J. R. Statist. Soc. B (2007)
69, *Part* 4, *pp.* 589–605

On-line inference for multiple changepoint problems

Paul Fearnhead and Zhen Liu

Lancaster University, UK

[Received October 2005. Final revision March 2007]

Summary. We propose an on-line algorithm for exact filtering of multiple changepoint problems. This algorithm enables simulation from the true joint posterior distribution of the number and position of the changepoints for a class of changepoint models. The computational cost of this exact algorithm is quadratic in the number of observations. We further show how resampling ideas from particle filters can be used to reduce the computational cost to linear in the number of observations, at the expense of introducing small errors, and we propose two new, optimum resampling algorithms for this problem. One, a version of rejection control, allows the particle filter to choose the number of particles that are required at each time step automatically. The new resampling algorithms substantially outperform standard resampling algorithms on examples that we consider; and we demonstrate how the resulting particle filter is practicable for segmentation of human G+C content.

Keywords: Direct simulation; Isochores; Particle filtering; Rejection control; Sequential Monte Carlo methods; Stratified sampling

1. Introduction

Changepoint models are commonly used to model heterogeneity of data (usually over time). Given a set of observations collected over time, these models introduce a (potentially random) number of changepoints which split the data into a set of disjoint segments. It is then assumed that the data arise from a single model within each segment, but with different models across the segments. Examples of changepoint problems include Poisson processes with changing intensity (Ritov *et al.*, 2002), changing linear regressions (Lund and Reeves , 2002), Gaussian models with changing variance (Johnson *et al.*, 2003) and Markov models with time varying transition matrices (Braun and Muller, 1998). These models have been applied to problems in a range of areas, including engineering, physical and biological sciences and finance.

We consider Bayesian inference for changepoint models where the number of changepoints is unknown. The most common approach to such inference for these models is the use of Markov chain Monte Carlo (MCMC) methods (see for example Chib (1998) and Stephens (1994)) and reversible jump MCMC methods (Green, 1995). However, for many changepoint models (e.g. the models in Green (1995) and Punskaya *et al.* (2002)) it is possible to simulate independent realizations directly from the posterior distribution. This idea was used for deoxyribonucleic acid (DNA) segmentation by Liu and Lawrence (1999) and has been proposed more generally by Fearnhead (2005, 2006). The ideas for direct simulation are based on exact methods for calculating posterior means (Barry and Hartigan, 1992). The advantages of direct simulation methods over MCMC and reversible jump MCMC methods are that

Address for correspondence: Paul Fearnhead, Department of Mathematics and Statistics, Fylde College, Lancaster University, Lancaster, LA1 4YF, UK.
E-mail: p.fearnhead@lancaster.ac.uk

(a) there is no need to diagnose whether the MCMC algorithm has converged, and
(b) as the draws from the posterior distribution are independent it is straightforward to quantify uncertainty in estimates of features of the posterior distributions based on them.

For examples of the potential difficulties with MCMC methods caused by (a), compare the inferences that were obtained for the coal-mining disaster data which were analysed in Green (1995) with those based on the direct simulation method of Fearnhead (2006).

In this paper we extend the direct simulation algorithms to on-line problems, where the data are obtained incrementally over time, and new inferences are required each time that an observation is made. The use of on-line algorithms has also been suggested for static problems (e.g. Chopin (2002) and Del Moral *et al.* (2006)).

The computational cost of our exact on-line algorithm increases linearly over time; however, the on-line version of direct simulation is similar to particle filter algorithms, and we consider using resampling algorithms taken from particle filters to reduce the computational cost of our direct simulation algorithm (at the expense of introducing error). We propose two simple extensions of existing resampling algorithms that are particularly well suited to changepoint models. One is a stratified extension of the rejection control (RC) approach of Liu *et al.* (1998), which can limit the maximum amount of error that is introduced by each resampling step. In simulation studies we find that this stratified version can reduce the error of the resulting algorithm by about a third compared with the non-stratified version.

The resulting on-line algorithm can be viewed as a Rao–Blackwellized version of the particle filter algorithm of Chopin (2006): we have integrated out the parameters that are associated with each segment. Note that this is an extremely efficient version of Rao–Blackwellization. For example, consider analysing n data points. Our algorithm with n particles will give exact inference and thus will always outperform the algorithm of Chopin (2006) regardless of the number of particles that are used in that particle filter. Note, however, that the filter of Chopin (2006) can be used more widely, as it does not require that the parameters within each segment can be integrated out.

The outline of the paper is as follows. We introduce the class of changepoint models that we consider in Section 2. In Section 3 we introduce the on-line direct simulation algorithm and approximate versions of it that utilize various resampling ideas. We test these approximate algorithms, and compare different resampling algorithms, on simulated data in Section 4 before applying our algorithm to the analysis of the C+G structure of human DNA data (Section 5). The paper concludes with a discussion.

2. Models and notation

Assume that we have data $\mathbf{y}_{1:n} = (y_1, y_2, \ldots, y_n)$. We consider changepoint models for the data with the following conditional independence property: given the position of a changepoint, the data before that changepoint are independent of the data after the changepoint. These models can be described in terms of the following hierarchical structure.

Firstly we model the changepoint positions via a Markov process. This Markov process is determined by a set of transition probabilities,

$$\Pr(\text{next changepoint at } t \mid \text{changepoint at } s). \qquad (1)$$

For this paper we make the simplification that these transition probabilities depend only on the distance between the two changepoints. Extending our work to the general case, where the distribution of the length of a segment could depend on the time at which it starts, is straightforward. Specifically we let $g(\cdot)$ be the probability mass function for the distance between two successive changepoints (equivalently the length of segments), so that expression (1) is $g(t-s)$. We further let $G(l)=\Sigma_{i=1}^{l} g(i)$ be the distribution function of this distance and assume that $g(\cdot)$ is the probability mass function for the position of the first changepoint.

Any such model implies a prior distribution on the number of changepoints. For example, if a geometric distribution is used for $g(\cdot)$, then our model implies that there is a fixed probability of a changepoint at any time point, independent of other changepoints. Hence this model implies a binomial distribution for the number of changepoints.

Now we condition on m changepoints at times $\tau_1, \tau_2, \ldots, \tau_m$. We let $\tau_0=0$ and $\tau_{m+1}=n$, so our changepoints define $m+1$ segments, with segment i consisting of observations $\mathbf{y}_{\tau_i+1:\tau_{i+1}}$ for $i=0,\ldots,m$. We allow a set of $\bar{p}$ possible models for the data from each segment, labelled $\{1,2,\ldots,\bar{p}\}$, and assume an arbitrary prior distribution for models, which is common across segments, with the model in a given segment being independent of the models in all other segments.

For a segment consisting of observations $\mathbf{y}_{s+1:t}$ and model q we shall have a set of unknown parameters, β say. We have a prior distribution, $\pi(\beta)$ for β (which may depend on q), but we assume that the parameters for this segment are independent of the parameters in other segments. Finally we define

$$P(s,t,q)=\int \Pr(\mathbf{y}_{s+1:t}|\beta, \text{model } q)\, \pi(\beta)\, \mathrm{d}\beta \tag{2}$$

and assume that these probabilities can be calculated for all $s<t$ and q. This requires either conjugate priors for β or the use of numerical integration. Numerical integration is often computationally feasible in practice if the dimension of the part of β that cannot be integrated analytically is low (see Fearnhead (2006) for an example).

