Journal of the Royal Statistical Society
SERIES C Applied Statistics

Appl. Statist. (2009)
58, *Part* 5, *pp.* 641–662

Realtime sequential inference of static parameters with expensive likelihood calculations

P. Robins, V. E. Rapley and N. Green

Defence Science and Technology Laboratory, Porton Down, UK

[Received July 2007. Final revision January 2009]

Summary. A methodology is developed for making inference about parameters of a possible covert chemical or biological atmospheric release from sensor readings. The key difficulty in performing this inference is that the results must be obtained in a very short timescale (5 min) to make use of the inference for protection. The methodology that is developed uses some of the components in a sequential Monte Carlo algorithm. However, this inference problem is different from many other sequential Monte Carlo problems, in that there are no state evolution equations, the forward model is highly non-linear and the likelihoods are non-Gaussian. The algorithm that is developed can use stored output from complex physics models for more rapid update of the posterior from new data without having to rerun the models. The use of differential evolution Markov chain sampling allows new samples to diverge rapidly from degenerate sample sets. Results for inferences made of atmospheric releases (both real and simulated) of material are presented, demonstrating that the sampling scheme performs adequately despite constraints of a short time span for calculations.

Keywords: Differential evolution Markov chain; Likelihood calculations; Metropolis acceptance; Realtime computation; Sequential Monte Carlo methods

1. Introduction

Standard Bayesian analysis normally relies on Monte Carlo algorithms to perform thousands of likelihood calculations. In situations where likelihood calculations are computationally expensive and there are time or processor constraints, standard approaches may prove infeasible. There are several new Monte Carlo techniques which allow Bayesian computation when there are computational constraints including sequential Monte Carlo (SMC) methods (Doucet *et al.*, 2000, 2001), approximate Bayesian computation (Beaumont *et al.*, 2002; Sisson *et al.*, 2007) and SMC samplers (Del Moral *et al.*, 2006; Peters, 2005; Peters *et al.*, 2008).

In this paper, we use components from these approaches and combine them with some innovative, problem-specific techniques, to make inference about a highly complex multimodal posterior distribution where likelihood calculations are computationally expensive and sequential information about an event in the past is received in realtime.

The motivating example for this work is determining the static source of an atmospheric pollutant from inexact sensor readings.

Previous work in the static parameter setting includes the development of an efficient SMC scheme for problems that require inference in addition to evolving state variables (Liu and West, 2001). This approach uses parameter shrinkage and automatic kernel calculation. Alternatively, Chopin (2002) proposed a ‘black box’ framework with a Metropolis–Hastings ‘move’ kernel

Address for correspondence: V. E. Rapley, Room 102, Building 5M, Defence Science and Technology Laboratory, Porton Down, Wiltshire, SP4 0JQ, UK.
E-mail: vrapley@dstl.gov.uk

and suggested better performance than common estimation procedures in terms of robustness and execution time.

One method of addressing the source term estimation problem is to precompute dispersion runs which fill the parameter space and interpolate between them. However, this requires large amounts of processing power and storage and therefore is not suitable for this application. A second option is to undertake standard parameter inference. In this case the likelihood would be recalculated by using the physics model after each new piece of data becomes available; this would again result in significant processing requirements and so is also unsuitable.

Our methodology proposes the combination of Markov chain Monte Carlo (MCMC) sampling with aspects of an SMC algorithm. This approach is designed to minimize the computational burden of evaluating a time-dependent posterior and to minimize the likelihood of becoming 'stuck' in a local minimum.

The integration of alternative MCMC methods into the SMC algorithm has been previously suggested by Fearnhead (2002), Berzuini *et al.* (1997) and Gilks and Berzuini (2001). The standard Metropolis–Hastings acceptance scheme that was used in these references is modified for our particular problem, so that poorer source term estimates are rejected before the need to run the costly dispersion model. In this way, the computation time is better used for more likely estimates.

A crucial issue in Monte Carlo algorithms is the choice of proposal distribution. Details of this problem have been discussed at length in several references (Roberts and Rosenthal, 2001; Gelman *et al.*, 1996) and are therefore not dealt with here. To overcome the proposal choice problem in this particular application we adopt the method of ter Braak (2006) and employ differential evolution Markov chain (DEMC) methods. A local differential evolution optimization algorithm (Storn and Price, 1997) is developed into an MCMC algorithm. This idea allows the proposal to be drawn by using current samples and allows a natural scaling of the proposal distribution, thus eliminating the precise tuning of proposal distribution parameters and associated inefficiency.

Further, part of the likelihood calculation for this problem requires running a complex physics simulation. It would be prohibitively time consuming to rerun these simulations for all samples, for every new piece of data. Since it is possible to run simulations into the future, we shall save the output for inexpensive likelihood calculations from future data.

We shall also propose retaining reweighted source term estimates rather than discarding them as in previous static parameter SMC schemes. This will further help to minimize costly likelihood calculations.

The overall aim of this work is to detect jointly as well as to estimate the source parameters. The problem has many similarities to the recursive track-before-detect method that was described in Ristic *et al.* (2004), except that the likelihood functions are different and we treat the problem as a static parameter estimation problem. In previous approaches (Ristic and Gunatilaka, 2008), to perform detection of a source term, an additional binary random variable indicating the absence or presence of the source is employed. Joint detection and estimation is then treated as a hybrid (continuous–discrete) estimation problem. Alternatively, we shall incorporate the detection of the source term with the determination of the mass. This will further maximize the use of existing calculations and determine the source term existence directly from the core filter.

A detailed description of the motivating problem is given in Section 2. A derivation of the models that we shall use is given in Section 2.2. Owing to the nature of the problem there will be separate models for each type of sensor from which we receive data about a possible release. The necessary complexity of the models makes Monte Carlo computation the only practical approach for inference. However, standard computational techniques such as MCMC and SMC sampling are unsuitable on their own. We outline the computational approach that is taken in Section 3. Finally, the results for both real and simulated data are presented in Section 4, with conclusions drawn in Section 5.

2. Motivating example

If a hazardous pollutant is released into the atmosphere either from an industrial accident or a deliberate attack there is an urgent requirement to warn surrounding populations. This is achieved through hazard predictions that are obtained from dispersion models. However, to run a dispersion model we must infer the location and type of release (a source term) from continuously monitored sensor data. In addition, it is necessary to determine whether such a release has actually occurred.

Several methods of source term estimation have been proposed. However, the techniques that are implemented for posterior estimation require either the simplifying assumption of a continuous release (Thompson *et al.*, 2007) or allow a relatively large computational overhead (Delle Monache *et al.*, 2008). In addition, algorithms to update directly an estimate of the current position of a cloud of radioactive material dispersed from a facility whose location is known have been presented by Smith and French (1993) using sequentially arriving data.

In this example, we consider the source term θ to consist of location in two dimensions (l_1, l_2), release time t and mass m, i.e. $\theta = (l_1, l_2, t, m)$. These parameters are highly correlated; a more massive release further back in time and further away from the sensors may produce similar sensor readings to a small release closer to the sensors. Sensor readings will be received at arbitrary time intervals and will have uncertainty associated with them. A requirement for the inference system is that it can run continuously, i.e. it makes the most efficient use of computing resources before an event and has the ability to reset itself a predetermined length of time after data arriving at the system no longer correspond to the release.

2.1. Prior model

We cannot proceed to make inferences about source term values unless we first determine its existence. If the incoming data do not support this hypothesis, then erroneous inferences will be made. As a consequence, and to simplify the sampling, we shall use a surrogate mass parameter m^* that can assume negative values and define all sampled parameter sets with $m^* \leqslant 0$ as no release with $m = 0$. The prior on the surrogate mass is a double-exponential distribution as follows:

$$p(m^*) = \frac{1}{2\mu_{m^*}} \exp\left(-\frac{|m^*|}{\mu_{m^*}} \right). \tag{1}$$

The mean μ_{m^*} is determined according to operational information about likely release masses. When the surrogate mass parameter $m^* \leqslant 0$, then the other parameters, $\theta_{/m} = (l_1, l_2, t)$, are irrelevant. This use of a surrogate mass prior variable is a computational convenience that simplifies the sampling process (see Section 3.2) and removes the requirement for an explicit trans-dimensional sampling scheme.

We shall assume inference over a 30 km × 30 km square domain and assign a uniform prior distribution; additional narrow half-normal tails will allow the mode of the posterior to be outside the parameter space if the data suggest.

We apply a uniform prior for release times between the current time T and an hour into the past; a half-normal tail into the past, with a standard deviation of half an hour, is further appended. Note that this prior moves with the passage of realtime (see Section 3.5).

2.2. Likelihood model

We build a likelihood model to determine the source term parameters, with components of a sensor model and a dispersion model. Dispersion is a stochastic process in nature. Therefore, the physics models that are developed to represent dispersion tend to be ensemble models which account for atmospheric turbulence. In this application, we use a Gaussian dispersion model owing to the relatively fast run time in comparison with other models. This model assumes that the concentration c, given mean μ and variance σ^2 at a given location, is described by a clipped normal distribution (Lewellen and Sykes, 1986), with distribution function

$$F(c|\mu,\sigma)=\begin{cases}0 & c<0,\\ \Phi\{(c-\mu)/\sigma\} & c\geqslant 0,\end{cases} \tag{2}$$

where Φ is the standard normal distribution function.

Calculation of the dispersion parameters is undertaken by using a complex physical model which is implemented as a deterministic simulation code. Therefore the mean and variance of the clipped normal distribution (μ,σ^2) and the source term parameters (l_1,l_2,t,m) are interchangeable via the simulation, assuming that the meteorological environment is well characterized. We shall refer to μ and σ^2 as the source term for the likelihood calculations; however, this is purely a notational convenience. The simulation is treated as a 'black box' to allow different dispersion models to be used. The equations underlying the model that is used in this paper can be found in Cimorelli *et al.* (2004). The dispersion model that is used is also similar to that described in Smith and French (1993).

