MCMC, Sufficient Statistics and Particle Filters

Paul Fearnhead

Department of Statistics

University of Oxford

July 2001

Address for Correspondence: Department of Statistics,
1, South Parks Road,
Oxford,
OX1 3TG.
email: fhead@stats.ox.ac.uk

Abstract

We consider how to implement Markov chain Monte Carlo (MCMC) moves within a particle filter. Previous, similar, attempts have required the complete history ('trajectory') of each particle to be stored. We show how certain MCMC moves can be introduced within a particle filter when only summaries of each particles' trajectory are stored. These summaries are based on sufficient statistics. Using this idea, the storage requirement of the particle filter can be substantially reduced, and MCMC moves can be implemented more efficiently. We illustrate how this idea can be used for both the bearings-only tracking problem and a model of stochastic volatility. We give a detailed comparison of the performance of different particle filters for the bearings-only tracking problem. MCMC, combined with a sensible initialization of the filter and stratified resampling, produces substantial gains in the efficiency of the particle filter.

Keywords: Bearings-only tracking, Importance sampling, Stochastic Volatility

1 Introduction

Dynamic state-space models are commonly used in engineering, econometrics, geophysics, and bio-medical and other scientific disciplines. They model processes where there is an underlying state of interest, which evolves over time. The state is partially observed at successive time-points, and these observations are used to make inference about the state of the system. For such problems, inferences need to be made in real-time. Computationally intensive Bayesian methods, based on Markov chain Monte Carlo (MCMC) simulation, analyzing a batch of observations at once, will generally be too slow. Hence it will be necessary to use recursive methods, which estimate the current state using the estimate of the state at the previous time-point, and the latest observation.

The idea of the particle filter is to approximate the posterior distribution of the state (the *filtering density*) by a set of possible realisations of the state (these realisations are called *particles*). Each particle is assigned a weight, and the filtering density is approximated by a discrete distribution whose support is the set of particles, and with the probability mass of each particle being proportional to its weight. The particle filter algorithm specifies how the particles and their weights are propagated through time (normally via a derivative of importance sampling), to model the dynamics of the state and take account of the information in new observations. The efficiency of the particle filter depends largely on how the weighted particles are updated, and numerous schemes have been proposed (For example, Gordon *et al.* 1993, Liu and Chen 1995, Kitagawa 1996). See Liu and Chen (1998) and Doucet *et al.* (2001) for a review of particle filters, and their application to many interesting and illustrative examples. For all these particle filter algorithms, rigorous convergence results exist (see Crisan and Doucet, 2000, and references therein), which prove that as the number of particles tends to infinity, the particle filter's discrete approximation converges to the true filtering density. Under stronger regularity conditions, a Central Limit Theorem exists (Del Moral and Guionnet, 1999).

One problem with particle filters (and recursive algorithms in general) is that, over time, there is a continued loss of accuracy in the approximation of the posterior distribution of the state. Due to the recursive nature of the algorithm, the approximation of the filtering density at time $t+1$ is obtained by updating the weighted particles that approximate the filtering density at time t. Updating the weighted particles only introduces further inaccuracies, and these inaccuracies will accumulate over time. In practice this is seen by particles which are highly correlated (clustered in just one region of the state-space).

One solution to this problem is to integrate MCMC methods into the particle filter algorithm as suggested by Fearnhead (1998), Berzuini and Gilks (2001), and Gilks and Berzuini (2001). If each particle stores a realisation of the complete history of the state (a *trajectory*), and not just a realisation of the current state, then MCMC can be used to simulate new trajectories. Applying the MCMC moves to each particle will reduce the correlation between the particles.

In this paper we will describe how sufficient statistics can be used to summarize the trajectory of each particle, so that MCMC moves can be applied. Storing sufficient statistics, as opposed to complete trajectories, will substantially reduce the computational and memory requirements of the algorithm. We give a number of examples of how this idea can be applied in practice, and give a detailed study of these methods for the bearings-only tracking problem. For this problem, we can summarize the history of a particle by a 5-dimensional vector. Storing the trajectory over t time points requires storing a $(2t-2)$-dimensional vector. In the example we consider, we track the ship over 100 time points (in practice a much longer tracking period could be imagined), which results in a 40-fold reduction in the memory requirements of the filter, and a gain in the computational efficiency of implementing the MCMC steps.

2 State-Space Models

We consider the following state-space model. Let θ_t denote the state at time t, and β a set of unknown parameters. We assume a prior, $\pi(\theta_0, \beta)$, and that the state evolves according to the *system equation*,

$$\theta_t = f(\theta_{t-1}, w_t, \beta), \tag{1}$$

where f is a known, possibly nonlinear function, and w_t is a realisation of a random variable W_t. In order to draw inference about θ_t and β we make observations z_t, which satisfy the *observation equation*,

$$z_t = g(\theta_t, v_t, \beta), \tag{2}$$

where g is a known, possible nonlinear, function, and v_t is a realisation of a random variable V_t. We make the assumptions that the density functions of both V_t and W_t are known (for all t), and that V_t, V_τ, W_t, W_τ are independent for $t \neq \tau$.

To make this model concrete, we now introduce two examples. These examples will be used later in the paper to illustrate how sufficient statistics can be used to reduce the storage burden of implementing MCMC within particle filters.

Example 1 *Bearings-only Tracking*

Consider tracking a ship, solely using measurements of the bearing of the ship with a stationary observer. The state, $\theta_t = (x_t, \dot{x}_t, y_t, \dot{y}_t)^T$, is the position and the velocity of the ship. This satisfies the following system equation (taken from Carpenter *et al.*, 1999)

$$\theta_t = \begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix} \theta_{t-1} + \begin{pmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \end{pmatrix} w_t, \tag{3}$$

where w_t is the realisation of a bivariate Gaussian random variable, with zero-mean, and covariance $\nu^2\mathbf{I}_2$ (where $\mathbf{I}_2$ is the 2×2 identity matrix). For this model, the definition of the velocity at time t is just the change in position between time t and time $t-1$; and w_t is the effect of acceleration between time $t-1$ and t. (For the motivation of this model see Section 5 and references therein.)

