Label Switching in Bayesian Mixture Models: Deterministic Relabeling Strategies

Carlos E. RODRÍGUEZ and Stephen G. WALKER

Label switching is a well-known problem in the Bayesian analysis of mixture models. On the one hand, it complicates inference, and on the other hand, it has been perceived as a prerequisite to justify Markov chain Monte Carlo (MCMC) convergence. As a result, nonstandard MCMC algorithms that traverse the symmetric copies of the posterior distribution, and possibly genuine modes, have been proposed. To perform component-specific inference, methods to undo the label switching and to recover the interpretation of the components need to be applied. If latent allocations for the design of the MCMC strategy are included, and the sampler has converged, then labels assigned to each component may change from iteration to iteration. However, observations being allocated together must remain similar, and we use this fundamental fact to derive an easy and efficient solution to the label switching problem. We compare our strategy with other relabeling algorithms on univariate and multivariate data examples and demonstrate improvements over alternative strategies. Supplementary materials for this article are available online.

Key Words: Assignment problem; Gibbs sampler; Identifiability; Loss function.

1. INTRODUCTION

The finite mixture model with \( k \) components, and we assume \( k \) is known, is written as

\[
p(y_i | \mathbf{w}, \mathbf{\theta}) = \sum_{j=1}^{k} w_j f(y_i | \theta_j), \quad \text{independently for } i = 1, \ldots, n, \tag{1}
\]

and let \( \mathbf{\theta} = \{\theta_j\}_{j=1}^{k} \) and \( \mathbf{w} = \{w_j\}_{j=1}^{k} \). For (1) to be a density, the weights, \( \mathbf{w} \), must be nonnegative and sum to one. A useful idea when working with model (1) is to include latent allocation variables: \( \mathbf{z} = (z_i)_{i=1}^{n} \) such that given \( z_i \), the component from which \( y_i \) has been drawn is known, that is,

\[
p(y_i | \theta, z_i) = f(y_i | \theta_{z_i}), \quad \text{with } z_i \sim \mathcal{M}_k(1|w_1, \ldots, w_k). \tag{2}
\]

Carlos E. Rodríguez is Ph.D. student (E-mail: cerh2@kent.ac.uk), and Stephen G. Walker is Professor (E-mail: S.G.Walker@kent.ac.uk), School of Mathematics, Statistics and Actuarial Science, University of Kent, Canterbury, Kent CT2 7NF, UK.

© 2014 American Statistical Association, Institute of Mathematical Statistics, and Interface Foundation of North America
Journal of Computational and Graphical Statistics, Volume 23, Number 1, Pages 25–45
DOI: 10.1080/10618600.2012.735624
Color versions of one or more of the figures in the article can be found online at www.tandfonline.com/rjcgst.
Mixture models can be used to approximate irregular densities or to model heterogeneity. In a density estimation context, any distribution on the real line can be approximated to within any degree of accuracy, in the $L_1$ norm, using enough normal components; see Ferguson (1983). In this case, the flexibility of (1) is extended by allowing an infinite number of components. On the other hand, when a mixture model is used to model heterogeneity, there is a proper interpretation for (1): $k$ represents the number of clusters within the data, $w_j$ is the weight of cluster $j$ in the population, and $f(y|\theta_j)$ is a parametric distribution that models the behavior of cluster $j$. The latent allocations (2) automatically induce a clustering structure over the observations.

In a Bayesian mixture modeling setup, to perform inference for any characteristic of the components, the objective is to find the posterior distribution, and then calculate posterior summaries that are approximated via Markov chain Monte Carlo (MCMC) methods. A problem emerges when we note that the likelihood of (1) is invariant under permutation of the indices, and there are $k!$ permutations. Thus, under symmetric priors, the posterior will inherit the likelihood’s invariance. As a result, in any MCMC algorithm, labels of the components can permute multiple times between iterations of the sampler. This makes ergodic averages to estimate characteristics of the components useless. This is known as the label switching problem.

Paradoxically, as noted by Celeux, Hurn, and Robert (2000), Frühwirth-Schnatter (2001), Jasra, Holmes, and Stephens (2005), and Papastamoulis and Iliopoulos (2010), among others, label switching is a prerequisite for MCMC convergence. If there is no label switching, it means that the sampler is not exploring all the modes of the posterior distribution of (1). It should visit all the $k!$ symmetric modes. Notably, when the modes of the posterior are well separated, the standard Gibbs sampler fails the label switching test (Jasra, Holmes, and Stephens 2005, p. 55) as it becomes trapped in one of the symmetric modes. While this may be meaningful for inference, it becomes impossible to justify convergence.

To address this problem, three alternative ideas have been suggested. The first one keeps the standard Gibbs sampler and incorporates a Metropolis–Hastings move that proposes a random permutation of the labels; see Frühwirth-Schnatter (2001) and Papastamoulis and Iliopoulos (2010). The second approach is to use more sophisticated and complex MCMC methods to improve mixing of the sampler, and specifically avoiding the use of (2); see Jasra, Holmes, and Stephens (2005).

Initial attempts to “undo the label switching” were focused on imposing artificial constraints on the parameter space via the prior distribution aiming to break the symmetry of the posterior distribution and force a unique labeling; see Diebolt and Robert (1994) or Richardson and Green (1997). However, Stephens (2000) showed that these constraints do not always solve the problem. Another idea, based on identifiability constraints over the parameter space, was proposed by Frühwirth-Schnatter (2001). The main criticism for this method is the difficulty to select the adequate constraints among all the possible choices.

More elaborate solutions aim to undo the label switching deterministically. This means finding permutations of the parameters that minimize a loss function; see Celeux (1998), Stephens (2000), Nobile and Fearnside (2007), and Grün and Leisch (2009). Recent methods combine similar ideas with the use of the maximum a posteriori (MAP) estimator; see Martin, Mengersen, and Robert (2005) and Papastamoulis and Iliopoulos (2010). Other approaches focus on devising a loss function invariant to permutations of the parameters.
LABEL SWITCHING IN BAYESIAN MIXTURE MODELS

(see, e.g., Celeux, Hurn, and Robert 2000), thus solving the label switching problem immediately. The drawback of this method is that it is computationally expensive and to define such a loss function is not always possible; see Celeux, Hurn, and Robert (2000) and Jasra, Holmes, and Stephens (2005). Alternative proposals, which move away from the loss function framework, incorporate uncertainty in the estimation of the relabeling process. These methods include probabilistic relabeling and the ideas described by Yao (2012). Probabilistic relabeling strategies were first proposed by Jasra (2005) and then developed and extended by Sperrin, Jaki, and Wit (2010).

Our proposed solution to the label switching problem lies in the meaning of the relationship between the allocation variables (2) and the observations. The key is to use this relationship to incorporate the data directly within the loss function used to undo the label switching. From iteration to iteration of an MCMC algorithm, the labels of the clusters may change. But if the sampler has converged, the clusters must remain roughly the same. We use this fundamental fact to derive an easy and efficient solution to the label switching problem, and we compare our proposal with current relabeling alternatives. To assess the effect of the chosen MCMC strategy on the resulting inference, we compare results obtained via the standard Gibbs sampler against those obtained via a transdimensional MCMC algorithm.

The article is structured as follows. In Section 2, we describe the label switching problem with associated ideas. Then, in Section 3, the key ideas under a deterministic relabeling strategy are outlined. In Section 4, our relabeling algorithm is motivated and derived. In Section 5, we illustrate and compare our proposal against other relabeling methodologies. The article concludes with a discussion in Section 6.

2. LABEL SWITCHING IN MIXTURE MODELS

The ultimate objective under any relabeling algorithm is to gain identifiability for the components and then to be able to perform component-specific inference.