The only data that we receive about the source term is from downwind sensor readings. Therefore, to create a likelihood model for μ and σ^2, we need to develop sensor models from which we can determine the probability of a sensor reading given a particular concentration. These sensor models will obviously be specific to the technology that is used. The inclusion of a natural background will also be dependent on the ability of the sensor to distinguish a release from the background. We shall denote a generic sensor reading by x. The true value of the concentration is never directly observed and is therefore integrated out of the likelihood model,

$$p(x|\mu,\sigma^2)=\int \underbrace{p(x|c)}_{\substack{\text{measurement}\\ \text{density}}}\quad \underbrace{p(c|\mu,\sigma^2)}_{\substack{\text{concentration}\\ \text{density}}}\,\mathrm{d}c. \tag{3}$$

An example of a sensor model which can discriminate a signal from a background is given in Appendix A.1 and a model of a sensor which cannot discriminate a signal from the background is given in Appendix A.2.

3. Computation

We propose a posterior sampling algorithm that can make use of stored data from physics models using recent temporally varying information. The algorithm shares some of the attributes of an SMC method (Doucet *et al.*, 2001); knowledge of the posterior is represented by a discrete approximation of samples and weights as usual. However, the algorithm differs in that there are no equations for state prediction, and weighted samples from the posterior are retained as data are processed, rather than discarded, to reduce the cost of the likelihood computations.

A step-by-step description of the full method is presented in algorithmic form in Appendix C and described in the following sections.

We denote the current time as T and define a data window $[T-T_D,T]$ over which data will be considered. Data afore $T-T_D$ are considered obsolete. Thus, a maximum for likelihood computation complexity can be specified given a constant rate of data. As an alternative, a maximum number of data could be specified, thereby guaranteeing constant complexity when the data arrive at varying rates. However, we use time as a criterion because data may be delayed in transmission from the sensor to the inference system and arrive out of sequence.

When the inference engine is not processing incoming data, new samples are generated according to the current posterior distribution. We shall propose new samples by using a DEMC algorithm (ter Braak, 2006). This algorithm is a combination of differential evolution genetic optimization (Storn and Price, 1997) with standard random-walk Metropolis MCMC sampling (Metropolis *et al.*, 1953).

For each unique new sample, the dispersion model is run and the concentration probability density parameters μ and σ are stored at regular time intervals in $[T-T_D, T+T_D]$. This allows likelihood calculations for existing samples to be rapidly evaluated on receipt of new data without rerunning the dispersion model. Storing output that is associated with times in the future ensures that the likelihoods and weights of samples generated at time T can be updated from any new data arriving up to T_D in the future.

The unnormalized posterior probability of the sample proposed is calculated by using the product of the prior distribution and the likelihoods given all the data x_i currently stored, where the data are assumed conditionally independent given the source term parameter:

$$p(\theta|x) \propto p(\theta) \prod_{i=1}^{N_x} p(x_i|\theta), \tag{4}$$

where N_x is the number of data points.

3.1. Data processing

On the receipt of new data several tasks are performed.

(a) Owing to the temporal nature of the inference, both samples and data will become obsolete and are removed according to the algorithm below. Define the time at which a sample $\theta^{(i)}$ ran the dispersion code as $\tau^{(i)}$ and the time at which data x_j were measured as ψ_j.
  (i) For all pairs $\{\theta^{(i)}, \tau^{(i)}\}$, $i=1,\ldots,N$, if $\tau^{(i)} < T-T_D$ then remove $\theta^{(i)}$.
  (ii) For all pairs $\{x_j, \psi_j\}$, $j=1,\ldots,N_x$, if $\psi_j < T-T_D$ then remove x_j.
  If some data x_j are removed, the samples that are left have their likelihoods divided by the likelihood component due to the data being removed $p(x_j|\theta)$. This ensures that each sample has its total likelihood calculated from the same amount of data.

(b) If the sampler becomes overwhelmed with the overhead of updating its list of samples with data (i.e. the system is struggling to update a large number of samples with a small effective samples size in realtime), the sample list is trimmed by using sampling–importance resampling (Gordon *et al.*, 1993). Thus, the number of samples is constantly changing given computational constraints and the rate of incoming data.

(c) The weights of all the stored samples are multiplied by the likelihood of the new data given the stored model outputs. The stored total likelihood of each sample is also multiplied by the likelihood of the new data for use in the DEMC algorithm.

(d) After every data update, the DEMC chain ends are randomly redistributed among the existing samples according to their weights (see Section 3.4).

3.2. Adding new samples between data

The resample–move SMC algorithm of Berzuini *et al.* (1997) proposed to use a Metropolis–Hastings MCMC step to add diversity to a parameter estimation SMC scheme. In that case, MCMC sampling is used to sample from the state evolution equations. In our application, there are no state evolution equations as the source term is a fixed event in space and time. However, as we have defined a finite amount of data to use in our inference, we can sample directly from the posterior.

In general, the random-walk Metropolis accept–reject MCMC (O'Hagan and Forster, 2004) framework is a popular choice for an MCMC algorithm; however, the success of this depends on determining a suitable value for the size of the random steps. In the multivariate normal case, this amounts to defining the covariance matrix Σ. In our problem, the form of the posterior probability density is a function of the available data and various scenario parameters, in particular the local time varying meteorology. Therefore, it is difficult to specify a proposal distribution that will be suitable for all scenarios at all times. This restricts the use of more general and potentially more efficient algorithms, e.g. Metropolis–Hastings algorithms (Hastings, 1970). For these reasons, an adaptive scheme was required whereby the proposal distribution can be automatically modified according to the current posterior samples.

In DEMC sampling, several parallel MCMC chains are generated simultaneously. The population of current Markov chain states, $\theta^{(i)}$, $i=1,2,\ldots,N_{\text{DEMC}}$, is used to generate jump proposals:

$$\theta^{*(i)}=\theta^{(i)}+\gamma(\theta^{(j)}-\theta^{(k)})+\varepsilon, \qquad i\neq j\neq k, \tag{5}$$

where ε is a narrow normally distributed multivariate parameter and γ is a scalar indicating the size of jumps relative to the Markov chain differences.

In the source term estimation model, the indices j and k are chosen randomly from the N_{DEMC} chain ends. The index i may be iterated or chosen randomly. We choose the former. DEMC sampling allows the jump proposal to adapt itself to the current estimate of the posterior and so removes the responsibility of the user to provide a reasonable jump sampling distribution. As the indices j and k are chosen randomly and ε is symmetric, this algorithm is a special case of the random-walk Metropolis MCMC algorithm.

The DEMC algorithm is much more aggressive at expanding posterior sample sets than other resampling methods (ter Braak, 2006), e.g. mixtures of normal distributions. This property is useful when the samples become degenerate. It also means that the behaviour of the sampler is not very sensitive to the choice of ε as long as it is smaller than the target distribution.

After ter Braak (2006), we shall use

$$\gamma=2.38/\surd(2d), \tag{6}$$

where $d=\dim(\theta)=4$. We use independent normal variates for the components of ε with standard deviations of 1 m for the location parameters (l_1,l_2), 1 s for the time parameter t and 1% of the surrogate mass parameter m^*. These values are smaller than the accuracy to which we would hope to make inference in the best of cases. We shall use $N_{\text{DEMC}}=10d=40$ as suggested by ter Braak (2006).

Use of a surrogate mass m^* and maintaining the rest of the parameter vector for no-release samples simplifies the sampling algorithm by allowing the DEMC algorithm to jump between release and no release without having explicitly to jump between two models of different dimensionality.

3.2.1. Two-step acceptance

The optimal acceptance ratio for random-walk Metropolis MCMC algorithms is 0.234 (Roberts *et al.*, 1997) for multivariate normal target distributions with identical marginal densities. In general, the optimal acceptance ratio may be anywhere between 0.1 and 0.4 (O'Hagan and

Forster, 2004). Assuming an acceptance probability of about 0.25, approximately three out of every four computationally expensive dispersion model calculations would be wasted. We first note that, before a source term release, the posterior probability is dominated by the prior. As the prior distribution of a sample does not require a dispersion model calculation for its evaluation, we thus carry out the accept–reject stage of the sampling in two steps; we initially accept or reject on the prior distribution,

$$\text{uniform}(0,1) < p(\theta')/p(\theta). \tag{7}$$

If the sample is accepted according to the prior, we carry out a dispersion calculation and continue to accept if

$$\text{uniform}(0,1) < \prod_{i=1}^{N_x} p(\theta'|x_i) \Big/ \prod_{i=1}^{N_x} p(\theta|x_i), \tag{8}$$

although a dispersion calculation is only required for this likelihood calculation if the new surrogate mass $m^* > 0$.

When the posterior probability is dominated by the prior, the majority of model calculations are stored and not wasted, because the likelihood in the region of the prior is flat and nearly all the proposals which pass the prior accept–reject stage also pass through the likelihood accept–reject stage. This means that a larger population of samples that have used recent meteorological information is available to the system when data indicating a release is first received. Proof that this method satisfies the detailed balance equation is given in Appendix B.

3.3. Relative weighting of blocks of differential evolution Markov chain samples

Each block of samples that is created in between updates is an unbiased estimator of the true posterior whether the block has been created solely from MCMC sampling or a combination of MCMC sampling followed by multiple reweightings. Thus we are free to choose any mixture of these sample blocks as our total unbiased posterior estimator

$$\hat{E}[g(\hat{\theta})] = \sum_{i=1}^{N_x} \beta_i \hat{E}[g(\hat{\theta^i})], \tag{9}$$

within the constraints $\beta_i > 0$ and $\Sigma \beta_i = 1$.