The observation equation is

$$z_t = \tan^{-1}(y_t/x_t) + \pi I(x_t < 0) + v_t, \tag{4}$$

where v_t is the realisation of a univariate Gaussian random variable with zero-mean, and variance η^2; and $I(x_t < 0)$ is an indicator function which takes the value

1 is $x_t < 0$ and the value 0 otherwise. We assume that $\tan^{-1}(\alpha) \in (-\pi/2, \pi/2]$, and hence $z_t \in (-\pi/2, 3\pi/2]$. For this model $\beta = (\eta^2, \nu^2)$.

Example 2 *Stochastic Volatility*

Stochastic volatility models are used in econometrics to model the changing volatility of asset returns (for a review, see Shephard, 1996). A simple univariate stochastic volatility model (Taylor, 1982) is

$$\theta_t = \phi\theta_{t-1} + w_t, \tag{5}$$

$$z_t = v_t \sigma \exp\{\theta_t/2\}, \tag{6}$$

where w_t and v_t are realisations from independent univariate Gaussian distributions, each with zero-mean, and with variances ν^2 and 1 respectively. (In this model, θ_t is the unobserved volatility and z_t is the observed asset return.) For this model, $\beta = (\phi, \nu^2, \sigma^2)$. For examples of MCMC and particle filter methods being applied to this problem see Pitt and Shephard (1998), and references therein.

3 Particle Filters

We could be interested in drawing inference about the past, present or future values of the state, or about the parameters. In this paper we consider only the filtering problem (inference about the current value of the state and the parameters). All the information about the state and parameters is contained in their joint posterior distribution. Denoting $z_{1:t} = \{z_1, \ldots, z_t\}$ to be the set of observations up to time t, then the filtering problem reduces to calculating (or approximating) the filtering density, $p(\theta_t, \beta | z_{1:t})$.

The idea of the particle filter is to approximate the filtering density using a set of weighted particles. Each particle consists of a possible value of the state and parameters. The filtering density is approximated by a discrete distribution, whose support is the set of particles. The probability mass assigned to each particle is proportional to that particle's weight. Given a function h, and a set of particles

$\{(\theta_t{}^i, \beta^i)\}_{i=1,\ldots,N}$, with weights $\{q_t^i\}_{i=1,\ldots,N}$, the approximation of $\mathrm{E}(h(\theta_t, \beta)|z_{1:t})$ is

$$\left(\sum_{i=1}^{N} q_t^i h(\theta_t{}^i, \beta^i)\right) \Bigg/ \left(\sum_{i=1}^{N} q_t^i\right).$$

After the $(t-1)$st iteration of the particle filter we have a set of weighted particles which approximate the filtering density at time $t-1$. Using this approximation, we can obtain the following approximation to the filtering density at time t:

$$\begin{aligned} p(\theta_t, \beta|z_{1:t}) &\propto p(z_t|\theta_t, \beta) \int p(\theta_t|\theta_{t-1}, \beta) p(\theta_{t-1}, \beta|z_{1:t-1}) \mathrm{d}\theta_{t-1} \\ &\approx p(z_t|\theta_t, \beta) \sum_{i=1}^{N} q_{t-1}^i p(\theta_t|\theta_{t-1}{}^i, \beta^i) I(\beta = \beta^i), \end{aligned} \tag{7}$$

where $I(\beta = \beta^i)$ takes the value 1 if $\beta = \beta^i$, and the value 0 otherwise. A set of weighted particles which approximate (7) can be generated using importance sampling (MCMC and rejection sampling have also been suggested; see Berzuini *et al.* 1996 and Hürzeler and Künsch 1998). A natural proposal density is

$$\sum_{i=1}^{N} q_{t-1}^i p(\theta_t|\theta_{t-1}{}^i, \beta^i) I(\beta = \beta^i), \tag{8}$$

in which case the new particles are given weights proportional to the likelihood of the observation at time t. This proposal density is used in the Bayesian Bootstrap filter of Gordon *et al.* (1993), however it is not optimal. Improved proposal densities are used in the ASIR filter (Pitt and Shephard, 1999); in fact Pitt and Shephard (1999) show that it is sometimes possible to sample directly from (7).

Note that resampling is inherent in sampling from (8), or any proposal density. To sample from (8), we can first resample the particles at time $t-1$ and then propagate each resampled particles, using the system equation, to produce particles at time t. An improvement on multinomial resampling is the stratified resampling scheme of Carpenter *et al.* (1999); such a resampling scheme produces a stratified sample from (8) instead of an independent one. It also possible to implement a particle filter step without any resampling. (See Liu and Chen, 1998, for further details and a discussion of the advantages and disadvantages of resampling.)

The important factor that affects the efficiency of the particle filter, is the accumulation of inaccuracies over many time-steps. Even if direct simulation from (7) is possible, inaccuracies will still accumulate due to the discrete approximation of the filtering density. The result is often the clustering of particles in small areas of the state-space. It is a particular problem for parameters, as particles may contain only a small number of distinct values for the parameters.

One solution to this problem is to use MCMC to allow particles to move to areas of high filtering density, and to generate new parameter values for the particles (see Gilks and Berzuini, 2001; Berzuini and Gilks, 2001). In the next section, we consider this idea in more detail, and show how gains in computational efficiency and in storage requirements can be made over previously suggested methods.

