Biometrika (2013), **100**, 4, *pp.* 817–830
Printed in Great Britain
doi: 10.1093/biomet/ast024
Advance Access publication 11 July 2013

Practical perfect sampling using composite bounding chains: the Dirichlet-multinomial model

By NATHAN M. STEIN and XIAO-LI MENG

Department of Statistics, Harvard University, Cambridge, Massachusetts 02138, U.S.A.
nmstein@post.harvard.edu meng@stat.harvard.edu

Summary

A discrete data augmentation scheme together with two different parameterizations yields two Gibbs samplers for sampling from the posterior distribution of the hyperparameters of the Dirichlet-multinomial hierarchical model under a default prior distribution. The finite-state space nature of this data augmentation permits us to construct two perfect samplers using bounding chains that take advantage of monotonicity and anti-monotonicity in the target posterior distribution, but both are impractically slow. We demonstrate that a composite algorithm that strategically alternates between the two samplers' updates can be substantially faster than either individually. The speed gains come because the composite algorithm takes a divide-and-conquer approach in which one update quickly shrinks the bounding set for the augmented data, and the other update immediately coalesces on the parameter, once the augmented-data bounding set is a singleton. We theoretically bound the expected time until coalescence for the composite algorithm, and show via simulation that the theoretical bounds can be close to actual performance.

Some key words: Anti-monotonicity; Bounding chain; Coupling from the past; Data augmentation; Markov chain Monte Carlo; Monotonicity; Perfect sampling.

1. The Dirichlet-multinomial model

The multinomial model and its conjugate Dirichlet prior distribution are common building blocks of more elaborate models for categorical data, with applications from topic modelling (Blei et al., 2003) to biology (Holmes et al., 2012). Despite the model's popularity, there is room to improve algorithms for Bayesian inference. Although fast Newton–Raphson iterations can find maximum likelihood estimates, as demonstrated in the 2003 Microsoft Resarch technical report by T. P. Minka, 'Estimating a Dirichlet distribution,' sampling from the posterior distribution of the parameters in a Bayesian setting is more challenging, especially in high dimensions.

In this paper we present a data augmentation scheme that yields a practical Gibbs sampler and facilitates perfect sampling algorithms based on composite bounding chains. The central idea of the composite perfect sampling algorithm is to strategically combine two or more bounding chains, such that the composite chain has a much faster coalescence time than the individual samplers. Our main theoretical result is a bound on the expected running time of the composite algorithm, and we prove it using a somewhat general language about bounding chains, suggesting that similar speed gains may be possible for other bounding chain algorithms. The dramatic increase in speed we achieve is a small but encouraging step on the road toward perfect samplers that can be routinely used in practice for Bayesian computation.

Let y be an $N \times k$ matrix of observed counts with ith row $y_i^{\mathrm{T}} = (y_{i1}, \ldots, y_{ik})$, and let μ be an $N \times k$ matrix with ith row $\mu_i^{\mathrm{T}} = (\mu_{i1}, \ldots, \mu_{ik})$, a k-dimensional probability vector with $\mu_{ij} \geqslant 0$, $\sum_{j=1}^{k} \mu_{ij} = 1$. Conditioned on μ, the vectors y_i are independent multinomial random variables with probabilities μ_i and fixed sample sizes n_i:

$$\mathrm{pr}(y_i \mid \mu) = \frac{n_i!}{\prod_{j=1}^{k} y_{ij}!} \prod_{j=1}^{k} \mu_{ij}^{y_{ij}} \quad (i = 1, \ldots, N).$$

The probabilities μ_i in turn are independent draws from a $\mathrm{Dir}(\alpha_1, \ldots, \alpha_k)$ distribution:

$$p(\mu_i \mid \alpha_1, \ldots, \alpha_k) = \frac{\Gamma(\sum_{j=1}^{k} \alpha_j)}{\prod_{j=1}^{k} \Gamma(\alpha_j)} \prod_{j=1}^{k} \mu_{ij}^{\alpha_j - 1} \quad (i = 1, \ldots, N).$$

Integrating over μ, the posterior of the hyperparameters under a prior $\pi(\alpha_1, \ldots, \alpha_k)$ is

$$p(\alpha_1, \ldots, \alpha_k \mid y) \propto \pi(\alpha_1, \ldots, \alpha_k) \prod_{i=1}^{N} \left\{ \frac{\Gamma(\sum_{j=1}^{k} \alpha_j)}{\Gamma(\sum_{j=1}^{k} \alpha_j + n_i)} \prod_{j=1}^{k} \frac{\Gamma(\alpha_j + y_{ij})}{\Gamma(\alpha_j)} \right\}. \tag{1}$$

Constructing a practical perfect sampler for (1) is the main subject of this paper.

2. A DISCRETE DATA AUGMENTATION STRATEGY AND TWO GIBBS SAMPLERS

For reasons that will soon be clear, we shall use both the parameterization $\alpha = (\alpha_1, \ldots, \alpha_k)$ and the parameterization $\theta = (\omega, \lambda)$, where $\omega = \sum_{j=1}^{k} \alpha_j$ is a concentration parameter and $\lambda = (\lambda_1, \ldots, \lambda_k)$ is a mean vector with $\lambda_j = \alpha_j / \omega$. Throughout this paper, we assume that the prior distribution on λ and ω factors into independent priors

$$\lambda \sim \mathrm{Dir}(\delta_1, \ldots, \delta_k), \quad \omega \sim \pi_0. \tag{2}$$

Conditioning on the observed data y and the parameter θ, we construct our data augmentation scheme. If $y_{ij} > 0$, we define conditionally independent Bernoulli random variables $v_{ij} = (v_{ij,1}, \ldots, v_{ij,y_{ij}})$ with

$$\mathrm{pr}(v_{ij,m} = 1 \mid y, \theta) = \frac{\omega \lambda_j}{\omega \lambda_j + m - 1} \quad (m = 1, \ldots, y_{ij});$$

and if $y_{ij} = 0$, then v_{ij} is defined as empty.

This data augmentation strategy has a nice interpretation in terms of a double-replacement sampling scheme. The distribution of y given θ, integrating over μ, is equivalent to supposing that there are N urns that initially have α_j balls of colour j for $j = 1, \ldots, k$, and we draw balls independently from each urn following a double-replacement scheme. That is, when we draw a ball of colour j, we replace it and add another ball of the same colour before sampling the next ball. The observation y_{ij} is the number of balls sampled from urn i with colour j, and the data augmentation $v_{ij,m}$ is the indicator for the mth ($m = 1, \ldots, y_{ij}$) sampled ball whether it was drawn from the original pool, in which case $v_{ij,m} = 1$, or from the balls that were added to the urns through the double-replacement scheme, in which case $v_{ij,m} = 0$. Thus, it makes sense that if $y_{ij} = 0$, then v_{ij} is empty because there is nothing to indicate, and if $y_{ij} > 0$, then $v_{ij,1} = 1$ because the first sampled ball must have come from the original pool.

