixture models with a prior on the number of components. *Journal of the American Statistical Association*, 113 (521):340–356, 2018.

B. G. Mirkin and L. B. Chernyi. Measurement of the distance between distinct partitions of a finite set of objects. *Automation and Remote Control*, 5:120–127, 1970.

Warwick Nash, T.L. Sellers, S.R. Talbot, A.J. Cawthorn, and W.B. Ford. The Population Biology of Abalone (Haliotis species) in Tasmania. I. Blacklip Abalone (H. rubra) from the North Coast and Islands of Bass Strait. *Sea Fisheries Division, Technical Report*, 48, 01 1994.

Radford M Neal. Circularly-coupled markov chain sampling. Technical report, University of Toronto, 1992.

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

James B. Orlin. A faster strongly polynomial minimum cost flow algorithm. *Operations Research*, 41(2):338–350, 1993.

Jim Pitman. *Combinatorial Stochastic Processes: Ecole d'Eté de Probabilités de Saint-Flour XXXII-2002*. Springer, 2006.

Sandhya Prabhakaran, Elham Azizi, Ambrose Carr, and Dana Pe'er. Dirichlet process mixture model for correcting technical variation in single-cell gene expression data. In *International Conference on Machine Learning*, 2016.

Jonathan K Pritchard, Matthew Stephens, and Peter Donnelly. Inference of Population Structure Using Multilocus Genotype Data. *Genetics*, 155(2):945–959, 06 2000.

James Gary Propp and David Bruce Wilson. Exact sampling with coupled Markov chains and applications to statistical mechanics. *Random Structures & Algorithms*, 9(1-2):223–252, 1996.

Maxim Rabinovich, Elaine Angelino, and Michael I Jordan. Variational consensus monte carlo. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015.

William M. Rand. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association*, 66(336):846–850, 1971.

Albert Reuther, Jeremy Kepner, Chansup Byun, Siddharth Samsi, William Arcand, David Bestor, Bill Bergeron, Vijay Gadepally, Michael Houle, Matthew Hubbell, Michael Jones, Anna Klein, Lauren Milechin, Julia Mullen, Andrew Prout, Antonio Rosa, Charles Yee, and Peter Michaleas. Interactive supercomputing on 40,000 cores for machine learning and data analysis. In *2018 IEEE High Performance extreme Computing Conference (HPEC)*, pages 1–6. IEEE, 2018.

Steven L Scott, Alexander W Blocker, Fernando V Bonassi, Hugh A Chipman, Edward I George, and Robert E McCulloch. Bayes and big data: The consensus Monte Carlo algorithm. *International Journal of Management Science and Engineering Management*, 11(2):78–88, 2016.

Sanvesh Srivastava, Cheng Li, and David B. Dunson. Scalable Bayes via barycenter in Wasserstein space. *The Journal of Machine Learning Research*, 19(1): 312–346, 2018.

Stephen M. Stigler. The Asymptotic Distribution of the Trimmed Mean. *The Annals of Statistics*, 1(3): 472 – 477, 1973.

Stephen M. Stigler. The 1988 Neyman Memorial Lecture: A Galtonian Perspective on Shrinkage Estimators. *Statistical Science*, 5(1):147–155, 1990.

Robert H. Swendsen and Jian-Sheng Wang. Replica Monte Carlo simulation of spin-glasses. *Physical Review Letters*, 57(21):2607, 1986.

Andrea Tancredi, Rebecca Steorts, and Brunero Liseo. A Unified Framework for De-Duplication and Population Size Estimation (with Discussion). *Bayesian Analysis*, 15(2):633 – 682, 2020.

John W. Tukey and Donald H. McLaughlin. Less vulnerable confidence and significance procedures for location based on a single sample: Trimming/winsorization 1. *Sankhyā: The Indian Journal of Statistics, Series A (1961-2002)*, 25(3):331–352, 1963.

Kai Xu, Tor Erlend Fjelde, Charles Sutton, and Hong Ge. Couplings for multinomial Hamiltonian Monte Carlo. In *International Conference on Artificial Intelligence and Statistics*, 2021.

Amit Zeisel, Ana B. Muñoz-Manchado, Simone Codeluppi, Peter Lönnerberg, Gioele La Manno, Anna Juréus, Sueli Marques, Hermany Munguba, Liqun He, and Christer Betsholtz. Cell types in the mouse cortex and hippocampus revealed by single-cell RNA-seq. *Science*, 347(6226):1138–1142, 2015.

A RELATED WORK

Couplings of Markov chains have a long history in MCMC. Historically, they have primarily been a theoretical tool for analyzing convergence of Markov chains (see e.g. Lindvall [2002] and references therein). Some works prior to Jacob et al. [2020] used coupled Markov chains for computation, but do not provide guarantees of consistency in the limit of many processes or are not generally applicable to Markov chains over partitions. E.g., Propp and Wilson [1996] and follow-up works generate exact, i.i.d. samples but require a partial ordering of the state space that is almost surely preserved by applications of an iterated random function representation of the Markov transition kernel [Jacob, 2020, Chapter 4.4]. It is unclear what such a partial ordering looks like for the space of partitions. Neal [1992] proposes estimates obtained using circularly coupled chains that can be computed in parallel and aggregated, but these estimates are not unbiased and so aggregated estimates are not asymptotically exact. Parallel tempering methods [Swendsen and Wang, 1986] also utilize coupled chains to improve MCMC estimates but, like naive parallelism, provide guarantees asymptotic only in the number of transitions, not in the number of processes.

Outside of couplings, other lines of work have sought to utilize parallelism to obtain improved MCMC estimates in limited time. To our best knowledge, that work has focused on challenges introduced by large datasets and has subsequently focused on distributing datasets across processors. For example, Rabinovich et al. [2015], Scott et al. [2016], Srivastava et al. [2018] explore methods running multiple chains in parallel on small subsets of a large dataset, and Lao et al. [2020] proposes using data parallelism on GPUs to accelerate likelihood computations. However, these methods offer little help in the current setting as the partition is the quantity of interest in our case; even if distributions over partitions of subsets are found at each processor, these distributions are not trivial to combine across processors. Also, the operations that avail themselves to GPU acceleration (such as matrix multiplications) are not immediately present in Markov chains on partitions.

B FUNCTIONS OF INTEREST

We express functions of interest, h, in partition notation. Suppose there are N observations, and the partition is $\Pi = \{A^1, A^2, \ldots, A^K\}$. To compute largest component proportion (LCP), we first rank the clusters by decreasing size, $|A^{(1)}| \geq |A^{(2)}| \geq \ldots \geq |A^{(K)}|$, and report the proportion of data in the largest cluster: $|A^{(1)}|/N$. If we are interested in the co-clustering probability of data points indexed by j_1 and j_2, then we let h be the co-clustering indicator. Namely, if j_1 and j_2 belong to the same element of Π (i.e. there exists some $A \in \Pi$ such that $j_1, j_2 \in A$), then $h(\Pi)$ equals 1; otherwise, it equals 0.

In addition to these summary statistics of the partition, we can also estimate cluster-specific parameters, like cluster centers. For the Gaussian DPMM from Section 2.1, suppose that we care about the mean of clusters that contain a particular data point, say data point 1. This expectation is $\mathbb{E}(\mu_A \text{ s.t. } 1 \in A)$. This is equivalent to $\mathbb{E}[\theta_i \,|\, x]$ in the notation of MacEachern [1994]. In Section 2.1, we use μ_A to denote the cluster center for all elements $i \in A$, while MacEachern [1994] uses individual θ_i's to denote cluster centers for individual data points, with the possibility that $\theta_i = \theta_j$ if data points i and j belong in the same partition element. We can rewrite the expectation as $\mathbb{E}(\mathbb{E}[\mu_A \text{ s.t. } 1 \in A \,|\, \Pi])$, using the law of total expectation. $\mathbb{E}[\mu_A \text{ s.t. } 1 \in A \,|\, \Pi]$ is the posterior mean of the cluster that contains data point 1, which is a function only of the partition Π.

C UNBIASEDNESS THEOREM

Lemma 1 (Transition kernel is aperiodic and irreducible for Gaussian DPMM) *Denote by* $\mathbf{1}$ *the partition of* $[N]$ *where all elements belong to one cluster. For Gaussian DPMM, the transition kernel from Algorithm 1 satisfies*

- *For any* $X \in \mathcal{P}_N$, $T(X, X) > 0$.
- *For any* $X \in \mathcal{P}_N$, $T(X, \mathbf{1}) > 0$.
- *For any* $X \in \mathcal{P}_N$, $T(\mathbf{1}, X) > 0$.

Proof [Proof of Lemma 1] For any starting $X \in \mathcal{P}_N$, we observe that there is positive probability to stay at the state after the $T(X, \cdot)$ transition i.e. $T(X, X) > 0$. In Gaussian DPMM, because the support of the Gaussian

distribution is the whole Euclidean space (see also Equation (14)), when the nth data point is left out (resulting in the conditional $p_{\Pi|\Pi(-n)}(\cdot \mid X_{-n})$), there is positive probability that nth is re-inserted into the same partition element of X i.e. $p_{\Pi|\Pi(-n)}(X \mid X_{-n}) > 0$. Since $T(X, \cdot)$ is the composition of these N leave–outs and re-inserts, the probability of staying at X is the product of the probabilities for each $p_{\Pi|\Pi(-n)}(\cdot \mid X_{-n}))$, which is overall a positive number.

One series of updates that transform X into $\mathbf{1}$ in one sweep is to a) assign 1 to its own cluster and b) assign $2, 3, \ldots, N$ to the same cluster as 1. This series of update also has positive probability in Gaussian DPMM.

On transforming $\mathbf{1}$ into X, for each component A in X, let $c(A)$ be the smallest element in the component. For instance, if $X = \{\{1, 2\}, \{3, 4\}\}$ then $c(\{1, 2\}) = 1, c(\{3, 4\}) = 3$. We sort the components A by their $c(A)$, to get a list $c_1 < c_2 < \ldots < c_{|X|}$. For each $1 \leq n \leq N$, let $l(n) = c(A)$ for the component A that contains n. In the previous example, we have $c_1 = 1$ and $c_2 = 3$, while $l(1) = 1, l(2) = 1, l(3) = 3, l(4) = 3$. One series of updates that transform $\mathbf{1}$ into X is

- Initialize $j = 1$.
- for $1 \leq n \leq N$, if $n = c_j$, then make a new cluster with n and increment $j = j + 1$. Else, assign n to the cluster that currently contains $l(n)$.

This series of update also has positive probability in Gaussian DPMM. ■

Proof [Proof of Theorem 1]

Because of Jacob et al. [2020, Proposition 3], it suffices to check Jacob et al. [2020, Assumptions 1–3].

Checking Assumption 1. Because the sample space $\mathcal{P}_N$ is finite, $\max_{\pi \in \mathcal{P}_N} h(\pi)$ is finite. This means the expectation of any moment of h under the Markov chain is also bounded. We show that $\mathbb{E}[h(X^t)] \xrightarrow{t \to \infty} H^*$ by standard ergodicity arguments.[4]

- Aperiodic. From Lemma 1, we know $T(X, X) > 0$ for any X. This means the Markov chain is aperiodic [Levin and Peres, 2017, Section 1.3].
- Irreducible. From Lemma 1, for any X, Y, we know that $T(X, \mathbf{1}) > 0$ and $T(\mathbf{1}, Y) > 0$, meaning that $T^2(X, Y) > 0$. This means the Markov chain is irreducible.
- Invariant w.r.t. p_Π. The transition kernel $T(X, \cdot)$ from Algorithm 1 leaves the target p_Π invariant because each leave–out conditional $p_{\Pi|\Pi(-n)}$ leaves the target p_Π invariant. If $X \sim p_\Pi$, then $X_{-n} \sim p_{\Pi_{-n}}$. Hence, if $\widetilde{X} \mid X \sim p_{\Pi|\Pi(-n)}(\cdot \mid X_{-n})$ then by integrating out X, we have $\widetilde{X} \sim p_\Pi$.

