STAT 222: Computational Approaches to Marginal Posterior Simulation

Kevin McKinnon
UC Santa Cruz

1. Introduction

Simulating marginal posterior distributions in the context of mixtures of Dirichlet Processes (DP) is necessary for models to properly capture the messy, real-world complexity of many datasets. Because of this, approaches for accomplishing this simulation is at the heart of Bayesian non-parametrics (BNP). In its fledgling days, BNP was restricted to simple models with easily-described posterior distributions when studying a dataset, but there has been substantial work since then to create algorthims that enable the use of more complicated models. In this paper, I will explain and explore three such methods, as first presented in MacEachern and Müller (1998) and Neal (2000). Throughout the body of this work, I will refer to these as the No Gaps, the No Gaps + Auxiliary, and the Metropolis-Hastings methods for marginal posterior simulation of mixtures of DP models.

The remainder of this paper is layed out as follows: Section 2 explains the notation and chosen model, Section 3 provides a brief motivation for this work, Section 4 introduces the various simulation algorithms, Section 5 compares these algorithms, and Section 6 summarize these findings. The `python`-based Jupyter Notbook created for this analysis is available on my GitHub.

2. Notation and Model

The data we will be studying throughout this work are $n = 250$ datapoints generated by

$$y_i \sim 0.2N(-5,1) + 0.5N(0,1) + 0.3N(3.5,1),\ i = 1,\dots,n, \tag{1}$$

as presented in homework set 3 for this course. As in the same assignment, we consider the model

$$\begin{aligned} y_i|\theta_i,\phi &\overset{ind.}{\sim} N(y_i|\theta_i,\phi),\ \ i=1,\dots,n \\ \theta_i|G &\overset{i.i.d.}{\sim} G,\ \ i=1,\dots,n \\ G|\alpha,\mu,\tau^2 &\sim DP(\alpha, G_0 = N(\mu,\tau^2)) \\ \alpha|a_\alpha,b_\alpha &\sim Ga(\alpha|a_\alpha,b_\alpha) \\ \mu|a_\mu,b_\mu &\sim IG(\mu|a_\mu,b_\mu) \\ \tau^2|a_{\tau^2},b_{\tau^2} &\sim IG(\tau^2|a_{\tau^2},b_{\tau^2}) \\ \phi|a_\phi,b_\phi &\sim IG(\phi|a_\phi,b_\phi). \end{aligned} \tag{2}$$

In this notation the kernel density, $k(y|\theta,\phi)$, is the normal distribution shown above, $N(y|\theta,\phi)$. Because this model will allow for clustering of the θ_i values, we introduce the notation of $\vec{\theta}^* = \{\theta^*_j,\ j=1,\dots,n^*\}$, which are the unique θ_i values in $\vec{\theta} = \{\theta_i,\ i=1,\dots,n\}$, where n^* is thus the number of unique clusters. In this notation, we also have n_j is the number of observations that cluster with a given θ^*_j and $s_i = j$ iff $\theta_i = \theta^*_j$ being the indicator. As we will rely on a Polya Urn scheme throughout this paper, we also point out that "variable$^-$" indicates what happens to a given variable when a y_i or θ_i is removed (e.g. n^{*-} is the number of unique clusters in $\vec{\theta}^- = \{\theta_{i^-},\ i^- = 1,\dots,n-1\}$, which is $\vec{\theta}$ with θ_i removed).

This model gives us convenient forms for the posterior full conditionals on all parameters, which enables standard Gibbs sampling approaches. This will serve as the baseline to compare the novel methods of simulation. Specifically, for ϕ, μ, and τ^2 we find:

$$p(\phi|a_\phi,b_\phi,\vec{\theta},\text{data}) \propto IG\left(a_\phi + \frac{n}{2}, b_\phi + \frac{1}{2}\sum_{i=1}^{n}(y_i-\theta_i)^2\right) \tag{3}$$

and

$$p(\mu|a_\mu,b_\mu,\vec{\theta},\tau^2,\text{data}) \propto N\left(\frac{a_\mu\tau^2 + b_\mu\sum_{j=1}^{n^*}\theta^*_j}{\tau^2+n^*b_\mu}, \frac{\tau^2 b_\mu}{\tau^2+n^*b_\mu}\right) \tag{4}$$

and

$$p(\tau^2|a_{\tau^2}, b_{\tau^2}, \vec{\theta}, \mu, \text{data}) \propto IG\left(a_{\tau^2} + \frac{n^*}{2}, b_{\tau^2} + \frac{1}{2}\sum_{j=1}^{n^*}(\theta_j^* - \mu)^2\right). \tag{5}$$