Letting $v = \{v_{ij}\}_{i,j}$, we then obtain the complete-data likelihood $p(y, v \mid \theta)$ as the product $p(v \mid y, \theta)p(y \mid \theta)$, clearly preserving the margin of interest $p(y \mid \theta)$. Since $\Gamma(x+1) = x\Gamma(x)$,

$$p(y, v \mid \theta) = \prod_{i=1}^{N} \frac{\Gamma(\omega)}{\Gamma(\omega + n_i)} \prod_{j=1}^{k} \left\{ \prod_{m=1}^{y_{ij}} (\omega\lambda_j)^{v_{ij,m}} (m-1)^{1-v_{ij,m}} \right\}, \tag{3}$$

where we take the term in braces equal to 1 if $y_{ij} = 0$. If the mean and concentration parameters are independent in their prior distribution, which we will assume throughout, then they will also be independent in their complete-data posterior distribution, since terms involving ω and λ factor in (3). More intuition about (3) appears at the end of this section.

This data augmentation scheme yields a Gibbs sampler that is easy to implement. Denoting the sufficient statistics of the augmented data $z_j = \sum_{i=1}^{N} \sum_{m=1}^{y_{ij}} v_{ij,m}$, where we take $\sum_{m=1}^{y_{ij}} v_{ij,m} = 0$ if $y_{ij} = 0$, we transition from (z, θ) to (z', θ') by alternating between updating the parameters given the complete data by drawing θ' from

$$p(\theta' \mid z, y) \propto \pi(\omega', \lambda')\, \omega'^{\sum_{j=1}^{k} z_j} \left\{ \prod_{i=1}^{N} \frac{\Gamma(\omega')}{\Gamma(\omega' + n_i)} \right\} \left\{ \prod_{j=1}^{k} \lambda_j'^{z_j} \right\} \tag{4}$$

and updating the missing data given the parameters by drawing independently, for $j = 1, \ldots, k$,

$$(z_j' \mid \theta', y) \sim \sum_{i=1}^{N} \sum_{m=1}^{y_{ij}} \mathrm{Ber}\left(\frac{\omega'\lambda_j'}{\omega'\lambda_j' + m - 1} \right). \tag{5}$$

We will call equations (4)–(5) the standard Gibbs sampler. The discreteness of $z = (z_1, \ldots, z_k)$ is a major advantage of this algorithm, as it enables perfect samplers to coalesce in finite time. Another benefit of this algorithm is that using a Dirichlet prior distribution on λ leads to a conjugate Dirichlet update for the multivariate λ, and the only step that requires special attention is sampling the univariate ω, for which many standard methods are available, including grid-based and rejection methods. Additionally, the density $p(\omega \mid z, y)$ arises in the context of Dirichlet process mixture models, and Escobar & West (1995) suggest a Gibbs sampler for this distribution.

Following Craiu & Meng (2011), we say that z' given θ' and y in (5) has a nonhomogeneous binomial distribution. In general, a nonhomogeneous binomial random variable $x \sim \mathrm{NhBin}\{N; (p_1, \ldots, p_N)\}$ can be represented as the sum of N independent Bernoulli random variables $b_i \sim \mathrm{Ber}(p_i)$, each with its own success probability p_i. The nonhomogeneous binomial distribution can be easily generalized to the nonhomogeneous multinomial distribution, which is familiar from the traditional data augmentation approach to fitting finite mixture models. To illustrate this, suppose data $y = (y_1, \ldots, y_N)$ are independent and identically distributed according to the mixture model

$$p(y_i \mid \eta) = \sum_{j=1}^{k} \eta_j f_j(y_i), \tag{6}$$

where for simplicity we can suppose that the densities $f_j(\cdot)$ are fully known and the mixture weights $\eta = (\eta_1, \ldots, \eta_k)$ are of interest. The usual model-fitting approach augments y with a nonhomogeneous multinomial variable $z = (z_1, \ldots, z_N)$ such that

$$\mathrm{pr}(z_i = j \mid y, \eta) = \frac{\eta_j f_j(y_i)}{\eta_1 f_1(y_i) + \cdots + \eta_k f_k(y_i)} \quad (j = 1, \ldots, k).$$

The complete-data likelihood is therefore $p(y, z \mid \eta) = \prod_{i=1}^{N} \prod_{j=1}^{k} \{\eta_j f_j(y_i)\}^{1(z_i=j)}$, which as a product can lead to straightforward Gibbs sampling and expectation-maximization algorithms that are much easier to work with than the sums in (6). See Hobert et al. (1999), Murdoch & Meng (2001), Casella et al. (2002), and Mukhopadhyay & Bhattacharya (2012) for examples in the context of perfect sampling. The nonhomogeneous binomial augmentation in the Dirichlet-multinomial model plays the same role as in finite mixture models: it turns sums into products. In the Dirichlet-multinomial model, we can rewrite (1) as

$$p(\omega, \lambda \mid y) \propto \left\{ \pi(\omega, \lambda) \prod_{i=1}^{N} \frac{\Gamma(\omega)}{\Gamma(\omega + n_i)} \right\} \left\{ \prod_{i=1}^{N} \prod_{j=1}^{k} \prod_{m=1}^{y_{ij}} (\omega\lambda_j + m - 1) \right\}. \tag{7}$$

The z augmentation turns the sums $\omega\lambda_j + (m-1)$ on the right-hand side of (7) into products as shown in (3), which are much more convenient for sampling.

Before developing our perfect samplers, it is helpful to introduce another Gibbs sampler based on this same data augmentation strategy. However, instead of using the θ parameterization, we use the original α parameterization and alternate between draws from

$$p(\alpha'_j \mid \alpha_{[-j]}, z, y), \quad p(z'_j \mid \alpha'_j, \alpha_{[-j]}, z_{[-j]}, y) \quad (j = 1, \ldots, k), \tag{8}$$

where $\alpha_{[-j]} = (\alpha_1, \ldots, \alpha_{j-1}, \alpha_{j+1}, \ldots, \alpha_k)$ and $z_{[-j]}$ is similarly defined. We will see in § 3·2 that while this algorithm sacrifices the factorization between the multivariate λ density and the univariate ω density, it offers convenient monotonicity properties that help in designing bounding chains.

3. PERFECT SAMPLING USING BOUNDING CHAINS

3·1. *Bounding chains*

Propp & Wilson (1996) introduced coupling from the past to obtain exact draws from the stationary distribution of a Markov chain in finite time. The key idea is to run coupled Markov chains from the past to the present with the same transition probabilities but starting from different states. The construction guarantees that if the chains couple by time 0, then the value at time 0 is an exact draw from the stationary distribution of the chains. Propp & Wilson (1996) recognized that the computation is much simpler when the updates of the chain are monotone with respect to some partial order on the state space.

In general, it can be difficult to construct monotone chains. Bounding chains were therefore introduced in Huber (1998) and Häggström & Nelander (1999) and developed in Huber (2004), among others. They provide a solution for perfect sampling without requiring the monotonicity used by Propp & Wilson (1996). A bounding chain for a Markov chain x_t on a state space Ω is a set-valued Markov chain X_t on the set of all subsets of Ω, such that the current state $x_t \in X_t$, for every starting value that could have been used for the x_t chain; see Huber (2004) for discussion and a slightly more general definition. When bounding chains are used in the context of coupling from the past, coalescence is detected and the algorithm returns a draw from the stationary distribution when X_0 is a singleton.