Clearly, some choices of β_i will be better than others. Early blocks are likely to have smaller effective sample sizes $\hat{N}_{\text{eff}}^{(i)}$ (Kong *et al.*, 1994) than later blocks.

$$\hat{N}_{\text{eff}}^{(i)} = 1 \Big/ \sum_{j=1}^{N_i} w^{(ij)^2}. \tag{10}$$

The latest block of samples which is derived solely from MCMC sampling will have an effective sample size equal to the total number of samples in the block as all the weights are equal. We choose a heuristic scheme such that block weights β_i are updated recursively according to their effective sample size. For instance, if a single datum x_1 has been processed and MCMC sampling has continued into a second block, the relative weight of the first block is

$$\hat{N}_{\text{eff}}^{(1)} = 1 \Big/ \sum_{j=1}^{N_1} w^{(1j)^2}. \tag{11}$$

Therefore, the normalized block weights are

$$\beta_1 = \frac{\hat{N}_{\text{eff}}^{(1)}}{\hat{N}_{\text{eff}}^{(1)} + N_2}, \tag{12}$$

$$\beta_2 = \frac{N_2}{\hat{N}_{\text{eff}}^{(1)} + N_2}. \tag{13}$$

In general, after a datum x_{N_x}, the new block weights β_i' are calculated as follows. First we calculate the effective sample size of all the updated blocks by using their block weights β_i calculated after the previous datum x_{N_x-1}:

$$\hat{N}_{\text{eff}}^{(1\ldots N_x)} = 1 \bigg/ \sum_{i=1}^{N_x} \left\{ \sum_{j=1}^{N_i} (\beta_i w^{(ij)})^2 \right\}. \tag{14}$$

Then, the normalized block weights are

$$\beta_i' = \frac{\beta_i \hat{N}_{\text{eff}}^{(1\ldots N_x)}}{\hat{N}_{\text{eff}}^{(1\ldots N_x)} + N_{N_x+1}}, \qquad i = 1, \ldots, N_x, \tag{15}$$

$$\beta_{N_x+1}' = \frac{N_{N_x+1}}{\hat{N}_{\text{eff}}^{(1\ldots N_x)} + N_{N_x+1}}. \tag{16}$$

We use this sample block weighting scheme whenever inference is required. It is also used for the sampling–importance resampling in the case of data overwhelming the system (Section 3.1, point (b)) and the importance resampling of the DEMCs after data updates.

3.4. Restarting the differential evolution Markov chains after a datum update

After every data update x_i, the DEMCs are randomly redistributed between the existing samples according to their weights. For each chain, we first sample a block $i \in \{1, \ldots, N_x\}$ and

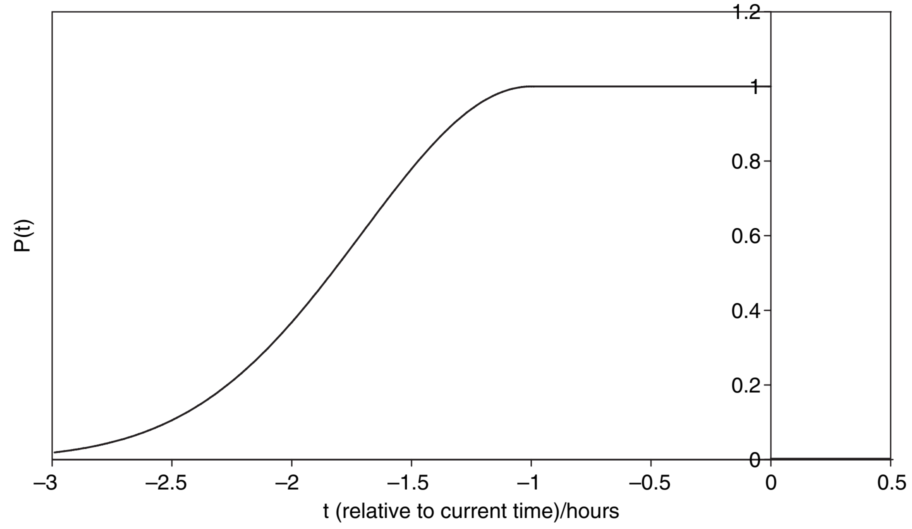

Fig. 1. Assumed distribution of the time prior relative to the current system time T

then choose a $\theta^{(ij)}$ by using $w^{(ij)}$. This is an importance sampling of the posterior approximation. Thus, for each block of samples between data updates, the chains start with an approximation of the current posterior distribution and therefore have equal weight.

3.5. *Calculation of the time prior*

The time prior is specified relatively to the current time T. This means that, as a function of absolute time, the prior distribution is evolving (Fig. 1). If we consider a sample with a time parameter 1 h in the past relative to the current time, this sample will have a time prior of 1. If, 10 min later, we proposed a new sample at the same absolute time, i.e. now 1 h 10 min ago, the prior would be much lower, leading to low acceptance probability if the latter sample is proposed from the former. Therefore, the time component of the prior for existing samples must be recalculated by using the current realtime for each instance that the prior is required for Metropolis sampling. Accordingly, weights of existing samples should be modified with the passage of time for use in inference. This is achieved by recalculating the time prior each time that a sample weight is required and modifying the weight. This applies the prior twice, which is irrelevant when the time prior is mostly uniform.

3.6. *Calculation of the probability of release*

To model the presence or absence of a release may seem to require an alternative model where the release parameters (l_1, l_2, t, m) are no longer relevant. If a trans-dimensional model was constructed, an algorithm would be required to transition from the no-release model to a release model whereby (l_1, l_2, t, m) must be generated. Instead, our surrogate mass parameter model is more akin to the product space formulation of trans-dimensional MCMC sampling (Carlin

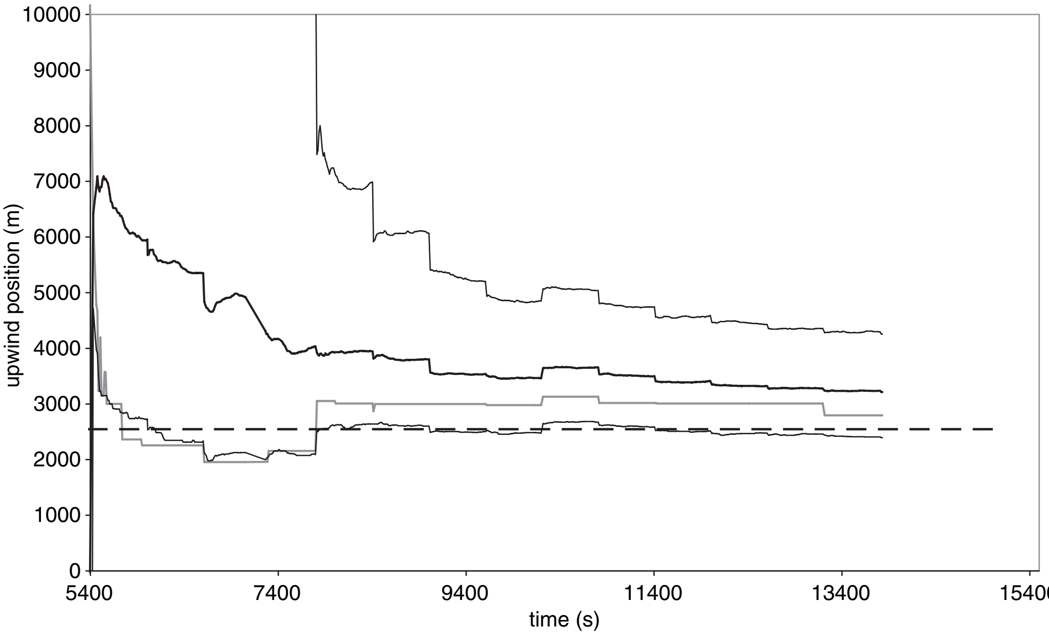

Fig. 2. Median (——) and mode (——) against the true value (– – –) with the 5% and 95% marginal intervals (——) for the l_1 (upwind) location parameter

and Chib, 1995) whereby (l_1, l_2, t) are maintained even when $m^* \leqslant 0$, using the actual priors as the pseudopriors. This maintains a constant dimensionality thus allowing DEMC sampling to be used with no further complication. Note that the surrogate mass parameter m^* could be considered as the product of a release indicator variable $r \in \{1, -1\}$ and a strictly positive real release mass m.

The likelihood for no release is inexpensive to calculate because $\mu = \sigma^2 = 0$, requiring no dispersion calculations and an analytic likelihood calculation; the concentration distribution is now a δ-function at zero instead of the clipped normal distribution in equation (2).

In an operational system, the prior on release $p(r=1)$ is likely to be small, which would lead to a proliferation of no-release samples which, in turn, could be a large computational and storage overhead despite their individually small storage requirements. The prior on release is therefore ignored for sampling as it is independent of all the other parameters and only applied when inferences are required. The value $p(r=1)=0.1$ has been used for illustrative purposes to calculate the results in this paper.

We calculate the posterior probability of release as follows:

$$p(r=1|x) = \frac{p(r=1)\sum_{i=1}^{N_x}\left(\sum_{j\in M_+^i} \beta_i w^{(ij)}\right)}{p(r=1)\sum_{i=1}^{N_x}\left(\sum_{j\in M_+^i} \beta_i w^{(ij)}\right) + \{1-p(r=1)\}\sum_{i=1}^{N_x}\left(\sum_{j\in M_-^i} \beta_i w^{(ij)}\right)}, \qquad (17)$$

where $M_+^i = \{1,2,\ldots,N_i : m_{ij}^* > 0\}$, $M_-^i = \{1,2,\ldots,N_i : m_{ij}^* \leqslant 0\}$ and m_{ij}^* are the surrogate mass parameters of the samples θ^{ij}.