To simulate α, we use an auxiliary variable, η. Specifically, $(\eta|\alpha, \text{data}) \sim Beta(\alpha+1, n)$ and

$$p(\alpha|\eta, \text{data}) \sim \epsilon Ga(\alpha|a_\alpha + n^*, b_\alpha - \log(\eta)) + (1-\epsilon)Ga(\alpha|a_\alpha + n^* - 1, b_\alpha - \log(\eta)) \tag{6}$$

where $\epsilon = \frac{a_\alpha + n^* - 1}{n(b_\alpha - \log(\eta)) + a_\alpha + n^* - 1}$. Finally, the sampling for $\vec{\theta}$ relies on updating each θ_i in turn (as shown in Escobar and West, 1995); we see that $p(\theta_i|\vec{\theta}_{i-}, \dots)$ is a mixture of the n^{*-} unique θ_j^{*-} values and the posterior for θ_i from y_i:

$$\frac{1}{\alpha q_0 + \sum_{j=1}^{n^{*-}} n_j^- q_j}\left[\alpha q_0 h(\theta_i|\mu, \tau^2, \phi, y_i) + \sum_{j=1}^{n^{*-}} n_j^- q_j \delta_{\theta_j^{*-}}(\theta_i)\right] \tag{7}$$

with g_0 being the density of G_0, $q_j = k(y_i|\theta_j^{*-}, \phi)$, and the two remaining values/distributions, q_0 and $h(\theta_i|\mu, \tau^2, \phi, y_i)$, working out to

$$\begin{aligned} h(\theta_i|\mu, \tau^2, \phi, y_i) &\propto k(y_i|\theta_i, \phi) g_0(\theta_i|\mu, \tau^2) \\ &\propto N\left(\theta_i|\frac{\tau^2 y_i + \mu\phi}{\tau^2 + \phi}, \frac{\tau^2\phi}{\tau^2 + \phi}\right) \end{aligned} \tag{8}$$

and

$$\begin{aligned} q_0 &= \int k(y_i|\theta, \phi) g_0(\theta|\mu, \tau^2) d\theta \\ &= N\left(y_i|\mu, \tau^2 + \phi\right). \end{aligned} \tag{9}$$

The specific hyperparameters used for the prior distributions are shown in Table 1. The these values were chosen to be relatively close to the generator distribution where they could showcase a relatively well-behaved result.

Table 1: Values of hyperparameters used in the priors on ϕ, μ, τ^2, and α, which are motivated by the data.

Values	**Hyperprior Parameter Choices** ϕ	μ	τ^2	α	**Implied** $E(n^*|\alpha)$
(a, b)	(2.62,1.62)	(0,4)	(2.5,4.5)	(2,4)	3
Resulting Prior (mean, variance)	$(1, 1^2)$	$(0, 2^2)$	$(3, 2^2)$	$(0.5, 0.125^2)$	

3. Motivation

Here, we take a moment to recognize that the Gibbs sampling approach only works for this model because we had conjugacy between $k(y|\theta, \phi)$ and $g_0(\theta|\mu, \tau^2)$. Were this not the case, simulating from $h(\theta_i|\mu, \tau^2, \phi, y_i)$ would be more difficult and evaluating q_0 would require more computationally-expensive approaches or approximations. Additionally, the mixing of the unique θ_i values can be quite slow with this algorithm; that is, the probability of a clustered point moving to a new unique group can be quite low, though others have shown that adding a step to resample $\vec{\theta}^*$ can improve this process (for example, West et al. 1994, Bush and MacEachern, 1996). This is a general problem of Polya Urn schemes because, as αq_0 becomes small compared to q_j (often the case in mixtures of DPs), there is a much higher probability of an observation being assigned to previously-existing cluster.

To address these issues, I will explore three alternative methods for simulating $\vec{\theta}$ from $p(\vec{\theta}|\dots)$ as found in the statistics literature. These methods avoid the potentially-complicated integral of q_0, which enable their use in non-conjugate models. We will focus our exploration on the conjugate Model 2, however, to compare the various methods with the standard Gibbs sampling approach of Escobar and West (1995).

4. Simulation Methods

There are many algorithms for simulating marginal posteriors, but here, we will focus on the most prominent ones presented in MacEachern and Müller (1998) and Neal (2000). As motivated in the introduction above above, these methods seek to expand the scope of mixtures of DPs beyond conjugate cases. Specifically, we will explore different methods for simulating $\vec{\theta}$ from $p(\vec{\theta}|\dots)$, as this is the main difference between these algorithms.