Specifically, if the original x_t chain can be written as a stochastic recursive sequence $x_{t+1} = \phi(x_t, u_t)$, where ϕ is a deterministic function and u_t is a random input, then the set-valued chain

$$X_{t+1} = \Phi(X_t, u_t) \tag{9}$$

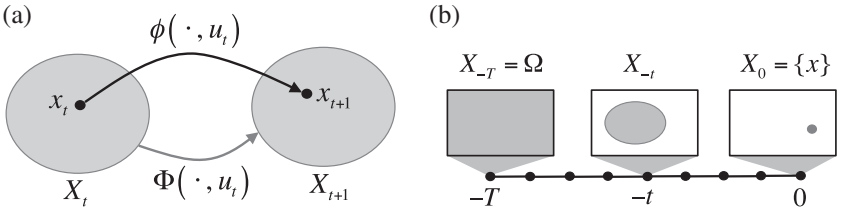

Fig. 1. (a) The relationship of the bounding chain $\Phi(\cdot, u_t)$ to the underlying Markov chain $\phi(\cdot, u_t)$. (b) Coupling from the past using bounding chains.

is a valid bounding chain if $\phi(x_t, u_t) \in \Phi(X_t, u_t)$ for every $x_t \in X_t$, as illustrated in Fig. 1(a). Coupling from the past for this bounding chain proceeds as follows: first, set $T = T_0$ for some fixed T_0 such as 1, and $X_{-T} = \Omega$. Then, run the chain forward from X_{-T} by repeatedly calling (9) until we obtain X_0, as illustrated in Fig. 1(b). If X_0 is a singleton $\{x\}$, then x is an exact draw from the stationary distribution of the chain defined by ϕ. Otherwise, set $T_{\text{old}} = T$, set $T = T_{\text{new}} > T_{\text{old}}$, where typically $T_{\text{new}} = 2T_{\text{old}}$, and repeat the procedure, drawing new random inputs $u_{-T}, \ldots, u_{-T_{\text{old}}-1}$ and reusing the sequence $u_{-T_{\text{old}}}, \ldots, u_{-1}$.

To define the bounding sets used in our perfect samplers, we use the partial orders $z \preceq \tilde{z}$ when $z_j \leqslant \tilde{z}_j$, and $\alpha \preceq \tilde{\alpha}$ when $\alpha_j \leqslant \tilde{\alpha}_j$, for all $j = 1, \ldots, k$. We say that $\theta \preceq \tilde{\theta}$ when $\alpha \preceq \tilde{\alpha}$, and $(z, \theta) \preceq (\tilde{z}, \tilde{\theta})$ when $z \preceq \tilde{z}$ and $\theta \preceq \tilde{\theta}$. These partial orders admit a minimum and maximum state for z, but only a minimum state for α:

$$z^{\min} = \left(\sum_{i=1}^{N} 1(y_{i1} > 0), \ldots, \sum_{i=1}^{N} 1(y_{ik} > 0) \right), \quad z^{\max} = \sum_{i=1}^{N} y_i, \quad \alpha^{\min} = (0, \ldots, 0).$$

It may seem surprising that $z^{\min}$ is not 0, but recall that if $y_{ij} > 0$, then $v_{ij,1} = 1$, so $z_j^{\min}$ is the number of nonzero entries in the jth column of y. These partial orders allow us to construct bounding sets, the details of which appear below in the context of the two algorithms we use.

3·2. *Componentwise algorithm*

For our first algorithm, we work with the parameterization $(\alpha_1, \ldots, \alpha_k)$ instead of (ω, λ). To guarantee the necessary monotonicity, it is sufficient for the prior density on ω to satisfy the following property.

Property 1. The ratio

$$\frac{\pi_0(\alpha_j + \tilde{s}_j)}{\pi_0(\alpha_j + s_j)} \left(\frac{\alpha_j + \tilde{s}_j}{\alpha_j + s_j} \right)^{1 - \sum_{j=1}^{k} \delta_j}$$

is increasing in α_j whenever $\tilde{s}_j \geqslant s_j$, where $\delta_1, \ldots, \delta_k$ are the Dirichlet parameters in (2).

From the conditional density

$$p(\alpha_j \mid y, z, \alpha_{[-j]}) \propto \pi_0 \left(\alpha_j + s_j\right) \left(\alpha_j + s_j\right)^{1 - \sum_{j=1}^{k} \delta_j} \alpha_j^{\delta_j + z_j - 1} \prod_{i=1}^{N} \frac{\Gamma\left(\alpha_j + s_j\right)}{\Gamma(\alpha_j + s_j + n_i)}, \quad (10)$$

where $s_j = \sum_{\ell \neq j} \alpha_\ell$, we can show that for any prior satisfying Property 1, the conditional density (10) will enable monotone updates of α_j. One such prior is $\omega \sim \text{Ga}(b_0, b_1)$ for $0 < b_0 \leqslant \sum_{j=1}^{k} \delta_j$ and any positive b_1.

The following lemma, which is a slightly more general version of Lemma 1 in Møller (1999), can be used to demonstrate why Property 1 enables monotone updates for α_j.

LEMMA 1. *Suppose X and Y are univariate random variables with densities f and g, respectively, with respect to the same measure ν. Let $S_X = \{x : f(x) > 0\}$ and $S_Y = \{x : g(x) > 0\}$. If the function h defined below is increasing for $x \in S_X \cup S_Y$,*

$$h(x) = \begin{cases} 0, & (x \in S_X^C \cap S_Y), \\ f(x)/g(x), & (x \in S_X \cap S_Y), \\ \infty, & (x \in S_X \cap S_Y^C), \end{cases}$$

then X stochastically dominates Y, that is, $\mathrm{pr}(X \leqslant c) \leqslant \mathrm{pr}(Y \leqslant c)$ for all c.

Proof. For $h(x)$ to be increasing in x, it must be true that if $x_1 \in S_X^C \cap S_Y$, $x_2 \in S_X \cap S_Y$, and $x_3 \in S_X \cap S_Y^C$, then $x_1 < x_2 < x_3$. Therefore, if $c \in S_X^C \cap S_Y$, then $\mathrm{pr}(X \leqslant c) = 0 \leqslant \mathrm{pr}(Y \leqslant c)$. Similarly, if $c \in S_X \cap S_Y^C$, then $\mathrm{pr}(Y \leqslant c) = 1 \geqslant \mathrm{pr}(X \leqslant c)$.