4 MCMC

MCMC is a method for simulating from a target distribution which is known only upto a normalising constant (see Gilks *et al.*, 1996, for a review of MCMC methods). Both Fearnhead (1998) and Gilks and Berzuini (2001) (see also the companion paper Berzuini and Gilks, 2001) suggest applying MCMC to particle filters. A Markov chain which has a marginal stationary distribution of $p(\theta_t, \beta | z_{1:t})$ is designed, and the transitions of this Markov chain are then applied to the particles in the particle filter. Afterwards, the weighted particles are still a representation of the filtering density, and the correlation between the particles will be reduced. This is one method of overcoming the accumulation of inaccuracies over time (if sufficiently many MCMC moves are applied, then the new set of weighted particles will be a good representation of the filtering density, regardless of the values of the initial set of weighted particles). Note that, as compared to standard MCMC, there is no need for a burn-in period, as the initial set of weighted particles can be viewed as an approximate (weighted) sample from the target density.

In order to design a Markov chain with the correct stationary distribution, it is necessary to know that stationary distribution up to a normalising constant. While this is not the case for $p(\theta_t, \beta | z_{1:t})$, it is the case for the joint posterior density of the trajectory (the states at all time points) and the parameters. Denoting

$\theta_{0:t} = \{\theta_0, \dots, \theta_t\}$:

$$p(\theta_{0:t}, \beta | z_{1:t}) \propto \pi(\theta_0, \beta) \prod_{i=1}^{t} p(\theta_i | \theta_{i-1}, \beta) p(z_i | \theta_i, \beta). \tag{9}$$

Gilks and Berzuini (2001) suggests that each particle should store the complete trajectory of the state ($\theta_{0:t}$), and not just the current value. It is then possible to design a Markov chain whose stationary distribution is (9), and to apply the transitions of this Markov chain to the particles. They suggest applying the Markov chain transitions after the resampling stage of the particle filter algorithm. It is also valid to implement the MCMC moves at other points in the particle filter algorithm, for example before resampling. However applying the MCMC moves immediately after resampling is likely to be most efficient. To illustrate this, consider the ideal MCMC move: one which generates a new particle, $(\theta_{0:t}, \beta)$, independently from the correct posterior distribution. Applying such a move after resampling will produce an independent sample of particles, each with identical weights; while implementing such a move prior to resampling will produce an independent sample of particles, each with unequal weights. Inference (estimates of posterior means) based on the former is more accurate.

Implementing MCMC within the particle filter algorithm in this way suffers from the problem that the trajectories need to be stored. This leads to memory storage problems if long time-series are analysed. Furthermore, the computational cost of implementing an MCMC move will increase with t (as it involves evaluating Equation 9; see the comments in Gilks and Berzuini 2001). We now show how both these problems can be overcome. The idea is that summaries of the trajectory may contain all the information about certain parameters, or functions of the state. Only these summaries need to be stored in order to use MCMC moves to update these parameters or functions. This idea is based on the idea of sufficient statistics. The summaries of the trajectories are sufficient statistics for the parameters or functions of the state which we wish to apply MCMC moves to.

Consider writing $(\theta_{0:t}, \beta) = (\gamma, \delta)$ (perhaps after a suitable re-parameterisation). For example, $\gamma = (\beta, \theta_t)$ (the current state and parameters), and $\delta = (\theta_{0:t-1})$ (the trajectory to time $t-1$). Let $s = s(\delta, z_{1:t})$ be a statistic for γ. The statistic s is sufficient for γ if $p(\gamma|\delta, z_{1:t}) = p(\gamma|s)$. A necessary and sufficient condition for a statistic to be sufficient is given by the Factorisation Theorem. That is, a statistic s is sufficient for γ if and only if there exist functions $k_1(\cdot)$ and $k_2(\cdot)$ such that

$$p(\gamma, \delta, z_{1:t}) = k_1[\gamma, s(\delta, z_{1:t})]k_2(\delta, z_{1:t}). \tag{10}$$

4.1 Use of Sufficient Statistics in Particle Filters

The idea behind using sufficient statistics, is that given a sufficient statistic s for γ, a Markov chain can be designed whose stationary distribution is $p(\gamma|s)$. This stationary distribution is equal to $p(\gamma|\delta, z_{1:t})$, and hence nothing is lost by updating γ conditional on s, rather than conditional on all the measurements and the rest of the trajectory. (The drawback of such an implementation is that, for any given sufficient statistic, the set of possible MCMC moves is restricted: we can only update γ, and not δ.)

To design such a Markov chain, we need to know $p(\gamma|s)$ upto a normalising constant. However we know $p(\gamma, \delta, z_{1:t})$ up to a normalising constant (from Equation 9) and we can factorise this density according to (10), to obtain $k_1[\gamma, s(\delta, z_{1:t})]$. Yet, $p(\gamma|s) \propto k_1[\gamma, s(\delta, z_{1:t})]$, so we can calculate $p(\gamma|s)$ up to a normalising constant, and hence we can use standard MCMC algorithms to design a suitable Markov chain. As particle filters are recursive algorithms, in order to use these sufficient statistics to summarise the trajectories of the particles, it is necessary to be able to update them recursively. (We must be able to calculate the value of the statistic at time t from its value at time $t-1$ and the values of the state and observation at time t).

We now give a number of examples, both to make this idea concrete, and to show how the idea can be implemented in practice. All these examples are based on either the bearings-only tracking model (see Example 1) or the stochastic volatility model (see Example 2); and the sufficient statistics that are found can be updated recursively, and hence used to summarise trajectories in the particle filter. See Section 5 for a detailed analysis of how these ideas work in practice when applied to the bearings-only tracking model.

