

Mixture Kalman Filters

Rong Chen

Jun S. Liu

Department of Statistics

Department of Statistics

Texas A&M University

Stanford University

College Station, TX 77843

Stanford, CA 94305

Abstract

In treating dynamic systems, sequential Monte Carlo methods use discrete samples to represent a complicated probability distribution and use rejection sampling, importance sampling, and weighted resampling to complete the on-line “filtering” task. In this article we propose a special sequential Monte Carlo method, the mixture Kalman filter, which uses random mixture of normal distributions to represent a target distribution. It is designed for on-line estimation and prediction of conditional and partial conditional dynamic linear models, which are themselves a class of widely used nonlinear system and also serve to approximate many other nonlinear systems. Compared with a few available filtering methods including Monte Carlo ones, the efficiency gain provided by the mixture Kalman filter can be very substantial. Another contribution of this article is the formulation of many nonlinear systems into conditional or partial conditional linear form, to which the mixture Kalman filter can be applied. Examples in target tracking and digital communications are given to demonstrate the proposed procedures.

1 Introduction

Dynamic system is widely used in almost all fields of applications such as computer vision, economic and financial data analysis, feed-back control systems, mobile communication, radar or sonar surveillance systems, just to name a few. Because most of these systems are nonlinear and non-Gaussian, a main challenge to researchers is to find efficient methods for on-line (in real time) estimation and prediction (filtering) of the ever-changing system characteristics, along with the continuous flow of the information (observations) from the system.

For a Gaussian linear dynamic system, Kalman (1960) provided a genius algorithm (the famous Kalman filter) for on-line filtering. To date, however, there has yet been a universally effective algorithm for dealing with nonlinear and non-Gaussian systems. Depending on the features of individual problems, some generalizations of the Kalman filtering methodology to nonlinear systems can be effective. For example, a few well-known extensions are the extended Kalman filters (Gelb, 1974), the Gaussian sum filters (Anderson and Moore, 1979), and the iterated extended Kalman

filters (Jazwinski, 1970). Most of these methods are based on local linear approximations of the nonlinear system. More recently, researchers began to pay attention to a new class of filtering methods based on the sequential Monte Carlo approach.

Sequential Monte Carlo can be loosely defined as a set of methods that use Monte Carlo simulation to solve *on-line* estimation and prediction problems. More precisely, sequential Monte Carlo techniques achieve the filtering task by recursively generating Monte Carlo samples of the state variables or some other latent variables. These methods are often more adaptive to features of the target system because of the flexible nature of Monte Carlo simulations. Since the appearance of two such methods, the *bootstrap filter* (also called particle filter) for nonlinear state-space models (Gordon, Salmond, and Smith 1993) and the *sequential imputation* for Bayesian missing data problems (Kong, Liu, and Wong 1994), Monte Carlo filtering techniques have caught attentions from researchers in very different areas that require dynamic modeling. There have also been many recent modifications and improvements on the method (Berzuini, Best, Gilks, and Larizza 1997; Carpenter, Clifford, and Fearnhead 1997; Doucet 1998; Hürzeler and Künsch 1995; Liu and Chen 1995; Pitt and Shephard 1999). A *sequential importance sampling* (SIS) framework is proposed in Liu and Chen (1998) to unify and generalize these related techniques. In the following context, we refer to all these methods applied to state-space models as *Monte Carlo filters*.

In this article, we focus on the popular state space model of the following form:

$$\begin{aligned} \text{(state equation):} \quad & x_{t+1} \sim f_t(\cdot | x_t) \\ \text{(observation equation):} \quad & y_{t+1} \sim q_t(\cdot | x_{t+1}), \end{aligned} \tag{1}$$

and a special case of which, the *conditional dynamic linear model* (CDLM). Here the x_t are unobserved state variables and the y_t are observed signals. Let $\mathbf{y}_t = (y_1, \dots, y_t)$ be the information available up to time t . Of interest in these systems are usually (a) *estimation* of the state variable, say $E(x_t | \mathbf{y}_t)$, by using all available information; (b) *prediction* of a future state, say $E(x_{t+1} | \mathbf{y}_t)$; and (c) *revision* of the previous state estimations given new information, e.g., $E(x_{t-l} | \mathbf{y}_t)$. The main challenge is that these tasks need to be done on-line in real time, which makes it critical for a filtering method behave recursively (e.g., modify the estimations or predictions quickly as new observation comes in).

An important feature of the CDLM, whose precise definition can be found in Section 3, is that given the trajectory of an indicator variable (vector), the system is Gaussian and linear, for which the Kalman filter can be used. Thus, by using the *marginalization* technique for Monte Carlo computation (Rubinstein, 1981), we derive a Monte Carlo filter that focuses its full attention on

the space of indicator variable. We call this filter a *mixture Kalman filter* (MKF). By doing so we can drastically reduce Monte Carlo variances associated with a standard sequential importance sampler applied directly to the space of the state variable.

The MKF idea can also be applied to those systems that are only partially conditional linear and Gaussian, i.e., those systems whose state variable consists of a component that is conditionally linear and a component that is completely nonlinear. By conditioning on an indicator and the value of the nonlinear component of the state variable, the system (both the state equation and the observation equation) becomes linear and Gaussian. We call such a system *partial CDLM*. In this case, the linear component of the state variable can be ‘marginalized’ out before running a sequential importance sampler. The marginalization operation is again achieved by the Kalman filter operation. We call this method an extended MKF.

Given the importance of the CDLM in system modeling, it is perhaps not surprising that approaches similar to the MKF described in this article have been proposed earlier. Indeed, the earlier work of Ackerson and Fu (1970), Akashi and Kumamoto (1977), and Tugnait (1982), and recent work of Liu and Chen (1995) and Doucet (1998) are all closely related. We will provide a more detailed account on each of these approaches in Section 3.

The rest of the paper is organized as follows. In Section 2 we provide a brief overview of the celebrated Kalman filter and the central idea of a Monte Carlo filter, which serves as the backbone of our procedure and the basis of comparisons. In Section 3, we give a detailed description of the CDLMs and the proposed mixture Kalman filter. Section 4 is devoted to the partial conditional dynamic linear models and the extended mixture Kalman filters. In Section 5, we give several applications of the proposed MKF and extended MKF, including three examples in target tracking and two examples in telecommunications. A brief summary is given in Section 6.

2 Kalman Filter and Monte Carlo Filter

2.1 Kalman filter

When the transition functions f_t and g_t in the state-space model (1) are linear Gaussian (i.e., with linear mean functions and constant covariance matrices), we have that $x_t | \mathbf{y}_t \sim N(\boldsymbol{\mu}_t, \Sigma_t)$. Based on this fact, Kalman (1960) found a very fast algorithm for the on-line incorporation of the new observation y_{t+1} , i.e., for updating from $(\boldsymbol{\mu}_t, \Sigma_t)$ to $(\boldsymbol{\mu}_{t+1}, \Sigma_{t+1})$. Specifically, consider the following

linear and Gaussian state space model

$$\begin{cases} x_t &= H_t x_{t-1} + W_t w_t, \\ y_t &= G_t x_t + V_t v_t, \end{cases}$$

with known coefficients H_t, G_t, W_t and V_t , and $w_t \sim N(0, I)$ and $v_t \sim N(0, I)$. Assume $x_0 \sim N(\boldsymbol{\mu}_0, \Sigma_0)$, then $p(x_t | \mathbf{y}_t)$ is also Gaussian. Its mean vector and covariance matrix can be obtained recursively. Engineers are often in favor of the following recursions:

$$\begin{aligned} P_{t+1} &= H_t \Sigma_t H_t' + W_t W_t', \\ S_{t+1} &= G_{t+1} P_{t+1} G_{t+1}' + V_t V_t', \\ \boldsymbol{\mu}_{t+1} &= H_{t+1} \boldsymbol{\mu}_t + P_{t+1} G_{t+1}' S_{t+1}^{-1} (y_{t+1} - G_{t+1} H_{t+1} \boldsymbol{\mu}_t), \\ \Sigma_{t+1} &= P_{t+1} - P_{t+1} G_{t+1}' S_{t+1}^{-1} G_{t+1} P_{t+1}. \end{aligned} \tag{2}$$

In addition, the predictive distribution is

$$p(y_{t+1} | \mathbf{y}_t) \sim N(G_{t+1} H_{t+1} \boldsymbol{\mu}_t, S_{t+1}). \tag{3}$$

2.2 Sequential importance sampler for the state space model

Consider the state space model (1). Suppose at time t we have the posterior distribution of x_t as $p(x_t | \mathbf{y}_t)$, then the predictive distribution for x_{t+1} is

$$p(x_{t+1} | \mathbf{y}_t) = \int f_{t+1}(x_{t+1} | x_t) p(x_t | \mathbf{y}_t) dx_t. \tag{4}$$

