Stat Comput (2015) 25:95–110
DOI 10.1007/s11222-014-9526-5

On the use of Markov chain Monte Carlo methods for the sampling of mixture models: a statistical perspective

Randal Douc · Florian Maire · Jimmy Olsson

Received: 14 March 2014 / Accepted: 21 October 2014 / Published online: 19 December 2014

Abstract In this paper we study asymptotic properties of different data-augmentation-type Markov chain Monte Carlo algorithms sampling from mixture models comprising discrete as well as continuous random variables. Of particular interest to us is the situation where sampling from the conditional distribution of the continuous component given the discrete component is infeasible. In this context, we advance *Carlin & Chib's pseudo-prior method* as an alternative way of infering mixture models and discuss and compare different algorithms based on this scheme. We propose a novel algorithm, the *Frozen Carlin & Chib sampler*, which is computationally less demanding than any Metropolised Carlin & Chib-type algorithm. The significant gain of computational efficiency is however obtained at the cost of some asymptotic variance. The performance of the algorithm vis-à-vis alternative schemes is, using some recent results obtained in Maire et al. (Ann Stat 42: 1483–1510, 2014) for inhomogeneous Markov chains evolving alternatingly according to two different π^*-reversible Markov transition kernels, investigated theoretically as well as numerically.

Keywords Asymptotic variance · Carlin & Chib's pseudo-prior method · Inhomogeneous Markov chains · Metropolisation · Mixture models · Peskun ordering

R. Douc (✉) · F. Maire
CNRS UMR 5157 SAMOVAR, Institut Télécom/Télécom SudParis, Evry, France
e-mail: randal.douc@it-sudparis.eu;
randal.douc@telecom-sudparis.eu

F. Maire
e-mail: florian.maire@telecom-sudparis.eu

J. Olsson
KTH Royal Institute of Technology, Stockholm, Sweden
e-mail: jimmyol@kth.se

1 Introduction

To sample from *mixture probability distributions* π^* of a discrete and a continuous random variable, denoted by M and Z, respectively, is a fundamental problem in statistics. In this paper we study the use of different data-augmentation-type Markov chain Monte Carlo (MCMC) algorithms for this purpose. Of particular interest to us is the situation where sampling from the conditional distribution of Z given M is infeasible.

In many applications the most natural approach to sampling from π^* goes via the *Gibbs sampler*, which samples alternatingly from the conditional distributions $M \mid Z$ and $Z \mid M$. Since the component M is discrete, the former sampling step is most often feasible (at least when π^* is known up to a normalising constant). On the contrary, drawing $Z \mid M$ is in general infeasible; in that case this sampling step is typically *metropolised* by replacing, with a Metropolis-Hastings probability, the value of Z obtained at the previous iteration by a candidate drawn from some proposal kernel. This yields a so-called *Metropolis-within-Gibbs*—or *hybrid*—sampler.

However, when the modes of the mixture distribution are well-separated, implying a strong correlation between M and Z, the Gibbs sampler has in general very limited capacity to move flexibly between the different modes, and exhibits for this reason most often very poor mixing (see Hurn et al. (2003) for some discussion). Since this problem is due to model dependence, it effects the standard Gibbs as well as the hybrid sampler. In order to cope with this well-known problem, we consider *Carlin & Chib's pseudo-prior method* (Carlin and Chib 1995) as an alternative way to infer mixture models. The method extends the target model with a set of auxiliary variables that are used to help moving the discrete component. When the distribution of the auxiliary variables (determined by a set of *pseudo-priors*) is chosen optimally (an idealised situation however), the method produces indeed i.i.d. samples from the marginal distribution of M under π^*. Given M, the Z component is sampled from $Z \mid M$ in accordance with the Gibbs sampler, with possible metropolisation in the case where exact sampling is infeasible. The latter scheme will be referred as the *Metropolised Carlin & Chib-type* (MCC) *sampler*.

Surprisingly, it turns out that passing directly and deterministically the value of the Mth auxiliary variable, obtained through sampling from the pseudo-priors at the beginning of the loop, to the Z component yields a Markov chain that is still π^*-reversible (see Lemma 8). Moreover, using some recent results obtained in Maire et al. (2014) on the comparison of asymptotic variance for inhomogeneous Markov chains, we are able to prove (see Theorem 7) that this novel MCMC algorithm, referred to as the *Frozen Carlin & Chib-type* (FCC) *sampler*, generates a Markov chain whose sample path averages have always higher asymptotic variance than those of the MCC sampler for a large class of objective functions. This is well in line with our expectations, as the MCC sampler "refreshes" more often the Z component. On the other hand, since this component is already modified through sampling from the pseudo-priors, which, when well-designed, should be close to the true conditional distribution $Z \mid M$, we may expect that the additional mixing provided by the MCC sampler is only marginal. This is also confirmed by our simulations, which indicate only a small advantage of the MCC sampler to the FCC sampler in terms of autocorrelation. As the FCC algorithm omits completely the Metropolis-Hastings operation of the MCC sampler, it is considerably more computationally efficient. Thus, we consider the FCC sampler as a strong alternative to the MCC sampler in terms of efficiency (inverse standard error per unit CPU).

The paper is structured as follows: in Sect. 2 we introduce some notation and describe precisely the mixture model framework under consideration. Section 3 describes the Carlin & Chib-type MCMC samplers studied in the paper. In Sect. 4 we prove that the involved algorithms are indeed π^*-reversible and provide a theoretical comparison of the MCC and FCC samplers. Finally, in the implementation part, Sect. 5, we illustrate and compare numerically the algorithms on two examples: a toy mixture of Gaussian distributions and a model where the mixture variables are only partially observed. The latter is applied to two different settings including the estimation of a warping parameter for handwritten digits analysis.

2 Preliminaries

2.1 Notation

We assume throughout the paper that all variables are defined on a common probability space $(\Omega, \mathcal{F}, \mathbb{P})$. We will use upper case for random variables and lower case for realisations of the same, and write "$X \rightsquigarrow x$" when x is realisation of X. We write "$X \sim \mu$" to indicate that the random variable X is distributed according to the probability measure μ. For any μ-integrable function h we let $\mu(h) := \int h(x)\mu(dx)$ be the expectation of $h(X)$ under μ. Similarly, for Markov transition kernels M, we write $Mf(x) := \int f(x')M(x, dx')$ whenever this integral is well-defined. For any two probability measures μ and μ' defined on some measurable spaces $(\mathsf{X}, \mathcal{X})$ and $(\mathsf{X}', \mathcal{X}')$, respectively, we denote by $\mu(dx)\mu'(dx')$ the product measure $\mu \otimes \mu'(dx \times dx')$ on $(\mathsf{X} \times \mathsf{X}', \mathcal{X} \otimes \mathcal{X}')$. For $(m, n) \in \mathbb{Z}^2$ such that $m \leq n$, we denote by $[\![m, n]\!] := \{m, m+1, \ldots, n\} \subset \mathbb{Z}$. Moreover, we denote by $\mathbb{N}^* := \mathbb{N} \setminus \{0\}$ the set of positive integers.

Finally, given some probability measure π on $(\mathsf{X}, \mathcal{X})$ we recall, first, that a Markov transition kernel M is called π-*reversible* if $\pi(dx)M(x, dx') = \pi(x')M(x', dx)$ and, second, that π-reversibility of M implies straightforwardly that this kernel allows π as a stationary distribution.

2.2 Mixture models

Throughout this paper, our main objective is to sample a probability distribution π^* on some product space $\mathsf{Y} := [\![1, n]\!] \times \mathsf{Z}$, where n is at least 2 and $(\mathsf{Z}, \mathcal{Z})$ is some (typically uncountable) measurable space, associated with the σ-field $\mathcal{Y} := 2^{[\![1,n]\!]} \otimes \mathcal{Z}$. Thus, a π^*-distributed random variable $Y = (M, Z)$ comprises a $[\![1, n]\!]$-valued (discrete) random variable M and a Z-valued (typically continuous) random variable Z.

In the following we assume that $\pi^*(dm \times dz)$ is dominated by a product measure $|dm|\nu(dz)$, where $|dm|$ denotes the counting measure on $[\![1, n]\!]$ and ν is some nonnegative measure on $(\mathsf{Z}, \mathcal{Z})$, and denote by $\pi^*(m, z)$ the corresponding density function on $[\![1, n]\!] \times \mathsf{Z}$. We may then define the marginal probability functions

$$\pi^*(m) := \int \pi^*(m, z)\nu(dz)$$

$$\pi^*(z) := \sum_{m=1}^{n} \pi^*(m, z)$$

(w.r.t. $|dm|$ and ν, respectively) on $[\![1, n]\!]$ and Z, respectively, as well as the conditional probability functions

$$\pi^*(m \mid z) := \frac{\pi^*(m, z)}{\pi^*(z)},$$

$$\pi^*(z \mid m) := \frac{\pi^*(m, z)}{\pi^*(m)} \tag{1}$$

(w.r.t. $|dm|$ and ν, respectively) on $[\![1, n]\!]$ and Z, respectively.

Remark 1 We stress at this stage that our focus is set on the problem of how to sample efficiently from a given mixture model; in particular, Bayesian model selection problems, where the dimensionality of the parameter vector is typically not fixed (see, e.g., Green (1995), Petralias and Dellaportas (2013)), fall outside the scope of this paper.

3 Markov chain Monte Carlo methods for mixture models

Using the conditional distributions (1), a natural way of sampling π^* consists in implementing a standard Gibbs sampler simulating a Markov chain $\{Y_k^{(1)}; k \in \mathbb{N}\}$ with transitions described by the following algorithm—the Markov chain superscript refers to the algorithm index.

Algorithm 1 Gibbs sampler

```
Require: $Y_k^{(1)} = (m, z)$,
  (i) draw $M' \sim \pi^*(dm \mid z)$ and call the outcome $m'$ (abbr. $\rightsquigarrow m'$),
  (ii) draw $Z' \sim \pi^*(dz \mid m') \rightsquigarrow z'$,
  (iii) set $Y_{k+1}^{(1)} \leftarrow (m', z')$.
```

Remark 2 Since M is a discrete random variable it is most often possible to sample $M \sim \pi^*(dm \mid z)$. In contrast, sampling $Z \sim \pi^*(dz \mid m)$ is not always possible. In that case, one may replace Step (ii) by a Metropolis-Hastings step, yielding a Metropolis-within-Gibbs algorithm (see [Robert and Casella (2004), section 10.3.3] for details).

