Monte Carlo Filters and Its Applications in Target Tracking and Wireless Communications [1]

Rong Chen
Department of Information and Decision Science,
University of Illinois at Chicago,
Chicago, IL 60607.
rongchen@uic.edu

1 Introduction

Stochastic systems are routinely used in science, engineering and economics. Many of these systems have a natural dynamic structure; others can often be built up dynamically. Except for a few special cases such as the linear Gaussian models or the discrete hidden Markov models, statistical analysis of these systems still present major challenges to researchers. The Monte Carlo filter (or sequential Monte Carlo) approach recently emerged in the fields of statistics and engineering shows a great promise in solving a large class of nonlinear filtering/prediction problems and general optimization problems, opening up new frontiers for cross-fertilization between statistical science and a wide spectrum of application areas such as telecommunications, bioinfomatics, and business data analysis.

Monte Carlo Filters (MCF) can be loosely defined as a family of methodologies that use Monte Carlo simulation to solve *on-line* estimation and prediction problems in dynamic systems. By recursively generating Monte Carlo samples of the state variables or some other latent variables, these methods can easily adapt to the dynamics of the underlying stochastic systems. Although the basic principle behind MCF dates back to the "growth Monte Carlo" method (Rosenbluth & Rosenbluth, 1955) known in molecular physics in the 50's, a complete theoretical framework for the MCF only appeared recently (Liu & Chen, 1998).

Efficient MCF has been designed for a number of problems including blind deconvolution, target tracking problems, and digital signal extraction in fading channels. It has also been shown that MCF complements MCMC and the two can be fruitfully combined (MacEachern et al. 1999). It has been used successfully in many problems such as energy minimization in molecular modeling and combinatorial optimization (Grassberger 1997; Wong & Liang 1997), speech recognition (Rabiner, 1989), target tracking problem (Gordon et al. 1993, Avitzour 1995), computer vision (Isard and Blake, 1996), economical time series (Pitt & Shephard, 1997); and DNA and protein sequence

[1]This work was supported in part by the U.S. National Science Foundation (NSF) under grants CCR-9980599 and DMS-0073601.

analysis (Churchill 1989; Krough et al. 1994; Liu et al. 1999), probabilistic expert systems (Spiegelhalter & Lauritzen 1990, Kong et al. 1994), simulating protein structures (Vasquez & Scherago 1985) and genetics (Irwing et al. 1994).

In section 2 we introduce the general stochastic dynamic systems, and one of its most commonly seen forms – the state space model. Several applications including target tracking and wireless communications will be introduced. In section 3 we detail the framework of MCF and its implementation issues. Section 4 contains an efficient variation of the MCF, the mixture Kalman filter. In section 5 we present several approaches to improve the efficiency of the MCF. Application examples are presented in Section 6.

2 Stochastic Dynamic Systems

A *stochastic dynamic system* (SDS) is defined as a sequence of evolving probability distributions, $\pi_t(\boldsymbol{x}_t)$, $t = 0, 1, 2, \ldots$, where the dimensionality of the state variable $\boldsymbol{x}_t$ often increases as the system evolves, i.e., $\boldsymbol{x}_t = (\boldsymbol{x}_{t-1}, x_t)$. In many scientific problems, it is of interest to evaluate expectations of a function of the state variable with respect to π_t at any time t.

Consider the *generalized state space model* of the form

$$\begin{array}{rlll} \text{(state equation):} & x_t = \phi_t(\boldsymbol{x}_{t-1}, \varepsilon_t) & \text{or} & x_t \sim f_t(\cdot \mid \boldsymbol{x}_{t-1}) \\ \text{(observation equation):} & y_t = \gamma_t(\boldsymbol{x}_t, e_t) & \text{or} & y_t \sim g_t(\cdot \mid \boldsymbol{x}_t), \end{array} \tag{1}$$

where $\boldsymbol{x}_t = (x_1, \ldots, x_t)$ is the unobserved state variable and $\boldsymbol{y}_t = (y_1, \ldots, y_t)$ is the observed information available up to time t, and ε_t and e_t are noises which lead to the conditional densities f_t and g_t. When x_t is discrete and f_t is Markovian, the model is often termed the *hidden Markov Model* and is widely used in speech recognition and biological sequence analysis. When x_t is continuous, the model is widely used in engineering and time series analysis. This model can also take the form of a "dynamic Bayesian network" (Boyen & Koller, 1998). Important problems with many of these systems are (a) the on-line estimation of the "true state characteristics," e.g. $E[h(x_t) \mid \boldsymbol{y}_t]$ (b) the prediction of a future behavior, e.g. $E[h(x_{t+1}) \mid \boldsymbol{y}_t]$, and (c) the revision of previous states when given new information (smoothing), e.g. $E[h(x_{t-s}) \mid \boldsymbol{y}_t]$. Thus, because of the Bayes theorem, one is most interested in the computation of expectations with respect to the *a posteriori* SDS, $\pi_t(\boldsymbol{x}_t) = p(\boldsymbol{x}_t \mid \boldsymbol{y}_t)$, a task that often eludes all theoretical attempts.

The state space modeling has a long history and the list of its applications is endless. When the system is linear and Gaussian, the posterior distribution $p(x_t \mid \boldsymbol{y}_t)$ is Gaussian and can be obtained recursively through the Kalman filter; when x_t only takes on a few discrete values, the problems can also be solved by a forward-backward approach. Otherwise, either a suboptimal solution or a crude approximation to the Bayes estimator has to be employed. Previously proposed methods

include the extended Kalman filer (e.g. Anderson & Moore 1979, Gelb 1974), Gaussian sum filters (Anderson & Moore 1979), the iterated extended Kalman Filter (Jazwinski 1970) and many others. Recently, researchers in statistics, engineering, and AI communities began to turn their attention to Monte Carlo-based filtering algorithms.

2.1 Target Tracking

Designing a sophisticated target tracking algorithm is an important task for both civilian and military surveillance systems, particularly when a radar, sonar, or optical sensor is operated in the present of clutter or when innovations are non-Gaussian (Bar-Shalom and Fortmann, 1988). We show three examples of target tracking using the MKF: (a) targets in the presence of random interference (clutter); (b) targets with non-Gaussian innovations; and (c) targets with maneuvering.

2.1.1 Random (Gaussian) accelerated target in clutter

Suppose the target follows a linear and Gaussian state space model:

$$\left\{ \begin{array}{rcl} x_t & = & Hx_{t-1} + Ww_t \\ y_t & = & Gx_t + Vv_t \end{array} \right. \tag{2}$$

where x_t is the state variable (location and velocity) of the target and w_t, v_t are white noises with identity covariance matrix. For targets moving on a straight line, we have $x_t = (s_t, \nu_t)$ where s_t is the true target location and ν_t is its current velocity. In this case

$$H = \left(\begin{array}{cc} 1 & T \\ 0 & T \end{array} \right); \quad W = \sigma_w^2 \left(\begin{array}{c} T/2 \\ 1 \end{array} \right); \quad G = (1, 0) \quad \text{and} \quad V = \sigma_v^2, \tag{3}$$

where T is the time duration between two observations and the random acceleration is assumed to be constant in the period, with rate $\sigma_w^2 w_t/T$. For targets moving in two (three) dimensional space, the state variable becomes $x_t = (\boldsymbol{s}_t, \boldsymbol{v}_t)$ with $\boldsymbol{s}_t$ and $\boldsymbol{v}_t$ being two (three) dimensional vectors. The corresponding matrixes can be expanded similarly.