Since analytical computation with this distribution is often difficult, a possible alternative is the Monte Carlo approximation. More precisely, if we can draw $x_t^{(m)}$ (either iid or dependently) from $p(x_t | \mathbf{y}_t)$, then we can approximate $p(x_{t+1} | \mathbf{y}_t)$ by

$$\hat{p}(x_{t+1} | \mathbf{y}_t) = \frac{1}{M} \sum_{m=1}^M f_t(x_{t+1} | x_t^{(m)}). \tag{5}$$

In many cases, however, directly sampling from $p(x_t | \mathbf{y}_t)$ is infeasible, but drawing from a trial distribution $\alpha(x_t | \mathbf{y}_t)$ is easy. If this is the case, we need to modify (5) by adjusting the *weight* of each sampled $x_t^{(m)}$. That is, if the $x_t^{(m)}$ are drawn from $\alpha(x_t | \mathbf{y}_t)$, then we can modify the approximation (5) as:

$$\tilde{p}(x_{t+1} | \mathbf{y}_t) = \frac{1}{W_t} \sum_{m=1}^M w_t^{(m)} f_t(x_{t+1} | x_t^{(m)}) \tag{6}$$

where $w_t^{(m)} = p(x_t^{(m)} | \mathbf{y}_t) / \alpha(x_t^{(m)} | \mathbf{y}_t)$, and $W_t = \sum_{m=1}^M w_t^{(m)}$. The $w_t^{(m)}$ are usually called ‘‘importance weights.’’

When a new data y_{t+1} is observed, the posterior distribution of x_{t+1} is

$$p(x_{t+1} | \mathbf{y}_{t+1}) \propto g_{t+1}(y_{t+1} | x_{t+1})p(x_{t+1} | \mathbf{y}_t),$$

which, by using (6), can be approximated as

$$\tilde{p}(x_{t+1} | \mathbf{y}_{t+1}) \propto \frac{1}{W_t} \sum_{m=1}^M w_t^{(m)} g_{t+1}(y_{t+1} | x_{t+1}) f_{t+1}(x_{t+1} | x_t^{(m)}). \quad (7)$$

From this approximation, one can conduct a *Sampling-Importance-Resampling* (Rubin, 1987) step to obtain an *approximate* sample from the approximate posterior distribution (7), which forms the basis of the bootstrap filter (Gordon et al., 1993). Many improvements upon this basic algorithm have been proposed, see Liu and Chen (1998) for a recent summary.

In many situations as those focused in this article, we find that quantities

$$\begin{aligned} u_{t+1}^{(m)} &= \int g_{t+1}(y_{t+1} | x_{t+1}) f_{t+1}(x_{t+1} | x_t^{(m)}) dx_{t+1} \\ p(x_{t+1} | y_{t+1}, x_t^{(m)}) &= \frac{1}{u_{t+1}^{(m)}} g_{t+1}(y_{t+1} | x_{t+1}) f_{t+1}(x_{t+1} | x_t^{(m)}) \end{aligned}$$

can often be worked out analytically. Then we can rewrite (7) as

$$\tilde{p}(x_{t+1} | \mathbf{y}_{t+1}) = \frac{1}{W_{t+1}} \sum_{m=1}^M w_{t+1}^{(m)} p(x_{t+1} | y_{t+1}, x_t^{(m)}), \quad (8)$$

where $w_{t+1}^{(m)} = w_t^{(m)} \times u_{t+1}^{(m)}$ and $W_{t+1} = \sum_{m=1}^M w_{t+1}^{(m)}$. This can be used to construct a more efficient algorithm. Firstly, we can draw an *exact* sample from (8) instead of an *approximate* one from (7). Secondly, for any measurable function $h(\cdot)$, we can estimate $E[h(x_{t+1}) | \mathbf{y}_{t+1}]$ by

$$\hat{h}_{t+1} = \frac{1}{W_{t+1}} \sum_{m=1}^M w_{t+1}^{(m)} h(x_{t+1}^{(m)}).$$

Because the weights $w_{t+1}^{(m)}$ incorporate new information from y_{t+1} , an earlier estimate of, say, $h(x_{t-s})$ (i.e., $E\{h(x_{t-s}) | \mathbf{y}_{t-s}\}$), can also be modified to $E\{h(x_{t-s}) | \mathbf{y}_{t+1}\}$ by using these new weights. In order to proceed to time $t + 2$, we may use (8) directly or draw a *sample* of the $x_{t+1}^{(m)}$ from (8), and go back to (4) with t replaced by $t + 1$. The operations from (4) to (8) thus define the basic recursive procedure of a Monte Carlo filter. Some detailed discussions on the implementation and the efficiency of the sequential importance sampler can be found in Liu and Chen (1998).

3 Model and the Method

3.1 Conditional Dynamic Linear Models (CDLM)

A conditional dynamic linear model (CDLM) can be generally described as follows:

$$\begin{cases} x_t = H_\lambda x_{t-1} + W_\lambda w_t \\ y_t = G_\lambda x_t + V_\lambda v_t \end{cases} \quad \text{if } \Lambda_t = \lambda \quad (9)$$

where $w_t \sim N(0, I)$, $v_t \sim N(0, I)$ and all coefficient matrices are known given λ . The Λ_t , which can be either continuous or discrete, is a latent indicator process with certain probabilistic structure.

The CDLM is a direct generalization of the dynamic linear model (DLM) (West and Harrison, 1989) and has been widely used in practice. With discrete indicator variables, the model can be used to deal with outliers, sudden jumps, system failures, environmental changes, and clutters. With carefully chosen continuous indicator variables, CDLM can also accommodate dynamic linear models with non-Gaussian innovations (e.g. t -distributions, logistic distributions, etc).

Example 1: A special CDLM is the linear system with non-Gaussian errors. Suppose

$$\begin{cases} x_t = Hx_{t-1} + Ww_t \\ y_t = Gx_t + Vv_t, \end{cases}$$

where w_t and v_t are mixture Gaussian, i.e., conditioning on some unobserved variables η_{1t} and η_{2t} , $w_t|\eta_{1t} \sim N(\boldsymbol{\mu}_1(\eta_{1t}), \Sigma_1(\eta_{1t}))$ and $v_t|\eta_{2t} \sim N(\boldsymbol{\mu}_2(\eta_{2t}), \Sigma_2(\eta_{2t}))$, for some mean functions $\boldsymbol{\mu}_1, \boldsymbol{\mu}_2$ and variance functions Σ_1 and Σ_2 . This model is clearly a CDLM with η_{1t} and η_{2t} being its latent indicator processes. This class of error models include, in addition to the discrete mixture of Gaussian distributions, also the t distributions, double exponential distributions, exponential power family, logistic distributions, etc. Even if w_t and v_t are not mixture Gaussian, most of the time they can be satisfactorily approximated by mixture Gaussian distributions. However, one has to balance complexity and accuracy: greater efficiency can be achieved if the distribution is approximated accurately, with relatively simple indicators. In section 5 we show and analyze several CDLMs in practice.

Engineers have begun to deal with special forms of the CDLM since 1970s. In a pioneering work, Ackerson and Fu (1970) consider a linear system operating in switching environments, which they formulate as the model in Example 1 with the Λ_t being a finite discrete Markovian indicator process. To deal with the computational difficulty, they propose an approximate filtering procedure in which the posterior probability of the indicator variable Λ_t (given y_t) is recursively updated

under a conditional independence assumption, and then used in a Gaussian approximation of the posterior of x_t . Their approach can be easily generalized to update a segment $(\Lambda_{t-k}, \dots, \Lambda_t)$ of the indicator process recursively (Tugnait, 1982). In dealing with the same CDLM, Akashi and Kumamoto (1977) introduce essentially a sequential importance sampling method for the indicator process in which an “optimal” sampling distribution is used (Doucet 1998; Liu and Chen 1998). Thus, Akashi and Kumamoto’s algorithm is closest to the MKF proposed in this article. However, the key resampling and rejection steps are missing in their method, which makes it perform much less satisfactorily. By formulating the MKF in a general sequential Monte Carlo framework, we are able to incorporate various Monte Carlo techniques, such as resampling, rejection control, and auxiliary variable approach into the scheme and to greatly extend the applicability of the method. More recently, the methods used in Svetnik (1986), Liu and Chen (1995) and Doucet (1998) have all captured some attractive aspects of the CDLM and the MKF, but they are limited in scope.