Using the output of Algorithm 1, any expectation $\pi^*(f)$, where f is some π^*-integrable objective function on Y, can be estimated by the sample path average

$$\hat{\pi}_n^{(1)}(f) := \frac{1}{n}\sum_{k=0}^{n-1} f(Y_k^{(1)}).$$

A well-known problem with this approach is that even though Algorithm 1 generates a Markov chain $\{Y_k^{(1)}; k \in \mathbb{N}\}$ with stationary distribution π^*, the discrete component $\{M_k^{(1)}; k \in \mathbb{N}\}$ tends in practice to get stuck in a few states. Indeed, when the variable Z is sampled from its conditional distribution given $M = m$, the probability of jumping to another index $m' \neq m$ is proportional to $\pi^*(m', z)$, which may be very low when the index component M is informative concerning the localisation of Z. This will lead to poor mixing and, consequently, high variance of any estimator $\hat{\pi}_n^{(1)}(f)$. To be specific, let h be some function on $[\![1, n]\!]$ and assume that we run the Gibbs sampler in Algorithm 1 to estimate $\int h(m)\pi^*(dm)$. Then $\{M_k^{(1)}; k \in \mathbb{N}\}$ is itself a Markov chain with transition kernel $G(m, dm') := \int \pi^*(dz \mid m)\pi^*(dm' \mid z)$ and starting with $M_0^{(1)} \sim \pi^*(dm)$, we have, as π^* is invariant under G,

$$\begin{aligned}&\mathrm{Cov}(h(M_0^{(1)}), h(M_1^{(1)}))\\ &= \mathbb{E}\left(h(M_0^{(1)})Gh(M_0^{(1)})\right) - \mathbb{E}^2\left(h(M_0^{(1)})\right),\end{aligned}$$

where, by Jensen's inequality,

$$\begin{aligned}\mathbb{E}\left(h(M_0^{(1)})Gh(M_0^{(1)})\right) &= \int \pi^*(dz)\left(\int \pi^*(dm \mid z)h(m)\right)^2\\ &\geq \mathbb{E}^2(h(M_0^{(1)})).\end{aligned}$$

Consequently,

$$\mathrm{Cov}\left(h(M_0^{(1)}), h(M_1^{(1)})\right) \geq 0.$$

Combining this with the fact that G is $\pi^*(dm)$-reversible, we obtain

$$\mathrm{Cov}\left(h(M_0^{(1)}), h(M_{2k+1}^{(1)})\right) = \mathrm{Cov}(G^k h(M_0^{(1)}), G(G^k h)(M_0^{(1)})) \geq 0.$$

Moreover, using again that G is $\pi^*(dm)$-reversible,

$$\mathrm{Cov}\left(h(M_0^{(1)}), h(M_{2k}^{(1)})\right) = \mathrm{Cov}\left(G^k h(M_0^{(1)}), G^k h(M_0^{(1)})\right) \geq 0.$$

Finally, letting $f(m, z) \equiv h(m)$, we obtain

$$\mathrm{Var}\left(\sqrt{n}\hat{\pi}_n^{(1)}(f)\right) \geq \mathrm{Var}\left(h(M_0^{(1)})\right),$$

showing that the Gibbs sampler approximates the index less accurately than i.i.d. sampling from $\pi^*(dm)$.

The *pseudo-prior method* of B. P. Carlin and S. Chib (Carlin and Chib 1995) was introduced in the context of model selection and can successfully be adapted to mixture models. By introducing some auxiliary variables, this method increases the number of moves of the index component. The algorithm may be regarded as a Gibbs sampler-based data-augmentation algorithm targeting the distribution π defined on the extended state space $[\![1, n]\!] \times \mathsf{Z}^n$ by

$$\pi(dm \times du) := \pi^*(dm \times du_m) \bigotimes_{j \neq m} \rho_j(du_j), \tag{2}$$

where $u = (u_1, \ldots, u_n) \in \mathsf{Z}^n$ and the probability measures $\{\rho_j; j \in [\![1, n]\!]\}$ are referred to as *pseudo-priors* (the terminology comes from Carlin and Chib (1995)). We assume that also the pseudo-priors are dominated jointly by the nonnegative measure ν and use the same symbols ρ_j for denoting the corresponding densities. As a consequence, the measure π defined in (2) admits the density

$$\pi(m, u) := \pi^*(m, u_m) \prod_{j \neq m} \rho_j(u_j) \quad ((m, u) \in [\![1, n]\!] \times \mathsf{Z}^n)$$

with respect to $|dm| \otimes \nu^{\otimes n}(du)$. In the following, the density π is assumed to be positive, i.e., $\pi(m, u) > 0$ for all $(m, u) \in [\![1, n]\!] \times \mathsf{Z}^n$.

Remark 3 The pseudo-priors constitute a design parameter of the method that has to be tuned by the user provided that (i) the pseudo-priors are analytically tractable and (ii) can be sampled from. Even though the Markov chain is in general ergodic for any choice of the pseudo-priors, the design of the same impacts significantly the performance of the algorithm in practice. Methods for designing the pseudo-priors are often problem specific; still, some possible guidelines are provided in Sect. 5.

Denote by $\{Y_k^{(2)}; k \in \mathbb{N}\}$ the Markov chain generated by this algorithm, which we will in the following refer to as the *Carlin & Chib-type* (CC-type) *sampler* and whose transitions comprise the $n+1$ sub-steps described in Algorithm 2 below.

Algorithm 2 CC-type sampler

```
Require: $Y_k^{(2)} = (m, u_m)$,
    (i) for all $j \neq m$, draw $U_j \sim \rho_j \rightsquigarrow u_j$,
   (ii) draw $M' \sim \pi(dm \mid u) \rightsquigarrow m'$,
  (iii) draw $U'_{m'} \sim \pi^*(dz \mid m') \rightsquigarrow u'_{m'}$,
   (iv) set $Y_{k+1}^{(2)} \leftarrow (m', u'_{m'})$.
```

Intuitively, Algorithm 2 allows jumps between different indices to occur more frequently than in the Gibbs sampler; indeed, in Step (ii) the probability of moving to the index m' is

$$\pi(m' \mid u) \propto \pi^*(m', u_{m'})/\rho_{m'}(u_{m'}), \tag{3}$$

where the right hand side is close to $\pi^*(m')$ if the pseudo-priors are chosen such that $\rho_\ell(z)$ is close to $\pi^*(z \mid \ell)$ for all $(\ell, z) \in [\![1, n]\!] \times \mathsf{Z}$. The optimal case where $\rho_\ell(z) \equiv \pi^*(z \mid \ell)$ implies, via (3), that

$$\pi(m' \mid u) \propto \pi^*(m').$$

Thus, in this case, Step (ii) draws actually M' according to the *exact* marginal $\pi^*(dm)$ of the class index random variable regardless the value of u, which implies that the algorithm simulates *i.i.d. samples* according to π^*. This actually gives a more efficient approximation than that produced by the Gibbs sampler, whose variance w.r.t. the index component is, as we remarked previously, always larger than that obtained through i.i.d. sampling from $\pi^*(dm)$. However, this ideal situation requires the quantity $\pi^*(z \mid \ell)$ to be tractable, which is typically not the case.

As in Remark 2, one may replace Step (iii) in Algorithm 2 by a Metropolis-Hastings step if sampling from $\pi^*(dz \mid m')$ is infeasible. This is most often the case when $\pi^*(dm \times dz)$ is the *a posteriori* distribution of (M, Z) conditionally on one or several observations (see Sect. 5.2 for an example). In that case π^* is known only up to a normalizing constant, which prevents sampling from the conditional density $\pi^*(dz \mid m')$. The resulting algorithm will in the following be referred to as the Metropolised CC-type (MCC) sampler and is presented in Algorithm 3, where $\{R_\ell; \ell \in [\![1, n]\!]\}$ is a set of proposal kernels on $\mathsf{Z} \times \mathcal{Z}$. Assume for simplicity that all these kernels are jointly dominated by the reference measure ν and denote by $\{r_\ell; \ell \in [\![1, n]\!]\}$ the corresponding transition densities with respect to this measure. Throughout the paper, we will assume that $r_\ell(u, z) > 0$ for all $\ell \in [\![1, n]\!]$ and all $(u, z) \in \mathsf{Z}^2$. Introducing also the Metropolis-Hastings acceptance probability

$$\alpha_\ell(u, z) := \frac{\pi^*(\ell, z) r_\ell(z, u)}{\pi^*(\ell, u) r_\ell(u, z)} \wedge 1 \quad ((\ell, u, z) \in [\![1, n]\!] \times \mathsf{Z}^2), \tag{4}$$

the MCC algorithm is described as follows.

Algorithm 3 MCC sampler

```
Require: $Y_k^{(3)} = (m, u_m)$,
    (i) for all $j \neq m$, draw $U_j \sim \rho_j \rightsquigarrow u_j$,
   (ii) draw $M' \sim \pi(dm \mid u) \rightsquigarrow m'$,
(iii.1) draw $Z' \sim R_{m'}(u_{m'}, dz) \rightsquigarrow z'$,
(iii.2) set $U'_{m'} \leftarrow \begin{cases} z & \text{w. pr. } \alpha_{m'}(u_{m'}, z'), \\ u_{m'} & \text{otherwise,} \end{cases} \rightsquigarrow u'_{m'}$,
   (iv) set $Y_{k+1}^{(3)} \leftarrow (m', u'_{m'})$.
```

Note that Step (iii) generates, given $u_{m'}$, $U'_{m'} \sim K_{m'}(u_{m'}, du')$, where

$$K_\ell(u, du') := R_\ell(u, du')\alpha_\ell(u, u') \\ +\delta_u(du')\left(1 - \int R_\ell(u, du'')\alpha_\ell(u, u'')\right) \\ ((u, \ell) \in \mathsf{Z} \times [\![1, n]\!]). \tag{5}$$

It can be easily checked (using (4)) that $K_{m'}$ is indeed a Metropolis-Hastings kernel with respect to $\pi(dz \mid m')$; it is thus $\pi(dz \mid m')$-reversible. Moreover, in the particular case where $R_\ell(u, du') = \pi^*(du' \mid \ell)$ for all $\ell \in [\![1, n]\!]$, the MCC algorithm and Algorithm 2 coincide.

Remarkably, it turns out that Step (iii) in Algorithm 3 may be omitted, which may, in some cases, imply a significant gain of computational complexity. The MCC algorithm sampler then simplifies to what we will refer to as the Frozen CC-type (FCC) sampler, which is described formally in Algorithm 4.

Figure 1 compares the transitions of Algorithms 2, 3, and 4, and shows clearly that the algorithms differ only in the way the continuous component is updated. More specifically, from the diagram it is clear that the three algorithms under consideration can be regarded as inhomogeneous Markov chains evolving alternatingly according to two different kernels comprising Steps (i)–(ii) and Steps (iii)–(iv), respectively. The first kernel updates (m, z) according to a π^*-reversible transition. The second kernel may, as for the