In a clutter environment, we observe m_t signals $\{z_{t1}, \ldots, z_{tm_t}\}$ at time t, with

$$m_t \sim \text{Bernoulli}(p_d) + \text{Poisson}(\lambda\Delta),$$

where p_d is the probability of a true signal y_t being detected, λ is the rate of a Poisson random field, and Δ is the area of the surveillance region. In words, at time t the true signal has probability p_d to be detected, together with false signals, such as deceiving objects, electro-magnetic interferences, etc., which are distributed as a Poisson point process in the space. The problem is to track the real target on line in real time.

2.1.2 Random (Non-Gaussian) accelerated target in a clean environment

Consider again model (2), but with non-Gaussian errors w_t and v_t. Here we analyze the case when $w_t \sim t_{k_1}$ and $v_t \sim t_{k_2}$, but the same approach can be applied to other mixed-Gaussian settings. By defining $\Lambda_t = (\Lambda_{t1}, \Lambda_{t2})$, where $\Lambda_{ti} \sim \chi^2_{k_i}$ independently, we can rewrite model (2) as:

$$\begin{cases} x_t &= Hx_{t-1} + (\sqrt{k_1}/\sqrt{\lambda_1})We_t \\ y_t &= Gx_t + (\sqrt{k_2}/\sqrt{\lambda_2})V\varepsilon_t \end{cases} \quad \text{if} \quad (\Lambda_{t1}, \Lambda_{t2}) = (\lambda_1, \lambda_2)$$

with $e_t \sim N(0, I)$ and $\varepsilon_t \sim N(0, I)$. Again, it is a nonlinear/nonGaussian state space model.

2.1.3 Maneuvered target in a clean environment:

This situation can be modeled as follows:

$$\begin{aligned} x_t &= Hx_{t-1} + Fu_t + Ww_t \\ y_t &= Gx_t + Vv_t \end{aligned}$$

where u_t is the maneuvering acceleration. Here we consider an example of Bar-Shalom and Fortmann (1988) in which a two-dimensional target's position is sampled every $T = 10s$. The target moves in a plane with constant course and speed until $k = 40$ when it starts a slow 90^o turn which is completed in 20 sampling periods. A second, fast, 90^o turn starts at $k = 61$ and is completed in 5 sampling times. Figure 1 shows the trajectory of the target and its x-direction and y-direction velocity in one simulated run. Consequently, the matrices in this example are

$$H = \begin{pmatrix} 1 & 0 & 10 & 0 \\ 0 & 1 & 0 & 10 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}; G = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}; F = \begin{pmatrix} 5 & 0 \\ 0 & 5 \\ 1 & 0 \\ 0 & 1 \end{pmatrix}; W = \sigma_w^2 \begin{pmatrix} 5 & 0 \\ 0 & 5 \\ 1 & 0 \\ 0 & 1 \end{pmatrix}; V = \sigma_v^2 \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix};$$

The slow turn is the result of acceleration inputs $u_t^x = u_t^y = 0.075$ $(40 < t \leq 60)$, and the fast turn is from $u_t^x = -u_t^y = -0.3$ $(61 < t \leq 65)$. Other u_t's are zero (i.e. no maneuvering). We will analyze this example in later sessions.

2.2 Fading Channel in Wireless Communications

Many mobile communication channels can be modeled as Rayleigh flat-fading channels, which have the following form:

$$\text{State Equations:} \quad \begin{cases} \boldsymbol{x}_t &= F\boldsymbol{x}_{t-1} + Ww_t \\ \alpha_t &= G\boldsymbol{x}_t \\ s_t &\sim p(\cdot \mid s_{t-1}) \end{cases}$$

$$\text{Observation Equation:} \quad y_t = \alpha_t s_t + Vv_t$$

where s_t are the input digital signals (symbols), y_t are the received complex signals, and α_t are the unobserved (changing) fading coefficients. Both w_t and v_t are complex Gaussian with identity covariance matrices. Given the input signals s_t, the system is linear in $\boldsymbol{x}_t$ and y_t.

3 The General Form of Sequential Monte Carlo

One of the key components of MCF is importance sampling. Suppose a set of Monte Carlo samples, $\{\boldsymbol{x}^{(j)}, j = 1, \ldots, m\}$, has been generated from a trial distribution q. In order to compute the expectation, $\mu = E_\pi h(\boldsymbol{x})$, say, under distribution π, we can make an 'adjustment' by giving $\boldsymbol{x}^{(j)}$ a weight $w^{(j)} \propto \pi(\boldsymbol{x}^{(j)})/q(\boldsymbol{x}^{(j)})$. Then μ can be estimated by a weighted average of the $h(\boldsymbol{x}^{(j)})$. Since the weights are independent of function $h(\cdot)$, in a practical sense we can think of π as being approximated by a discrete distribution supported on the $\boldsymbol{x}^{(j)}$ with probabilities proportional to $w^{(j)}$. Based on this weighted-sample principle, Liu and Chen (1998), formulated the following MCF framework for applying Monte Carlo methods to a SDS.

Suppose $\{\pi_t(\boldsymbol{x}_t), t = 0, 1, \ldots\}$ is the SDS of interest. Let $S_t = \{\boldsymbol{x}_t^{(j)}, j = 1, \ldots, m\}$ be a set of Monte Carlo samples at time t with a corresponding set of weights $W_t = \{w_t^{(j)}, j = 1, \ldots, m\}$. We call $\{\boldsymbol{x}_t^{(j)}, w_t^{(j)}\}_{j=1}^m$ a properly weighted sample of π_t if $E_{\pi_t} h(\boldsymbol{x}_t)$, for any $h(\cdot)$, can be estimated by a weighted average of the $\boldsymbol{x}_t^{(j)}$ using the $w_t^{(j)}$. When the system evolves from stage t to $t+1$, we can produce a properly weighted sample of π_{t+1} by the following *Sequential Importance Sampling (SIS)* step:

(A) For each j, $j = 1, \ldots, m$, generate a $x_{t+1}^{(j)}$ (or multiple of them) from a trial distribution $q_{t+1}(x_{t+1} \mid \boldsymbol{x}_t^{(j)})$; attach it to $\boldsymbol{x}_t^{(j)}$ to form $\boldsymbol{x}_{t+1}^{(j)} = (\boldsymbol{x}_t^{(j)}, x_{t+1}^{(j)})$.

(B) Compute the "incremental weight"

$$u_{t+1}^{(j)} = \frac{\pi_{t+1}(\boldsymbol{x}_{t+1}^{(j)})}{\pi_t(\boldsymbol{x}_t^{(j)}) q_{t+1}(x_{t+1}^{(j)} \mid \boldsymbol{x}_t^{(j)})}; \quad \text{and let} \quad w_{t+1}^{(j)} = u_{t+1}^{(j)} w_t^{(j)}. \tag{4}$$

MCF is achieved by recursively applying the SIS step to a SDS. For a Markovian state space model, SIS only needs to keep a record of $x_t^{(j)}$, instead of the whole path $\boldsymbol{x}_t^{(j)}$, when proceeding from t to $t+1$. In a simplest form of MCF for this case, the "bootstrap filter" (or "particle filter") (Gorden et al. 1993; Kitagawa, 1996), one starts with equally weighted samples at time t and uses the state equation, $f_{t+1}(\cdot \mid x_t)$ as a trial distribution. The incremental weight is then proportional to the likelihood of the new observation, i.e., $g_{t+1}(\boldsymbol{y}_{t+1} \mid x_{t+1}^{(j)})$. Finally, one can resample so as to obtain a set of equally weighted sample for time $t+1$. In many applications, however, a more careful design of the MCF procedure can yield a much improved algorithm. The following issues are important for the design of MCF procedures.