Several Markov chain Monte Carlo algorithms for this type of models have been proposed (See Carlin, Stoffer, and Polson, 1992; Carter and Kohn, 1994). In particular, Carter and Kohn (1994) present an efficient Gibbs sampler in which the indicator Λ_t is the only latent variable to be imputed and the state variable x_t is explicitly integrated out via a clever use of forward and backward Kalman filtering. A main problem with all the MCMC algorithms for dynamic systems, however, is that they can not be effectively used for on-line estimation and prediction. Whereas a fast and efficient on-line algorithm in problems such as target tracking and digital signal processing is essential.

The bootstrap filter (Avitzour 1995; Gordon et al. 1993; and Kitagawa 1996) can be directly applied to the CDLM for on-line estimation and prediction. In such a procedure, multiple Monte Carlo samples of the state variable x_t are recursively imputed using the sampling-importance-resampling technique. A more efficient alternative, sequential imputation with rejuvenation, was proposed in Liu and Chen (1995) for a blind deconvolution problem. However, an even more sophisticated algorithm can be obtained by making further use of the conditional Gaussian structure, as in Akashi and Kumamoto (1977), Carter and Kohn (1994), Doucet (1998), and Liu and Chen (1998). That is, when conditioning on $\Lambda_1, \dots, \Lambda_t$, the CDLM defined by (9) becomes a Gaussian DLM, and all the x_s , $s = 1, \dots, t$, can be integrated out recursively by using a standard Kalman filter. West (1992) first suggest to use mixture distributions for a general adaptive importance sampling strategy in dynamic systems. In a CDLM, mixture Gaussian distribution becomes an obvious choice, because of the efficient Kalman filter.

3.2 The Method of Mixture Kalman Filtering

Let $\mathbf{y}_t = (y_1, \dots, y_t)$ and $\mathbf{\Lambda}_t = (\Lambda_1, \dots, \Lambda_t)$. We observe that

$$p(x_t | \mathbf{y}_t) = \int p(x_t | \mathbf{\Lambda}_t, \mathbf{y}_t) p(\mathbf{\Lambda}_t | \mathbf{y}_t) d\mathbf{\Lambda}_t,$$

where

$$p(x_t | \mathbf{\Lambda}_t, \mathbf{y}_t) \sim N(\boldsymbol{\mu}_t(\mathbf{\Lambda}_t), \Sigma_t(\mathbf{\Lambda}_t)),$$

in which $(\boldsymbol{\mu}_t(\mathbf{\Lambda}_t), \Sigma_t(\mathbf{\Lambda}_t))$ can be obtained by running a Kalman filter with given trajectory $\mathbf{\Lambda}_t$.

The main idea of the MKF is to use a *weighted sample* of the indicators,

$$\mathcal{S}_t = \{(\boldsymbol{\lambda}_t^{(1)}, w_t^{(1)}), \dots, (\boldsymbol{\lambda}_t^{(m)}, w_t^{(m)})\},$$

to represent the distribution $p(\mathbf{\Lambda} | \mathbf{y}_t)$, and then use a random mixture of Gaussian distributions,

$$\sum_{j=1}^m w_t^{(j)} N(\boldsymbol{\mu}_t(\boldsymbol{\lambda}_t^{(j)}), \Sigma_t(\boldsymbol{\lambda}_t^{(j)})),$$

to approximate the target distribution $p(x_t | \mathbf{y}_t)$. For any integrable function $h(\cdot)$, we can approximate the quantity of interest $E\{h(x_t) | \mathbf{y}_t\}$ as

$$\hat{E}(h(x_t) | \mathbf{y}_t) = \sum_{j=1}^m w_t^{(j)} \int h(x) \varphi(x; \boldsymbol{\mu}_t(\boldsymbol{\lambda}_t^{(j)}), \Sigma_t(\boldsymbol{\lambda}_t^{(j)})) dx,$$

where φ is the Gaussian density function.

A straightforward sequential Monte Carlo method as described in Section 2.2 simply uses a weighted sample of the state variable, $\{(x_t^{(1)}, w_t^{(1)}), \dots, (x_t^{(m)}, w_t^{(m)})\}$, to approximate $p(x_t | \mathbf{y}_t)$. Whereas the MKF tries to sample in the indicator space instead, which is equivalent to marginalizing out the x_t . This operation has been shown to improve a Gibbs sampling algorithm (Liu, Wong, and Kong, 1994). Its advantage in a usual importance sampling scheme is shown in MacEachern et al. (1998). In the MKF setting, there is no clear theory available at this point. Our limited experience shows that the efficiency gain of the MKF can be very significant. Intuitively, the usual sequential Monte Carlo recursively approximates the posterior of x_t by a discrete sample; whereas the MKF approximates the posterior of x_t by a mixture of Gaussian distributes. Note that the true posterior of x_t in a CDLM is indeed a mixture of Gaussian, although the number of its components increases exponentially along with t .

Let $KF_t^{(j)} = (\boldsymbol{\mu}_t(\boldsymbol{\lambda}_t^{(j)}), \Sigma_t(\boldsymbol{\lambda}_t^{(j)}))$, the mean and variance matrix of x_t obtained by Kalman filter at time t given a trajectory $\boldsymbol{\lambda}_t^{(j)}$. Then the MKF consists of recursive applications of the following:

MKF updating Step : For $j = 1, \dots, m$,

(M1) : generate $\lambda_{t+1}^{(j)}$ from a trial distribution $g(\lambda_{t+1} | \boldsymbol{\lambda}_t^{(j)}, KF_t^{(j)}, y_{t+1})$

(M2) : Obtain $KF_{t+1}^{(j)}$ by one-step Kalman filter, as shown in (2), conditional on $\{KF_t^{(j)}, y_{t+1}, \lambda_{t+1}^{(j)}\}$.

(M3) : update the new weight as $w_{t+1}^{(j)} = w_t^{(j)} \times u_{t+1}^{(j)}$, where

$$u_{t+1}^{(j)} = \frac{p(\boldsymbol{\lambda}_t^{(j)}, \lambda_{t+1}^{(j)} | \mathbf{y}_{t+1})}{p(\boldsymbol{\lambda}_t^{(j)} | \mathbf{y}_t)g(\lambda_{t+1}^{(j+1)} | \boldsymbol{\lambda}_t^{(j)}, KF_t^{(j)}, y_{t+1})}.$$

(M4) : if the coefficient of variation of the w_{t+1} exceeds a threshold value, we resample a new set of KF_{t+1} from $\{KF_{t+1}^{(1)}, \dots, KF_{t+1}^{(m)}\}$ with probability proportional to the weights $w_{t+1}^{(j)}$.

When Λ_t takes value in a finite discrete set, \mathcal{I} , then the most efficient trial distribution for Λ_{t+1} is $g(\lambda_{t+1} | \boldsymbol{\lambda}_t, KF_t, y_{t+1}) = p(\lambda_{t+1} | \boldsymbol{\lambda}_t, KF_t, y_{t+1})$, which can be obtained by inspecting all the possible values of Λ_t . The incremental weight $u_{t+1}^{(j)}$ is then simplified as

$$u_{t+1}^{(j)} \propto p(y_{t+1} | KF_t^{(j)}) = \sum_{i \in \mathcal{I}} p(y_{t+1} | \Lambda_{t+1} = i, KF_t^{(j)})p(\Lambda_{t+1} = i | \boldsymbol{\lambda}_t^{(j)}).$$

Specifically, a MKF updating step in this case becomes: For $j = 1, \dots, m$,

(M0) : For each $\Lambda_{t+1} = i, i \in \mathcal{I}$, run a Kalman filter to obtain

$$v_i^{(j)} \propto p(y_{t+1} | \Lambda_{t+1} = i, KF_t^{(j)})p(\Lambda_{t+1} = i | \boldsymbol{\lambda}_t^{(j)}),$$

where $p(\Lambda_{t+1} = i | \boldsymbol{\lambda}_t^{(j)})$ is the prior transition probability for the indicator and $p(y_{t+1} | \Lambda_{t+1} = i, KF_t^{(j)})$ is a by-product of the Kalman filter, using (3).

(M1) : Sample a $\lambda_{t+1}^{(j)}$ from the set \mathcal{I} , with probability proportional to $v_i^{(j)}$.

(M2) : Let $KF_{t+1}^{(j)}$ be the one with $\Lambda_{t+1} = \lambda_{t+1}^{(j)}$.

(M3) : The new weight is $w_{t+1}^{(j)} = w_t^{(j)} \sum_{i \in \mathcal{I}} v_i^{(j)}$.