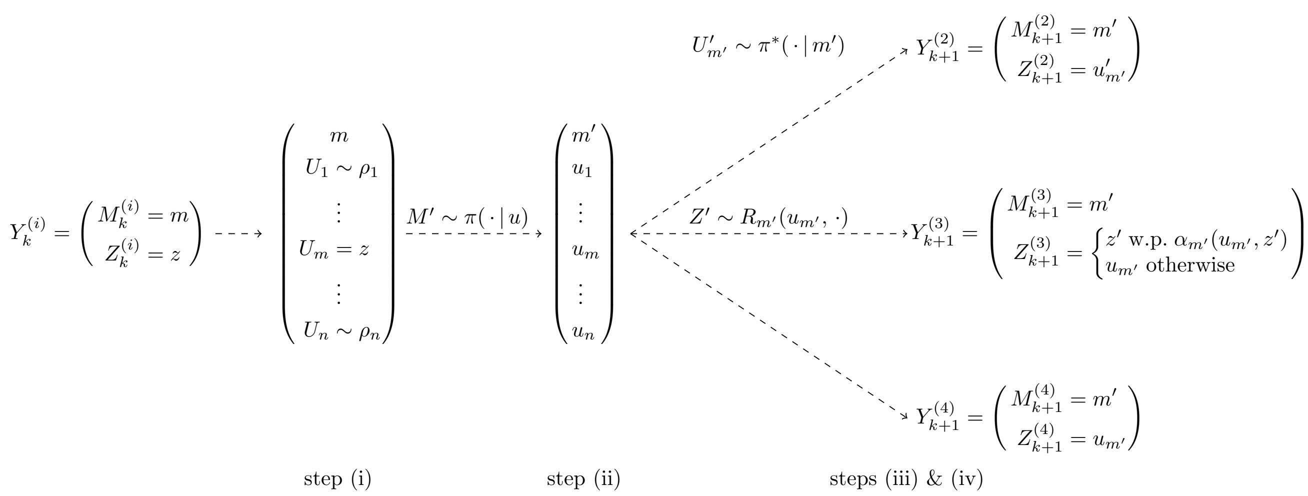

$Y_k^{(i)} = \begin{pmatrix} M_k^{(i)} = m \\ Z_k^{(i)} = z \end{pmatrix}$
$\begin{pmatrix} m \\ U_1 \sim \rho_1 \\ \vdots \\ U_m = z \\ \vdots \\ U_n \sim \rho_n \end{pmatrix}$
$M' \sim \pi(\cdot \mid u)$
$\begin{pmatrix} m' \\ u_1 \\ \vdots \\ u_m \\ \vdots \\ u_n \end{pmatrix}$
$U'_{m'} \sim \pi^*(\cdot \mid m')$
$Y_{k+1}^{(2)} = \begin{pmatrix} M_{k+1}^{(2)} = m' \\ Z_{k+1}^{(2)} = u'_{m'} \end{pmatrix}$
$Z' \sim R_{m'}(u_{m'}, \cdot)$
$Y_{k+1}^{(3)} = \begin{pmatrix} M_{k+1}^{(3)} = m' \\ Z_{k+1}^{(3)} = \begin{cases} z' \text{ w.p. } \alpha_{m'}(u_{m'}, z') \\ u_{m'} \text{ otherwise} \end{cases} \end{pmatrix}$
$Y_{k+1}^{(4)} = \begin{pmatrix} M_{k+1}^{(4)} = m' \\ Z_{k+1}^{(4)} = u_{m'} \end{pmatrix}$
step (i)
step (ii)
steps (iii) & (iv)

Fig. 1 Comparison of the CC-type samplers

Algorithm 4 FCC sampler

```
Require: $Y_k^{(4)} = (m, u_m)$,
    (i) for all $j \neq m$, draw $U_j \sim \rho_j \rightsquigarrow u_j$,
    (ii) draw $M' \sim \pi(dm \mid u) \rightsquigarrow m'$,
    (iii) $u'_{m'} \leftarrow u_{m'}$,
    (iv) set $Y_{k+1}^{(4)} \leftarrow (m', u'_{m'})$.
```

CC and MCC algorithms, correspond to some specific π^*-reversible kernel or can, as in the FCC algorithm, be simply the identity kernel (which is, being reversible with respect to any distribution, straightforwardly π^*-reversible). Depending on the context, this shortcut makes FCC considerably less computationally demanding. Nevertheless, as stated in Theorem 7 below, FCC is always less efficient in terms of asymptotic variance than the corresponding MCC sampler. Intuitively, this stems from the fact that once the index M' is drawn, the associated continuous component is selected deterministically without being "refreshed" (on the contrary to Step (iii) in Algorithm 3). In addition, as our numerical simulations indicate that the gain of the asymptotic variance obtained by refreshing, as in the MCC sampler, this component instead of freezing the same as in the FCC sampler seems to be limited (see Sect. 5 for details), we definitely regard the FCC algorithm as a strong challenger of the MCC sampler.

4 Theoretical results

4.1 Comparison of asymptotic variance of inhomogeneous Markov chains

In this section we recall briefly the main result of [Maire et al. (2014), Theorem 4], which is propelling the coming analysis. The following—now classical—orderings of Markov kernels turn out to be highly useful.

Definition 1 Let P_0 and P_1 be Markov transition kernels on some state space $(\mathsf{X}, \mathcal{X})$ with common invariant distribution π. We say that P_1 *dominates* P_0

- *on the off-diagonal*, denoted $P_1 \succeq P_0$, if for all $\mathsf{A} \in \mathcal{X}$ and π-a.s. all $x \in \mathsf{X}$,

$$P_1(x, \mathsf{A} \setminus \{x\}) \geq P_0(x, \mathsf{A} \setminus \{x\}).$$

- *in the covariance ordering*, denoted $P_1 \succcurlyeq P_0$, if for all $f \in \mathsf{L}^2(\pi)$,

$$\int f(x) P_1 f(x) \pi(dx) \leq \int f(x) P_0 f(x) \pi(dx).$$

The covariance ordering, which was introduced implicitly in [Tierney (1998), p. 5] and formalised in Mira (2001), is an extension of the off-diagonal ordering, since, according to [Tierney (1998), Lemma 3], $P_1 \succeq P_0$ implies $P_1 \succcurlyeq P_0$. Moreover, it turns out that for reversible kernels, $P_1 \succcurlyeq P_0$ implies that the asymptotic variance of sample path averages of chains generated by P_1 is smaller than or equal to that of chains generated by P_0 (see the proof of [Tierney (1998), Theorem 4]).

In algorithms of Gibbs-type, the ordering in Definition 1 is usually not applicable, since the fact that all candidates are accepted with probability one prevents the chain from remaining in the same state. The ordering is however still meaningful when a component is discrete.

In the following, let P_i and Q_i, $i \in [\![0, 1]\!]$, be Markov transition kernels on $(\mathsf{X}, \mathcal{X})$ and let $\{X_k^{(0)}; k \in \mathbb{N}\}$ and $\{X_k^{(1)}; k \in \mathbb{N}\}$ be inhomogeneous Markov chains evolving as follows:

$$X_0^{(i)} \xrightarrow{P_i} X_1^{(i)} \xrightarrow{Q_i} X_2^{(i)} \xrightarrow{P_i} X_3^{(i)} \xrightarrow{Q_i} \cdots \tag{6}$$

This means that for all $k \in \mathbb{N}$ and $i \in \{0, 1\}$,

- $\mathbb{P}\left(X_{2k+1}^{(i)} \in dx \mid \mathcal{F}_{2k}^{(i)}\right) = P_i(X_{2k}^{(i)}, dx)$,
- $\mathbb{P}\left(X_{2k+2}^{(i)} \in dx \mid \mathcal{F}_{2k+1}^{(i)}\right) = Q_i(X_{2k+1}^{(i)}, dx)$,

where $\mathcal{F}_n^{(i)} := \sigma(X_0^{(i)}, \ldots, X_n^{(i)})$, $n \in \mathbb{N}$. Now, impose the following assumption.

(A1) (i) P_i and Q_i, $i \in [\![0, 1]\!]$, are π-reversible,
(ii) $P_1 \succcurlyeq P_0$ and $Q_1 \succcurlyeq Q_0$.

As mentioned above, $P_1 \succeq P_0$ implies $P_1 \succcurlyeq P_0$; thus, in practice, a sufficient condition for (**A1**)(ii) is that $P_1 \succeq P_0$ and $Q_1 \succeq Q_0$. Under these assumptions, Maire et al. (2014) established the following result.

Theorem 4 [*Maire et al. (2014)*] *Assume that P_i and Q_i, $i \in [\![0, 1]\!]$, satisfy* (**A1**) *and let $\{X_k^{(i)}; k \in \mathbb{N}\}$, $i \in [\![0, 1]\!]$, be Markov chains evolving as in (6) with initial distribution π. Then for all $f \in \mathsf{L}^2(\pi)$ such that for $i \in [\![0, 1]\!]$,*

$$\sum_{k=1}^{\infty} \left(\left| \operatorname{Cov}(f(X_0^{(i)}), f(X_k^{(i)})) \right| + \left| \operatorname{Cov}(f(X_1^{(i)}), f(X_{k+1}^{(i)})) \right| \right) < \infty \tag{7}$$

it holds that

$$v_1(f) \leq v_0(f), \tag{8}$$

where

$$v_i(f) := \lim_{n \to \infty} \frac{1}{n} \operatorname{Var}\left(\sum_{k=0}^{n-1} f(X_k^{(i)}) \right) \quad (i \in [\![0, 1]\!]). \tag{9}$$

Remark 5 As shown in [Maire et al. (2014), Proposition 9], under the assumption that the product kernels $P_i Q_i$, $i \in [\![0, 1]\!]$, are both V-*geometrically ergodic* (according to Definition 7 in the same paper), the absolute summability assumption (7) holds true for all objective functions f such that f and $P_i f$, $i \in [\![0, 1]\!]$, have all bounded $\sqrt{V}$-norm; see again Maire et al. (2014) for details.

4.2 The MCC sampler versus the FCC sampler

In the light of the remarks following Algorithm 4 it is reasonable to assume that the CC-type sampler (Algorithm 2) and the MCC sampler provides more accurate estimates than the FCC sampler. However, since $\{Y_k^{(3)}; k \in \mathbb{N}\}$ is not π^*-reversible, [Tierney (1998), Theorem 4] does not allow these two algorithms to be compared. Nevertheless, using Theorem 4 we may provide a theoretical justification advocating the MCC and CC-type samplers ahead of the FCC sampler in terms of asymptotic variance. To do this we decompose the transition kernels of $\{Y_k^{(3)}; k \in \mathbb{N}\}$ and $\{Y_k^{(4)}; k \in \mathbb{N}\}$ into products of two π^*-reversible kernels. More specific, let $\{X_k^{(3)}; k \in \mathbb{N}\}$ and $\{X_k^{(4)}; k \in \mathbb{N}\}$ be the inhomogeneous Markov chains defined on $\mathsf{Y} = [\![1, n]\!] \times \mathsf{Z}$ through, for $i \in [\![3, 4]\!]$,

$$X_{2k}^{(i)} = \begin{pmatrix} M_k^{(i)} \\ Z_k^{(i)} \end{pmatrix} \xrightarrow{P_i} X_{2k+1}^{(i)} = \begin{pmatrix} \check{M}_{k+1}^{(i)} \\ \check{Z}_{k+1}^{(i)} \end{pmatrix}$$
$$\xrightarrow{Q_i} X_{2k+2}^{(i)} = \begin{pmatrix} M_{k+1}^{(i)} \\ Z_{k+1}^{(i)} \end{pmatrix} \xrightarrow{P_i} \cdots \tag{10}$$

Here we have introduced the kernels

- $P_3((m, z), d\check{m} \times d\check{z}) := \int \cdots \int \left(\prod_{j \neq m} \rho_j(du_j) \right) \delta_z(du_m) \pi(d\check{m} \,|\, u) \delta_{u_{\check{m}}}(d\check{z})$,
- $P_4 := P_3$,
- $Q_3((\check{m}, \check{z}), dm \times dz) := \delta_{\check{m}}(dm) K_{\check{m}}(\check{z}, dz)$ (where $K_{\check{m}}$ is defined in (5)),
- $Q_4((\check{m}, \check{z}), dm \times dz) := \delta_{\check{m}}(dm) \delta_{\check{z}}(dz)$.

Setting $Y_k^{(i)} := (M_k^{(i)}, Z_k^{(i)})$, $k \in \mathbb{N}$, $i \in [\![1, 2]\!]$, it can be checked easily that $\{Y_k^{(3)}; k \in \mathbb{N}\}$ and $\{Y_k^{(4)}; k \in \mathbb{N}\}$ have indeed exactly the same distribution as the output of Algorithm 3 and Algorithm 4, respectively. Using the decomposition (10), we may obtain the following results.