For concreteness we describe a specific example of this model which we shall use in Section 4 (see also Punskaya *et al.* (2002) and Fearnhead (2005)). Here we model the data as piecewise linear regressions. So within a segment we have a linear regression model of unknown order. For a given model order q, we have

$$\mathbf{y}_{s+1:t}=\mathbf{H}\beta+\varepsilon \tag{3}$$

where $\mathbf{H}$ is a $(t-s)\times q$ matrix of basis functions, β is a vector of q regression parameters and ε is a vector of independent and identically distributed Gaussian noise with mean 0 and variance σ^2.

We assume conjugate priors. The variance of the Gaussian noise has an inverse gamma distribution with parameters $\nu/2$ and $\gamma/2$, and the components of the regression vector have independent Gaussian priors. The prior for the jth component is Gaussian with mean 0 and variance $\sigma^2\delta_j^2$.

The likelihood function of $\mathbf{y}_{s+1:t}$ conditional on a model q is obtained by integrating out the regression parameters β and variance σ^2:

$$P(s,t,q)=\pi^{-(t-s)/2}\left(\frac{|\mathbf{M}|}{|\mathbf{D}|}\right)^{1/2}\frac{\gamma^{\nu/2}}{(\|\mathbf{y}_{s+1:t}\|_{\mathbf{P}}^2+\gamma)^{(t-s+\nu)/2}}\,\frac{\Gamma\{(t-s+\nu)/2\}}{\Gamma(\nu/2)}, \tag{4}$$

where $\mathbf{M}=(\mathbf{H}^{\mathrm{T}}\mathbf{H}+\mathbf{D}^{-1})^{-1}$, $\mathbf{P}=(\mathbf{I}-\mathbf{HMH}^{\mathrm{T}})$, $\|\mathbf{y}\|_{\mathbf{P}}^2=\mathbf{y}^{\mathrm{T}}\mathbf{P}\mathbf{y}$, $\mathbf{D}=\mathrm{diag}(\delta_1^2,\ldots,\delta_q^2)$ and $\mathbf{I}$ is a $(t-s)\times(t-s)$ identity matrix.

3. On-line inference

We consider on-line inference for the multiple changepoint model of Section 2. We assume that observations accrue over time, so that y_t is the observation at time t. At each time step, our aim is to calculate the posterior distributions of interest on the basis of all the observations to date. To do this efficiently requires updating our posterior distributions at the previous time step to take account of the new observation. Note that on-line algorithms can be used to analyse batch data by introducing an artificial time for each observation.

We focus on on-line inference of the position of the changepoints. Under the modelling assumptions of Section 2, inference for the parameters conditional on knowing the number and position of the changepoints is straightforward. We first describe an exact on-line algorithm, which is an on-line version of the direct simulation method of Fearnhead (2005). The computational cost of this exact algorithm increases over time, so we then present an approximate on-line algorithm which uses resampling ideas from particle filters, and which has constant computational cost over time.

3.1. Exact on-line inference

We introduce a state at time t, C_t, which is defined to be the time of the most recent changepoint before t (with $C_t=0$ if there have been no changepoints before time t). Initially we focus on calculating the posterior distribution for C_t given the observation $\mathbf{y}_{1:t}$. We then describe how, if these distributions are stored for all t, it is straightforward to simulate from the joint posterior distribution of the position of all changepoints before the current time.

3.1.1. *Filtering recursions*

The state C_t can take values in $0, 1, \ldots, t-1$, and $C_1, C_2, \ldots, C_t, \ldots$ is a Markov chain. Conditional on $C_t=j$, either $C_{t+1}=j$, which corresponds to no changepoint at time t, or $C_{t+1}=t$, if there is a changepoint at time t. The transition probabilities for this Markov chain can thus be calculated as

$$\Pr(C_{t+1}=j|C_t=i)=\begin{cases} \dfrac{1-G(t-i)}{1-G(t-i-1)} & \text{if } j=i,\\ \dfrac{G(t-i)-G(t-i-1)}{1-G(t-i-1)} & \text{if } j=t,\\ 0 & \text{otherwise,}\end{cases} \tag{5}$$

where $G(\cdot)$ is the distribution function of distance between two successive changepoints.

Now, standard filtering recursions give

$$\Pr(C_{t+1}=j|\mathbf{y}_{1:t+1}) \propto \Pr(y_{t+1}|C_{t+1}=j, \mathbf{y}_{1:t}) \Pr(C_{t+1}=j|\mathbf{y}_{1:t})$$

and

$$\Pr(C_{t+1}=j|\mathbf{y}_{1:t})=\sum_{i=0}^{t-1} \Pr(C_{t+1}=j|C_t=i) \Pr(C_t=i|\mathbf{y}_{1:t}).$$

Thus, if we let $w_{t+1}^{(j)}=\Pr(y_{t+1}|C_{t+1}=j, \mathbf{y}_{1:t})$, and substitute in the transition probabilities (5), we obtain

$$\Pr(C_{t+1}=j|\mathbf{y}_{1:t+1}) \propto \begin{cases} w_{t+1}^{(j)} \dfrac{1-G(t-j)}{1-G(t-j-1)} \Pr(C_t=j|\mathbf{y}_{1:t}) & \text{if } j<t,\\ w_{t+1}^{(t)} \displaystyle\sum_{i=0}^{t-1} \dfrac{G(t-i)-G(t-i-1)}{1-G(t-i-1)} \Pr(C_t=i|\mathbf{y}_{1:t}) & \text{if } j=t.\end{cases}$$

If we define $P(s,t,q)$ as in equation (2), and we denote by $p(q)$ the prior probability of model q, then we obtain for $j<t$

$$w_{t+1}^{(j)} = \frac{\sum_{q=1}^{\bar{p}} P(j,t+1,q)\; p(q)}{\sum_{q=1}^{\bar{p}} P(j,t,q)\; p(q)}, \tag{6}$$

whereas if $j=t$ then the weight is $\Sigma_{q=1}^{\bar{p}} P(t,t+1,q)\; p(q)$.

In most situations, such as for the linear regression models that were described in Section 2, the incremental weights $w_{t+1}^{(j)}$ can be calculated efficiently, as each $P(j,t,q)$ depends on a set of summary statistics of the observations $y_{j+1:t}$, which can be updated recursively. Efficient calculation of the incremental weights $w_{t+1}^{(j)}$ is possible by storing these summary statistics for each set of values for the time of the last changepoint, j, and the model for the current segment, q. These can be updated to give the summary statistics that are required for each $P(j,t+1,q)$. Thus the computational cost of calculating each $w_{t+1}^{(j)}$ is fixed and does not increase with $t-j$.

3.1.2. *Simulating changepoints*

If we store the filtering densities $\Pr(C_t|\mathbf{y}_{1:t})$ for all $t=1,\ldots,n$, then it is straightforward to simulate from the joint posterior distribution of the position of all changepoints before time n, using the idea of Chopin (2006). To simulate one realization from this joint density, perform the following algorithm.

Step 1: simulate t_1 from $\Pr(C_n|\mathbf{y}_{1:n})$, set $k=1$ and go to step 3.
Step 2: simulate t_{k+1} from the distribution proportional to $\Pr(C_{t_k}|\mathbf{y}_{1:t_k})\Pr(C_{t_k+1}=t_k|C_{t_k})$, and set $k=k+1$.
Step 3: if $t_k>0$ return to step 2; otherwise output the set of simulated changepoints, t_{k-1}, $t_{k-2},\ldots,t_1$.