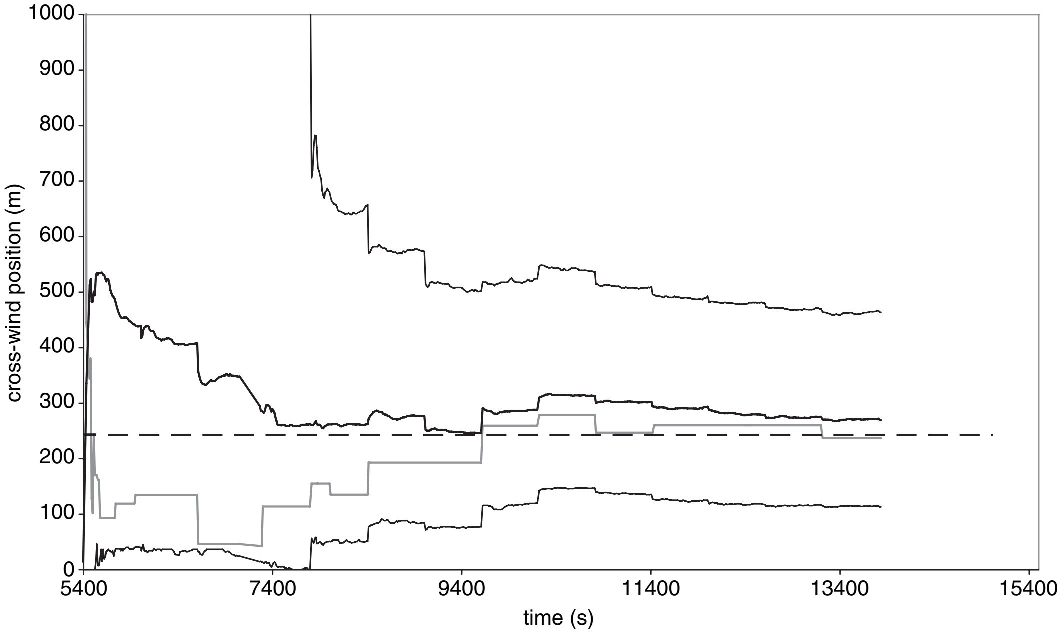

Fig. 3. Median (——) and mode (——) against the true value (– – –) with the 5% and 95% marginal intervals (——) for the l_2 (cross-wind) location parameter

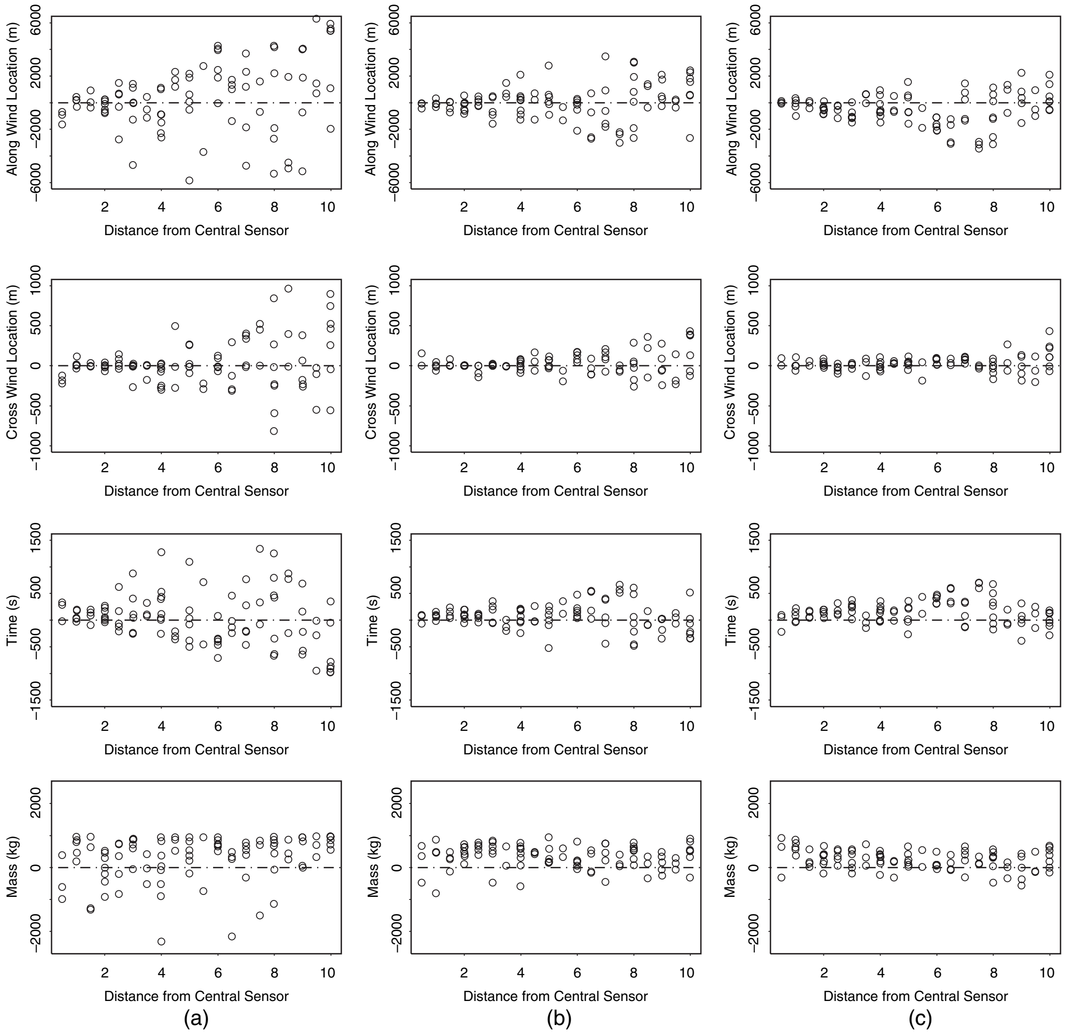

Fig. 4. Difference between the true source parameters and four model runs for a variety of possible sources (the results are shown at intervals after the first positive sensor reading; all the graphs show actual value minus predicted values and the zero line would represent perfect results; all distances are measured in kilometres): (a) instantaneous; (b) 5 min; (c) 10 min

4. Results

To test the sampling algorithm that was described above, it has been implemented in C++ on a desktop personal computer as a static library within a multithreaded message passing simulation environment that has all the necessary probabilistic algorithms and dispersion code to stimulate the sampler with realistic sensor messages. In the following sections, we describe a performance against simulated data and an inference that was made by using real data from an atmospheric release experiment using a harmless tracer gas.

4.1. Convergence

Owing to the sequential nature of this problem any standard convergence tests are difficult to employ. We first show marginal posterior statistics (mode, median and fifth and 95th percentiles) of the location parameters (l_1, l_2) in Figs 2 and 3. The time axis in these plots starts from receipt of the first data indicating a release. Inspection of the parameter statistics as a function of time indicates that, although the posterior probability distribution does change with the passage of time, the majority of that change occurs within the first 10 min. The modes of the parameter values do tend towards the true values. The difference in the median and the modal values shows the extreme skew of the posterior distribution.

4.2. Performance of the source term estimator with simulated data

To analyse the performance of the source term estimator, the system has been stimulated with realistic simulated data to mimic operational usage as closely as possible. The results of trials using synthetic data can be seen in Fig. 4. These data are generated from an arrangement of 36 sensors placed over a 2 km-diameter circular area. Sources were placed between 1 and 10 km from the central sensor location. A value of 1000 kg was used for the prior mean surrogate mass parameter μ_{m^*}. The data are presented to the source term estimator in realtime by using concurrent programming threads. The number of DEMC iterations between sensor data is therefore unpredictable. The approximate number depends on the following factors: whether the posterior is dominated by the prior; whether the dispersion code requires running; how much simulation time is required to run the dispersion code from the release time t to $T+T_D$; the complexity of the simulated environment and the temporal spacing of the data. Typical values are between 20 and 50 iterations per second, with each sensor sending data at 60-s intervals. The plots show the difference between best parameter estimates and true values immediately after a positive sensor reading and at 5- and 10-min intervals after this time. It is clear that the

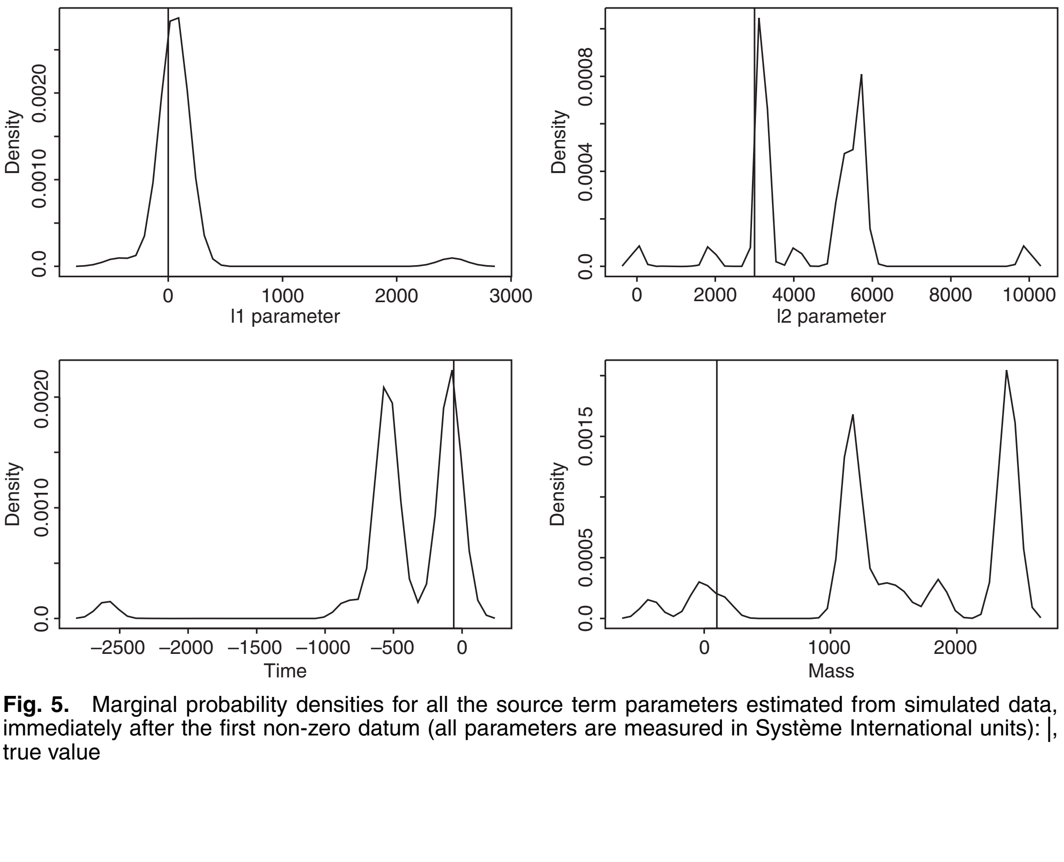

Fig. 5. Marginal probability densities for all the source term parameters estimated from simulated data, immediately after the first non-zero datum (all parameters are measured in Système International units): |, true value