(A2) It holds that

$$\sup_{(m,u) \in \mathsf{Y}} \omega(m, u) < \infty,$$

where $\omega(m, u) := \pi^*(m, u) / \rho_m(u)$.

Theorem 6 *The Markov chains $\{Y_k^{(3)}; k \in \mathbb{N}\}$ and $\{Y_k^{(4)}; k \in \mathbb{N}\}$ generated by Algorithms 3 and 4, respectively, have π^* as invariant distribution. Moreover, under* (**A2**)*, the chains are uniformly ergodic.*

Proof The first part of the theorem is established by noting that Q_4 defined above is reversible with respect to any distribution, and in particular it is π^*-reversible. Moreover, according to Lemma 8 (below), $P_3 = P_4$ and Q_3 are also π^*-reversible. This completes the proof of the first part.

To prove the second part, note that for each $i \in [\![3, 4]\!]$, the transition kernel of $\{Y_k^{(i)}; k \in \mathbb{N}\}$ is $P_i Q_i$. Moreover, since $P_4 Q_4 = P_4 = P_3$, it is enough to establish that P_3 and $P_3 Q_3$ are uniformly ergodic. By definition, for all $(m, u_m) \in \mathsf{Y}$,

$$P_3((m, u_m), d\ell \times dv) = \sum_{k \in [\![1,n]\!]} \int \cdots \int \left(\prod_{i \neq m} \rho_i(du_i) \right) \pi(k \mid u) \delta_k(d\ell) \delta_{u_k}(dv), \tag{11}$$

and plugging the expression

$$\pi(k \mid u) = \frac{\omega(k, u_k)}{\sum_{j=1}^{n} \omega(j, u_j)},$$

where ω is defined in (**A2**), into (11) yields

$$\begin{aligned} & P_3((m, u_m), d\ell \times dv) \\ & \quad = \sum_{k \neq m} \kappa(m, u_m, k, v) \nu_k(d\ell \times dv) \\ & \qquad + \check{\kappa}(m, u_m) \delta_{(m,u)}(d\ell \times dv), \end{aligned} \tag{12}$$

where we have defined

$$\begin{aligned} & \nu_k(d\ell \times dv) := \delta_k(d\ell)\nu(dv), \\ & \kappa(m, u_m, k, u_k) := \rho_k(u_k) \int \prod_{i \neq m, i \neq k} \rho_i(du_i) \frac{\omega(k, u_k)}{\sum_j \omega(j, u_j)}, \\ & \check{\kappa}(m, u_m) := \int \prod_{i \neq m} \rho_i(du_i) \frac{\omega(m, u_m)}{\sum_j \omega(j, u_j)}. \end{aligned}$$

Note that (**A2**) implies

$$\underline{\kappa}(k, v) := \inf_{(m,u) \in \mathsf{Y}} \kappa(m, u, k, v) > 0; \tag{13}$$

thus, for all $(u, \mathsf{A}) \in \mathsf{Z} \times \mathcal{Y}$, we obtain

$$\begin{aligned} & P_3^2((1, u), \mathsf{A}) \\ & \quad \geq \iint P_3((1, u), dk \times dw) \mathbb{1}_{\{2\}}(k) \\ & \qquad \times P_3((2, w), d\ell \times dv) \mathbb{1}_{\{1\}}(\ell) \mathbb{1}_{\mathsf{A}}(\ell, v) \\ & \quad \geq \int \underline{\kappa}(2, w)\nu(dw) \int_{\mathsf{A}} \underline{\kappa}(1, v)\nu_1(d\ell \times dv). \end{aligned} \tag{14}$$

Now, fix $m \neq 1$; then, reusing (13) yields

$$\begin{aligned} & P_3^2((m, u), \mathsf{A}) \\ & \quad \geq \iint P_3((m, u), dk \times dw) \mathbb{1}_{\{1\}}(k) \\ & \qquad \times P_3((2, w), d\ell \times dv) \mathbb{1}_{\{1\}}(\ell) \mathbb{1}_{\mathsf{A}}(\ell, v) \\ & \quad \geq \int_{\mathsf{A}} \underline{\kappa}(1, v)\check{\kappa}(1, v)\nu_1(d\ell \times dv). \end{aligned}$$

Consequently, for all $(m, u) \in \mathsf{Y}$ and all $\mathsf{A} \in \mathcal{Y}$,

$$P_3^2((m, u), \mathsf{A}) \geq \varepsilon \bar{\eta}_1(\mathsf{A}),$$

where $\bar{\eta}_1 := \eta_1/\eta_1(\mathsf{Y})$, $\varepsilon := \eta_1(\mathsf{Y})$, and

$$\begin{aligned} & \eta_1(d\ell \times dv) := \\ & \quad \left(\int \underline{\kappa}(2, w)\rho_2(dw) \wedge \check{\kappa}(1, v) \right) \underline{\kappa}(1, v)\nu_1(d\ell \times dv). \end{aligned}$$

This establishes that $P_3 = P_4 Q_4$ is uniformly ergodic.

We now turn to the product $P_3 Q_3$. By definition of Q_3, we have $Q_3((m, u), d\ell \times dv) = \delta_m(d\ell) K_m(u, dv)$. Thus, we obtain, using (5),

$$Q_3((m, u), d\ell \times dv) \geq \varrho(u, v)\nu_m(d\ell \times dv), \tag{15}$$

where by assumption,

$$\varrho(u, v) := \inf_{m \in [\![1,n]\!]} r_m(u, v)\alpha_m(u, v) > 0,$$

Using (15) twice, we obtain for all $(m, u) \in \mathsf{Y}$ and all $\mathsf{A} \in \mathcal{Y}$,

$$\begin{aligned} & (P_3 Q_3)^2((m, u), \mathsf{A}) \geq \\ & \quad \iint P_3((m, u), dk \times dw)\varrho(w, w')\nu(dw') \\ & \qquad \times P_3((k, w'), d\ell \times dw'')\varrho(w'', v)\nu(dv)\mathbb{1}_{\mathsf{A}}(\ell, v), \end{aligned} \tag{16}$$

which implies, along the lines of (14),

$$(P_3 Q_3)^2((1, u), \mathsf{A}) \geq \epsilon_1 \int_{\mathsf{A}} g_1(v)\nu_1(d\ell \times dv),$$

where

$$\begin{aligned} & \epsilon_1 := \int \underline{\kappa}(2, w)\nu_2(dw)\varrho(w, w')\nu(dw'), \\ & g_1(v) := \int \underline{\kappa}(1, w'')\varrho(w'', v)\nu(dw''). \end{aligned}$$

Similarly, for all $m \neq 1$,

$$(P_3 Q_3)^2((m, u), \mathsf{A}) \geq \int_{\mathsf{A}} g_2(v)\nu_1(d\ell \times dv),$$

where

$$\begin{aligned} g_2(v) := & \iint \underline{\kappa}(1, w)\nu(dw)\varrho(w, w')\nu(dw') \\ & \times \check{\kappa}(1, w')\varrho(w', v). \end{aligned}$$

Thus, along previous lines, for all $(m, u) \in \mathsf{Y}$ and all $\mathsf{A} \in \mathcal{Y}$,

$$(P_3 Q_3)^2((1, u), \mathsf{A}) \geq \epsilon \bar{\eta}_2(\mathsf{A}),$$

where $\bar{\eta}_2 := \eta_2/\eta_2(\mathsf{Y})$, $\epsilon := \eta_2(\mathsf{Y})$, and

$$\eta_2(d\ell \times dv) := (\epsilon_1 g_1(v)) \wedge g_2(v)\nu_1(d\ell \times dv),$$

implying that $P_3 Q_3$ is uniformly ergodic. The proof is completed. □

Theorem 7 *Let $\{X_k^{(i)}; k \in \mathbb{N}\}$, $i \in [\![3, 4]\!]$, be the Markov chains (10) starting with $X_0^{(i)} \sim \pi^*$ for $i \in [\![3, 4]\!]$. Then, under (**A2**), for all real-valued functions h on $[\![1, n]\!]$, it holds that*

$$\lim_{n \to \infty} \frac{1}{n} \mathrm{Var} \left(\sum_{k=1}^{n} h(M_k^{(3)}) \right) \leq \lim_{n \to \infty} \frac{1}{n} \mathrm{Var} \left(\sum_{k=1}^{n} h(M_k^{(4)}) \right).$$

Proof By Theorem 6, the processes $\{X_k^{(1)}; k \in \mathbb{N}\}$ and $\{X_k^{(2)}; k \in \mathbb{N}\}$ are both inhomogeneous Markov chains that evolve alternatingly according to the π^*-reversible kernels P_i and Q_i, $i \in [\![1, 2]\!]$, respectively. Moreover, $P_3 = P_4 \succeq P_4$, and since Q_4 has no off-diagonal component, $Q_3 \succeq Q_4$. Now, define Markov chains $\{X_k^{(i)}; k \in \mathbb{N}\}$, $i \in [\![3, 4]\!]$, as in (10) with $X_0^{(i)} \sim \pi$, and set $f(m, z) \equiv h(m)$. By construction, $\check{M}_k^{(i)} = M_k^{(i)}$ for all $(i, k) \in [\![3, 4]\!] \times \mathbb{N}^*$, implying that

$$\sum_{k=1}^{\infty} \left(|\mathrm{Cov}(f(X_0^{(i)}), f(X_k^{(i)}))| + |\mathrm{Cov}(f(X_1^{(i)}), f(X_{k+1}^{(i)}))| \right)$$
$$= \pi f^2 - \pi^2 f + 4 \sum_{k=1}^{\infty} |\mathrm{Cov}(h(M_0^{(i)}), h(M_k^{(i)}))| < \infty$$
$$(i \in [\![3, 4]\!]), \tag{17}$$

where finiteness follows since $\{M_k^{(i)}; k \in \mathbb{N}\}$ are the index components of the Markov chain $\{Y_k^{(i)}; k \in \mathbb{N}\}$, which is uniformly ergodic by Theorem 6, and since the function h is bounded (being defined on a finite set). Moreover, for all $n \in \mathbb{N}^*$,

$$\mathrm{Var}\left(\sum_{k=1}^{n} h(M_k^{(i)})\right) = \mathrm{Var}\left(\sum_{k=1}^{n} h(\check{M}_k^{(i)})\right)$$
$$= \frac{1}{4}\mathrm{Var}\left(\sum_{k=1}^{2n} f(X_k^{(i)})\right) \quad (i \in [\![3, 4]\!]),$$

implying, by (17), that

$$\lim_{n\to\infty} \frac{1}{n}\mathrm{Var}\left(\sum_{k=1}^{n} h(M_k^{(i)})\right)$$
$$= \frac{1}{2} \lim_{n\to\infty} \frac{1}{n}\mathrm{Var}\left(\sum_{k=1}^{n} f(X_k^{(i)})\right) \quad (i \in [\![3, 4]\!]).$$

Finally, by (17) we may apply Theorem 4 to the chains $\{X_k^{(i)}; k \in \mathbb{N}\}$, $i \in [\![0, 1]\!]$, which establishes immediately the statement of the theorem. □