2.1 COMPONENT-SPECIFIC INFERENCE IN MIXTURE MODELS

The latent variables (2) introduce a natural clustering structure over the observations and are the basis for making inference about the hidden groups. Under this framework, the goal is to find the cluster from which each \( y_i \) is drawn. Hence, to group a set of observations into \( k \) clusters, we make inference about the classification probabilities of each observation. If a single best clustering is all that is needed, each observation is assigned to the cluster with the highest classification probability. In the rare case that the parameters of the mixture are known, Bayes’ theorem can be used to calculate the classification probabilities directly:

\[
p(z_i = j | y_i, w, \theta) = \frac{w_j f(y_i | \theta_j)}{\sum_{l=1}^{k} w_l f(y_i | \theta_l)},
\]

(3)

When the parameters of the mixture are unknown, we need to calculate the posterior classification probabilities:

\[
p(z_i = j | y) = \int_{\Theta} \int_{\Omega} \frac{w_j f(y_i | \theta_j)}{\sum_{l=1}^{k} w_l f(y_i | \theta_l)} p(w, \theta | y) dw d\theta,
\]

(4)
where \( y = \{y_i\}_{i=1}^n \). Using MCMC methods, (4) can be approximated via

\[
p(z_i = j | y) \approx \frac{1}{N} \sum_{t=1}^{N} \frac{w_j^t f(y_i | \theta_j^t)}{ \sum_{j=1}^{k} w_j^t f(y_i | \theta_j^t)},
\]

(5)

where \((w^t, \theta^t) \sim p(w, \theta | y)\) for \( t = 1, \ldots, N \) (\( t \) indexes the iterations of the MCMC). If we want to classify a new observation \( y^* \), with \( y^* \) conditionally independent of \( y \), then

\[
p(z^* = j | y^*, y) \propto \int \int_{\Theta \times \Omega} w_j f(y^* | \theta_j) p(w, \theta | y) dwd\theta.
\]

(6)

This can be estimated in an analogous way to (5), with obvious changes. Note that (6) tells us that to make inference for the scaled predictive of \( y^* \) is equivalent to the estimation of its classification probabilities. Finally, with the same MCMC output, posterior mean estimates for components’ parameters (or functions of them) can be approximated by averaging over the parameters identified by the index of interest:

\[
\mathbb{E}(h(w_j, \theta_j) | y) = \int \int_{\Theta \times \Omega} h(w_j, \theta_j) p(w, \theta | y) dwd\theta \approx \frac{1}{N} \sum_{t=1}^{N} h(w_j^t, \theta_j^t).
\]

(7)

Later on in the article, our aim will be to estimate classification probabilities (4), scale densities (6), and posterior means (7).

2.2 THE LABEL SWITCHING PROBLEM: A COMPLICATION FOR INFERENCE

The likelihood of a mixture model is invariant under permutations of its parameters: let \( \mathcal{P}_k \) be the set of the \( k! \) permutations of the labels \( \{1, \ldots, k\} \). If for some \( \rho \in \mathcal{P}_k \), we define \( \rho(w, \theta) := ((w_{\rho(1)}, \ldots, w_{\rho(k)}), (\theta_{\rho(1)}, \ldots, \theta_{\rho(k)}) \), it is easy to observe that for every \( \rho, \nu \in \mathcal{P}_k \),

\[
p(y | \rho(w, \theta)) = \prod_{i=1}^{n} \left[ \sum_{j=1}^{k} w_{\rho(j)} f(y_i | \theta_{\rho(j)}) \right] = \prod_{i=1}^{n} \left[ \sum_{j=1}^{k} w_{\nu(j)} f(y_i | \theta_{\nu(j)}) \right] = p(y | \nu(w, \theta)).
\]

Consequently, if the support of the parameters is the same, for example, under symmetric priors across the components, the posterior distribution will inherit the likelihood’s invariance. Hence, in an MCMC algorithm, the indices of the parameters can permute multiple times between iterations. As a result, we cannot identify the hidden groups that make (5) and all other ergodic averages to estimate characteristics of the components useless.

A popular idea when working with mixture models is to include the latent allocation variables (2). Note that, in this case, for every \( \rho, \nu \in \mathcal{P}_k \),

\[
p(y | \rho(\theta, z)) = \prod_{i=1}^{n} f(y_i | \theta_{\rho(z_i)}) = \prod_{i=1}^{n} f(y_i | \theta_{\nu(z_i)}) = p(y | \nu(\theta, z)),
\]

(8)

where for some \( \rho \in \mathcal{P}_k \), \( \rho(\theta, z) := ((\theta_{\rho(1)}, \ldots, \theta_{\rho(k)}), (\rho(z_1), \ldots, \rho(z_n))) \). Hence, given the allocations \( z \), the likelihood (8) is invariant under permutations of the parameters as well.
2.3 MCMC Convergence in Mixture Models

Under a mixture model setup, to assess convergence of an MCMC strategy, marginal posterior estimates are monitored. For the Gibbs sampler, plots of estimates show that in some cases, the labels of the components do not permute (see Celeux, Hurn, and Robert 2000, pp. 960, 962; Jasra, Holmes, and Stephens 2005, p. 55). This has been perceived as a symptom of poor mixing of the sampler.

To solve the lack of label switching, Frühwirth-Schnatter (2001) added a Metropolis–Hastings move that proposes a random permutation of the labels, which is accepted with probability one. This solution can be implemented in a postprocessing stage, without further modification of the sampler. However, it was discarded by Celeux, Hurn, and Robert (2000, p. 959). “Our insistence on searching for an algorithm that can achieve symmetry without such a move is that any concerns over convergence are not necessarily dealt by such a strategy, which simply alleviates the most obvious symptom;” and in Jasra, Holmes, and Stephens (2005, p. 55), it is stated that “this course of action is only appropriate if the posterior distribution is not genuinely multimodal (which would not be known a priori to simulation).” Both sets of authors concluded that alternative and more complex MCMC samplers should be considered. A broader discussion along the same lines can be found in Geweke (2007), who is not sympathetic to the construction of complex MCMC algorithms.

To address the convergence problem, Papastamoulis and Iliopoulos (2010) implemented at each iteration of a Gibbs sampler, a Metropolis–Hastings step that proposes a random permutation between two labels of the mixture: \( j, l \in \{1, \ldots, k\} \). The acceptance probability for this proposal is given by \( \alpha(j, l) = \min\{1, \left(\frac{w_j}{w_l}\right)^{n_j - n_l}\} \). However, this idea was designed to introduce label switching in infinite mixtures, and under this setting, the weights are weakly identifiable; see Papaspiliopoulos and Roberts (2008, pp. 178–179). This is not the case for the weights of a finite mixture model, so this proposal is an alternative to the ideas described by Frühwirth-Schnatter (2001).

The last alternative is to use a transdimensional sampler. The point here is that the moving \( k \) sampler is able to move around the modes of the posterior distribution of the fixed \( k \) model via models of lower or higher dimensions, thus taking less iterations to escape trapping states and in general mixing better within \( k \). Note that these samplers can be computationally demanding and may be inefficient, because the chain will not necessarily stay in the model of interest for the entire run; see Richardson and Green (1997), Stephens (1997, sec. 4.4.1), and Jasra, Holmes, and Stephens (2005).

A further justification for complex MCMC and transdimensional samplers is possible genuine multimodality. Genuine multimodality is more than the modes being identical up to permutation of the labels; see Stephens (1997, secs. 2.3 and 3.4.2). Hence, the argument against the standard Gibbs sampler is that in the presence of genuine multimodality, it can get trapped in one of the symmetric modes, and not visit genuine modes.

3. Deterministic Relabeling Strategies

Suppose a sample from the posterior distribution of (1) is generated: \( (w', \theta') \sim p(w, \theta | y) \), for \( t = 1, \ldots, N \). The aim under any relabeling strategy is to find a
permutation $\rho_t \in \mathcal{P}_k$ such that the permuted sample

$$
\rho_t(w^t, \theta^t) = ((w^t_{\rho(t)(1)}, \ldots, w^t_{\rho(t)(k)}), (\theta^t_{\rho(t)(1)}, \ldots, \theta^t_{\rho(t)(k)})�)
$$

for $t = 1, \ldots, N$, recovers identifiably for the mixture components.