A simple extension of this algorithm which enables a large sample of realizations of sets of changepoints to be simulated efficiently is described in Fearnhead (2006).

3.1.3. *Maximum a posteriori estimation*

We can obtain an on-line Viterbi algorithm for calculating the maximum *a posteriori* (MAP) estimate of the positions of the changepoints and the model orders for each segment as follows. We define $\mathcal{M}_j$ to be the event that, given a changepoint at time j, the MAP choice of changepoints and model occurs before time j. For $t=1,\ldots,n$, $j=0,\ldots,t-1$ and $q=1,\ldots,\bar{p}$,

$$P_t(j,q)=\Pr(C_t=j, \text{model } q, \mathcal{M}_j, \mathbf{y}_{1:t})$$

and

$$P_t^{\text{MAP}}=\Pr(\text{changepoint at } t, \mathcal{M}_t, \mathbf{y}_{1:t}).$$

We obtain the equations

$$P_t(j,q)=\{1-G(t-j-1)\}P(j,t,q)\; p(q)P_j^{\text{MAP}}$$

and

$$P_t^{\text{MAP}}=\max_{j,q}\left\{\frac{P_t(j,q)\; g(t-j)}{1-G(t-j-1)}\right\}. \tag{7}$$

At time t, the MAP estimates of C_t and the current model order are given respectively by the values of j and q which maximize $P_t(j,q)$. Given an MAP estimate of C_t, $\hat{c}_t$, we can then calculate the MAP estimates of the changepoint before $\hat{c}_t$ and the model order of that segment by the values of j and q that maximized the right-hand side of equation (7). This procedure can be repeated to find the MAP estimates of all changepoint positions and model orders.

3.2. Approximate inference

The computational and memory costs of the recursions for exact inference that were presented in Section 3.1 both increase with time. The computational cost of both the filtering recursion and the MAP recursion at time t is proportional to t, the number of possible values of C_t, whereas the memory cost of storing all filtering densities up to time t, which is necessary to simulate from the joint posterior of all changepoints before t, increases quadratically with t. For large data sets, these computational and memory costs may become prohibitive.

A similar problem of increasing computational cost occurs in the analysis of some hidden Markov models—though generally computational cost increases exponentially with time (Chen and Liu, 2000). Particle filters have been successfully applied to these problems (Fearnhead and Clifford, 2003) by using a resampling step to limit the computational cost at each time step. Here we show how similar resampling ideas can be applied to the on-line inference of the changepoint models that we are considering. We present a variation on the optimal resampling (OR) method of Fearnhead and Clifford (2003) which is specifically designed for changepoint models, and we show theoretically why this is an optimal resampling algorithm in this case. We also present an extension of the RC approach of Liu *et al.* (1998) which is suitable for the analysis of batch data, and for which it is possible to control the amount of error that is introduced at each resampling step.

3.2.1. *Controlling computational cost*

Our first approach is to control the average (and maximum) computational cost for analysing each new observation. At time t our exact algorithm stores the complete posterior distribution of the time of the last changepoint $\Pr(C_t = c_t | \mathbf{y}_{1:t})$, for $c_t = 0, 1, \ldots, t-1$. We can approximate this by a discrete distribution with fewer, N, support points. This approximate distribution can be described by the set of support points, $c^{(1)}, \ldots, c^{(N)}$, which are henceforth called *particles*, and the probability mass that is associated with each of these particles, $w^{(1)}, \ldots, w^{(N)}$, which we call weights. (The particles and their weights will depend on t; we have suppressed this dependence to simplify the notation.)

We impose a maximum number of particles to be stored at any one time, N, such that whenever we have N particles we immediately perform resampling to reduce the number of particles to $M < N$. The average computational cost per iteration will thus be proportional to $(M+N+1)/2$, and the maximum computational cost per iteration will be proportional to N.

Assume that at the current time point we have N particles and wish to reduce these to M particles. We propose the following stratified version of the OR algorithm of Fearnhead and Clifford (2003), which we call stratified optimal resampling (SOR).

Initialization: assume that we currently have a set of ordered particles $c^{(1)} < c^{(2)} < \ldots < c^{(N)}$, with associated weights $w^{(1)}, \ldots, w^{(N)}$, which sum to 1.

SOR step 1: calculate α, the unique solution to $\Sigma_{i=1}^{N} \min(1, w^{(i)}/\alpha) = M$.

SOR step 2: for $i = 1, \ldots, N$ if $w^{(i)} \geqslant \alpha$ then keep particle $c^{(i)}$ with weight $w^{(i)}$. Assume that A particles are kept.

SOR step 3: use the stratified resampling algorithm of Carpenter *et al.* (1999) to resample

$M-A$ times from the ordered set of the remaining $N-A$ particles (without shuffling). Each resampled particle is assigned a weight α.

The stratified resampling algorithm that is used in SOR step 3 is given in Appendix A. The use of stratified resampling in SOR step 3 means that at most one copy of each particle is kept, as the expected number of times a particle with weight w is resampled is $w/\alpha < 1$ (note that there is no advantage in having multiple copies of particles; see Fearnhead and Clifford (2003)). The only difference between this SOR algorithm and the original algorithm of Fearnhead and Clifford (2003) is that particles are ordered before resampling in SOR step 3.

As shown in Fearnhead and Clifford (2003), if we denote by $W^{(i)}$ the (random) weight of a particle after resampling (so $W^{(i)}$ equals $w^{(i)}$, α or 0 depending on whether the respective particle is kept, resampled or not resampled), then SOR is optimal over all resampling algorithms that satisfy $E(W^{(i)}) = w^{(i)}$ in terms of minimizing the mean-square error: $E\{\Sigma_{i=1}^{N}(W^{(i)} - w^{(i)})^2\}$. By ordering the particles in SOR step 3 we obtain the following further property.

Theorem 1. Consider a set of N particles, $c^{(1)} < c^{(2)} < \ldots < c^{(N)}$ with weights $w^{(1)}, \ldots, w^{(N)}$. Let $W^{(i)}$ be the (random) weight of particle $c^{(i)}$ after resampling. Define the maximum Kolmogorov–Smirnov distance for a resampling algorithm as

$$\mathrm{mKSD} = \max\left\{ \max_i \left| \sum_{j=1}^{i} (w^{(j)} - W^{(j)}) \right| \right\} \qquad (8)$$

where the first maximization is over realizations of $W^{(1)}, \ldots, W^{(N)}$ with positive probability. Then the SOR algorithm above satisfies $\mathrm{mKSD} \leqslant \alpha$ (where α is defined as in SOR step 1). Furthermore

(a) for a resampling algorithm to have $\mathrm{mKSD} \leqslant \alpha$ then all particles with $w^{(i)} > \alpha$ must be propagated without resampling and
(b) the mKSD for the SOR algorithm above is less than or equal to the mKSD of the OR algorithm of Fearnhead and Clifford (2003), and the RC algorithm of Liu *et al.* (1998).

For a proof of theorem 1, see Appendix B.

Kolmogorov–Smirnov distance is a natural metric for the distributions of one-dimensional random variables. By using equation (8) as a measure of error of a resampling algorithm, we are considering the bound on Kolmogorov–Smirnov distance that a resampling algorithm can introduce. The theorem gives a simple interpretation of the α that is calculated in SOR step 1, in terms of an upper bound on the Kolmogorov–Smirnov distance between the original and resampled weights.