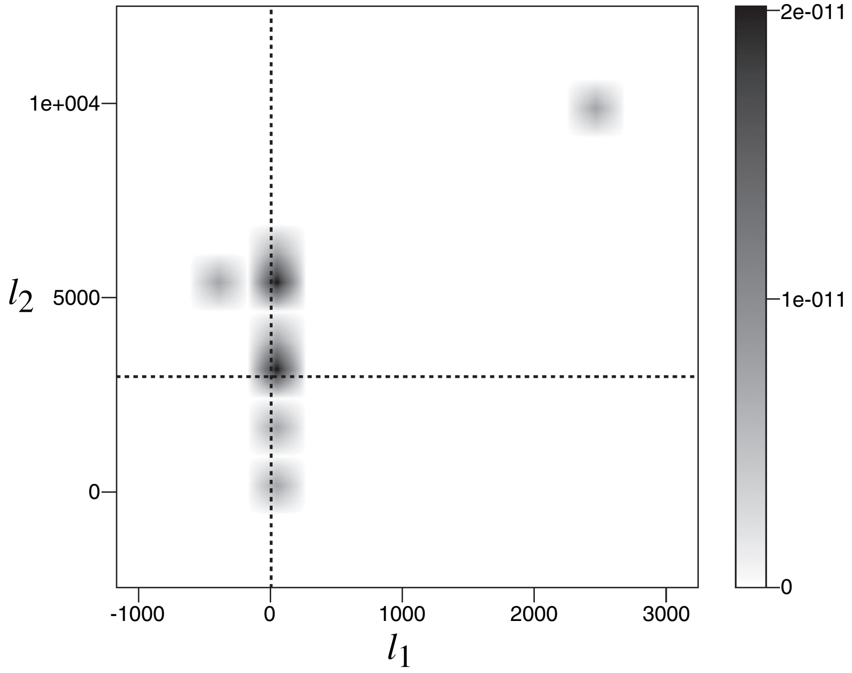

Fig. 6. Marginal posterior distribution for the location parameters immediately after the first non-zero datum (the centre of the sensor network was arbitrarily set as location (0,0) and all values are measured in metres; the wind direction is from the top down): ┼, true location

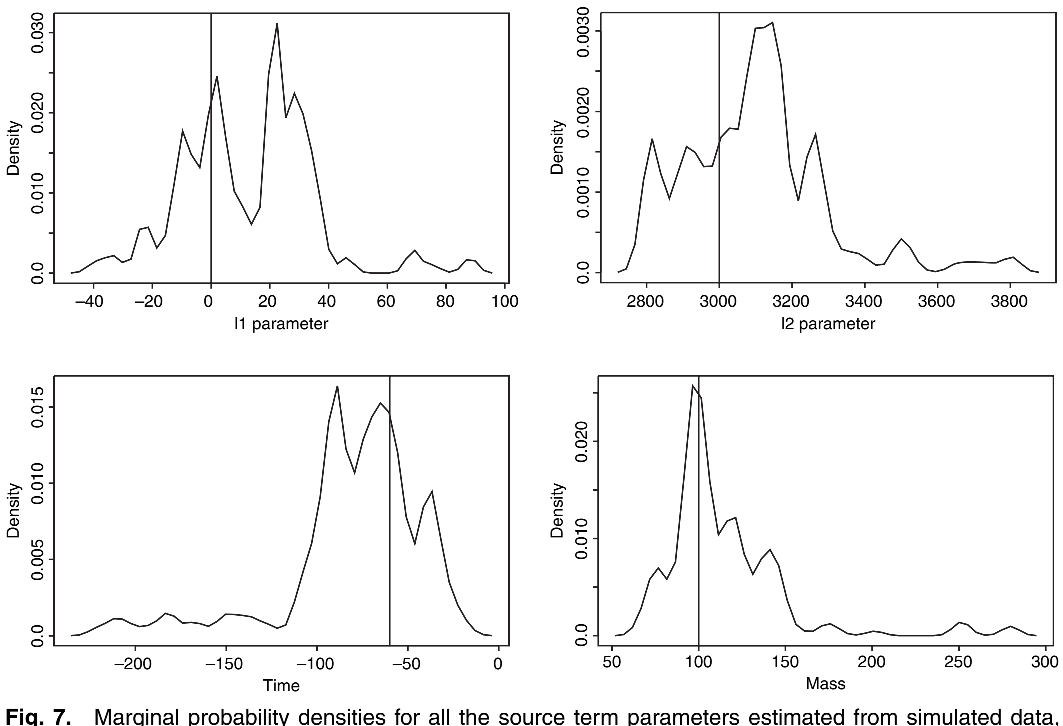

Fig. 7. Marginal probability densities for all the source term parameters estimated from simulated data, 300s after the first non-zero datum (all parameters are measured in Système International units): |, true value

estimates improve over time, as would be expected; however, the improvement between 5 and 10 min is small, indicating that convergence is being approached in the required 5-min interval.

Inference is more accurate for the cross-wind parameter. This can be explained by consideration of the problem space. There is a far more direct relationship between cross-wind position and which sensors receive positive concentrations. In addition, inference on the cross-wind position is mostly independent of the other release parameters. In the along-wind direction there is a correlation with mass and time (a small release closer to the sensor and nearer in time is similar to a larger release further away and further back in time, given early data). This correlation adds further marginal uncertainty in these dimensions. In general the source term estimator performs better when the true source is closer to the sensors, as is illustrated by the reduction in spread around the zero lines for small distances from the centre of the sensor arrangement. This is to be expected as the further away a source is from the sensors the more uncertainty there is in possible location as the material has had more time to disperse and be affected by random perturbations.

The marginal posterior probability densities for one of these inferences are displayed for the same three times as the summary plots in Figs 5–10. In Figs 5 and 6 it is clear that the inference engine is suffering severe sample impoverishment. However, the samples that are present represent a high level of uncertainty about the true answer and provide useful locations in the parameter space for the DEMC sampling to proceed. Figs 7 and 8 show the marginal posterior distributions 300 s after the first non-zero datum. The samples provide a reasonable representation of the posterior probability distribution and are unbiased. The spread in the

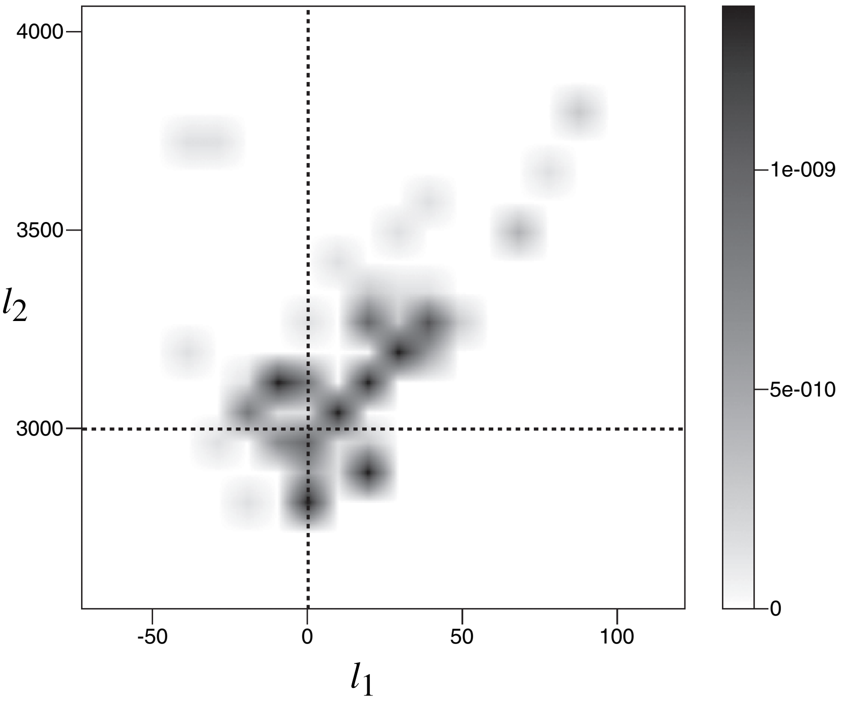

Fig. 8. Marginal posterior distribution for the location parameters 300 s after the first non-zero datum (the centre of the sensor network was arbitrarily set as location (0,0) and all values are measured in metres; the wind direction is from the top down): ┈┼┈, true location

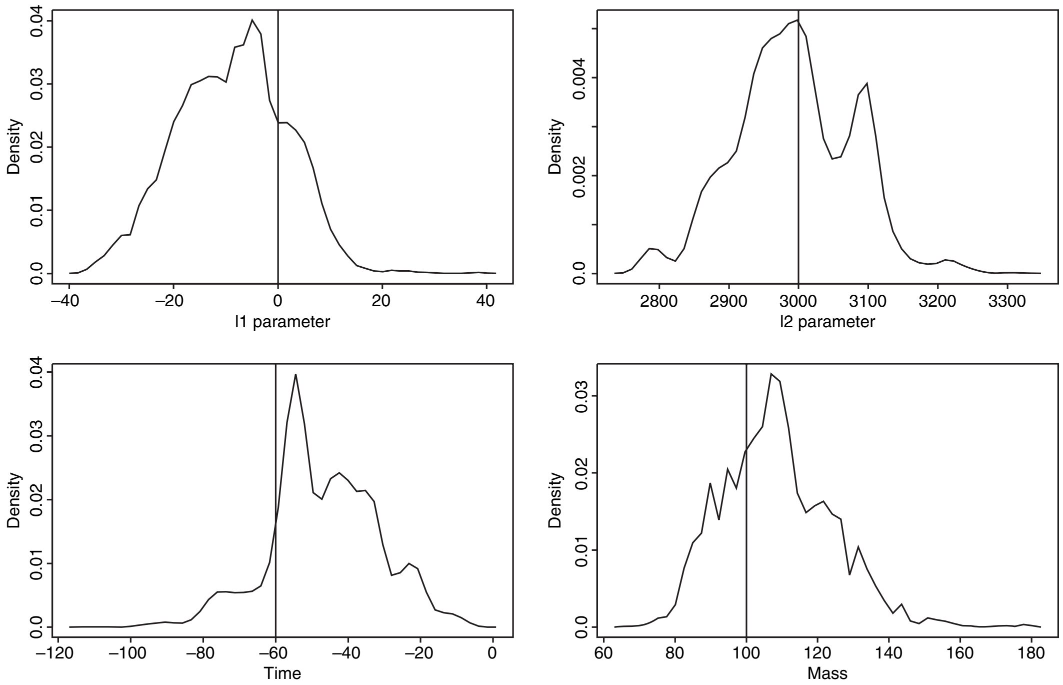

Fig. 9. Marginal probability densities for all the source term parameters estimated from simulated data, 600 s after the first non-zero datum (all parameters are measured in Système International units): |, true value