Now consider the case $c \in S_X \cap S_Y$. Let $A = \{X \in S_Y\}$. Since $1(Y \leqslant c)$ is decreasing in Y and $h(Y)$ is increasing,

$$0 \geqslant \mathrm{cov}\{1(Y \leqslant c), h(Y)\} = \mathrm{pr}(\{X \leqslant c\} \cap A) - \mathrm{pr}(Y \leqslant c)\mathrm{pr}(A),$$

whence $\mathrm{pr}(\{X \leqslant c\} \cap A) \leqslant \mathrm{pr}(Y \leqslant c)$. But $\mathrm{pr}(\{X \leqslant c\} \cap A^C) = 0$, since if $c \in S_X \cap S_Y$ and $x \in S_X \cap S_Y^C$, then $x > c$. Thus, $\mathrm{pr}(X \leqslant c) \leqslant \mathrm{pr}(Y \leqslant c)$ for all c. □

If $\tilde{\alpha}_{[-j]} \succeq \alpha_{[-j]}$ and $\tilde{z} \succeq z$, then the conditional distribution of α_j given $(y, \tilde{z}, \tilde{\alpha}_{[-j]})$ will stochastically dominate the distribution of α_j given $(y, z, \alpha_{[-j]})$. This enables a monotone update using inverse transform sampling by coupling upper and lower chains, which serve as upper and lower bounds for our bounding sets, with the same uniform random input. We can update z given (α, y) by drawing $\mathrm{Un}(0, 1)$ random variables u_{im} for $i = 1, \ldots, N$ and $m = 1, \ldots, y_{ij} - 1$, and then setting

$$z_j = \sum_{i=1}^{N} 1(y_{ij} > 0) \left\{ 1 + \sum_{m=1}^{y_{ij}-1} 1\left(u_{im} \leqslant \frac{\alpha_j}{\alpha_j + m} \right) \right\}. \tag{11}$$

To update $\tilde{z}_j$, we use the same random inputs u but replace α_j by $\tilde{\alpha}_j$, which makes it clear that if $\tilde{\alpha}_j \geqslant \alpha_j$, then the updated $\tilde{z}_j$ satisfies $\tilde{z}_j \geqslant z_j$.

It is natural to ask why any other algorithms are needed, since perfect sampling using monotone coupling from the past is well established. Two problems make this algorithm impractical. First, since the update for α_j conditions on $\alpha_{[-j]}$, this is not a two-step Gibbs sampler, and hence the marginal sequence $\{z_t : t = 1, \ldots\}$ does not form a Markov chain, where t indexes iterations rather than components. Therefore, if we checked only the coalescence on z, we would ignore the possibility that θ may be different in the lower and upper chains, which can in future steps allow z to uncoalesce. Unfortunately, updating α_j via inverse transform sampling guarantees that θ will almost surely never coalesce, because $\mathrm{pr}(\alpha_j^{\mathrm{U}} > \alpha_j^{\mathrm{L}}) = 1$ at every time t, where U and L denote respectively the upper and lower chains; a similar situation occurs in Møller (1999).

Second, to make the problem worse, because there is no natural maximum state for α, we do not even have a method to draw the initial values α^{U} and α^{L}, conditioning on $z = z^{\max}$ in the upper chain and $z = z^{\min}$ in the lower chain respectively, that would guarantee $\alpha^{\mathrm{U}}_{[-j]} \succeq \alpha^{\mathrm{L}}_{[-j]}$ for all j. The coordinate-wise approach using (10) cannot initialize the full parameter vector with this guarantee because it only updates each α_j conditional on all the other values $\alpha_{[-j]}$.

3·3. *Vector algorithm*

Our vector algorithm solves both problems above, but introduces a new one. It first updates the entire parameter vector given the complete data, and then updates the augmented data given the parameter. All random draws in the upper and lower chains are coupled by using probability integral transform sampling with the same random quantiles. First, we draw

$$\omega^{\mathrm{L}} \sim p(\omega \mid y, z^{\mathrm{L}}), \quad \omega^{\mathrm{U}} \sim p(\omega \mid y, z^{\mathrm{U}}). \tag{12}$$

Here, the actual implementation of (12) will depend on the choice of π_0, which we assume satisfies Property 1. Then, for $j = 1, \ldots, k$, we draw

$$\gamma_j^{\mathrm{L}} \sim \mathrm{Ga}(\delta_j + z_j^{\mathrm{L}}, 1), \quad \gamma_j^{\mathrm{U}} \sim \mathrm{Ga}(\delta_j + z_j^{\mathrm{U}}, 1). \tag{13}$$

If we were to set $\alpha_j^{\mathrm{L}} = \omega^{\mathrm{L}}\gamma_j^{\mathrm{L}} / \sum_{\ell=1}^k \gamma_\ell^{\mathrm{L}}$ and $\alpha_j^{\mathrm{U}} = \omega^{\mathrm{U}}\gamma_j^{\mathrm{U}} / \sum_{\ell=1}^k \gamma_\ell^{\mathrm{U}}$, these would be valid draws from the complete-data posterior distributions $p(\alpha \mid y, z^{\mathrm{L}})$ and $p(\alpha \mid y, z^{\mathrm{U}})$, as given in (4). Unfortunately, this would not guarantee that $\alpha^{\mathrm{L}} \preceq \alpha^{\mathrm{U}}$. Even if $\gamma_j^{\mathrm{L}} \leqslant \gamma_j^{\mathrm{U}}$ for all j, it is possible for $\gamma_j^{\mathrm{L}} / \sum_{\ell=1}^k \gamma_\ell^{\mathrm{L}} > \gamma_j^{\mathrm{U}} / \sum_{\ell=1}^k \gamma_\ell^{\mathrm{U}}$ for some j, and the difference may be large enough that $\omega^{\mathrm{L}}\gamma_j^{\mathrm{L}} / \sum_{\ell=1}^k \gamma_\ell^{\mathrm{L}} > \omega^{\mathrm{U}}\gamma_j^{\mathrm{U}} / \sum_{\ell=1}^k \gamma_\ell^{\mathrm{U}}$, even though $\omega^{\mathrm{L}} < \omega^{\mathrm{U}}$.

However, we can achieve a valid bounding chain algorithm that preserves the order $\alpha^{\mathrm{L}} \preceq \alpha^{\mathrm{U}}$ by dividing by the sum of γ_ℓ in the opposite chain:

$$\alpha_j^{\mathrm{L}} = \frac{\omega^{\mathrm{L}}\gamma_j^{\mathrm{L}}}{\sum_{\ell=1}^k \gamma_\ell^{\mathrm{U}}}, \quad \alpha_j^{\mathrm{U}} = \frac{\omega^{\mathrm{U}}\gamma_j^{\mathrm{U}}}{\sum_{\ell=1}^k \gamma_\ell^{\mathrm{L}}}. \tag{14}$$

If $z^{\mathrm{L}} = z^{\mathrm{U}}$ immediately before this parameter update, then $\alpha_j^{\mathrm{L}} = \alpha_j^{\mathrm{U}}$ for all $j = 1, \ldots, k$, so that it is possible to coalesce on z and α.