We define a resampling algorithm to be unbiased if $E(W^{(i)}) = w^{(i)}$ for all i. (This is related to the properly weighted condition of Liu *et al.* (2001).) The OR algorithm of Fearnhead and Clifford (2003) and RC are currently the only other unbiased resampling algorithms which satisfy condition (a) of theorem 1. (Note that RC will not produce a fixed number of particles after resampling, though implementing RC with a threshold of α will produce on average N resampling particles, and further that, although $E(W^{(i)}) = w^{(i)}$, the resampled weights do not necessarily sum to 1.) So results (a) and (b) of theorem 1 show that our SOR algorithm is optimal over all existing unbiased resampling algorithms in terms of minimizing mKSD.

We have presented the SOR algorithm in terms of the general case of resampling M particles from N current particles. For on-line inference, where there is a fixed amount of time to analyse each observation, it is natural to choose N to be the largest number of particles that enable an observation to be analysed in less than this amount of time, and to set $M = N-1$. In this case there is no difference between SOR and the existing OR algorithm. In practice, it may be better to choose $N - M > 1$ (see Section 4) as this enables the particles to be removed to be jointly chosen in a stratified manner.

3.2.2. Controlling resampling error

An alternative to basing resampling on the average and maximum number of particles to be kept at each time step is to choose the amount of resampling to control the size of error that is introduced at each time step. Such an approach is most suitable for using on-line algorithms to analyse batch data. For realtime data, the frequency of observations will place an upper bound on the central processor unit time, and hence the number of particles, that can be used to process each observation. By controlling the resampling error, rather than the number of particles, we cannot ensure that the number of particles always stays below this error.

The idea behind controlling the resampling error is given by the interpretation of α for SOR that comes from theorem 1. The value of α defines the maximum error (as defined by Kolmogorov–Smirnov distance) that is introduced by the resampling error. So rather than specifying the number of resampled particles which in turn defines α, and hence the amount of error that we introduce, we can instead specify α, which will then define the number of resampled particles.

Our method for controlling the resampling error is to use a stratified version of RC (Liu *et al.*, 1998), rather than adapt the SOR algorithm. For a prespecified value of α, our stratified rejection control (SRC) algorithm is as follows.

Initialization: assume that we currently have a set of ordered particles $c^{(1)} < c^{(2)} < \ldots < c^{(N)}$, with associated weights $w^{(1)}, \ldots, w^{(N)}$, which sum to 1.
SRC step 1: for $i = 1, \ldots, N$ if $w^{(i)} \geqslant \alpha$ then keep particle $c^{(i)}$ with weight $w^{(i)}$. Assume that A particles are kept.
SRC step 2: use the stratified resampling algorithm of Carpenter *et al.* (1999) to resample from the ordered set of the remaining $N - A$ particles (without shuffling). The expected number of times that particle $c^{(i)}$ is resampled is $w^{(i)}/\alpha$. Each resampled particle is assigned a weight α.

Again the use of stratified resampling in SRC step 2 means that at most one copy of each particle is kept. Note that the sum of the particles' weights after resampling will not necessarily be 1 (though they lie between $1 - \alpha$ and $1 + \alpha$) and should be normalized to produce a probability distribution.

The difference between SRC and RC (Liu *et al.*, 1998) is that particles are ordered and stratified resampling is used in SRC step 2, as opposed to independent resampling of each particle. The use of stratified resampling means that the maximum error of the unnormalized weights that is introduced by SRC, as measured by Kolmogorov–Smirnov distance, is α (this can be proved in an identical manner to theorem 1). Furthermore, the error of the normalized weights can be shown to be bounded above by $\alpha/(1 - \alpha) = \alpha + o(\alpha)$ (see Appendix C).

4. Numerical examples

We tested our algorithm on three simulated examples: the 'Blocks' and 'HeaviSine' examples from Donoho and Johnstone (1994) and a piecewise autoregressive model. Each of the three data sets are analysed under a piecewise regression model. The design matrices for the piecewise autoregressive model and the piecewise polynomial regression model (for the Blocks and HeaviSine data) are

$$\mathbf{H}_{s:t} = \begin{pmatrix} y_{s-1} & y_{s-2} & y_{s-3} \\ y_s & y_{s-1} & y_{s-2} \\ \vdots & \vdots & \vdots \\ y_t & y_{t-1} & y_{t-2} \end{pmatrix}, \qquad \mathbf{H}_{s:t} = \begin{pmatrix} 1 & x_s & x_s^2 \\ 1 & x_{s+1} & x_{s+1}^2 \\ \vdots & \vdots & \vdots \\ 1 & x_t & x_t^2 \end{pmatrix}$$

respectively, where $x_s = s/n$ and n is the number of data points. In each case we perform model choice within each segment, choosing between model orders 1, 2 and 3. We allow for different variances of the measurement error within each segment, although for the Blocks and HeaviSine examples we simulated data with a common error variance across segments. Further details of the model, and calculations required for calculating the $P(s,t,q)$s, are given in Section 2 (see also Punskaya *et al.* (2002) and Fearnhead (2005)).

Our focus here is on the performance of different possible resampling algorithms. The Blocks data set (Fig. 1) is a particularly simple data set to analyse, and all reasonable resampling algorithms will give almost identical results. We show the results here to demonstrate how the SRC algorithm naturally adapts the number of particles that are kept. The Blocks data set has a number of obvious changepoints, and when each of these is encountered the number of particles that are needed to be kept is reduced to close to 1.

For the HeaviSine example (see Fig. 1) we compared the accuracy of various resampling algorithms: SRC, RC, SOR and OR. We considered two values of α for SRC and RC, and, for a meaningful comparison, fixed the mean number of particles in OR and SOR to the mean number of particles kept by SRC for each of these two values. If we set the number of resampled particles (M) to be one less than the number of particles before resampling (N), then OR and SOR are identical. We tested both $N = M + 1$ and $N = M + 5$.

Our comparison is based on the Kolmogorov–Smirnov distance KSD between the true filtering distribution of the most recent changepoint, $p(C_t|\mathbf{y}_{1:t})$ (calculated by using the on-line algorithm with no resampling), and its approximation based on the various resampling algorithms, for each t. Results are given in Table 1. The results show that the mean KSD error is reduced by a third by using SRC rather than RC. Both of these methods perform better than the resampling algorithms that use a fixed number of particles (for the same average number of particles), showing the advantage of allowing the number of particles that are used to adapt to

Table 1. Mean Kolmogorov–Smirnov distance in $P(C_t|\mathbf{y}_{1:t})$ averaged over t for the HeaviSine and autoregressive models and the four resampling algorithms†

Model	*KSD for the following methods:*			
	SRC	*RC*	*SOR*	*OR*
HeaviSine	1.3×10^{-2}	2.0×10^{-2}	4.2×10^{-2}	6.4×10^{-2}
Autoregressive	1.3×10^{-6}	2.2×10^{-6}	2.2×10^{-4}	3.5×10^{-4}

†SRC and RC were implemented with $\alpha = 10^{-6}$; these algorithms used an average number of 43 and 70 particles for the HeaviSine and autoregressive models respectively. OR was implemented with $N = M + 1 = 49$ and $N = M + 1 = 90$; SOR used $N = M + 5 = 51$ and $N = M + 5 = 92$ (chosen so that the average number of particles is the same for all algorithms for each data set). The results are based on 50 replications of each algorithm for one version of each data set. The true distribution $P(C_t|\mathbf{y}_{1:t})$ was calculated by using the exact on-line algorithm.