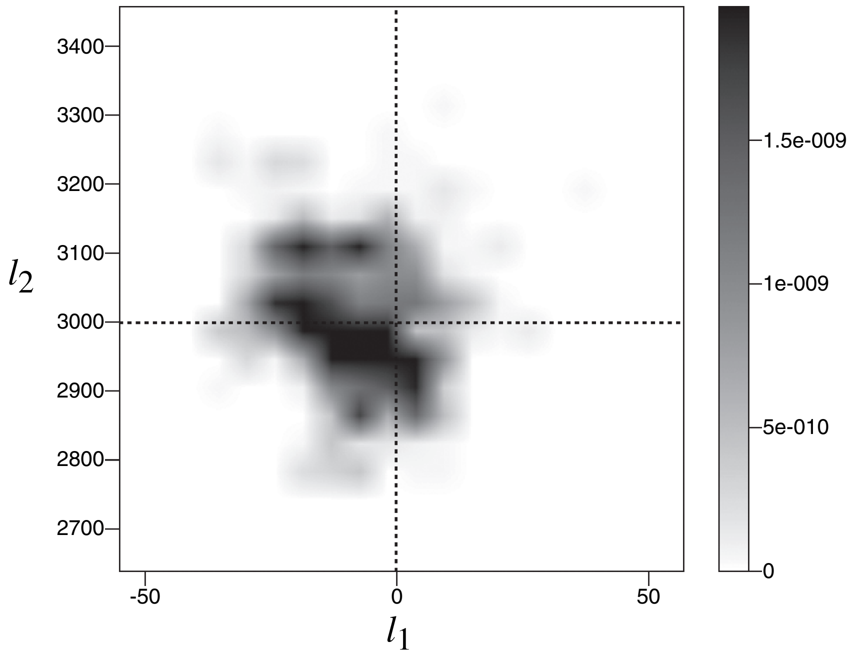

Fig. 10. Marginal posterior distribution for the location parameters 600 s after the first non-zero datum (the centre of the sensor network was arbitrarily set as location (0,0) and all values are measured in metres; the wind direction is from the top down): ┄┼┄, true location

posterior distribution is sufficiently small to provide useful information for hazard prediction. Figs 9 and 10 show the marginal posterior distributions 600 s after the first non-zero datum. The samples now have a very small spread and remain unbiased. This demonstrates that the inference engine can converge to the correct answer in a timely fashion when presented with a large amount of data.

4.3. Performance of the source term estimator with real experimental data

To test this algorithm against real data, we use sensor readings from the dipole pride data set (Biltoft, 1998). The portion of the total data set that was used provides high frequency sensor readings, following a release of a tracer gas, from a single row of sensors.

In this example the sensor readings were passed to the source term estimator in time order. A value of 100 kg was used for the prior mean of the surrogate mass, μ_{m*}. A full posterior estimate was then generated 5 min after the first positive sensor reading. The marginal probability density plots for l_1, l_2, t and m can be seen in Fig. 11.

The actual posterior distribution is multimodal, but in all cases the true values lie within the support of the posterior probability distribution. Fig. 12 gives a clearer indication of the form of the posterior distribution for the location parameters and shows that, although the true release location lies within the posterior, there is a large amount of correlation along the wind direction. The wind is in the direction of the superimposed arrow. This is consistent with distributions that are observed in the simulated data.

Atmospheric dispersion is naturally chaotic and therefore exact inference cannot be achieved. In light of this the source term estimator has exceeded expectations in that within the allocated

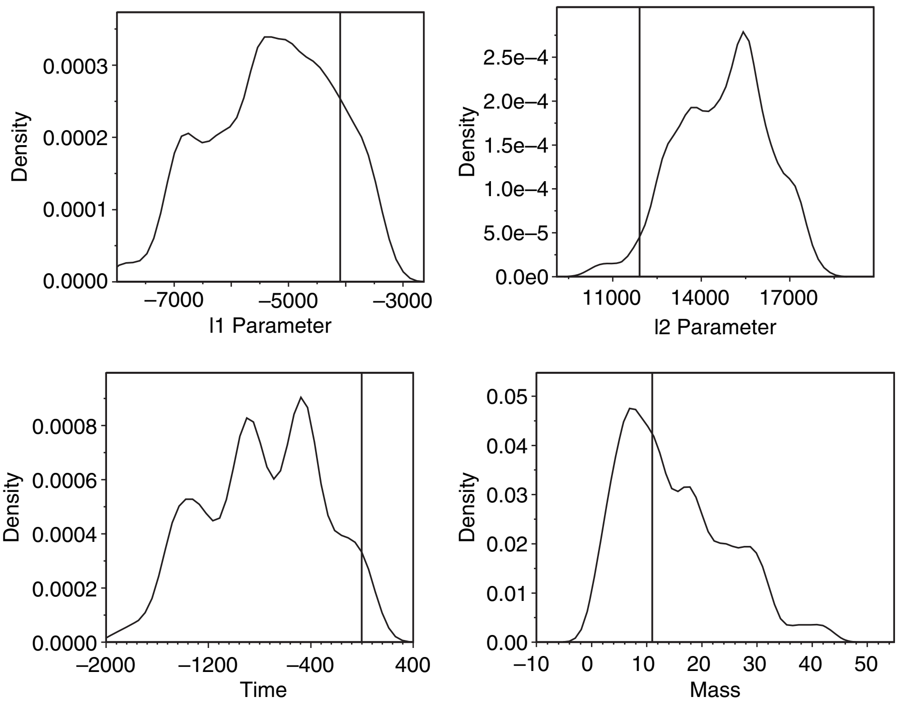

Fig. 11. Marginal probability densities for all the source term parameters (all parameters are measured in Système International units): |, true value

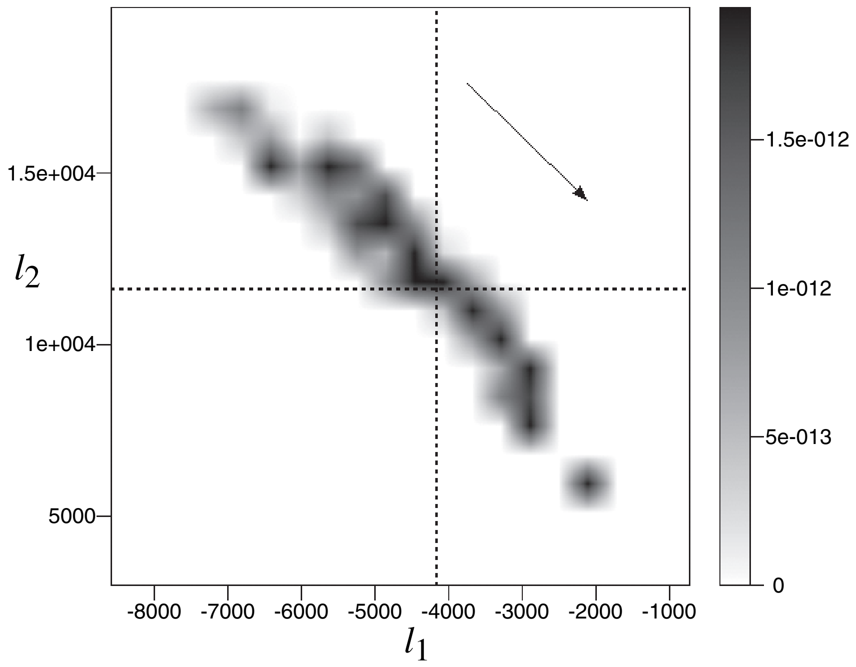

Fig. 12. Marginal posterior distribution for the location parameters (the centre of the sensor network was arbitrarily set as location (0,0) and all values are measured in metres): ┄┼┄, true location; ↘, wind direction

timeframe of 5 min a posterior distribution has been generated which contains the true source parameters and is significantly smaller than the prior distribution. This information would then be adequate to produce hazard estimates as required.

5. Conclusions

We have presented a sampling algorithm that yields rapid inferences when likelihood calculation requires output from complex models and the data arrive sequentially after the event of interest, but the parameters to be estimated are not evolving in time. We obtained reasonable sampling of the posterior probability density and subsequent inferences given the data received in a short timeframe.

Acknowledgements

We thank the Ministry of Defence Research Acquisition Office for funding this research. We also thank Kevin Watkins, Paul Thomas and Miles Elsden for their support in writing this paper. We are grateful for reviewer's comments which allowed us to improve this paper significantly. Finally, we thank Professor Jon Forster of Southampton University for assistance in the probability of release sampling and in clarifying the description of the relative weighting of samples from different blocks of DEMC samples.

This work is Crown copyright, Dstl 2009, and is published with the permission of the Defence Science and Technology Laboratory on behalf of the Controller of Her Majesty's Stationery Office.

Appendix A: Sensor likelihood models

A.1. A discriminating sensor

We model a sensor that returns a substance type and a bar reading indicating the concentration in a sampled volume of air. The thresholds for the bars are spread logarithmically. This sensor can measure substances of interest regardless of other substances in the atmosphere. We assume that the sensors send data at 60-s intervals to avoid correlation between sensor readings; any data at higher frequency are not used in the inference. Previous research (Jones, 2002) has shown that the assumption of independence is valid for this timescale under the assumptions that we have already made about the parameter space.