In all cases, we sample for 5000 iterations and throw away the first 2500 of those steps as a conservative burnin period. Initial values are random draws from their respective priors. The sampling process for each iteration is as follows:

(i) Update $\vec{\theta}$ from $p(\vec{\theta}|\dots)$ using the chosen method;

(ii) Update ϕ from $p(\phi|a_\phi, b_\phi, \vec{\theta}, \text{data})$;

(iii) Update μ from $p(\mu|a_\mu, b_\mu, \vec{\theta}, \tau^2, \text{data})$;

(iv) Update τ^2 from $p(\tau^2|a_{\tau^2}, b_{\tau^2}, \vec{\theta}, \mu, \text{data})$;

(v) Sample $\eta|\alpha, \text{data}$ to update α using $p(\alpha|\eta, \text{data})$;

(vi) Draw a sample from the posterior predictive density, $p(y_0|\text{data})$ to have as a comparison with the generator distribution.

4.1 No Gaps Method

Presented first in MacEachern and Müller (1998), the authors recognized that the difficulty in evaluating non-conjugate models revolved around the q_0 integral; every time a new cluster θ_j^* is created, we have to integrate over this value, which is computationally inefficient at best. Previous work had developed approximations for q_0, but this paper points out that these approximations can be wildly inaccurate in many cases.

To counteract this, the authors of this paper presented a parameterization of $\vec{\theta}^*$ that considers both full (i.e. those with associated observations) and empty (i.e. those with no associated observations) cluster components. This looks like

$$\theta_F^* = \{\theta_j^*,\ j = 1, \dots, n^*\}$$

with $n_j > 0$ for the "full" clusters and

$$\theta_E^* = \{\theta_j^*,\ j = n^* + 1, \dots, n\}$$

with $n_j = 0$ for the "empty" clusters, where we have the same number of potential clusters as the size of data. In this case, q_0 integrals are no longer necessary as they are replaced with simple likelihood evaluations. As will be the case for the other algorithms we examine, the updating of θ_i is done by changing the s_i indicator values, and then updating the $\theta_j^*|\vec{s}, \text{data}$ values after.

The algorithm for updating the $\vec{\theta}$ is:

(i) For $i = 1, \dots, n$: If s_i is a singleton (i.e. $n_{s_i} = 1$), then leave s_i unchanged with probability $(n^* - 1)/n^*$; otherwise, relabel the clusters such that $s_i = n^*$ and resample with the following probabilities (with $n^{*-} = n^* - 1$):

$$Pr(s_i = j|y_i, \vec{s}-, \vec{\theta}^*) \propto k(y_i|\theta_j^*, \phi) \times \begin{cases} n_j^- & j = 1, \dots, n^{*-} \\ \frac{\alpha}{n^{*-}+1} & j = n^{*-} + 1, \dots, n \end{cases} \tag{10}$$

In the singleton case, if s_i is assigned to $n^{*-} + 1$, then θ_i is unchanged. If s_i is not a singleton (i.e. $n_{s_i} > 1$), then resample s_i with probabilites as in Eq. 10, except with $n^{*-} = n^*$.

(ii) For $j = 1, \dots, n$: Sample $\theta_j^*|\vec{s}, \text{data}$. For $j = n^* + 1, \dots, n$, this is simply $\theta_j^* \sim G_0$. For $j = 1, \dots, n^*$, this is

$$p(\theta_j^*|\vec{s}, n^*, \text{data}) \propto g_0(\theta_j^*|\mu, \tau^2) \prod_{\{i: s_i = j\}} k(y_i|\theta_j^*, \phi).$$

From the initial positions discussed above and the chosen priors, this sampling process tended to converge to the posterior after the first few hundred iterations. An in depth comparison between the posteriors of the various methods will be discussed later.

4.2 No Gaps Method + Auxiliary Components

As is relatively evident in the last method, keeping all the extra, empty θ_j^* values can become computationally inefficient/expensive in memory depending on the application. This is especially true when the number of effective clusters is significantly smaller than the number of observations (e.g. as is the case for the data we are considering in our model). Slightly less obvious is the problem that the No Gaps method has of moving an observation assigned to a cluster to a newly created component; as is pointed out in Neal (2000), this probability is reduce from expected by a factor of $n^{*-}+1$. Ultimately, this means that the No Gaps method can be taxing in terms of computer memory and inefficient in terms of the time it takes to truely sample the posterior.