Example 3 *System Parameters*

Consider finding sufficient statistics for the system noise, ν, in the bearings-only tracking model. Using the factorisation theorem (10), we get (remembering that $\pi(\cdot)$ denotes the prior)

$$k_1[\nu, s(\theta_{0:t}, z_{1:t}, \eta)] = \pi(\nu)\nu^{-2(t-1)} \exp\left\{-\frac{1}{2\nu^2}\sum_{i=2}^{t}[(\dot{x}_i - \dot{x}_{i-1})^2 + (\dot{y}_i - \dot{y}_{i-1})^2]\right\}.$$

Thus $s = \sum_{i=2}^{t}[(\dot{x}_i - \dot{x}_{i-1})^2 + (\dot{y}_i - \dot{y}_{i-1})^2]$ is a sufficient statistic for ν. Furthermore, if we assume a conjugate, inverse gamma prior for ν^2 then the conditional density for ν^2 (given s) will also be inverse gamma, and can be sampled from directly (i.e. a Gibbs sampling move can be performed); see Gilks and Berzuini (2001).

For the stochastic volatility model, the system equation is parameterized by (ϕ, ν). The statistic $\sum_{i=1}^{t}(\theta_i - \phi\theta_{i-1})^2$ is sufficient for ν, while applying the factorisation theorem for the parameter ϕ gives

$$k_1[\phi, s(\theta_{0:t}, z_{1:t}, \nu, \sigma)] = \pi(\phi) \exp\left\{-\frac{1}{2\nu^2}\sum_{i=1}^{t}\left(\phi^2{\theta_{i-1}}^2 - 2\phi\theta_i\theta_{i-1}\right)\right\}.$$

So $\sum_{i=1}^{t}{\theta_{i-1}}^2$, $\sum_{i=1}^{t}\theta_i\theta_{i-1}$ and ν^2 are sufficient statistics for ϕ.

Example 4 *Fixed-Lag Statistics*

Due to the Markov nature of the system equation, it is possible to use the conditional independence structure of the states to obtain a simple summary of the trajectory. The idea is to note that for any $t > \tau > 0$, the states $\theta_{(t-\tau+1):t}$ are independent of $\theta_{0:(t-\tau-1)}$ given $(\beta, \theta_{t-\tau})$. This means that β and $\theta_{t-\tau}$ are sufficient statistics for $\theta_{(t-\tau+1):t}$.

The choice of τ involves a trade-off between the amount of storage and the dependence of the current state on the sufficient statistic (which affects the efficiency of using MCMC moves to update the value of the current state). For the stochastic volatility model the correlation of θ_t and $\theta_{t-\tau}$ is just ϕ^τ, and this could be used as a guideline for choosing τ.

For the Stochastic Volatility model and this choice of sufficient statistic, a sensible MCMC update is given by Shephard and Pitt (1998). Their idea is to update blocks of state values in one MCMC move, using the Metropolis-Hastings algorithm with a multivariate Gaussian proposal density.

Example 5 *Re-parameterisation*

For the bearings-only tracking model, consider the re-parameterisation

$$\lambda^2 = x_0^2 + y_0^2,\ \rho_i^2 = \frac{x_i^2 + y_i^2}{x_0^2 + y_0^2},\ \text{for } i = 1, \ldots, t,$$

$$\alpha_i = \tan^{-1}(y_i/x_i) + \pi I(x_i < 0),\ \text{for } i = 0, \ldots, t.$$

The interpretation of this parameterisation is that λ is the target's range at time 0; ρ_i is the ratio of the range of the target at time i to its range at time 0; and α_i is the bearing of the target at time i. Updating λ, conditional on the other parameters, is equivalent to scaling the whole trajectory of the target in or out.

Fearnhead (1998; Proposition 6.1) shows that x_0, y_0, x_1, y_1, and $\sum_{i=2}^{t}[(\dot{x}_i - \dot{x}_{i-1})^2 + (\dot{y}_i - \dot{y}_{i-1})^2]$ are sufficient statistics for λ. Furthermore, Fearnhead (1998) suggests using rejection sampling to simulate from the conditional distribution of λ. The idea of using MCMC moves which scale the track comes from the work of Carpenter *et al.* (1996), where this scaling move was found to be central to designing an MCMC algorithm for the bearings-only tracking problem which mixes well.

5 Bearings-only Tracking

We demonstrate the effectiveness of incorporating MCMC within the particle filter algorithm, and the use of sufficient statistics for summarising trajectories, by a detailed analysis of the bearings-only tracking problem (see Example 1). This is a much studied model, on which many different filters have been tested (see Aidala, 1979; Aidala and Hammel, 1983; Peach, 1995; Anderson and Itlis, 1996; LeCadre and Tremois, 1998, for a few examples).

Particle filters have been applied to this problem by Gordon *et al.* (1993), Fearnhead (1998; Section 6.2), Carpenter *et al.* (1999), Pitt and Shephard (1999) and Gilks and Berzuini (2001). The exact model used has varied, mainly through the definition of the velocity. Here we use the model of Carpenter *et al.* (1999), where the velocity at time t is the change in position between times $t-1$ and t. The model of Pitt and Shephard 1999 or Gilks and Berzuini 2001, defines the velocity at time t as the change in position between times t and $t+1$. Our choice of velocity improves the performance of the particle filter by delaying the simulation of the velocity by one time unit; as the choice of velocity can depend on one further measurement. The implementation of the simplest particle filter on our model is equivalent to the implementation of Pitt and Shephard's ASIR filter on the alternative model.