By Levin and Peres [2017, Theorem 4.9], there exists a constant $\alpha \in (0, 1)$ and $C > 0$ such that

$$\max_{\pi \in \mathcal{P}_N} \|T^t(\pi, \cdot) - p_\Pi\|_{\mathrm{TV}} \leq C\alpha^t.$$

Since the sample space is finite, the total variation bound implies that for any π, expectations under $T^t(\pi, \cdot)$ are close to expectations under p_Π,

$$\max_{\pi \in \mathcal{P}_N} |\mathbb{E}_{X^t \mid X^0 = \pi} h(X^t) - H^*| \leq (\max_{\pi \in \mathcal{P}_N} h(\pi)) C\alpha^t.$$

Taking expectations over the initial condition $X^0 = \pi$,

$$|\mathbb{E}_{X^t} h(X^t) - H^*| = |\mathbb{E}_{X^0}[\mathbb{E}_{X^t \mid X^0 = \pi} h(X^t) - H^*]| \leq \mathbb{E}_{X^0} |\mathbb{E}_{X^t \mid X^0 = \pi} h(X^t) - H^*]| \leq (\max_{\pi \in \mathcal{P}_N} h(\pi)) C\alpha^t.$$

Since the right hand side goes to zero as $t \to \infty$, we have shown that $\mathbb{E}[h(X^t)] \xrightarrow{t \to \infty} H^*$.

[4] MacEachern [1994, Theorem 1] states a geometric ergodicity theorem for the Gibbs sampler like Algorithm 1 but does not provide verification of the aperiodicity, irreducibility or stationarity.

Checking Assumption 2. To show that the meeting time is geometric, we show that there exists $\bar{\epsilon}$ such that for any X and Y, under one coupled sweep from Algorithm 2 ($(\widetilde{X}, \widetilde{Y}) \sim \widehat{T}(\cdot, (X, Y))$),

$$\mathbb{P}(\widetilde{X} = \widetilde{Y} = \mathbf{1} \,|\, X, Y) \geq \bar{\epsilon}. \tag{8}$$

If this were true, we have that $\mathbb{P}(\widetilde{X} = \widetilde{Y} \,|\, X, Y) \geq \bar{\epsilon}$, and

$$\mathbb{P}(\tau > t) = \mathbb{P}\left(\cap_{i=0}^{t} X^{i+1} \neq Y^i\right) = \mathbb{P}(X^1 \neq Y^0) \prod_{i=1}^{t} \mathbb{P}(X^{i+1} \neq Y^i \,|\, X^i \neq Y^{i-1}),$$

where we have used the Markov property to remove conditioning beyond $X^i \neq Y^{i-1}$. Since $\min_{X,Y} \mathbb{P}(\widetilde{X} = \widetilde{Y} \,|\, X, Y) \geq \bar{\epsilon}$, $\mathbb{P}(X^{i+1} \neq Y^i \,|\, X^i \neq Y^{i-1}) \leq 1 - \bar{\epsilon}$, meaning $\mathbb{P}(\tau > t) \leq (1 - \bar{\epsilon})^t$.

To see why Equation (8) is true, because of Lemma 1, there exists a series of intermediate partitions $x^1, x^2, \ldots, x^{N-1}$ ($x^0 = X, x^N = \mathbf{1}$) such that for $1 \leq n \leq N$, $p_{\Pi|\Pi(-n)}(x^n \,|\, x_n^{n-1}) > 0$. Likewise, there exists a series $y^1, y^2, \ldots, y^{N-1}$ for Y. Because the coupling function ψ satisfies $u^{ij} > \epsilon$, for any n, there is at least probability ϵ of transitioning to (x^n, y^n) from (x^{n-1}, y^{n-1}). Overall, there is probability at least ϵ^N of transitioning from (X, Y) to $(\mathbf{1}, \mathbf{1})$. Since the choice of X, Y has been arbitrary, we have proven Equation (8) with $\bar{\epsilon} = \epsilon^N$.

Checking Assumption 3. By design, the chains remain faithful after coupling. ■

D TIME COMPLEXITY

Proposition 1 *Given the atom sizes $a_k, b_{k'}$ and atom locations $\pi^k, \nu^{k'}$ in the sense of Definition 2, we can compute the coupling matrix $\mu^{k,k'}$ for OT coupling function in $O(\widetilde{K}^3 \log \widetilde{K})$ time.*

Proof [Proof of Proposition 1] To find $\mu^{k,k'}$, we need to solve the optimization problem that is Equation (4). However, given just the marginal distributions ($a_k, b_{k'}$ and $\pi^k, \nu^{k'}$), we do not have enough "data" in the optimization problem, since the pairwise distances $\mathrm{d}(\pi^k, \nu^{k'})$ for $k \in [K], k' \in [K']$, which define the objective function, are missing. We observe that it is not necessary to compute $\mathrm{d}(\pi^k, \nu^{k'})$; it suffices to compute $\mathrm{d}(\pi^k, \nu^{k'}) - c$ for some constant c in the sense that the solution to the optimization problem in Equation (4) is unchanged when we add a constant value to every distance. In particular, because for any coupling γ, $\sum_{k=1}^{K} \sum_{k'=1}^{K'} u^{k,k'} = 1$,

$$\gamma^* := \underset{\text{couplings } \gamma}{\arg\min} \sum_{k=1}^{K} \sum_{k'=1}^{K'} u^{k,k'} \mathrm{d}(\pi^k, \nu^{k'}) = \underset{\text{couplings } \gamma}{\arg\min} \sum_{k=1}^{K} \sum_{k'=1}^{K'} u^{k,k'} [\mathrm{d}(\pi^k, \nu^{k'}) - c]. \tag{9}$$

We now show that if we set $c = \mathrm{d}(\pi(-n), \nu(-n))$, then we can compute all $O(\widetilde{K}^2)$ values of $\mathrm{d}(\pi^k, \nu^{k'}) - c$ in $O(\widetilde{K}^2)$ time. First, if we use A_n^k and $B_n^{k'}$ to denote the elements of π^k and $\nu^{k'}$ respectively, containing data-point n, then for any n we may write

$$\begin{aligned} d(\pi^k, \nu^{k'}) = d(\pi(-n), \nu(-n)) + \left[|A_n^k|^2 - (|A_n^k| - 1)^2\right] + \left[|B_n^{k'}|^2 - (|B_n^{k'}| - 1)^2\right] + \\ - 2\left[|A_n^k \cap B_n^{k'}|^2 - (|A_n^k \cap B_n^{k'}|^2 - 1)^2\right]. \end{aligned} \tag{10}$$

Simplifying some terms, we can also write

$$\begin{aligned} d(\pi^k, \nu^{k'}) &= d(\pi(-n), \nu(-n)) + \left[2|A_n^k| - 1\right] + \left[2|B_n^{k'}| - 1\right] - 2\left[2|A_n^k \cap B_n^{k'}| - 1\right] \\ &= d(\pi(-n), \nu(-n)) + 2\left[|A_n^k| + |B_n^{k'}| - 2|A_n^k \cap B_n^{k'}|\right], \end{aligned}$$

which means

$$d(\pi^k, \nu^{k'}) - d(\pi(-n), \nu(-n)) = 2\left[|A_n^k| + |B_n^{k'}| - 2|A_n^k \cap B_n^{k'}|\right].$$

At first it may seem that this still does not solve the problem, as directly computing the size of the set intersections is $O(N)$ (if cluster sizes scale as $O(N)$). However, Equation (9) is just our final stepping stone. If we additionally keep track of sizes of intersections at every step, updating them as we adapt the partitions, it will take only constant time for each update. As such, we are able to form the $K \times K'$ matrix of $\mathrm{d}(\pi^k, \nu^{k'}) - c$ in $O(\widetilde{K}^2)$ time.

With the array of $\mathrm{d}(\pi^k, \nu^{k'}) - \mathrm{d}(\pi(-n), \nu(-n))$, we now have enough "data" for the optimization problem that is the optimal transport. Regardless of N, the optimization itself may be computed in $O(\widetilde{K}^3 \log \widetilde{K})$ time with Orlin's algorithm [Orlin, 1993]. ■

The next proposition provides estimates of the time taken to construct the Gibbs conditionals ($\beta(N, K)$) for Gaussian DPMM.

Proposition 2 (Gibbs conditional runtime with dense Σ_0, Σ_1) *Suppose the covariance matrices Σ_0 and Σ_1 are dense i.e. the number of non-zero entries is $\Theta(D^2)$. The standard implementation takes time $\beta(N, K) = O(ND + KD^3)$. By spending $O(D^3)$ time precomputing at beginning of sampling, and using additional data structures, the time can be reduced to $\beta(N, K) = O(KD^2 + D^3)$.*

Proof [Proof of Proposition 2] We first mention the well-known posterior formula of a Gaussian model with known covariances [Bishop, 2006, Chapter 2.3]. Namely, if $\mu \sim \mathcal{N}(\mu_0, \Sigma_0)$ and $W_1, W_2, \ldots W_M \,|\, \mu \overset{\text{indep}}{\sim} \mathcal{N}(\mu, \Sigma_1)$ then $\mu \,|\, W_1, \ldots, W_M$ is a Gaussian with covariance Σ_c and mean μ_c satisfying

$$\begin{aligned} \Sigma_c &= (\Sigma_0^{-1} + M\Sigma_1^{-1})^{-1} \\ \mu_c &= \Sigma_c \left(\Sigma_0^{-1}\mu_0 + \Sigma_1^{-1} \left[\sum_{m=1}^{M} W_m \right] \right). \end{aligned} \tag{11}$$

Suppose $|\Pi| = K$. Based on the expressions for the Gibbs conditional in Equation (14), the computational work involved for a held-out observation W_n can be broken down into three steps

1. Evaluating the prior likelihood $\mathcal{N}(W_n \,|\, \mu_0, \Sigma_0 + \Sigma_1)$.

2. For each cluster $c \in \Pi(-n)$, compute μ_c, Σ_c, $(\Sigma_c + \Sigma_1)^{-1}$ and the determinant of $(\Sigma_c + \Sigma_1)^{-1}$.

3. For each cluster $c \in \Pi(-n)$, evaluate the likelihood $\mathcal{N}(W_n \,|\, \mu_c, \Sigma_c + \Sigma_1)$.

Standard implementation. The time to evaluate the prior $\mathcal{N}(W_n \,|\, \mu_0, \Sigma_0 + \Sigma_1)$ is $O(D^3)$, as we need to compute the precision matrix $(\Sigma_0 + \Sigma_1)^{-1}$ and its determinant. With time $O(KD^3)$, we can compute the various cluster-specific covariances, precisions and determinants (where D^3 is the cost for each cluster). To compute the posterior means μ_c, we need to compute the sums $\sum_j W_j$ for all clusters, which takes $O(ND)$, as we need to iterate over all D coordinates of all N observations. The time to evaluate $\mathcal{N}(W_n \,|\, \mu_c, \Sigma_c + \Sigma_1)$ across clusters is $O(KD^2)$. Overall this leads to $O(ND + KD^3)$ runtime.

Optimized implementation. By precomputing $(\Sigma_0 + \Sigma_1)^{-1}$ (and its determinant) once at the beginning of sampling for the cost of $O(D^3)$, we can solve Step 1 in time $O(D^2)$, since that is the time to compute the quadratic form involved in the Gaussian likelihood. Once we have the mean and precisions from Step 2, the time to complete Step 3 is $O(KD^2)$: for each cluster, it takes time $O(D^2)$ to evaluate the likelihood, and there are K clusters. It remains to show how much time it takes to solve Step 2. We note that quantities like $\Sigma_0^{-1}\mu_0$ and Σ_1^{-1} can also be computed once in $O(D^3)$ time at start up.

Regarding the covariance Σ_c and the precisions $(\Sigma_c + \Sigma_1)^{-1}$, at all points during sampling, the posterior covariance Σ_c only depends on the number of data points in the cluster (Equation (11)), and leaving out data point n only changes the number of points in exactly one cluster. Hence, if we maintain Σ_c, $(\Sigma_c + \Sigma_1)^{-1}$ (and their determinants) for all clusters $c \in \Pi$, when a data point is left out, we only need to update one such Σ_c and $(\Sigma_c + \Sigma_1)^{-1}$. Namely, suppose that $\Pi = \{A^1, A^2, \ldots, A^K\}$. We maintain the precisions are $(\Sigma(A^1) + \Sigma_1)^{-1}, (\Sigma(A^2) + \Sigma_1)^{-1}, \ldots, (\Sigma(A^k) + \Sigma_1)^{-1}$. Let A^j be the cluster element that originally contained n.