Neal (2000) developed a similar approach involving auxiliary components to address those concerns, which I term the "No Gaps + Auxiliary Components" method. The key difference is that, instead of a set of "empty" clusters (for θ_j^*, $j = n^* + 1, \ldots, n$), we consider m extra θ_j^* auxiliary components. This new algorithm procedes as:

(i) Define $h = n^{*-} + m$. For $i = 1, \ldots, n$: If s_i is a singleton, relabel the clusters such that $s_i = n^{*-} + 1$, then draw θ_j^* values from G_0 for $j = n^{*-} + 2, \ldots, h$. If s_i is not a singleton, then draw θ_j^* values from G_0 for $j = n^{*-} + 1, \ldots, h$. Set s_i to j with probabilities:

$$Pr(s_i = j | y_i, \vec{s}-, \vec{\theta}^*) \propto k(y_i|\theta_j^*, \phi) \times \begin{cases} \frac{n_j^-}{n-1+\alpha} & j = 1, \ldots, n^{*-} \\ \frac{\alpha/m}{n-1+\alpha} & j = n^{*-} + 1, \ldots, h \end{cases} \tag{11}$$

In the singleton case, if s_i is assigned to $n^{*-} + 1$, then θ_i is unchanged.

(ii) For $j = 1, \ldots, n^*$: Sample $\theta_j^* | \vec{s}$, data.

For our analysis, I choose to consider both $m = 1$ and $m = 3$ to compare the effect of changing the number of auxiliary components. Visually examining the trace plots for the sampler chains show that both chains converge to the posterior much faster than in the No Gaps method above. As before, we will exame the outputs of this algorithm later in the paper.

Considering the limiting cases, if $m = 1$, then this approach becomes extremely similar to the No Gaps method of MacEachern and Müller (1998). The main positive difference is that this method has a greater probability of an observation associated to with a cluster becoming a singleton (and same in reverse). This leads to better mixing, especially when α is small, as is often the case in mixtures of DPs. Additionally, we no longer need to carry all the "empty" cluster vector around in computer memory. This paper makes the argument that the $m = 1$ method should always be used in place of MacEachern and Müller's (1998) No Gaps method. On the other hand, as $m \to \infty$, this algorithm begins to look very similar to a Monte Carlo (MC) approximation of the q_0 integral from the standard Gibbs sampler.

4.3 Metropolis-Hastings Step

The final algorithm we consider is also presented in Neal (2000), and it is a Metropolis-Hastings approach to updating the indicator values. As a quick reminder, the standard Metropolis-Hastings approach uses a proposal distribution to suggest a new value/parameter vector when attempting to sample from a distribution. This proposal, x', is accepted over the current value, x, with probability:

$$a(x', x) = \min\left[1, \frac{c(x|x')}{c(x'|x)} \frac{\pi(x')}{\pi(x)}\right] \tag{12}$$

where c is the proposal distribution, and π is the distribution you wish to sample from; otherwise, the next sample for x does not change from its previous value. This has been invaluable to the study of statistics as it enables simulating from distributions that do not have readily available forms in ways that were not previously possible.

The insight that Neal (2000) gives is to say that the proposal distribution should be set to the prior full conditionals on θ_i (i.e. the familiar Polya Urn representation) such that the acceptance fraction is reduced to evaluating the kernel at the current and proposed positions. Presented in algorithmic form, this becomes:

(i) For $i = 1, \ldots, n$: If s_i is a singleton, draw the propsed indicator, s_i' from $\vec{s}-$ with probabilities $n_j^-/(n-1)$, and set the new s_i to s_i' with probability:

$$a(s_i', s_i) = \min\left[1, \frac{n-1}{\alpha} \frac{k(y_i|\theta_{s_i'}^*, \phi)}{k(y_i|\theta_{s_i}^*, \phi)}\right]. \tag{13}$$

If s_i is not a singleton, let s'_i be a newly created component (i.e $s'_i = n^* + 1$) with $\theta^*_{s'_i}$ drawn from G_0, and set the new s_i to s'_i with probability:

$$a(s'_i, s_i) = \min\left[1, \frac{\alpha}{n-1}\frac{k(y_i|\theta^*_{s'_i})}{k(y_i|\theta^*_{s_i})}\right] \tag{14}$$

(ii) For $i = 1, \ldots, n$: If s_i is a singleton, do not change it's value; otherwise, update s_i to a new value, $s_j \in \vec{s}-$, using the following probabilites:

$$Pr(s_i = s_j|\vec{s}-, y_i, \vec{\theta}^*) \propto \frac{n_j^-}{n-1}k(y_i|\theta^*_{s_j}, \phi) \tag{15}$$

(iii) For $j = 1, \ldots, n^*$: Sample $\theta^*_j|\vec{s}, \text{data}$.