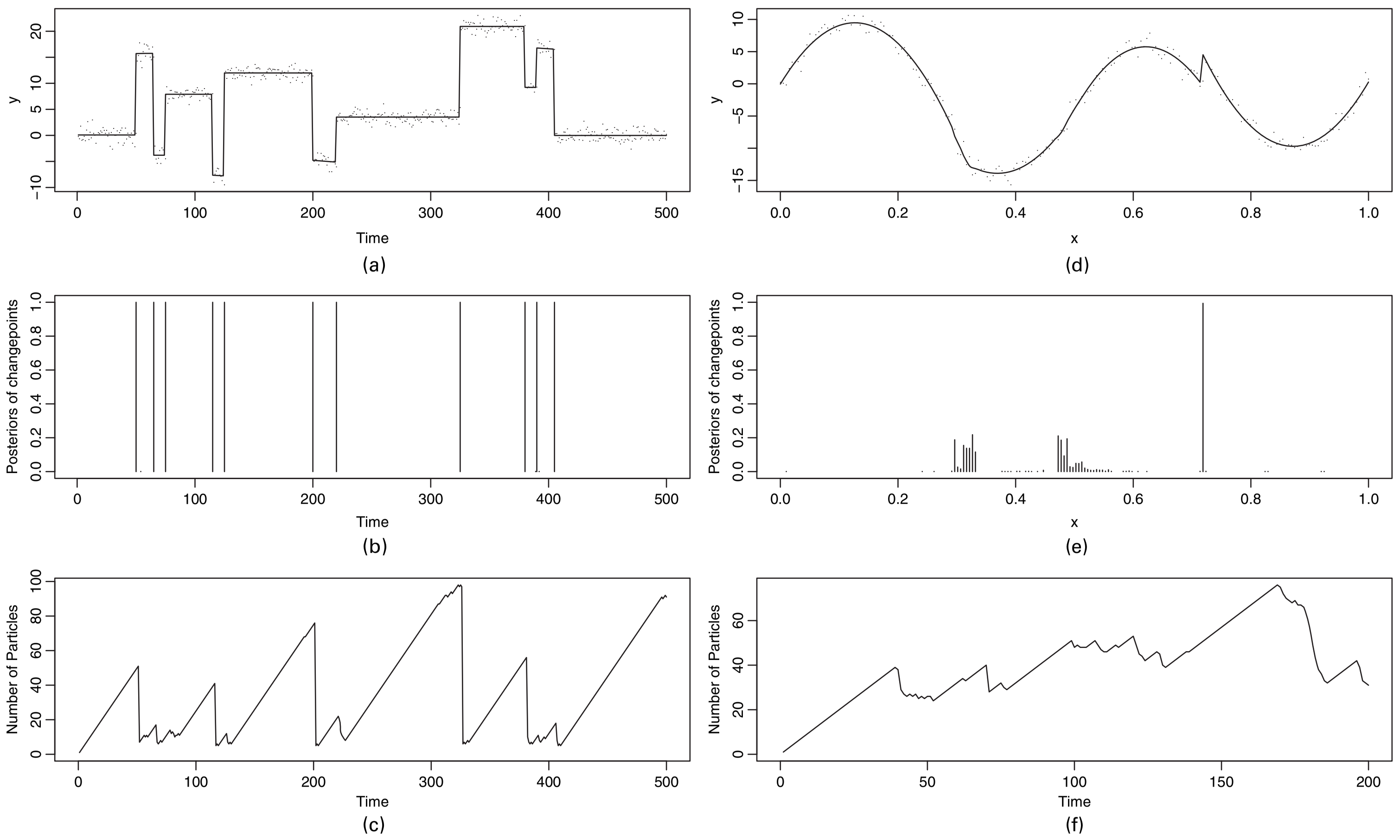

Fig. 1. (a)–(c) Blocks data set and (d)–(f) HeaviSine data set together with the results of analysis by the SRC algorithm with $\alpha = 10^{-6}$: (a), (d) data and inferred signal; (b), (e) marginal probability of changepoints; (c), (f) number of particles kept