When we leave out data point n to form $\Pi(-n)$, the only precision that needs to be changed is $(\Sigma(A^j) + \Sigma_1)^{-1}$. Let the new cluster be $\widetilde{A^j}$: the time to compute $\Sigma(\widetilde{A^j})$, $(\Sigma(\widetilde{A^j}) + \Sigma_1)^{-1}$, and its determinant is $O(D^3)$.

Regarding the means μ_c, the use of data structures similar to the covariances/precisions removes the apparent need to do $O(ND)$ computations. If we keep track of $\sum_{i \in c} W_i$ for each cluster c, then when data point n is left out, we only need to update $\sum_{i \in c} W_i$ for the cluster c that originally contained n, which only takes $O(D)$. With the $\sum_j W_j$ in place, to evaluate each of K means μ_c takes $O(D^2)$; hence the time to compute the means is $O(KD^2)$. Overall, the time spent in Step 2 is $O(KD^2+D^3)$, leading to an overall $O(KD^2+D^3)$ runtime. ■

The standard implementation is used, for instance, in de Valpine et al. [2017] (see the CRP_conjugate_dmnorm_dmnorm() function from NIMBLE's source code). Miller and Harrison [2018] uses the standard implementation in the univariate case (see the Normal.jl function).

Corollary 2 (Gibbs conditional runtime with diagonal Σ_0, Σ_1) *Suppose the covariances Σ_0 and Σ_1 are diagonal matrices i.e. there are only $\Theta(D)$ non-zero entries. Then a standard implementation takes time $\beta(N, K) = O(ND)$. Using additional data structures, the time can be reduced to $\beta(N, K) = O(KD)$.*

Proof [Proof of Corollary 2] When the covariance matrices are diagonal, we do not incur the cubic costs of inverting $D \times D$ matrices. The breakdown of computational work is similar to the proof of Proposition 2.

Standard implementation. The covariances and precision matrices each take only time $O(D)$ to compute: as there are K of them, the time taken is $O(KD)$. To compute the posterior means μ_c, we iterate through all coordinates of all observations in forming the sums $\sum_j W_j$, leading to $O(ND)$ runtime. Time to evaluate the Gaussian likelihoods are just $O(D)$ because of the diagonal precision matrices. Overall the runtime is $O(ND)$.

Optimized implementation. By avoiding the recomputation of $\sum_j W_j$ from scratch, we reduce the time taken to compute the posterior means to $O(KD)$. Overall the runtime is $O(KD)$. ■

E LABEL-SWITCHING

E.1 Example 1

Suppose there are 4 data points, indexed by 1,2,3,4. The labeling of the X chain is $z_1 = [1, 2, 2, 2]$, meaning that the partition is $\{\{1\}, \{2, 3, 4\}\}$. The labeling of the Y chain is $z_2 = [2, 1, 1, 2]$, meaning that the partition is $\{\{1, 4\}, \{2, 3\}\}$. The Gibbs sampler temporarily removes the data point 4. For both chains, the remaining data points is partitioned into $\{\{1\}, \{2, 3\}\}$. We denote $\pi^1 = \{\{1, 4\}, \{2, 3\}\}$, $\pi^2 = \{\{1\}, \{2, 3, 4\}\}$, $\pi^3 = \{\{1\}, \{2, 3\}, \{4\}\}$: in the first two partitions, the data point is assigned to an existing cluster while in the last partition ,the data point is in its own cluster. There exists three positive numbers a^1, a^2, a^3, summing to one, such that

$$p_{\Pi|\Pi(-4)}(\cdot \mid X(-4)) = p_{\Pi|\Pi(-4)}(\cdot \mid Y(-4)) = \sum_{k=1}^{3} a^k \delta_{\pi^k}(\cdot).$$

Since the two distributions on partitions are the same, couplings based on partitions like ψ_η^{OT} will make the chains meet with probability 1 in the next step. However, this is not true under labeling–based couplings like maximal or common RNG. In this example, the same partition is represented with different labels under either chains. The X chain represents π^1, π^2, π^3 with the labels 1, 2, 3, respectively. Meanwhile, the Y chain represents π^1, π^2, π^3 with the labels 2, 1, 3, respectively. Let z_X be the label assignment of the data point in question (recall that we have been leaving out 4) under the X chain. Similarly we define z_Y. Maximal coupling maximizes the

probability that $z_X = z_Y$. However, the coupling that results in the two chains X and Y meeting is the following

$$\Pr(z_X = u, z_Y = v) = \begin{cases} a^3 & \text{if } u = v = 3 \\ a^1 & \text{if } u = 1, v = 2 \\ a^2 & \text{if } u = 2, v = 1 \\ 0 & \text{otherwise.} \end{cases}$$

In general, $a^1 \neq a^2$, meaning that the maximal coupling is different from this coupling that causes the two chains to achieve the same partition after updating the assignment of 4. A similar phenomenon is true for common RNG coupling.

E.2 Example 2

For the situation in Appendix E.1, the discussion of Ju et al. from Tancredi et al. [2020] proposes a relabeling procedure to better align the clusters in the two partitions before constructing couplings. Indeed, if z_2 were relabeled $[1, 2, 2, 1]$ (the label of each cluster is the smallest data index in that cluster), then upon the removal of data point 4, both the label-based and partition-based couplings would agree. However, such a relabeling fix still suffer from label-switching problem in general, since the smallest data index does not convey much information about the cluster. For concreteness, we demonstrate an example where the best coupling from minimizing label distances is different from the best coupling minimizing partition distances.

Suppose there are 6 data points, indexed from 1 through 6. The partition of the X chain is $\{\{1, 3, 4\}, \{2, 5, 6\}\}$. The partition of the Y chain is $\{\{1, 5, 6\}, \{2, 3, 4\}\}$. Using the labeling rule from above, the label vector for X is $z_X = [1, 2, 1, 1, 2, 2]$ while that for Y is $z_Y = [1, 2, 2, 2, 1, 1]$. The Gibbs sampler temporarily removes the data point 1. The three next possible states of the X chain are the partitions ν_1, ν_2, ν_3 where $\nu_1 = \{\{1, 3, 4\}, \{2, 5, 6\}\}$, $\nu_2 = \{\{3, 4\}, \{1, 2, 5, 6\}\}$ and $\nu_3 = \{\{3, 4\}, \{2, 5, 6\}, \{1\}\}$. The labelings of data points 2 through 6 for all three partitions are the same; the only different between the labeling vectors are the label of data point 1: for ν_1, $z_X(1) = 1$, for ν_2, $z_X(1) = 2$ and for ν_1, $z_X(1) = 3$. On the Y side, the three next possible states of the Y chain are the partitions μ_1, μ_2, μ_3 where $\mu_1 = \{\{1, 5, 6\}, \{2, 3, 4\}\}$, $\mu_2 = \{\{5, 6\}, \{1, 2, 3, 4\}\}$ and $\mu_3 = \{\{5, 6\}, \{2, 3, 4\}, \{1\}\}$. As for the labeling of 1 under Y, for μ_1, $z_Y(1) = 1$, for μ_2, $z_Y(1) = 2$ and for μ_3, $z_Y(1) = 3$. Suppose that the marginal assignment probabilities are the the following:

- $\Pr(X = \nu_1) = \Pr(X = \nu_2) = 0.45, \Pr(X = \nu_3) = 0.1$.
- $\Pr(Y = \mu_1) = \Pr(Y = \mu_2) = 0.45, \Pr(Y = \mu_3) = 0.1$.

Under label-based couplings, since $\Pr(z_X(1) = a) = \Pr(z_Y(1) = a)$ for $a \in [1, 2, 3]$, the coupling that minimizes the distance between the labels will pick $\Pr(z_X(1) = z_Y(1)) = 1$, which means the following for the induced partitions:

$$\Pr(X = \nu, Y = \mu) = \begin{cases} 0.45 & \text{if } \nu = \nu_1, \mu = \mu_1 \\ 0.45 & \text{if } \nu = \nu_2, \mu = \mu_2 \\ 0.1 & \text{if } \nu = \nu_3, \mu = \mu_3 \end{cases}. \tag{12}$$

Under the partition-based transport coupling, the distance between partitions (Equation (5)) is the following.

	μ_1	μ_2	μ_3
ν_1	16	10	12
ν_2	10	16	14
ν_3	12	14	8

Notice that the distances $d(\nu_1, \mu_1)$ and $d(\nu_2, \mu_2)$ are actually larger than $d(\nu_1, \mu_2)$ and $d(\nu_2, \mu_1)$: in other words, the label-based coupling from Equation (12) proposes a coupling with larger-than-minimal expected distance. In fact, solving the transport problem, we find that the coupling that minimizes the expected partition distance is actually

$$\Pr(X = \nu, Y = \mu) = \begin{cases} 0.45 & \text{if } \nu = \nu_1, \mu = \mu_2 \\ 0.45 & \text{if } \nu = \nu_2, \mu = \mu_1 \\ 0.1 & \text{if } \nu = \nu_3, \mu = \mu_3 \end{cases}. \tag{13}$$

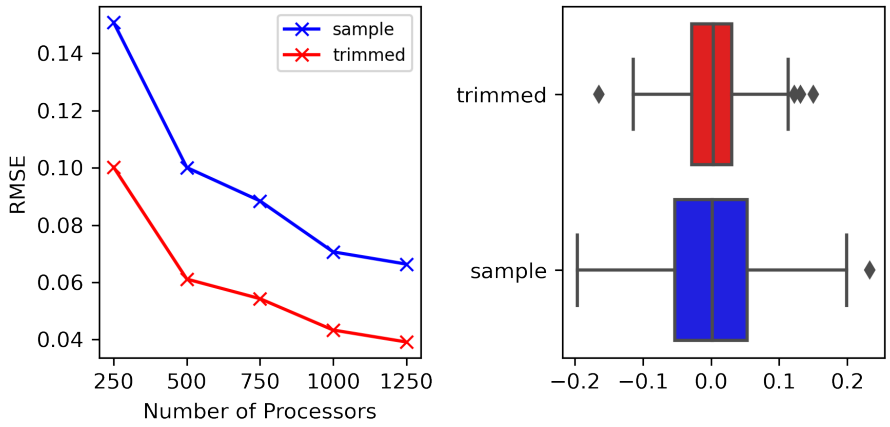


Figure 4: Trimmed mean has better RMSE than sample mean on Example 1. Left panel plots RMSE versus J. Right panel gives boxplots $J = 1000$.

F TRIMMING

We consider the motivating situation in Example 1. This is a case where trimming outliers before taking the average yields a more accurate estimator (in terms of mean squared error) than the regular sample mean. For reference, the RMSE of an estimator $\widehat{\mu}$ of a real-valued unknown quantity μ is

$$\sqrt{\mathbb{E}\|\widehat{\mu} - \mu\|^2}.$$

Example 1 (Mixture distribution with large outliers) *For $\mu > 0$, $p < 1$, consider the mixture distribution $(0.5 - p/2)\mathcal{N}(-\mu, 1) + p\mathcal{N}(0, 1) + (0.5 - p/2)\mathcal{N}(\mu, 1)$. The mean is 0. The variance is $1 + (1 - p)\mu^2$. Therefore, the RMSE of the sample mean computed using J iid draws is $\sqrt{1 + (1 - p)\mu^2}/\sqrt{J}$.*

In Example 1, increasing μ, which corresponds to larger outlier magnitude, increases the RMSE.

In trimmed means (Section 3.4), the quantity α determines how much trimming is done. Intuitively, for Example 1, if we trim about $0.5 - p/2$ of the top and bottom samples from the mixture distribution in Example 1, what remain are roughly samples from $\mathcal{N}(0, 1)$. The mean of these samples should have variance only $1/J$, resulting in an RMSE which does not suffer from large μ.

In Figure 4, we illustrate the improvement of trimmed mean over sample mean for problems like Example 1. We set $p = 0.9$, $\mu = 7$, and $\alpha = 1.2(0.5 - p/2)$. Similar to Figure 1, RMSE is estimated by adding another level of simulation to capture the variability across aggregates. The left panel shows that RMSE of trimmed mean is smaller than that of sample mean. The right panel explains why that is the case. Here, we box plot the trimmed mean and sample mean, where the randomness is from the iid Monte Carlo draws from the target mixture for $J = 1000$. The variance of trimmed mean is smaller than that of sample mean, which matches the motivation for trimming.

For other situations where there exist better estimators than the sample mean, we refer to the literature on Stein's paradox [Stigler, 1990].