We follow the example of Gilks and Berzuini (2001), and assume that the system noise ν is unknown (but that the observation noise is known). We take a conjugate, gamma prior on ν^{-2}, and a Gaussian prior on θ_0. The gamma distribution has shape parameter 1 and scale parameter 10^{-5}. The Gaussian prior on the state $(x_0, y_0, \dot{x}_0, \dot{y}_0)^T$ has mean $(0.01, 1.0, 0.002, -0.01)^T$ and a diagonal covariance matrix, diag$\{1.0, 1.0, 0.01, 0.01\}$. A track of length 100 was simulated, with $\nu^2 = 5 \times 10^{-6}$ and $(x_0, y_0, \dot{x}_0, \dot{y}_0)^T = (-1.25, 2.0, 0.05, -0.04)^T$. The simulated track is shown in Figure 1.

[Figure 1 about here]

We compared six different implementations of particle filters:

- the SIR filter of Gordon *et al.* (1993).
- the SIR filter, but with stratified sampling (Carpenter *et al.*, 1999), and an alternative proposal density for particles at the initialisation step (this proposal density takes account of the first observation; see Fearnhead 1998, Section 6.2.6.). We call this particle filter the ISS (initialised, stratified sampling) filter.
- the ASIR file of Pitt and Shephard (1999) (see Carpenter *et al.*, 1999, for how to implement the ASIR filter for this bearings-only tracking problem). This filter also used the initialisation step and stratified sampling of the ISS filter.
- the SIR, ISS and ASIR filters, but each with MCMC moves.

The MCMC moves that we used were to scale the complete trajectory in and out (see Example 5), and to update the system noise parameter (see Example 3). In order to perform these MCMC moves, the trajectory can be summarised by five sufficient statistics (see Examples 3 and 5 for details). The particles were altered by the MCMC moves every second iteration of the particle filter algorithm. This regime appeared to give the best trade-off between the gain in accuracy and the increase in computational cost. We did not implement the 'local perturbation' move of Gilks and Berzuini (2001), because the whole trajectory needs to be stored to use this move. If the whole trajectory is stored, then there is an infinite choice of possible MCMC moves that could be used, and the work of Carpenter *et al.* (1996) suggests that local moves have poor mixing properties.

We chose the number of particles used by each filter so that one iteration of each particle filter could be performed in one second (on a 600MHz PC). The performance of each filter was summarised using the effective sample size (ESS) of Carpenter *et al.* (1999). This is the ratio of an estimate of the posterior variance of a parameter (or function of the state), to the variance of the estimates of the posterior mean of that parameter (or function) across 1,000 independent runs of the algorithm. (The posterior variance was estimated using the particle filter output from all 6,000 runs: 1,000 runs for each of the six filters.) The ESS can be interpreted as an estimate of the number of independent samples from the true posterior which would be required to estimate the posterior mean as accurately as the particle filter does. Comparing algorithms based on their ESS is equivalent to comparing them based on the variance, across independent runs of each algorithm, of estimates of this posterior mean. The ESS can vary over choices of the parameter (or function of the state), and we record it for the range of the target and the system noise. The ESSs for the Cartesian coordinates of the target are almost identical to those for the range.

[Table 1 about here]

The results are summarised in Table 1. Using MCMC moves improves the efficiency of all three particle filters. The gain in efficiency is largest for the SIR filter, and for estimating the range of the target. The gains in efficiency increase with time. Substantial gains are also obtained from the improvements incorporated in the ISS filter. Only small improvements are obtained by incorporating either the intialisation scheme or stratified sampling on its own (results not shown). The initialisation scheme is needed in order to start the particle filter with an accurate representation of the filtering density at time 1. The stratified sampling is then better able, than multinomial sampling, at maintaining a good representation of the filtering density at later times. Any increase in accuracy obtained by using the ASIR filter is more than offset by the increase in computing time (compare the ISS and ASIR filters, both with and without MCMC moves). The ISS filter with MCMC moves is the most efficient.

[Figure 2 about here]

The output from one run of the ISS filter (using MCMC moves) is summarised in Figures 1–2. The filtered estimates of the target's positions are shown in Figure 1. Initially, the filter struggles to track the target, but when the target passes close to the observer, an accurate estimate of the target's position is obtained. This is maintained over future time steps. The observations are most informative when the target passes close to the observer; this is not only shown by the better estimates of the target's position, but also by the smaller uncertainty in the estimate of the target's range.

Also of interest is how well the filter is able to estimate the unknown parameter ν. It is hard to estimate ν because of the difficulty differentiating between a close target which travels and accelerates slowly, and a fast-moving (and accelerating) target which is much further away. The observations are only informative about ν and the range of the target, r, through the ratio r/ν. The joint posterior distribution for r and ν at time 100, shows strong positive correlation between r and ν (estimated correlation of 0.43), which is an artifact of this. The information about the actual values of r and ν comes from their joint prior; thus for the particle filter to be able to estimate these quantities well, this information must be kept over all the iterations of the filter. The true values of r and ν lie in an area of high posterior probability (results not shown).

The uncertainty in the value of ν is still large, with posterior standard deviation of 1.4×10^{-3} (the true value of ν is 2.2×10^{-3}). The observations contained information about ν; the standard deviation of the posterior distribution decreases over time (see Figure 2). The short period when the standard deviation increases corresponds with the time when the target makes its largest manoeuvre. This resulted in an increase, in both the mean and the standard deviation of the posterior distribution for ν. For large t, the posterior standard deviation decreases at rate $t^{-1/2}$; which is the expected rate for independent, equally informative observations.

6 Discussion

We have extended the idea of incorporating MCMC into particle filters (Gilks and Berzuini 2001 and Berzuini and Gilks 2001). In general to update particles using MCMC moves requires storing the complete trajectory of each particle. Storing trajectories requires a large computer-memory, and for large trajectories, implementing the MCMC moves can become slow, as it is necessary to evaluate functions of the complete trajectory.

