An optimal data ordering scheme for Dirichlet process mixture models

Xue Wang[a], Stephen G. Walker[b,*]

[a] *School of Mathematics, Statistics and Actuarial Science, University of Kent, UK.*
[b] *Department of Mathematics, University of Texas at Austin, Texas, USA.*

Abstract

In recent years there has been increasing interest in Bayesian nonparametric methods due to their flexibility, and the availability of Markov chain Monte Carlo (MCMC) methods for sampling from the posterior distribution. As MCMC methods are generally time consuming for computation, there is a need for faster methods, which can be executed within a matter of seconds. A fast alternative to MCMC for sampling the well known and widely used Dirichlet process mixture (DPM) model is investigated to draw approximate independent and identically distributed samples from the posterior distribution of the latent allocations, and then to draw samples from the weights and locations conditional on the allocations. To address the order depend issue of the proposed algorithm, an optimal ordering scheme based on a sequence of optimizations is proposed to first obtain an optimal order of the data, and then run the algorithm on this ordering. The fast sampling algorithm is assisted by parallel computing using commands within MATLAB.

Keywords: `Allocation variables; predictive density; optimal ordering.`

1. Introduction

This paper is concerned with fast prediction from mixture models. The starting point is the well known Dirichlet process mixture (DPM) model ([1]), which places a Dirichlet process (DP) prior ([2]) on a mixing distribution. The density model, in the independent and identically distributed case, takes the form

$$f(y) = \int K(y;\theta)\,\mathrm{d}P(\theta),$$

where $K(y;\theta)$ is a density function for each θ, and $P(\cdot)$ is a random distribution function, specifically a DP with the representation ([3]) given by

$$P(\cdot) = \sum_{j=1}^{\infty} w_j\,\delta_{\theta_j}(\cdot),$$

where $\delta_\theta(\cdot)$ denotes the point mass at θ. Here the (w_j) are random weights obtained via $w_1 = v_1$ and $w_j = v_j \prod_{l<j}(1-v_l)$, for $j > 1$ and the (v_j) are independent and identically distributed as beta$(1, c)$ for some $c > 0$. The (θ_j) are independent and identically distributed from some density function g. If

*Corresponding author
Email address: `s.g.walker@math.utexas.edu` (Stephen G. Walker)

G is the distribution function corresponding to g then (c, G) are the parameters of the prior and, in particular, $\mathbb{E}\, P = G$.

There is now a rich literature on how to sample the posterior model in order to construct a predictive density estimate. The complexity of the model determines that the sampling needs to be done using MCMC methods, and the first such algorithms were presented in the PhD Thesis of Escobar ([4]). Since then numerous types of MCMC algorithms have been proposed and for a recent review we refer the reader to [5].

Recently there has been proposed a number of fast algorithms which avoid the use of MCMC. The most notable of which is the iterative, or sequential, update procedure described in [6] and [7]. This is an approximation to the posterior and requires some non-trivial numerical integration. More recently a fast algorithm has been put forward by [8] which works with an approximated marginal likelihood for the allocation variables of the mixture model. Maximizing this marginal likelihood then yields a set of allocation variables from which it is easy to obtain (approximate but correct given the approximated allocation variables) predictive samples. As with all sequential techniques, including Newton and Zhang's, this method relies on the chosen ordering of the data. To reduce the impact of the ordering problem various proposals have been made which include trying out a few random permutations and then, in the case of [8], to select the permutation which provides the best fit to the data according to some decision criterion.

Our proposal is similar to the [8] idea but with a notable extension. Our main goal and result is that we find an optimal permutation of the data. This is based on a sequence of optimizations; conditionally maximizing the sequence of predictives given the allocation variables, iterated with conditionally maximizing the probability of the allocation variables given the current order of the data. The idea is that we find the permutation or ordering of the data which maximizes the marginal likelihood.

Describing the layout of the article; in section 2 we describe and derive the conditional posterior distribution of the latent allocation variables, which tells us the probability of which individuals belong to which groups. Section 3 details how we obtain approximate samples from the distribution of allocation variables and then how obtain an optimal ordering for the data using a sequence of conditional maximization routines. Section 4 discusses the extension to the regression model. Section 5 then puts all these pieces together to present some illustrations including both real and simulated data sets, and, finally, section 6 contains a brief discussion.

2. The marginal model for allocations

A standard starting point for sampling the DPM model is given by the joint density function

$$p(y, d|\theta, w) = \prod_{i=1}^{n} w_{d_i}\, K(y_i; \theta_{d_i}).$$

Here the data is presented by $y = (y_1, \ldots, y_n)$ and the $d = (d_1, \ldots, d_n)$ are the latent allocation variables, telling us who is in each group. Summing over the independent (d_i), which range over the positive integers, returns the original likelihood function based on the mixture model. The (w_{d_i}) are the stick

breaking mixture weights, with random parameters (v_j), which are i.i.d. as beta$(1, c)$ for some $c > 0$. The idea now, as in [9] and [10], is to marginalize over the weights and locations, to obtain the conditional distribution

$$p(d_1, \ldots, d_n | y_1, \ldots, y_n).$$

For the DPM model, with suitable choice of kernel and prior, this marginalizing can be done explicitly. We will in fact demonstrate this for the components model

$$K(y|\theta) = \mathrm{N}(y|\mu, 1/\lambda),$$

where λ is the reciprocal of the variance, and the prior chosen is the regular conjugate prior for an unknown normal mean and variance; i.e. $\pi(\mu|\lambda) = \mathrm{N}(\nu, \tau^2/\lambda)$ and $\pi(\lambda) = \mathrm{Ga}(a, b)$. We can now obtain

$$p(d_1, \ldots, d_n | y) = \mathbb{E}_{w,\theta} \left\{ \prod_{i=1}^{n} w_{d_i} \, K(y_i|\theta_{d_i}) \right\}.$$

The first expectation is easy to compute;

$$\mathbb{E}_w \left[\prod_{i=1}^{n} w_{d_i} \right] = \prod_{j=1}^{\infty} \mathbb{E}_{v_j} \left[v_j^{n_j} (1 - v_j)^{m_j} \right] = \prod_{j=1}^{\infty} \int v_j^{n_j} (1 - v_j)^{m_j} \, c \, (1 - v_j)^{c-1} \, \mathrm{d}v_j,$$

where $n_j = \#\{d_i = j\}$ and $m_j = \#\{d_i > j\}$. This is given by

$$c^D \prod_{j=1}^{D} \frac{\Gamma(n_j + 1) \, \Gamma(m_j + c)}{\Gamma(n_j + m_j + c + 1)}, \tag{1}$$

where $D = \max\{d_1, \ldots, d_n\}$.