Smith and Winter (1978) proposed a deterministic filtering method, called *split-track filter*, which has a similar flavor to the MKF we proposed. It is designed for CDLMs with discrete latent indicator variables. In split-track filters, one always keeps m trajectories of the latent indicators. At a future time step, it evaluates the likelihoods of all possible propagations from the m trajectories kept at previous step, then find and keep the m new trajectories with the highest likelihood values. In contrast, our MKF selects these trajectories randomly, according to the weights (which is the predictive likelihood value), and uses the associated weights to measure how good each trajectory

is. The important step of resampling is naturally built into MKF which can overcome some weaknesses of the split-track filter. More sophisticated sampling and estimation methods can also be incorporated. A comparison of the MKF and the split-track filter in a target tracking problem is presented in Section 5.

When Λ_t is a continuous random variable a simpler but less efficient algorithm is

- (M1) : Sample a $\lambda_{t+1}^{(i)}$ from $p(\Lambda_{t+1} | \lambda_t^{(i)})$, the prior structure of the indicator variable.
- (M2) : Run one step of Kalman Filter on $\{\lambda_{t+1}^{(i)}, KF_t^{(i)}, y_{t+1}\}$ to obtain $KF_{t+1}^{(i)}$, using (2).
- (M3) : The new weight is $w_{t+1}^{(i)} = w_t^{(i)} p(y_{t+1} | \lambda_{t+1}^{(i)}, KF_t^{(i)})$ using (3).

Methods of Berzuini et al. (1997) and Pitt and Shephard (1999) can be applied to improve the efficiency of this algorithm.

4 The Extended Mixture Kalman Filters

4.1 Partial conditional dynamic linear models

Suppose the state variable has two components: $x_t = (x_{t1}, x_{t2})$. The following system is called a partial conditional dynamic linear model (PCDLM):

$$\text{State Equation: } \begin{cases} x_{t,1} = H_t(x_{t-1,2}, \Lambda_t)x_{t-1,1} + W_t(x_{t-1,2}, \Lambda_t)w_t \\ x_{t,2} = g_t(x_{t-1,2}, \Lambda_t, \varepsilon_t) \end{cases}$$

$$\text{Observation Equation: } y_t = G_t(x_{t,2}, \Lambda_t)x_{t,1} + h_t(x_{t,2}, \Lambda_t) + V_t(x_{t,2}, \Lambda_t)v_t$$

with $w_t \sim N(0, I)$ and $v_t \sim N(0, I)$. The matrices H_t, G_t, W_t, V_t are known given the values of $\{x_{t-1,2}, x_{t,2}, \Lambda_t\}$. The functions g_t and h_t are known and ε_t has a known distribution.

There is in fact no absolute distinction between the PCDLM and the CDLM because if we regard the “nonlinear” component $x_{t,2}$ of the state variable in a PCDLM as part of the indicator variable, the system becomes a CDLM. However, unlike in CDLM where we have no interest in the latent indicator, the inference of the ‘nonlinear component’ of the state variable in a PCDLM is often of great interest. Note that in our model formulation the state propagation of the nonlinear component does not depend on the linear component.

Example 2. Fading Channel: Many mobile communication channels can be modeled as

Rayleigh flat-fading channels, which has the following form:

$$\begin{aligned} \text{State Equations: } & \begin{cases} \mathbf{x}_t = F\mathbf{x}_{t-1} + Ww_t \\ \alpha_t = G\mathbf{x}_t \\ s_t \sim p(\cdot | s_{t-1}) \end{cases} \\ \text{Observation Equation: } & y_t = \alpha_t s_t + Vv_t \end{aligned}$$

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. This system is clearly a PCDLM. Given the input signals s_t , the system is linear in \mathbf{x}_t and y_t . In Section 5 we show how to use the extended MKF for extracting digital signals transmitted over such channels.

Example 3. Blind Deconvolution. Consider the following system in digital communication

$$y_t = \sum_{i=1}^q \theta_i s_{t-i} + \varepsilon_t,$$

where s_t is a discrete process taking values on a known set S . In a blind deconvolution problem, s_t is to be estimated from the observed signals $\{y_1, \dots, y_t\}$, without knowing the channel coefficients θ_i . This system can be formulated as a PCDLM. Let $\boldsymbol{\theta}_t = (\theta_{t1}, \dots, \theta_{tq})$ and $x_t = (s_t, \dots, s_{t-q})'$. We can define

$$\begin{aligned} \text{State Equation: } & \begin{cases} \boldsymbol{\theta}_t = \boldsymbol{\theta}_{t-1} \\ x_t = Hx_{t-1} + Ws_t \end{cases} \\ \text{Observation equation: } & y_t = \boldsymbol{\theta}_t x_t + \varepsilon_t \end{aligned}$$

where H is a $q \times q$ matrix with lower off-diagonal element being one and all other elements being zero and $W = (1, 0, \dots, 0)'$. In this case, the unknown system coefficients are part of the state variable, and is linear conditional on the digital signal x_t . Liu and Chen (1995) studied this problem with a procedure which is essentially an extended MKF as described in the next subsection. This PCDLM formulation can be easily extended to deal with a blind deconvolution problem with time-varying system coefficients.

4.2 The Extended MKF

The main idea of the extended MKF (EMKF) is to extract as many linear and Gaussian components from the system as possible, and then integrate these components out (marginalized) using the Kalman Filter before running a Monte Carlo filter to the remaining components. Thus, in EMKF

we generate discrete samples in the joint space of the latent indicator and the nonlinear state component. More intuitively, because of the fact that $p(x_{t1}, x_{t2} | \mathbf{y}_t) = p(x_{t1} | x_{t2}, \mathbf{y}_t)p(x_{t2} | \mathbf{y}_t)$, the approximation of $p(x_{t1}, x_{t2} | \mathbf{y}_t)$ in EMKF is decomposed as a Monte Carlo approximation of the marginal distribution $p(x_{t2} | \mathbf{y}_t)$ and an exact Gaussian conditional distribution $p(x_{t1} | x_{t2}, \mathbf{y}_t)$. The EMKF algorithm is as follows: suppose at time t , we have a sample $(\boldsymbol{\lambda}_t^{(j)}, x_{t,2}^{(j)}, KF_t^{(j)}, w_t^{(j)})$, $j = 1, \dots, m$, where $KF_t^{(j)} = (\boldsymbol{\mu}_t(\boldsymbol{\lambda}_t^{(j)}, x_{t,2}^{(j)}), \Sigma_t(\boldsymbol{\lambda}_t^{(j)}, x_{t,2}^{(j)}))$, the mean and variance matrix of $p(x_{t1} | x_{t2}^{(j)}, \boldsymbol{\lambda}_t^{(j)}, \mathbf{y}_t)$ obtained by the Kalman filter. At time $t + 1$, we update these filters as follows:

(E1) : generate $(\lambda_{t+1}^{(j)}, x_{t+1,2}^{(j)})$ from a trial distribution $g(\lambda_{t+1}, x_{t+1,2} | \boldsymbol{\lambda}_t^{(j)}, x_{t,2}^{(j)}, KF_t^{(j)}, y_{t+1})$

(E2) : run one step Kalman filter conditioning on $(\lambda_{t+1}^{(j)}, x_{t+1,2}^{(j)}, KF_t^{(j)}, y_{t+1})$ and obtain $KF_{t+1}^{(j)}$.

(E3) : calculate the incremental weight

$$u_{t+1}^{(j)} = \frac{p(\boldsymbol{\lambda}_t^{(j)}, x_{t+1,2}^{(j)}, \lambda_{t+1}^{(j)} | \mathbf{y}_{t+1})}{p(\boldsymbol{\lambda}_t^{(j)}, x_{t+1,2}^{(j)} | \mathbf{y}_t)g(\lambda_{t+1}^{(j)}, x_{t+1,2}^{(j)} | \boldsymbol{\lambda}_t^{(j)}, KF_t^{(j)}, y_{t+1})}$$

and update the new weight as $w_{t+1}^{(j)} = w_t^{(j)} u_{t+1}^{(j)}$.

(E4) : resampling as in (M4) if necessary.

From the weighted sample obtained at each time t , we can estimate quantities of interest, e.g.,

$$E\{h(x_t) | y_1, \dots, y_t\} \approx W_t^{-1} \sum_{j=1}^m w_t^{(j)} \int h(x_1, x_{t,2}^{(j)}) \varphi(x_1; \boldsymbol{\mu}_t^{(j)}, \Sigma_t^{(j)}) dx_1$$

where $W_t = \sum w_t^{(j)}$. In particular,

$$E\{h_1(x_{t,1}) | y_1, \dots, y_t\} \approx W_t^{-1} \sum_{j=1}^m w_t^{(j)} \int h_1(x_1) \phi(x_1; \boldsymbol{\mu}_t^{(j)}, \Sigma_t^{(j)}) dx_1,$$

$$E\{h_2(x_{t,2}) | y_1, \dots, y_t\} \approx W_t^{-1} \sum_{j=1}^m w_t^{(j)} h_2(x_{t,2}^{(j)}).$$

5 Some Numerical Examples

5.1 A linear system with mixture Gaussian errors

To illustrate how the MKF works, we consider the following simple state space model with outliers:

$$\begin{aligned} x_t &= ax_{t-1} + \epsilon_t, \quad \text{with } \epsilon_t \sim N(0, \sigma_j^2) \quad \text{if } J_t = j; \\ y_t &= x_t + \eta_t \quad \text{with } \eta_t \sim N(0, \sigma_\eta^2). \end{aligned}$$

And we assume that $P(J_t = j \mid \mathbf{J}_{t-1}, \mathbf{x}_{t-1}) = P(J_t = j) = \pi_j$, which can be easily extended to a Markovian case.