To see that alternating between updating α using (12)–(14) and then updating z given α and y yields a valid bounding chain algorithm, suppose $(z_t, \theta_t) \in Z_t \times \Theta_t$, where $Z_t = \{z : z_t^{\mathrm{L}} \preceq z \preceq z_t^{\mathrm{U}}\}$ and $\Theta_t = \{\theta : \theta_t^{\mathrm{L}} \preceq \theta \preceq \theta_t^{\mathrm{U}}\}$. Then, we can draw $\omega \sim p(\omega \mid y, z_t)$ and $\gamma_j \sim \mathrm{Ga}(\delta_j + z_{t,j}, 1)$, where $z_{t,j}$ is the jth component of z at time t, and draw $\omega^{\mathrm{L}}, \omega^{\mathrm{U}}, \gamma^{\mathrm{L}}, \gamma^{\mathrm{U}}$ using (12) and (13), guaranteeing that $\omega^{\mathrm{L}} \leqslant \omega \leqslant \omega^{\mathrm{U}}$ and $\gamma_j^{\mathrm{L}} \leqslant \gamma_j \leqslant \gamma_j^{\mathrm{U}}$. Therefore,

$$\frac{\omega^{\mathrm{L}}\gamma_j^{\mathrm{L}}}{\sum_{\ell=1}^k \gamma_\ell^{\mathrm{U}}} \leqslant \frac{\omega\gamma_j}{\sum_{\ell=1}^k \gamma_\ell} \leqslant \frac{\omega^{\mathrm{U}}\gamma_j^{\mathrm{U}}}{\sum_{\ell=1}^k \gamma_\ell^{\mathrm{L}}},$$

so that $\theta_{t+1} \in \Theta_{t+1}$. The z draw (11) ensures that $z_{t+1} \in Z_{t+1}$.

The new problem is that this bounding chain is loose, making it impractical even on low-dimensional, low-count datasets; see § 5. However, strategically alternating between the vector algorithm and the componentwise algorithm can yield a much speedier composite algorithm, as we demonstrate empirically in § 5, after a theoretical investigation in § 4.

4. THEORETICAL RESULTS: COMPOSITE BOUNDING CHAINS

Suppose we wish to sample from $p(\theta \mid y)$, and we have an augmented-data model $p(z, \theta \mid y)$. To detect coalescence, it is helpful for z to be discrete, but see Murdoch & Green (1998) for perfect samplers on continuous state spaces. We assume two underlying Markov chains with stochastic recursive sequences $x_{t+1} = \phi(x_t, u_t)$ and $x_{t+1} = \psi(x_t, v_t)$, where u_t and v_t are random inputs and $x_t = (z_t, \theta_t)$. We use bounding chains $X_t \ni x_t$ with associated stochastic recursive sequences $X_{t+1} = \Phi(X_t, u_t)$ and $X_{t+1} = \Psi(X_t, v_t)$. We assume that the bounding sets can be written as $X_t = Z_t \times \Theta_t$, with $z_t \in Z_t$ and $\theta_t \in \Theta_t$.

Since our goal is to draw samples of θ, a perfect sampler will not terminate before Θ_t is a singleton. Thus, if Φ can quickly reduce Z_t to a singleton and if Ψ can reduce Θ_t to a singleton once Z_t is a singleton, then alternating between Φ and Ψ can be faster than either individually. To take an extreme case, an alternating algorithm can coalesce on (z, θ) even if Φ can never coalesce on θ and Ψ can never coalesce on z.

Our composite algorithm alternates between $(M-1)$-fold compositions $\Phi \circ \cdots \circ \Phi = \Phi^{M-1}$ for some pre-chosen $M > 1$, and single instances of Ψ; more general combination strategies are of course possible. That is, in notation, the composite stochastic recursion function is $\Psi \circ \Phi^{M-1}$.

A stochastic recursion function ψ is monotone if and only if $x \preceq \tilde{x}$ implies $\psi(x, v) \preceq \psi(\tilde{x}, v)$ for all random inputs v. For bounding chains, we can use the subset relationship as a partial order. That is, a bounding chain $\mathcal{B}$ is monotone if for all its random inputs v, $\mathcal{B}(X, v) \subseteq \mathcal{B}(\tilde{X}, v)$ whenever $X \subseteq \tilde{X}$.

LEMMA 2. *If we let Φ denote the componentwise algorithm of § 3·2 and Ψ denote the vector algorithm of Section 3·3, then both Φ and Ψ are monotone bounding chains.*

Proof. This is easy to verify directly once we recognize that if $z \preceq \tilde{z}$ and $\alpha_{[-j]} \preceq \tilde{\alpha}_{[-j]}$, then $p(\alpha_j \mid y, \tilde{z}, \tilde{\alpha}_{[-j]})$ stochastically dominates $p(\alpha_j \mid y, z, \alpha_{[-j]})$, $p(\omega \mid y, \tilde{z})$ stochastically dominates $p(\omega \mid y, z)$, and $\mathrm{Ga}(\delta_j + \tilde{z}_j, 1)$ stochastically dominates $\mathrm{Ga}(\delta_j + z_j, 1)$. Then, the partial ordering $\tilde{x}^{\mathrm{L}} \preceq x^{\mathrm{L}} \preceq x^{\mathrm{U}} \preceq \tilde{x}^{\mathrm{U}}$ is preserved by using inverse transform sampling with the same uniform random inputs to draw from the distributions in § 3·2 and § 3·3. □

To bound the expected running time until coalescence for $\Psi \circ \Phi^{M-1}$, we assume that if Z_t is a singleton, then Ψ causes $Z_{t+1} \times \Theta_{t+1}$ to be a singleton:

$$\Psi(\{z\} \times \Theta, v) = \{z'\} \times \{\theta'\} \tag{15}$$

for any z, Θ, v. We call this property the ability to conditionally induce coalescence on θ. While this may seem restrictive, our focus is on Bayesian computation where θ is the parameter of interest and z is part of an augmented-data model. In such settings it is often possible to develop bounding chain algorithms that can conditionally induce coalescence by including draws of the full parameter vector θ given the observed and augmented data.

The time until coalescence is $\tau = \min(t : X_t = \{x\})$, where t is incremented each time either Φ or Ψ is called, that is, we define the time index t via

$$X_{t+1} = \begin{cases} \Phi(X_t, u_t), & t+1 \not\equiv 0 \pmod{M}, \\ \Psi(X_t, v_t), & t+1 \equiv 0 \pmod{M}. \end{cases} \tag{16}$$

We check for coalescence only after each time Ψ is called. To initialize the chain, we pass the state space to Ψ:

$$X_0 = \Psi(\Omega_x, v_0). \tag{17}$$

In a coupling-from-the-past implementation, we would initialize $X_{-T} = \Psi(\Omega_x, v_{-T})$ for $T > 0$. Incrementing t each time we use either update, rather than with each block, guarantees that different values of τ corresponding to different choices of M can be compared directly, in that smaller τ roughly corresponds to faster computation. We then have the following theorem.

THEOREM 1. *Suppose Φ and Ψ define monotone bounding chains, and that Ψ conditionally induces coalescence on θ as in* (15)*. Then, letting Z_{M-1} be the bounding set for z at the end of the first sequence Φ^{M-1} and letting $|Z_{M-1}|$ be its cardinality, the expected time until coalescence of the composite algorithm* (16) *and* (17) *satisfies*

$$M \leqslant E(\tau) \leqslant \frac{M}{\mathrm{pr}(|Z_{M-1}| = 1)}. \tag{18}$$

In our composite algorithm for the Dirichlet-multinomial model, we let Φ be the componentwise algorithm of § 3·2 and Ψ be the vector algorithm of § 3·3. It is then straightforward to show that the assumptions of Theorem 1 apply. Moreover, if we let ϕ be a stochastic recursive sequence for the Gibbs sampler in (8) and ψ correspond to the sampler in (4)–(5), then the composite chain $\psi \circ \phi^{M-1}$ has the correct stationary distribution $p(z, \theta \mid y)$, and $\Psi \circ \Phi^{M-1}$ is a valid bounding chain for this composite Gibbs sampler. Thus, a perfect sampler using the composite updates $\Psi \circ \Phi^{M-1}$ will return a genuine draw from $p(z, \theta \mid y)$.