Lemma 8 *The Markov kernels P_3 and Q_3 are both π^*-reversible.*

Proof Write, using the identity

$$\nu(dz)\delta_z(du_m)\delta_{u_{\check{m}}}(d\check{z}) \prod_{j \neq m} \nu(du_j)$$
$$= \delta_{u_m}(dz)\delta_{u_{\check{m}}}(d\check{z}) \prod_{j=1}^{n} \nu(du_j),$$

for any nonnegative measurable function f on $(\mathsf{Y}, \mathcal{Y})$,

$$\iint f(y, \check{y})\pi^*(dy)P_3(y, d\check{y})$$
$$= \int \cdots \int f(y, \check{y})\pi^*(m, z)|dm|\nu(dz)\delta_z(du_m)$$
$$\times \left(\prod_{j \neq m} \rho_j(du_j)\right) \frac{\pi^*(\check{m}, u_{\check{m}}) \prod_{j \neq \check{m}} \rho_j(u_j)}{\sum_{k=1}^{n} \pi^*(k, u_k) \prod_{\ell \neq k} \rho_\ell(u_\ell)}$$
$$\times |d\check{m}|\delta_{u_{\check{m}}}(d\check{z})$$
$$= \int \cdots \int f(y, \check{y})\pi^*(m, z)\pi^*(\check{m}, u_{\check{m}})$$
$$\times \frac{\prod_{j \neq m} \rho_j(u_j) \prod_{j \neq \check{m}} \rho_j(u_j)}{\sum_{k=1}^{n} \pi^*(k, u_k) \prod_{\ell \neq k} \rho_\ell(u_\ell)}$$
$$\times \left(\prod_{j=1}^{n} \nu(du_j)\right) |dm||d\check{m}|\delta_{u_m}(dz)\delta_{u_{\check{m}}}(d\check{z}).$$

Thus, integrating first over z and $\check{z}$ and defining

$$A(m, \check{m}, u)$$
$$:= \pi^*(m, u_m)\pi^*(\check{m}, u_{\check{m}}) \frac{\prod_{j \neq m} \rho_j(u_j) \times \prod_{j \neq \check{m}} \rho_j(u_j)}{\sum_{k=1}^{n} \pi^*(k, u_k) \prod_{\ell \neq k} \rho_\ell(u_\ell)}$$

yields

$$\iint f(y, \check{y})\pi^*(dy)P_2(y, d\check{y})$$
$$= \int \cdots \int f((m, u_m), (\check{m}, u_{\check{m}}))A(m, \check{m}, u)$$
$$\times \prod_{j=1}^{n} \nu(du_j)|dm||d\check{m}|.$$

Now, the symmetry $A(m, \check{m}, u) = A(\check{m}, m, u)$ implies the identity

$$\iint f(y, \check{y})\pi^*(dy)P_3(y, d\check{y}) = \iint f(y, \check{y})\pi^*(d\check{y})P_3(\check{y}, dy), \tag{18}$$

and as f was chosen arbitrarily, (18) implies that

$$\pi^*(dy)P_3(y, d\check{y}) = \pi^*(d\check{y})P_3(\check{y}, dy),$$

which establishes the π^*-reversibility of P_3.

We show that Q_3 is π^*-reversible. Again, let f be some nonnegative measurable function on $(\mathsf{Y}, \mathcal{Y})$. Then, using that K_m is reversible with respect to $\pi^*(dz \mid m)$ for all $m \in [\![1, n]\!]$, we obtain, denoting $\check{y} := (\check{m}, \check{z})$ and $y := (m, z)$,

$$\int \cdots \int f(\check{y}, y)\pi^*(d\check{y})K_{\check{m}}(\check{z}, dz)\delta_{\check{m}}(dm)$$
$$= \int \cdots \int f(\check{y}, y)\pi^*(d\check{m})\pi^*(d\check{z} \mid \check{m})K_{\check{m}}(\check{z}, dz)\delta_{\check{m}}(dm)$$

$$= \int \cdots \int f(\check{y}, y)\pi^*(d\check{m})\pi^*(dz \mid \check{m})K_{\check{m}}(z, d\check{z})\delta_{\check{m}}(dm)$$
$$= \int \cdots \int f(\check{y}, y)\pi^*(dm)\pi^*(dz \mid m)K_m(z, d\check{z})\delta_m(d\check{m}).$$

This implies

$$\iint f(\check{y}, y)\pi^*(dy)Q_3(y, d\check{y})$$
$$= \iint f(\check{y}, y)\pi^*(d\check{y})Q_3(\check{y}, dy),$$

which completes the proof. □

5 Numerical illustrations

In this section we compare numerically the performances of the different algorithms described in the previous section. The comparisons will be based on three different models: first, a simple toy model consisting of a mixture of two Gaussian strata; second, a model where only partial observations of the mixture variables are available; third, a real-world high-dimensional missing data inference problem where the posterior distribution of a class index and a deformation field is estimated given a set of template patterns and a specific observation. All implementations are in MATLAB, running on a MacBook Air with a 1.8 GHz Inter Core i7 processor (for the first two examples) and a Dell Precision T1500 with a 3.3 GHz Inter Core i5 processor (for the third example).

5.1 Mixture of Gaussian strata

Let $\mathsf{Y} = [\![1, 2]\!] \times \mathbb{R}$ (i.e., $\mathsf{Z} = \mathbb{R}$ in this case) and consider a pair of random variables (M, Z) distributed according to the Gaussian mixture model

$$\pi^*(m, z) = \frac{1}{2}\phi(z; \mu_m, \sigma^2) \quad ((m, z) \in \mathsf{Y}), \tag{19}$$

where $\sigma > 0$, $(\mu_1, \mu_2) = (-1, 1)$, and $\phi(z; \mu, \sigma^2)$ denotes the Gaussian probability density function with mean μ and variance σ^2. Even though it is straightforward to generate i.i.d. samples from this simple toy model, we use it for illustrating and comparing the performances of the algorithms proposed in the previous; in particular, since $\pi^*(z \mid m)$ is simply a Gaussian distribution in this case, it is possible to execute Step (iii) in Algorithm 2 (which is, as mentioned, far from always the case; see the next example). For small values of σ, such as the value $\sigma = \sqrt{.2}$ used in this simulation, the two modes are well-separated, implying a strong correlation between the discrete and continuous components. As a consequence we may expect the naive Gibbs sampler to exhibit a very sub-optimal performance in this case. In order to improve mixing we introduced Gaussian pseudo-priors

$$\rho_\ell(u) := \phi(u; \tilde{\mu}_\ell, \tilde{\sigma}_\ell^2) \quad ((\ell, u) \in [\![1, 2]\!] \times \mathbb{R})$$

on $\mathbb{R}$, where $(\tilde{\mu}_1, \tilde{\mu}_2, \tilde{\sigma}_1^2, \tilde{\sigma}_2^2) = (-.5, .5, .205, .205)$, and executed, using these pseudo-priors, Algorithms 2, 3, and 4. Note that the slight over-dispersion of the pseudo-priors provides a model that satisfies (**A2**), which allows Theorem 7 to be used for comparing theoretically the algorithms. Moreover, the naive Gibbs sampler was implemented for comparison. Algorithm 3 used the proposal

$$R_\ell(u, dz) = \rho_\ell(dz) \quad ((\ell, u) \in [\![1, 2]\!] \times \mathbb{R}),$$

yielding an algorithm that can be viewed as a hybrid between Algorithms 2 and 4 in the sense that it "refreshes" randomly the continuous component $U_{m'}$ obtained after Step (ii) by replacing, with the Metropolis-Hastings probability $\alpha_{m'}$, the same by a draw from ρ'_m. Cf. Algorithms 2 and 4, where $U_{m'}$ is refreshed systematically according to ρ'_m and kept frozen, respectively. For each of these algorithms we generated an MCMC trajectory comprising 301,000 iterations (where the first 1,000 iterations were regarded as burn-in and discarded) and estimated the corresponding autocorrelation functions. The outcome, which is displayed in Fig. 2 below, indicates increasing autocorrelation for the CC, MCC, FCC, and Gibbs algorithms, respectively, confirming completely the theoretical results obtained in the previous section. Interestingly, the FCC algorithm has, despite being close to twice as efficient in terms of CPU with our implementation, only slightly higher autocorrelation than the MCC algorithm for both the components; indeed, for the mixture index component the plots of the corresponding estimated autocorrelation functions are more or less indistinguishable. As expected, the Gibbs sampler suffers from very large autocorrelation as it tends to get stuck in the different modes, while the CC algorithm has the highest performance at a computational complexity that is comparable to that of the FCC algorithm in this case (due to MATLAB's very efficient Gaussian random number generator). Qualitatively, similar outcomes are obtained if the parametrisations of the target distribution or the pseudo-priors are changed.

5.2 Partially observed mixture variables

In this example we consider a model with two layers, where a pair $Y = (M, Z)$ of random variables, forming a mixture model $\tilde{\pi}$ on $\mathsf{Y} = [\![1, n]\!] \times \mathsf{Z}$ of the form described in Sect. 2.2, is only *partially observed* through some random variable X taking values in some other state space $(\mathsf{X}, \mathcal{X})$. More specifically, we assume that the distribution of X conditionally on Y is given by some Markov transition density g on $\mathsf{Y} \times \mathsf{X}$, i.e.,

$$X \mid (M, Z) \sim g((M, Z), x)\lambda(dx),$$

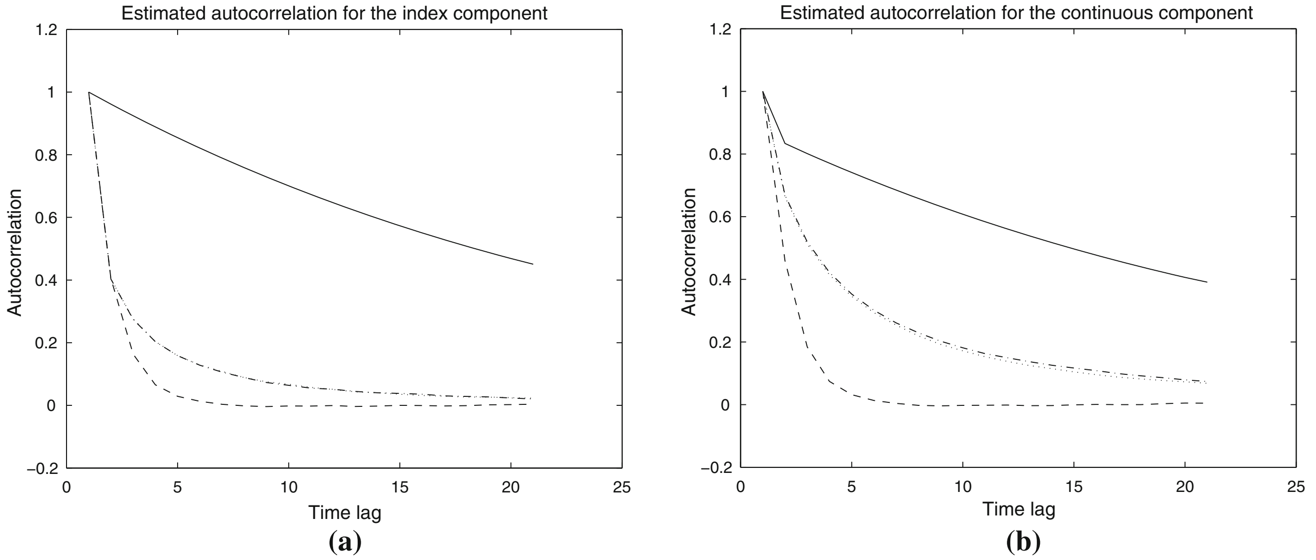


Fig. 2 Plot of estimated autocorrelation for the standard Gibbs sampler (*solid line*), Algorithm 2 (*dashed line*), Algorithm 3 (*dotted line*), and Algorithm 4 (*dash-dotted line*) when applied to the model (19). The estimates are based on 300,000 MCMC iterations, **a** M-component, **b** Z-component