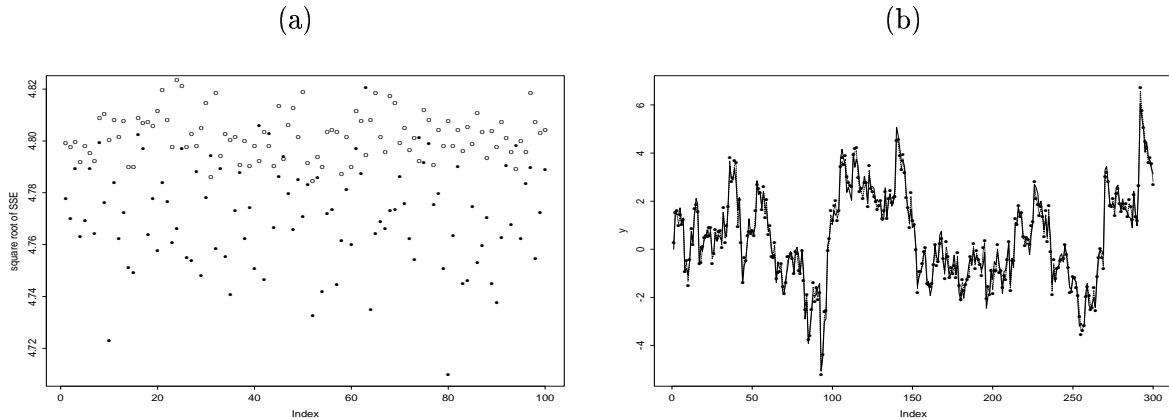


Figure 1: (a) The mean-squared-error of the estimates of x_t for using the MKF with (dots) and without (circles) resampling in 100 repeated experiments. (b) The true x_t versus their on-line estimation from MKF: dots — observations (y_t); line — true values (x_t); dotted lines — estimates (i.e., $E(x_t \mid \mathbf{y}_t)$).

We simulated a system with $a=0.9$, $\sigma_1=0.5$, $\sigma_2=1.5$, $\pi_1=0.7$, and $\sigma_\eta=0.3$, and applied the MKF for on-line estimation. We also tested the effect of a partial resampling scheme for the MKF (Liu et al., 1998). The numerical result was very satisfactory: the MKF estimated the state variable x_t very accurately disregarding several sudden jumps in the system. With Monte Carlo sample size $m=50$, we already produced a result better than that in Kitagawa (1996), who routinely used $m=1,000$ “particles.” The figure also shows that the application of resampling can effectively reduce the mean squared errors of the estimates.

5.2 Target tracking

Designing sophisticated target tracking algorithm is an important task to 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.

5.2.1 Random (Gaussian) accelerated target in clutter

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

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

where x_t is the state variable (location and velocity) of the target and w_t, v_t are white Gaussian with identity covariance matrix.

For targets moving on a straight line, we have $x_t = (s_t, v_t)$ where s_t is the true target location and v_t is its current velocity. In this case

$$H = \begin{pmatrix} 1 & 0 \\ 0 & T \end{pmatrix}; \quad W = \sigma_w^2 \begin{pmatrix} T/2 \\ 1 \end{pmatrix}; \quad G = (1, 0) \quad \text{and} \quad V = \sigma_v^2,$$

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 $x_t = (s_t, v_t)$ with s_t and v_t being two (three) dimensional vectors. The corresponding matrixes can be expanded similarly.

In a clutter environment, we observe m_t signals $\{y_{t1}, \dots, y_{tm_t}\}$ 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 z_t being detected, λ is the rate of a Poisson random field and Δ is the surveillance region. In words, at time t we observe the true signal with probability p_d . We also observe false signals, such as deceiving objects, electro-magnetic interferences, etc., distributed as a Poisson process in the detection region.

This model can be easily formulated as a CDLM: let Λ_t be the identifier of the target at time t . That is, $\Lambda_t = 0$ if the target is not observed, and $\Lambda_t = i$ if the i -th object on the detection screen is the true signal generated from the target, i.e. $y_{tj} = z_t$. Given the indicators, the system is linear and Gaussian, and the remaining y signals bear no information.

Figure 2 shows the plots of tracking errors (the difference of the estimated and true target location) of 50 simulated runs of a one-dimensional target, with $r^2 = 1.0, q^2 = 1.0, p_d = 0.9$ and $\lambda = 0.1$. Five hundred Monte Carlo samples are used for both MKF and an ordinary sequential importance sampler. We also run the split-track filter, which, at each step, saves the 500 trajectories with the highest likelihood values. We can see that the MKF performs much better than the other two algorithms.

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

This situation is usually modeled as follows:

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

with w_t and v_t are non-Gaussian errors. If w_t and v_t are mixture Gaussian distributions, this model is clearly a CDLM. For example, if $w_t \sim t_{k_1}$ and $v_t \sim t_{k_2}$, we can define $\Lambda_t = (\Lambda_{t1}, \Lambda_{t2})$ with prior distributions as independent $\chi_{k_1}^2$ and $\chi_{k_2}^2$ respectively. Then the above model can be rewritten as:

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

with $e_t \sim N(0, I)$ and $\varepsilon_t \sim N(0, I)$.

Simulation were carried out with the following systems:

$$H = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}; \quad W = q \begin{pmatrix} 1/2 \\ 1 \end{pmatrix}; \quad G = (1, 0) \quad \text{and} \quad V = r$$

and $w_t \sim t_3$ and $v_t \sim t_3$. The following table shows a comparison of the MKF and a standard sequential importance sampler 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)	Particle		MKF	
		cpu time	# miss	cpu time	# miss
$q^2 = 16.00$ $r^2 = 1600$	20	9.49843	72	19.4277	1
	50	20.1622	20	51.6061	1
	200	80.3340	7	181.751	1
	500	273.369	4	500.157	1
	1500	1063.36	3	2184.67	1

We observed that the MKF takes about twice as much CPU time as the standard sequential sampler for the same m . Figure 3 shows the tracking mean squares error, after the lost tracks are eliminated. Clearly, the MKF performed much better in this example for the same amount of CPU time.

We also tested the idea of using a finite mixture of Gaussians 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 using

this mixture approach is that with discrete indicators a more efficient MKF can be used. However, the approximation also causes some bias.

5.2.3 Maneuvered target in a clean environment:

This situation is usually 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. The prior structure of u_t is the key of this model. First, maneuvering can be classified into several categories, indicated by an indicator. For example: in 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, ($u_t \sim N(0, \sigma_i^2)$, $\sigma_1^2 < \sigma_2^2$). One can also specify a transition probabilities $P(I_t = j \mid I_{t-1} = i) = p_{ij}$ for the maneuvering status. 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. This is the easiest but not a very realistic model. Other possible models are currently under investigation.

We applied the MKF to 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° turn which is completed in 20 sampling periods. A second, fast, 90° turn starts at $k = 61$ and is completed in 5 sampling times. The slow turn is the result of acceleration inputs $u^x = u^y = 0.075$ ($40 < k \leq 60$), and the fast turn is from $u^x = -u^y = -0.3$ ($61 < k \leq 65$). Figure 4 shows the trajectory of the target and its x -direction and y -direction velocity in one simulated run. In Figure 5 we present the root mean square errors of the MKF estimates of the target position for 50 simulated runs with three noise levels: 0.000001, 1 and 36. 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.

5.3 A simple illustration of the extended MKF

Consider the system

$$\begin{aligned}x_{t,1} &= 0.9x_{t-1,1} + 0.2x_{t-1,2}^2 + q_1e_{t1} \\x_{t,2} &= x_{t-1,2} + q_2e_{t2} \\y_t &= 0.5x_{t,1}x_{t,2} + rv_t\end{aligned}$$

with $e_{t1}, e_{t2}, v_t \sim N(0, 1)$. Note that, even though $x_{t,2}$ has a linear propagation equation, it is actually a nonlinear component, because it appears in the other state equation in a nonlinear form. Given $x_{t-1,2}$ and $x_{t,2}$, the system is linear in both the first state equation and the observation equation.