Similar to the No Gaps + Auxiliary Components method, a visual examination of the sample traces indicate the samples converge to the posterior much faster than in the No Gaps method. By eye, it is difficult to determine the difference in convergence speed between this method and the No Gaps + Auxiliary algorithm, but we will examine this more closely in the next section.

5. Comparison of Methods

To begin exploring the differences in the results produced by each algorithm, we look at the posterior distributions on ϕ, μ, τ^2, α, n^*, and θ_0, as shown in Figure 1. θ_0, in this context, is the θ_i associated with a new observation, y_0, as drawn from

$$p(\theta_0|\vec{\theta}, \alpha, \mu, \tau^2) = \frac{1}{\alpha + n}\left[\alpha g_0(\theta_0|\mu, \tau^2) + \sum_{j=1}^{n^*} n_j \delta_{\theta^*_j}(\theta_0)\right] \tag{16}$$

using the updated parameters at each iteration. In general, both the distributions along the top panels and the 95% confidence estimates in the bottom panels are in agreement between the various methods. The most obvious differences are in the α and n^* distributions, specifically in the size of 95% confidence intervals. We will focus the discussion on n^*, as these two parameters are intricately linked and talking about one is akin to studying both.

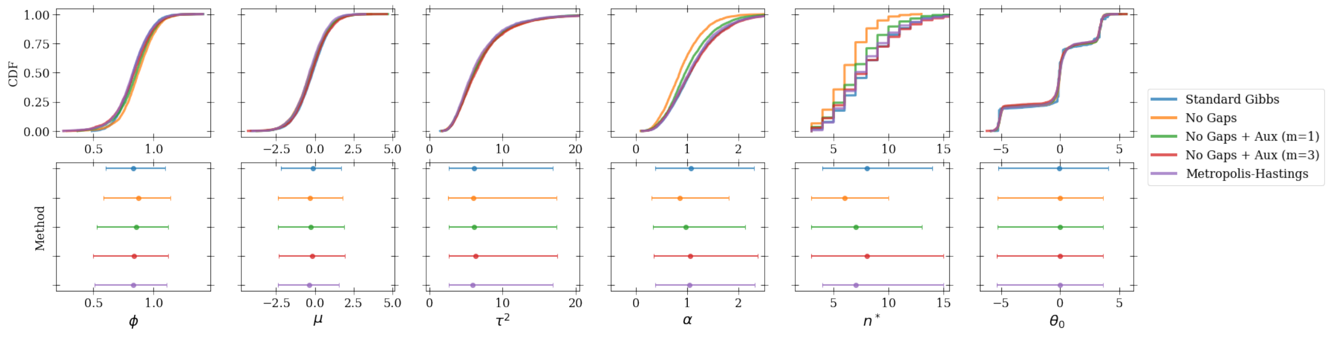
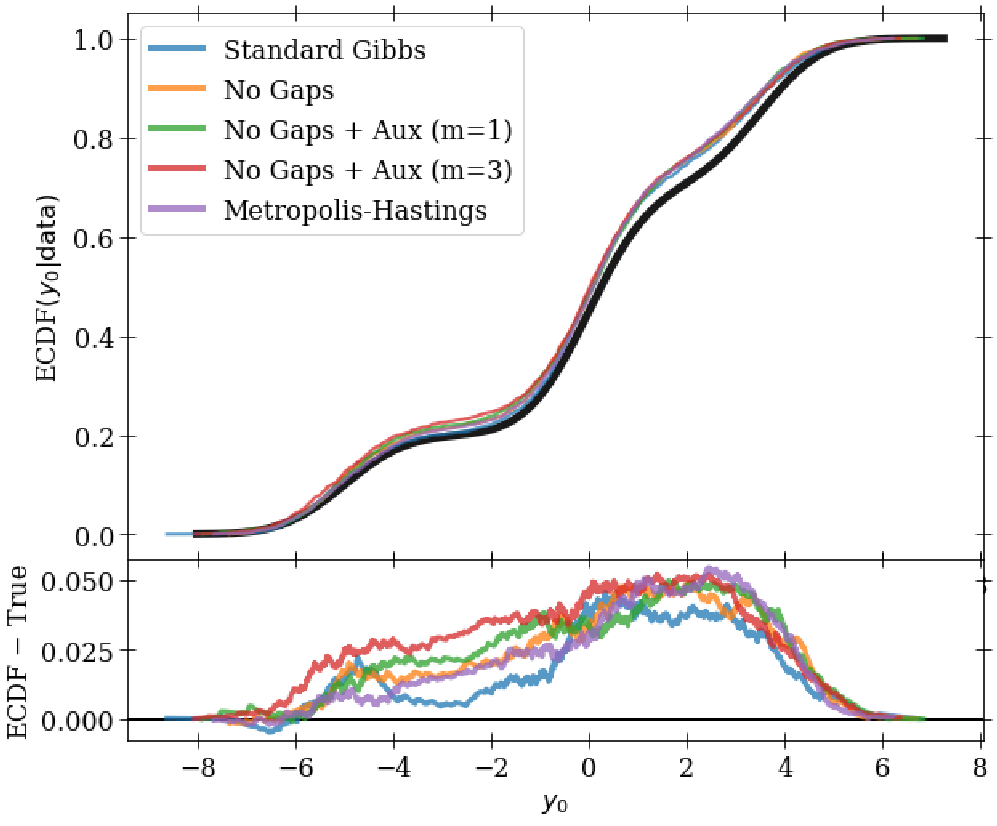