Table 1 Specifications of the two setups studied in Sect. 5.2

setup	Gaussian mixture model	Deformable template model
Y	$\llbracket 1, 2\rrbracket$	$\llbracket 1, 4\rrbracket$
Z	$\mathbb{R}$	$\mathbb{R}^{72}$
X	$\mathbb{R}$	$\mathbb{R}^{16} \times \mathbb{R}^{16}$
Observation model	$X = Z^2 + \varsigma\varepsilon, \varepsilon \sim \mathcal{N}(0, 1)$	$X = f(M, Z) + \varsigma\varepsilon, \varepsilon \sim \mathcal{N}(0, \mathbf{I}_{\lvert\mathsf{X}\rvert})$
Observed data x	.4	A random handwritten digit “5”
Parameters	$\sigma = \sqrt{.2}, \varsigma = \sqrt{.1}$	$\sigma^2 = 2\mathbf{I}_{\lvert\mathsf{Z}\rvert}, \varsigma = \sqrt{2}$
Pseudo-prior design	Prior distribution $\{\tilde{\pi}(\cdot \mid m)\}_{m=1}^{2}$	Markov chain-based method

where λ is some reference measure on $(\mathsf{X}, \mathcal{X})$. When operating on a model of this form one is typically interested in computing the conditional distribution of the latent variable Y given some distinguished value $X = x \in \mathsf{X}$ of the observed variable. This posterior distribution has the density

$$\pi^*(m, z \mid x) = \frac{g((m, z), x)\tilde{\pi}(m, z)}{\iint g((m, z), x)\tilde{\pi}(m, z)|dm|\nu(dz)} \quad ((m, z, x) \in \mathsf{Y} \times \mathsf{X})$$

w.r.t. the product measure $|dm|\nu(dz)$. Since the observation x is fixed, we simply omit this quantity from the notation and write $\pi^*(m, z \mid x) = \pi^*(m, z)$. Note that π^* is again a mixture model on Y, and our objective is to sample from this distribution.

In order to evaluate, in this framework, the performances of the MCMC samplers discussed in the previous section we consider two different setups, namely

1. a Gaussian mixture model related to the toy example in Sect. 5.1.
2. a real-world image analysis problem consisting in sampling jointly a high-dimensional warping parameter and a cluster index, parameterising typically a mixture of deformable template models (Allassonnière et al. 2007).

The two setups are summarised in Table 1.

5.2.1 Gaussian mixture model

Let, as in the previous example, $\mathsf{Y} = \llbracket 1, 2\rrbracket \times \mathbb{R}$, and consider the Gaussian mixture model

$$\tilde{\pi}(m, z) = \alpha_m \phi\left(z; \mu_m \sigma^2\right) \quad ((m, z) \in \llbracket 1, 2\rrbracket \times \mathbb{R}), \tag{20}$$

where $\alpha_1 = 1/4$, $\alpha_2 = 3/4$, $\mu_1 = -1$, $\mu_2 = 1$, and $\sigma = \sqrt{.2}$. (Note that letting $\alpha_1 = \alpha_2 = 1/2$ yields the mixture model (19) of the previous example.) In addition, we let (M, Z) be partially observed through

$$X = Z^2 + \varsigma\varepsilon, \tag{21}$$

where $\varsigma = \sqrt{.1}$ and ε is a standard Gaussian noise variable which is independent of Z. Consequently, the measurement density (with respect to Lebesgue measure) is given by $g((m, z), x) = \phi(x; z^2, \varsigma^2)$, $x \in \mathbb{R}$, in this case. For the fixed observation value $x = .4$ we sampled from the posterior distribution

$$\pi^*(m, z) \propto \alpha_m \phi(z; \mu_m, \sigma^2)\phi(x; z^2, \varsigma^2) \quad ((m, z) \in \llbracket 1, 2 \rrbracket \times \mathbb{R})$$

and estimated the posterior index probability $\alpha_2^* = \pi^*(m)|_{m=2}$ and the posterior mean $\mu^* := \int z\pi^*(dz)$ using Algorithms 3 and 4. Note that we are unable to sample directly the conditional distribution $\pi^*(z \mid m)$ in this case due to the nonlinearity of the observation equation (21); thus, Algorithm 2 is excluded from our comparison. In addition, we implemented the Gibbs sampler in Algorithm 1 with Step (ii) replaced by a Metropolis-Hastings operation, yielding a Metropolis-within-Gibbs (MwG) sampler. This Metropolis-Hastings operation as well as in the corresponding operation in Step (iii) of the MCC sampler (Algorithm 3) used the conditional prior distribution as proposal, e.g.,

$$R_\ell(u, dz) = \tilde{\pi}(dz \mid \ell) \quad ((\ell, u) \in \llbracket 1, 2 \rrbracket \times \mathbb{R}).$$

This distribution was also used for designing the pseudo-priors in the MCC and FCC algorithms, e.g.,

$$\rho_\ell(dz) = \tilde{\pi}(dz \mid \ell) \quad ((\ell, u) \in \llbracket 1, 2 \rrbracket \times \mathbb{R}),$$

and consequently the MCC sampler can, as in the previous example, be viewed as a "random refreshment"-version (using the terminology of Maire et al. (2014)) of the FCC sampler. We note that $\omega(m, u) \propto \alpha_m \phi(x; u^2, \varsigma^2)$ is uniformly bounded in $(m, u) \in \llbracket 1, 2 \rrbracket \times \mathbb{R}$, providing a model satisfying (**A2**); we may hence compare theoretically the algorithms using Theorem 7.

After prefatory burn-ins comprising 1,000 iterations, trajectories of length 300,000 were generated using each algorithm. The resulting autocorrelation function estimates are displayed in Fig. 3, which shows that the FCC and MCC algorithms are clearly superior, in terms of autocorrelation, to the MwG sampler. Even though the MCC sampler has, as expected from Theorem 7, a small advantage to the FCC sampler in terms of autocorrelation, both samplers exhibit very similar mixing properties. This is particularly appealing in the light of the CPU times reported in Table 2, which shows that the FCC sampler is almost twice as fast as the MCC sampler for our implementation. Tables 2 and 3 report also the posterior mean and probability estimates obtained using the output of the different algorithms, and apparently the slow mixing of the MwG sampler rubs off on the precision of the corresponding estimate. The true values $\alpha_2^* = .750$ (which is very close to the corresponding prior probability α_2 for the given observation $x = .4$) and $\mu^* = .315$ were obtained using numerical integration. The asymptotic variance estimates of the different samplers were obtained using the method of *overlapping batch means* (see Meketon and Schmeiser (1984)), where each batch contained 4,481 ($= \lfloor 300{,}000^{2/3} \rfloor$) values (this batch size is consistent with the recommendations of [Flegal and Jones (2010), Section 4]), and as a measure of precision per computational effort we defined *efficiency* as inverse asymptotic standard error over computational time. As clear from Tables 2 and 3, MCC and FCC have, as expected, a clear advantage over MwG in terms of efficiency for both estimators. Moreover, FCC is about 1.8 and 1.5 times more efficient than MCC for the discrete and continuous components, respectively.

Table 2 Estimates of the posterior index probability α_2^* delivered by the MwG, MCC, and FCC algorithms for the partially observed mixture model (21) together with the corresponding asymptotic standard errors, CPU times, and efficiencies (inverse standard error per unit CPU time)

Algorithm	Estimate	SE	Time (s)	Efficiency
MwG	.746	.0466	191	.112
MCC	.747	.0120	194	.429
FCC	.750	.0116	112	.774

The true posterior probability (for $x = .4$) is $\alpha_2^* = .750$

Table 3 Estimates of the posterior mean μ^* delivered by the MwG, MCC, and FCC algorithms for the partially observed mixture model (21) together with the corresponding asymptotic standard errors, CPU times, and efficiencies (inverse standard error per unit CPU time)

Algorithm	Estimate	SE	Time (s)	Efficiency
MwG	.311	.0618	191	.0847
MCC	.311	.0157	194	.329
FCC	.315	.0185	112	.485

The true posterior mean (for $x = .4$) is $\mu^* = .315$

Figure 4 displays the estimate of the marginal posterior density $\pi^*(z)$ obtained by applying a Gaussian kernel smoothing function to the output of the FCC algorithm. The exact posterior, obtained using numerical integration, is plotted for comparison.

5.2.2 *Sampling a high-dimensional warping parameter for handwritten digits*

We now consider the problem of warping a random observation of a 16 × 16 pixels handwritten digit "5" from the MNIST dataset (LeCun and Cortes 2010) with a (known) collection of prototype patterns, referred to as *templates*, of the digit in question; see Fig. 5a. In this setting, M, taking values in $\llbracket 1, 4 \rrbracket$, is the template number and Z, taking values in $\mathsf{Z} = \mathbb{R}^{72}$, is the warping parameter. As in the previous example, the variables (M, Z) are only partially observed

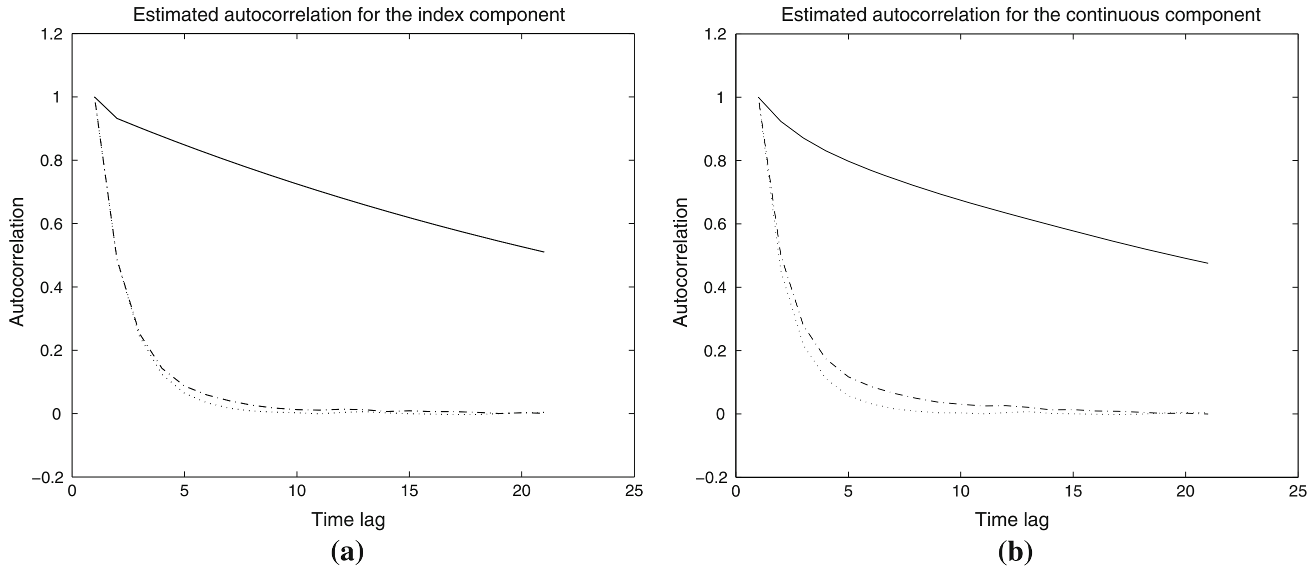


Fig. 3 Plot of estimated autocorrelation for the Metropolis-within-Gibbs sampler (*solid line*), Algorithm 3 (*dotted line*), and Algorithm 4 (*dash-dotted line*) when applied to the model (21), **a** M-component, **b** Z-component