An obvious way to proceed is to find artificial constraints over the parameter space of $(w, \theta)$. It is important to note that, in this case, the permutation $\rho_t \in \mathcal{P}_k$ will be the one for which the constraints are satisfied. We can start with basic constraints and then use summaries of the MCMC output to update them; see Frühwirth-Schnatter (2001). While this might work in the univariate case, it is challenging to extend to the multivariate setting. Thus, the objective is to find constraints automatically via deterministic algorithms. This is reduced to finding the permutation $\rho_t$ that minimizes a loss function. It is important to stress that such a strategy can be implemented as an online method or postprocessing the sample, without modifying the MCMC, hence allowing us to make inference with the permuted MCMC sample. For a formal justification, see Stephens (1997, proposition 3.1).

The key points of any deterministic relabeling strategy are the definition of the loss function, the selection of a pivot, and the minimization step. Let $L_t$ be the loss function to minimize, and $c_{l,j}^t$ be the cost associated if permutation $\rho_t(l) = j$ is chosen. The minimization step for the sampled values at iteration $t$ of the MCMC can be formulated via the well-known assignment problem (see Burkard, Dell’Amico, and Martello 2009):

$$
\text{Minimize } L_t = \sum_{l=1}^k \sum_{j=1}^k a_{l,j} c_{l,j}^t \tag{10}
$$

subject to $\sum_{j=1}^k a_{l,j} = \sum_{l=1}^k a_{l,j} = 1$ $(l, j = 1, \ldots, k)$, $a_{l,j} \in \{0, 1\}$ $(l, j = 1, \ldots, k)$.

We need to solve (10) by minimizing over the $(a_{l,j})$; if for some $t$, $(\hat{a}_{l,j})$ is an optimal solution for the problem and it is that $\hat{a}_{l,j} = 1$, then this corresponds to the permutation $\rho_t(l) = j$. Also, let $\hat{L}_t$ be the the loss function (10) evaluated at $(\hat{a}_{l,j})$. This formulation was first used by Stephens (1997). In the next section, we describe the algorithms, loss functions, and pivots used by Stephens (2000) and Papastamoulis and Iliopoulos (2010).

### 3.1 Kullback–Leibler Relabeling

The idea behind Stephens’ Kullback–Leibler (KL) relabeling is straightforward; see Stephens (2000). First, during the MCMC algorithm, the classification probabilities are stored. Let $P'_{n \times k}$ be the matrix of classification probabilities (at iteration $t$ of the MCMC), that is,

$$
P' = \{\{p'_{i,j}\}_{j=1}^k \}_{i=1}^n \text{ with } p'_{i,j} = \frac{w'_i f(y_i|\theta'_j)}{\sum_{l=1}^k w'_i f(y_i|\theta'_l)}.
$$

Second, let $Q_{n \times k}$ be the matrix of “true classification probabilities.” Under Stephens’ setting, the matrix $Q$ is the pivot, and the goal is to find a permutation $\rho_t \in \mathcal{P}_k$, of the
columns of $P'$: $\rho_t(P') := \{(p'_{i,\rho_t(j)})_{j=1}^k \}_{i=1}^n$ such that the KL divergence between $\rho_t(P')$ and $Q$ is minimum over all permutations. Clearly, $Q$ is not known; thus, it is approximated via a recursive algorithm. We provide a description of this strategy in Algorithm 1.

**Algorithm 1** KL relabeling strategy

1. Initialize $\rho_1, \ldots, \rho_N$. {Usually set to the identity: $\rho_t = \{1, \ldots, k\}$ for all $t$}
2. For $i = 1$ to $n$ and $j = 1$ to $k$, calculate $\hat{q}_{i,j} = 1/N \sum_{t=1}^N p'_{i,\rho_t(j)}$. {estimating $Q$}
3. For $t = 1$ to $N$, find $\rho_t$ solving (10) with costs: $c'_{i,j} = \sum_{i=1}^n p'_{i,j} \log \left( \frac{p'_{i,j}}{\hat{q}_{i,j}} \right)$.
4. If an improvement in $\sum_{t=1}^N \hat{L}_t$ has been achieved, return to 2. Finish otherwise.

### 3.2 Equivalent Classes Representatives Relabeling

Equivalent classes representatives (ECR) method (Papastamoulis and Iliopoulos 2010) is a fast and easy strategy to undo the label switching. The idea is as follows. We can define two allocations $z^1$ and $z^2$ as equivalent if there is a permutation $\rho \in \mathcal{P}_k$ such that $z^1 = \rho(z^2) = \{\rho(z^2_1), \ldots, \rho(z^2_n)\}$. Hence, if we let $g = \{g_i\}_{i=1}^n$ be the vector of “true allocations” and $z'$ be the vector of allocations being sampled at iteration $t$ of the MCMC algorithm, to undo the label switching, we find the permutation $\rho_t \in \mathcal{P}_k$ that minimizes the simple matching distance (SMD) between $g$ and $\rho_t(z') = (\rho_t(z'_1), \ldots, \rho_t(z'_n))$, for $t = 1, \ldots, N$: 

$$\text{SMD}(g, \rho_t(z')) = n - \sum_{i=1}^n \mathbb{I}\{g_i = \rho_t(z'_i)\}.$$ 

Thus, in this case, the pivot will be the vector $g$. An easy solution that avoids a recursive algorithm to estimate $g$ is to use the MCMC output to choose a “good” allocation vector, and following Martin, Mengersen, and Robert (2005), Papastamoulis and Iliopoulos (2010) used the MAP estimator: at each iteration of the MCMC, we store the latent allocations, $z'$, and the log-posterior distribution, $\ell' = \log \{p(w', \theta', z'|y)\}$. To choose the MAP estimate, we find $t^*$ such that $\ell'^* = \max_{t = 1, \ldots, N} \ell'^* \ell'$ and let $g_{\text{MAP}} = z'^*$. A description of the ECR relabeling strategy is provided in Algorithm 2: note that $n'_j = \#\{i : z'_i = j\}$ and $n'_{j,l} = \#\{i : z'_i = j, g_{\text{MAP}} = l\}$.

**Algorithm 2** ECR relabeling strategy

1. Find $g_{\text{MAP}}$.
2. For $t = 1$ to $N$, find $\rho_t$ solving (10) with costs: $c'_{i,j} = n'_j - n'_{j,l}$.

From Algorithm 2, it is not clear how the costs work. We have devised a simple example to understand this point: let $z'$ and $g_{\text{MAP}}$ be as in Table 1. It is clear the permutation that minimizes the SMD is $\rho_t = \{3, 1, 2, 4\}$; hence, $\rho_t(1) = 3$, $\rho_t(2) = 1,$
Table 1. Example ECR: $z'$ and $g^{\text{MAP}}$

| $i$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|-----|---|---|---|---|---|---|---|---|---|----|----|----|
| $z'$| 2 | 2 | 2 | 3 | 3 | 3 | 3 | 1 | 1 | 1  | 1  | 4  |
| $g^{\text{MAP}}$| 1 | 1 | 1 | 1 | 2 | 2 | 2 | 2 | 3 | 3  | 3  | 4  |

$\rho_t(3) = 2$, and $\rho_t(4) = 4$. Then, the costs are matching distances translated as comparisons of counts.

According to Papastamoulis and Iliopoulos (2010), besides the MAP estimate, other valid choices for good allocation vectors are the most probable allocation and the allocation vector corresponding to the maximum of the likelihood (8).

We stress that in this summary of the ECR algorithm, we have restricted our attention to the costs. However, to show that the sampler converges to the correct posterior distribution, Papastamoulis and Iliopoulos (2010) gave more emphasis on arguments about the equivalent classes induced by $\rho(z)$ for $\rho \in \mathcal{P}_k$ and convergence proofs: the name representatives comes from taking a set $Z_0$ consisting of exactly one representative from each equivalent class and then showing that a sample from the posterior $p(w, \theta | y)$ can be obtained by calculating the posterior restricted to $Z_0$ and then permuting the labels of the simulated values.

3.3 Comments on KL and ECR Algorithms

The KL and ECR algorithms need estimated pivot variables $\hat{Q}$ and $\hat{g}$, respectively. We stress that these must be chosen as close as possible to the mode of one of the $k!$ symmetric modes of the posterior distribution: the aim is to eliminate the influence of the remaining symmetric copies. The KL algorithm uses a posterior mean and the ECR strategy aims for a posterior mode. This is an important point because it is valid for all relabeling algorithms. KL achieves its pivot starting from a bad estimate and improving it via fixed point theory ideas, thus leading to an iterative algorithm that converges to a local minimum, while ECR aims to find a good estimate from the start (usually the MAP estimate). Once the pivot variables are obtained, these are used to permute the MCMC output, that is, (9), and obtain a representative of one of the modes of the posterior distribution.