Proof of Theorem 1. The lower bound is trivially true because we check for coalescence only at the end of a block, and it takes M steps to reach the end of the first block.

For the upper bound, we will show that τ is stochastically dominated by a scaled geometric random variable with expectation $M/\mathrm{pr}(|Z_{M-1}| = 1)$. Consider a modified version of Ψ, called Ψ_0, which first resets $X = \Omega_x = \Omega_z \times \Omega_\theta$ and then updates Ω_x via Ψ. That is, $\Psi_0(X, v) = \Psi(\Omega_x, v)$ for any X. Because we use Ψ_0 to set the algorithm's starting values, we can write the composite algorithm as $\cdots \circ \Psi \circ \Phi^{M-1} \circ \Psi \circ \Phi^{M-1} \circ \Psi_0$. Since Ψ conditionally induces coalescence on θ, if Z has coalesced at the end of Φ^{M-1}, then Ψ will cause Θ to coalesce in the next step. Thus we can group the composite algorithm's updates

$$\cdots \circ (\Phi^{M-1} \circ \Psi) \circ (\Phi^{M-1} \circ \Psi) \circ (\Phi^{M-1} \circ \Psi_0)$$

and check whether Z has coalesced at the end of each block, though in practice we must be sure to still call Ψ after Z has coalesced to obtain a valid draw (z, θ). By using the same sequence of random inputs, we couple this composite algorithm to a modified algorithm that substitutes Ψ_0 for every Ψ, and we similarly group the sequence of updates in the modified algorithm:

$$\cdots \circ (\Phi^{M-1} \circ \Psi_0) \circ (\Phi^{M-1} \circ \Psi_0) \circ (\Phi^{M-1} \circ \Psi_0).$$

Assuming that the random inputs to the stochastic recursive sequences at different steps are independent, then since Ψ_0 deterministically resets X ignoring the current states, the blocks in the modified algorithm are independent, and the indicators I_m for Z-coalescence in block m are independent and identically distributed, with $\mathrm{pr}(I_m = 1) = \mathrm{pr}(|Z_{M-1}| = 1)$. Therefore the number of steps until the first $I_m = 1$ is a scaled geometric random variable with mean $M/\mathrm{pr}(|Z_{M-1}| = 1)$, where the scaling is necessary because there are M steps per group.

Finally, since Φ and Ψ are monotone bounding chains, when we couple the composite and modified algorithm by using the same sequence of random inputs, Z-coalescence in the modified algorithm implies Z-coalescence in the composite algorithm, yielding the upper bound in (18). □

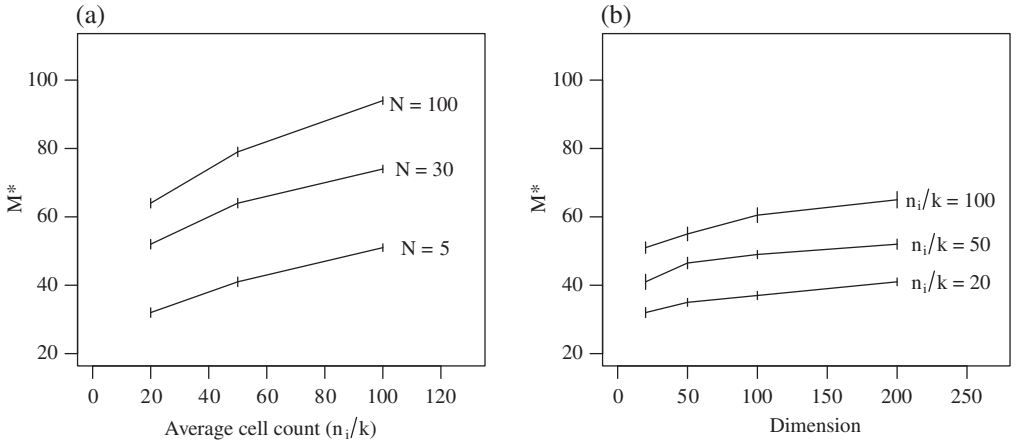

Fig. 2. Plot of the dependence of estimates of M^*, where $\text{pr}(|Z_{M^*-1}| = 1) = 0{\cdot}5$, on the number of rows N, the dimension k, and the average cell count n_i/k. In (a), the dimension is fixed at 20, and in (b), the number of rows is fixed at 5. Estimates of M^* are based on 100 simulated datasets under each condition, and error bars are ± 2 standard deviations estimated via bootstrap.

Since both M and $\text{pr}(|Z_{M-1}| = 1)$ are monotonically increasing functions of M, Theorem 1 suggests an intuitively sensible trade-off that occurs when choosing M, the block size. If M is too small, then $\text{pr}(|Z_{M-1}| = 1)$ is also small, and the upper bound on the expected time until coalescence blows up, so that we have much weaker guarantees on the expected running time. However, if M is so large that $\text{pr}(|Z_{M-1}| = 1) \approx 1$ and Z typically coalesces well before M steps, then much of the within-block computation will be wasted. In the next section, we will see from simulation that minimizing the upper bound in (18) can lead to a choice of M that is not far from optimal; see Fig. 3.

5. NUMERICAL ILLUSTRATION

To explore how the composite algorithm scales with the dimension k, the row total n_i, and the number of rows N, we simulated datasets under various conditions. In Fig. 2(a), we fixed the dimension $k = 20$ and simulated datasets with number of rows $N = 5$, 30, and 100, and with average cell count $n_i/k = 20$, 50, and 100. For Fig. 2(b), we fixed the number of rows $N = 5$ and simulated datasets with $k = 20$, 50, 100, and 200, and with average cell count $n_i/k = 20$, 50, and 100, for all i. For each choice of N, n_i, and k, we simulated and fit 100 datasets under independent exponential priors with mean 1 on $\alpha_1, \ldots, \alpha_k$. Figure 2 plots the results for running the componentwise algorithm on each dataset. The vertical axes are estimates of M^*, the block size such that $\text{pr}(|Z_{M^*-1}| = 1) = 0{\cdot}5$, so that if the block size $M = M^*$, then the expected time until coalescence for the composite algorithm satisfies $M^* \leqslant E(\tau) \leqslant 2M^*$. Encouragingly, M^* appears to increase slower than linearly with N, k, and n_i/k, and our method is not restricted to small samples, as we are able to fit datasets with up to 200 000 total counts.

However, our implementation leaves room for improvement. With $N = 100$, each iteration took approximately 14 seconds on a 2·80 GHz CPU. We do not view this as a fundamental limitation of our method, however, since we made no attempt to optimize our code, and further numerical and computational work could reduce the computation time for the draws from (10) and (12), just as there are now fast algorithms to evaluate the inverses of common cumulative distribution functions, for instance. Of course, the computation time per iteration must also increase linearly with k, since the componentwise algorithm must cycle through each dimension.