Figure 6 shows a comparison of the MSE in estimating $x_{t,1}$ and $x_{t,2}$ by using EMKF with Monte Carlo sample size $m = 300$ and the standard sequential importance sampler with $m = 900$. Fifty simulated runs of the above PCDLM with $q_1 = 10, q_2 = 0.1, r = 1$ were carried out. Resampling were done at every time step. The CPU time of the EMKF is almost the same as a standard sequential importance sampler for the same m . Other configurations showed similar results.

5.4 Digital Signal Extraction in Fading Channels

Consider Example 2 in Section 4.1 with binary input signals $s_t = \{1, -1\}$. The fading coefficient takes complex values, with independent real and imaginary parts following the same state equation. Simulation were done with the following configurations

$$F = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & .9391 & -2.8763 & 2.9372 \end{pmatrix}; \quad G' = 10^{-4} \begin{pmatrix} .0376 \\ .1127 \\ .1127 \\ .0376 \end{pmatrix}; \quad W = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}; \quad \text{and} \quad V = r.$$

That is, both of the real and the imaginary parts of α_t follow an ARMA(3,3) process

$$\alpha_t - 0.9391\alpha_{t-1} + 2.8763\alpha_{t-2} - 2.9372\alpha_{t-3} = 0.0376e_t + 0.1127e_{t-1} + 0.1127e_{t-2} + 0.0376e_{t-3}$$

where $e_t \sim N(0, 0.01^2)$. This is, in the communication literature, a (lowpass) Butterworth filter of order 3 with cutoff frequency 0.01. It is also normalized to have stationary variance 1.

We are interested in estimating the differential code $d_t = s_t s_{t-1}$. Figure 7 shows the bit error rate of different signal to noise ratios (SNR), using EMKF, the differential filter (DPSK) $\hat{d}_t = \text{sign}(\text{real}(y_t y_{t-1}^*))$ and a lower bound. The lower bound is obtained using the true fading coefficients α_t and $\hat{d}_t = \text{sign}(\text{real}(\alpha_t^* y_t y_{t-1}^* \alpha_{t-1}))$.

Monte Carlo sample size m was 100 for MKF, except in the case when $\text{SNR} \leq 10$, in which $m=500$. We can see that the simple DPSK works very well in low SNR cases and no significant improvement can be expected. However, DPSK has an apparent bit error rate floor for high SNR cases. The MKF managed to break that floor, by using the structure of the fading coefficients.

6 DISCUSSION

In this article, we propose the *mixture Kalman filter* for on-line estimation and prediction in conditional dynamic linear models. The methodology is further extended to deal with partial conditional dynamic linear models. The MKF is a sequential Monte Carlo technique in which we a *marginalization* operation is employed for improving its efficiency. All of our numerical examples show that the MKF approach gains significantly over the earlier sequential Monte Carlo approaches, e.g., the bootstrap filter and the sequential imputation.

The developments in this article also show that the *sequential importance sampling* is a very general and powerful platform that allows people to design various improved algorithms by making use of special structure of a given problem. We hope that the new tools we add to the nonlinear filtering toolkit are of interest to researchers, as well as practitioners, in this field.

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*, Prentice-Hall
- Avitzour, D. (1995), A stochastic simulation Bayesian approach to multitarget tracking, *IEE Proceedings on 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. (1996), Dynamic conditional independence models and Markov chain Monte Carlo methods, *J. Amer. Statist. Assoc.*, to appear.
- Carlin, B.P, Polson, N. G. and Stoffer, D. S. (1992), A Monte Carlo approach to nonnormal and nonlinear state-space modeling, *Journal of the American Statistical Association*, **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.

- Doucet, A. (1998). On sequential simulation-based methods for Bayesian filtering. *Technical report* TR.310, Department of Engineering, University of Cambridge.
- Gelb, A. (1974), *Applied Optimal Estimation*, 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 Proceedings on Radar and Signal Processing*, **140**, 107-113.
- Hürzeler, M. and Künsch, H.R. (1995), “Monte Carlo approximations for general state space models.” *Research Report 73*, ETH, Zürich.
- Jazwinski, A. (1970), *Stochastic Processes and Filtering Theory*, 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, *Journal of Computational and Graphical Statistics*, **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.
- Liu, J.S. and Chen, R. (1995), Blind deconvolution via sequential imputations, *Journal of the American Statistical Association*, **90**, 567-576.
- (1998), Sequential Monte Carlo methods for dynamic systems, *Journal of the American Statistical Association*, **93**, 1032-1044
- Liu, J.S., Chen, R., and Wong, W.H. (1998), Rejection control for sequential importance sampling, *Journal of the American Statistical Association*, **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. (1998), Sequential Importance Sampling for Non-parametric Bayes Models: The Next Generation, *Canadian Journal of Statistics*, in press.
- Pitt, M.K. and Shephard, N. (1999), Filtering via simulation: auxiliary particle filters, *J. Amer. Statist. Asso.*, in press.
- 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, *Journal of the American Statistical Association*, **52**, 543-546.

- Rubinstein, R.Y. (1981). *Simulation and the Monte Carlo Method*, New York: Wiley.
- 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 Deigo, CA. January 1979
- Svetnik, V.B. (1986). Applying the Monte Carlo method for optimum estimation in systems with random structure. *Auto. Remo. Cont.*, **47**, 818-827.
- 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-Verlag.

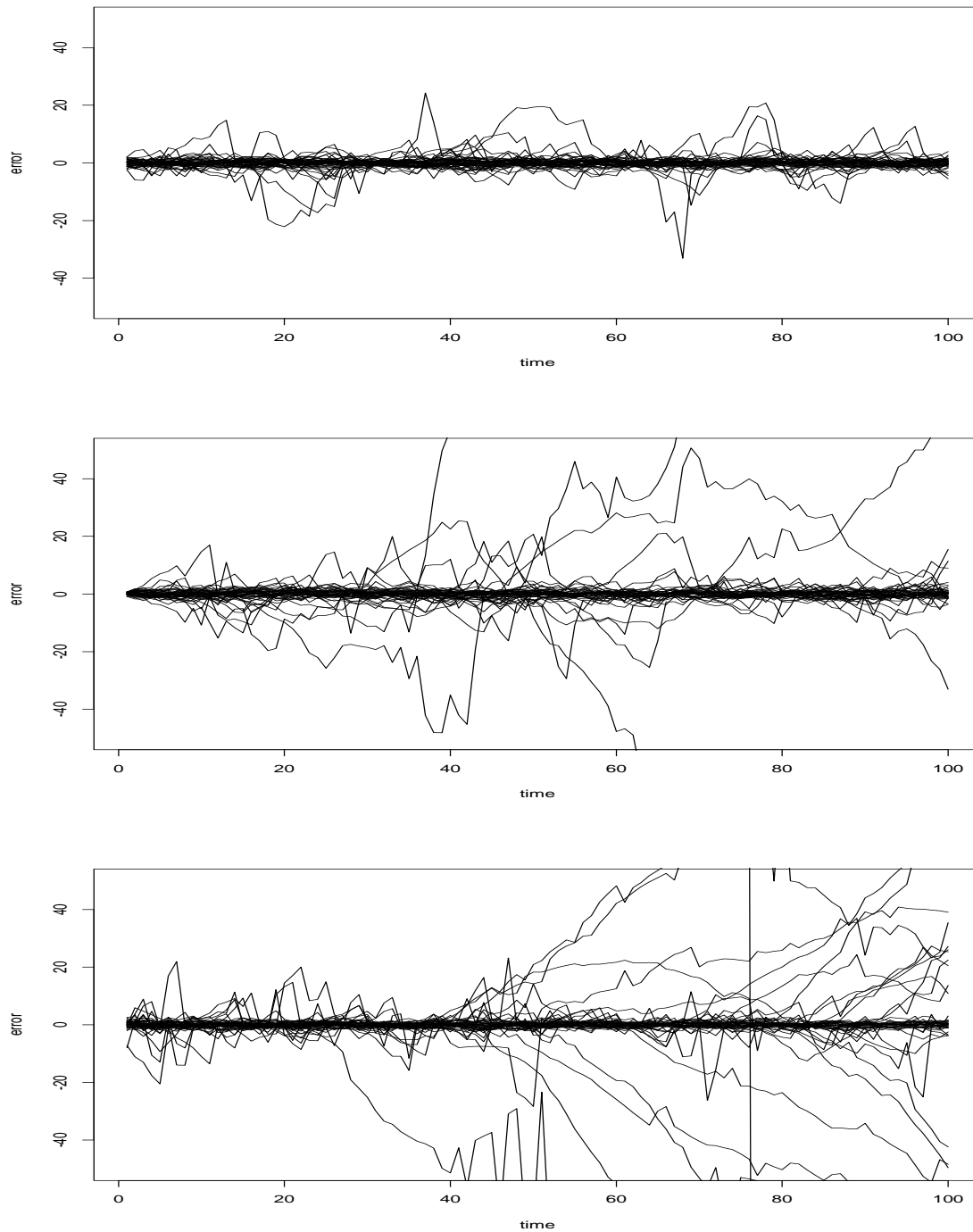


Figure 2: The tracking errors of 50 runs of the MKF (top), a sequential importance sampler (middle), and the split-track filter (bottom) for a simulated target moving system.