The real difference between the algorithms is the costs used in the loss function (10). It is possible to use the MAP estimator of $Q$: $Q^{\text{MAP}}$ on the KL algorithm. In the same manner, via algorithm 1 of Stephens (1997, p. 800), an estimate $\hat{g} = (\hat{g}_i)_{i=1}^n$ of $g$ can be obtained. A description of the ECR strategy via an iterative process is provided in Algorithm 3, and in this case, $n'_{j,l} = \#\{i : z_i' = j, \hat{g}_i = l\}$. Now we start with a bad estimate $\hat{g}$ and improve it via an iterative algorithm. The last iteration will be exactly the original ECR algorithm but where the pivot was obtained in an alternative manner.

Note however that neither in the original ECR nor in the iterative version, the pivot $\hat{g}$ will coincide with the final estimate for the single best clustering. An alternative that will ensure this is provided in Algorithm 4. This algorithm will be computationally demanding.
Algorithm 3 ECR iterative relabeling strategy 1
1: Initialize $\rho_1, \ldots, \rho_N$. \{Usually set to the identity: $\rho_t = \{1, \ldots, k\}$ for all $t$\}
2: For $i = 1$ to $n$, calculate $\hat{g}_i = \text{mode}(\rho_1(z^1_i), \ldots, \rho_N(z^N_i))$. \{estimating $g$\}
3: For $t = 1$ to $N$, find $\rho_t$ solving (10) with costs: $c^t_{i,j} = n^t_j - n^t_{j,i}$.
4: If an improvement in $\sum_{t=1}^N \hat{L}^t$ has been achieved, return to 2. Finish otherwise.

Algorithm 4 ECR iterative relabeling strategy 2
1: Initialize $\rho_1, \ldots, \rho_N$. \{Usually set to the identity: $\rho_t = \{1, \ldots, k\}$ for all $t$\}
2: For $i = 1$ to $n$ and $j = 1$ to $k$, calculate $\hat{q}_{i,j} = \frac{1}{N} \sum_{t=1}^N p^t_{i,\rho_t(j)}$.
3: For $i = 1$ to $n$, set $\hat{g}_i = j$ if $\hat{q}_{i,j} = \max_{l=1}^k \{\hat{q}_{i,l}\}$. \{estimating $g$\}
4: For $t = 1$ to $N$, find $\rho_t$ solving (10) with costs: $c^t_{i,j} = n^t_j - n^t_{j,i}$.
5: If an improvement in $\sum_{t=1}^N \hat{L}^t$ has been achieved, return to 2. Finish otherwise.

4. USING THE DATA TO UNDO THE LABEL SWITCHING

Our aim is to combine all the information about the clusters gathered by the latent allocations, (2), and the data to devise a simple loss function. Then, as with other relabeling algorithms, to undo the label switching, we will find the permutation of the indices that minimizes a loss function.

The key point is that if the MCMC has converged, from iteration to iteration, the labels of each cluster may change. But the clusters should be roughly the same. The difference should be small enough for us to discover where a cluster of the data has moved to. Hence, all that we need is to keep track of the $k$ clusters throughout each iteration of the sampler. For now, to explain our ideas, we will restrict our attention to the univariate case. The multivariate problem is a trivial generalization and is explained in the illustrations for multivariate data.

4.1 A SIMPLE LOSS FUNCTION

Let us assume that the center and the dispersion of the clusters within the data are known: $\{m_1, \ldots, m_k\}$ and $\{s_1, \ldots, s_k\}$. These will be our pivot variables. To undo the label switching, we can find the permutation, $\rho_t \in P_k$ for $t = 1, \ldots, N$, that minimizes

$$\sum_{i=1}^k \sum_{j=1}^k n^t_{\rho_t(j)} \left\{ \sum_{l: \rho_t(z^t_i) = j} \left( \frac{y_i - m_l}{s_l} \right)^2 \right\},$$

(11)

where $n^t_{\rho_t(j)} = \#\{i : \rho_t(z^t_i) = j\}$. With this loss, the costs in (10) will be given by

$$c^t_{i,j} = n_j \sum_{l: z^t_i = j} \left( \frac{y_i - m_l}{s_l} \right)^2.$$

If the clustering assumptions induced by (1) and (2) are met, once the MCMC has converged, at each iteration of the sampler, there must be $k$ clusters identified by the conditional
relationship between $y_i$ and $z^j_i$. Thus, a $k$-means type of diverging measure, (11), will be able to keep track of each cluster. To measure deviations from the center of each cluster, $m_j$, we eliminate the effect of the scale, and as the number of observations allocated within each cluster can be rather distinct, we make the divergences proportional to the size of each group. These considerations improved the loss function greatly.

4.1.1 Means and Standard Deviations. There are different alternatives for measures of central tendency and dispersion for the clusters, and here, we derive algorithms based on the mean and standard deviation. Later on, in the article, we describe how to use more robust alternatives.

To estimate the means and standard deviations for each cluster, we could use the mixture parameters or functions of them; for example, in the case of the normal mixture, the obvious options are posterior means for the means and standard deviations of each component, that is, $\hat{m}_j = \mathbb{E}(\mu_j | y)$ and $\hat{s}_j = \mathbb{E}(\sigma_j | y)$, for $j = 1, \ldots, k$. However, there are two problems with this approach. First, the algorithm would be restricted to mixtures with a particular component’s distribution. Second, we would need to estimate these posterior means from the MCMC output, and the parameters sampled via the MCMC are random and can exhibit great amounts of variability depending on whether the Markov chain traverses regions of high or low probability in the posterior distribution. To solve these problems, we will work with functions depending only on the data and the latent allocations. The data are fixed, and there is only certain amount of variability that subsets of it will admit. On the other hand, the distributional assumptions of the model are contained in the clusters induced by the latent allocations.

With these ideas, to estimate the means and standard deviations of each cluster, we use posterior means for the sample mean and sample standard deviation:

$$\hat{m}_j = \mathbb{E}(\bar{y}_j | y) \quad \text{and} \quad \hat{s}_j = \mathbb{E}(s_j | y), \quad \text{for} \quad j = 1, \ldots, k, \quad (13)$$

where $\bar{y}_j = \frac{1}{n_j} \sum_{i: z_i^j = j} y_i$ and $s_j^2 = \frac{1}{n_j - 1} \sum_{i: z_i^j = j} (y_i - \bar{y}_j)^2$.

Note that quantities such as (13) are valid posterior means. To see this, first let $Z = \{1, \ldots, k\}^n$, and then for $h : \mathbb{R}^n \times Z \times \{1, \ldots, k\} \to \mathbb{R}$, calculate

$$\mathbb{E}(h(y, z, j) | y) = \sum_{z \in Z} \int_{\theta} \int_{\Omega} h(y, z, j) p(z, w, \theta | y) dw d\theta,$$

$$= \sum_{z \in Z} h(y, z, j) p(z | y). \quad (14)$$

Thus, for (13), the function $h(y, z, j)$ will be given by $\bar{y}_j$ and $s_j$ for the sample mean and sample standard deviation, respectively.

The posterior means in (13) will be estimated using the MCMC output and calculating

$$\bar{y}_j^t = \frac{1}{n_j^t} \sum_{i: z_i^j = j} y_i \quad \text{and} \quad s_j^t = \left(\frac{1}{n_j^t - 1} \sum_{i: z_i^j = j} (y_i - \bar{y}_j^t)^2\right)^{1/2}.$$

Section 4.2, specifically Algorithm 5, will explain the technical details of $n_j^t = 0$ or 1.
4.2 The Algorithm

As with the previous algorithms, to define the costs (12), a complete knowledge about the pivot variables is assumed—in our case, the means and standard deviations of each group. We have two ways to proceed: via a recursive algorithm or using estimates. We use the second approach simply because it is faster, but for a comprehensive explanation we include both.