(1). Trial sampling distribution: Choosing good trial distributions q_t is the critical first step in designing a good MCF scheme. Kong et al. (1994) and Liu & Chen (1995, 1998) suggested using $q_{t+1}(x_{t+1} \mid \boldsymbol{x}_t) = \pi_{t+1}(x_{t+1} \mid \boldsymbol{x}_t)$. In the state-space model, this is just the "local posterior distribution"

$$q_{t+1}(x_{t+1} \mid \boldsymbol{x}_t, y_{t+1}) = p(x_{t+1} \mid \boldsymbol{x}_t, y_{t+1}) \propto f_{t+1}(y_{t+1} \mid x_{t+1}) g_{t+1}(x_{t+1} \mid \boldsymbol{x}_t) \tag{5}$$

with incremental weight $u_{t+1} = \int f_{t+1}(y_{t+1} \mid \boldsymbol{x}_{t+1}) g_{t+1}(x_{t+1} \mid \boldsymbol{x}_t) dx_{t+1}$. This, if achievable, is apparently better than the one used by the bootstrap filter. More sophisticated choices of q_t will be discussed in later sections. When sampling from (5) is infeasible, methods such MCMC or Gaussian approximation can be used. See Liu & Chen (1998) for a summary.

(2). Resampling: Resampling is an indispensable component of MCF. Suppose $S_t = \{\boldsymbol{x}_t^{(j)}, j = 1, \ldots, m\}$ is properly weighted by $W_t = \{w_t^{(j)}, j = 1, \ldots m\}$ with respect to π_t. Instead of carrying the weight W_t as the system evolves, it is legitimate, and sometimes preferable (Liu & Chen 1995), to insert a resampling/reallocation step between SIS recursions in order to stabilize the weight distribution. The following scheme is typical: (i) draw (or systematically allocate) a new set of samples (denoted as S_t') from S_t with probability proportional to $w_t^{(j)}$; and (ii) assign equal weights to the samples in S_t'. Some theoretical and heuristic arguments of resampling are given in Liu & Chen (1995). Note that if the weights $w_t^{(j)}$ are nearly constant, resampling only reduces the number of distinctive samples and incurs extra Monte Carlo variation. However, when the weights become very skewed, carrying many samples with very small weights in an SIS setting is apparently wasteful. Resampling can provide chances for the good (i.e., "important") samples to amplify themselves and hence "rejuvenate" the sampler to produce better samples for the future states.

(3). Marginalization: When implementing Monte Carlo strategies, it is often a good practice to carry out as much analytical computation as possible (Hammersley & Hanscomb 1965). In importance sampling, it can be easily shown that the algorithm is more efficient after some components of the system are integrated out (marginalization). Liu et al. (1994) show that marginalization is also beneficial for Gibbs sampling. MacEachern et al. (1999) demonstrate that marginalization can greatly improve a MCF algorithm in a nonparametric Bayes problem.

4 The Mixture Kalman Filter

Many dynamic systems belong to the class of conditional dynamic linear models (CDLM) of the form

$$\boldsymbol{x}_t = \boldsymbol{F}_{\lambda_t} \boldsymbol{x}_{t-1} + \boldsymbol{G}_{\lambda_t} \boldsymbol{u}_t, \tag{6}$$

$$\boldsymbol{y}_t = \boldsymbol{H}_{\lambda_t} \boldsymbol{x}_t + \boldsymbol{K}_{\lambda_t} \boldsymbol{v}_t, \tag{7}$$

where $\boldsymbol{u}_t \sim N(0, \boldsymbol{I})$ and $\boldsymbol{v}_t \sim N(0, \boldsymbol{I})$ are the state and observation noise, respectively; and λ_t is a sequence of random indicator variables which may form a Markov chain, but are independent of $\boldsymbol{u}_t$ and $\boldsymbol{v}_t$ and the past $\boldsymbol{x}_s$ and $\boldsymbol{y}_s$, $s < t$. The matrices $\boldsymbol{F}_{\lambda_t}$, $\boldsymbol{G}_{\lambda_t}$, $\boldsymbol{H}_{\lambda_t}$ and $\boldsymbol{K}_{\lambda_t}$ are known given λ_t.

We observe that for a given trajectory of the indicator λ_t in a CDLM, the system is both linear and Gaussian, for which the Kalman filter provides a complete statistical characterization of the system dynamics. We proposed a novel sequential Monte Carlo method, the mixture Kalman filter (MKF) for on-line filtering and prediction of CDLM's; it exploits the conditional Gaussian property and utilizes a marginalization operation to improve the algorithmic efficiency. Instead of dealing with both $\boldsymbol{x}_t$ and λ_t, the MKF draws Monte Carlo samples only in the indicator space and uses a mixture of Gaussian distributions to approximate the target distribution. Compared with the generic sequential Monte Carlo method, the MKF is substantially more efficient (e.g., it produces more accurate results with the same computing resources).

Let $\boldsymbol{Y}_t = (\boldsymbol{y}_0, \boldsymbol{y}_1, \cdots, \boldsymbol{y}_t)$ and $\boldsymbol{\Lambda}_t = (\lambda_0, \lambda_1, \cdots, \lambda_t)$. By recursively generating a set of properly weighted random samples $\left\{ \left(\boldsymbol{\Lambda}_t^{(j)}, w_t^{(j)} \right) \right\}_{j=1}^m$ to represent $p(\boldsymbol{\Lambda}_t | \boldsymbol{Y}_t)$, the MKF approximates the target distribution $p(\boldsymbol{x}_t | \boldsymbol{Y}_t)$ by a random mixture of Gaussian distributions

$$\sum_{j=1}^{m} w_t^{(j)} \mathcal{N}_c \left(\boldsymbol{\mu}_t^{(j)}, \boldsymbol{\Sigma}_t^{(j)} \right),$$

where $\boldsymbol{\mu}_t^{(j)} = \boldsymbol{\mu}_t \left(\boldsymbol{\Lambda}_t^{(j)} \right)$ and $\boldsymbol{\Sigma}_t^{(j)} = \boldsymbol{\Sigma}_t \left(\boldsymbol{\Lambda}_t^{(j)} \right)$ are obtained with a Kalman filter on the system (6)-(7) for the given indicator trajectory $\boldsymbol{\Lambda}_t^{(j)}$. Denote $\kappa_t^{(j)} \triangleq \left[\boldsymbol{\mu}_t^{(j)}, \boldsymbol{\Sigma}_t^{(j)} \right]$. Thus, a key step in the MKF is the production at time t of the weighted samples of indicators, $\left\{ \left(\boldsymbol{\Lambda}_t^{(j)}, \kappa_t^{(j)}, w_t^{(j)} \right) \right\}_{j=1}^m$, based on the set of samples, $\left\{ \left(\boldsymbol{\Lambda}_{t-1}^{(j)}, \kappa_{t-1}^{(j)}, w_{t-1}^{(j)} \right) \right\}_{j=1}^m$, at the previous time $(t-1)$. For details, see Chen & Liu (2000).