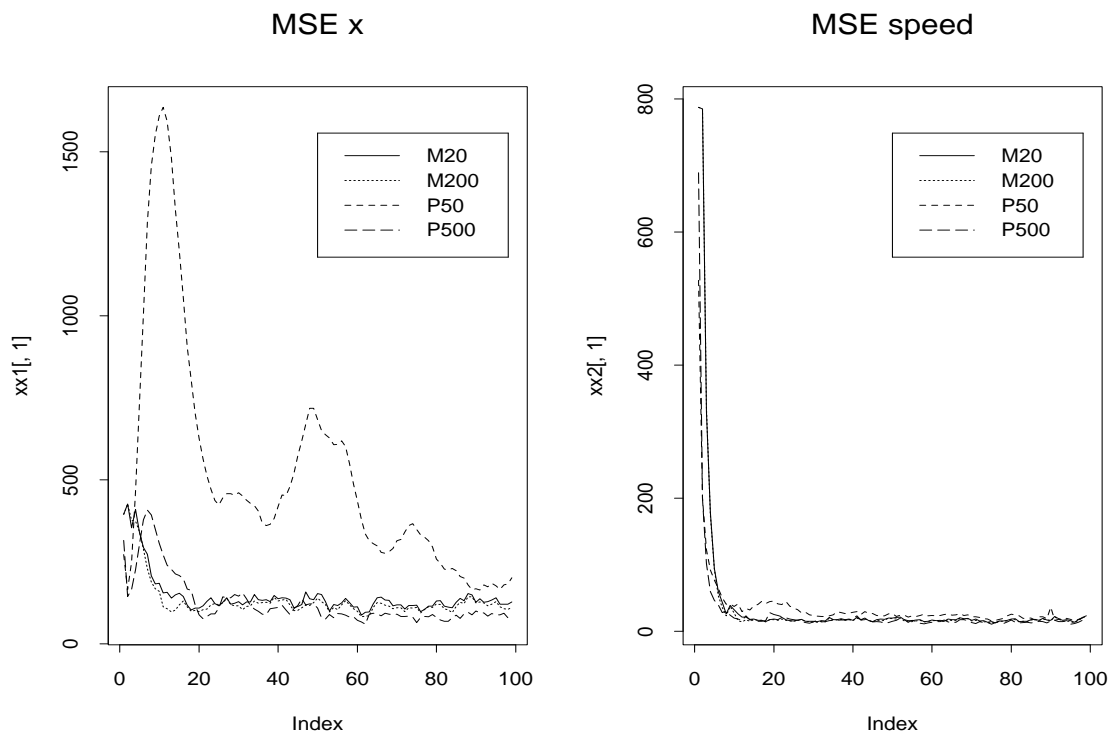


Figure 3: The MSE's 50 runs of the MKF and a standard sequential importance sampler for a simulated target moving system with different Monte Carlo sample size

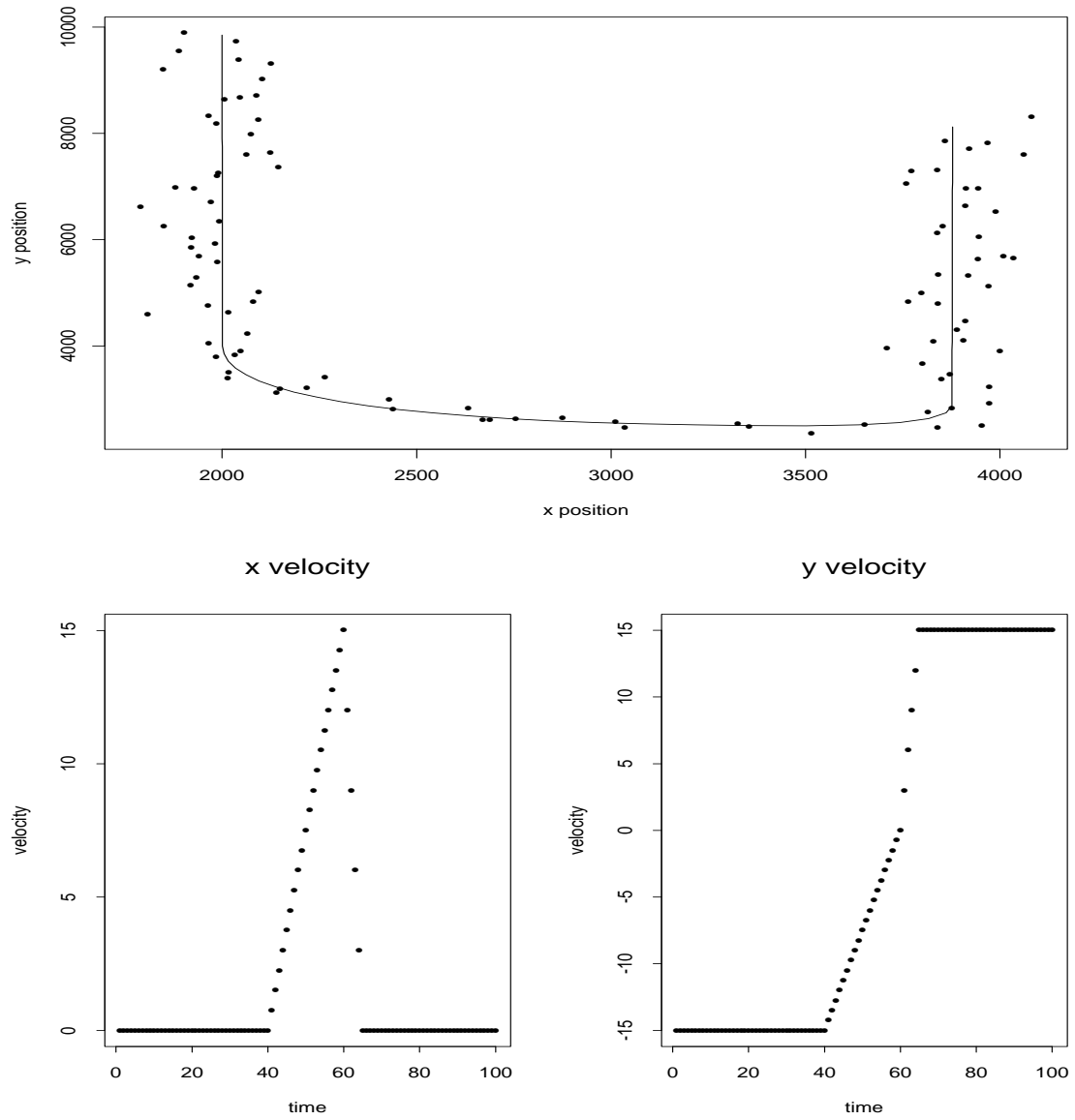


Figure 4: The position and velocity of a simulated 2 dimensional maneuvering target. (Top) Position. (Bottom) Velocity