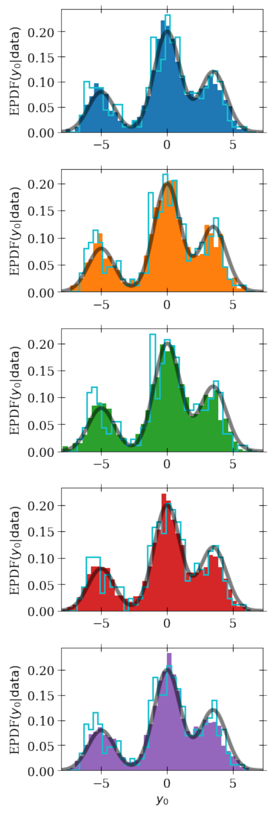
Figure 1: Comparison of posterior estimates for parameters when varying the $\vec{\theta}$ simulation method. The top panels show CDFs for the posterior samples while the lower panels show the median value and 95% confidence interval. Points and lines are coloured by the methods shown in the legend on the right.

The No Gaps method tends to favour fewer unique clusters (i.e. smaller n^*) when compared to the other approaches. This is most likely due to the slower mixing and lower probability of an observation that is currently assigned to a cluster becoming its own new group, as was discussed in previous sections. This would tend to cause data that straddle the boundary between one group and another to be grouped in an existing cluster with higher certainty than is probably true. The other methods, with their generally faster mixing and higher probability of creating singletons, have a greater chance of putting these straddlers into their own groups, thereby increasing the number of unique clusters.

Next, we look at the resulting posterior predictive density, $p(y_0|\text{data})$, that is, the distribution of a new observation; samples from $p(y_0|\text{data})$ at each iteration are drawn from $k(y_0|\theta_0, \phi)$ using the updated parameters. Theoretically, this should match the generator distribution in Eq. 1 if the model perfectly recovered the truth. Empirical CDFs and PDFs for the y_0 samples for the discussed algorithms are shown in Figure 2, where the black line shows the true generator distribution and the coloured lines trace the ECDFs/EPDFs. A residual comparison between the ECDFs and the truth is shown in the lower panel is shown in the bottom left panel. In the right panels, the $n = 250$ datapoints are shown as a light blue histogram.



(a) CDF Summary



(b) PDF Summary

Figure 2: Comparison of the resulting posterior predictive density using the various algorithms. In all panels, black lines represent the data generating distribution. **(a):** In the top panel, the coloured lines show the ECDF generated by the posterior predictive samples. The lower panel shows the difference between the ECDFs and the data generator. **(b)**: Histograms of the $p(y_0|\text{data})$ samples using the same colour scheme as in the CDF plot, with the PDF of the generator distribution also shown in black. The $n = 250$ datapoints are represented with the light blue histograms.

Though the ECDFs and EPDFs generally do a good job of capturing the data generating distribution, they are not perfect. In all cases, the ECDFs from the various algorithms agree quite closely with each other, but they also all tend to have reduced mass around the $y = 3.5$ peak when compared to the truth. This can be partially explained as an effect of the randomly generated data, as the data samples also have a slight deficit in this peak when compared to the generator distribution. Thus it isn't very surprising that the $y_0|$data samples tend to look a little more like the data than the underlying distribution.

One other possible explanation for the differences between the posterior samples and estimates could be due to the values used to initialize the sampler chain. In re-running the analysis with different random seeds, the shapes of the distributions and the posterior estimates for the various parameters tend to vary on the order of the differences shown between the methods. For example, in some runs, the n^* distributions looked much more similar than in Figure 1, while in others there was ever more severe disagreement. Future work should focus on capturing this variability from the initial conditions.