We model the internal process of obtaining a bar reading from the sensor as

$$p(V) = \frac{1}{\surd\{2\pi(\kappa c + J)\}} \exp\left\{-\frac{1}{2}\frac{(V-c)^2}{\kappa c + J}\right\} \tag{18}$$

where V is an internal signal in the sensor, which is a representation of the concentration c at a given point in time and is normally distributed with a measurement error of $\surd(\kappa c + J)$. We assume that the variance on the sensor reading is proportional to the concentration, with the constant of proportionality denoted as κ. To allow for the fact that there will never be a zero variance a small constant J is then added. The values of κ and J are assumed to be known. The sensor produces one of a fixed number B of bar readings, so the measurement is thresholded between perfectly known limits $(C_1 \ldots C_B)$; uncertainty in the threshold values is accounted for in the uncertainty of the signal voltage. We can then obtain the likelihood of obtaining a particular reading given a particular concentration by integrating the distribution that is given in equation (18) between the appropriate concentration thresholds C_i for the respective bars (bar_i):

$$p(\text{bar}_i|c) = \frac{1}{\surd\{2\pi(\kappa c + J)\}} \int_{C_i}^{C_{i+1}} \exp\left\{-\frac{1}{2}\frac{(V-c)^2}{\kappa c + J}\right\} \mathrm{d}V. \tag{19}$$

The likelihood from the sensor must be combined with the likelihood from a dispersion model as described in Section 2.2. This results in the model

$$\begin{aligned} p(\text{bar}_i|\mu,\sigma^2) = \int_0^\infty & \frac{1}{\surd\{2\pi(\kappa c + J)\}} \int_{C_i}^{C_{i+1}} \exp\left\{-\frac{1}{2}\frac{(V-c)^2}{\kappa c + J}\right\} \mathrm{d}V \\ & \times \left[\Phi\left(\frac{-\mu}{\sigma}\right)\delta(c) + \frac{1}{\sigma\surd(2\pi)} \exp\left\{-\frac{(c-\mu)^2}{2\sigma^2}\right\}\right] \mathrm{d}c. \end{aligned} \tag{20}$$

The term in square brackets is the probability density function of the clipped normal distribution that is defined in equation (2). The integral is precomputed for a range of values of bar_i, μ and σ^2 by using Romberg numerical integration (Press *et al.*, 2002) and stored for interpolation.

A.2. A non-discriminating sensor

Non-discriminating sensors pose further difficulties in that they cannot distinguish between a naturally occurring background measurement and a measurement that is attributable to a release of interest, e.g. a simple particle counter. A particle counter returns the number of particles that are seen in a set time period in a known volume of air. These particles are a combination of those generated by a release of interest and those that are present naturally in the atmosphere.

For a sensor to be able to distinguish a release of interest from background particle counts, a statistical model of the background must be estimated. The simplest background model configuration would be a single scalar probability construct for all the sensors; however, this would not account for variation between sensors as a consequence of their location. The most flexible model would be a multivariate joint probability construct for the entire network of sensors; however, this poses large complexities in model implementation. We therefore use a single scalar model for each sensor.

The actual number of particles that are sampled at the sensor is assumed to be Poisson with mean n_T, where $n_T = n_p + b$; here n_p is the mean number of release particles in a sampled volume of air and b denotes the background. We assume that the background can be modelled as a stationary or at least very slowly varying distribution compared with the signal from a release. An exponentially weighted moving average

(Hull, 2005) model is used to calculate an estimate $\hat{b}$ of the current value of b:

$$\hat{b}_T = (1-\lambda)\hat{b}_{T-1} + \lambda n_T \tag{21}$$

where n_T denotes the particle count from the sensor at time T. The value λ is calculated dynamically by assuming that the characteristic timescale of the fastest variation in background can be estimated as ΔT:

$$\lambda = \exp\left(-\frac{\delta T}{\Delta T}\right) \tag{22}$$

where δT is the time between sensor readings, which need not be constant.

To compensate for a release signal changing the estimated mean and variance of the background distribution, the probability that a new sensor reading is derived solely from the background distribution is calculated. If this probability is more than an arbitrary value of 5 standard deviations away from the background model it is not updated with the new particle count. The value of 5 standard deviations can be adjusted according to the type of background that is encountered.

The particle counters are assumed to be perfectly efficient so all particles entering the counter are counted successfully, with no omissions or double counts. The value $\hat{b}_T$ is an estimate for the true background b_T, which adds an uncertainty of magnitude $\sigma_{\hat{b}_T}$. The full likelihood integral should therefore be

$$p(n_T|\hat{b}_T, \sigma_{\hat{b}_T}, \mu, \sigma^2) = \int_0^\infty \int_0^\infty \mathrm{Po}(n_T|n_p + b_T)\, N(b_T|\hat{b}_T, \sigma^2_{\hat{b}_T})\, \mathrm{CN}(n_p|\mu, \sigma^2)\, \mathrm{d}b_T\, \mathrm{d}c, \tag{23}$$

where CN denotes the clipped normal probability density.

The uncertainty $\sigma_{\hat{b}_T}$ on the estimate of the background is ignored to avoid the calculation of a complex double integral. For a reasonable rate of data (i.e. $\delta T \ll \Delta T$), the uncertainty in the background estimation should be a small factor in the likelihood calculation compared with the uncertainty in the mass concentration c; thus uncertainty in $\hat{b}_T$ is also ignored. In addition, the Poisson distribution is replaced by a normal distribution, which is a reasonable approximation if $n_p + \hat{b}_T$ is large. This results in the following model for a non-discriminating sensor, where V denotes the volume of air sampled and μ_p denotes the mean mass of a release particle.

$$p(n_T|\hat{b}_T, \mu, \sigma^2) = \int_0^\infty N\left(n_T|n_p + \hat{b}_T, \frac{cV}{\mu_p} + \hat{b}_T\right) \mathrm{CN}(n_p|\mu, \sigma^2)\, \mathrm{d}c. \tag{24}$$

The appearance of the mean number of particles in a volume of air, n_p, in the variance of the normal distribution prevents this integral from being evaluated analytically. The integral is evaluated by using Romberg (Press *et al.*, 2002) integration with finite bounds calculated to contain the domain where the integrand is significant.

Appendix B: Proof of detailed balance for two-step rejection

The basic equation for detailed balance is (O'Hagan and Forster, 2004)

$$p(\theta|\theta')\, \pi(\theta') = p(\theta'|\theta)\, \pi(\theta), \tag{25}$$

where $\pi(\theta)$ is the desired probability density function and $p(\theta'|\theta)$ the probability density function of making a new sample θ' from a current sample θ. We define $\pi(\theta)$ as a posterior probability density function composed as the product of a prior $p(\theta)$ and a likelihood $p(x|\theta)$ divided by a normalizing constant:

$$\pi(\theta) = \frac{p(x|\theta)\, p(\theta)}{\int p(x|\theta)\, p(\theta)\, \mathrm{d}\theta}. \tag{26}$$

Substituting this into equation (25) we obtain the equation for two-step detailed balance:

$$p(\theta|\theta')\, p(\theta')\, p(x|\theta') = p(\theta'|\theta)\, p(\theta)\, p(x|\theta). \tag{27}$$

We define the proposal distribution as $g(\theta'|\theta)$. To prove that condition (27) is satisfied, it is necessary to consider four sampling possibilities:

(a) $p(\theta) > p(\theta')$ and $p(x|\theta) > p(x|\theta')$;

(b) $p(\theta) > p(\theta')$ and $p(x|\theta) < p(x|\theta')$;
(c) $p(\theta) < p(\theta')$ and $p(x|\theta) > p(x|\theta')$;
(d) $p(\theta) < p(\theta')$ and $p(x|\theta) < p(x|\theta')$.

B.1. Case (a): $p(\theta) > p(\theta')$ and $p(x|\theta) > p(x|\theta')$

In the forward step, $\theta \to \theta'$, the algorithm definitely accepts θ' so

$$p(\theta'|\theta) = g(\theta'|\theta). \tag{28}$$

In the reverse step, $\theta' \to \theta$, so $\theta \sim g(\theta|\theta')$ with probability

$$\min\left\{1, \frac{p(\theta)}{p(\theta')}\right\} \min\left\{1, \frac{p(x|\theta)}{p(x|\theta')}\right\} = \frac{p(\theta)}{p(\theta')} \frac{p(x|\theta)}{p(x|\theta')} \tag{29}$$

in this case. It follows that

$$p(\theta|\theta') = \frac{p(\theta)}{p(\theta')} \frac{p(x|\theta)}{p(x|\theta')} g(\theta|\theta'). \tag{30}$$

Substituting equations (30) and (28) back into equation (27) yields

$$g(\theta'|\theta)\; p(\theta) p(x|\theta) = \frac{p(\theta)}{p(\theta')} \frac{p(x|\theta)}{p(x|\theta')} g(\theta|\theta')\; p(\theta')\; p(x|\theta'), \tag{31}$$

which is satisfied for $g(\theta|\theta') = g(\theta'|\theta)$.

B.2. Case (b): $p(\theta) > p(\theta')$ and $p(x|\theta) < p(x|\theta')$

In the forward step, $\theta \to \theta'$, the algorithm definitely accepts on the first step and then accepts with probability

$$\min\left\{1, \frac{p(x|\theta')}{p(x|\theta)}\right\} = \frac{p(x|\theta')}{p(x|\theta)}, \tag{32}$$

in this case. Therefore,

$$p(\theta'|\theta) = g(\theta'|\theta) \frac{p(x|\theta')}{p(x|\theta)}. \tag{33}$$

In the reverse step, $\theta' \to \theta$, the algorithm accepts in the first step with probability

$$\min\left\{1, \frac{p(\theta)}{p(\theta')}\right\} = \frac{p(\theta)}{p(\theta')}, \tag{34}$$

in this case and then definitely accepts in the second step. Therefore,

$$p(\theta|\theta') = g(\theta|\theta') \frac{p(\theta)}{p(\theta')}. \tag{35}$$

Substituting equations (33) and (35) back into equation (27) yields

$$g(\theta|\theta') \frac{p(x|\theta)}{p(x|\theta')}\; p(\theta')\, p(x|\theta') = g(\theta'|\theta) \frac{p(\theta')}{p(\theta)}\; p(\theta)\, p(x|\theta), \tag{36}$$

which is satisfied for $g(\theta|\theta') = g(\theta'|\theta)$.