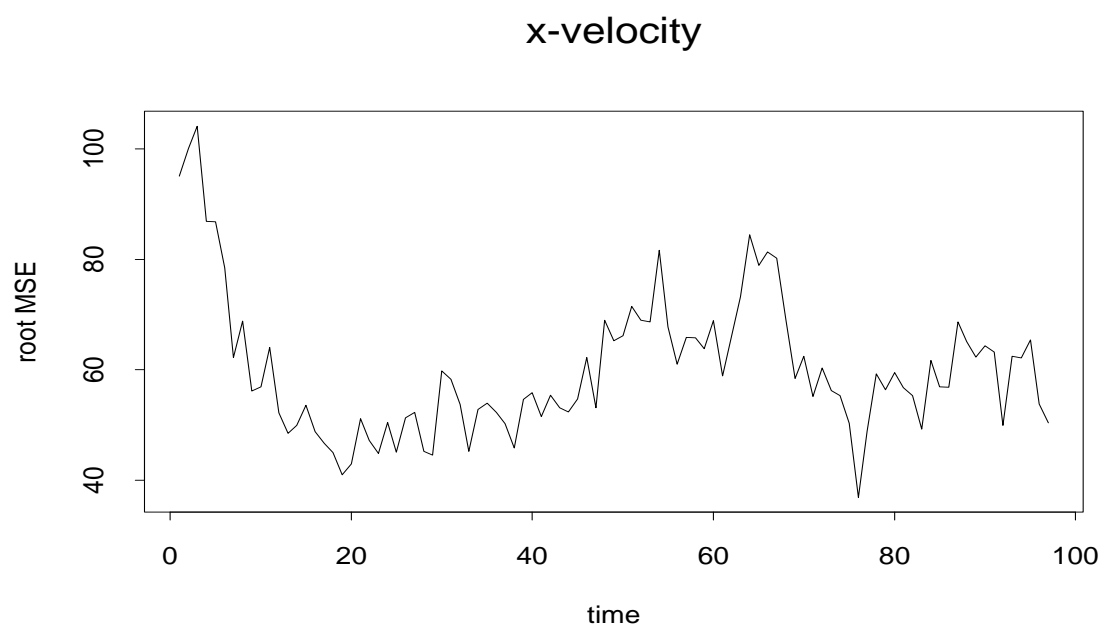
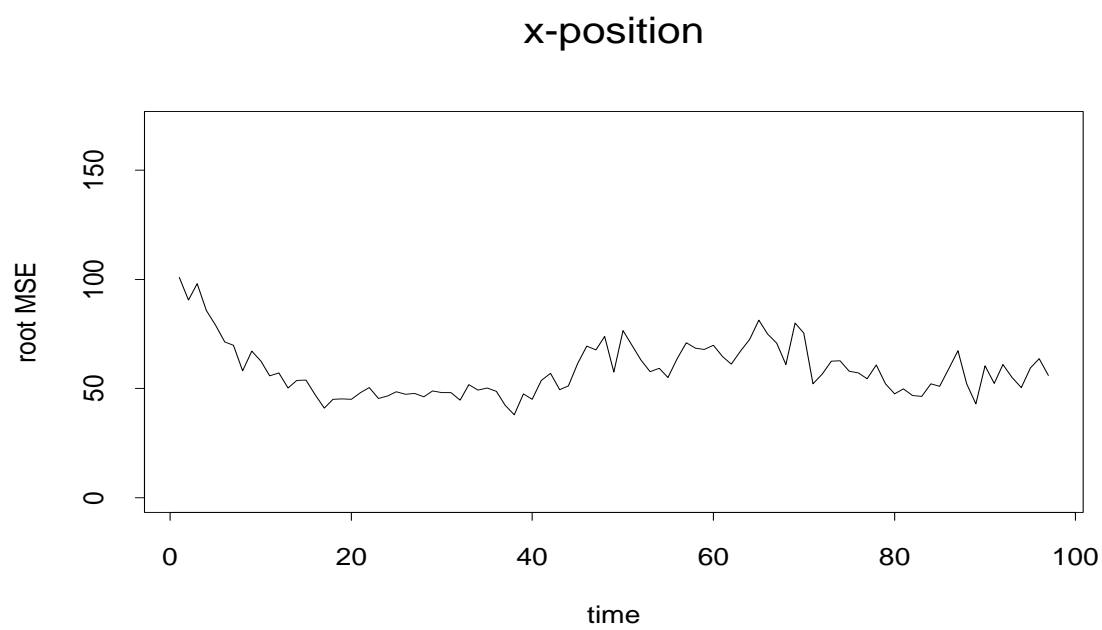


Figure 5: The root MSE's 50 runs of MKF for a simulated target moving system with maneuvering

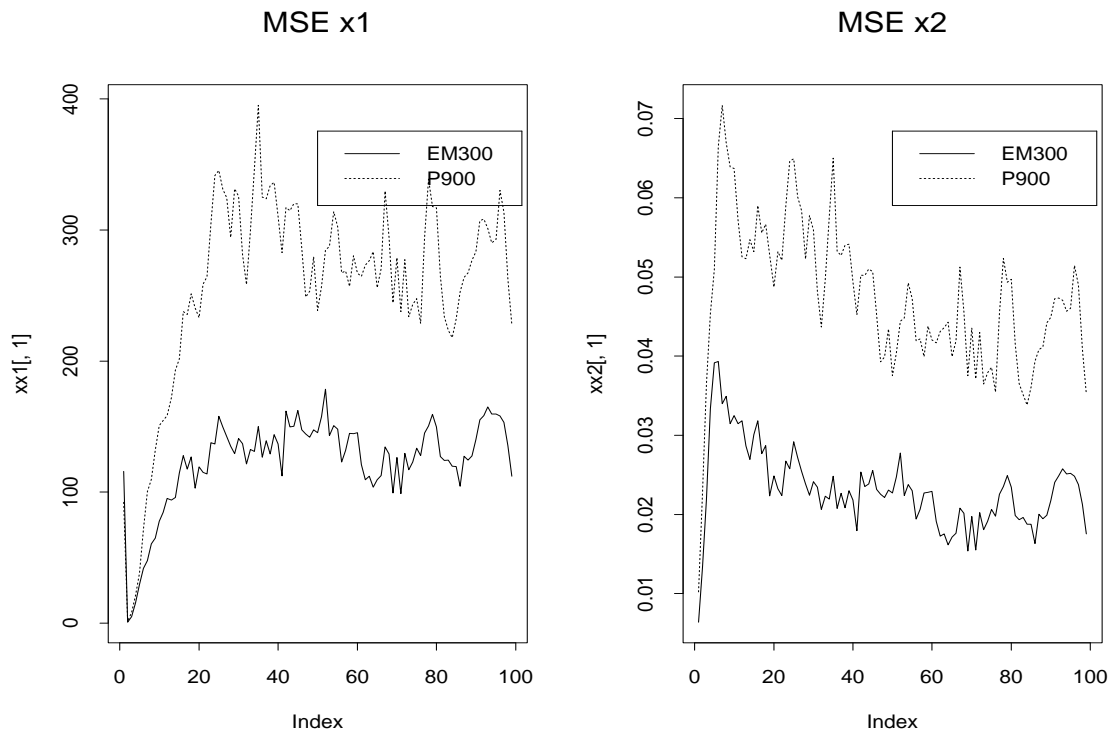


Figure 6: The MSEs of 50 runs of the EMKF and the standard sequential importance sampler for a simulated PCDLM.

Bit Error Rate Comparison

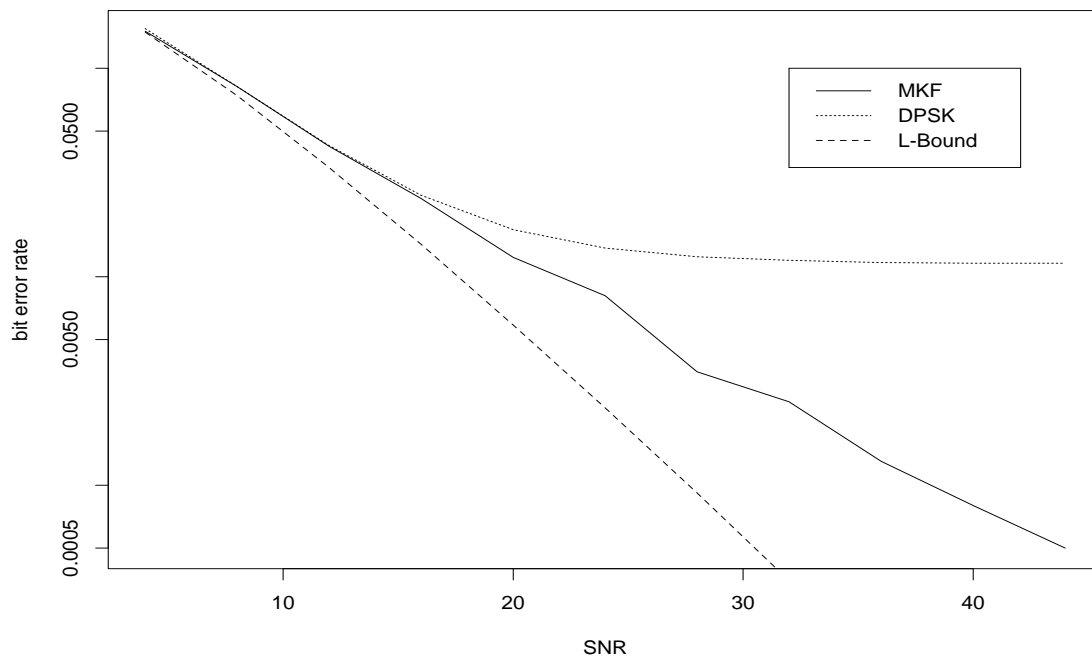


Figure 7: The bit error rate of extracting differential binary signals from a fading channel using MKF and DPSK. A lower bound that assumes the exact knowledge of the fading coefficients is also shown.