G ADDITIONAL EXPERIMENTAL DETAILS

G.1 Target Distributions And Gibbs Conditionals

DPMM. Denote $\mathcal{N}(x \,|\, \mu, \Sigma)$ to be the Gaussian density at x for a Gaussian distribution with mean μ and covariance Σ. For the Gaussian DPMM from Section 2.1, the Gibbs conditional have the form

$$\Pr(z_n = c \,|\, \Pi(-n), W_{1:N}) = \begin{cases} \beta \frac{\alpha}{N-1+\alpha} \mathcal{N}(W_n \,|\, \mu_0, \Sigma_0 + \Sigma_1) & \text{if } c \text{ is new cluster} \\ \beta \frac{\text{size of cluster c}}{N-1+\alpha} \mathcal{N}(W_n \,|\, \mu_c, \Sigma_c + \Sigma_1) & \text{if } c \text{ is an existing cluster,} \end{cases} \tag{14}$$

where β is a normalization constant so that $\sum_c \Pr(z_n = c \,|\, \Pi(-n), W_{1:N}) = 1$, c is an index into the clusters that comprise $\Pi(-n)$ (or a new cluster), μ_c and Σ_c are the posterior parameters of the cluster indexed by c. See Neal [2000] for derivations.

Graph coloring. Let G be an undirected graph with vertices $V = [N]$ and edges $E \subset V \otimes V$, and let $Q = [q]$ be set of q colors. A graph coloring is an assignment of a color in Q to each vertex satisfying that the endpoints of each edge have different colors. We here demonstrate an application of our method to a Gibbs sampler which explores the uniform distribution over valid $q-$colorings of G, i.e. the distribution which places equal mass on ever proper coloring of G.

To employ Algorithm 2, for this problem we need only to characterise the p.m.f. on partitions of the vertices implied by the uniform distribution on its colorings. A partition corresponds to a proper coloring only if no two adjacent vertices are in the element of the partition. As such, we can write

$$p_{\Pi_N}(\pi) \propto \mathbb{1}\{|\pi| \leq q \text{ and } A(\pi)_{i,j} = 1 \to (i,j) \notin E,\ \forall i \neq j\} \binom{q}{|\pi|} |\pi|!,$$

where the indicator term checks that π can correspond to a proper coloring and the second term accounts for the number of unique colorings which induce the partition π. In particular it is the product of the number of ways to choose $|\pi|$ unique colors from Q ($\binom{q}{|\pi|} := \frac{q!}{|\pi|!(q-|\pi|)!}$) and the number of ways to assign those colors to the groups of vertices in π.

The Gibbs conditionals have the form

$$p_{\Pi|\Pi(-n)}(\Pi = y \,|\, \Pi(-n)) = \frac{\frac{q!}{(q-|y|)!}}{\sum_{x \text{ consistent with } \Pi(-n)} \frac{q!}{(q-|x|)!}} = \frac{\frac{1}{(q-|y|)!}}{\sum_{x \text{ consistent with } \Pi(-n)} \frac{1}{(q-|x|)!}}. \tag{15}$$

In Equation (15), x and y are partitions of the whole set of N vertices.

In implementations, to simulate from the conditional Equation (15), it suffices to represent the partition with a color vector. Suppose we condition on $\Pi(-n)$) i.e. when the colors for all but the n vertex are fixed, and there are q' unique colors that have been used (q' can be strictly smaller than q). n can either take on a color in $[q']$ (as long as the color is not used by a neighbor), or take on the color $q'+1$ (if $q' < q$). The transition probabilities are computed from the induced partition sizes $|x|$.

G.2 General Markov Chain Settings

Ground truth. For clustering, we run 10 single-chain Gibbs samplers for 10,000 sweeps each; we discard the first 1,000 sweeps. For graph coloring, we also run 10 chains, but each for 100,000 sweeps and discard the first 10,000. We compute an unthinned MCMC estimate from each chain and use the average across the 10 chains as ground truth. The standard errors across chains are very small. Dividing the errors by the purported ground truth yields values with magnitude smaller than 5×10^{-3}. In percentage error, this is less than 0.5%, which is orders of magnitude smaller than the percentage errors from coupled chains or naive parallel estimates.[5]

Sampler initializations. In clustering, we initialize each chain at the partition where all elements belong to the same element i.e. the one-component partition. In graph coloring, we initialize the Markov chain by greedily coloring the vertices. Our intuition suggests that coupling should be especially helpful relative to naively parallel chains when samplers require a large burn-in – since slow mixing induces bias in the uncoupled chains. In general, one cannot know in advance if that bias is present or not, but we can try to encourage suboptimal initialization in our experiments to explore its effects. For completeness, we consider alternative initialization schemes, such as k-means, in Figure 17.

Choice of hyperparameters in aggregate estimates. Recall that Equation (2) involves two free hyperparameters, ℓ and m, that we need to set. A general recommendation from Jacob et al. [2020, Section 3.1] is to select $m = 10\ell$ and ℓ to be a large quantile of the meeting time distribution. We take heed of these suggestions, but also prioritize m's that are small because we are interested in the time-limited regime. Larger m leads to

[5]The percentage errors for LCP are typically 0.01%, while percentage errors for co-clustering are typically 0.1%.

longer compute times across both coupled chains and naively parallel chains, and the bias in naively parallel chains is more apparent for shorter m: see Figure 16. In the naive parallel case, we discard the first 10% of sweeps completed in any time budget as burn-in steps. In our trimmed estimates, we remove the most extreme 1% of estimates (so 0.5% in either directions).

Simulating many processes. To quantify the sampling variability of the aggregate estimates (sample or trimmed mean across J processors), we first generate a large number (V = 180,000) of coupled estimates $H_{\ell:m}(X^j, Y^j)$ (and V naive parallel estimates U^j, where the time to construct $H_{\ell:m}(X^j, Y^j)$ is equal to the time to construct U^j).[6] For each J, we batch up the V estimates in a consistent way across coupled chains and naive parallel, making sure that the equality between coupled wall time and naive parallel wall time is maintained. There are $I = V/J$ batches. For the ith batch, we combine $H_{\ell:m}(X^j, Y^j)$ (or U^j) for indices j in the list $[(i-1)J+1, iJ]$ to form $H^{(i)}_{c,J}$ (or $H^{(i)}_{u,J}$) in the sense of Section 5.2. By this batching procedure, smaller values of J have more batches I. The largest J we consider for GENE, K-REGULAR and ABALONE is 2,750 while that for SYNTHETIC and SEED is 1,750. This mean the largest J has at least 57 batches.

To generate the survival functions (last column of Figure 2), we use 600 draws from the (censored) meeting time distribution by simulating 600 coupling experiments.

G.3 Datasets Preprocessing, Hyperparameters, Dataset-Specific Markov Chain Settings

gene i.e. single-cell RNAseq. We extract $D = 50$ genes with the most variation of $N = 200$ cells. We then take the log of the features, and normalize so that each feature has mean 0 and variance 1. We target the posterior of the probabilistic model in Section 2.1 with $\alpha = 1.0$, $\mu_0 = 0_D$, diagonal covariance matrices $\Sigma_0 = 0.5 I_D$, $\Sigma_1 = 1.3 I_D$. Notably, this is a simplification of the set-up considered by Prabhakaran et al. [2016], who work with a larger dataset and additionally perform fully Bayesian inference over these hyperparameters. That the prior variance is smaller than the noise variance yields a "challenging" clustering problem, where the cluster centers themselves are close to each other and observations are noisy realizations of the centers. We set $\ell = 30$ and $m = 300$.

seed i.e. wheat seed measurements. The original dataset from Charytanowicz et al. [2010] has 8 features; we first remove the "target" feature, which contains label information for supervised learning. Overall there are $N = 210$ observations and $D = 7$ features. We normalize each feature to have mean 0 and variance 1. We target the posterior of the probabilistic model in Section 2.1 with $\alpha = 1.0$, $\mu_0 = 0_D$, diagonal covariance matrices $\Sigma_0 = 1.0 I_D$, $\Sigma_1 = 1.0 I_D$. We set $\ell = 10$ and $m = 100$.

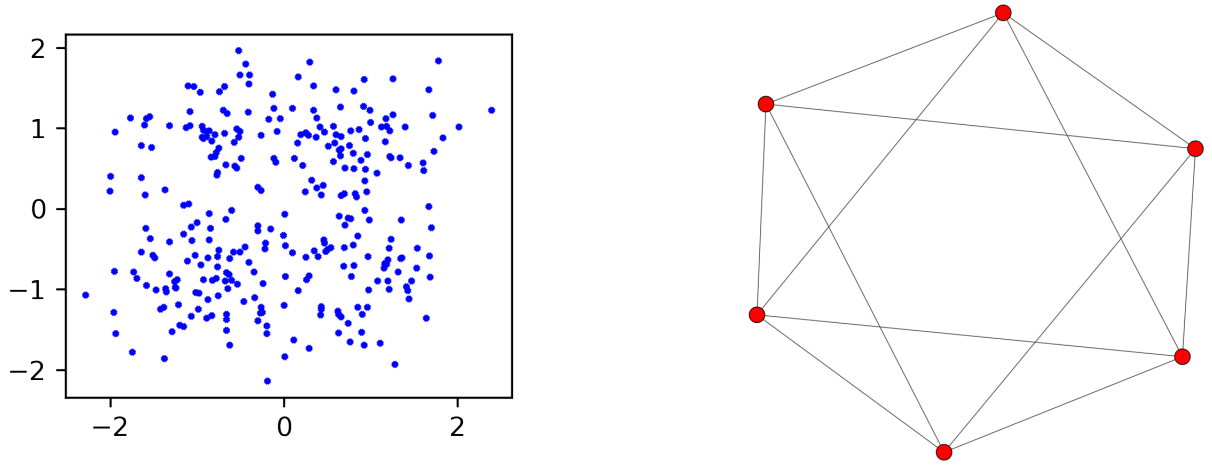
synthetic. We generate $N = 300$ observations from a 4-component mixture model in 2 dimensions. The four cluster centers are $[-0.8, -0.8], [-0.8, 0.8], [0.8, -0.8], [0.8, 0.8]$ Each data point is equally likely to come from one of four components; the observation noise is isotropic, zero-mean Gaussian with standard deviation 0.5. These settings result in a dataset where the observations form clear clusters, but there is substantial overlap at the cluster boundaries – see Figure 5a.

On this data, we target the posterior of the probabilistic model in Section 2.1 with $\alpha = 0.2$, $\mu_0 = 0_D$, diagonal covariance matrices $\Sigma_0 = 0.75 I_D$, $\Sigma_1 = 0.7 I_D$. Different from GENE, the prior variance is larger than the noise variance for SYNTHETIC. We set $\ell = 10$, $m = 100$.

abalone i.e. physical measurements of abalone specimens. The original dataset from Nash et al. [1994] has 9 features; we first remove the "Rings" feature, which contains label information for supervised learning, and the "Sex" feature, which contains binary information that is not compatible with the Gaussian DPMM generative model. Overall there are N = 4,177 observations and $D = 7$ features. We normalize each feature to have mean 0 and variance 1. We target the posterior of the probabilistic model in Section 2.1 with $\alpha = 1.0$, $\mu_0 = 0_D$, diagonal covariance matrices $\Sigma_0 = 2.0 I_D$, $\Sigma_1 = 2.0 I_D$. We set $\ell = 10$ and $m = 100$.

k-regular. Anticipating that regular graphs are hard to color, we experiment with a 4-regular, 6-node graph – see Figure 5b. The target distribution is the distribution over vertex partitions induced by uniform colorings

[6] The best computing infrastructure we have access to has only 400 processors, so we generate these V estimates by sequential running $nEst/400$ batches, each batch constructing 400 estimates in parallel.



(a) SYNTHETIC data (b) K-REGULAR data

Figure 5: Visualizing synthetic data

using 4 colors. We set $\ell = 1$, $m = 4$.

G.4 Visualizing Synthetic Data

Figure 5 visualizes the two synthetic datasets.

H ALL FIGURES

H.1 gene

Figure 6 shows results for LCP estimation on GENE – see Figure 15 for results on co-clustering. The two panels that did not appear in Figure 2 are the left panel of Figure 6b and the right panel of Figure 6c. The left panel of Figure 6b is the same as Figure 1: the y-axis plots the RMSE instead of the range of losses. As expected from the bias-variance decomposition, the RMSE for coupled estimates decreases with increasing J because of unbiasedness, while the RMSE for naive parallel estimates does not go away because of bias. The right panel of Figure 6c plots typical d distances between coupled chains under different couplings as a function of the number of sweeps done. d decreases to zero very fast under OT coupling, while it is possible for chains under maximal and common RNG couplings to be far from each other even after many sampling steps.