5 Delayed-sampling Method

Dynamic systems often possess strong short term or long term "memory,", i.e., future observations can reveal substantial information on the current state. In the case of sudden noise spike and system malfunction, future observations become critical in combating temporary loss of information or mis-information. It is a good practice to have a "waiting" period, i.e., a buffer, in which information accumulates before being used for generating new Monte Carlo samples. However, a MCF scheme usually does not go back to "regenerate" past samples in view of new information (it is possible to design strategies for a MCF scheme to revise previous draws), although the past estimations can be adjusted by using the new importance weights. Hence, if the samples are generated by using false information (i.e. from a sampling distribution that is far away from the truth), MCF can quickly

loose track of the state variable. A natural method to overcome this difficulty is the *delayed-sample method*, which uses future information in generating sample of the current state.

To achieve this end for the state space model (1), we can define the target SDS as $\{\pi_t(\boldsymbol{x}_t) = p(\boldsymbol{x}_t \mid \boldsymbol{y}_{t+\Delta}), t = 1, 2, \ldots\}$ for some $\Delta \geq 0$. Suppose at time $t+\Delta$ we have a set of properly weighted samples $\{(\boldsymbol{x}_t^{(j)}, w_t^{(j)})\}_{j=1}^m$ of the new SDS. Then at time $t+\Delta+1$, we can use $q_{t+1}(x_{t+1}^{(j)} \mid \boldsymbol{x}_t^{(j)}, \boldsymbol{y}_{t+\Delta+1}) = p(x_{t+1}^{(j)} \mid \boldsymbol{x}_t^{(j)}, \boldsymbol{y}_{t+\Delta+1})$, to generate Monte Carlo samples of x_{t+1}. This involves computing

$$p(x_{t+1} \mid \boldsymbol{x}_t, \boldsymbol{y}_{t+\Delta+1}) \propto \int p(\boldsymbol{x}_{t+\Delta+1}, \boldsymbol{y}_{t+\Delta+1}) dx_{t+2} \cdots dx_{t+\Delta+1}. \tag{8}$$

The weight can be computed as $w_{t+1}^{(j)} \propto w_{t-1} p(y_{t+\Delta+1}, \boldsymbol{y}_{t+\Delta} \mid \boldsymbol{x}_t^{(j)}) / p(\boldsymbol{y}_{t+\Delta} \mid \boldsymbol{x}_t^{(j)})$. A main difficulty with this approach is that one needs to evaluate the multidimensional integrals such as (8). Here we discuss two approaches: the straightforward exact evaluation approach and a pilot evaluation approach.

(1). Exact Sampling and Evaluation: When the state variable takes values in a discrete set $\mathcal{A} = \{a_1 \ldots, a_J\}$ (e.g., when the state variable represents the transmitted signal in digital communications), the aforementioned integration can be done by exhausting all possible combinations of the $x_{t+1}, \ldots, x_{t+\Delta+1}$ string, which involves growing/trimming a tree with $J^{\Delta+1}$ branches. Specifically, the sampling distribution is

$$p(x_{t+1} \mid \boldsymbol{x}_t^{(j)}, \boldsymbol{y}_{t+\Delta+1}) \propto \sum_{x_{t+1}^{t+\Delta+1} \in J^{\Delta+1}} \prod_{d=1}^{\Delta+1} p(y_{t+d} \mid \boldsymbol{y}_{t+d-1}, x_{t+1}^{t+d}, \boldsymbol{x}_t^{(j)})$$

and the weight is

$$w_{t+1} = w_t \frac{\sum_{x_{t+1}^{t+\Delta+1} \in J^{\Delta+1}} \prod_{d=1}^{\Delta+1} p(y_{t+d} \mid \boldsymbol{y}_{t+d-1}, x_{t+1}^{t+d}, \boldsymbol{x}_t^{(j)})}{\sum_{x_{t+1}^{t+\Delta} \in J^{\Delta}} \prod_{d=0}^{\Delta} p(y_{t+d} \mid \boldsymbol{y}_{t+d-1}, x_{t+1}^{t+d}, \boldsymbol{x}_t^{(j)})},$$

where $x_{t+1}^{t+d} = (x_{t+1}, \ldots, x_{t+d})$. This approach has the difficulty that the complexity increases exponentially with J and Δ.

(2). Pilot-delayed Sampling and Evaluation:

When the cardinality J of $\mathcal{A}$ is large or when the state variable is continuous, the previous approach is infeasible. However, (8) can be approximated by using a relative small number of pilot streams. For example, in the discrete case we can send out J groups of pilots, each with k members. Every individual in the j-th group starts with the same value a_j of x_{t+1}, and propagates Δ SIS steps to time $t+\Delta+1$. The sum of the weights of every pilot in j-th group, called "pilot weight for a_j", is given to value a_j. Finally we draw $x_{t+1} = a_j$ with probability proportional to the pilot weight. To correct the bias introduced by pilot approximation, we need to multiply the

usual incremental weight (4) by the inverse of the pilot weight. When the state variable consists of an "important" but simple (say, binary) component and an "unimportant" one, the pilot streams can be generated to cover all possible combinations of the "important" components but only cover a random portion of the "unimportant" part. When the state variable x_t is continuous, one can generate for each $\boldsymbol{x}_t^{(j)}$ a pilot group of x_{t+1} from $g_{t+1}(\cdot|\boldsymbol{x}_t^{(j)})$. Each individual in the group is then processed with SIS recursions to $t+\Delta+1$ and obtains a pilot weight. The weight of each individual member can then be used to resample a x_{t+1} from the starting pilot group. The final weight can be adjusted similarly.

6 Some Numerical Examples

In this section we provide some numerical results for the examples given in section 2.

6.1 Target tracking

6.1.1 Random (Gaussian) accelerated target in clutter

By letting Λ_t be the identifier of the target, Liu and Chen (1998) formulated the problem stated in section 2.1 into a CDLM. More precisely, they let $\Lambda_t = 0$ if the target is not observed, and $\Lambda_t = i$ if the i-th observed object is the signal generated from the true target, i.e., $y_t = z_{ti}$. Then the system is linear and Gaussian with given Λ_t, and the remaining z signals bear no information. Some of their results are shown in Figure 2 (a) and (b), which reveal the tracking errors (the differences between the estimated and true target locations) of 50 simulated runs of the tracking model, with $r^2 = 1.0, q^2 = 1.0, p_d = 0.9$ and $\lambda = 0.1$. Five hundred Monte Carlo samples were used for both the MKF and a standard Monte Carlo filter (i.e. an SIS with resampling applied to the state variable x_t). Here we also tested the split-track filter (Figure 2 (c)), which, at each step, kept 500 trajectories with the highest likelihood values (recursively). The MKF performed much better than the other two algorithms in this problem.

6.1.2 Random (Non-Gaussian) accelerated target in a clean environment

Simulations were carried out with the matrices (3) with $T = 1$ and no interference and $w_t \sim t_3$ and $v_t \sim t_3$. The following table shows a comparison of the MKF and a standard Monte Carlo filter in terms of the number of times the target was lost ($|x_t - \hat{x}_t| > 1200$) and the cpu time for one hundred simulated runs.

noise variance	MC size (m)	MC Filter		MKF	
		cpu time	# miss	cpu time	# miss
	20	9.49843	72	19.4277	1
	50	20.1622	20	51.6061	1
σ_w^2= 16.00	200	80.3340	7	181.751	1
σ_v^2= 1600	500	273.369	4	500.157	1
	1500	1063.36	3	2184.67	1

Figure 3 shows the tracking mean squared error, after the lost tracks are eliminated. We observe that although it takes about twice as much CPU time as the standard Monte Carlo filter with the same m, the MKF performs much more efficiently in the same CPU time.