The idea presented in this paper is to use sufficient statistics to summarise trajectories. Parameters for which summary statistics exist, can be updated using MCMC moves if their summary statistics are stored (there is no need to store the rest of the trajectory). Summarising the trajectory in this way leads to a large reduction in the storage requirements, and the cost of implementing the MCMC moves no longer increases with the length of the trajectory. A disadvantage of this idea is that only parameters which have suitable summary statistics can be updated using MCMC moves. However in Section 4.1 we gave a number of examples of how this idea can be used in practice. In particular, the use of fixed-lag statistics (see Example 4) is possible for any state-space model.

For any choice of summary statistics, the set of possible MCMC moves is restricted; whereas if the complete trajectory of each particle is stored, then any MCMC move could be implemented. If MCMC moves based on our choice of summary statistics mix poorly, then it is likely that storing the whole trajectory (if this is feasible), and implementing MCMC moves conditional on the whole trajectory will be more efficient. (In practice, the best implementation of MCMC moves within the particle filter will be problem dependent.)

Furthermore, if the complete trajectory is stored then it will always be possible to generate a set of (virtually) independent particles from an arbitrarily close approximation to the true posterior, just by applying a sufficiently large number of MCMC moves to each particle (though this will be impracticable for real-time problems). If only sufficient statistics are stored then this appealing, albeit theoretical, property will no longer hold. Applying the MCMC moves conditional on values of the sufficient statistics, can at best generate a new particle from its conditional distribution given the values of the statistics. As more and more observations are processed, the set of values of the sufficient statistics stored by the particles will become highly positively correlated. As a result, the new particles generated using the MCMC moves will also be positively correlated (regardless of efficacy of the MCMC moves, and the number of moves applied to each particle), and the values of the new particles will be biased towards values suggested by the current set of sufficient statistics (as opposed to being a representative sample from the true posterior).

The efficacy of using MCMC moves will depend on the choice of transitions. Our recommendation is, where possible, to use experience gained from applying MCMC to the problem of interest. For example, the choice of the scaling move used for the bearings-only tracking problem was based on the results of the MCMC scheme of Carpenter *et al.* (1996). When designing transitions for the stochastic volatility model, the results and work of Shephard and Pitt (1998) should be used.

In Section 5 we gave a detailed comparison of different particle filters for the bearings-only tracking problem. MCMC moves can be used to update the system noise variance, and to scale whole trajectories, provided each particles' trajectory is summarised by six sufficient statistics. These MCMC moves result in a large increase in efficiency for all three particle filter algorithms that were considered.

Throughout this paper we have considered dynamic state-space models. Chopin (2000) proposes a particle filter algorithm for static problems, and the idea of using particle filters for static problems is mentioned in Gilks and Berzuini (2001). Central to such algorithms will be the incorporation of MCMC methodology (and the ideas presented here). Whether, and for what types of static problems, particle filters will be more efficient than standard MCMC remains to be seen.

Acknowledgements I would like to thank Peter Clifford for helpful discussions about this work, and comments on early forms of this paper.

References

Aidala, V. J. (1979) Kalman filter behaviour in bearings-only tracking applications. *IEEE Transactions on Aerospace and Electronic Systems*, **15**, 29–39.

Aidala, V. J. and Hammel, S. E. (1983) Utilization of modified polar coordinates for bearings-only tracking. *IEEE Transactions on Automatic Control*, **28**, 283–294.

Anderson, K. L. and Itlis, R. A. (1996) A consistent estimator criterion for multi-sensor bearings-only tracking. *IEE Transactions on Aerospace and Electronic Systems*, **32**, 108–119.

Berzuini, C. and Gilks, W. (2001) RESAMPLE-MOVE filtering with cross-model jumps. In *Sequential Monte Carlo in Practice* (Eds A. Doucet, J. F. G. de Freitas and N. J. Gordon), New York. Springer-Verlag.

Berzuini, C., Best, N. G., Gilks, W. R. and Larizza, C. (1996) Dynamic conditional independence models and Markov chain Monte Carlo methods. *Journal of the American Statistical Association*, **92**, 1403–1412.

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

Carpenter, J. R., Clifford, P. and Fearnhead, P. (1996) Sampling strategies for Monte Carlo filters of non-linear systems. In *IEE Colloquium Digest, Target Tracking and Control, DERA, Malvern* (Ed. D. P. Atherton), volume 6, 253, pp. 1–3, London.

Chopin, N. (2000) A sequential particle filter for static models. Technical report, Laboratoire de Statistique, CREST, INSEE, Paris. available from http://www.statslab.cam.ac.uk/~mcmc/pages/listam.html.

Crisan, D. and Doucet, A. (2000) Convergence of sequential Monte Carlo methods. Technical report, Signal Processing Group, Cambridge University. available from http://www-sigproc.eng.cam.ac.uk/~ad2/arnaud_doucet.html.

Del Moral, P. and Guionnet, A. (1999) Central limit theorem for nonlinear filtering and interacting particle systems. *The Annals of Applied Probability*, **9**, 275–297.

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

Fearnhead, P. (1998) *Sequential Monte Carlo methods in filter theory*. Ph.D. thesis, Oxford Unversity. Available from http://www.stats.ox.ac.uk/~fhead/index.html.

Gilks, W. R. and Berzuini, C. (2001) Following a moving target - Monte Carlo inference for dynamic Bayesian models. *Journal of the Royal Statistical Society, Series B*, **63**.

Gilks, W. R., Richardson, S. and Spiegelhalter, D. J. (1996) *Markov chain Monte Carlo in practice*. London: Chapman and Hall.

Gordon, N., Salmond, D. and Smith, A. F. M. (1993) Novel approach to nonlinear/non-Gaussian Bayesian state estimation. *IEE proceedings-F*, **140**, 107–113.

Hurzeler, M. and Kunsch, H. R. (1998) Monte Carlo approximations for general state-space models. *Journal of Computational and Graphical Statistics*, **7**, 175–193.