H.2 synthetic

Figure 7 shows results for LCP estimation on SYNTHETIC – see Figure 15 for results on co-clustering.

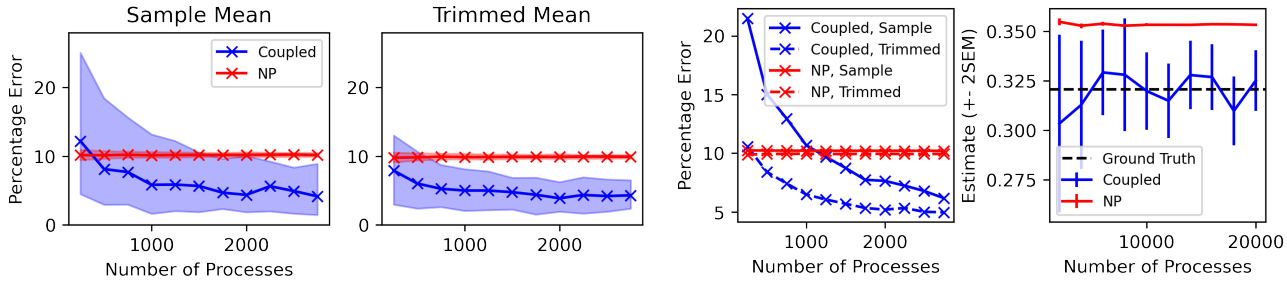
H.3 seed

Figure 8 shows results for LCP estimation on SEED – see Figure 15 for results on co-clustering.

H.4 abalone

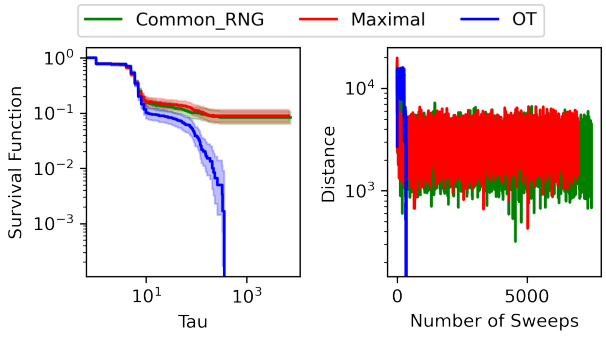
Figure 9 shows results for LCP estimation on ABALONE. In Figure 9a and Figure 9b, we do not report results for the trimmed estimator with the default trimming amount (0.01 i.e. 1%). This trimming amount is too large for the application, and in Figure 10, we show that trimming the most extreme 0.1% yields much better estimation.

In Figure 10, the first panel (from the left) plots the errors incurred using the trimmed mean with the default $\alpha = 1\%$. Trimming of coupled chains is still better than naive parallelism, but worse than sample mean of coupled chains. In the second panel, we use $\alpha = 0.1\%$, and the trimming of coupled chains performs much better. In the third panel, we fix the number of processes to be 2000 and quantify the RMSE as a function of the trimming



(a) Losses

(b) RMSE and intervals



(c) Coupling choice

Figure 6: Results on GENE.

amount (expressed in percentages). We see a gradual decrease in the RMSE as the trimming amount is reduced, indicating that this is a situation in which smaller trimming amounts is prefered.

H.5 k-regular

Figure 11 shows results for CC$(2, 4)$ estimation on K-REGULAR.

I METRIC IMPACT

I.1 Definition Of Variation Of Information Metric

Variation of information, or VI, is defined in Meilă [2007, Equation 16]. We replicate the definition in what follows. Let π and ν be two partitions of $[N]$. Denote the clusters in π by $\{A^1, A^2, \ldots, A^K\}$ and the clusters in ν by $\{B^1, B^2, \ldots, B^{K'}\}$. For each $k \in [K]$ and $k' \in K'$, define the number $P(k, k')$ to be

$$P(k, k') := \frac{|A^k \cap B^{k'}|}{N}.$$

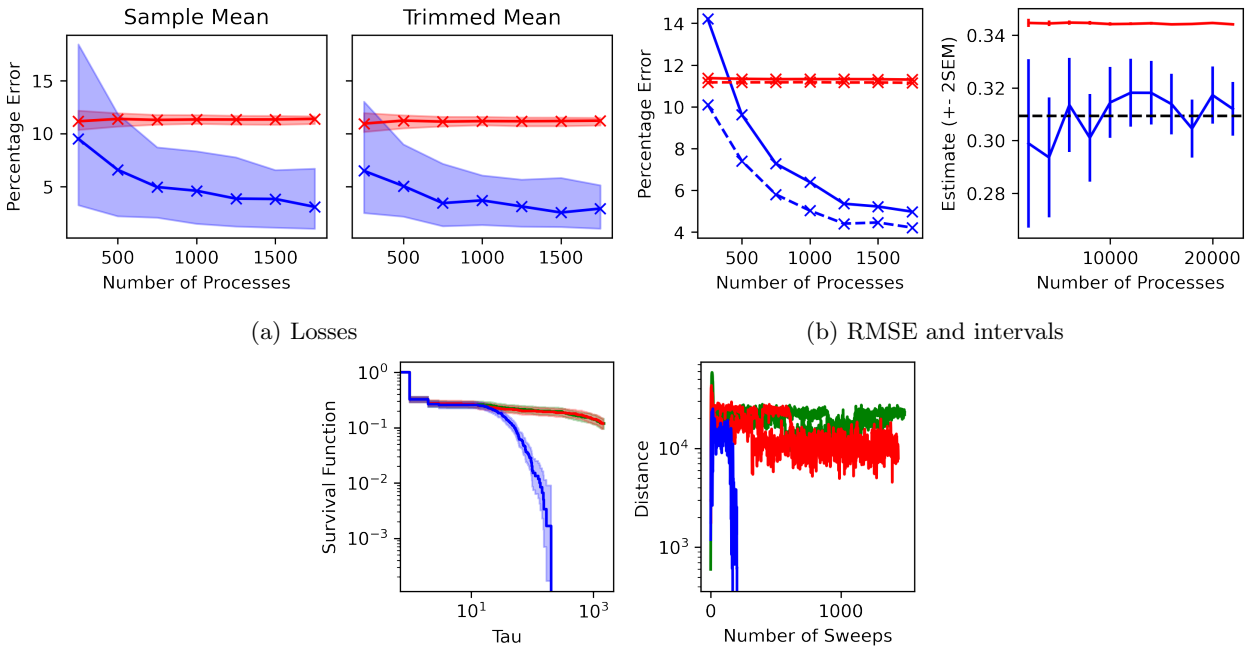
$|A^k \cap B^{k'}|$ is the size of the overlap between A^k and $B^{k'}$. Because of the normalization by N, the $P(k, k')$'s are non-negative and sum to 1, hence can be interpreted as probability masses. Summing across all k (or k') has a marginalization effect, and we define

$$P(k) := \sum_{k'=1}^{K'} P(k, k').$$

Similarly we define $P'(k') := \sum_{k=1}^{K} P(k, k')$. The VI metric is then

$$d_I(\pi, \nu) = \sum_{k=1}^{K} \sum_{k'=1}^{K'} P(k, k') \log \frac{P(k, k')}{P(k)P(k')}. \tag{16}$$

In terms of theoretical properties, Meilă [2007, Property 1] shows that d_I is a metric for the space of partitions.



(a) Losses

(b) RMSE and intervals

(c) Coupling choice

Figure 7: Results on SYNTHETIC. Figure legends are the same as Figure 6. The results are consistent with Figure 2.

I.2 Impact Of Metric On Meeting Time

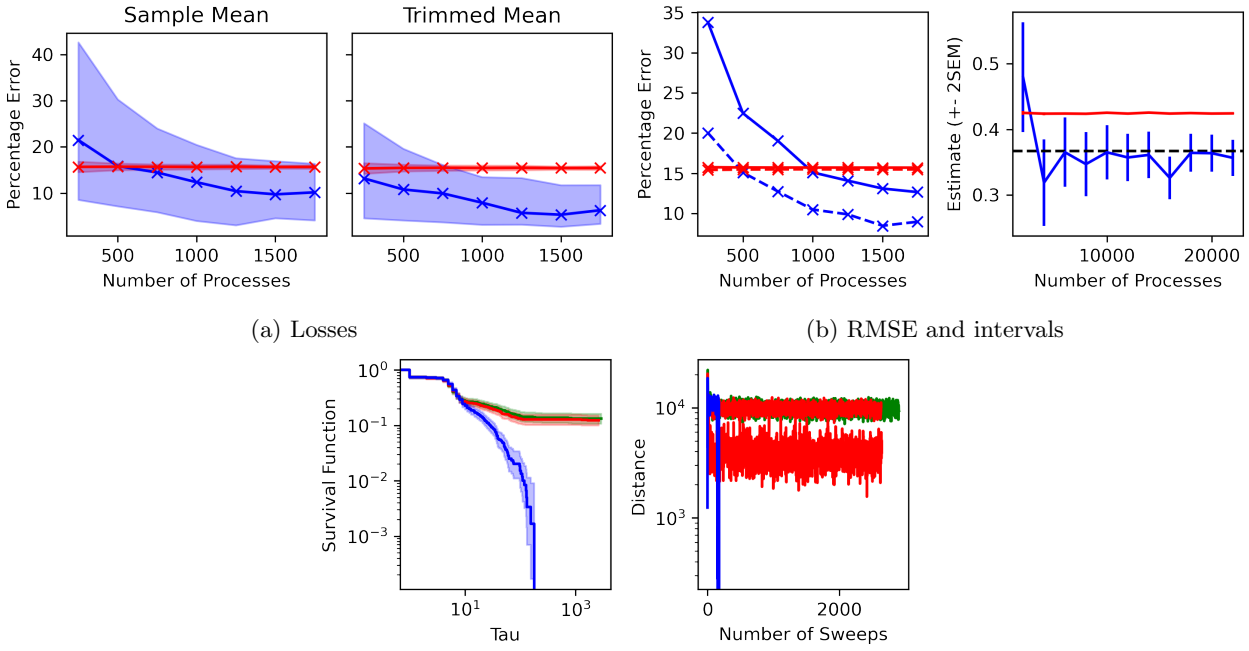
In Figures 12a to 12d, we examine the effect of metric on the meeting time for coupled chains. In place of the Hamming metric in Equation (5), we can use the variation of information (VI) metric from Equation (16) in defining the OT problem (Equation (4)). Based on the survival functions, the meeting time under VI metric is similar to meeting time under the default Hamming metric: in all cases, the survival functions lie mostly right on top of each other. Time is measured in number of sweeps taken, rather than processor time, because under Hamming metric we have a fast implementation (Section 4.2) while we are not aware of fast implementations for the VI metric. Hence, our recommended metric choice is Hamming (Equation (5)).

J EXTENSION TO SPLIT-MERGE SAMPLER

SplitMerge(i, j, X) is the "Restricted Gibbs Sampling Split–Merge Procedure" from Jain and Neal [2004], where our implementation proposes 1 split–merge move and uses 5 intermediate Gibbs scan to compute the proposed split (or merge) states.

We refer to Appendix G for comprehensive experimental setup. The LCP estimation results for GENE are given in Figure 13. Instead of the one-component initialization, we use a k-means clustering with 5 components as initialization. m is set to be 100, while ℓ is 10. Switching from pure Gibbs sampler to split-merge samplers can reduce the bias caused by a bad initialization. But there is still bias that does not go away with replication, and the results are consistent with Figure 2.

We also have split-merge results for estimation of CC$(0, 1)$ on SYNTHETIC in Figure 14. m is set to be 50, while ℓ is 5.



(a) Losses

(b) RMSE and intervals

(c) Coupling choice

Figure 8: Results on SEED. Figure legends are the same as Figure 6. The results are consistent with Figure 2.

K MORE RMSE PLOTS

K.1 Different Functions Of Interest

Figure 15 displays co-clustering results for clustering data sets. The results are consistent with those for LCP estimation. Co-clustering appears to be a more challenging estimation problem than LCP, indicated by the higher percentage errors for the same m.

K.2 Different Minimum Iteration (m) Settings

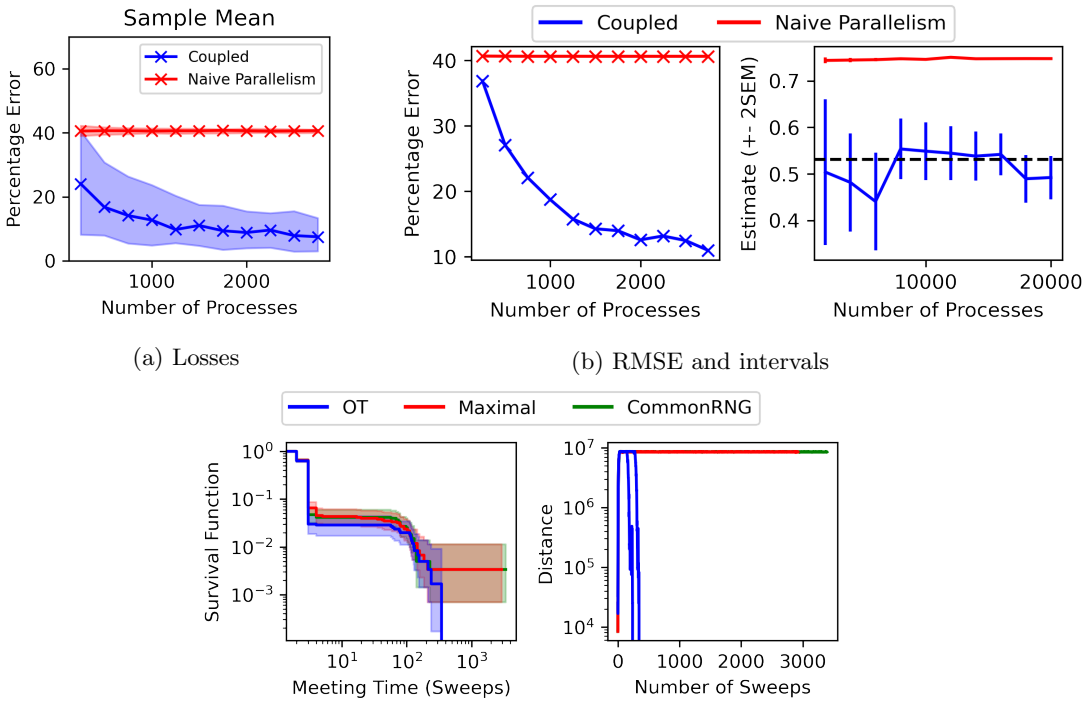
In Figure 16, with an increase in m (from the default 100 to 150), the bias in the naive parallel approach reduces (percentage error goes from 15% to 10%, for instance), and the variance of coupled chains' estimates also reduce.

K.3 Different Initialization

In Figure 17, we initialize the Markov chains with the clustering from a k-means clustering with 5 clusters, instead of the one-component initialization. Also see Figure 13 for more kmeans initialization results. The bias from naive parallel is smaller than when initialized from the one-component initialization (RMSE in Figure 17 is around 5% while RMSE in Figure 6 is about 10%). However, the bias is still significant enough that even with a lot of processors, naive parallel estimates are still inadequate.

K.4 Different DPMM Hyperparameters

For convenience, throughout our experiments, we use diagonal covariance matrices $\Sigma_0 = s_0 I_D$ and $\Sigma_1 = s_1 I_D$, where the variances in different dimensions are the same. We find that the bias of standard MCMC is influenced by s_0 and s_1: some settings cause naive parallel chains to have meaningfully large bias, while others do not. Figure 18 illustrates on SYNTHETIC that when s_1 is small compared to s_0, standard MCMC actually has small bias even when run for a short amount of time. For values of s_1 that are closer to (or larger than s_0), the bias in standard MCMC is much larger. m and ℓ are set to be 100 and 10 across these settings of s_1.



(a) Losses

(b) RMSE and intervals

(c) Coupling choice

Figure 9: Results on ABALONE. Similar to Figure 2, coupled chains perform better than naive parallelism with more processes, and our coupling yields smaller meeting times than label-based couplings. See Figure 10 for the performance of trimmed estimators.

L MORE MEETING TIME PLOTS

In Figure 19, we generate Erdős-Rényi random graphs, including each possible edge with probability 0.2. The graph in the first two panels has $N = 25$ vertices, while the one in the latter two panels has $N = 30$. We determine a sufficient number of colors by first greedily coloring the vertices. It turns out that 6 colors is sufficient to properly color the vertices in either set of panels.

M ESTIMATES OF PREDICTIVE DENSITY

M.1 Data, Target Model, And Definition Of Posterior Predictive

As the posterior predictive is easiest to visualize in one dimension, we draw artificial data from a univariate, 10-component Gaussian mixture model with known observational noise standard deviation $\sigma = 2.0$, and use a DPMM to analyze this data. The cluster proportions were generated from a symmetric Dirichlet distribution with mass 1 for all 10-coordinates. The cluster means were randomly generated from $\mathcal{N}(0, 10^2)$. Since this is an artificial dataset, we can control the number of observations: we denote GMM-100 to be the dataset of 100 observations, for instance.

The target DPMM has $\mu_0 = 0, \alpha = 1, \Sigma_0 = 3.0$, and $\Sigma_1 = 2.0$

The function of interest is the posterior predictive density

$$\Pr(W_{N+1} \in dx \,|\, W_{1:N}) = \sum_{\Pi_{N+1}} \Pr(W_{N+1} \in dx \,|\, \Pi_{N+1}, W_{1:N}) \Pr(\Pi_{N+1} \,|\, W_{1:N}). \tag{17}$$

In Equation (17), Π_{N+1} denotes the partition of the data $W_{1:(N+1)}$. To translate Equation (17) into an integral over just the posterior over Π_N (the partition of $W_{1:N}$) we break up Π_{N+1} into (Π_N, Z) where Z is the cluster

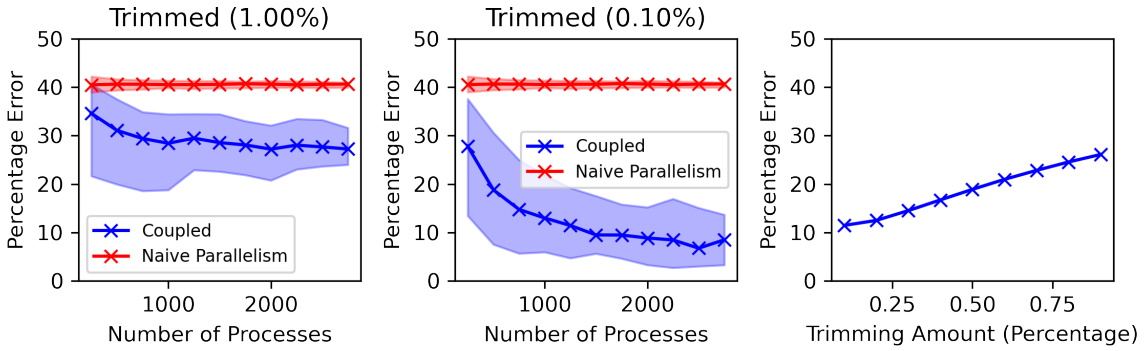
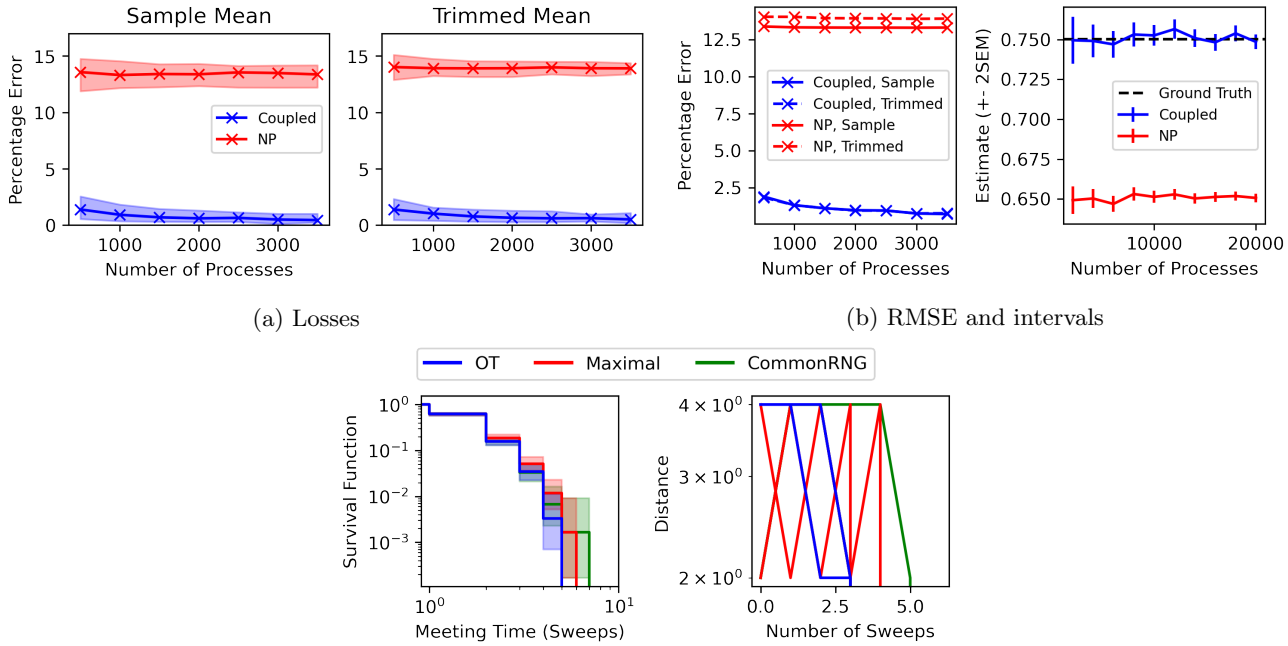


Figure 10: Effect of trimming amount on ABALONE.



(a) Losses

(b) RMSE and intervals

(c) Coupling choice

Figure 11: Results on K-REGULAR. Figure legends are the same as Figure 6.

indicator specifying the cluster of Π_N (or a new cluster) to which W_{N+1} belongs. Then

$$\Pr(W_{N+1} \in dx \,|\, W_{1:N}) = \sum_{\Pi_N} \left[\sum_{Z} \Pr(W_{N+1} \in dx, Z \,|\, \Pi_N, W_{1:N}) \right] \Pr(\Pi_N \,|\, W_{1:N})$$

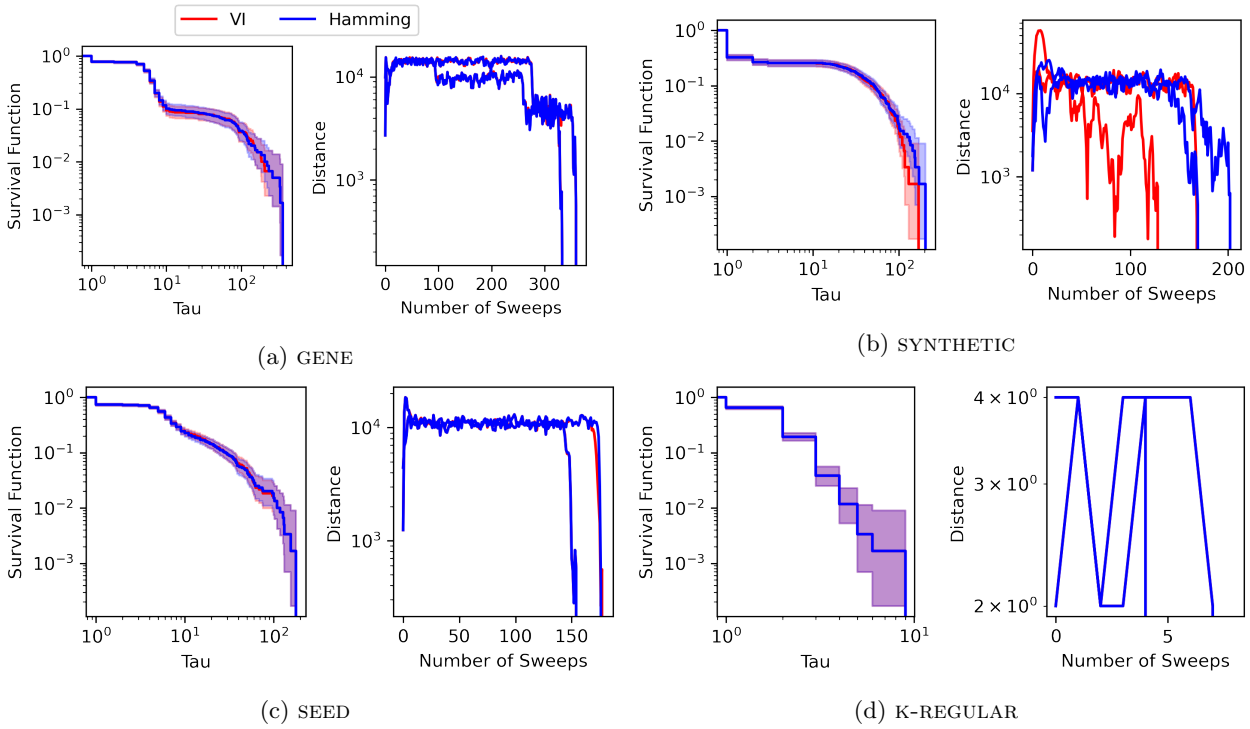
Each $\Pr(W_{N+1} \in dx, Z \,|\, \Pi_N, W_{1:N})$ is computed using the prediction rule for the CRP and Gaussian conditioning. Namely

$$\Pr(W_{N+1} \in dx, Z \,|\, \Pi_N, W_{1:N}) = \underbrace{\Pr(W_{N+1} \in dx \,|\, Z, \Pi_N, W_{1:N})}_{\text{Posterior predictive of Gaussian}} \times \underbrace{\Pr(Z \,|\, \Pi_N)}_{\text{CRP prediction rule}} .$$

The first term is computed with the function used during Gibbs sampling to reassign data points to clusters. In the second term, we ignore the conditioning on $W_{1:N}$, since Z and $W_{1:N}$ are conditionally independent given Π_N.

M.2 Estimates Of Posterior Predictive Density

We first discretize the domain using 150 evenly-spaced points in the interval $[-20, 30]$: these are the locations at which to evaluate the posterior predictive. We set $m = 100$ and $\ell = 10$ in constructing the estimate from



(a) GENE (b) SYNTHETIC

(c) SEED (d) K-REGULAR

Figure 12: Hamming and VI metric induce similar meeting time

Equation (2). We average the results from 400 coupled chain estimates. In each panel of Figure 20, the solid blue curve is an unbiased estimate of the posterior predictive density: the error across replicates is very small and we do not plot uncertainty bands. The black dashed curve is the true density of the population i.e. the 10-component Gaussian mixture model density. The grey histogram bins the observed data.

M.3 Posterior Predictives Become More Alike True Data Generating Density

In Figure 20, by visual inspection, the distance between the posterior predictive density and the underlying density decreases as N increases. This is related to the phenomenon of posterior concentration, where with more observations gathered, the Bayesian posterior concentrates more and more on the true data generating process. We refer to Ghosal et al. [1999], Lijoi et al. [2005] for more thorough discussions of posterior concentration. In what follows, we justify the concentration behavior for Gaussian DPMM, when the observation noise is correctly specified.

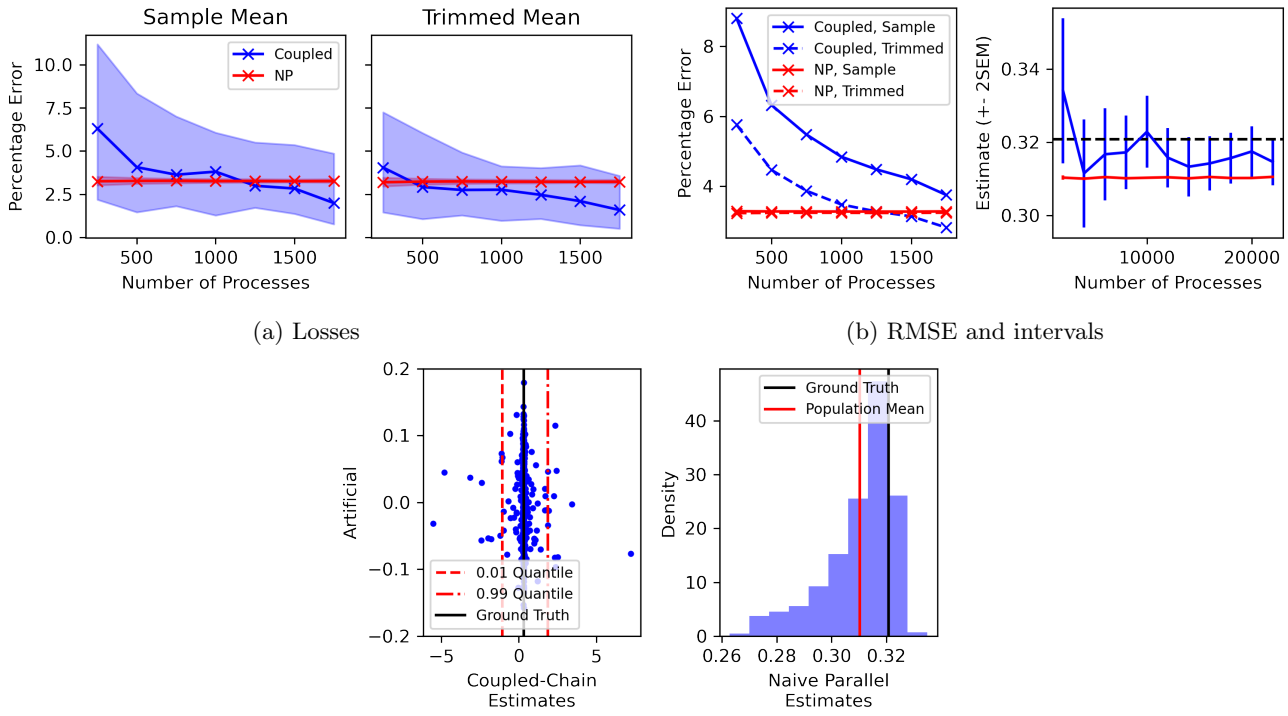
Theorem 2 (DP mixtures prior is consistent for finite mixture models) *Let $f_0(x) := \sum_{i=1}^{m} p_i \mathcal{N}(x \mid \theta_i, \sigma_1^2)$ be a finite mixture model. Suppose we observe iid data $X_1, \ldots, X_n$ from f_0. Consider the following probabilistic model*

$$\begin{aligned} \widehat{P} &\sim \mathrm{DP}(\alpha, \mathcal{N}(0, \sigma_0^2)) \\ \theta_i \mid \widehat{P} &\overset{iid}{\sim} \widehat{P} && i = 1, 2, \ldots, n \\ X_i \mid \theta_i &\overset{indep}{\sim} \mathcal{N}(\theta_i, \sigma_1^2) && i = 1, 2, \ldots, n \end{aligned}$$

Let $\widehat{P}_n$ be the posterior predictive distribution of this generative process. Then with a.s. P_{f_0}

$$d_{TV}\left(\widehat{P}_n, P_{f_0}\right) \xrightarrow{n \to \infty} 0.$$

To prove Theorem 2, we first need some definitions and auxiliary results.



(a) Losses

(b) RMSE and intervals

(c) Estimates

Figure 13: Split-merge results on GENE

Definition 3 (Strongly consistent priors) *Suppose iid data $X_1, X_2, \ldots, X_n$ is generated from some probability measure measure that is absolutely continuous with respect to Lebesgue measure. Denote the density of this data generating measure by f_0. Let $\mathcal{F}$ be the set of all densities on $\mathbb{R}$. Consider the probabilistic model where we put a prior Π over densities f, and observations X_i are conditionally iid given f. We use P_f to denote the probability measure with density f. For any measurable subset A of $\mathcal{F}$, the posterior of A given the observations X_i is denoted $\Pi(A \,|\, X_{1:N})$. A strong neighborhood around f_0 is any subset of $\mathcal{F}$ containing a set of the form $V = \{f \in \mathcal{F} : \int |f - f_0| < \epsilon\}$ according to Ghosal et al. [1999]. The prior Π is strongly consistent at f_0 if for any strong neighborhood U,*

$$\lim_{n\to\infty} \Pi(U|X_{1:n}) = 1, \tag{18}$$

holds almost surely for $X_{1:\infty}$ distributed according to $P_{f_0}^{\infty}$.

Proposition 3 (Ghosh and Ramamoorthi [2003, Proposition 4.2.1]) *If a prior Π is strongly consistent at f_0 then the predictive distribution, defined as*

$$\widehat{P}_n(A \mid X_{1:n}) := \int_f P_f(A)\Pi(f \mid X_{1:n}) \tag{19}$$

also converges to f_0 in total variation in a.s. $P_{f_0}^{\infty}$

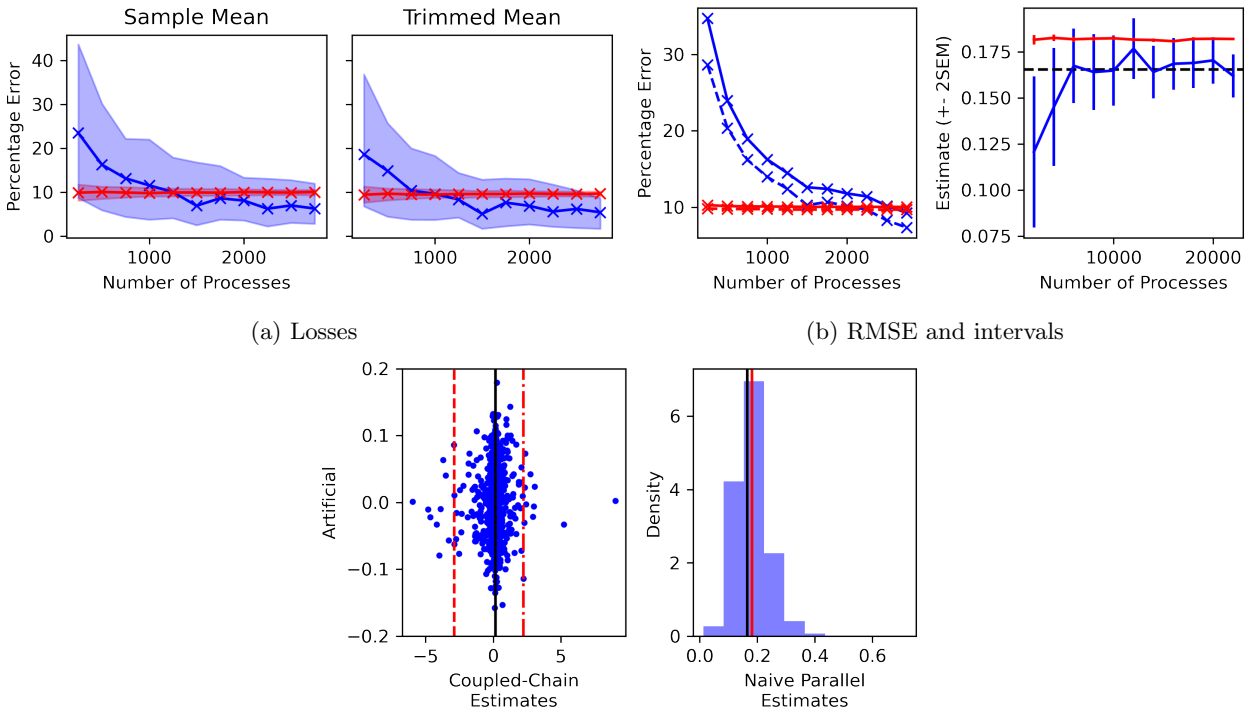
$$d_{TV}\left(\widehat{P}_n, P_{f_0}\right) \to 0.$$

The definition of posterior predictive density in Equation (19) can equivalently be rewritten as

$$\widehat{P}_n(A \mid X_{1:n}) = \Pr(X_{n+1} \in A \,|\, X_{1:n}),$$

since $P_f(A) = P_f(X_{n+1} \in A)$ and all the X's are conditionally iid given f.

We are ready to prove Theorem 2.



(a) Losses

(b) RMSE and intervals

(c) Estimates

Figure 14: Split-merge results on SYNTHETIC

Proof [Proof of Theorem 2] First, we can rewrite the DP mixture model as a generative model over continuous densities f

$$\begin{aligned} \widehat{P} &\sim \mathrm{DP}(\alpha, \mathcal{N}(0, \sigma_0^2)) \\ f &= \mathcal{N}(0, \sigma_1^2) * \widehat{P} \\ X_i \mid f &\overset{iid}{\sim} f \qquad\qquad i = 1, 2, \ldots, n \end{aligned} \tag{20}$$

where $\mathcal{N}(0, \sigma_1^2) * \widehat{P}$ is a convolution, with density $f(x) := \int_\theta \mathcal{N}(x - \theta | 0, \sigma_1^2) d\widehat{P}(\theta)$.

The main idea is showing that the posterior $\Pi(f | X_{1:n})$ is strongly consistent and then leveraging Proposition 3. For the former, we verify the conditions of Lijoi et al. [2005, Theorem 1].

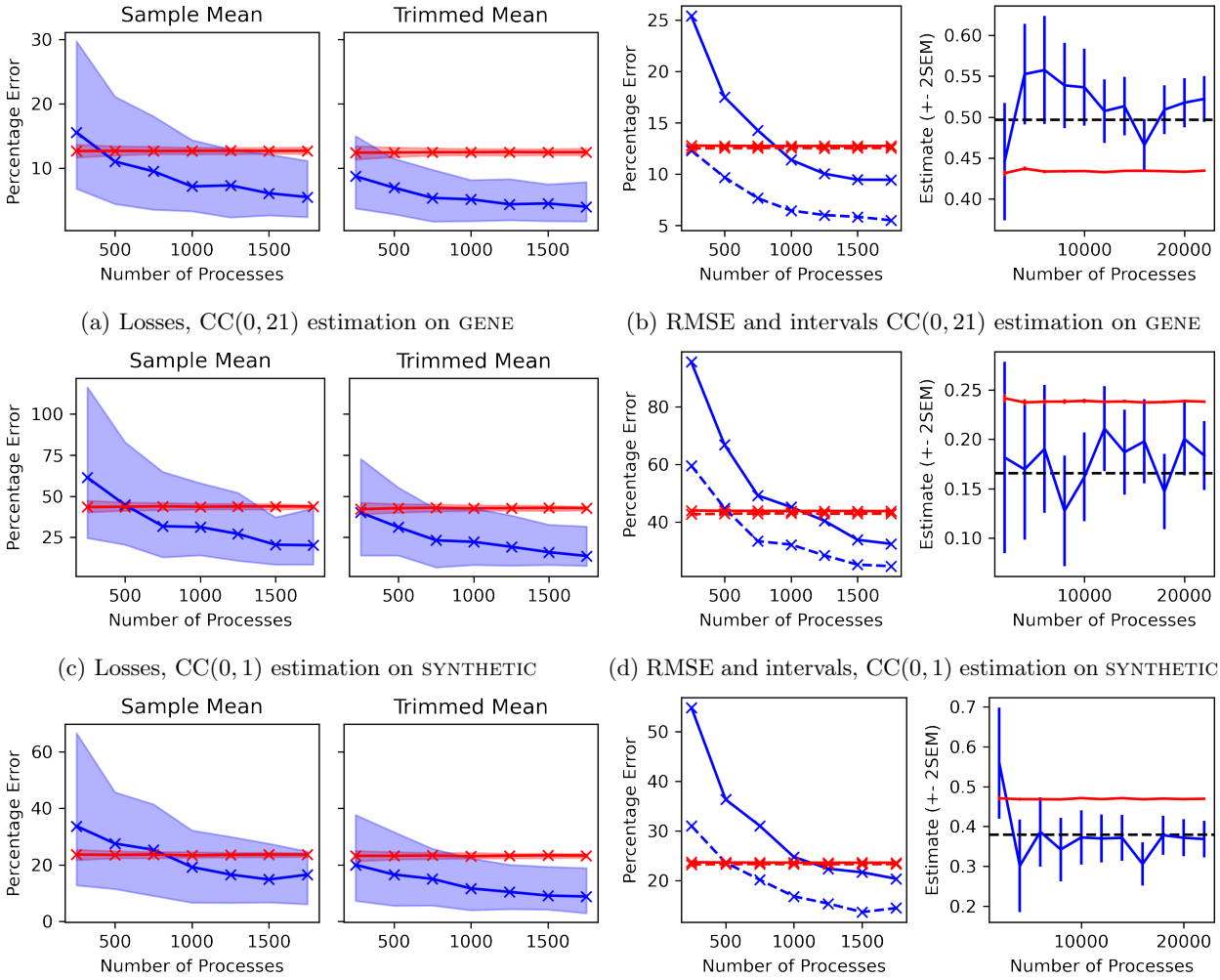
The first condition of Lijoi et al. [2005, Theorem 1] is that f_0 is in the K-L support of the prior over f in Equation (20). We use Ghosal et al. [1999, Theorem 3]. Clearly f_0 is the convolution of the normal density $\mathcal{N}(0, \sigma_1^2)$ with the distribution $P(.) = \sum_{i=1}^m p_i \delta_{\theta_i}$. $P(.)$ is compactly supported since m is finite. Since the support of $P(.)$ is the set $\{\theta_i\}_{i=1}^m$ which belongs in $\mathbb{R}$, the support of $\mathcal{N}(0, \sigma_0^2)$, by Ghosh and Ramamoorthi [2003, Theorem 3.2.4], the conditions on P are satisfied. The condition that the prior over bandwidths cover the true bandwidth is trivially satisfied since we perfectly specified σ_1.

The second condition of Lijoi et al. [2005, Theorem 1] is simple: because the prior over $\widehat{P}$ is a DP, it reduces to checking that

$$\int_{\mathbb{R}} |\theta| \mathcal{N}(\theta \mid 0, \sigma_0^2) < \infty$$

which is true.

The final condition trivial holds because we have perfectly specified σ_1: there is actually zero probability that σ_1 becomes too small, and we never need to worry about setting γ or the sequence σ_k. ■



(a) Losses, CC$(0, 21)$ estimation on GENE

(b) RMSE and intervals CC$(0, 21)$ estimation on GENE

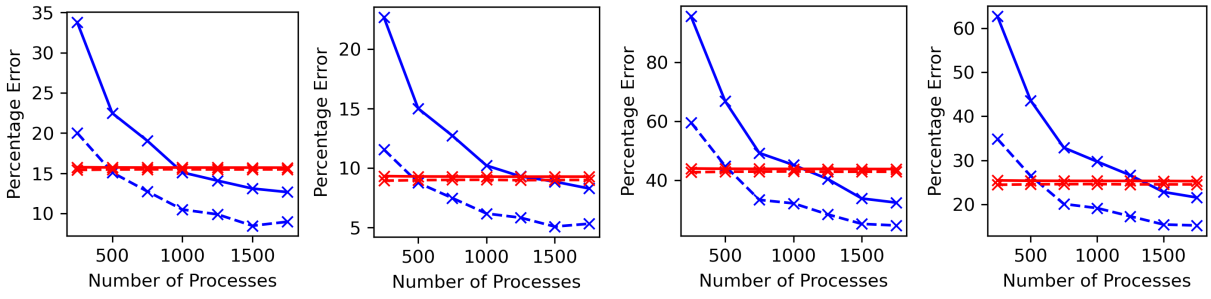
(c) Losses, CC$(0, 1)$ estimation on SYNTHETIC

(d) RMSE and intervals, CC$(0, 1)$ estimation on SYNTHETIC

(e) Losses, CC$(0, 19)$ estimation on SEED

(f) RMSE and intervals, CC$(0, 19)$ estimation on SEED

Figure 15: Co-clustering results for clustering data sets.



(a) $m = 100$, SEED

(b) $m = 150$, SEED

(c) $m = 100$, SYNTHETIC

(d) $m = 150$, SYNTHETIC

Figure 16: Impact of different m on the RMSE. The first two panels are LCP estimation for SEED. The last two panels are CC$(0, 1)$ estimation for SYNTHETIC.

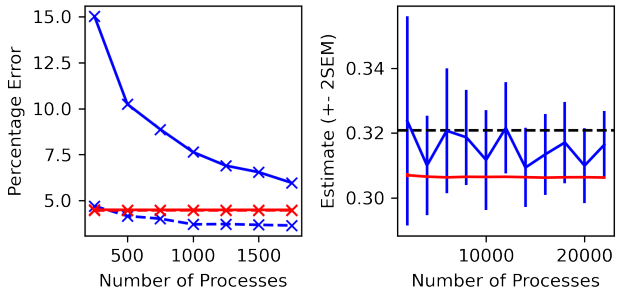
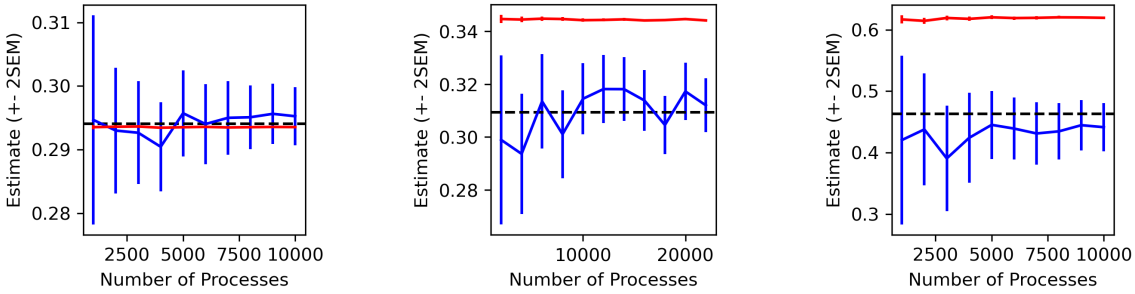
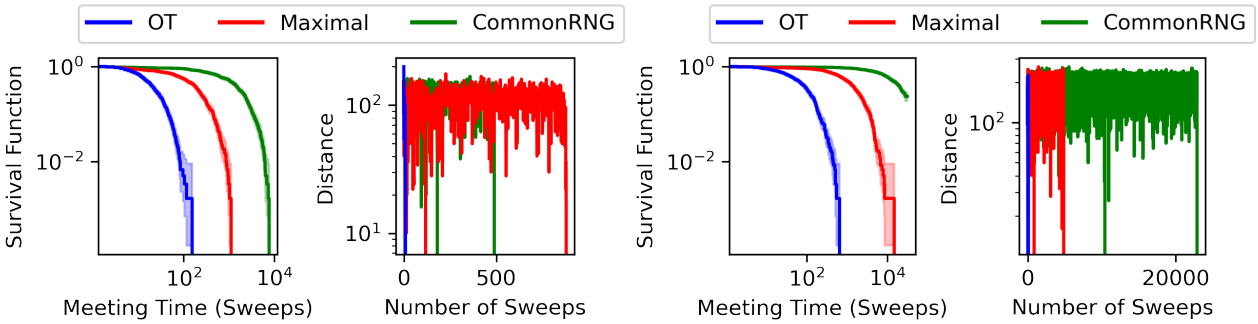


Figure 17: RMSE and intervals for GENE on k-means initialization.



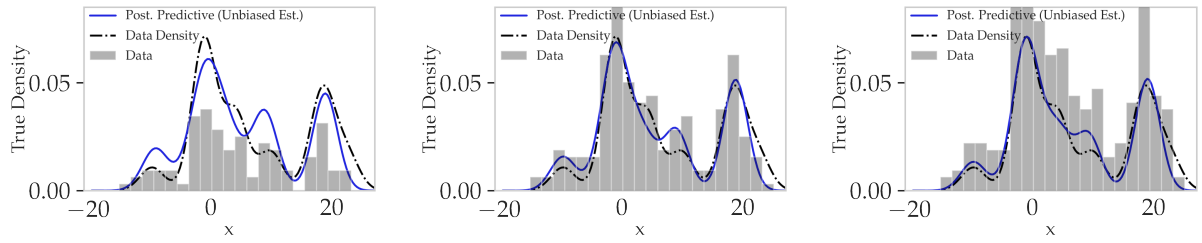
(a) $s_1 = 0.5, s_0 = 0.75$ (b) $s_1 = 0.7, s_0 = 0.75$ (c) $s_1 = 0.85, s_0 = 0.75$

Figure 18: The bias in naive parallel estimates is a function of the DPMM hyperparameters.



(a) $N = 25$ (b) $N = 30$

Figure 19: Meeting time under OT coupling is better than alternative couplings on Erdos–Renyi graphs, indicated by the fast decrease of the survival functions.



(a) GMM-100 (b) GMM-200 (c) GMM-300

Figure 20: Posterior predictive density for different number of observations N.