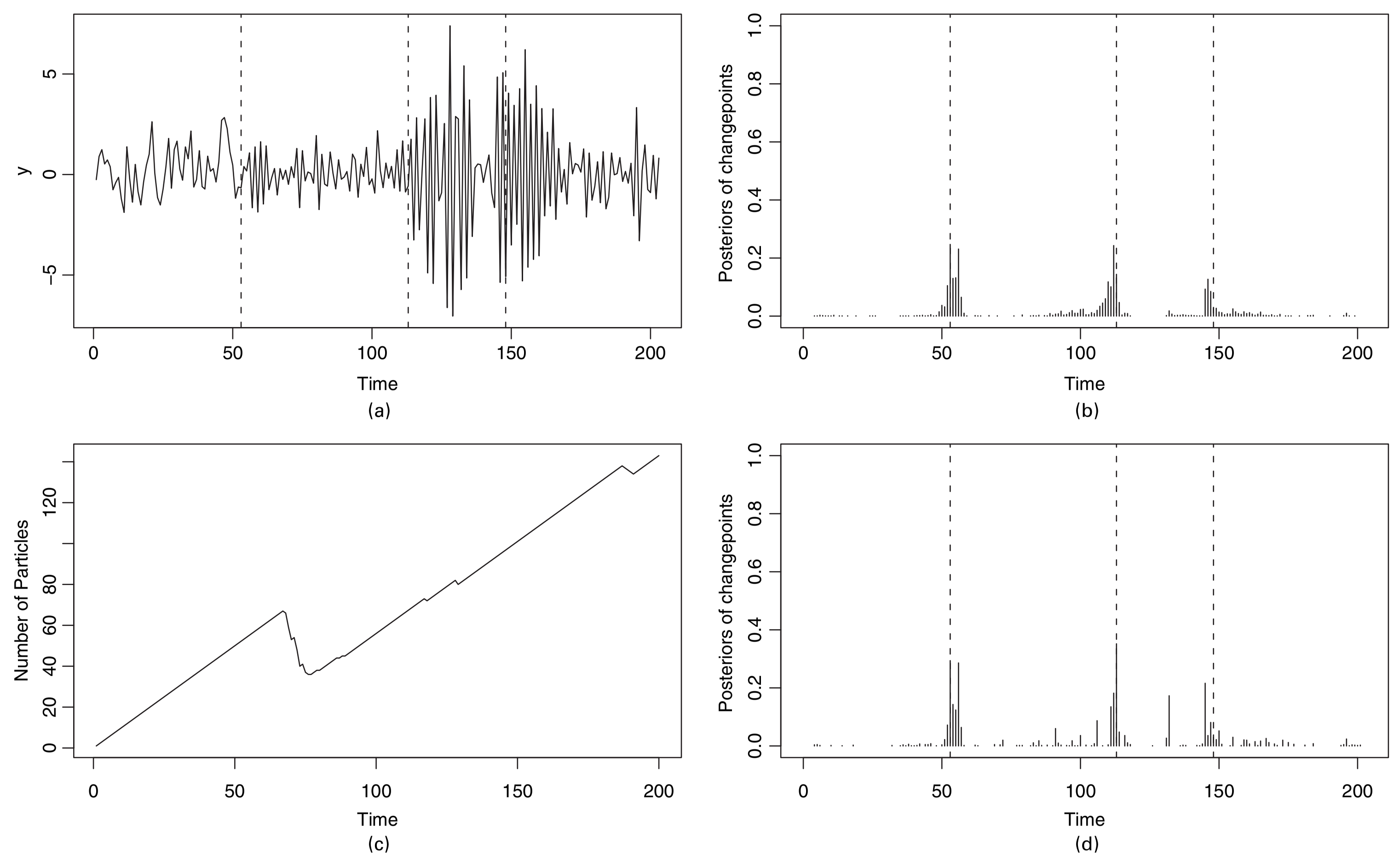

Fig. 2. Results of analysing the autoregressive data set by using SRC with $\alpha = 10^{-6}$, and the particle filter of Chopin (2006) with 50 000 particles (the true autoregressive models for the four segments have model orders 1, 1, 2 and 3; the corresponding parameters are $\beta = 0.4$, $\beta = -0.6$, $\beta = (-1.3, -0.36, 0.25)$ and $\beta = (-1.1, -0.24)$ with error variances 1.2^2, 0.7^2, 1.3^2 and 0.9^2 respectively): (a) data; (b) marginal probabilities of a changepoint for SRC; (c) number of particles kept by using SRC; (d) marginal probabilities of a changepoint for the particle filter of Chopin (2006)

the filtering density being approximated. Of the two algorithms considered which use a fixed number of particles, we see an improvement of using SOR where we remove five particles at each resampling step over OR (or equivalently SOR) where one particle is removed at each resampling step. By removing many particles in one step, SOR can jointly choose the particles to remove in a stratified way to reduce the error that is introduced. (Note that OR where we remove five particles at each resampling step has worse results than the OR results that are shown in Table 1.)

Although all resampling algorithms introduce small errors at each resampling step, it is possible for these errors to accumulate. The reason for this appears to be that the evidence for a changepoint at a given time t can change substantially as more data are collected. If the evidence is small (and hence the filtering probability of a changepoint at t is less than α) at a resampling step, this can lead to the corresponding particle being removed. Such a particle cannot be 'resurrected' as future observations are made, even if they carry strong evidence for a changepoint at t. However, stratified resampling should ensure that a particle corresponding to a changepoint that is close to t is kept, and thus the error in estimating the position of the changepoints will still be small.

We repeated this analysis for a piecewise autoregressive model. The results of the SRC analysis with $\alpha = 1 \times 10^{-6}$ are given in Fig. 2, and results of the accuracy of each resampling method are given in Table 1. We observe similar results to the HeaviSine example in terms of the relative performance of the resampling algorithms. In this case SRC again outperforms RC by about a third. The difference in performance between SRC and RC compared with SOR and OR is quite substantial in this case, because towards the end of the time series it is forced to use too few particles to approximate the filtering densities adequately. This again demonstrates the potential gains to be obtained by allowing the number of particles that are used to change over time and to adapt to the filtering distribution that is being approximated.

We also ran the particle filter of Chopin (2007). This filter does not integrate out the parameters that are associated with each segment, so each particle consists of a time for the last changepoint together with a value of the parameters for the current segment. The filter uses MCMC sampling to update the parameters of a subset of particles at each iteration. We ran the filter with 50000 particles, using a Gibbs sampler update on the parameters of a third of the particles at each iteration. This took over an order of magnitude longer to run than the SRC algorithm and even is substantially more time consuming to implement than the exact on-line algorithm.

The results for the estimate of the marginal probabilities of the changepoints is shown in Fig. 2. The filter of Chopin (2007) suffers from a loss of diversity in the particles—with many positions being assigned zero probability of being a changepoint, when in fact there is a non-negligible probability as can be seen from the output of the SRC filter. To give a quantitative comparison of the two methods we calculated the mean absolute error between the estimates of the marginal probabilities of the changepoints that are shown in Fig. 2 with those based on the exact particle filter algorithm. These were 0.010 and 0.002 for the filter of Chopin (2007) and the SRC filter respectively.

5. DNA segmentation

In recent years there has been an explosion in the amount of data describing the genetic make-up of different organisms; for example the complete DNA sequence of one human genome is now known as a result of the human genome project. There is interest in learning about the genomic features of different organisms, and learning how these features may have evolved and how they correlate with each other.

We consider the problem of understanding the structure of C+G content within the genome. A common model for the C+G content of the human genome is that there are large, of the order of 300 kilobases, regions of roughly homogeneous C+G content, called isochores (see Bernardi (2000) for background). Furthermore C+G content is known to correlate with various features of the genome, such as high recombination rates and gene density (Hardison *et al.*, 2003).

Currently, the most common method for segmenting an organism's genome into regions of different C+G content is implemented in the computer program `IsoFinder` (Oliver *et al.*, 2004). This is based on a recursive segmentation procedure, which initially classifies a large genomic region as consisting of a single isochore (region of common C+G content). It then considers in turn each possible position for adding a changepoint, and splitting the data into two isochores. For each possible position, a t-statistic is calculated for testing whether the mean C+G content is different in the two putative isochores. For each changepoint, a p-value is calculated for its value of the t-statistic by using a bootstrap procedure and, if the smallest p-value is less than some predefined threshold, then the corresponding changepoint is added. This procedure is repeated, with at each step each current isochore being tested for whether it can be split into two isochores. See Oliver *et al.* (2004) for more details.

We consider a Bayesian approach to segmenting a genomic region into isochores. The potential advantages of a Bayesian approach include

(a) quantifying and averaging over the uncertainty in the number and positions of the isochores,
(b) jointly estimating all isochore positions (which Braun *et al.* (2000) showed to be more accurate than segmentation procedures) and
(c) that the large amount of data that are available for each organism makes it straightforward to construct sensible prior distributions.

One of the computational challenges of such an analysis is the large amount of data that need to be analysed (for example human chromosomes consist of around 100 million bases). We simplify this burden by first summarizing our data by the number of DNA sites which are C or G within consecutive windows (each window being of the order of a few kilobases in width), an approach which also has the advantage of averaging out the very local high variation in C+G content caused for example by CpG islands and Alu elements. We then hope that our on-line changepoint algorithm will be able to analyse the resulting data efficiently, and one of the main aims of the study that we present here is to test whether such an approach is computationally practicable for analysing the large amount of genomic data that are currently available.

The model that we use is based on the following simple model for the data $\mathbf{y}_{1:n}$, which is similar to the implicit model that is assumed by `IsoFinder`. A data set is shown in Fig. 3. The tth data point y_t represents the number of DNA bases which are either C or G within the tth window. If this window lies within the ith isochore then we assume that

$$y_t = \mu_i + \sigma_i \varepsilon_t,$$

where μ_i is the mean C+G content of each window within the ith isochore, σ_i^2 is the error variance within the ith isochore and ε_t is some independent error. We assume that ε_t has a Gaussian distribution and we assume standard conjugate priors (see Section 2) for the μ_is and σ_is, with the prior parameters chosen from an initial analysis of C+G data with a moving median filter. We assumed a geometric distribution for the length of each isochore.

Results of our analysis using SRC with $\alpha = 10^{-6}$ are shown in Fig. 3. Our main focus is on the computational practicability of a Bayesian analysis of such data, and our method took 6 s on a desktop computer to analyse this data set.