The idea to find estimates is as follows: first, we obtain initial estimates of our pivot variables in one of the modes of the posterior distribution, in this case (13). If the MCMC has converged, we choose the first set of allocations after the burn-in period and calculate our starting sample means and sample standard deviations with it. At this point, the MCMC should be traversing one of the \( k! \) modes of the posterior distribution, and for our propose, it is not important which particular mode this is. To move these initial pivots closer to the mode of this particular mode of the posterior distribution, we average the same estimates for each group over all the iterations of the MCMC algorithm, preserving identifiability via (11). The final estimates will be close to the mode of one of the modes of the posterior distribution, and we can use these as pivots to find the permutations (9). A formal description of the strategy to find estimates is given in Algorithm 5 (note that \( R = \gamma(n) - \gamma(1) \)).

Algorithm 5 Data-based relabeling: finding estimates

1: For \( j = 1 \) to \( k \), initialize \( n^m_j = 1, n^s_j = 1, \hat{m}_j = y_{(1)} + R \left( \frac{j}{k+1} \right) \), and \( \hat{s}_j = R/k \).

2: For \( t = 1 \) to \( N \), find \( \rho_t \) solving (10) with costs: \( c^t_{l,j} = n^t_j \sum_{\{i:z^t_i = j\}} \left( \frac{y_i - \hat{m}_l}{\hat{s}_l} \right)^2 \).

For \( j = 1 \) to \( k \), update \( \hat{m}_j, \hat{s}_j, n^m_j, \) and \( n^s_j \)

If \( n^t_{\rho_t(j)} > 0 \) \( \Rightarrow \hat{m}_j = (n^m_j - 1)\hat{m}_j + \gamma^t_{\rho_t(j)})/n^m_j \) and \( n^m_j = n^m_j + 1 \).

If \( n^t_{\rho_t(j)} > 1 \) \( \Rightarrow \hat{s}_j = (n^s_j - 1)\hat{s}_j + s^t_{\rho_t(j)})/n^s_j \) and \( n^s_j = n^s_j + 1 \).

A description of the Data relabeling via estimates is provided in Algorithm 6. We have defined a two-step procedure: in Algorithm 5, the aim is to find estimates for the mean and standard deviation for each cluster. While in Algorithm 6, the interest lies in the permutations \( \rho_t \), for \( t = 1, \ldots, N \). It is important to say that several experiments were performed using Algorithm 5, to obtain the permutations directly, and in some cases, the results were as good as with the two-step procedure. However, in general, the two-step procedure was more accurate.

Algorithm 6 Data-based relabeling: estimates strategy

1: Find estimates \( \hat{m}_j \) and \( \hat{s}_j \) for \( j = 1 \) to \( k \).

2: For \( t = 1 \) to \( N \), find \( \rho_t \) solving (10) with costs: \( c^t_{l,j} = n^t_j \sum_{\{i:z^t_i = j\}} \left( \frac{y_i - \hat{m}_j}{\hat{s}_j} \right)^2 \).
The corresponding recursive algorithm follows directly from algorithm 1 of Stephens (1997, p. 800). A description of this strategy is given in Algorithm 7.

Algorithm 7 Data-based relabeling: iterative strategy

1: Initialize $\rho_1, \ldots, \rho_N$. {Usually set to the identity: $\rho_t = \{1, \ldots, k\}$ for all $t$}

2: For $j = 1$ to $k$, calculate

$$\hat{m}_j = \frac{1}{n^m} \sum_{t=1}^{N} y^t_{\rho_t(j)} 1\{n^t_{\rho_t(j)} > 0\}$$
and

$$\hat{s}_j = \frac{1}{n^s} \sum_{t=1}^{N} s^t_{\rho_t(j)} 1\{n^t_{\rho_t(j)} > 1\},$$

where $n^m = \sum_{t=1}^{N} 1\{n^t_{\rho_t(j)} > 0\}$ and $n^s = \sum_{t=1}^{N} 1\{n^t_{\rho_t(j)} > 1\}$.

3: For $t = 1$ to $N$, find $\rho_t$ solving (10) with costs:

$$c^t_{l,j} = n^t_j \sum_{i: z^t_i = j} \left( \frac{y_i - \hat{m}_j}{\hat{s}_j} \right)^2.$$ 

4: If an improvement in $\sum_{t=1}^{N} \hat{L}^t$ has been achieved, return to 2. Finish otherwise.

4.3 ADDITIONAL COMMENTS

In this section, we describe the ideas under the definition of the costs (12) and possible alternatives.

4.3.1 Main Ideas. There are two important ideas under the definition of (12). First, it is easier to obtain estimates (pivots) close to the mode of one of the modes of the posterior distribution using general characteristics of the components such as location and dispersion, rather than via those related to individual observations such as classification probabilities or allocations. If the algorithm has converged, we can obtain estimates for the location and dispersion for the groups directly from the MCMC output: the relationship between the observations and the allocations is the key. Second, the observations are incorporated directly within the loss, and not only via the MCMC output. Indeed, note that all the quantities needed for the calculation of the costs, (12), are based on the data, while the groups are indicated by the allocations. This should reduce the variability of the costs and thus lead to a more stable and robust relabeling algorithm.

4.3.2 Extensions. Our costs (12) can be computed efficiently and in the examples that we have studied, they have good performance. However, it is straightforward to derive different alternatives; we can start by using robust measures of location and dispersion for the groups, for example, the median and median absolute deviation. Thus, in (13), we substitute $(\bar{y}_j, s_j)$ by $(\text{med}_j, \text{mad}_j)$. With this change, the costs (12) will be resistant to outliers. This will be useful when the sampler visits the tails of the posterior distribution. To find $(\hat{m}_j, \hat{s}_j)$, for $j = 1, \ldots, k$, in Algorithm 5, we would use

$$\text{med}'_j = \text{med} \left( y_{i: z^t_i = j} \right)$$
and

$$\text{mad}'_j = \text{med} \left( |y_i - \text{med}'_j| \right).$$

The calculation of these statistics will lead to a slower algorithm but providing accurate component-specific inference. Another alternative, as suggested by a referee, is to use in
(10) the costs
\[ c_{l,j} = - \sum_{i: z_i = j} \log \{ f(y_i | \theta_l) \}, \]
thus devising label switching algorithms for particular component’s distributions. The idea again will be to find posterior estimates for \( \theta_l \) via (14), using the data and the allocations. Under this setting, for a mixture of normals, the costs will be given by
\[ c_{l,j} = \frac{n_j}{2} \log (2\pi \sigma_l^2) + \frac{1}{2} \sum_{i: z_i = j} \left( \frac{y_i - \mu_l}{\sigma_l} \right)^2. \] (15)
To estimate \( \mu_l \) and \( \sigma_l \), we can use again (13). Importantly, connecting all the elements of the costs directly with the data.

5. COMPARISONS

In this section, using simulated and real datasets, we compare the performance of the relabeling algorithms: Data, ECR, and KL. First, all comparisons are carried out under a univariate setting. Then we move to the multivariate case.

5.1 UNIVARIATE MIXTURES

For our first comparison, we take samples from three models involving mixtures:

Model 1 : 0.4N(y|0.63, 0.00032) + 0.6N(y|0.65, 0.00016).
Model 2 : 0.25N(y|−3, 1) + 0.25N(y|−1, 1) + 0.25N(y|1, 1) + 0.25N(y|3, 1).
Model 3 : 0.20N(y|19, 5) + 0.20N(y|19, 1) + 0.25N(y|23, 1) + 0.20N(y|29, 0.5) + 0.15N(y|33, 2).

We generated samples of size: \( n = 1000 \), \( n = 200 \), and \( n = 600 \), from Models 1, 2, and 3, respectively. To improve comparability, the samples were not randomly drawn. We evaluated the quantile function of each mixture on a grid in \((0, 1)\), and a friendly implementation of this can be found in the package “nor1mix” of R; see Mächler (2011). The idea is that under noninformative priors, inference is as good as the sample and since the objective is to compare our proposal with other alternatives, using as reference the true values is the best way to proceed. This type of data has been used before by Nobile and Fearnside (2007) to illustrate the performance of their transdimensional MCMC for mixture models.

Model 1 was designed to exhibit the same characteristics as the crab data, which was first analyzed by Pearson (1894), and comprises the measurements of the ratio of forehead to body length of 1000 crabs. Model 2 is a mixture of symmetric and poorly separated components, and Model 3 has been used by Papastamoulis and Iliopoulos (2010) to demonstrate the performance of the ECR algorithm. All mixtures have overlapping components and present challenging scenarios for any relabeling algorithm.

For the real data, we have chosen enzyme data. The enzyme dataset concerns the distribution of enzymatic activity in the blood of an enzyme involved in the metabolism
of a carcinogenic substance, among a group of 245 unrelated individuals. This dataset was analyzed by Richardson and Green (1997).

To compare the effect on inference related to the choice of MCMC strategy, we will compare posterior mean estimates, obtained via the standard Gibbs sampler, with those obtained via its reversible jump extension.

5.1.1 Relabeling Algorithms, Gibbs Sampler. For Models 1–3, we work with the Gibbs sampler for normal mixtures described by Richardson and Green (1997), along with their data-based priors. In each case, we generated an MCMC with 60,000 iterations, throwing away the first 30,000 as a burn-in period. Then, with the output of the same MCMC, we undo the label switching using each algorithm: Data, ECR, and KL. Thus, for each case, posterior mean estimates for the weights, means, and variances are calculated. To measure the accuracy of the point estimates, we determined the relative error:

$$\text{Rel. error} = \frac{1}{k} \sum_{j=1}^{k} \left| \frac{\mathbb{E}(h(w_j, \theta_j) | y) - h(w_j, \theta_j)}{h(w_j, \theta_j)} \right|,$$

where the posterior means for the functions $h_1(w_j, \theta_j) = w_j$ and $h_2(w_j, \theta_j) = \theta_j$ will be estimated via (7). To have a fair comparison, this was repeated 100 times.

In Table 2, we present the first comparison with the dataset from Model 1. In this case, from the column of the average relative errors, it is clear that Data and KL algorithms are more accurate than the ECR strategy. There is no difference in precision between Data and KL. In Figure 1(a)–1(c), we display the traces for the means (just 10,000 iterations) generated after the algorithms data, ECR, and KL, respectively, were used to undo the label switching. From these graphics, it is clear that while the Data and KL strategies introduce the constraint $\mu_1 < \mu_2$, over all 10,000 iterations, the ECR relabeling introduced the same constraint on iterations where regions of high probability of the posterior distribution were explored. Hence, as Papastamoulis and Iliopoulos (2010) proposed more alternatives to obtain good vectors of allocations, we changed $g_{\text{MAP}}$ for the true single best

|     | $w_j$ | 0.4  | 0.6  | Average Rel. error | Std. dev Rel. error |
|-----|-------|------|------|-------------------|-------------------|
| Data | $\mathbb{E}(w_j | y)$ | 0.349 | 0.651 | 0.214             | 0.064             |
| ECR  | $\mathbb{E}(w_j | y)$ | 0.332 | 0.668 | 0.283             | 0.050             |
| KL   | $\mathbb{E}(w_j | y)$ | 0.349 | 0.651 | 0.214             | 0.064             |
| True | $\mu_j$ | 0.63  | 0.65  |                   |                   |
| Data | $\mathbb{E}(\mu_j | y)$ | 0.627 | 0.649 | 0.006             | 0.001             |
| ECR  | $\mathbb{E}(\mu_j | y)$ | 0.628 | 0.648 | 0.006             | 0.001             |
| KL   | $\mathbb{E}(\mu_j | y)$ | 0.627 | 0.649 | 0.006             | 0.001             |
| True | $\sigma^2_j$ | 0.00032 | 0.00016 |             |                   |
| Data | $\mathbb{E}(\sigma^2_j | y)$ | 0.000277 | 0.000165 | 0.166             | 0.033             |
| ECR  | $\mathbb{E}(\sigma^2_j | y)$ | 0.000262 | 0.000180 | 0.308             | 0.035             |
| KL   | $\mathbb{E}(\sigma^2_j | y)$ | 0.000277 | 0.000165 | 0.166             | 0.033             |
clustering (there is no better alternative) on the ECR algorithm, and the results were then the same as with the Data and KL strategies. To test convergence problems, we performed individual runs of 500,000 iterations (taking 400,000 as a burn-in period), but the results calculated via the ECR algorithm and $g^{\text{MAP}}$ were similar to those presented in Table 2 and Figure 1.

For our second comparison, we use the dataset from Model 2, and the results are shown in Table 3. In this case, we have a symmetric mixture of four components and where the only difference between components is the mean. From average relative errors for the weights and means, in this case, we observe that Data relabeling is far more accurate than either ECR or KL strategies. Estimates of variances are equally bad across the three algorithms.

Table 3. Average of posterior mean estimates and relative errors across 100 experiments for the dataset from Model 2

| $j$ | 1      | 2      | 3      | 4      | Average   | Std. dev |
|-----|--------|--------|--------|--------|-----------|----------|
|     | $w_j$  | 0.25   | 0.25   | 0.25   | 0.25      |          |
| True|        |        |        |        |           |          |
| Data| $E(w_j|y)$ | 0.224  | 0.277  | 0.273  | 0.226    | 0.442    | 0.089    |
| ECR | $E(w_j|y)$ | 0.381  | 0.122  | 0.122  | 0.375    | 1.993    | 0.048    |
| KL  | $E(w_j|y)$ | 0.328  | 0.168  | 0.176  | 0.328    | 1.587    | 0.281    |
|     | $\mu_j$| $-3$   | $-1$   | 1      | 3        |          |          |
| True|        |        |        |        |          |          |
| Data| $E(\mu_j|y)$ | $-2.191$ | $-0.886$ | 0.879  | 2.202    | 0.776    | 0.065    |
| ECR | $E(\mu_j|y)$ | $-1.898$ | $-0.480$ | 0.453  | 1.930    | 1.801    | 0.507    |
| KL  | $E(\mu_j|y)$ | $-2.165$ | $-0.406$ | 0.405  | 2.171    | 2.064    | 0.455    |
|     | $\sigma^2_j$| 1      | 1      | 1      | 1        |          |          |
| True|        |        |        |        |          |          |
| Data| $E(\sigma^2_j|y)$ | 1.916  | 2.734  | 2.729  | 1.910    | 5.289    | 0.085    |
| ECR | $E(\sigma^2_j|y)$ | 2.270  | 2.388  | 2.396  | 2.235    | 5.289    | 0.085    |
| KL  | $E(\sigma^2_j|y)$ | 2.039  | 2.612  | 2.606  | 2.031    | 5.289    | 0.085    |
In the scaled components, predictive and single best clustering, presented in Figure 2, we observe again that Data relabeling performs much better than the alternatives. True scaled densities were plotted via \( w_j N(y | \mu_j, \sigma_j) \) and the true single best clustering was determined using Equation (3). Here we can observe that the single best clustering achieved with the data relabeling, Figure 2(d), is almost identical to the one calculated with the true values [Figure 2(a)]. For curiosity, we used the true single best clustering instead of \( \mathbf{g}^{\text{MAP}} \) on the ECR algorithm and the results shown in Table 3 with the data relabeling remain more accurate. To test convergence problems, we ran the Gibbs sampler for 1,000,000 iterations discarding the first 900,000, and the results were almost identical to those described above. Our insistence in maintaining a maximum of 100,000 iterations for the analysis lies in the storage requirements for the classification probabilities and the time needed by the KL algorithm to converge (in this case approx. 16 min). See the Appendix (given in the online supplementary materials) for more about the storage requirements.

In Model 3, we have three separated components: components 3, 4, and 5. Two components have the same mean and weight: components 1 and 2. But component 1 has a larger variance than component 2. Here, posterior mean estimates for the separated components are acceptable, but for components 1 and 2 we observe difficulties, see Table 4. In this case,

Table 4. Average of posterior mean estimates and relative errors across 100 experiments for the dataset of Model 3

|       | \( j \)     | 1     | 2     | 3     | 4     | 5     | Average Rel. error | Std. dev Rel. error |
|-------|-------------|-------|-------|-------|-------|-------|-------------------|-------------------|
| True  | \( w_j \)   | 0.2   | 0.2   | 0.25  | 0.2   | 0.15  | 0.149             | 0.818             |
| Data  | \( \mathbb{E}(w_j | y) \) | 0.218 | 0.177 | 0.256 | 0.2   | 0.149 | 0.818             | 0.230             |
| ECR   | \( \mathbb{E}(w_j | y) \) | 0.097 | 0.298 | 0.256 | 0.2   | 0.149 | 1.044             | 0.051             |
| KL    | \( \mathbb{E}(w_j | y) \) | 0.101 | 0.294 | 0.256 | 0.2   | 0.149 | 1.061             | 0.050             |
| True  | \( \mu_j \) | 19    | 19    | 23    | 29    | 33    | 29.011            | 33.003            |
| Data  | \( \mathbb{E}(\mu_j | y) \) | 18.833| 18.824| 22.988| 29.011| 33.003| 0.047             | 0.013             |
| ECR   | \( \mathbb{E}(\mu_j | y) \) | 18.590| 19.067| 22.988| 29.011| 33.004| 0.029             | 0.014             |
| KL    | \( \mathbb{E}(\mu_j | y) \) | 18.569| 19.089| 22.988| 29.011| 33.003| 0.031             | 0.014             |
| True  | \( \sigma^2_j \) | 5     | 1     | 1     | 0.5   | 2     | 1.546             | 1.754             |
| Data  | \( \mathbb{E}(\sigma^2_j | y) \) | 2.654 | 2.145 | 1.041 | 0.546 | 1.988 | 1.754             | 0.253             |
| ECR   | \( \mathbb{E}(\sigma^2_j | y) \) | 2.695 | 2.105 | 1.042 | 0.546 | 1.987 | 1.706             | 0.093             |
| KL    | \( \mathbb{E}(\sigma^2_j | y) \) | 2.609 | 2.191 | 1.041 | 0.546 | 1.988 | 1.808             | 0.106             |

Figure 2. True, estimated scaled densities and single best clustering for Model 2.
Table 5. Average system time to undo the label switching across 100 experiments (for the enzyme data, only one experiment was performed)

| Dataset | $k$ | $n$   | $N$  | Data find est. | Data relabeling | ECR   | KL     |
|---------|-----|-------|------|----------------|-----------------|-------|--------|
| Model 1 | 2   | 1000  | 30,000 | 4.855 sec      | 4.919 sec       | 4.853 sec | 1.12 min |
| Model 2 | 4   | 200   | 30,000 | 1.053 sec      | 1.088 sec       | 1.111 sec | 2.66 min |
| Model 3 | 5   | 600   | 30,000 | 3.012 sec      | 3.020 sec       | 2.992 sec | 4.45 min |
| Enzyme  | 4   | 245   | 100,000 | 5.329 sec     | 5.471 sec       | 5.44 sec | 14.77 min |

Figure 3. Estimated scaled densities and single best clustering for the enzyme data.

posterior mean estimates for the weights calculated via our relabeling algorithm are more accurate. For the other variables, the ECR algorithm performs marginally better.

For the enzyme data, we ran the Gibbs sampler for normal mixtures described by Richardson and Green (1997), along with their data-based priors. Here, assuming $k = 4$, we generated 200,000 iterations and used the first 100,000 as a burn-in period. Then, with the output of the same MCMC, we undo the label switching using each algorithm: Data, ECR, and KL. Estimated scaled densities and single best clustering are displayed in Figure 3. Here, there are differences in the single best clustering produced with the three algorithms; see groups 2 and 4 (in red and purple, respectively). Also, there are differences in the estimates for the scaled densities.

In Table 5, we present the recorded average time required to undo the label switching for each algorithm. For the Data relabeling, we are differentiating between the registered time to find estimates, Algorithm 5, and the actual relabeling time, Algorithm 6. For the ECR strategy, the time required for the calculation of the log-posterior distribution and the time to find and extract the MAP estimate was not recorded. Given the order of the system times displayed in Table 5 (in seconds), we believe that if these calculations are considered, Data and ECR relabeling should be equally fast. We remark that this is for the univariate case.

5.1.2 Relabeling Algorithms, Reversible Jump. Here, the results obtained in the last section for Models 1–3 are compared with estimates obtained via the reversible jump sampler, for normal mixtures, described by Richardson and Green (1997) and their
data-based priors. First, to obtain convergence, we generated 500,000 iterations and kept
the last 100,000 for Model 1, and 200,000 for Models 2 and 3. Then the output relevant to a
given \( k \) was extracted: \( k = 2, 4, \) and \( 5 \) for the data-sets of Models 1, 2, and 3, respectively.
Finally, after relabeling the output, posterior mean estimates were calculated. In this case,
because reversible jump is computationally more expensive than the Gibbs sampler, we
did not carry out 100 experiments.

A remarkable difference between the two samplers is how they explore the support of
the posterior distribution. The reversible jump sampler uniformly visits regions of high and
low probability in the posterior distribution, while the Gibbs sampler explores mainly high
probability regions and it does so in a nonuniform manner. This can clearly be observed in
Figure 1 where traces for the means for the dataset from Model 1 are displayed, for only
10,000 iterations. Figure 1(d)–1(f) are from the reversible jump’s output and undoing the
label switching using the Data, ECR, and KL algorithms, respectively. Figure 1(a)–1(c) are
the corresponding traces generated with the output of the Gibbs sampler.

For Model 1, with reversible jump, we did not observe major improvements in accuracy
and all the estimates were very close to those shown in Table 2.

For the dataset from Model 2, we observed an improvement in the accuracy for the
estimation of the weights. For example, with the reversible jump and Data relabeling, we
achieved a relative error of 0.269, while the average relative error via the Gibbs sampler was
0.442, with a standard deviation of just 0.089. Similar improvements were obtained with
the ECR and KL strategies. Curiously, estimates for the means and variances went in the
opposite direction. In the case of the Data relabeling, with the Gibbs sampler, we obtained
an average relative error of 0.776 and a standard deviation of 0.065, and via reversible
jump, a relative error of 0.923 was recorded.

For Model 3, we observed again an improvement in the estimation of the weights. This
time it happened just with the Data relabeling: from an average relative error of 0.818 (and
a standard deviation of 0.230), it went to 0.455. All the other estimates were of the same
order as those displayed in Table 4.

To conclude this section, it is important to say that the computational cost of the reversible
jump sampler is far greater than that of the Gibbs sampler. For example, to obtain the results
for Model 1, it took almost 9 min to run the MCMC and a further 4 min to extract and
calculate the classification probabilities for \( k = 2 \). For the standard Gibbs sampler, less that
2 min were needed overall. Hence, given the evidence observed in our experiments, we see
no reason to use the transdimensional MCMC rather than the standard Gibbs sampler.

### 5.2 Multivariate Normal Mixtures

Here, we extend the loss function (11) to allow for multivariate data. In this case,
\( y = \{ (y_{i,r})_{r=1}^{p} \}_{i=1}^{n} \), and hence, each group has \( p \) means and standard deviations:
\( m_j = \{ m_{j,r} \}_{r=1}^{p} \) and \( s_j = \{ s_{j,r} \}_{r=1}^{p} \). With this consideration, the loss function (11) becomes

\[
\sum_{l=1}^{k} \sum_{j=1}^{k} n_{l_{(j)}} \sum_{y_{i,j} = j} \left\{ \sum_{r=1}^{p} \left( \frac{y_{i,r} - m_{l,r}}{s_{l,r}} \right)^2 \right\},
\]  

(16)
Table 6. Parameters for the simulated data

| j | $w_j$ | $\mu_j$ | $\Sigma_j$ |
|---|---|---|---|
| 1 | 0.25 | (4.5, −2.5) | (0.5, −0.25, 0.5) |
| 2 | 0.25 | (−3.0, 4.0) | (0.5, −0.25, 0.5) |
| 3 | 0.25 | (6.5, 7.0) | (4, 2.5, 4) |
| 4 | 0.25 | (7.0, −3.0) | (4, 2.5, 9) |

and hence, the costs become

$$c_{i,j}^l = n_j^f \sum_{\{i: z_i^l=j\}} \left\{ \sum_{r=1}^p \left( \frac{y_{i,r} - m_{l,r}}{\sigma_{l,r}} \right)^2 \right\}. $$

To illustrate the performance of our methodology in a multivariate setting, the Gibbs sampler for multivariate normals described by Stephens (1997), with default priors, was implemented. We simulated $n = 200$ observations from the distribution $\sum_{j=1}^4 w_j N_2(\mu_j, \Sigma_j)$ with actual values shown in Table 6 (note that $\Sigma = (\Sigma_{11}, \Sigma_{12} = \Sigma_{21}, \Sigma_{22})$). This mixture model has been considered by Papastamoulis and Iliopoulos (2010). In our experiments, we took $k = 4$. For the analysis, we kept the last 30,000 iterations from a chain of 100,000 iterations.