For the second expectation we are interested in computing

$$\mathbb{E}_\theta \left[\prod_{i=1}^{n} K(y_i|\theta_{d_i}) \right],$$

which is comprised of integrals of the type

$$\int \prod_{d_i = j} K(y_i|\theta_j) \, \pi(\mathrm{d}\theta_j).$$

Dropping the subscript j, it is well known that for a sample size of $n(= n_j)$ and data $y = (y_1, \ldots, y_n)$,

$$\int \int \prod_{i=1}^{n} \mathrm{N}(y_i|\mu, 1/\lambda) \, \pi(\mathrm{d}\mu|\lambda) \, \pi(\mathrm{d}\lambda), \tag{2}$$

is given by, up to some constant which does not depend on the data or the (d_i),

$$\left(1/\sqrt{2\pi}\right)^n \, \frac{1}{\sqrt{n + 1/\tau^2}} \, \frac{\Gamma(a + n/2)}{(b + \Psi/2)^{a+n/2}},$$

where

$$\Psi = \sum_{i=1}^{n} y_i^2 + \nu^2/\tau^2 - \nu_n^2(n + 1/\tau^2),$$

and

$$\nu_n = \frac{n\bar{y} + \nu/\tau^2}{n + 1/\tau^2}.$$

Hence, up to a constant of proportionality,

$$p(d|y) \propto c^D \prod_{j=1}^{D} \frac{\Gamma(n_j + 1)\,\Gamma(m_j + c)}{\Gamma(n_j + m_j + c + 1)} \times \prod_{j=1}^{D} \frac{1}{\sqrt{n_j + 1/\tau^2}} \frac{\Gamma(a + n_j/2)}{\Gamma(a)} \frac{b^a}{(b + \Psi_j/2)^{a+n_j/2}}, \tag{3}$$

where

$$\Psi_j = \sum_{d_i=j} y_i^2 + \nu^2/\tau^2 - \nu_j^2(n_j + 1/\tau^2)$$

and

$$\nu_j = \frac{\sum_{d_i=j} y_i + \nu/\tau^2}{n_j + 1/\tau^2}.$$

This is a fully specified probability model for the allocations; if we can sample this $p(d|y)$ then, as we have previously mentioned, we can easily go on and sample the weights and locations and hence sample from the predictive. Quite literally, the allocations are everything. Unfortunately, no-one has yet found a way to sample from $p(d|y)$ exactly without using MCMC. So direct sampling, i.e. obtaining i.i.d. samples, can only be done approximately. We show in section 3 how to obtain an approximate sample from $p(d|y)$.

Before this, we note that we can easily generalize the prior on the random (v_j) to be $v_j \sim \text{beta}(\alpha_j, \beta_j)$. We recover the DP if $\alpha_j = 1$ and $\beta_j = c$; whereas the Pitman–Yor process arises if $\alpha_j = 1 - \delta$ and $\beta_j = c + j\delta$, where $0 \le \delta < 1$ and $c > -\delta$. To ensure that $\sum_{j=1}^{\infty} w_j = 1$ a.s. we need $\sum_{j=1}^{\infty} \log(1 + \alpha_j/\beta_j) = \infty$. Then for E_w we have

$$\begin{aligned} & \prod_{j=1}^{\infty} \int v_j^{n_j}(1 - v_j)^{m_j} \frac{1}{B(\alpha_j, \beta_j)} v_j^{\alpha_j - 1}(1 - v_j)^{\beta_j - 1} \mathrm{d}v_j \\ = & \prod_{j=1}^{D} \frac{B(n_j + \alpha_j, m_j + \beta_j)}{B(n_j, m_j)}, \end{aligned}$$

where $D = \max\{d_1, \cdots, d_n\}$. For further details for such (v_j), see [11].

3. The algorithm

We split this section into two parts; in subsection 3.1 we discuss how to generate approximate samples from the posterior once we have an optimal permutation of the data. In subsection 3.2 we provide the details of the optimal ordering.

3.1. Generating approximate independent samples

The correct way to sample d from $p(d|y)$ is complicated in that it needs MCMC methods. The aim here is to propose a fast alternative to MCMC. To do so, we need to find a good approximation to

the correct $p(d_i|d_1,\ldots,d_{i-1},y_1,\ldots,y_n)$, which can be sampled sequentially. This is taken from [8] and described here. Now

$$\begin{aligned} & p(d_i|d_1,\ldots,d_{i-1},y_1,\ldots,y_n) \\ \propto \quad & p(d_i|d_1,\ldots,d_{i-1},y_1,\ldots,y_{i-1},y_i)\times p(y_{i+1},\ldots,y_n|d_i,d_{i-1},\ldots,d_1,y_1,\ldots,y_i). \end{aligned}$$

In $p(d_i|d_1,\ldots,d_{i-1},y_1,\ldots,y_{i-1},y_i)$, the d_i only depends on observations $(y_j, j=1,\ldots,i)$. Therefore, sampling d_i from this conditional can be carried out without MCMC. To use this approximation we have left out the term

$$p(y_{i+1},\ldots,y_n|d_i,d_{i-1},\ldots,d_1,y_1,\ldots,y_i),$$

which is the key to the approximation and therefore should not vary much as d_i varies. Whether this does or not is a difficult problem to assess exactly however the simulation study provided in section 5 confirms that the approximation is good, as we recover good estimates for the predictive.