Cases (c) and (d) follow by symmetry (simply replace θ by θ' and vice versa in cases (a) and (b)). Thus detailed balance is demonstrated for the two-step rejection method by using Metropolis acceptance at each step.

Appendix C: Step-by-step description of source term estimation algorithm

Step 1: initialization—

(a) for $i = 1, \ldots, N$, sample $\theta^{(i)} \sim \pi(\theta)$; assign initial weights $w^{(i)} \propto 1$, for $i = 1, \ldots, N$; set $N_x = 0$;

(b) select N_{DEMC} chain indices j_k according to weights $\{w^{(i)}\}$; set $\theta^{(k)}_{\text{DEMC}} = \theta^{(j_k)}$ for $k = 1, \ldots, N_{\text{DEMC}}$.

Step 2: update—if new sensor data x^{new} received, set $N_x = N_x + 1$.

(a) For each $\{\theta^{(i)}, \tau^{(i)}\}$, $i = 1, \ldots, N$, if $\tau^{(i)} < T - T_D$, remove $\theta^{(i)}$.

(b) For each $\{x_j, \phi_j\}$, $j = 1, \ldots, N_x$, if $\phi_j < T - T_D$, remove x_j; reweight according to

$$\tilde{w}^{(i)} = w^{(i)} / \prod_{j:\phi_j < T - T_D} p(x_j|\theta^{(i)})$$

for all $i = 1, \ldots, N$; modify total likelihoods

$$L^{(i)} = L^{(i)} / \prod_{j:\phi_j < T - T_D} p(x_j|\theta^{(i)})$$

for all $i = 1, \ldots, N$.

(c) Update total likelihoods $\Pi_{j=1}^{N_x} p(x_j|\theta^{(i)})$ for each $i = 1, \ldots, N$; reweight according to $\tilde{w}^{(i)} = \tilde{w}^{(i)} p(x^{\text{new}}|\theta^{(i)})$ for $i = 1, \ldots, N$; normalize in blocks $w^{ij} = \tilde{w}^{(ij)} / \Sigma_{j=1}^{N_i} \tilde{w}^{(ij)}$ for $i = 1, \ldots, N_x$ for $j = 1, \ldots, N_i$.

(d) Importance resample new DEMCs: select i according to block weights β_i (Section 3.3); select sample j from block i according to weights $\{w^{(ij)}\}$; set $\theta^{(k)}_{\text{DEMC}} = \theta^{(ij)}$.

Step 3: resample—if the system is 'overwhelmed' then retain $\{\theta^{(ij)}, w^{(ij)}\}$ with probability

$$\beta_i w^{(ij)} / \max_{ij}(\beta_i w^{(ij)})$$

for $i = 1, \ldots, N_x$ for $j = 1, \ldots, N_i$; recalculate $\beta_i = N_i / \Sigma_{i=1}^{N_x} N_i$ for $i = 1, \ldots, N_x$; normalize in blocks $w^{ij} = 1/N_i$ for $i = 1, \ldots, N_x$ for $j = 1, \ldots, N_i$; resample DEMCs as in step 2(d).

Step 4: proposal—generate a new sample point according to DEMC sampling, $\theta' = \theta^{(i)}_{\text{DEMC}} + \gamma(\theta^{(j)}_{\text{DEMC}} - \theta^{(k)}_{\text{DEMC}}) + \varepsilon$, for $i, j, k \in \{1, \ldots, N_{\text{DEMC}}\}$ and $i \neq j \neq k$.

Step 5: two-step Metropolis sampling—set $\theta^{(N+1)} = \theta'$ and $\theta^{(i)}_{\text{DEMC}} = \theta'$ with probability

$$\min\{1, p(\theta'|X) / p(\theta^{(i)}_{\text{DEMC}}|X)\};$$

otherwise set $\theta^{(N+1)} = \theta^{(i)}_{\text{DEMC}}$.

Step 6: loop—repeat from step 2 until convergence.

References

Beaumont, M. A., Zhang, W. Y. and Balding, D. (2002) Approximate Bayesian computation in population genetics. *Genetics*, **162**, 2025–2035.

Berzuini, C., Best, N., Gilks, W. and Larizza, C. (1997) Dynamic conditional independence models and markov chain monte carlo methods. *J. Am. Statist. Ass.*, **92**, 1403–1412.

Biltoft, C. A. (1998) Dipole pride 26: Phase II of defense special weapons agency transport and dispersion model validation. *Technical Report.* US Army Dugway Proving Ground, Dugway.

ter Braak, C. J. (2006) A Markov chain Monte Carlo version of the genetic algorithm Differential Evolution: easy Bayesian computing for real parameter spaces. *Statist. Comput.*, **16**, 230–249.

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

Chopin, N. (2002) A sequential particle filter method for static models. *Biometrika*, **89**, 539–552.

Cimorelli, A. J., Perry, S. G., Venkatram, A., Weil, J. C., Paine, R. J., Wilson, R. B., Lee, R. F., Peters, W. D., Brode, R. W. and Paumier, J. O. (2004) Aermod: description of model formulation. *Report EPA-454/R-03-004.* Environmental Protection Agency, Washington DC.

Delle Monache, L., Lundquist, J. K., Kosovic, B., Johannesson, G., Dyer, K. M., Aines, R. D., Chow, F. K., Belles, R. D., Hanley, W. G., Larsen, S. C., Loosmore, G. A., Nitao, J. J., Sugiyama, G. A. and Vogt, P. J. (2008) Bayesian inference and Markov Chain Monte Carlo sampling to reconstruct a contaminant source at continental scale. *J. Appl. Meteorol. Clim.*, to be published.

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

Doucet, A., de Freitas, N. and Gordon, N. (eds) (2001) *Sequential Monte Carlo Methods in Practice*. New York: Springer.

Doucet, A., Godsill, S. J. and Andrieu, C. (2000) On sequential Monte Carlo sampling methods for Bayesian filtering. *Statist. Comput.*, **10**, 197–208.

Fearnhead, P. (2002) MCMC, sufficient statistics and particle filters. *J. Computnl Graph. Statist.*, **11**, 848–862.
Gelman, A., Roberts, G. and Gilks, W. (1996) Efficient metropolis jumping rules. *Bayesn Statist.*, **5**, 559–608.
Gilks, W. R. and Berzuini, C. (2001) Following a moving target—Monte Carlo inference for dynamic Bayesian models. *J. R. Statist. Soc.* B, **63**, 127–146.
Gordon, N. J., Salmond, D. J. and Smith, A. F. M. (1993) Novel approach to nonlinear non-Gaussian Bayesian state estimation. *IEE Proc.* F, **140**, 107–113.
Hastings, W. K. (1970) Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, **57**, 97–109.
Hull, J. (2005) *Options, Futures and Other Derivatives*. Englewood Cliffs: Prentice Hall.
Jones, C. (2002) Predicting concentration time structure: final report. *Report Dstl/TR04034.* Defence Science and Technology Laboratory, Porton Down.
Kong, A., Liu, J. and Wong, W. (1994) Sequential imputation and Bayesian missing data problems. *J. Am. Statist. Ass.*, **89**, 278–288.
Lewellen, W. S. and Sykes, R. I. (1986) Analysis of concentration fluctuations from lidar observations of atmospheric plumes. *J. Clim. Appl. Meteorol.*, **2**, 1145–1154.
Liu, J. and West, M. (2001) Combined parameter and state estimation in simulation based filtering. In *Sequential Monte Carlo Methods in Practice* (eds A. Doucet, N. de Freitas and N. Gordon). New York: Springer.
Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H. and Teller, E. (1953) Equations of state calculations by fast computing machines. *J. Chem. Phys.*, **21**, 1087–1092.
O'Hagan, A. and Forster, J. (2004) *Kendall's Advanced Theory of Statistics*, vol. 2B, 2nd edn. London: Arnold.
Peters, G. W. (2005) Topics in sequential Monte Carlo samplers. *MSc Thesis*. University of Cambridge, Cambridge.
Peters, G., Sisson, S. and Fan, Y. (2008) On sequential Monte Carlo, partial rejection control and approximate Bayesian computation. *Probab. Surv.*, to be published.
Press, W. H., Teukolsky, S. A., Vetterling, W. T. and Flannery, B. P. (2002) *Numerical Recipes in C++: the Art of Scientific Computing*, 2nd edn. Cambridge: Cambridge University Press.
Ristic, B., Arulampalam, M. and Gordon, A. (2004) *Beyond the Kalman Filter: Particle Filters for Tracking Applications*. Norwood: Artech House.
Ristic, B. and Gunatilaka, A. (2008) Information driven localisation of a radiological point source. *Informn Fusn*, **9**, 317–326.
Roberts, G. O., Gelman, A. and Gilks, W. R. (1997) Weak convergence and optimal scaling of random walk Metropolis algorithms. *Ann. Appl. Probab.*, **7**, 110–120.
Roberts, G. and Rosenthal, J. (2001) Optimal scaling for various Metropolis-Hastings algorithms. *Statist. Sci.*, **16**, 351–367.
Sisson, S. A., Fan, Y. and Tanaka, M. M. (2007) Sequential Monte Carlo without likelihoods. *Proc. Natn. Acad. Sci. USA*, **104**, 1760–1765.
Smith, J. and French, S. (1993) Bayesian updating of atmospheric dispersion models for use after an accidental release of radioactivity. *Statistician*, **42**, 501–511.
Storn, R. and Price, K. (1997) Differential evolution: a simple and efficient heuristic for global optimization over continuous spaces. *J. Globl Optimizn*, **11**, 341–359.
Thompson, L., Hirst, B., Gibson, G., Gillespie, S., Jonathan, P., Skeldon, K. and Padgett, M. (2007) An improved algorithm for locating a gas source using inverse methods. *Atmos. Environ.*, **41**, 1128–1134.