We also tested the idea of using a finite mixture of Gaussian distributions to approximate the t distribution, i.e. approximating t_3 with $\sum_{i=1}^{k} p_i N(0, \sigma_i^2)$. Similar results were obtained. The advantage of this approach is that a more efficient MKF can be used for discrete indicators. But on the other hand, the approximation causes some biases.

6.1.3 Maneuvered target in a clean environment:

To apply the MKF to this application, we need to specify prior structure of u_t. First, we assume that maneuvering can be classified into several categories, indicated by an indicator. In particular, we assume a three level model, $I_t = 0$ indicates no maneuvering ($u_t = 0$), and $I_t = 1$ and 2 indicate slow and fast maneuvering, respectively, ($u_t \sim N(0, \sigma_i^2)$, $\sigma_1^2 < \sigma_2^2$). In this study we used $\sigma_1^2 = 1$ and $\sigma_2^2 = 36$. We also specify transition probabilities $P(I_t = j \mid I_{t-1} = i) = p_{ij}$ for the maneuvering status. Specifically, we assume $p_{ii} = 0.8$ and $p_{ij} = 0.1$ for $i \neq j$ (i.e. it is more likely to stay in a particular maneuvering state than to change the maneuvering state). Second, there are different ways of modeling the serial correlation of the u_t. Here we assume a multi-level white noise model, as in Bar-Shalom and Fortmann (1988), where the u_t are assumed independent, given the indicator. This is the easiest but not a very realistic model. Other possible models are currently under investigation.

In Figure 4 we present the root mean square errors of the MKF estimates of the target position for 50 simulated runs. Comparing our result with that of Bar-Shalom and Fortmann (1988, pp 143) who used the traditional detection-and-switching method, we see a clear advantage of the proposed MKF.

6.2 Digital Signal Extraction in Fading Channels

In this section, we provide some computer simulation examples to demonstrate the performance of the MCF in fading channels. The fading process is modeled by the output of a Butterworth filter

of order $r = 3$ driven by a complex white Gaussian noise process. The cutoff frequency of this filter is 0.05, corresponding to a normalized Doppler frequency (with respect to the symbol rate $\frac{1}{T}$) $f_dT = 0.05$, which is a fast fading scenario. Specifically, the fading coefficients $\{\alpha_t\}$ is modeled by the following ARMA(3,3) process:

$$\begin{aligned} & \alpha_t - 2.37409\alpha_{t-1} + 1.92936\alpha_{t-2} - 0.53208\alpha_{t-3} \\ = \;& 10^{-2}(0.89409u_t + 2.68227u_{t-1} + 2.68227u_{t-2} + 0.89409u_{t-3}), \end{aligned} \tag{9}$$

where $u_t \sim \mathcal{N}_c(0,1)$. The filter coefficients in (9) are chosen such that $\mathrm{Var}\{\alpha_t\} = 1$. It is assumed that BPSK modulation is employed, i.e., the transmitted symbols $s_t \in \{+1, -1\}$.

In order to demonstrate the high performance of the proposed adaptive receiver, in the following simulation examples we compare the performance (in terms of bit error rate) of the proposed sequential Monte Carlo receivers with that of the following three receiver schemes:

- *Known channel lower bound:* In this case, we assume that the fading coefficients $\{\alpha_t\}$ are known to the receiver.

- *Genie-aided lower bound:* In this case, we assume that a genie provides the receiver with an observation of the modulation-free channel coefficient corrupted by additive noise with the same variance, i.e., $\tilde{y}_t = \alpha_t + \tilde{n}_t$, where $\tilde{n}_t \sim \mathcal{N}_c(0, \sigma^2)$. The receiver then uses a Kalman filter to track the fading process based on the information provided by the genie; The transmitted symbols are then demodulated. It is clear that such a genie-aided bound is lower bounded by the known channel bound. It should also be noted that the genie is used only for calculating the lower bound. Our proposed algorithms estimate the channel and the symbols simultaneously with no help from the genie.

- *Differential detector:* In this case, no attempt is made to estimate the fading channel. Instead the receiver detects the phase difference in two consecutively transmitted bits by using the simple rule of differential detection: $\widehat{b_t b}_{t-1} = \mathrm{sign}\left(\Re\left\{y_t^* y_{t-1}\right\}\right)$.

The differential encoding and decoding are employed to resolve the phase ambiguity. The adaptive receiver implements the MKF algorithm described in Section 4. The number of Monte Carlo samples drawn at each time was empirically set as $m = 50$. Simulation results showed that the performance did not improve much when m was increased to 100, while it degraded notably when m was reduced to 20. The resampling procedure discussed in Section 3 was employed to maintain the efficiency of the algorithm, in which the effective sample size threshold is $\bar{m}_t = m/10$. The *delayed-weight* method discussed in Section 5 was used to extract further information from future received signals, which resulted in an improved performance compared with concurrent

estimation. In each simulation, the sequential Monte Carlo algorithm was run on 10000 symbols, (i.e., $t = 1, \cdots, 10000$). In counting the symbol detection errors, the first 50 symbols were discarded to allow the algorithm to reach the steady state. In Figure 5, the bit error rate (BER) performance versus the signal-to-noise ratio (defined as $\mathrm{Var}\{\alpha_t\}/\mathrm{Var}\{n_t\}$) corresponding to delay values $\delta = 0$ (concurrent estimate), $\delta = 1$, and $\delta = 2$ is plotted. In the same figure, we also plot the known channel lower bound, the genie-aided lower bound, and the BER curve of the differential detector. From this figure it is seen that, with only a small amount of delay the performance of the MKF can be significantly improved by the delayed-weight method compared with the concurrent estimate. Even with the concurrent estimate, the MKF does not exhibit an error floor, as does the differential detector. Moreover, with a delay δ=2, the MKF essentially achieves the genie-aided lower bound. We have also implemented the delayed-sample method for this case and found that it offers little improvement over the delayed-weight method.

REFERENCES

Ackerson, G.A. and Fu, K.S. (1970) On state estimation in switching environments. *IEEE Trans. Autom. Control*, **AC-15**, 10-17.

Akashi, H. and Kumamoto, H. (1977) Random sampling approach to state estimation in switching environments. *Automatica*, **13**, 429-434.

Anderson, B.D.O. and Moore, J.B. (1979). *Optimal Filtering.* New Jersey: Prentice-Hall.

Avitzour, D. (1995) A stochastic simulation Bayesian approach to multitarget tracking. *IEE Proc. Radar, Sonar and Navigation*, **142**, 41-44.

Bar-Shalom, Y. and Fortmann, T.E. (1988) *Tracking and Data Association*, Academic Press: Boston

Berzuini, C., Best, N.G., Gilks, W.R., and Larizza, C. (1997) Dynamic conditional independence models and Markov chain Monte Carlo methods. *J. Amer. Statist. Assoc.*, **92**, 1403-1412.

Boyen, X. and Koller, D. (1998). Tractable inference for complex stochastic processes. *Technical Report, Computer Sci. Dept., Stanford Univ.*

Carlin, B.P, Polson, N. G. and Stoffer, D. S. (1992) A Monte Carlo approach to nonnormal and nonlinear state-space modeling. *J. Amer. Statist. Assoc.*, **87** 493-500.

Carpenter, J., Clifford, P. and Fearnhead, P. (1997) An improved particle filter for non-linear problems. *Technical Report*, Oxford University.

Carter, C.K. and Kohn, R. (1994) On Gibbs sampling for state space models. *Biometrika*, **81**, 541-553.

Chen, R. and Liu, J.S. (2000). Mixture Kalman filters. *Journal of the Royal Statistical Society, Series B*, **62**, 493-508

Chen, R., Wang, X. and Liu, J.S. (2000). Adaptive joint detection and decoding in flat-fading channels via mixture Kalman filtering. *IEEE trans. Information Theory*, **46**, 2079-2094

Churchill, G.A. (1989), Stochastic Models for Heterogeneous DNA Sequences, *Bulletin of Mathematical Biology* **51**, 79-94.

Doucet, A., Godsill, S.J. and Andrieu, C. (1999) On sequential Monte Carlo sampling methods for Bayesian filtering. *Statist. Comput.*, in press.

Gelb, A. (1974) *Applied Optimal Estimation*. Boston: MIT press

Gordon, N.J., Salmon, D.J. and Smith, A.F.M. (1993) A novel approach to nonlinear/non Gaussian Bayesian state estimation. *IEE Proc. Radar Signal Process.*, **140**, 107-113.

Grassberger (1997). Pruned-enriched rosenbluth method: simulations of q polymers of chain length up to 1000000.. *Physical Review E*, **56**, 3682–3693.

Hammersley, J.M. and Handscomb, D.C. (1965). *Monte Carlo Methods*. Chapman & Hall.

Hürzeler, M. and Künsch, H.R. (1998) Monte Carlo approximations for general state-space models. *J. Comp. Graph. Statist.*, **7**, 175-193.

Irwing, M., Cox, N. and Kong, A. (1994). Sequential imputation for multilocus linkage analysis. *Proceedings of the National Academy of Science, USA*, **91**, 11684-11688.

Isard, M. and Blake, A. (1996). Contour tracking by stochastic propagation of conditional density. In *Computer Vision - ECCV' 96*, B. Buxton and R. Cipolla (eds), Springer: New York.

Jazwinski, A. (1970) *Stochastic Processess and Filtering Theory*. New York: Academic Press.

Kalman, R.E. (1960) A new approach to linear filtering and prediction problems. *J. Basic Engineering*, **82**, 35-45

Kitagawa, G. (1996) Monte Carlo filter and smoother for non-Gaussian nonlinear state space models. *J. Comp. Graph. Statist.*, **5**, 1-25.

Kong, A., Liu, J.S., and Wong, W.H. (1994) Sequential imputations and Bayesian missing data problems. *J. Amer. Statist. Assoc.*, **89**, 278-288.

Krogh A., Brown, M., Mian, I.S., Sjolander, K. and Haussler, D. (1994). Hidden Markov models in computational biology - Applications to protein modeling. *J. Mol. Biol.*, **235**, 1501-1531.

Liu, J.S. and Chen, R. (1995) Blind deconvolution via sequential imputations. *J. Amer. Statist. Assoc.*, **90**, 567-576.

— (1998) Sequential Monte Carlo methods for dynamic systems. *J. Amer. Statist. Assoc.*, **93**, 1032-1044

Liu, J.S., Chen, R., and Wong, W.H. (1998) Rejection control for sequential importance sampling. *J. Amer. Statist. Assoc.*, **93**, 1022-1031.

Liu, J.S., Wong, W.H., and Kong, A. (1994) Covariance structure of the Gibbs sampler with applications to the comparisons of estimators and augmentation schemes. *Biometrika*, **81**, 27-40.

MacEachern, S.N., Clyde, M.A., and Liu, J.S. (1999) Sequential importance sampling for non-parametric Bayes models: the next generation. *Canadian J. Statist.*, **27**, 251-267.

Pitt, M.K. and Shephard, N. (1999) Filtering via simulation: auxiliary particle filters. *J. Amer. Statist. Assoc.*, **94**, 590-599.

Rabiner, L.R. (1989), A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition,*Proceedings of the IEEE*, **77**, 257-286.

Rosenbluth, M.N. and Rosenbluth, A.W. (1955). Monte Carlo calculation of the average extension of molecular chains. *J. Chem. Phys.*, **23**, 356-359.

Rubin, D.B.(1987) A noniterative sampling/importance resampling alternative to the data augmentation algorithm for creating a few imputations when fractions of missing information are modest: the SIR algorithm. *J. Amer. Statist. Assoc.*, **52**, 543-546.

Rubinstein, R.Y. (1981) *Simulation and the Monte Carlo Method*. New York: Wiley.

Shephard, N. (1994) Partial non-Gaussian state space. *Biometrika*, **81**, 115-131.

Spiegelhalter, D.J. and Lauritzen, S.L. (1990), Sequential Updating of Conditional Probabilities on Directed Graphical Structures, *Network*, **20**, 579-605.

Smith, M.C. and Winter, E.M. (1978) On the detection of target trajectories in a multi-target environment. *Proc. 17th (1978) IEEE Conf. on Decision & Control.*, San Diego, CA.

Svetnik, V.B. (1986) Applying the Monte Carlo method for optimum estimation in systems with random structure. *Auto. Remo. Cont.*, **47**, 818-827.

Tanizaki, H. (1996) *Nonlinear Filters: Estimation and Applications*. New York: Springer.

Tugnait, J.K. (1982) Detection and estimation for abruptly changing systems. *Automatica*, **18**, 607-615.

West (1992) Mixture models, Monte Carlo, Bayesian updating and dynamic models. *Computer Science and Statistics*, **24**, 325-333.

West, M. and Harrison, J. (1989) *Bayesian Forecasting and Dynamic Models*. New York: Springer.

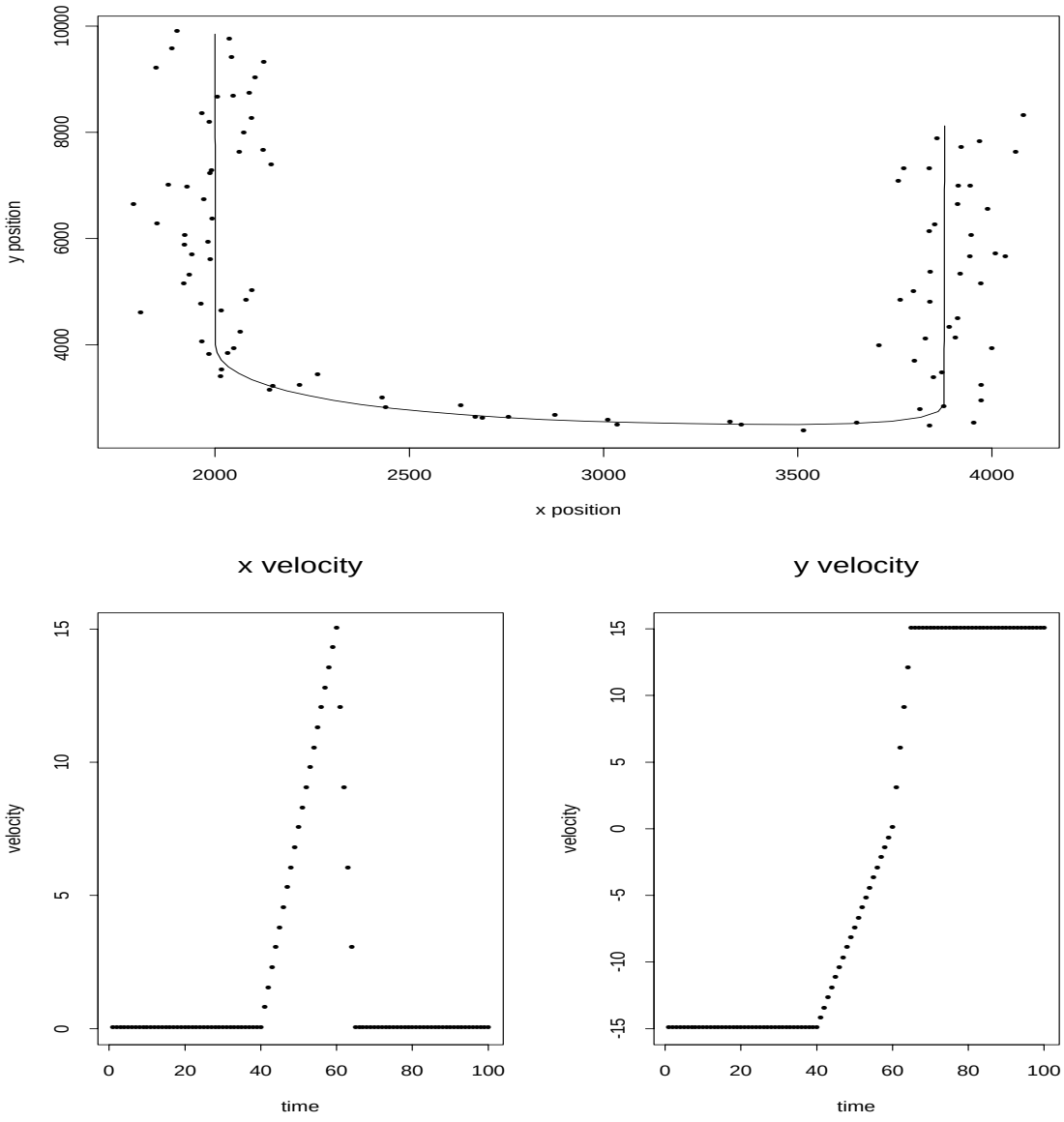
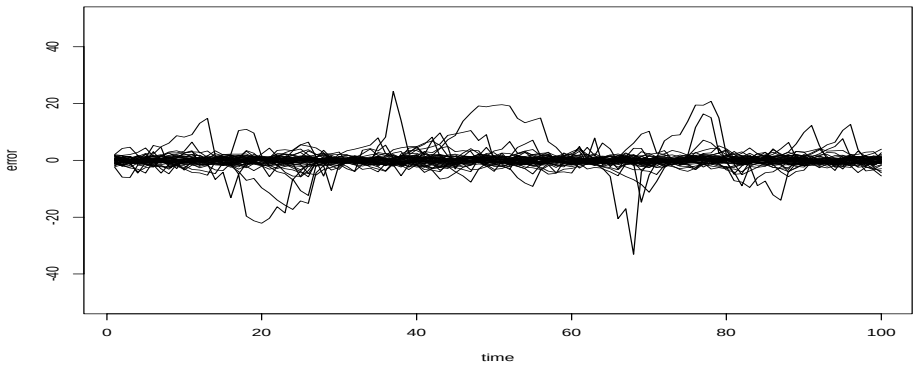
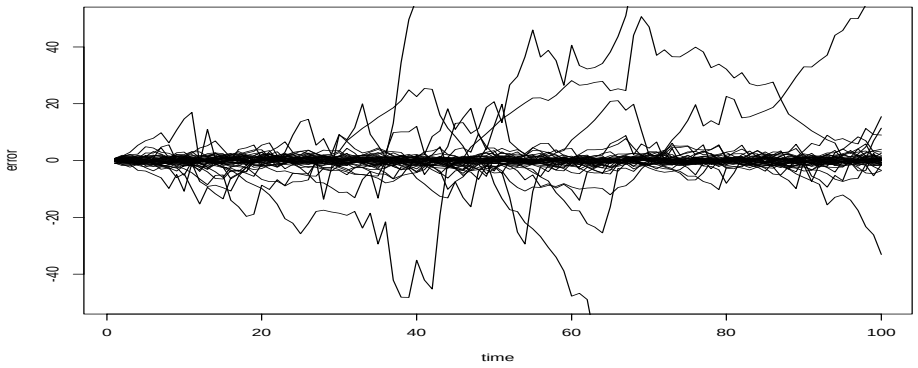


Figure 1: The position and velocity of a simulated 2 dimensional maneuvering target. (Top) Position. (Bottom) Velocity.

(a)



(b)



(c)

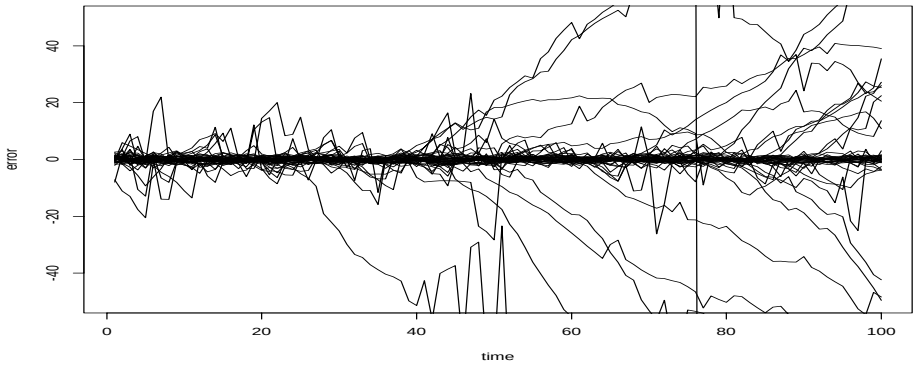


Figure 2: The tracking errors of 50 runs of the MKF (a), a standard Monte Carlo filter (b), and the split-track filter (c) for a simulated one-dimensional target moving system.

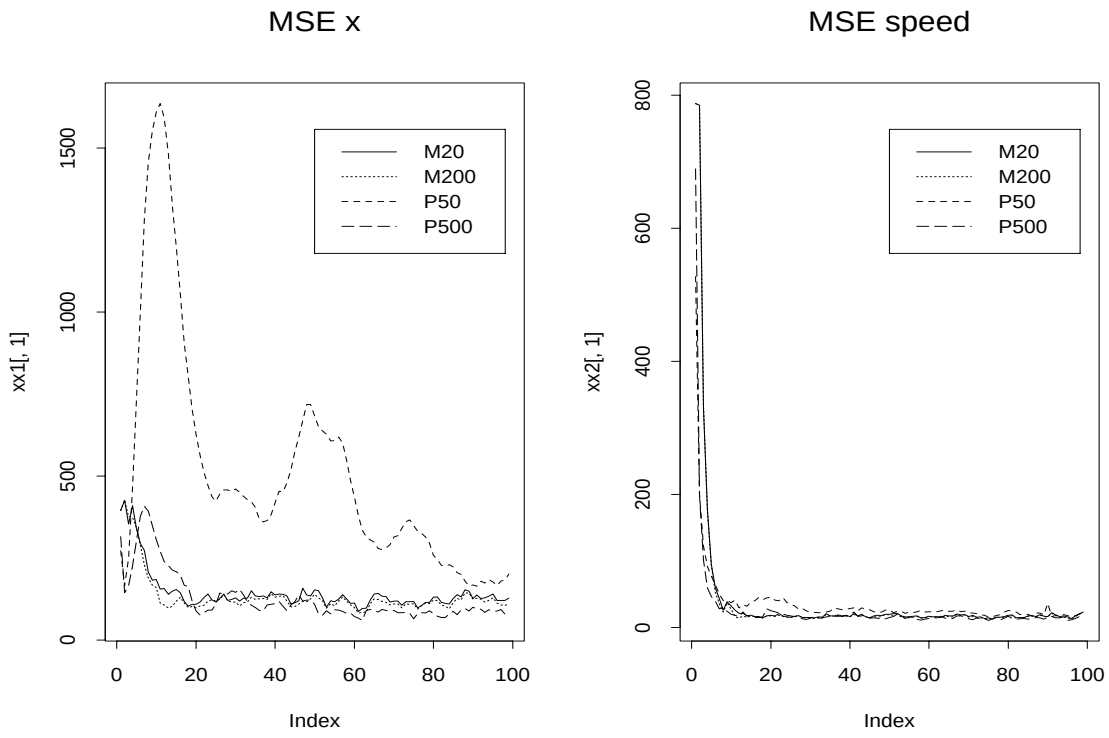


Figure 3: The MSE's of location and speed of 50 runs of the MKF and a standard MC filter for a simulated one-dimensional target moving system using different Monte Carlo sample sizes. 'MKF20' and 'MKF200' are the MSE's of the MKF with Monte Carlo sample size 20 and 200 respectively. 'MC50' and 'MC500' are the MSE's of a standard MC filter with Monte Carlo sample size 50 and 500 respectively.

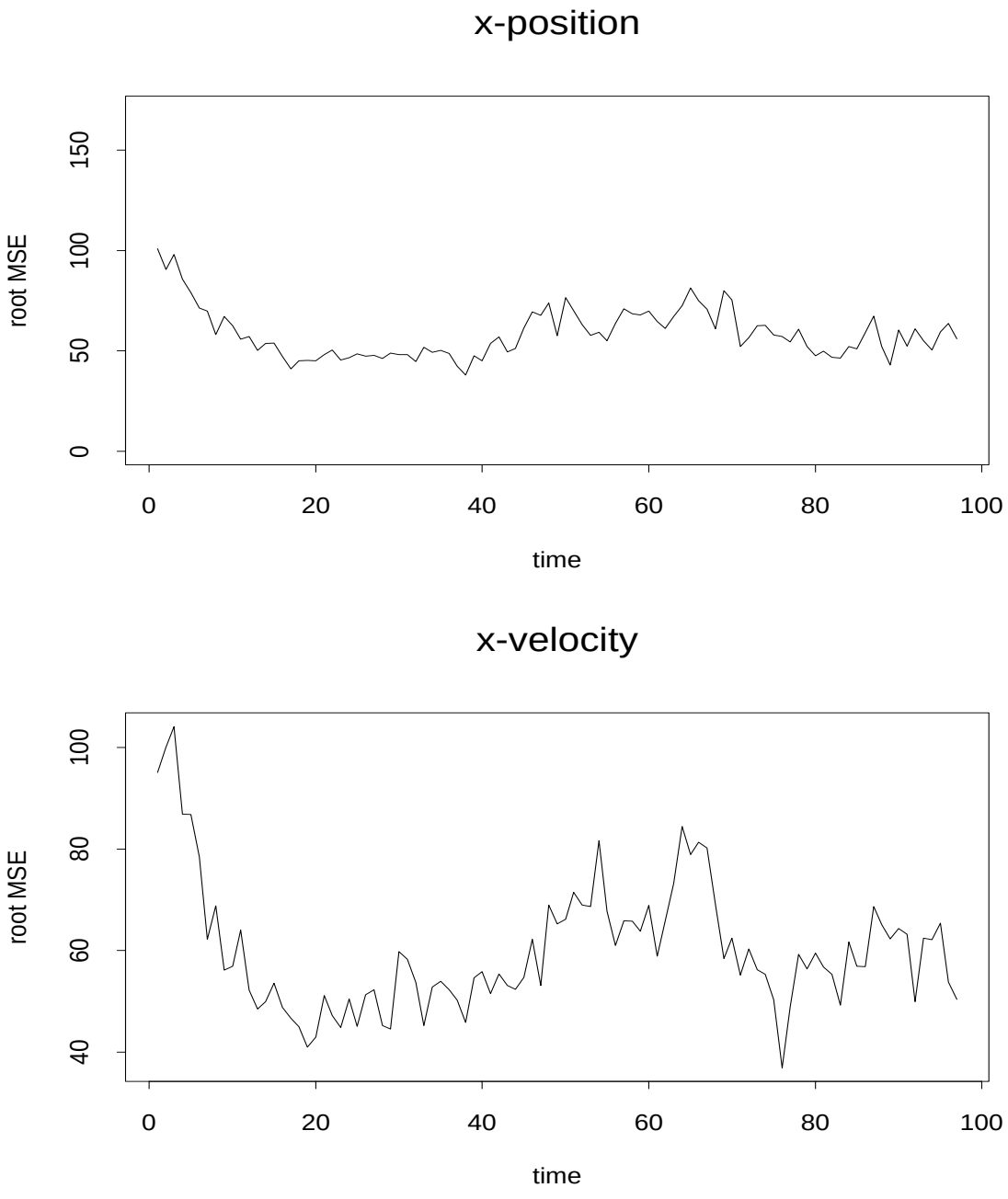


Figure 4: The root MSE's of the x-positiion and x-direction velocity of 50 runs of the MKF for a simulated two-dimensional target moving system with maneuvering.

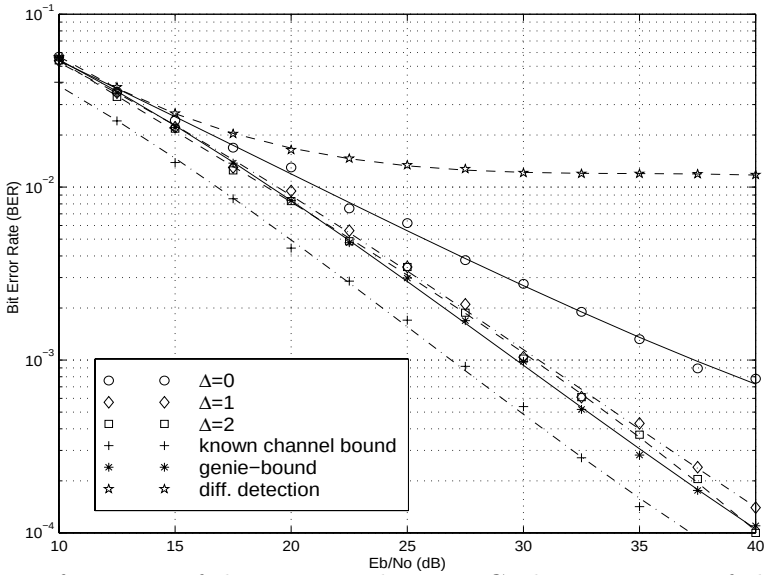


Figure 5: BER performance of the sequential Monte Carlo receiver in a fading channel with Gaussian noise and without coding. The delayed-weight method is used. The BER curves corresponding to delays $\delta = 0$, $\delta = 1$ and $\delta = 2$ are shown. Also shown in the same figure are the BER curves for the known channel lower bound, the genie-aided lower bound and the differential detector.