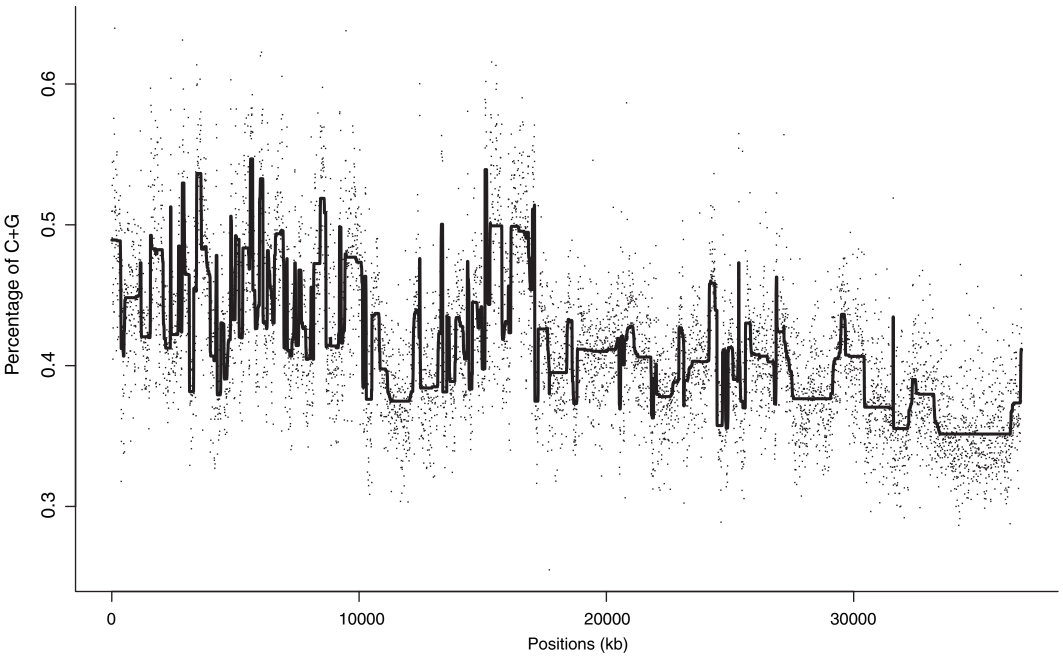

Fig. 3. Analysis of 35 megabases of data from human chromosome 1: ———, posterior mean G+C content

This application does not need to be analysed by an on-line algorithm, such as the one that we used. However, Fearnhead (2006) showed that the version of our algorithm without resampling can be more efficient for analysing changepoint models than some commonly used MCMC algorithms. Furthermore, by using resampling we have been able to reduce vastly the computational and storage cost of analysing the data. For example our implementation with resampling uses an average of 117 particles at each time step, whereas without resampling the algorithm would require an average of over 3500 particles for each time step.

6. Discussions

We have considered a class of changepoint models, which have a specific conditional independence structure (see Section 2), and have shown how the direct simulation algorithm of Fearnhead (2005) can be implemented on line. Such an algorithm can be viewed as an exact particle filter, and resampling ideas taken from particle filters can be used to reduce the computational complexity of the direct simulation algorithm (at the cost of introducing error). We have presented two simple extensions of existing resampling algorithms, which are particularly well suited to changepoint problems (or any problems where the underlying state of interest is one dimensional).

In simulation studies, our new resampling algorithms decreased the error of the resulting particle filter by up to a third, compared with particle filters using the existing resampling approaches. We have shown that the new resampling algorithms satisfy a minimax optimality criterion on the error, as measured by Kolmogorov–Smirnov distance, that is introduced by resampling. Furthermore this result gives a natural interpretation of the threshold that needs to be specified in the SRC algorithm which will aid its implementation in practice.

There is great flexibility with implementing resampling algorithms within particle filters which we have not explored. For example Liu and Chen (1998) discussed the frequency with which resampling should occur, and Liu *et al.* (1998) suggested the use of RC only when the variance of the particle filter weights exceeds some threshold. Although we have not fully investigated these issues, the results from Section 4 suggest that the advantages of using stratification within OR or RC will increase as the frequency of resampling decreases (or equivalently the amount of particles resampled increases at each resampling step).

Acknowledgements

This work is supported by Engineering and Physical Sciences Research Council grant C531558. We dedicate this paper to the memory of Nick Smith who helped with the application to detecting isochores.

Appendix A: Stratified resampling algorithm

We describe the stratified resampling algorithm of Carpenter *et al.* (1999) in terms of the SOR and SRC algorithms. Assume that we currently have a set of N ordered particles $c^{(1)} < c^{(2)} < \ldots < c^{(N)}$, with associated weights $w^{(1)}, \ldots, w^{(N)}$, which sum to 1. For the SOR algorithm define α as in SOR step 1; and for SRC we assume that the value of α is given. Resampling of M particles proceeds as follows.

(a) Simulate u a realization of a uniform random variable on $[0, \alpha]$. Set $i = 1$.
(b) If $w^{(i)} \geqslant \alpha$ then propagate particle $c^{(i)}$ with weight $w^{(i)}$; otherwise let $u = u - w^{(i)}$; if $u \leqslant 0$ then resample particle $c^{(i)}$ and assign a weight α, and set $u = u + \alpha$.
(c) Let $i = i + 1$; if $i \leqslant N$ then return to step (b).

Appendix B: Proof of theorem 1

Theorem 1 considers the error of a resampling algorithm as measured by

$$\text{mKSD} = \max\left\{ \max_i \left| \sum_{j=1}^{i} w^{(j)} - W^{(j)} \right| \right\}.$$

For SOR, if $w^{(i)} \geqslant \alpha$ then $W^{(i)} = w^{(i)}$ with probability 1. As such we can consider mKSD solely for the subset of particles which have $w^{(i)} < \alpha$. Assume that we have N' such particles, and relabel these particles $c^{(1)} < c^{(2)} < \ldots < c^{(N')}$.

The only randomness in the SOR algorithm is the simulation of u in step (a) of the algorithm that is detailed in Appendix A. Now for a given value of u

$$\sum_{j=1}^{i} W^{(j)} = \alpha \left[\left(\sum_{j=1}^{i} w^{(j)} + \alpha - u \right) \Big/ \alpha \right], \tag{9}$$

where $[x]$ is the integer part of x. Thus for all u and i

$$\left| \sum_{j=1}^{i} w^{(j)} - W^{(j)} \right| \leqslant \alpha,$$

so $\text{mKSD} \leqslant \alpha$.

For result (a) of theorem 1 it suffices to note that, if the probability of resampling particle $c^{(i)}$ is strictly less than 1, then $\text{mKSD} \geqslant w^{(i)}$.

For result (b) it is sufficient to note that both the OR algorithm of Fearnhead and Clifford (2003) (where particles are shuffled before stratified resampling) and RC (where each particle with weight less than α is resampled independently of all others) give positive probability to all realizations of weights $W^{(1)}, W^{(2)}, \ldots, W^{(N)}$ that our SOR algorithm does. It trivially follows that mKSD for these algorithms will be greater than that of our SOR algorithm.