Next, we use two metrics to compare the efficiencies of the algorithms: the physical time for each sampling iteration and the number of iteration steps required to properly sample the posterior. The first is measured as the time it takes for each algorithm to sample the $\vec{\theta}$ values, as this is the only difference between the methods. The second is approximated using the autocorrelation time (in units of iteration steps) for a given parameter; here we focus on α

and n^* (as a parameter of interest) and θ_1 (as an example of $\vec{\theta}$). When estimating the expectation of a quantity using samples, the autocorrelation time is the effective reduction factor in the sample size as compared to samples drawn from the posterior independently. In other words, it measures the number of steps required to have a completely independent draw from the posterior for a given quantity, so smaller autocorrelation times imply that the sampling is more efficient.

We first compare the runtimes of each $\vec{\theta}$ sampling iteration for each method, and the result of this is shown in Figure 3. Coloured histograms show the distribution in the measured sampling time for an iteration, and the errorbar points below the curves show the 68% spread and median values (with the legend stating this value and spread explicity). While the exact values for the sampling time depends on the architecture of the computer than is running the code (and how efficiently the code was written, though we will assume it is close to optimal here), the relative location of the distribution peaks should persist across devices. In this case, I ran this analysis using 2 cores on a 2017 MacBook Pro (16 GB RAM and 3.5 GHz Intel Core i7 Processor). The blue point and histogram show the Standard Gibbs sampling approach (i.e. the one that requires conjugacy of the data kernel and the baseline distribution), and this serves as a baseline to compare the speed of the other algorithms. This comes with the caveat that the other samplers have slightly more optimized code than the Standard Gibbs approach, so the runtimes for this method could be slightly improved in the future.

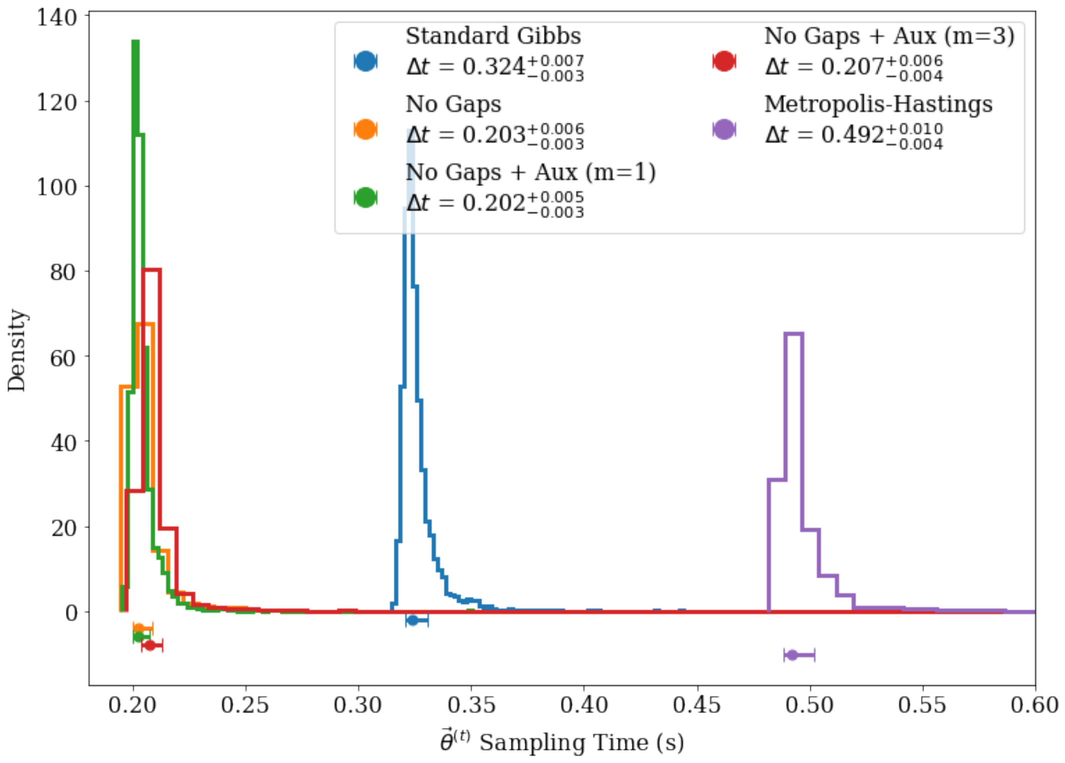


Figure 3: Histograms of the time taken to sample $\vec{\theta}$ at each iteration using the different algorithms. The errorbars below the histograms shown the 68% mass region for each method with the point representing the median. The legend shows the numerical values for the median and 68% mass region for each method.