To simplify the notation we will just use $p(d_1,\ldots,d_i)$ to denote the conditional density of $(d_1,\ldots,d_i)$ on $(y_1,\ldots,y_i)$. The basis for generating independent samples from $p(d_1,\ldots,d_i)$ is that the actual labels of the allocation variables are largely irrelevant. It is the groupings of the data which matters. Hence, we can take $d_1=1$ and then subsequently, for each $1<i\leq n$, we take d_i from the set $d_i\in S_{i-1}=\{d_1,\ldots,d_{i-1},k_i+1\}$, where $k_i=\max\{d_1,\ldots,d_{i-1}\}$. In essence, we are constraining the (d_i) to be taken from the first k integers only, when k is the number of distinct values.

We can find the probabilities to provide a single sequence $(d_1,d_2,\ldots,d_n)$. So we take $d_1=1$ and then compute

$$p(1,j) \quad \text{for} \quad j=1,2.$$

We sample d_2 as $j=1$ with probability

$$\frac{p(1,1)}{p(1,1)+p(1,2)},$$

else we take $j=2$.

Then we compute, in general, and given the values $(d_1,\ldots,d_{i-1})$,

$$p(d_1,\ldots,d_{i-1},j) \quad \text{for} \quad j=1,\ldots,k_i+1$$

where $k_i=\max\{d_1,\ldots,d_{i-1}\}$. We then take $d_i=j$ with probability

$$\frac{p(d_1,\ldots,d_{i-1},j)}{\sum_{j=1}^{k_i+1}p(d_1,\ldots,d_{i-1},j)}.$$

This process up to $i=n$ yields a single (approximate) sample $(d_1,\ldots,d_n)$ from the posterior $p(d|y)$. The aim is to take, for example, up to M such sequences, in order to provide a predictive density estimator. Running this in parallel, so that we obtain the M independent sequences in parallel which is important and necessary, and the command in MATLAB uses the `parfor` loop in order to generate M

sets of $(d_1, \ldots, d_n)$. The code returns us $n \times M$ values.

Once we have a sequence of (d_i) from $p(d|y)$ we can find the predictive density explicitly; it is given by

$$f(y) = \sum_{j=1}^{D} \bar{w}_j \, f_j(y) + \left(1 - \sum_{j=1}^{D} \bar{w}_j\right) f_0(y), \tag{4}$$

where $D = \max\{d_1, \ldots, d_n\}$, $\bar{w}_j = n_j/(c+n)$, and the weight on the second term is $c/(c+n)$,

$$f_j(y) = \mathrm{E}_{\pi_j} \mathrm{N}\big(y|\mu_j, \lambda_j^{-1}\big), \tag{5}$$

where

$$\pi_j(\mu_j, \lambda_j|d, y) = \mathrm{N}\left(\mu_j \,\middle|\, \nu_j, \frac{1}{\lambda_j\,(n_j + 1/\tau^2)}\right) \mathrm{Ga}(\lambda_j|a + n_j/2, b + \Psi_j/2), \tag{6}$$

and $f_0(y)$ is the prior predictive. The $f_j(y)$, including the $f_0(y)$, is available explicitly as a Student-t density

$$f_j(y) \propto \left(b_j + \tfrac{1}{2}(y - \nu_j)^2/(1+\tau^2)\right)^{-a_j - \frac{1}{2}}, \tag{7}$$

where $a_j = a + n_j/2$ and $b_j = b + \frac{1}{2}\Psi_j$.

The algorithm just described is similar in to the one presented in [8]. The difference is that [8] select the (d_i) by taking the d_i which maximizes

$$p(d_i, d_1, \ldots, d_{i-1}).$$

In the next section we will propose an optimal ordering of the data.

In both algorithms, i.e. ours and [8], the ordering by arbitrarily setting $d_1 = 1$ has an effect and changing this, to say putting $d_2 = 1$, will change the estimation, which is hard to assess completely. Consequently, [8] proposed to perform a number of random permutations of the orderings and then select the permutation which provides the best fit according to a pseudo-marginal likelihood criterion. On the other hand, we find an optimal ordering based on a sequential maximization routine, effectively selecting the order which maximizes the sequence of predictive density functions.

3.2. An optimal ordering of the data

Here we decompose $p(d, y)$ as

$$p(d_1, y_1)\, p(d_2, y_2|d_1, y_1) \ldots p(d_n, y_n|d_1, y_1, \ldots, d_{n-1}, y_{n-1}). \tag{8}$$

We sequentially maximize, with respect to both the y and d, over each term in (8). To explain this, let us look at the first term. Noting that we can write $p(d_1, y_1) = f(y_1)\, p(d_1|y_1)$, we take y_1 to be the observation which maximizes the prior predictive $f(y)$. We then allocate $d_1 = 1$.

We then proceed to find $y_2 \neq y_1$ which maximizes $f(y|d_1, y_1)$ and then take d_2 which maximizes, for

$j = 1, 2$,

$$\mathrm{P}(d_2 = j | d_1 = 1, y_1) \propto f_j(y_1)\, \mathrm{P}(d_2 = j | d_1 = 1)$$

where $f_1(y_1)$ will be described shortly as will $\mathrm{P}(d_2 = j | d_1 = 1)$.

In general, we take $y_k \neq y_1, \ldots, y_{k-1}$ to maximize

$$f(y | d_1, y_1, \ldots, d_{k-1}, y_{k-1}), \tag{9}$$

and then take d_k from $\{1, \ldots, K+1\}$, where $K = \max\{d_1, \ldots, d_{k-1}\}$, to maximize

$$\mathrm{P}(d_k = j | d_1, y_1, \ldots, d_{k-1}, y_{k-1}, y_k) \propto f_j(y_k)\, \mathrm{P}(d_k = j | d_1, \ldots, d_{k-1}). \tag{10}$$

Both (9) and (10) are easy to compute; see section 3.1. So (9) is the predictive given in (4) but with $k-1$ replacing n. All other aspects stay the same. And for (10), we have

$$\mathrm{P}(d_k = j | d_1, \ldots, d_{k-1}) = \frac{n_j}{c + k - 1},$$

for $j = 1, \ldots, K$, and $P(d_k = K+1 | d_1, \ldots, d_{k-1}) = c/(c+k-1)$.

With this optimal ordering, we can then obtain the approximate sample (d_i) from $p(d|y)$ and then we can obtain an explicit form for the predictive density.

We use the Kullback–Leibler divergence to assess the quality of the density estimator. So if f^* denotes the true density then we would consider comparative values of

$$\int f^*(y)\, \log \widehat{f}(y)\, dy,$$

where $\widehat{f}$ denotes the density estimate. If we approximate this by

$$n^{-1} \sum_{i=1}^{n} \log \widehat{f}(y_i), \tag{11}$$

we have that $\widehat{f}(y_i)$ depends on an ordering of the data, specifically

$$\widehat{f}(y_i) = \widehat{f}(y_i | y_{\sigma(1)}, \ldots, y_{\sigma(n)}),$$

for some permutation σ on $\{1, \ldots, n\}$. Obtaining the optimal σ here, i.e. maximize (11), is too difficult, as there is no sequential route.

Our next best agenda which does admit an easy to obtain optimal σ is to select the σ to maximize the marginal likelihood

$$p(y_{\sigma(1)}, \ldots, y_{\sigma(n)}).$$

This can be written sequentially as

$$\prod_{i=1}^{n} p(y_{\sigma(i)}|y_{\sigma(1)},\ldots,y_{\sigma(i-1)}).$$

Specifically, we include the allocation variables in this maximization; so we find sequentially the optimal σ by maximizing

$$p(y_{\sigma(i)}|y_{\sigma(1)},\ldots,y_{\sigma(i-1)},d_\sigma(1),\ldots,d_{\sigma(i-1)}),$$

and then maximizing

$$p(d_{\sigma(i)}|y_{\sigma(1)},\ldots,y_{\sigma(i-1)},d_\sigma(1),\ldots,d_{\sigma(i-1)},y_{\sigma(i)}),$$

iteratively from 1 to n.

This gives a σ and hence an ordering, optimal with respect to the value of the marginal likelihood. It is common for selection procedures to be based on the maximizing of a marginal likelihood; for example in empirical Bayes estimation of the prior.

So though this marginal likelihood optimal σ is not necessarily the one which maximizes (11), we believe it should be a good proxy for it. This is substantiated from the simulation studies.

4. Regression model

The theory we have developed in sections 2 and 3 can apply equally to a particular type of regression model. So suppose we have the model for $(y_i, x_i)_{i=1}^n$, where x_i is a $m \times p$ matrix of covariates, and y_i is a $m \times 1$ vector of observations, as

$$f_{P,\lambda}(y|x) = \int \mathrm{N}(y|x\beta, \lambda^{-1})\,\mathrm{d}P(\beta),$$

with the usual Dirichlet process prior on P, yielding

$$f(y|x) = \sum_{j=1}^{\infty} w_j\,\mathrm{N}(y|x\beta_j, \lambda^{-1}).$$

Now we need to find the corresponding version of (2); and for this we will remove the subscript j and assume the number of observations in component j is n_j. So

$$y_i \sim \mathrm{N}_m(\cdot|x_i\beta, \lambda^{-1}),$$

where β is a $p \times 1$ vector of coefficients, with prior distributions for β and λ given by

$$\pi(\beta|\lambda) = \mathrm{N}(\cdot|\mu, \Sigma/\lambda) \quad \text{and} \quad \lambda \sim \mathrm{Ga}(\cdot|a, b)$$

for given values of (μ, Σ, a, b).

Then, standard integration yields

$$\int\int \prod_{i=1}^{n} \mathrm{N}(y_i|x_i\beta, \lambda^{-1})\, \mathrm{N}(\mathrm{d}\beta|\mu, \Sigma/\lambda)\, \mathrm{Ga}(\mathrm{d}\lambda|a, b)$$

is given by, ignoring constants which will be common to all components,

$$|\Omega|^{p/2}\, \frac{\Gamma(a + nm/2)}{(b + y'y + \mu'\Sigma^{-1}\mu - \nu'\Omega\nu)^{a+nm/2}}, \tag{12}$$

where

$$\Omega = \left(\sum_{i=1}^{n} x_i'x_i + \Sigma^{-1}\right)^{-1} \quad \text{and} \quad \nu = \Sigma^{-1}\mu + \sum_{i=1}^{n} x_i'y_i.$$

This then allows us to run the algorithm described in section 3 for the regression model by simply swapping (2) for (12), and an obvious modification of (6).

5. Illustrations

Here we evaluate the performance of the proposed algorithm based on the optimal ordering (OO) and compare with two other methods; one is the standard MCMC algorithm with slice sampling (see [12]) and the other is the SUGS algorithm by [8]. For the simulated data, we sample observations from the true density given in section 5.1. We also demonstrate the optimal sampler algorithm on the galaxy data set in section 5.2, and to a real industry dataset in the regression case in section 5.3.

Before proceeding, we describe how in each case, i.e. MCMC, SUGS and our algorithm, we obtain the estimates which appear in the figures. For the MCMC algorithm, at each iteration, a (d_i) will be sampled, and then the predictive as a mixture of Student-t densities can be computed exactly, using (4) and (7). The final predictive is averaged over all iterations. SUGS obtains a (d_i) from a permutation and hence gets the mixture of Student-t predictive from each permutation. A best predictive is then selected from all those obtained from a number of permutations using the marginal likelihood criterion. For our algorithm, we obtain the optimal permutation and then simulate a number of the (d_i), each giving a predictive, and then we average over all these predictives. Also note that the (d_i) obtained are from the correct posterior with the MCMC algorithm; the errors in SUGS and our algorithm are that the (d_i) are obtained approximately according to the ideas in section 3.1.

5.1. Simulated example

To evaluate the performance of the proposed algorithm, we consider the following true density which is a mixture of three normal distributions.

$$f(y) = 0.3\mathrm{N}(y; -2, 0.4) + 0.5\mathrm{N}(y; 0, 0.3) + 0.2\mathrm{N}(y; 2.5, 0.3).$$

This mixture was also used by [8].

We take the sample sizes to be $n = 100$, $n = 200$ and $n = 500$ and for each case, we consider 100 simulated datasets. For all the analysis reported here, we used $c = 5$ and choose the prior for λ as

sample size	SUGS	MCMC	OO
100	0.0289	0.0321	0.0173
200	0.0155	0.0233	0.0091
500	0.0061	0.0152	0.0079

Table 1: The average KLD results of three algorithms based on three sample sizes $n = 100$, $n = 200$ and $n = 500$ with 100 repetitions.

data size	SUGS	OO
100	9.52	7.80
200	19.22	21.86
500	46.86	102.05

Table 2: The comparison of the time consumed (in seconds) of running SUGS and OO algorithms under the same condition based on three sample sizes $n = 100$, $n = 200$ and $n = 500$.

$\pi(\lambda) = \text{Ga}(a, b)$, with $a = 1.28 \times \ln n$ and $b = 0.5$, and $\tau^2 = 10$, in an attempt to be noninformative.

Using the algorithm scheme outlined in section 3, once we have the optimal ordering of the data, we can obtain the predictive density using (4) and (7). We then average this over 100 predictives through sampling 100 sets of (d_i) from the algorithm in section 3.1. The predictive estimates for the three sample sizes, compared with the estimates provided by the MCMC algorithm and SUGS, are given in Figs. 1 to 3. For the MCMC algorithm, we take $c = 0.8$, $a = 5$, $b = 1$ and $\tau^2 = 25$ as defaults. SUGS was repeated for 100 random orderings.

To measure the closeness of the proposed density estimates to the true densities, we calculate the Kullback-Leibler divergence (KLD) between densities f and g, defined as follows

$$K(f, g) = \int f(x) \log \left\{ \frac{f(x)}{g(x)} \right\} \mathrm{d}x$$

with f being the true density and g being an estimate. Table 1 summarizes the averages of calculated KLDs of three algorithms: the SUGS algorithm of [8]; MCMC with a slice sampling algorithm, and the optimal ordering algorithm, proposed in this paper. Also, we compare the speed with SUGS and present the results in Table 2.

To check if in general the proposed ordering scheme through the sequential maximization step would deliver the optimal (in terms of KLD) ordering of the data, we propose a simple simulation study focused on a small sample. We take 100 simulated datasets of length $n = 20$ from the above mixture of three normals. Then we calculate the KLDs of our approach and SUGS for each dataset. For each dataset, SUGS will pick the best result from 100 random permutations of the data. The results show that, within the 100 datasets, our approach outperformed SUGS 99 times.

For sample sizes up to around 200, we are highly competitive with SUGS in terms of both accuracy and speed; whereas around 500 sample size, we see a reversal. At 500 the exact permutation does not have an effect on the accuracy, so doing a few in number is not going to have a detrimental outcome. And

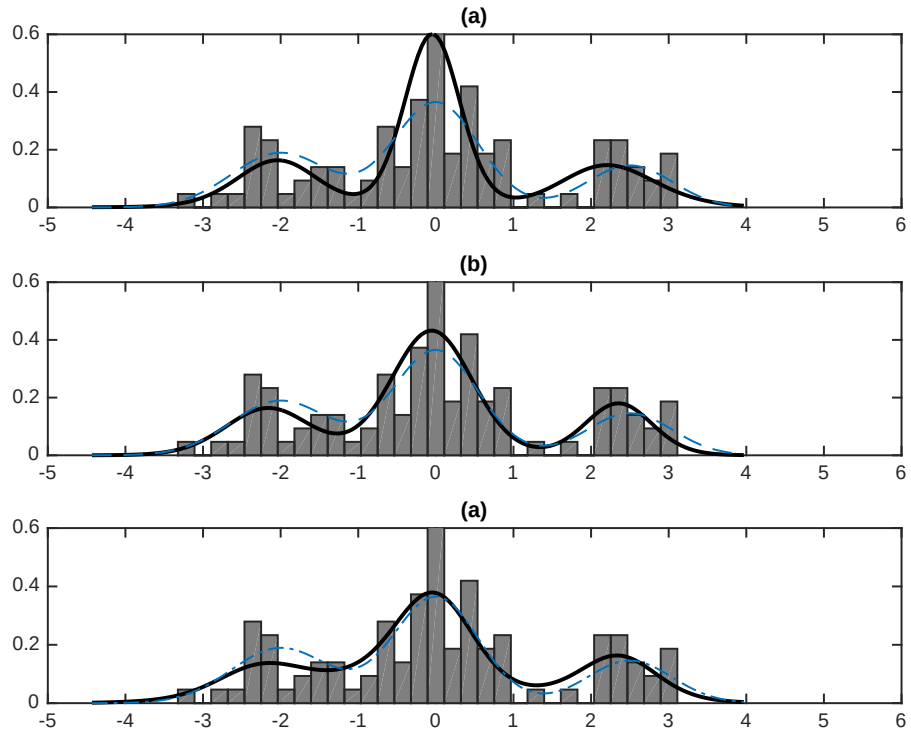

Figure 1: Comparison of the simulated data (n=100) with predictives (MCMC, SUGS and OO). Panel (a): the true density (dash) and the density estimates of MCMC method (solid); Panel (b): the true density (dash) and the density estimates of SUGS method (solid); panel (c): the true density (dash) and the density estimates of OO method using parfor, respectively.

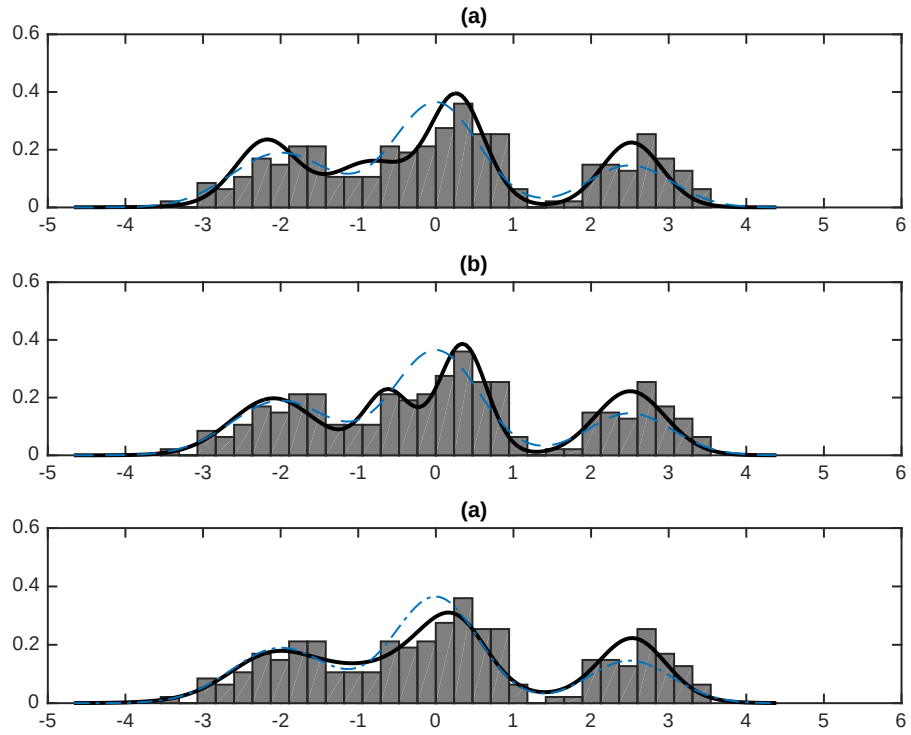

Figure 2: Comparison of the simulated data (n=200) with predictives (MCMC, SUGS and OO). Panel (a): the true density (dash) and the density estimates of MCMC method (solid); Panel (b): the true density (dash) and the density estimates of SUGS method (solid); panel (c): the true density (dash) and the density estimates of OO method using parfor, respectively.

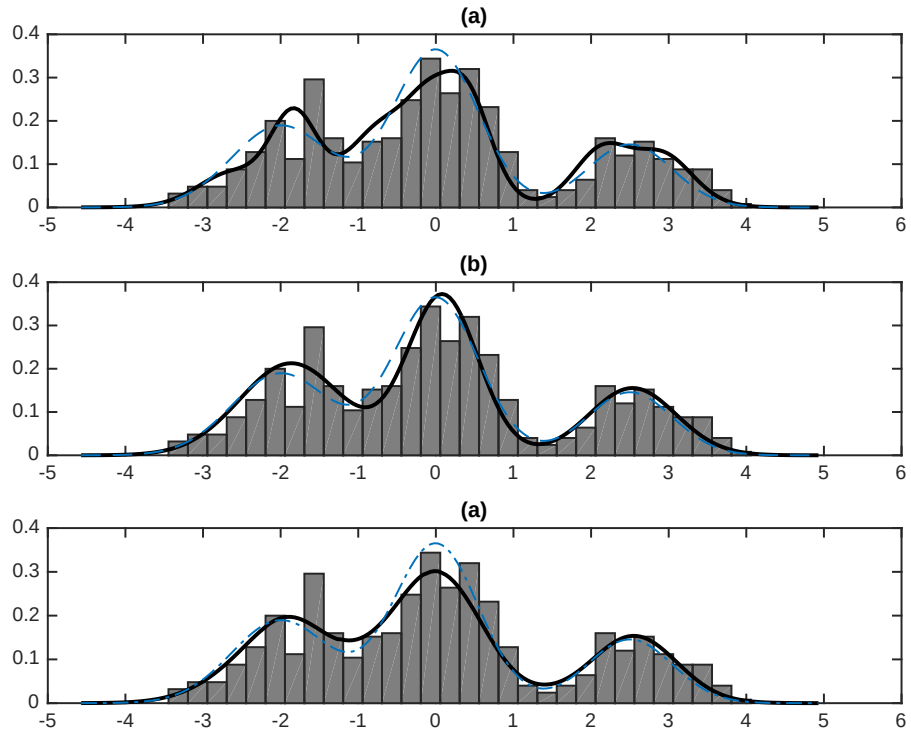

Figure 3: Comparison of the simulated data (n=500) with predictives (MCMC, SUGS and OO). Panel (a): the true density (dash) and the density estimates of MCMC method (solid); Panel (b): the true density (dash) and the density estimates of SUGS method (solid); panel (c): the true density (dash) and the density estimates of OO method using parfor, respectively.

then SUGS also picks out the best predictive whereas we estimate by averaging. However, for smaller sample sizes obtaining an optimal ordering appears the best choice.

5.2. Galaxy data set

It would be remiss for us not to estimate predictive densities using the widely studied galaxy data, which are the measured velocities of 82 galaxies; see, for example, the works of [13] and [14]. The sample size of the data is $n = 82$, which includes six will-separated conic sections of space. We use the priors with the same parameters set. The estimated predictive density functions are given in Fig. 4. This is compared with the MCMC estimate, the kernel smoothed density estimate, and the SUGS estimate. Our method gave five distinctive clusters, which agrees with the results of [13] and [14]. Also, the runtime (in seconds) of our method is very close to SUGS, which agrees with the findings in Table 2.

5.3. Regression case

The regression model we consider here comes from a real life case study between researchers at the University of Kent and a manufacturer of precision sensors; supported by a Knowledge Transfer Project and funded by the Technology Strategy Board, UK. The company manufacturing high precision sensors had accumulated huge historical databases on a type of sensor.

For the dataset we consider, there are response vectors $(\mathbf{y}_i)$ and input vectors $(\mathbf{x}_{i1}, \ldots \mathbf{x}_{ip})$ for each sensor i, where $i = 1, \ldots, n = 202$ denotes the number of sensors in the dataset and p denotes the number of independent factors related to the response. In practice, each $\mathbf{y}_i$ is a 32×1 vector and $X_i = (\mathbf{x}_{i1}, \ldots \mathbf{x}_{ip})$ is a 32×5 matrix. The model relating response and input vectors is the linear regression model:

$$\mathbf{y}_i = X_i \beta_i + \varepsilon_i, \tag{13}$$

where the ε_i is a 32×1 vector of independent errors all with a normal distribution with zero mean and variance λ^{-1}. The β_i is a 5×1 vector of regression coefficients; a different set for each i.

The ordinary least square (OLS) estimates of the parameters (β_i) demonstrate that the values exhibit multi-modality. Figure 5 shows the histogram of the 202 OLS estimates of the parameters (β_{i3}), where β_{i3} is the third component in the vector $\beta_i = (\beta_{i1}, \ldots, \beta_{i5})$ and $i = 1, \ldots, 202$. Thus in a Bayesian analysis, using a multivariate normal distribution for these random effects would be insufficient. Hence a large model for the distribution of the (β_i) is needed and we take this to be a Dirichlet process.

Consequently, the nonparametric regression model we use here is given by

$$f(\mathbf{y}|X) = \sum_{j=1}^{\infty} w_j \, \mathrm{MVN}_{32}(\mathbf{y}|X\beta_j, I_m/\lambda),$$

where the prior for each β_j is given by

$$\pi(\beta_j) = \mathrm{MVN}_5(\beta_0, \Sigma/\lambda)$$

and with $\pi(\lambda) = \mathrm{Ga}(\lambda|a, b)$. In the analysis we use the prior settings of $a = 3$ and $b = 1$ with $c = 50$.

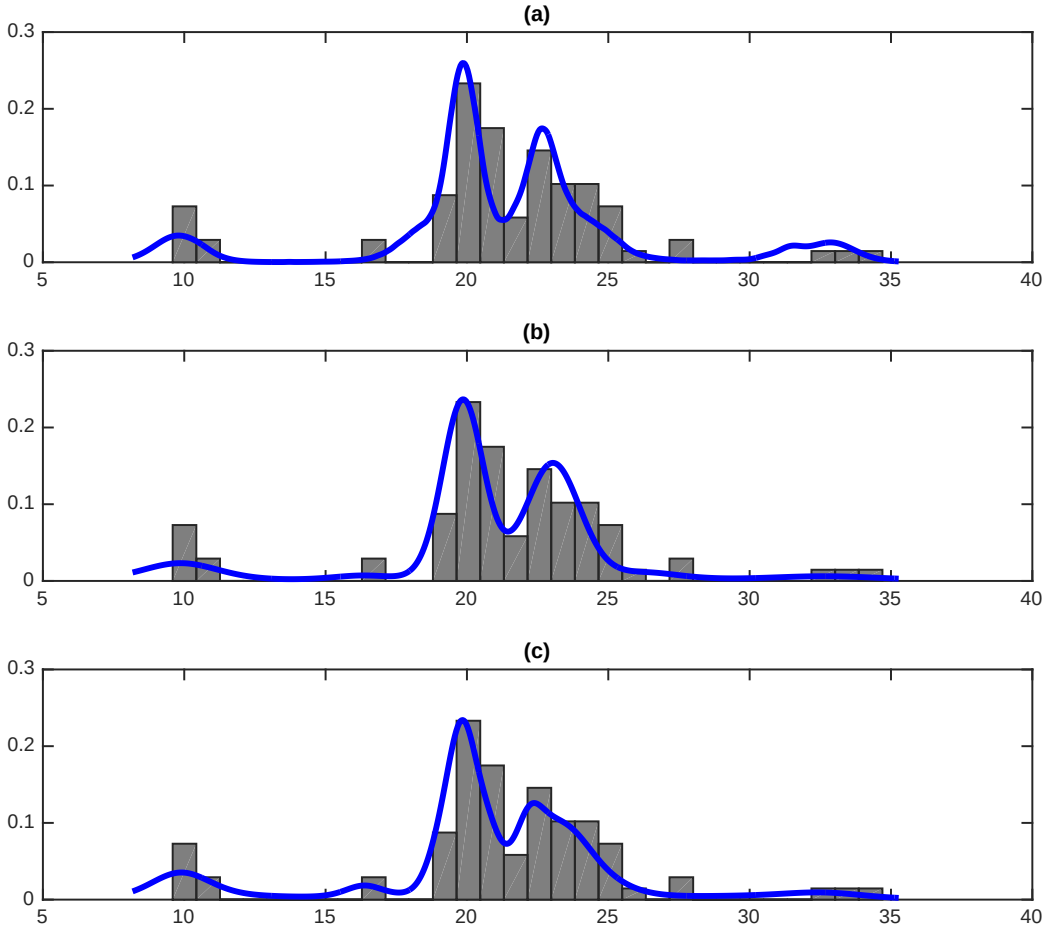

Figure 4: Comparison of the galaxy data with predictives (MCMC, SUGS and OO). Panel (a): the density estimates of MCMC method (solid); Panel (b): the density estimates of SUGS method (blue); panel (c): the density estimates of OO method using parfor, respectively.

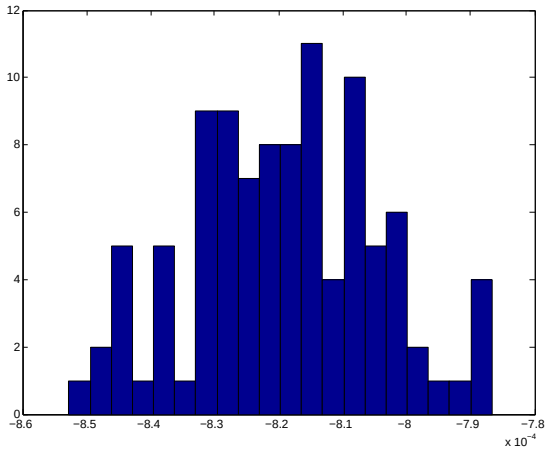

Figure 5: Histogram of 100 estimates of (β_{i3}), the third component in the vector (β_i), for $i = 1, \ldots, 100$.

The prior parameters (β_0, Σ) are derived from further available historical data and specifically

$$\beta_0' = (1.0001, 0.0004, -0.008, 0, 0)$$

and

$$\Sigma = \begin{pmatrix} 0.049 & 0.009 & -0.004 & 0 & 0 \\ 0.009 & 0.017 & -0.007 & 0 & 0 \\ -0.004 & -0.007 & 0.007 & 0 & 0 \\ 0 & 0 & 0 & 2 \times 10^{-6} & 0 \\ 0 & 0 & 0 & 0 & 4 \times 10^{-7} \end{pmatrix}$$

Following a similar procedure as the independent case, we generate $M = 100$ sequences of $(d_1, \ldots, d_{202})$. For each such sequence we sample the corresponding parameters (w, β, λ). Once we have these samples, we can sample predictive samples of the "next" sensor, i.e. sensor 203, which has a particular covariate value and in reality we do know what the observations from this sensor are to allow a comparison of the predictive sample with the true observations. Figure 6 shows the comparison of a true response with 100 predictive samples, and the fit is good.

6. Discussion

In this paper we have presented a fast approximate simulation of the MDP model which uses sequential conditional maximization routines to obtain an optimal ordering for the data. This is the key novelty of the paper. The algorithm does not use MCMC. We use the prior model for weights and locations to be the usual conjugate style priors. With such a prior we can, once we have a set of allocation variables, obtain an explicit form for the predictive density function; namely a mixture of Student-t density functions. The allocation variables can be sampled approximately in quite an easy way; for each set of variables we have an explicit form for the predictive and our overall estimate of the density is an average over these predictives.

We have compared our algorithm with both MCMC and SUGS. The idea is that our algorithm and SUGS do not use MCMC and hence are time-wise faster than MCMC. Our algorithm obtains an optimal ordering of the data and hence does not need multiple permutations which SUGS needs. SUGS then uses one (d_i) per permutation whereas we sample many (d_i) and average over the ensuing predictives.

Supplementary material. MATLAB code for the algorithm, with instructions, and the data sets for all the illustrations in section 5 are provided in the supplementary material.

References

[1] A. Y. Lo, On a class of Bayesian nonparametric estimates I: Density estimates, Annals of Statistics 12 (1984) 351–357.

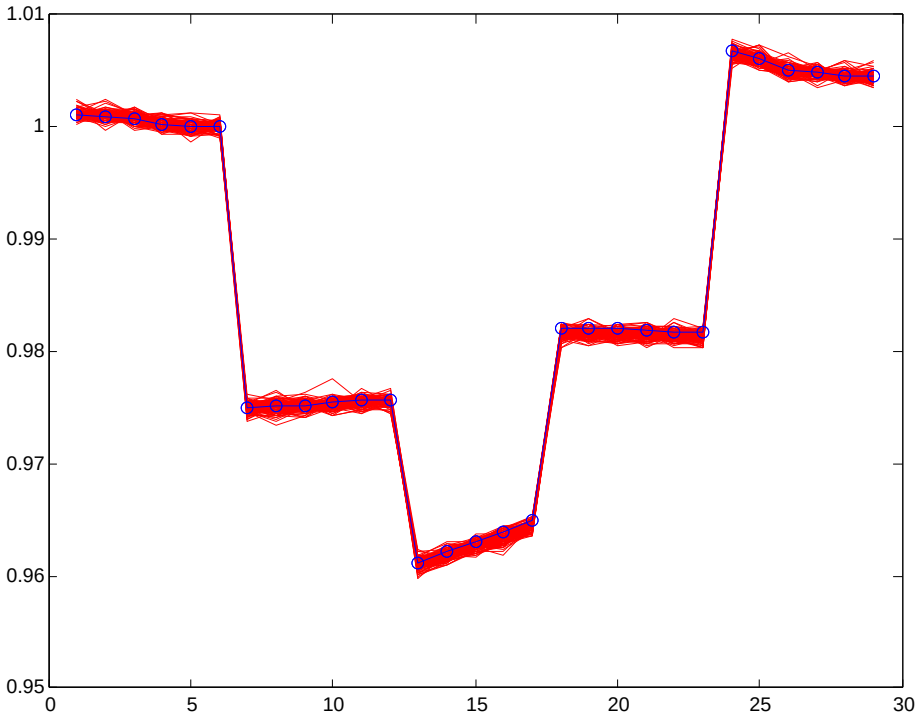

Figure 6: Comparison of the true response with predictives. The dotted line shows the true response and the solid lines show the 100 predictive values.

[2] T. S. Ferguson, A Bayesian analysis of some nonparametric problems, Annals of Statistics 1 (1973) 207–230.

[3] J. Sethuraman, A constructive definition of Dirichlet priors, Statistica Sinica 4 (1994) 639–650.

[4] M. D. Escobar, Estimating the means of several normal populations by nonparametric estimation of the distribution of the means, in: Unpublished PhD dissertation, Department of Statistics, Yale University, 1988.

[5] N. L. Hjort, C. C. Holmes, P. Müller, S. G. Walker (Eds.), Bayesian Nonparametrics, Cambridge University Press, Cambridge, 2010.

[6] M. A. Newton, Y. Zhang, A recursive algorithm for nonparametric analysis with missing data, Biometrika 86 (1999) 15–26.

[7] M. A. Newton, On a nonparametric recursive estimator of the mixing distribution, Sankhya 64 (2002) 306–322.

[8] L. Wang, D. B. Dunson, Fast Bayesian inference in Dirichlet process mixture models, Journal of Computational and Graphical Statistics 20 (2011) 196–216.

[9] C. A. Bush, L. J., S. N. MacEachern, Minimally informative prior distributions for non-parametric analysis, Journal of the Royal Statistical Society, Series B 72 (2010) 253–268.

[10] R. Fuentes-Garcia, R. H. Mena, S. G. Walker, A new Bayesian nonparametric mixture model, Communications in Statistics 39 (2010) 669–682.

[11] H. Ishwaran, L. F. James, Gibbs sampling methods for stick-breaking priors, Journal of the American Statistical Association 96 (2001) 161–173.

[12] M. Kalli, J. E. Griffin, S. G. Walker, Slice sampling mixture models, Statistics and Computing 21 (2011) 93–105.

[13] M. D. Escobar, M. West, Bayesian density estimation and inference using mixtures, Journal of the American Statistical Association 90 (1995) 577–588.

[14] S. Richardson, P. J. Green, On Bayesian analysis of mixtures with an unknown number of components, Journal of Royal Statistical Society, Series B 59 (1997) 731–792.