Appendix C: Error bound for stratified rejection control

Consider N particles, ordered so that $c^{(1)} < c^{(2)} < \ldots < c^{(N)}$. We denote the weight of these particles before resampling by $w^{(i)}$, the unnormalized weights after resampling by $W^{(i)}$ and the normalized weights after resampling by $\bar{W}^{(i)}$. We let u denote the realization of the uniform $[0, \alpha]$ random variable that is used in the stratified resampling algorithm. Finally we let

$$\varepsilon^{(i)} = \sum_{j=1}^{i} (w^{(j)} - W^{(j)}).$$

The sum of the resampling weights depends on the number of particles that are resampled in SRC stage 2. There is a constant β satisfying $0 \leqslant \beta < \alpha$ such that

$$\sum_{i=1}^{N} W^{(i)} = \begin{cases} 1+\alpha-\beta & u \leqslant \beta, \\ 1-\beta & u > \beta. \end{cases}$$

Fix u and β. From equation (9) it can be shown that $u - \alpha \leqslant \varepsilon^{(i)} \leqslant u$ for all i. We consider in turn the situation $u \leqslant \beta$ and $u > \beta$, corresponding to the two possible values of the sums of the unnormalized weights after resampling.

Firstly, assume that $u \leqslant \beta$. Then we have

$$\begin{aligned} \left| \sum_{j=1}^{i} (w^{(j)} - \bar{W}^{(j)}) \right| &= \left| \sum_{j=1}^{i} w^{(j)} - \frac{W^{(j)}}{1-\beta+\alpha} \right| \\ &= \frac{1}{1+\alpha-\beta} \left| \varepsilon^{(i)} + (\alpha-\beta) \sum_{j=1}^{i} w^{(j)} \right| \\ &\leqslant \frac{1}{1+\alpha-\beta} \max\left\{ u + (\alpha-\beta) \sum_{j=1}^{i} w^{(j)}, \alpha - u - (\alpha-\beta) \sum_{j=1}^{i} w^{(j)} \right\}, \end{aligned}$$

where the two terms that we are maximizing over correspond to the largest positive and negative values of $\varepsilon^{(i)}$. Now, as $u \leqslant \beta$ and $0 < \Sigma_{j=1}^{i} w^{(j)} < 1$, both these terms are bounded above by α. Thus we have mKSD $< \alpha$ in this case.

Now, if $u > \beta$, by a similar argument we obtain

$$\left| \sum_{j=1}^{i} (w^{(j)} - \bar{W}^{(j)}) \right| \leqslant \frac{1}{1-\beta} \max\left\{ u - \beta \sum_{j=1}^{i} w^{(j)}, \alpha - u + \beta \sum_{j=1}^{i} w^{(j)} \right\} \leqslant \frac{\alpha}{1-\beta}.$$

The last inequality uses the fact that $\beta < u \leqslant \alpha$ and $0 < \Sigma_{j=1}^{i} w^{(j)} < 1$. Finally as $\beta < \alpha$ we can obtain that mKSD $< \alpha/(1-\alpha)$.

References

Barry, D. and Hartigan, J. A. (1992) Product partition models for change point problems. *Ann. Statist.*, **20**, 260–279.

Bernardi, G. (2000) Isochores and evolutionary genomics of vertebrates. *Gene*, **241**, 3–17.

Braun, J. V., Braun, R. K. and Muller, H. G. (2000) Multiple changepoint fitting via quasilikelihood, with application to DNA sequence segmentation. *Biometrika*, **87**, 301–314.

Braun, J. V. and Muller, H. G. (1998) Statistical methods for DNA sequence segmentation. *Statist. Sci.*, **13**, 142–162.

Carpenter, J., Clifford, P. and Fearnhead, P. (1999) An improved particle filter for non-linear problems. *IEE Proc. Radar Sonar Navign*, **146**, 2–7.

Chen, R. and Liu, J. S. (2000) Mixture Kalman filters. *J. R. Statist. Soc.* B, **62**, 493–508.

Chib, S. (1998) Estimation and comparison of multiple change-point models. *J. Econometr.*, **86**, 221–241.

Chopin, N. (2002) A sequential particle filter method for static models. *Biometrika*, **89**, 539–551.

Chopin, N. (2007) Dynamic detection of change points in long time series. *Ann. Inst. Statist. Math.*, **59**, 349–366.

Del Moral, P., Doucet, A. and Jasra, A. (2006) Sequential Monte Carlo samplers. *J. R. Statist. Soc.* B, **68**, 411–436.

Donoho, D. L. and Johnstone, I. M. (1994) Ideal spatial adaptation by wavelet shrinkage. *Biometrika*, **81**, 425–455.

Fearnhead, P. (2005) Exact Bayesian curve fitting and signal segmentation. *IEEE Trans. Signal Process.*, **53**, 2160–2166.

Fearnhead, P. (2006) Exact and efficient inference for multiple changepoint problems. *Statist. Comput.*, **16**, 203–213.

Fearnhead, P. and Clifford, P. (2003) On-line inference for hidden Markov models via particle filters. *J. R. Statist. Soc.* B, **65**, 887–899.

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

Hardison, R. C., Roskin, K. M., Yanf, S., Diekhans, M., Kent, W. J., Weber, R., Elnitski, L., Li, J., O'Connor, M., Kolbe, D., Schwartz, S., Forey, T. S., Whelan, S., Goldman, N., Smit, A., Miller, W., Chiaromonte, F. and Haussler, D. (2003) Covariation in frequencies of substitution, deletion, transposition, and recombination during eutherian evolution. *Genome Res.*, **13**, 13–26.

Johnson, T. D., Elashoff, R. M. and Harkema, S. J. (2003) A Bayesian changepoint analysis of electromyographic data: detecting muscle activation patterns and associated applications. *Biostatistics*, **4**, 143–164.

Liu, J. S. and Chen, R. (1998) Sequential Monte Carlo methods for dynamic systems. *J. Am. Statist. Ass.*, **93**, 1032–1044.

Liu, J. S., Chen, R. and Logvinenko, T. (2001) A theoretical framework for sequential importance sampling with resampling. In *Sequential Monte Carlo Methods in Practice* (eds A. Doucet, N. de Freitas and N. Gordon), pp. 225–246. New York: Springer.

Liu, J. S., Chen, R. and Wong, W. H. (1998) Rejection control and sequential importance sampling. *J. Am. Statist. Ass.*, **93**, 1022–1031.

Liu, J. S. and Lawrence, C. E. (1999) Bayesian inference on biopolymer models. *Bioinformatics*, **15**, 38–52.

Lund, R. and Reeves, J. (2002) Detection of undocumented changepoints: a revision of the two-phase regression model. *J. Clim.*, **15**, 2547–2554.

Oliver, J. L., Carpena, P., Hackenberg, M. and Bernaola-Galvan, P. (2004) IsoFinder: computational prediction of isochores in genome sequences. *Nucleic Acids Res.*, **32**, W287–W292.

Punskaya, E., Andrieu, C., Doucet, A. and Fitzgerald, W. J. (2002) Bayesian curve fitting using MCMC with applications to signal segmentation. *IEEE Trans. Signal Process.*, **50**, 747–758.

Ritov, Y., Raz, A. and Bergman, H. (2002) Detection of onset of neuronal activity by allowing for heterogeneity in the change points. *J. Neursci. Meth.*, **122**, 25–42.

Stephens, D. A. (1994) Bayesian retrospective multiple-changepoint identification. *Appl. Statist.*, **43**, 159–178.