In Figure 4(a), we display the level curves calculated via the true model, along with the true single best clustering. In Figure 4(b)–4(d), we display level curves of the plug-in density estimates and single best clustering generated via ECR, Data, and KL algorithms, respectively. In this case, the results obtained with each algorithm are identical. We repeated the experiment several times and there was no difference between the estimates generated via the three algorithms.

The time required to find estimates for the data algorithm was 1.352 sec, and then it took a further 1.525 sec to undo the label switching. With the ECR and KL strategies, it took 1.457 and 30.319 sec, respectively, to undo the label switching.

![Figure 4. True, plug-in densities and single best clustering.](image-url)
6. DISCUSSION

We have described the key ideas for a deterministic relabeling algorithm, and from it we have derived an easy and efficient solution to the label switching problem. Our proposed solution, the Data relabeling, lies in the meaning of the relationship between the allocation variables (2) and the observations. Label switching can be considered under a general missing data model framework that includes as special cases finite mixtures, hidden Markov models, and Markov random fields; see Papastamoulis and Iliopoulos (2011). In this article, we restrict our attention to finite mixtures. An application is the mixture of Dirichlet process (MDP) model; see Lo (1984). In this setting, to deal with an infinite measure, there are two alternative ideas: marginal and conditional methods (see Griffin and Holmes 2010, sec. 6.3). If the Dirichlet process is integrated out, the so-called marginal algorithm, the labels of the clusters become unidentifiable; see Papaspiliopoulos and Roberts (2008). For the conditional algorithms, the weights of the Dirichlet process are only weakly identifiable, so there is label switching.

SUPPLEMENTARY MATERIALS

The supplementary materials include an Appendix with the computational issues and storage requirements to implement each algorithm. The actual code used for this article is also included: Data, ECR, and KL relabeling strategies and the standard Gibbs sampler for a fixed $k$ (LabelSwitching.zip).

ACKNOWLEDGMENTS

The authors thank two referees for their comments on an earlier version of the article. The first author is studying at the University of Kent with a grant support from CONACYT, the Mexican National Council for Science and Technology.

[Received April 2012. Revised September 2012.]

REFERENCES

Burkard, R. E., Dell’Amico, M., and Martello, S. (2009), Assignment Problems, Philadelphia, PA: Society for Industrial and Applied Mathematics (SIAM). [30]

Celeux, G. (1998), “Bayesian Inference for Mixtures: The Label Switching Problem,” in COMPSTAT 98, eds. R. Payne and P. J. Green, Berlin: Physica-Verlag, pp. 227–232. [26]

Celeux, G., Hurn, M., and Robert, C. P. (2000), “Computational and Inferential Difficulties With Mixture Posterior Distributions,” Journal of the American Statistical Association, 95, 957–970. [26,29]

Diebolt, J., and Robert, C. P. (1994), “Estimation of Finite Mixtures Distributions Through Bayesian Sampling,” Journal of the Royal Statistical Society, Series B, 56, 363–375. [26]

Ferguson, T. S. (1983), “Bayesian Density Estimation by Mixtures of Normal Distributions,” in Recent Advances in Statistics: Papers in Honor of Herman Chernoff on His Sixtieth Birthday, eds. H. Chernoff, J. S. Rustagi, M. H. Rizvi, and D. Siegmund, New York: Academic Press, pp. 287–302. [26]
Frühwirth-Schnatter, S. (2001), “Markov Chain Monte Carlo Estimation of Classical and Dynamic Switching and Mixture Models,” *Journal of the American Statistical Association*, 96, 194–209. [26,29,30]

Geweke, J. (2007), “Interpretation and Inference in Mixture Models: Simple MCMC Works,” *Computational Statistics & Data Analysis*, 51, 3529–3550. [29]

Griffin, J., and Holmes, C. C. (2010), “Computational Issues Arising in Bayesian Nonparametric Hierarchical Models,” in *Bayesian Nonparametrics*, eds. L. N. Hjort, C. C. Holmes, P. Müller, and S. G. Walker, Cambridge: Cambridge University Press, pp. 208–222. [44]

Grün, B., and Leisch, F. (2009), “Dealing With Label Switching in Mixture Models Under Genuine Multimodality,” *Journal of Multivariate Analysis*, 100, 851–861. [26]

Jasra, A. (2005), “Bayesian Inference for Mixture Models via Monte Carlo,” Ph.D. thesis, Imperial College London, University of London. [27]

Jasra, A., Holmes, C. C., and Stephens, D. A. (2005), “Markov Chain Monte Carlo Methods and the Label Switching Problem in Bayesian Mixture Modeling,” *Statistical Science*, 20, 50–67. [26,29]

Lo, A. Y. (1984), “On a Class of Bayesian Nonparametric Estimates: I. Density Estimates,” *The Annals of Statistics*, 12, 351–357. [44]

Mächler, M. (2011), “nor1mix: Normal (1-d) Mixture Models (S3 Classes and Methods),” R package version 1.1-3. Available at http://CRAN.R-project.org/package=nor1mix. [37]

Martin, J. M., Mengersen, K., and Robert, C. P. (2005), “Bayesian Modelling and Inference on Mixture of Distributions,” in *Handbook of Statistics* (Vol. 25), eds. D. Dey and C. R. Rao, Holland: Elsevier, pp. 459–507. [26,31]

Nobile, A., and Fearnside, A. T. (2007), “Bayesian Finite Mixtures With an Unknown Number of Components: The Allocation Sampler,” *Statistics and Computing*, 17, 147–162. [26,37]

Papaspiliopoulos, O., and Roberts, G. O. (2008), “Retrospective Markov Chain Monte Carlo Methods for Dirichlet Process Hierarchical Models,” *Biometrika*, 95, 169–186. [29,44]

Papastamoulis, P., and Iliopoulos, G. (2010), “An Artificial Allocations Based Solution to the Label Switching Problem in Bayesian Analysis of Mixtures of Distributions,” *Journal of Computational and Graphical Statistics*, 19, 313–331. [26,29,30,31,32,37,38,43]

——— (2011), “On the Convergence Rate of Random Permutation Sampler and ECR Algorithm in Missing Data Models,” *Methodology and Computing in Applied Probability*, 1–12. [44]

Pearson, K. (1894), “Contributions to the Mathematical Theory of Evolution,” *Philosophical Transactions of the Royal Society*, Series A, 185, 71–110. [37]

Richardson, S., and Green, P. J. (1997), “On Bayesian Analysis of Mixtures With an Unknown Number of Components” (with discussion), *Journal of the Royal Statistical Society*, Series B, 59, 731–792. [26,29,38,41]

Sperrin, M., Jaki, T., and Wit, E. (2010), “Probabilistic Relabelling Strategies for the Label Switching Problem in Bayesian Mixture Models,” *Statistics and Computing*, 20, 357–366. [27]

Stephens, M. (1997), “Bayesian Methods for Mixtures of Normal Distributions,” Ph.D. thesis, University of Oxford. Available at http://stephenslab.uchicago.edu/MSpapers/DPhilMS.ps.gz. [29,30,32,36,43]

——— (2000), “Dealing With Label Switching in Mixture Models,” *Journal of the Royal Statistical Society*, Series B, 62, 795–809. [26,30]

Yao, W. (2012), “Model Based Labeling for Mixture Models,” *Statistics and Computing*, 22, 337–347. [27]