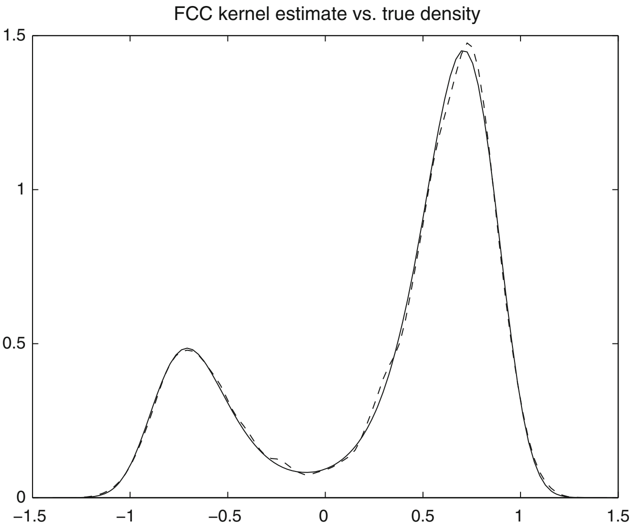


Fig. 4 Probability density estimate based on the sequence $\{Z_k^{(4)}; k \in [\![1{,}001, 10^5]\!]\}$ generated by Algorithm 4 (*dashed line*) for the partially observed mixture model (21) together with the exact posterior density (*solid line*)

through a single data point, namely the digit displayed in Fig. 5b, and we impose the same prior distribution as in (20), with $\alpha_m = 1/4$ and $\mu_m = \mathbf{0}_{|Z|}$ for all $m \in [\![1, 4]\!]$. We refer to Table 1 for a detailed comparison between this model and the model of the previous example.

The latent data (M, Z) are only partially observed through the deformable template model

$$X = f(M, Z) + \varsigma\varepsilon, \tag{22}$$

where $f : [\![1, 4]\!] \times \mathbb{R}^{72} \to \mathbb{R}^{16} \times \mathbb{R}^{16}$ is a deterministic mapping distorting the Mth template by the deformation parameterised by Z; see, e.g., Allassonnière et al. (2007) and the references therein for more details.

We aim at comparing the performances of the MwG, MCC, and FCC algorithms for the task of sampling the posterior distribution

$$\pi^*(m, z) \propto \alpha_m \phi(z; \mu_m, \sigma^2)\phi(x; f(m, z), \varsigma^2)$$
$$((m, z) \in [\![1, 4]\!] \times \mathbb{R}^{72}).$$

Setting, as in Sect. 5.2.1, $\rho_m(dz) = \tilde{\pi}(dz \mid m)$ for all $m \in [\![1, 4]\!]$ turns out to be too naive in this case; indeed, the high dimension of Z makes the prior highly non-informative, and the latter therefore differs significantly from the posterior $\pi^*(dz \mid m)$ (to which the pseudo-priors should be close in the optimal scenario). Thus, the framework under consideration calls for more sophisticated design of the pseudo-priors. *Laplace's approximation* suggests that

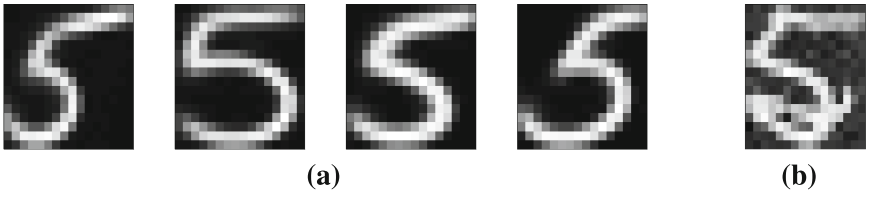


Fig. 5 Templates $\{f(m, \mathbf{0}_{|Z|})\}_{m=1}^4$ (the vector $\mathbf{0}_{|Z|}$ corresponds to no deformation) and observed digit, **a** Templates, **b** Observed handwritten digit

a Gaussian distribution with mean and covariance matrix given by

$$\begin{cases} z^*(m) := \arg\max_{z \in \mathsf{Z}} \log \pi^*(z \mid m), \\ \varsigma^2(m) := (-\nabla^2 \log \pi^*(z \mid m)|_{z=z^*(m)})^{-1}, \end{cases}$$

respectively, would provide a better proxy for $\pi^*(dz \mid m)$. However, in the deformable template model context, the function f in (22) is highly nonlinear and does not allow the log-likelihood to be maximised on closed-form. Optimisation in this space is very demanding from a computational viewpoint, and replacing $z^*(m)$ by a proxy is risky as the Hessian matrix needed for determining $\varsigma^2(m)$ is not necessarily invertible at that point.

Another alternative consists in sampling 4 independent Markov chains targeting $\pi^*(dz \mid m)$, $m \in [\![1, 4]\!]$, respectively, and to let $\rho_m(z) = \phi(z; \hat{\mu}(m), \hat{\gamma}^2(m))$, $z \in \mathsf{Z}$, for each m, where $\hat{\mu}(m)$ and $\hat{\gamma}^2(m)$ are the sampling mean and covariance estimate, respectively. Although appearing maybe not overly elegant at a first glance, this method turns out to be very useful in most situations lacking an obvious, natural choice of pseudo-priors. Moreover, since we only need a rough approximation of the posterior distribution, we can stop the simulation of these auxiliary chains after a limited number of iterations. In this setting, we stopped the Markov chains after collecting 500 samples from each chain, and discarded, as burn-ins, the first 100 samples of each chain in the estimation. Obviously, this step adds a computational burden to the main sampling scheme; nevertheless, we have in the following taken this additional time into consideration when evaluating the outcome of the simulations. Moreover, we concede that this way of designing the pseudo-priors may not be reasonable in situations when the number n of mixture components is very large; on the other hand, in such a case any CC-type approach is inappropriate due to the prohibitive number of auxiliary variables. Finally, a transition of the MwG algorithm is performed by a Metropolis-Hastings step with a Gaussian random walk proposal (with variance $\zeta^2 = .1$) for each component of the parameter. This is also the common choice of the Markov kernels R_m, $m \in [\![1, 4]\!]$, used in the MCC algorithm.

We compare the three algorithms through the estimation of the posterior weights (Table 4; Fig. 6) and the asymptotic variance of the class index (Table 5). Moreover, we compare the qualities of the image reconstructions provided by the different algorithms in two ways, namely

1. Graphically by displaying, in Fig. 8, some realisations of templates warped by the deformation parameter sampled by the three algorithms.
2. Quantitatively by plotting, first, in Fig. 7, the mapping $\mathbb{R} \ni t \mapsto S_t$, where each statistic S_t is defined by

$$S_t := \sum_{m=1}^{n} \|x - f(m, \bar{z}_m^{(t)})\|^2,$$

$$\bar{z}_\ell^{(t)} := \sum_{j \in \mathbb{N}: \tau(j) \le t} z_j \mathbb{1}_{m_j = \ell} \Bigg/ \sum_{j \in \mathbb{N}: \tau(j) \le t} \mathbb{1}_{m_j = \ell},$$

with $\tau : \mathbb{N} \to \mathbb{R}$ providing the CPU time needed for completing a given number of Markov transitions and, second, in Fig. 9, the estimated autocorrelation of one deformation parameter.

Table 4 Estimates of the posterior weights delivered by the MwG, MCC, MCC† (defined as MCC using the naive pseudo-priors), and FCC algorithms for the partially observed mixture model (22)

Algorithm \ class	$m = 1$	$m = 2$	$m = 3$	$m = 4$
MwG	.01	.00	.98	.01
MCC	.31	.12	.40	.16
MCC†	.03	.05	.90	.02
FCC	.32	.12	.39	.16

Table 5 Estimates of the class $m = 1$ posterior probability (given the observation in Fig. 5b) delivered by the MwG, MCC, and FCC algorithms together with the corresponding asymptotic standard error estimates (estimated using the method of overlapping batch means), CPU times, and efficiencies (for MwG, the efficiency had no meaning since the estimate is biased)

Algorithm	Estimate	SE	Time (s)	Efficiency
MwG	.01	.021	250	—
MCC	.31	.109	4700	.002
FCC	.32	.111	500	.018

These results confirm those obtained through the previous simulations: quantitatively, the difference in mixing between MCC and FCC is indeed minor and both these algorithms outperform significantly the MwG algorithm, which gets stuck most of the time in the class 3 (see Table 4). In this context, the huge gap in efficiency between FCC and MCC is particularly striking and stems from the high dimension of the deformation parameter, which, consequently, is very costly to sample using the Markov kernel—a burden avoided by FCC, which uses only the Gaussian pseudo-prior samples (Table 2). The FCC computational performance can also be observed graphically in Figs. 7 and 6. We conclude that FCC is a strong competitor in situations where the design of the pseudo-priors is demanding.

Qualitatively, both FCC and MCC allow deformations that are consistent with the observed data-point to be sampled, whereas MwG do so for only one template. Figure 8 shows

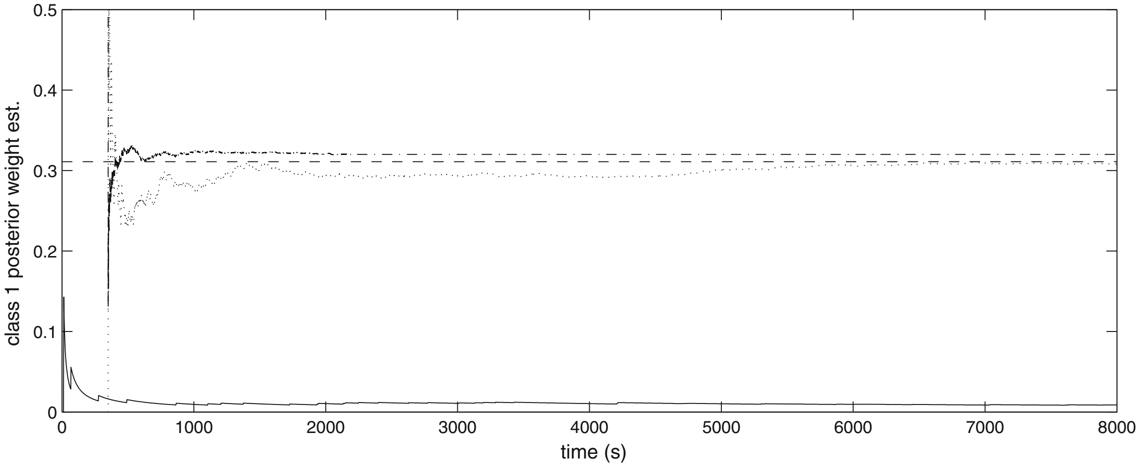


Fig. 6 Evolution of the class $m = 1$ posterior weight estimate delivered by the MwG (*solid line*), the MCC (*dotted line*), and the FCC (*dash-dotted line*) algorithms for the partially observed mixture model (22). The *dashed line* represents the class 1 asymptotic posterior weight, estimated after 100,000 iterations of MCC