The time comparison shows us that the Metropolis-Hastings based algorithm is the slowest considered approach. Almost certainly, this is because this method loops over the indicators twice in each iteration, and this explains its approximately $\times 2$ sampling time in comparison to the other techniques. The No Gaps and No Gaps + Auxiliary components methods all have runtimes that are very comparable and are the fastest of the considered techniques. This similarity is in agreement with the fact that these are all very similar algorithms in terms of the calculations they perform; here, the main difference is in the coefficients for the probabilities associated with each cluster. If reduced runtime is the goal of your sampler, then one of the No Gaps + Auxiliary variable methods is a strong option.

Finally, we consider the autocorrelation times for α, n^*, and one repersentative θ_i value. Autocorrelation time is defined as

$$\hat{\tau}_\theta(N) = 1 + 2\sum_{\tau=1}^{N} \hat{\rho}_\theta(\tau)$$

where

$$\hat{\rho}_\theta(\tau) = \frac{\hat{c}_\theta(\tau)}{\hat{c}_\theta(0)}, \quad \text{and} \quad \hat{c}_\theta(\tau) = \frac{1}{N-\tau}\sum_{n=1}^{N-\tau}(\theta^{(n)} - \mu_\theta)(\theta^{(n+\tau)} - \mu_\theta)$$

and μ_θ is the expectation of the $\theta^{(i)}$, $i = 1, \ldots, N$ samples. As stated above, this metric is a proxy for the number of iterations it takes for the sampler to forget its current position or to independently sample from the posterior. By this metric, we desire as small values as possible. The autocorrelation time for the various methods are shown in Table 2.

Table 2: Comparison of the $\vec{\theta}$ sampling time (in seconds) and autocorrelation time (in number of iteration steps) for the various methods. Bolded values highlight the best value for that metric.

Method	**Median time per $\vec{\theta}$ iteration (s)**	**Autocorrelation time for α (steps)**	**Autocorrelation time for n^* (steps)**	**Autocorrelation time for θ_1 (steps)**
Standard Gibbs	0.324	41.5	85.6	2.15
No Gaps	0.203	44.5	56.4	2.16
No Gaps + Auxiliary ($m = 1$)	**0.202**	**15.0**	**23.7**	1.68
No Gaps + Auxiliary ($m = 3$)	0.207	15.5	26.2	**1.54**
Metropolis-Hastings	0.492	23.4	44.7	1.94

As expected, the No Gaps + Auxiliary with $m = 1$ method performs similarly to the No Gaps method, except with significantly quicker independent sampling from the posterior. The No Gaps + Auxiliary with $m = 3$ is very similar to the same method with $m = 1$, and these two analyses tend to have the quickest runtimes and smallest autocorrelation times for all parameters (including the ones not shown in this table). Future work should explore the effect that m has on the runtimes and autocorrelation times, though this will likely also be a function of the model and data being considered. The Metropolis-Hastings approach, while slower than the No Gaps + Auxiliary approaches, produces autocorrelation times that are significantly faster than the No Gaps or Standard Gibbs approaches; depending on the particular problem being studied, this can more than make up for the reduced computation speed.

6. Conclusion

From the results of this exploration into three methods for simulating marginal posterior distributions – methods that can be extended into non-conjugate cases – I have shown that each method presents different costs and benifits. In general, the No Gaps + Auxiliary method is superior to the No Gaps method, and this is likely due to increased θ^* mixing when the auxiliary components are considered. In most applications and by many metrics, the Metropolis Approach is also superior to the No Gaps approach, but it comes at the cost of slower runtimes.

In summary, the No Gaps + Auxiliary method should be used whenever possible, but there may be particular applications in which using the Metropolis-Hastings algorithm is more desirable. Future work should explore how the m parameter of the No Gaps + Auxiliary method correlates with runtimes and autocorrelation times and also how the starting position of the sampler impacts the resulting posterior samples.

References

1 Escobar, M.D., and West, M. (1995): "Bayesian Density Estimation and Inference Using Mixtures," Journal of the American Statistical Association, 90, 577-588

2 MacEachern, S.N., and Müller, P. (1998): “Estimating Mixture of Dirichlet Process Models,” Jounal of Computational and Graphical Statistics, 7, 223-238

3 Neal, R.M. (2000), “Markov Chain Sampling Methods for Dirichlet Process Mixture Models” Journal of Computational and Graphical Statistics, 9, 249-265