Table 1. *Artificial datasets used to illustrate the speed gains of the composite algorithm*

	Artificial dataset 1		Artificial dataset 2	
	Class 1	Class 2	Class 1	Class 2
Observation 1	5	10	10	20
Observation 2	6	9	12	18

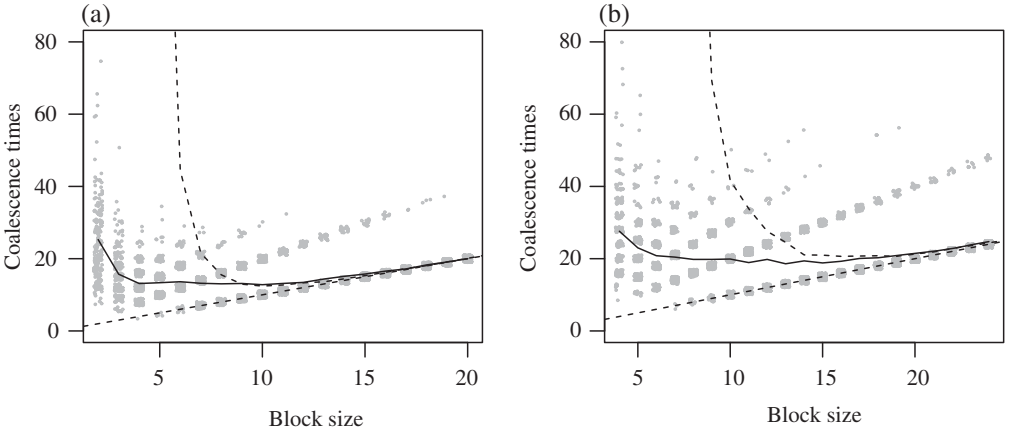

Fig. 3. Plots of the dependence of the composite algorithm's time until coalescence on the block size M, for the (a) low-count and (b) high-count artificial datasets in Table 1. Grey points are jittered values of the number of iterations until coalescence for different runs of the composite algorithm. Dashed lines are the lower bound and estimated upper bound in (18), and the solid lines connect average coalescence times for each block size.

To illustrate the speed gains in greater detail, we fit two simple artificial datasets, given in Table 1, assuming independent exponential prior distributions on α_1 and α_2. The median coalescence time for the vector algorithm applied to the first artificial dataset was 81 000 iterations, and 37% of runs had coalescence times greater than 10^5 iterations. We also applied the vector algorithm to the second artificial dataset, but we stopped it at 10^6 iterations without observing coalescence. Thus, it can take an extremely long time for the vector algorithm to return a value, even on low-dimensional, low-count problems with few observed rows. By itself, the vector algorithm is useless in practice, as is the componentwise algorithm, since the latter almost surely never returns a sample.

However, the composite algorithm (16) is much faster. Figure 3 shows the times until coalescence for the composite algorithm on both datasets, using various M. The gains are substantial. Whereas the vector algorithm could not return a single sample for the second artificial dataset in 10^6 iterations, typical coalescence times for the composite algorithm are in the dozens. Also plotted are the upper and lower bounds in (18) on the expected coalescence times. The denominator $\text{pr}(|Z_{M-1}| = 1)$ in the upper bound is estimated as the proportion with $|Z_{M-1}| = 1$ of 200 runs of the componentwise algorithm. On a 2·66 GHz CPU, code for the composite algorithm took on average 16·0 and 16·1 milliseconds per iteration, with standard deviations 0·2 milliseconds per iteration, for the low- and high-count artificial datasets, respectively.

6. Concluding remarks

The block structure of our composite algorithm is reminiscent of read-once coupling from the past (Wilson, 2000), which allows exact sampling without requiring the storage of random

inputs necessary for traditional coupling from the past. However, in Wilson (2000), each block restarts with the maximal bounding set if the chain fails to coalesce in the previous block. Thus, using the notation of § 4, a read-once implementation of our sampler could be based on blocks $\Psi \circ \Phi^{M-1} \circ \Psi_0$. Connection with another work in the literature is discussed in the Appendix.

On the limitation side, our approach assumes that the prior π on θ satisfies (2) and that the prior π_0 on ω satisfies Property 1. While this is limiting, importance sampling can be used if another prior is desired. Suppose we wish to estimate the expectation of a function $h(\theta)$ under the posterior distribution assuming the prior π_1. We can generate exact samples $\theta_{(1)}, \ldots, \theta_{(L)}$ from the posterior distribution assuming the prior π and use the importance sampling estimator

$$\hat{E}\{h(\theta) \mid y\} = \frac{\sum_{\ell=1}^{L} \frac{\pi_1(\theta_{(\ell)})}{\pi(\theta_{(\ell)})} h(\theta_{(\ell)})}{\sum_{\ell=1}^{L} \frac{\pi_1(\theta_{(\ell)})}{\pi(\theta_{(\ell)})}}.$$

An avenue for future work is to investigate the trade-off between using this importance sampling estimator based on independent exact samples under π, and using an approximate algorithm for sampling directly under π_1, such as any nonexact Markov chain sampler. A further challenge is to extend our approach to models that account for covariates.

Despite the limitations of our specific algorithm, we believe the idea of using divide-and-conquer composite bounding chains is suitably general, and it may provide an important step towards our ultimate goal of making perfect sampling a workhorse in statistical computing.

ACKNOWLEDGEMENT

We are grateful to two referees, the associate editor, and the editor for suggestions that substantially improved our presentation. We also thank Samuel Kou and Alexander Blocker for helpful discussions, Steven Finch for proofreading, and the U.S. National Science Foundation for partial financial support.

APPENDIX

Our data augmentation strategy was inspired by the algorithm of Kou & McCullagh (2009) for approximating the weighted matrix permanent, defined as

$$\mathrm{per}_{\omega}(A) = \sum_{\sigma \in \Pi_n} \omega^{\mathrm{cyc}(\sigma)} \prod_{i=1}^{n} A_{i,\sigma(i)},$$

where $A_{j,k}$ is the (j,k)th entry of an $n \times n$ matrix A; $\mathrm{cyc}(\sigma)$ is the number of cycles of the permutation σ; and Π_n is the set of all permutations of $\{1, \ldots, n\}$. The weighted permanent is the density function for permanental processes, a class of Cox processes that can be used for classification (McCullagh & Møller, 2006; McCullagh & Yang, 2006). The sequential importance sampling algorithm of Kou & McCullagh (2009) approximates the weighted permanent by drawing ordered partitions of $\{1, \ldots, n\}$, which can be mapped to permutations of $\{1, \ldots, n\}$ by viewing each block of the partition as a cycle, with the order of elements in each cycle determined by their order in the partition. If $A_{j,k} = 1$ for all j and k, then the weighted permanent is $\Gamma(\omega + n)/\Gamma(\omega)$. The algorithm of Kou & McCullagh (2009) can then be interpreted as first randomly ordering the elements of $\{1, \ldots, n\}$, and then partitioning them by drawing

$$z_i \sim \mathrm{Ber}\left(\frac{\omega}{\omega + i - 1}\right) \quad (i = 1, \ldots, n-1),$$

where $z_i = 0$ means that elements i and $i+1$ are in the same block.

In the aforementioned 2003 report by Minka, an exponential-family approximation is used to interpret the Dirichlet-multinomial model as a 'multinomial with "damped" counts.' Our method offers another perspective on that statement. Let us consider the likelihood for one observation $y = (y_1, \ldots, y_k)$:

$$\text{pr}(y \mid \omega, \lambda) = \frac{\Gamma(\omega)}{\Gamma(\omega + n)} \prod_{j=1}^{k} \frac{\Gamma(\omega\lambda_j + y_j)}{\Gamma(\omega\lambda_j)}. \tag{A1}$$

In (A1), we observe the class assignments $a_1, \ldots, a_n$ and count the number of individuals assigned to each class $y_j = \sum_{i=1}^{n} 1(a_i = j)$. Unlike in § 2, here we introduce an augmented variable σ via $p(y, \sigma \mid \theta) = p(\sigma \mid \theta)\ p(y \mid \sigma, \theta)$. We therefore must verify that $p(y \mid \theta) = \sum_\sigma p(\sigma \mid \theta)\ p(y \mid \sigma, \theta)$ is the same as (A1).

We first draw σ, a permutation of $\{1, \ldots, n\}$, from

$$p(\sigma \mid \theta) = \frac{\Gamma(\omega)}{\Gamma(\omega + n)} \omega^{\text{cyc}(\sigma)}.$$

This distribution depends only on ω, not λ, thus isolating the role of the parameter ω and giving us a natural interpretation. This permutation then partitions the individuals $\{1, \ldots, n\}$ according to the cycles of the permutation. We let $C_i(\sigma) \subseteq \{1, \ldots, n\}$ denote the set of elements included in the ith cycle of σ, where $i = 1, \ldots, \text{cyc}(\sigma)$, and the order can be chosen, for example, according to the smallest element in each cycle. We let B_{ij} denote the event that all of the individuals in cycle i are assigned to class j; that is, $B_{ij} = \{a_m = j \text{ for all } m \in C_i(\sigma)\}$. To ensure that class assignments are consistent within cycles of σ, we also define the event $D = \cap_{i=1}^{\text{cyc}(\sigma)} \cup_{j=1}^{k} B_{ij}$. Then, given σ, individuals are assigned to classes according to the multinomial distribution

$$p(y \mid \sigma, \theta) = 1_D \prod_{i=1}^{\text{cyc}(\sigma)} \prod_{j=1}^{k} \lambda_j^{1_{B_{ij}}},$$

where 1_D is the indicator for the event D. This distribution depends only on λ, not on ω. We can then verify that marginalizing $p(\sigma \mid \theta)\ p(y \mid \sigma, \theta)$ over σ yields (A1).

This augmented-data model quantifies several well-known and intuitive features of the Dirichlet-multinomial model. First, no approximation is required to interpret the Dirichlet-multinomial model as a 'multinomial with "damped" counts.' A draw from the Dirichlet-multinomial model can be exactly generated by first partitioning the data according to a permutation of individuals, and then by classifying individuals by a multinomial distribution acting on the blocks of the partition. The dampening is therefore a result of blocking observations according to the permutation σ.

Second, it is commonly known that small values of ω induce sparsity in the observed data, while the mean vector λ controls the expected frequencies of each class. Sparsity means that there may be many classes with zero observations and a few classes with many observations. Thus, sparsity is associated with high variance. In the permutation augmentation, small values of ω favour permutations with few cycles, so that most individuals are partitioned into just a few blocks. The multinomial allocation to classes then necessarily results in sparse observations, since the class assignment operates on blocks, not individuals. Conversely, large values of ω favour permutations with many cycles, corresponding to fine partitions of individuals and a lower chance of observing sparsity.

References

Blei, D. M., Ng, A. Y. & Jordan, M. I. (2003). Latent Dirichlet allocation. *J. Mach. Learn. Res.* **3**, 993–1022.

Casella, G., Mengersen, K. L., Robert, C. P. & Titterington, D. M. (2002). Perfect samplers for mixtures of distributions. *J. R. Statist. Soc.* B **64**, 777–90.

Craiu, R. V. & Meng, X.-L. (2011). Perfection within reach: Exact MCMC sampling. In *Handbook of Markov Chain Monte Carlo*, Ed. S. Brooks, A. Gelman, G. Jones and X.-L. Meng, pp. 205–32. Boca Raton: Chapman & Hall/CRC.

Escobar, M. D. & West, M. (1995). Bayesian density estimation and inference using mixtures. *J. Am. Statist. Assoc.* **90**, 577–88.

HÄGGSTRÖM, O. & NELANDER, K. (1999). On exact simulation from Markov random fields using coupling from the past. *Scand. J. Statist.* **26**, 395–411.
HOBERT, J. P., ROBERT, C. P. & TITTERINGTON, D. M. (1999). On perfect simulation for some mixtures of distributions. *Statist. Comp.* **9**, 287–98.
HOLMES, I., HARRIS, K. & QUINCE, C. (2012). Dirichlet multinomial mixtures: Generative models for microbial metagenomics. *PLoS ONE* **7**, e30126.
HUBER, M. (1998). Exact sampling and approximate counting techniques. In *Proc. 30th Symp. Theory Comp.* New York: Association for Computing Machinery, pp. 31–40.
HUBER, M. (2004). Perfect sampling using bounding chains. *Ann. Appl. Prob.* **14**, 734–53.
KOU, S. C. & MCCULLAGH, P. (2009). Approximating the α-permanent. *Biometrika* **96**, 635–44.
MCCULLAGH, P. & MØLLER, J. (2006). The permanental process. *Adv. Appl. Prob.* **38**, 873–88.
MCCULLAGH, P. & YANG, J. (2006). Stochastic classification models. In *Proc. Int. Cong. Math.*, vol. 3, Ed. M. Sanz-Sole', J. Soria, J. L. Varona and J. Verdera, pp. 669–86, Zurich: European Mathematical Society.
MØLLER, J. (1999). Perfect simulation of conditionally specified models. *J. R. Statist. Soc.* B **61**, 251–64.
MUKHOPADHYAY, S. & BHATTACHARYA, S. (2012). Perfect simulation for mixtures with known and unknown number of components. *Bayesian Anal.* **7**, 675–714.
MURDOCH, D. & MENG, X.-L. (2001). Towards perfect sampling for Bayesian mixture priors. In *Proc. Int. Soc. Bayesian Anal.* 2000, Ed. E. I. George, pp. 381–90. Luxembourg: Eurostat.
MURDOCH, D. J. & GREEN, P. J. (1998). Exact sampling from a continuous state space. *Scand. J. Statist.* **25**, 483–502.
PROPP, J. G. & WILSON, D. B. (1996). Exact sampling with coupled Markov chains and applications to statistical mechanics. *Random Struct. Algor.* **9**, 223–52.
WILSON, D. B. (2000). How to couple from the past using a read-once source of randomness. *Random Struct. Algor.* **16**, 85–113.

[*Received November* 2012. *Revised April* 2013]