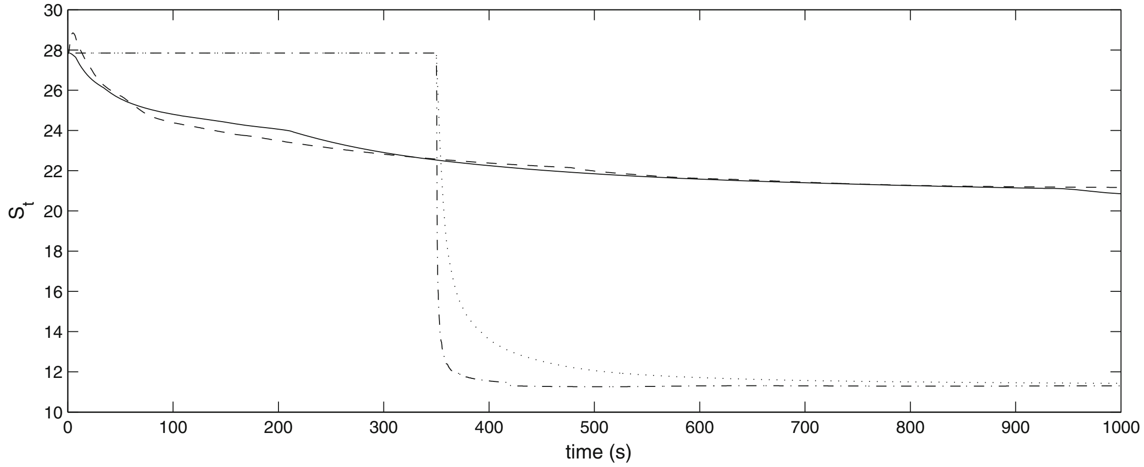
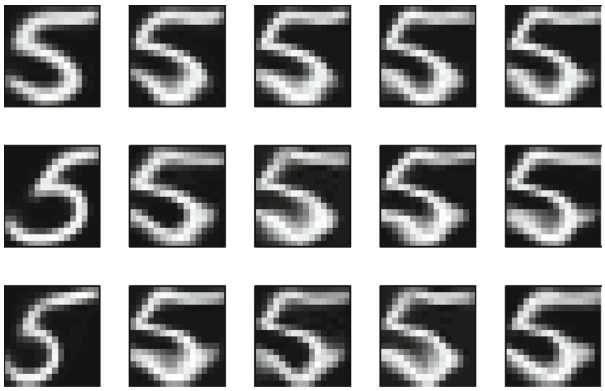


Fig. 7 Evolution of the statistic S_t as a function of the CPU time t for the MwG (*solid line*), the MCC (*dotted line*), the MCC† (*dashed line*), and FCC (*dash-dotted line*) algorithms within the framework of the partially observed mixture model (22). MCC† refers to a version of the MCC algorithm using the naive pseudo-priors



(a)

Algorithm	M_5	M_{150}	M_{500}	$M_{1,000}$	$M_{5,000}$
MwG	3	3	3	3	3
MCC	4	2	1	3	4
FCC	1	3	2	1	4

(b)

Fig. 8 Illustration of the template number and deformation parameter sampling for the MwG, MCC and FCC algorithms. On the *left hand side* (**a**), the first, second, and third rows correspond to the MwG, MCC, and FCC algorithms, respectively, while each column corresponds to a Wraped templates $f(M_k, Z_k)$ observed at different Markov chain iterations $k \in \{5, 150, 500, 1000, 5000\}$. The table on the *right hand side* (**b**) provides the template numbers sampled by each chain at the corresponding iterations

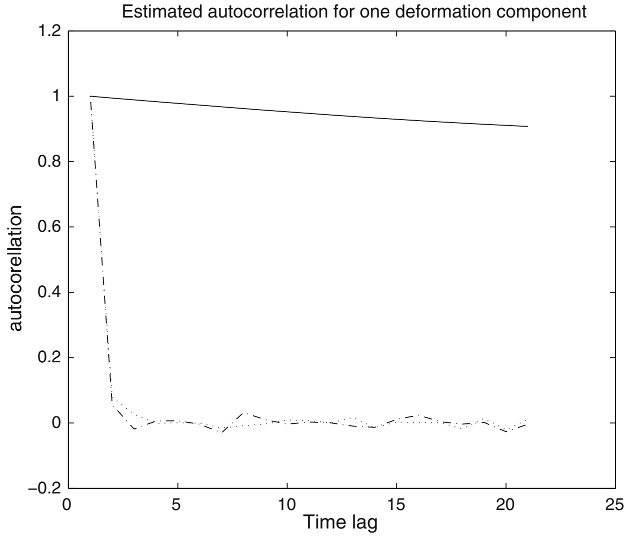


Fig. 9 Plot of estimated autocorrelation for the MwG (*solid line*), MCC (*dotted line*), and FCC(*dash-dotted line*) algorithms when applied to the model (22)

that the digit displayed in Fig. 5b can be reconstructed from any of the four templates Fig. 5a using the sampled deformations, a fact that is quantitatively confirmed using the statistic S_t Fig. 7.

Finally, note that these results depend strongly on the quality of the pseudo-priors: when using the naive pseudo-priors (defined as $\rho_m(dz) = \tilde{\pi}(dz \mid m)$ for all $m \in [\![1, 4]\!]$), the MCC behaves very similarly to the MwG; see Table 4 and Fig. 7. This shows that, in this example, the tradeoff between doing more MCMC iterations vs delaying the MCMC sampling scheme to specify decent pseudo-priors turns in favor of the latter.

Remark 9 The theoretical results developed in Theorem 7 are restricted to test functions h depending on the M-component only; however, the case of test functions depending on the Z-component (or even the pair (M, Z)), for which a comparison of the asymptotic variance between MCC and FCC is not available, is of course of interest as well. Nevertheless, the results displayed in Table 3, Fig. 3b, and Fig. 9 indicate, not surprisingly, that the mixing properties, with respect to the Z component, of the different algorithms seem to depend heavily on the ability (which is well-described by our theoretical results) of the same to move flexibly between different strata indices.

Remark 10 Following Carlin and Chib's pseudo-priors spirit, it would be possible to design, in a similar off-line scheme, a finely tuned proposal kernel to improve the Metropolis-within-Gibbs mixing performances. Such an approach is actually related to the recent development in Particle MCMC methods and in particular to the Particle independent Metropolis-Hastings sampler (PIMH) (Andrieu et al. 2010), in which a target-matching proposal is constructed from a set of weighted particles. However, adapting a PIMH algorithm to infer mixture distributions may also be difficult to setup in practice (choice of instrumental kernel, number of particles, risk of degeneracy, etc.). Moreover, a frozen equivalent to such an algorithm, which typically exists in Carlin and Chib's context thanks to the intermediate extended state space, would not exist, hence preventing to balance out the pre-processing step's computational burden as with the FCC.

6 Conclusion

We have compared some data-augmentation-type MCMC algorithms sampling from mixture models comprising a discrete as well as a continuous component. By casting Carlin & Chib's pseudo-prior into our framework we obtained a sampling scheme that is considerably more efficient than the standard Gibbs sampler, which in general exhibits poor state-space exploration due to strong correlation between the discrete and continuous components (as a result of the highly multimodal nature of the mixture model). In the case where simulation of the continuous component Z conditionally on M is infeasible, we used a metropolised version of the algorithm, referred to as the MCC sampler, that handled this issue by means of an additional Metropolis-Hastings step in the spirit of the hybrid sampler. In this case our simulations indicate, interestingly, that the loss of mixing caused by simply passing, as in the FCC algorithm, the value of the Mth auxiliary variable, generated by sampling from the pseudo-priors at the beginning of the loop, directly to Z without any additional refreshment is limited. Thus, we consider the FCC algorithm, which we proved to be π^*-reversible, as strong contender to the MCC sampler in terms of efficiency (variance per unit CPU).

Our theoretical results comparing the MCC and FCC samplers deal exclusively with mixing properties of the restriction of the MCMC output to the discrete component, and the extension of these results to the continuous component is left as an open problem. However, we believe that the discrete component is indeed the quantity of interest, as our simulations indicate that the degree mixing of the discrete component gives a limitation of the degree of mixing of the bivariate chain due to the multimodal nature of the mixture.

There are several possible improvements of the FCC algorithm. For instance, following Petralias and Dellaportas (2013), only a subset of the pseudo-priors (namely those with indices belonging to some neighborhood of the current M) could be sampled at each iteration, yielding a very efficient algorithm from a computational point of view. Such an approach could be also used for handling the case of an infinitely large index space (i.e. $n = \infty$).

Acknowledgments This work is supported by the Swedish Research Council, Grant 2011-5577.

References

Allassonnière, S., Amit, Y., Trouvé, A.: Towards a coherent statistical framework for dense deformable template estimation. J. R. Stat. Soc. Ser. B **69**(1), 3–29 (2007)

Andrieu, C., Doucet, A., Holenstein, R.: Particle Markov chain Monte Carlo methods. J. R. Stat. Soc. Ser. B **72**(3), 269–342 (2010)

Carlin, B.P., Chib, S.: Bayesian model choice via Markov chain Monte Carlo methods. J. R. Stat. Soc. Ser. B Methodol. **57**, 473–484 (1995)

Flegal, J.M., Jones, G.L.: Batch means and spectral variance estimators in Markov chain Monte Carlo. Ann. Stat. **38**(2), 1034–1070 (2010). doi:10.1214/09-AOS735

Green, P.J.: Reversible jump Markov chain Monte Carlo computation and Bayesian model determination. Biometrika **82**(4), 711–732 (1995)

Hurn, M., Justel, A., Robert, C.P.: Estimating mixtures of regressions. J. Comput. Gr. Stat. **12**(1), 55–79 (2003). doi:10.1198/1061860031329. http://www.tandfonline.com/doi/abs/10.1198/1061860031329

LeCun, Y., Cortes, C.: Mnist handwritten digit database. AT&T Labs [Online].http://yann.lecun.com/exdb/mnist (2010)

Maire, F., Douc, R., Olsson, J.: Comparison of asymptotic variances of inhomogeneous Markov chains with applications to Markov chain Monte Carlo methods. Ann. Stat. **42**, 1483–1510 (2014)

Meketon, M.S., Schmeiser, B.: Overlapping batch means: something for nothing? In: WSC 84: Proceedings of the 16th Conference on Winter Simulation, pp. 1722–1740. IEEE Press (1984)

Mira, A.: Ordering and improving the performance of Monte Carlo Markov chains. Stat. Sci. **16**, 340–350 (2001)

Petralias, A., Dellaportas, P.: An MCMC model search algorithm for regression problems. J. Stat. Comput. Simul. **83**(9), 1722–1740 (2013)

Robert, C.P., Casella, G.: Monte Carlo Statistical Methods. Springer, New York (2004)

Tierney, L.: A note on Metropolis-Hastings kernels for general state spaces. Ann. Appl. Probab. **8**, 1–9 (1998)