Kitagawa, G. (1996) Monte Carlo filter and smoother for non-gaussian nonlinear state space models. *Journal of Computational and Graphical Statistics*, **5**, 1–25.

LeCadre, J. P. and Tremois, O. (1998) Bearings-only tracking for maneuvering sources. *IEEE Transactions on Aerospace and Electronice Systems*, **34**, 179–193.

Liu, J. S. and Chen, R. (1995) Blind deconvolution via sequential imputations. *Journal of the American Statistical Association*, **90**, 567–576.

Liu, J. S. and Chen, R. (1998) Sequential Monte Carlo methods for dynamic systems. *Journal of the American Statistical Association.*, **93**, 1032–1044.

Peach, N. (1995) Bearings-only tracking using a set of range-parameterized extended Kalman filters. *IEE Procedings-Control Theory and Applications*, **142**, 73–80.

Pitt, M. K. and Shephard, N. (1998) Time-varying covariances: a factor stochastic volatility approach. In *Bayesian Statistics 6* (Eds J. M. Bernardo, J. O. Berger, A. P. Dawid and A. F. M. Smith), Oxford. Oxford University Press.

Pitt, M. K. and Shephard, N. (1999) Filtering via simulation: auxiliary particle filters. *Journal of the American Statistical Association*, **94**, 590–599.

Shephard, N. (1996) Statistical aspects of ARCH and stochastic volatility. In *Time Series Models in Econometrics, Finance and Other Fields* (Eds D. R. Cox, D. V. Hinkley and O. E. Barndorff-Nielsen), pp. 1–67, Chapman and Hall, London.

Shephard, N. and Pitt, M. K. (1998) Likelihood analysis of non-Gaussian measurement time series. *Biometrika*, **84**, 653–667.

Taylor, S. J. (1982) Financial returns modelled by the product of two stochastic processes - a study of the daily sugar prices 1961–1975. In *Time Series Analysis: Theory and Practice I* (Ed. O. D. Anderson). Amsterdam: North Holland.

Captions

Figure 1: A realisation from the bearings-only tracking model. The simulated trajectory is shown by the thick line, and the estimated trajectory obtained from the filtered estimates of the position of the target is shown by the thin full line. The true position of the target and the filtered estimate of the position of the target are shown every 10 time steps (by circles and triangles respectively). Error bounds for the estimates of the range (± 1 standard deviation) are shown every 20 time steps by the dashed lines. The cross marks the position of the observer.

Figure 2: The posterior standard deviation for ν at each time-point (full line); and a fitted curve, proportional to $(t+1)^{-1/2}$ (dotted line).

Table 1: Comparison of different particle filter algorithms. The number of particles used in each filter was chosen so that one iteration of the algorithm took one second. Effective sample sizes for estimating the range r, and the system noise standard deviation ν are given. The larger the effective sample size, the more accurate the estimate: doubling the number of particles in a filter will (roughly) double the filters' ESS for any parameter.

Filter	Particles	ESS for r at time				ESS for ν at time			
		10	20	50	100	10	20	50	100
no MCMC:									
SIR	150,000	767	320	36.0	19.4	1,650	488	37.1	15.9
ISS	150,000	9,930	3,630	246	119	17,500	4,980	262	96.6
ASIR	95,000	6,400	2,220	177	83.6	11,600	3.070	188	68.3
with MCMC:									
SIR	110,000	1,700	438	378	277	5,360	1,030	90.0	48.1
ISS	110,000	14,800	3,840	1,460	1,000	19,500	5,520	438	189
ASIR	65,000	8,310	2,540	867	621	11,200	3,330	269	120

Table 1:

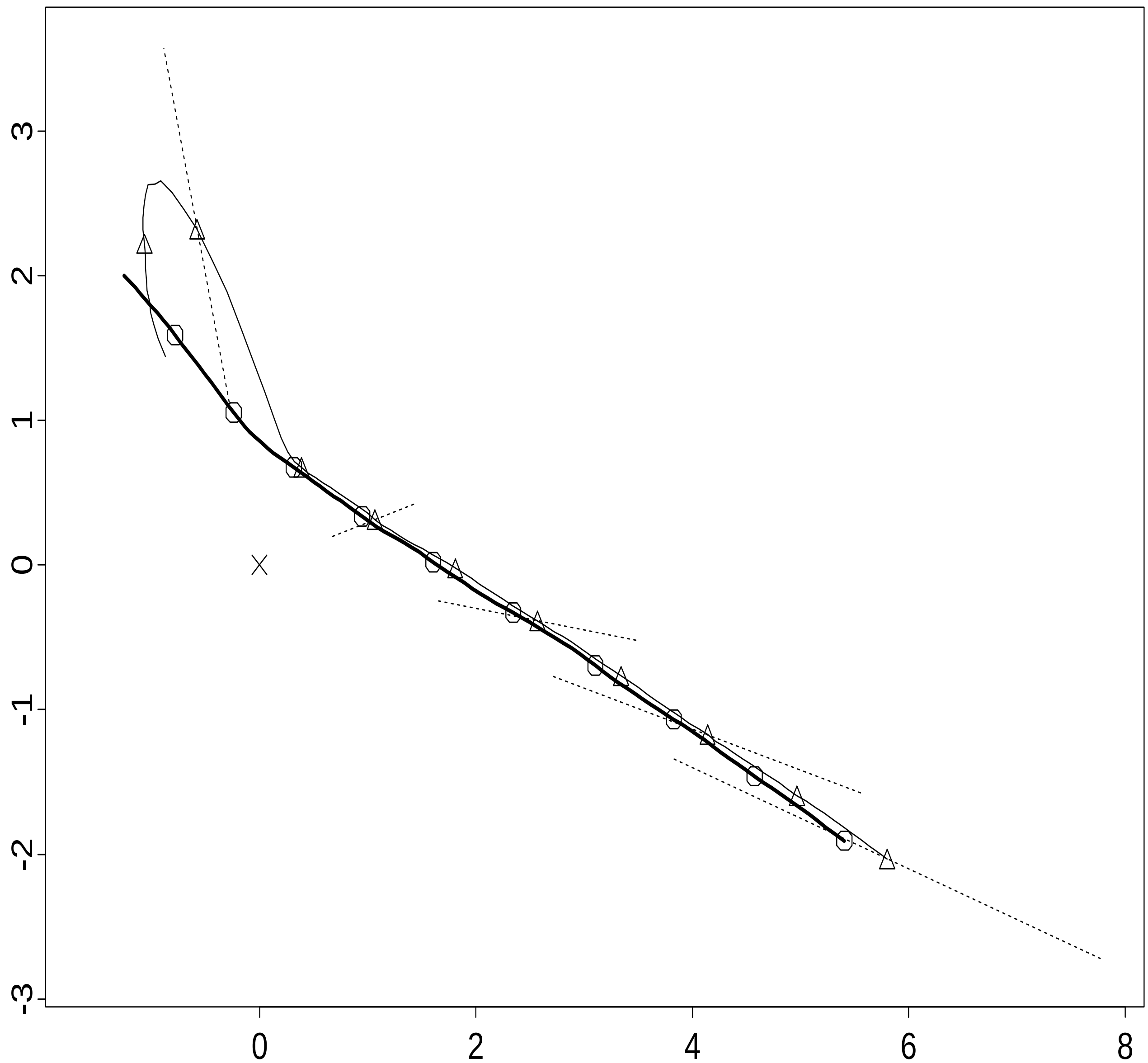

Figure 1:

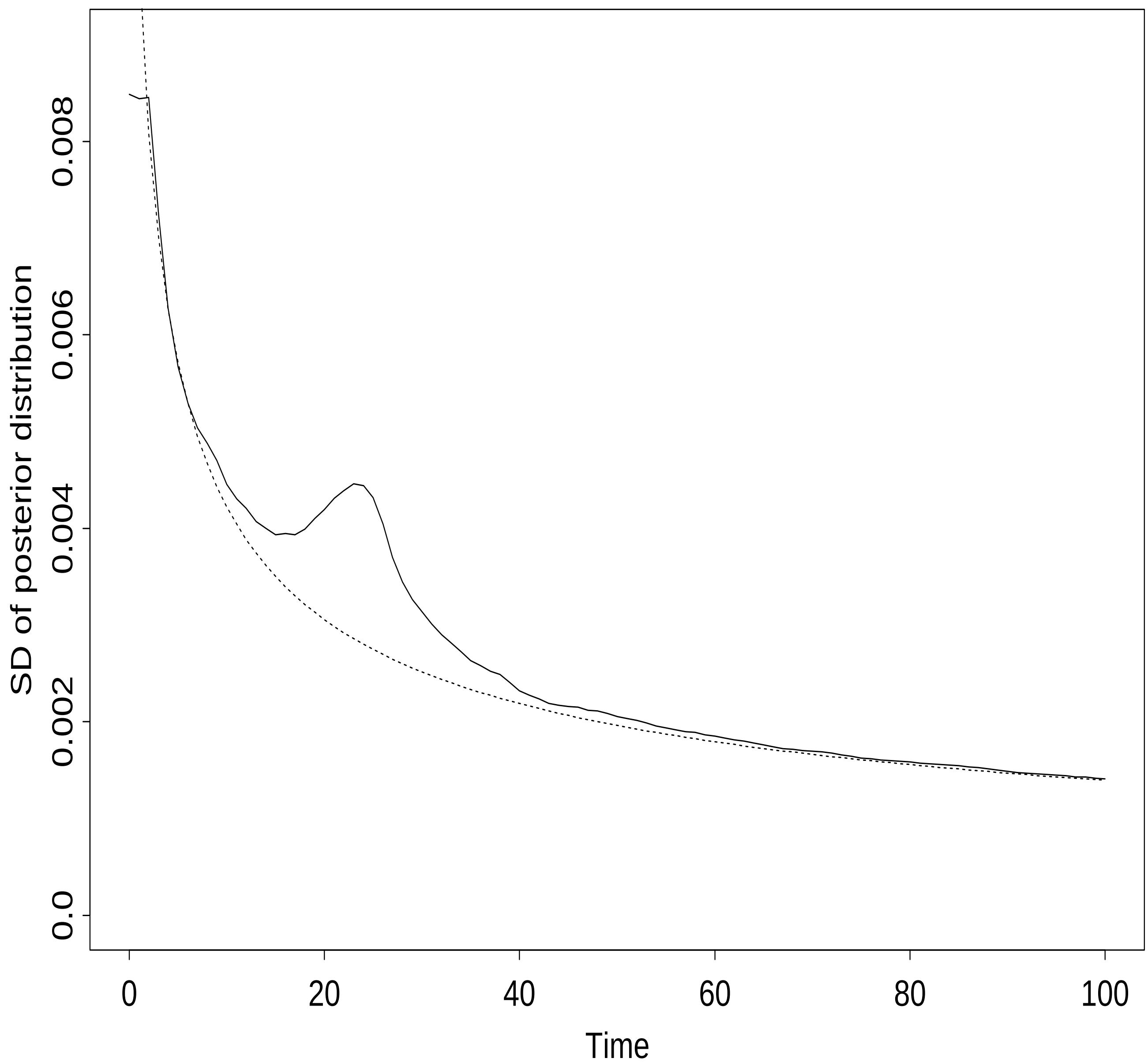
SD of posterior distribution
0.0
0.002
0.004
0.006
0.008
0
20
40
60
80
100
Time

Figure 2: