

Particle Filtering in Signal Processing

Guest Editors: Petar M. Djuri, Simon J. Godsill, and Arnaud Doucet



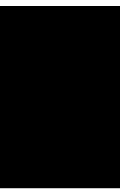
EURASIP Journal on Applied Signal Processing

Particle Filtering in Signal Processing

EURASIP Journal on Applied Signal Processing

Particle Filtering in Signal Processing

Guest Editors: Petar M. Djurić, Simon J. Godsill,
and Arnaud Doucet



Copyright © 2004 Hindawi Publishing Corporation. All rights reserved.

This is a special issue published in volume 2004 of "EURASIP Journal on Applied Signal Processing." All articles are open access articles distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Editor-in-Chief

Marc Moonen, Belgium

Senior Advisory Editor

K. J. Ray Liu, College Park, USA

Associate Editors

Gonzalo Arce, USA

Jaakko Astola, Finland

Kenneth Barner, USA

Mauro Barni, Italy

Sankar Basu, USA

Jacob Benesty, Canada

Helmut Bölcskei, Switzerland

Joe Chen, USA

Chong-Yung Chi, Taiwan

M. Reha Civanlar, Turkey

Luciano Costa, Brazil

Satya Dharanipragada, USA

Petar M. Djurić, USA

Jean-Luc Dugelay, France

Touradj Ebrahimi, Switzerland

Frank Ehlers, Germany

Moncef Gabbouj, Finland

Sharon Gannot, Israel

Fulvio Gini, Italy

A. Gorokhov, The Netherlands

Peter Handel, Sweden

Ulrich Heute, Germany

John Homer, Australia

Jiri Jan, Czech

Søren Holdt Jensen, Denmark

Mark Kahrs, USA

Thomas Kaiser, Germany

Moon Gi Kang, Korea

Aggelos Katsaggelos, USA

Walter Kellermann, Germany

Alex Kot, Singapore

C.-C. Jay Kuo, USA

Chin-Hui Lee, USA

Sang Uk Lee, Korea

Geert Leus, The Netherlands

Mark Liao, Taiwan

Yuan-Pei Lin, Taiwan

Bernie Mulgrew, UK

King N. Ngan, Hong Kong

Douglas O'Shaughnessy, Canada

Antonio Ortega, USA

Montse Pardas, Spain

Vincent Poor, USA

Phillip Regalia, France

Markus Rupp, Austria

Hideaki Sakai, Japan

Bill Sandham, UK

Dirk Slock, France

Piet Sommen, The Netherlands

John Sorensen, Denmark

Sergios Theodoridis, Greece

Dimitrios Tzovaras, Greece

Jacques Verly, Belgium

Xiaodong Wang, USA

Douglas Williams, USA

Xiang-Gen Xia, USA

Jar-Ferr Yang, Taiwan

Contents

Editorial, Petar M. Djurić, Simon J. Godsill, and Arnaud Doucet

Volume 2004 (2004), Issue 15, Pages 2239-2241

Global Sampling for Sequential Filtering over Discrete State Space, Pascal Cheung-Mon-Chan and Eric Moulines

Volume 2004 (2004), Issue 15, Pages 2242-2254

Multilevel Mixture Kalman Filter, Dong Guo, Xiaodong Wang, and Rong Chen

Volume 2004 (2004), Issue 15, Pages 2255-2266

Resampling Algorithms for Particle Filters: A Computational Complexity Perspective, Miodrag Bolić, Petar M. Djurić, and Sangjin Hong

Volume 2004 (2004), Issue 15, Pages 2267-2277

A New Class of Particle Filters for Random Dynamic Systems with Unknown Statistics,

Joaquín Míguez, Mónica F. Bugallo, and Petar M. Djurić

Volume 2004 (2004), Issue 15, Pages 2278-2294

A Particle Filtering Approach to Change Detection for Nonlinear Systems, Babak Azimi-Sadjadi and P. S. Krishnaprasad

Volume 2004 (2004), Issue 15, Pages 2295-2305

Particle Filtering for Joint Symbol and Code Delay Estimation in DS Spread Spectrum Systems in Multipath Environment, Elena Punsakaya, Arnaud Doucet, and William J. Fitzgerald

Volume 2004 (2004), Issue 15, Pages 2306-2314

Particle Filtering Equalization Method for a Satellite Communication Channel, Stéphane Sénécal, Pierre-Olivier Amblard, and Laurent Cavazzana

Volume 2004 (2004), Issue 15, Pages 2315-2327

Channel Tracking Using Particle Filtering in Unresolvable Multipath Environments, Tanya Bertozzi, Didier Le Ruyet, Cristiano Panazio, and Han Vu Thien

Volume 2004 (2004), Issue 15, Pages 2328-2338

Joint Tracking of Manoeuvring Targets and Classification of Their Manoeuvrability, Simon Maskell

Volume 2004 (2004), Issue 15, Pages 2339-2350

Bearings-Only Tracking of Manoeuvring Targets Using Particle Filters, M. Sanjeev Arulampalam, B. Ristic, N. Gordon, and T. Mansell

Volume 2004 (2004), Issue 15, Pages 2351-2365

Time-Varying Noise Estimation for Speech Enhancement and Recognition Using Sequential Monte Carlo Method, Kaisheng Yao and Te-Won Lee

Volume 2004 (2004), Issue 15, Pages 2366-2384

Particle Filtering Applied to Musical Tempo Tracking, Stephen W. Hainsworth and Malcolm D. Macleod

Volume 2004 (2004), Issue 15, Pages 2385-2395

Editorial

Petar M. Djurić

Department of Electrical and Computer Engineering, Stony Brook University, Stony Brook, NY 11794, USA
Email: djuric@ece.sunysb.edu

Simon J. Godsill

Department of Engineering, University of Cambridge, Cambridge CB2 1PZ, UK
Email: sjg@eng.cam.ac.uk

Arnaud Doucet

Department of Engineering, University of Cambridge, Cambridge CB2 1PZ, UK
Email: ad2@eng.cam.ac.uk

In most problems of sequential signal processing, measured or received data are processed in real time. Typically, the data are modeled by state-space models with linear or nonlinear unknowns and noise sources that are assumed either Gaussian or non-Gaussian. When the models describing the data are linear and the noise is Gaussian, the optimal solution is the renowned Kalman filter. For models that deviate from linearity and Gaussianity, many different methods exist, of which the best known perhaps is the extended Kalman filter.

About a decade ago, Gordon et al. published an article on nonlinear and non-Gaussian state estimation that captured much attention of the signal processing community [1]. The article introduced a method for sequential signal processing based on Monte Carlo sampling and showed that the method may have profound potential. Not surprisingly, it has incited a great deal of research, which has contributed to making sequential signal processing by Monte Carlo methods one of the most prominent developments in statistical signal processing in the recent years.

The underlying idea of the method is the approximation of posterior densities by discrete random measures. The measures are composed of samples from the states of the unknowns and of weights associated with the samples. The samples are usually referred to as particles, and the process of updating the random measures with the arrival of new data as particle filtering. One may view particle filtering as exploration of the space of unknowns with random grids whose nodes are the particles. With the acquisition of new data, the random grids evolve and their nodes are assigned weights to approximate optimally the desired densities. The assignment of new weights is carried out recursively and is based on Bayesian importance sampling theory.

The beginnings of particle filtering can be traced back to the late 1940s and early 1950s, which were followed in the last fifty years with sporadic outbreaks of intense activity [2]. Although its implementation is computationally intensive, the widespread availability of fast computers and the amenability of the particle filtering methods for parallel implementation make them very attractive for solving difficult signal processing problems.

The papers of the special issue may be arranged into four groups, that is, papers on (1) general theory, (2) applications of particle filtering to target tracking, (3) applications of particle filtering to communications, and (4) applications of particle filtering to speech and music processing. In this issue, we do not have tutorials on particle filtering, and instead, we refer the reader to some recent references [3, 4, 5, 6].

General theory

In the first paper, “Global sampling for sequential filtering over discrete state space,” Cheung-Mon-Chan and Moulines study conditionally Gaussian linear state-space models, which, when conditioned on a set of indicator variables taking values in a finite set, become linear and Gaussian. In this paper, the authors propose a global sampling algorithm for such filters and compare them with other state-of-the-art implementations.

Guo et al. in “Multilevel mixture Kalman filter” propose a new Monte Carlo sampling scheme for implementing the mixture Kalman filter. The authors use a multilevel structure of the space for the indicator variables and draw samples in a multilevel fashion. They begin with sampling from the highest-level space and follow up by drawing samples from associate subspaces from lower-level spaces. They

demonstrate the method on examples from wireless communication.

In the third paper, “Resampling algorithms for particle filters: A computational complexity perspective,” Bolić et al. propose and analyze new resampling algorithms for particle filters that are suitable for real-time implementation. By decreasing the number of operations and memory access, the algorithms reduce the complexity of both hardware and DSP realization. The performance of the algorithms is evaluated on particle filters applied to bearings-only tracking and joint detection and estimation in wireless communications.

In “A new class of particle filters for random dynamic systems with unknown statistics,” Míguez et al. propose a new class of particle filtering methods that do not assume explicit mathematical forms of the probability distributions of the noise in the system. This implies simpler, more robust, and more flexible particle filters than the standard particle filters. The performance of these filters is shown on autonomous positioning of a vehicle in a 2-dimensional space.

Finally, in “A particle filtering approach to change detection for nonlinear systems,” Azimi-Sadjadi and Krishnaprasad present a particle filtering method for change detection in stochastic systems with nonlinear dynamics based on a statistic that allows for recursive computation of likelihood ratios. They use the method in an Inertial Navigation System/Global Positioning System application.

Applications in communications

In “Particle filtering for joint symbol and code delay estimation in DS spread spectrum systems in multipath environment,” Punsakaya et al. develop receivers based on several algorithms that involve both deterministic and randomized schemes. They test their method against other deterministic and stochastic procedures by means of extensive simulations.

In the second paper, “Particle filtering equalization method for a satellite communication channel,” Sénécal et al. propose a particle filtering method for inline and blind equalization of satellite communication channels and restoration of the transmitted messages. The performance of the algorithms is presented by bit error rates as functions of signal-to-noise ratio.

Bertozzi et al. in “Channel tracking using particle filtering in unresolvable multipath environments,” propose a new timing error detector for timing tracking loops of Rake receivers in spread spectrum systems. In their scheme, the delays of each path of the frequency-selective channels are estimated jointly. Their simulation results demonstrate that the proposed scheme has better performance than the one based on conventional early-late gate detectors in indoor scenarios.

Applications to target tracking

In “Joint tracking of manoeuvring targets and classification of their manoeuvrability,” by Maskell, semi-Markov models are used to describe the behavior of manoeuvring targets. The author proposes an architecture that allows particle filters to be robust and efficient when they jointly track and classify targets. He also shows that with his approach, one can classify targets on the basis of their maneuverability.

In the other paper, “Bearings-only tracking of manoeuvring targets using particle filters,” Arulampalam et al. investigate the problem of bearings-only tracking of maneuvering targets. They formulate the problem in the framework of a multiple-model tracking problem in jump Markov systems and propose three different particle filters. They conduct extensive simulations and show that their filters outperform the trackers based on standard interacting multiple models.

Applications to speech and music

In “Time-varying noise estimation for speech enhancement and recognition using sequential Monte Carlo method,” Yao and Lee develop particle filters for sequential estimation of time-varying mean vectors of noise power in the log-spectral domain, where the noise parameters evolve according to a random walk model. The authors demonstrate the performance of the proposed filters in automated speech recognition and speech enhancement, respectively.

Hainsworth and Macleod in “Particle filtering applied to musical tempo tracking” aim at estimating the time-varying tempo process in musical audio analysis. They present two algorithms for generic beat tracking that can be used across a variety of musical styles. The authors have tested the algorithms on a large database and have discussed existing problems and directions for further improvement of the current methods.

In summary, this special issue provides some interesting theoretical developments in particle filtering theory and novel applications in communications, tracking, and speech/music signal processing. We hope that these papers will not only be of immediate use to practitioners and theoreticians but will also instigate further development in the field. Lastly, we thank the authors for their contributions and the reviewers for their valuable comments and criticism.

Petar M. Djurić
Simon J. Godsill
Arnaud Doucet

REFERENCES

- [1] N. J. Gordon, D. J. Salmond, and A. F. M. Smith, “Novel approach to nonlinear/non-Gaussian Bayesian state estimation,” *IEEE Proceedings Part F: Radar and Signal Processing*, vol. 140, no. 2, pp. 107–113, 1993.
- [2] J. S. Liu, *Monte Carlo Strategies in Scientific Computing*, Springer, New York, NY, USA, 2001.
- [3] A. Doucet, N. de Freitas, and N. Gordon, Eds., *Sequential Monte Carlo Methods in Practice*, Springer, New York, USA, 2001.
- [4] A. Doucet, S. J. Godsill, and C. Andrieu, “On sequential Monte Carlo sampling methods for Bayesian filtering,” *Stat. Comput.*, vol. 10, no. 3, pp. 197–208, 2000.
- [5] P. M. Djurić and S. J. Godsill, Eds., “Special issue on Monte Carlo methods for statistical signal processing,” *IEEE Trans. Signal Processing*, vol. 50, no. 2, 2002.
- [6] P. M. Djurić, J. H. Kotecha, J. Zhang, et al., “Particle filtering,” *IEEE Signal Processing Magazine*, vol. 20, no. 5, pp. 19–38, 2003.

Petar M. Djurić received his B.S. and M.S. degrees in electrical engineering from the University of Belgrade in 1981 and 1986, respectively, and his Ph.D. degree in electrical engineering from the University of Rhode Island in 1990. From 1981 to 1986, he was a Research Associate with the Institute of Nuclear Sciences, Vinca, Belgrade. Since 1990, he has been with Stony Brook University, where he is a Professor in the Department of Electrical and Computer Engineering. He works in the area of statistical signal processing, and his primary interests are in the theory of modeling, detection, estimation, and time series analysis and its application to a wide variety of disciplines including wireless communications and biomedicine.



Simon J. Godsill is a Reader in statistical signal processing in the Department of Engineering, Cambridge University. He is an Associate Editor for IEEE Transactions on Signal Processing and the Journal of Bayesian Analysis, and is a Member of IEEE Signal Processing Theory and Methods Committee. He has research interests in Bayesian and statistical methods for signal processing, Monte Carlo algorithms for Bayesian problems, modelling and enhancement of audio and musical signals, tracking, and genomic signal processing. He has published extensively in journals, books, and conferences. He has coedited in 2002 a special issue of IEEE Transactions on Signal Processing on Monte Carlo methods in signal processing and organized many conference sessions on related themes.



Arnaud Doucet was born in France on the 2nd of November 1970. He graduated from Institut National des Telecommunications in June 1993 and obtained his Ph.D. degree from Université Paris-Sud Orsay in December 1997. From January 1998 to February 2001 he was a research associate in Cambridge University. From March 2001 to August 2002, he was a Senior Lecturer in the Department of Electrical Engineering, Melbourne University, Australia. Since September 2002, he has been a University Lecturer in information engineering at Cambridge University. His research interests include simulation-based methods and their applications to Bayesian statistics and control.



Global Sampling for Sequential Filtering over Discrete State Space

Pascal Cheung-Mon-Chan

*École Nationale Supérieure des Télécommunications, 46 rue Barrault, 75634 Paris Cédex 13, France
Email: pcheung@tsi.enst.fr*

Eric Moulines

*École Nationale Supérieure des Télécommunications, 46 rue Barrault, 75634 Paris Cédex 13, France
Email: moulines@tsi.enst.fr*

Received 21 June 2003; Revised 22 January 2004

In many situations, there is a need to approximate a sequence of probability measures over a growing product of finite spaces. Whereas it is in general possible to determine analytic expressions for these probability measures, the number of computations needed to evaluate these quantities grows exponentially thus precluding real-time implementation. Sequential Monte Carlo techniques (SMC), which consist in approximating the flow of probability measures by the empirical distribution of a finite set of *particles*, are attractive techniques for addressing this type of problems. In this paper, we present a simple implementation of the sequential importance sampling/resampling (SISR) technique for approximating these distributions; this method relies on the fact that, the space being finite, it is possible to consider every offspring of the trajectory of particles. The procedure is straightforward to implement, and well-suited for practical implementation. A limited Monte Carlo experiment is carried out to support our findings.

Keywords and phrases: particle filters, sequential importance sampling, sequential Monte Carlo sampling, sequential filtering, conditionally linear Gaussian state-space models, autoregressive models.

1. INTRODUCTION

State-space models have been around for quite a long time to model dynamic systems. State-space models are used in a variety of fields such as computer vision, financial data analysis, mobile communication, radar systems, among others. A main challenge is to design efficient methods for online estimation, prediction, and smoothing of the hidden state given the continuous flow of observations from the system. Except in a few special cases, including linear state-space models (see [1]) and hidden finite-state Markov chain (see [2]), this problem does not admit computationally tractable exact solutions.

From the mid 1960s, considerable research efforts have been devoted to develop computationally efficient methods to approximate these distributions; in the last decade, a great deal of attention has been devoted to sequential Monte Carlo (SMC) algorithms (see [3] and the references therein). The basic idea of SMC method consists in approximating the conditional distribution of the hidden state with the empirical distribution of a set of random points, called *particles*. These particles can either give birth to offspring particles or die,

depending on their ability to represent the distribution of the hidden state conditional on the observations. The main difference between the different implementations of the SMC algorithms depends on the way this population of particles evolves in time. It is no surprise that most of the efforts in this field has been dedicated to finding numerically efficient and robust methods, which can be used in real-time implementations.

In this paper, we consider a special case of state-space model, often referred to in the literature as *conditionally Gaussian linear state-space models* (CGLSSMs), which has received a lot of attention in the recent years (see, e.g., [4, 5, 6, 7]). The main feature of a CGLSSM is that, conditionally on a set of indicator variables, here taking their values in a finite set, the system becomes linear and Gaussian. Efficient recursive procedures—such as the Kalman filter/smoothing—are available to compute the distribution of the state variable conditional on the indicator variable and the observations. By embedding these algorithms in the sequential importance sampling/resampling (SISR) framework, it is possible to derive computationally efficient sampling procedures which focus their attention on the space of indicator variables.

These algorithms are collectively referred to as *mixture Kalman filters (MKFs)*, a term first coined by Chen and Liu [8] who have developed a generic sampling algorithm; closely related ideas have appeared earlier in the automatic control/signal processing and computational statistics literature (see, e.g., [9, 10] for early work in this field; see [5] and the references therein for a tutorial on these methods; see [3] for practical implementations of these techniques). Because these sampling procedures operate on a lower-dimensional space, they typically achieve lower Monte Carlo variance than “plain” particle filtering methods.

In the CGLSSM considered here, it is assumed that the indicator variables are discrete and take a finite number of different values. It is thus feasible to consider every possible offspring of a trajectory, defined here as a particular realization of a sequence of indicator variables from initial time 0 to the current time t . This has been observed by the authors in [5, 7, 8], among many others, who have used this property to design appropriate proposal distributions for improving the accuracy and performance of SISR procedures.

In this work, we use this key property in a different way, along the lines drawn in [11, Section 3]; the basic idea consists in considering the population of every possible offspring of every trajectory and *globally* sampling from this population. This algorithm is referred to as the *global sampling (GS)* algorithm. This algorithm can be seen as a simple implementation of the SISR algorithm for the so-called *optimal* importance distribution.

Some limited Monte Carlo experiments on prototypical examples show that this algorithm compares favorably with state-of-the-art implementation of MKF; in a joint symbol estimation and channel equalization task, we have in particular achieved extremely encouraging performance with as few as 5 particles, making the proposed algorithm amenable to real-time applications.

2. SEQUENTIAL MONTE CARLO ALGORITHMS

2.1. Notations and definitions

Before going further, some additional definitions and notations are required. Let \mathbf{X} (resp., \mathbf{Y}) be a general set and let $\mathcal{B}(\mathbf{X})$ (resp., $\mathcal{B}(\mathbf{Y})$) denote a σ -algebra on \mathbf{X} (resp., \mathbf{Y}). If Q is a nonnegative function on $\mathbf{X} \times \mathcal{B}(\mathbf{Y})$ such that

- (i) for each $B \in \mathcal{B}(\mathbf{Y})$, $Q(\cdot, B)$ is a nonnegative measurable function on \mathbf{X} ,
- (ii) for each $x \in \mathbf{X}$, $Q(x, \cdot)$ is a measure on $\mathcal{B}(\mathbf{Y})$,

then we call Q a transition kernel from $(\mathbf{X}, \mathcal{B}(\mathbf{X}))$ to $(\mathbf{Y}, \mathcal{B}(\mathbf{Y}))$ and we denote $Q : (\mathbf{X}, \mathcal{B}(\mathbf{X})) \prec (\mathbf{Y}, \mathcal{B}(\mathbf{Y}))$. If for each $x \in \mathbf{X}$, $Q(x, \cdot)$ is a finite measure on $(\mathbf{Y}, \mathcal{B}(\mathbf{Y}))$, then we say that the transition is finite. If for all $x \in \mathbf{X}$, $Q(x, \cdot)$ is a probability measure on $(\mathbf{Y}, \mathcal{B}(\mathbf{Y}))$, then Q is said to be a *Markov transition kernel*.

Denote by $\mathcal{B}(\mathbf{X}) \otimes \mathcal{B}(\mathbf{Y})$ the product σ -algebra (the smallest σ -algebra containing all the sets $A \times B$, where $A \in \mathcal{B}(\mathbf{X})$ and $B \in \mathcal{B}(\mathbf{Y})$). If μ is a measure on $(\mathbf{X}, \mathcal{B}(\mathbf{X}))$ and Q is a transition kernel, $Q : (\mathbf{X}, \mathcal{B}(\mathbf{X})) \prec (\mathbf{Y}, \mathcal{B}(\mathbf{Y}))$, we denote

by $\mu \otimes Q$ the measure on the product space $(\mathbf{X} \times \mathbf{Y}, \mathcal{B}(\mathbf{X}) \otimes \mathcal{B}(\mathbf{Y}))$ defined by

$$\mu \otimes Q(A \times B) = \int_A \mu(dx) Q(x, B) \quad \forall A \in \mathcal{B}(\mathbf{X}), B \in \mathcal{B}(\mathbf{Y}). \quad (1)$$

Let $X : (\Omega, \mathcal{F}) \rightarrow (\mathbf{X}, \mathcal{B}(\mathbf{X}))$ and $Y : (\Omega, \mathcal{F}) \rightarrow (\mathbf{Y}, \mathcal{B}(\mathbf{Y}))$ be two random variables and μ and ν two measures on $(\mathbf{X}, \mathcal{B}(\mathbf{X}))$ and $(\mathbf{Y}, \mathcal{B}(\mathbf{Y}))$, respectively. Assume that the probability distribution of (X, Y) has a density denoted by $f(x, y)$ with respect to $\mu \otimes \nu$. We denote by $f(y|x) = f(x, y) / \int_{\mathbf{Y}} f(x, y) \nu(dy)$ the conditional density of Y given X .

2.2. Sequential importance sampling

Let $\{F_t\}_{t \geq 0}$ be a sequence of probability measures on $(\mathbf{Z}^{t+1}, \mathcal{P}(\mathbf{Z})^{\otimes(t+1)})$, where $\mathbf{Z} \stackrel{\text{def}}{=} \{z_1, \dots, z_M\}$ is a finite set with cardinal equal to M . It is assumed in this section that for any $\lambda_{0:t-1} \in \mathbf{Z}^t$ such that $f_{t-1}(\lambda_{0:t-1}) = 0$, we have

$$f_t([\lambda_{0:t-1}, \lambda]) = 0 \quad \forall \lambda \in \mathbf{Z}, \quad (2)$$

where for any $\tau \geq 0$, f_τ denotes the density of F_τ with respect to the counting measure. For any $t \geq 1$, there exists a finite transition kernel $Q_t : (\mathbf{Z}^t, \mathcal{P}(\mathbf{Z})^{\otimes t}) \prec (\mathbf{Z}, \mathcal{P}(\mathbf{Z}))$ such that

$$F_t = F_{t-1} \otimes Q_t. \quad (3)$$

We denote by q_t the density of the kernel Q_t with respect to the counting measure, which can simply be expressed as

$$q_t(\lambda_{0:t-1}, \lambda) = \begin{cases} \frac{f_t([\lambda_{0:t-1}, \lambda])}{f_{t-1}(\lambda_{0:t-1})} & \text{if } f_{t-1}(\lambda_{0:t-1}) \neq 0, \\ 0 & \text{otherwise.} \end{cases} \quad (4)$$

In the SIS framework (see [5, 8]), the probability distribution F_t on \mathbf{Z}^{t+1} is approximated by particles $(\Lambda^{(1,t)}, \dots, \Lambda^{(N,t)})$ associated to nonnegative weights $(w^{(1,t)}, \dots, w^{(N,t)})$; the estimator of the probability measure associated to this weighted particle system is given by

$$F_t^N = \frac{\sum_{i=1}^N w^{(i,t)} \delta_{\Lambda^{(i,t)}}}{\sum_{i=1}^N w^{(i,t)}}. \quad (5)$$

These trajectories and weights are obtained by drawing N independent trajectories $\Lambda^{(i,t)}$ under an instrumental probability distribution G_t on $(\mathbf{Z}^{t+1}, \mathcal{P}(\mathbf{Z})^{\otimes(t+1)})$ and computing the importance weights as

$$w^{(i,t)} = \frac{f_t(\Lambda^{(i,t)})}{g_t(\Lambda^{(i,t)})}, \quad i \in \{1, \dots, N\}, \quad (6)$$

where g_t is the density of the probability measure G_t with respect to the counting measure on $(\mathbf{Z}^{t+1}, \mathcal{P}(\mathbf{Z})^{\otimes(t+1)})$. It is assumed that for each t , F_t is absolutely continuous with respect to the instrumental probability G_t , that is, for all $\lambda_{0:t} \in \mathbf{Z}^{t+1}$ such that $g_t(\lambda_{0:t}) = 0$, $f_t(\lambda_{0:t}) = 0$. In the SIS

framework, these weighted trajectories are updated by drawing at each time step an offspring of each particle and then computing the associated importance weight. It is assumed in the sequel that the instrumental probability measure satisfies a decomposition similar to (3), that is,

$$G_t = G_{t-1} \otimes K_t, \quad (7)$$

where $K_t : (\mathbf{Z}^t, \mathcal{P}(\mathbf{Z})^{\otimes t}) \rightarrow (\mathbf{Z}, \mathcal{P}(\mathbf{Z}))$ is a Markov transition kernel: $\sum_{j=1}^M K_t(\lambda_{0:t-1}, \{z_j\}) = 1$. Hence, for all $\lambda_{0:t-1} \in \mathbf{Z}^t$, $\sum_{j=1}^M g_t([\lambda_{0:t-1}, z_j]) = g_{t-1}(\lambda_{0:t-1})$, showing that whenever $g_{t-1}(\lambda_{0:t-1}) = 0$, $g_t([\lambda_{0:t-1}, z_j]) = 0$ for all $j \in \{1, \dots, M\}$. Define by k_t the density of the Markov transition kernel K_t with respect to the counting measure:

$$k_t(\lambda_{0:t-1}, \lambda) = \begin{cases} \frac{g_t([\lambda_{0:t-1}, \lambda])}{g_{t-1}(\lambda_{0:t-1})} & \text{if } g_{t-1}(\lambda_{0:t-1}) \neq 0, \\ 0 & \text{otherwise.} \end{cases} \quad (8)$$

In the SIS framework, at each time t , for each particle $\Lambda^{(i,t-1)}$, $i \in \{1, \dots, N\}$, and then for each particular offspring $j \in \{1, \dots, M\}$, we evaluate the weights

$$\rho^{(i,j,t)} = k_t(\Lambda^{(i,t-1)}, z_j) \quad (9)$$

and we draw an index $J^{(i,t)}$ from a multinomial distribution with parameters $(\rho^{(i,1,t-1)}, \dots, \rho^{(i,M,t-1)})$ conditionally independently from the past:

$$\mathbb{P}[J^{(i,t)} = j \mid \mathcal{G}_{t-1}] = \rho^{(i,j,t)}, \quad i \in \{1, \dots, N\}, j \in \{1, \dots, M\}, \quad (10)$$

where \mathcal{G}_t is the history of the particle system at time t ,

$$\mathcal{G}_t = \sigma((\Lambda^{(j,\tau)}, w^{(j,\tau)}), 1 \leq j \leq N, 1 \leq \tau \leq t). \quad (11)$$

The updated system of particles then is

$$\Lambda^{(i,t)} = [\Lambda^{(i,t-1)}, z_{J^{(i,t)}}]. \quad (12)$$

If $(\Lambda^{(1,0)}, \dots, \Lambda^{(N,0)})$ is an independent sample from the distribution G_0 , it is then easy to see that at each time t , the particles $(\Lambda^{(1,t)}, \dots, \Lambda^{(N,t)})$ are independent and distributed according to G_t ; the associated (unnormalized) importance weights $w^{(i,t)} = f_i(\Lambda^{(i,t)})/g_t(\Lambda^{(i,t)})$ can be written as a product $w^{(i,t)} = u_t(\Lambda^{(i,t-1)}, z_{J^{(i,t)}})w^{(i,t-1)}$, where the incremental weight $u_t(\Lambda^{(i,t-1)}, z_{J^{(i,t)}})$ is given by

$$u_t([\lambda_{0:t-1}, \lambda]) \stackrel{\text{def}}{=} \frac{q_t(\lambda_{0:t-1}, \lambda)}{k_t(\lambda_{0:t-1}, \lambda)} \quad \forall \lambda_{0:t-1} \in \mathbf{Z}^t, \lambda \in \mathbf{Z}. \quad (13)$$

It is easily shown that the instrumental distribution k_t which minimizes the variance of the importance weights conditionally to the history of the particle system (see [5, Proposition 2]) is given by

$$k_t(\lambda_{0:t-1}, \cdot) = \frac{q_t(\lambda_{0:t-1}, \cdot)}{\sum_{j=1}^M q_t(\lambda_{0:t-1}, z_j)} \quad \text{for any } \lambda_{0:t-1} \in \mathbf{Z}^t. \quad (14)$$

The choice of the optimal instrumental distribution (14) has been introduced in [12] and has since then been used and/or rediscovered by many authors (see [5, Section II-D] for a discussion and extended references). Using this particular form of the importance kernel, the incremental importance sampling weights (13) are given by

$$u_t(\Lambda^{(i,t-1)}, z_{J^{(i,t)}}) = \sum_{j=1}^M q_t(\Lambda^{(i,t-1)}, z_j), \quad i \in \{1, \dots, N\}. \quad (15)$$

It is worthwhile to note that $u_t([\Lambda^{(i,t-1)}, z_j]) = u_t([\Lambda^{(i,t-1)}, z_l])$ for all $j, l \in \{1, \dots, M\}$; the incremental importance weights do not depend upon the particular offspring of the particle which is drawn.

2.3. Sequential importance sampling/resampling

The normalized importance weights $\tilde{w}^{(i,t)} \stackrel{\text{def}}{=} w^{(i,t)} / \sum_{i=1}^N w^{(i,t)}$ reflect the contribution of the imputed trajectories to the importance sampling estimate F_t^N . A weight close to zero indicates that the associated trajectory has a ‘‘small’’ contribution. Such trajectories are thus ineffective and should be eliminated.

Resampling is the method usually employed to combat the degeneracy of the system of particles. Let $[\Lambda^{(1,t-1)}, \dots, \Lambda^{(N,t-1)}]$ be a set of particles at time $t-1$ and let $[w^{(1,t-1)}, \dots, w^{(N,t-1)}]$ be the associated importance weights. An SISR iteration, in its most elementary form, produces a set of particles $[\Lambda^{(1,t)}, \dots, \Lambda^{(N,t)}]$ with equal weights $1/N$. The SISR algorithm is a two-step procedure. In the first step, each particle is updated according to the importance transition kernel k_t and the incremental importance weights are computed according to (12) and (13), exactly as in the SIS algorithm. This produces an intermediate set of particles $\tilde{\Lambda}^{(i,t)}$ with associated importance weights $\tilde{w}^{(i,t)}$ defined as

$$\begin{aligned} \tilde{\Lambda}^{(i,t)} &= [\Lambda^{(i,t-1)}, z_{J^{(i,t)}}], \\ \tilde{w}^{(i,t)} &= w^{(i,t-1)} u_t(\Lambda^{(i,t-1)}, z_{J^{(i,t)}}), \quad i \in \{1, \dots, N\}, \end{aligned} \quad (16)$$

where the random variables $J^{(i,t)}$, $i \in \{1, \dots, N\}$, are drawn conditionally independently from the past according to a multinomial distribution with parameters

$$\begin{aligned} \mathbb{P}[J^{(i,t)} = j \mid \mathcal{G}_{t-1}] &= k_t(\Lambda^{(i,t-1)}, z_j), \\ i &\in \{1, \dots, N\}, j \in \{1, \dots, M\}. \end{aligned} \quad (17)$$

We denote by $\tilde{\mathcal{S}}_t = ((\tilde{\Lambda}^{(i,t)}, \tilde{w}^{(i,t)}), i \in \{1, \dots, N\})$, this intermediate set of particles. In the second step, we resample the intermediate particle system. Resampling consists in transforming the weighted approximation of the probability measure $F_t, F_t^N = \sum_{i=1}^N \tilde{w}^{(i,t)} \delta_{\tilde{\Lambda}^{(i,t)}}$, into an unweighted one, $\tilde{F}_t^N = N^{-1} \sum_{i=1}^N \delta_{\tilde{\Lambda}^{(i,t)}}$. To avoid introducing bias during the resampling step, an unbiased resampling procedure should be used. More precisely, we draw with replacements N indices $I^{(1,t)}, \dots, I^{(N,t)}$ in such a way that $N^{(i,t)} = \sum_{k=1}^N \delta_{i, I^{(k,t)}}$,

the number of times the i th trajectory is chosen satisfies

$$\sum_{i=1}^N N^{(i,t)} = N, \quad \mathbb{E}[N^{(i,t)} \mid \tilde{\mathcal{G}}_t] = N\tilde{w}^{(i,t)} \quad (18)$$

for any $i \in \{1, \dots, N\}$,

where $\tilde{\mathcal{G}}_t$ is the history of the particle system *just before* the resampling step (see (11)), that is, $\tilde{\mathcal{G}}_t$ is the σ -algebra generated by the union of \mathcal{G}_{t-1} and $\sigma(\tilde{J}^{(1,t)}, \dots, \tilde{J}^{(N,t)})$:

$$\tilde{\mathcal{G}}_t = \mathcal{G}_{t-1} \vee \sigma(\tilde{J}^{(1,t)}, \dots, \tilde{J}^{(N,t)}). \quad (19)$$

Then, we set, for $k \in \{1, \dots, N\}$,

$$(I^{(k,t)}, J^{(k,t)}) = (I^{(k,t)}, \tilde{J}^{(I^{(k,t)},t)}), \quad (20)$$

$$\Lambda^{(k,t)} = [\Lambda^{(I^{(k,t)},t-1)}, z_{J^{(k,t)}}], \quad w^{(k,t)} = \frac{1}{N}.$$

Note that the sampling is done with replacement in the sense that the same particle can be either eliminated or copied several times in the final updated sample. We denote by $S_t = ((\Lambda^{(i,t)}, w^{(i,t)}), i \in \{1, \dots, N\})$ this set of particles.

There are several options to obtain an unbiased sample. The most obvious choice consists in drawing the N particles conditionally independently on $\tilde{\mathcal{G}}_t$ according to a multinomial distribution with normalized weights $(\tilde{w}^{(1,t)}, \dots, \tilde{w}^{(N,t)})$. In the literature, this is referred to as multinomial sampling. As a result, under multinomial sampling, the particles $\Lambda^{(i,t)}$ are conditional on $\tilde{\mathcal{G}}_t$ independent and identically distributed (i.i.d.). There are however better algorithms which reduce the added variability introduced during the sampling step (see the appendix).

This procedure is referred to as the SISR procedure. The particles with large normalized importance weights are likely to be selected and will be kept alive. On the contrary, the particles with low normalized importance weights are eliminated. Resampling provides more efficient samples of *future* states but increases sampling variation in the *past* states because it reduces the number of distinct trajectories.

The SISR algorithm with multinomial sampling defines a Markov chain on the path space. The transition kernel of this chain depends upon the choice of the proposal distribution and of the unbiased procedure used in the resampling step. These transition kernels are, except in a few special cases, involved. However, when the ‘‘optimal’’ importance distribution (14) is used in conjunction with multinomial sampling, the transition kernel has a simple and intuitive expression. As already mentioned above, the incremental weights for all the possible offsprings of a given particle are, in this case, identical; as a consequence, under multinomial sampling, the indices $I^{(k,t)}$, $k \in \{1, \dots, N\}$, are i.i.d. with multinomial distribution for all $k \in \{1, \dots, N\}$,

$$\mathbb{P}[I^{(k,t)} = i \mid \tilde{\mathcal{G}}_t] = \frac{(\sum_{j=1}^M q_t(\Lambda^{(i,t-1)}, z_j)) w^{(i,t-1)}}{\sum_{i=1}^N (\sum_{j=1}^M q_t(\Lambda^{(i,t-1)}, z_j)) w^{(i,t-1)}}, \quad i \in \{1, \dots, N\}. \quad (21)$$

Recall that, when the optimal importance distribution is used, for each particle $i \in \{1, \dots, N\}$, the random variables $\tilde{J}^{(i,t)}$, $i \in \{1, \dots, M\}$, are conditionally independent from \mathcal{G}_{t-1} and are distributed with multinomial random variable with parameters

$$\mathbb{P}[\tilde{J}^{(i,t)} = j \mid \mathcal{G}_{t-1}] = \frac{q_t(\Lambda^{(i,t-1)}, z_j)}{\sum_{j=1}^M q_t(\Lambda^{(i,t-1)}, z_j)}, \quad (22)$$

$i \in \{1, \dots, N\}, j \in \{1, \dots, M\}$.

We may compute, for $i, k \in \{1, \dots, N\}$ and $j \in \{1, \dots, M\}$,

$$\begin{aligned} \mathbb{P}[(I^{(k,t)}, J^{(k,t)}) = (i, j) \mid \mathcal{G}_{t-1}] &= \mathbb{E}[\mathbb{P}[I^{(k,t)} = i, \tilde{J}^{(i,t)} = j \mid \tilde{\mathcal{G}}_t] \mid \mathcal{G}_{t-1}] \\ &= \mathbb{E}[\mathbb{P}[I^{(k,t)} = i \mid \tilde{\mathcal{G}}_t] \mathbf{1}(\tilde{J}^{(i,t)} = j) \mid \mathcal{G}_{t-1}] \\ &= \frac{(\sum_{j=1}^M q_t(\Lambda^{(i,t-1)}, z_j)) w^{(i,t-1)}}{\sum_{i=1}^N (\sum_{j=1}^M q_t(\Lambda^{(i,t-1)}, z_j)) w^{(i,t-1)}} \\ &\quad \times \mathbb{P}[\tilde{J}^{(i,t)} = j \mid \mathcal{G}_{t-1}] \\ &= \frac{q_t(\Lambda^{(i,t-1)}, z_j) w^{(i,t-1)}}{\sum_{i=1}^N (\sum_{j=1}^M q_t(\Lambda^{(i,t-1)}, z_j)) w^{(i,t-1)}} = \tilde{w}^{(i,j,t)}, \end{aligned} \quad (23)$$

showing that the SISR algorithm is equivalent to drawing, conditionally independently from \mathcal{G}_{t-1} , N random variables out of $N \times M$ possible offsprings of the system of particles, with weights $(\tilde{w}^{(i,j,t)}, i \in \{1, \dots, N\}, j \in \{1, \dots, M\})$.

Resampling can be done at any time. When resampling is done at every time step, it is said to be systematic. In this case, the importance weights at each time t , $w^{(i,t)}$, $i \in \{1, \dots, N\}$, are all equal to $1/N$. Systematic resampling is not always recommended since resampling is costly from the computational point of view and may result in loss of statistical efficiency by introducing some additional randomness in the particle system. However, the effect of resampling is not necessarily negative because it allows to control the degeneracy of the particle systems, which has a positive impact on the quality of the estimates. Therefore, systematic resampling yields in some situations better estimates than the standard SIS procedure (without resampling); in some cases (see Section 4.2 for an illustration), it compares favorably with more sophisticated versions of the SISR algorithm, where resampling is done at random times (e.g., when the entropy or the coefficient of variations of the normalized importance weights is below a threshold).

2.4. The global sampling algorithm

When the instrumental distribution is the so-called optimal sampling distribution (14), it is possible to combine the sampling/resampling step above into a single sampling step. This idea has already been mentioned and worked out in [11, Section 3] under the name of deterministic/resample low weights (RLW) approach, yet the algorithm given below is not given explicitly in this reference.

Let $[\Lambda^{(1,t-1)}, \dots, \Lambda^{(N,t-1)}]$ be a set of particles at time $t-1$ and let $[w^{(1,t-1)}, \dots, w^{(N,t-1)}]$ be the associated importance weights. Similar to the SISR step, the GS algorithm produces

a set of particles $[\Lambda^{(1,t)}, \dots, \Lambda^{(N,t)}]$ with equal weights. The GS algorithm combines the two-stage sampling procedure (first, samples a particular offspring of a particle, updates the importance weights, and then resamples from the population) into a single one.

(i) We first compute the weights

$$w^{(i,j,t)} = w^{(i,t-1)} q_t(\Lambda^{(i,t-1)}, z_j), \quad i \in \{1, \dots, N\}, j \in \{1, \dots, M\}. \quad (24)$$

(ii) We then draw N random variables $((I^{(1,t)}, J^{(1,t)}), \dots, (I^{(N,t)}, J^{(N,t)}))$ in $\{1, \dots, N\} \times \{1, \dots, M\}$ using an unbiased sampling procedure, that is, for all $(i, j) \in \{1, \dots, N\} \times \{1, \dots, M\}$, the number of times of the particles (i, j) is

$$N^{(i,j,t)} \stackrel{\text{def}}{=} |\{k \in \{1, \dots, N\}, (I^{(k,t)}, J^{(k,t)}) = (i, j)\}| \quad (25)$$

thus satisfying the following two conditions:

$$\sum_{i'=1}^N \sum_{j'=1}^M N^{(i',j',t)} = N, \quad (26)$$

$$\mathbb{E}[N^{(i,j,t)} | \mathcal{G}_{t-1}] = N \frac{w^{(i,j,t)}}{\sum_{i'=1}^N \sum_{j'=1}^M w^{(i',j',t)}}.$$

The updated set of particles is then defined as

$$\Lambda^{(k,t)} = [\Lambda^{(I^{(k,t)}, t-1)}, z_{J^{(k,t)}}], \quad w^{(k,t)} = \frac{1}{N}. \quad (27)$$

If multinomial sampling is used, then the GS algorithm is a simple implementation of the SISR algorithm, which combines the two-stage sampling into a single one. Since the computational cost of drawing L random variables grows linearly with L , the cost of simulations is proportional to NM for the GS algorithm and $NM + N$ for the SISR algorithm. There is thus a (slight) advantage in using the GS implementation. When sampling is done using a different unbiased method (see the appendix), then there is a more substantial difference between these two algorithms. As illustrated in the examples below, the GS may outperform the SISR algorithm.

3. GLOBAL SAMPLING FOR CONDITIONALLY GAUSSIAN STATE-SPACE MODELS

3.1. Conditionally linear Gaussian state-space model

As emphasized in the introduction, CGLSSMs are a particular class of state-space models which are such that, conditional to a set of indicator variables, the system becomes linear and Gaussian. More precisely,

$$\begin{aligned} S_t &= \Psi_t(\Lambda_{0:t}), \\ \vec{X}_t &= A_{S_t} \vec{X}_{t-1} + C_{S_t} \vec{W}_t, \\ \vec{Y}_t &= B_{S_t} \vec{X}_t + D_{S_t} \vec{V}_t, \end{aligned} \quad (28)$$

where

- (i) $\{\Lambda_t\}_{t \geq 0}$ are the *indicator variables*, here assumed to take values in a finite set $\mathbf{Z} = \{z_1, z_2, \dots, z_M\}$, where M denotes the cardinal of the set \mathbf{Z} ; the law of $\{\Lambda_t\}_{t \geq 0}$ is assumed to be known but is otherwise not specified;
- (ii) for any $t \geq 0$, Ψ_t is a function $\Psi_t : \mathbf{Z}^{t+1} \rightarrow \mathbf{S}$, where \mathbf{S} is a finite set;
- (iii) $\{\vec{X}_t\}_{t \geq 0}$ are the $(n_x \times 1)$ *state vectors*; these state variables are not directly observed;
- (iv) the distribution of \vec{X}_0 is complex Gaussian with mean $\vec{\mu}_0$ and covariance Γ_0 ;
- (v) $\{\vec{Y}_t\}_{t \geq 0}$ are the $(n_y \times 1)$ *observations*;
- (vi) $\{\vec{W}_t\}_{t \geq 0}$ and $\{\vec{V}_t\}_{t \geq 0}$ are (complex) n_w - and n_v -dimensional (complex) Gaussian white noise, $\vec{W}_t \sim \mathcal{N}_c(0, I_{n_w \times n_w})$ and $\vec{V}_t \sim \mathcal{N}_c(0, I_{n_v \times n_v})$, where $I_{p \times p}$ is the $p \times p$ identity matrix; $\{\vec{W}_t\}_{t \geq 0}$ is referred to as the *state noise*, whereas $\{\vec{V}_t\}_{t \geq 0}$ is the *observation noise*;
- (vii) $\{A_s, s \in \mathbf{S}\}$ are the state transition matrices, $\{B_s, s \in \mathbf{S}\}$ are the observation matrices, and $\{C_s, s \in \mathbf{S}\}$ and $\{D_s, s \in \mathbf{S}\}$ are Cholesky factors of the covariance matrix of the state noise and measurement noise, respectively; these matrices are assumed to be known;
- (viii) the indicator process $\{\Lambda_t\}_{t \geq 0}$ and the noise observation processes $\{\vec{V}_t\}_{t \geq 0}$ and $\{\vec{W}_t\}_{t \geq 0}$ are independent.

This model has been considered by many authors, following the pioneering work in [13, 14] (see [5, 7, 8, 15] for authoritative recent surveys). Despite its simplicity, this model is flexible enough to describe many situations of interests including linear state-space models with non-Gaussian state noise or observation noise (heavy-tail noise), jump linear systems, linear state space with missing observations; of course, digital communication over fading channels, and so forth.

Our aim in this paper is to compute recursively in time an estimate of the conditional probability of the (unobserved) indicator variable Λ_n given the observation up to time $n + \Delta$, that is, $\mathbb{P}(\Lambda_n | \vec{Y}_{0:n+\Delta} = \vec{y}_{0:n+\Delta})$, where Δ is a nonnegative integer and for any sequence $\{\lambda_t\}_{t \geq 0}$ and any integer $0 \leq i < j$, we denote $\lambda_{i:j} \stackrel{\text{def}}{=} \{\lambda_i, \dots, \lambda_j\}$. When $\Delta = 0$, this distribution is called the *filtering distribution*; when $\Delta > 0$, it is called the *fixed-lag smoothing distribution*, and Δ is the lag.

3.2. Filtering

In this section, we describe the implementation of the GS algorithm to approximate the filtering probability of the indicator variables given the observations

$$f_t(\lambda_{0:t}) = \mathbb{P}[\Lambda_{0:t} = \lambda_{0:t} | Y_{0:t} = y_{0:t}] \quad (29)$$

in the CGLSSM (28). We will first show that the filtering probability F_t satisfies condition (3), that is, for any $t \geq 1$, $F_t = F_{t-1} \otimes Q_t$; we then present an efficient recursive algorithm to compute the transition kernel Q_t using the Kalman filter update equations. For any $t \geq 1$ and for any $\lambda_{0:t} \in \mathbf{Z}^{t+1}$,

under the conditional independence structure implied by the CGLSSM (28), the Bayes formula shows that

$$q_t(\lambda_{0:t-1}; \lambda_t) \propto f(\vec{y}_t | \vec{y}_{0:t-1}, \lambda_{0:t}) f(\lambda_t | \lambda_{0:t-1}). \quad (30)$$

The predictive distribution of the observations given the indicator variables $f(\vec{y}_t | \vec{y}_{0:t-1}, \lambda_{0:t})$ can be evaluated along each trajectory of indicator variables $\lambda_{0:t}$ using the Kalman filter recursions. Denote by $g_c(\cdot; \mu, \Gamma)$ the density of a complex circular Gaussian random vector with mean $\vec{\mu}$ and covariance matrix Γ , and for A a matrix, let A^\dagger be the transpose conjugate of A ; we have, with $s_t = \Psi_t(\lambda_{0:t})$ (and Ψ_t is defined in (28)),

$$\begin{aligned} f(\vec{y}_t | \lambda_{0:t}, \vec{y}_{0:t-1}) \\ = g_c(\vec{y}_t; B_{s_t} \mu_{t|t-1}[\lambda_{0:t}], B_{s_t} \Gamma_{t|t-1}[\lambda_{0:t}] B_{s_t}^\dagger + D_{s_t} D_{s_t}^\dagger), \end{aligned} \quad (31)$$

where $\vec{\mu}_{t|t-1}[\lambda_{0:t}]$ and $\Gamma_{t|t-1}[\lambda_{0:t}]$ denote the filtered mean and covariance of the state, that is, the conditional mean and covariance of the state given the indicators variables $\lambda_{0:t}$ and the observations up to time $t-1$ (the dependence of the predictive mean $\vec{\mu}_{t|t-1}[\lambda_{0:t}]$ on the observations $y_{0:t-1}$ is implicit). These quantities can be computed recursively using the following Kalman one-step prediction/correction formula. Denote by $\vec{\mu}_{t-1}([\lambda_{0:t-1}])$ and $\Gamma_{t-1}([\lambda_{0:t-1}])$ the mean and covariance of the filtering density, respectively. These quantities can be recursively updated as follows:

(i) predictive mean:

$$\vec{\mu}_{t|t-1}[\lambda_{0:t}] = A_{s_t} \vec{\mu}_{t-1}[\lambda_{0:t-1}]; \quad (32)$$

(ii) predictive covariance:

$$\Gamma_{t|t-1}[\lambda_{0:t}] = A_{s_t} \Gamma_{t-1}[\lambda_{0:t-1}] A_{s_t}^T + C_{s_t} C_{s_t}^T; \quad (33)$$

(iii) innovation covariance:

$$\Sigma_t[\lambda_{0:t}] = B_{s_t} \Gamma_{t|t-1}[\lambda_{0:t}] B_{s_t}^T + D_{s_t} D_{s_t}^T; \quad (34)$$

(iv) Kalman Gain:

$$K_t[\lambda_{0:t}] = \Gamma_{t|t-1}[\lambda_{0:t}] B_{s_t} (\Sigma_t[\lambda_{0:t}])^{-1}; \quad (35)$$

(v) filtered mean:

$$\vec{\mu}_t[\lambda_{0:t}] = \vec{\mu}_{t-1}[\lambda_{0:t-1}] + K_t[\lambda_{0:t}] (\vec{y}_t - B_{s_t} \vec{\mu}_{t-1}[\lambda_{0:t-1}]); \quad (36)$$

(vi) filtered covariance:

$$\Gamma_t[\lambda_{0:t}] = (I - K_t[\lambda_{0:t-1}] B_{s_t}) \Gamma_{t|t-1}[\lambda_{0:t-1}]. \quad (37)$$

Note that the conditional distribution of the state vector \vec{X}_t given the observations up to time t , $y_{0:t}$, is a mixture of Gaussian distributions with a number of components equal to M^{t+1} which grows exponentially with t . We have now at

hands all the necessary ingredients to derive the GS approximation of the filtering distribution. For any $t \in \mathbb{N}$ and for any $\lambda_{0:t} \in \mathbf{Z}^{t+1}$, denote

$$\gamma_t(\lambda_{0:t}) \stackrel{\text{def}}{=} \begin{cases} f(\vec{y}_0 | \lambda_0) f(\lambda_0) & \text{for } t = 0, \\ f(\vec{y}_t | \lambda_{0:t}, \vec{y}_{0:t-1}) f(\lambda_t | \lambda_{0:t-1}) & \text{for } t > 0. \end{cases} \quad (38)$$

With these notations, (30) reads $q_t(\lambda_{0:t-1}; \lambda_t) \propto \gamma_t(\lambda_{0:t})$.

The first step consists in initializing the particle tracks. For $t = 1$ and $i \in \{1, \dots, N\}$, set $\vec{\mu}^{(i,0)} = \vec{\mu}_0$ and $\Gamma^{(i,0)} = \Gamma_0$, where $\vec{\mu}_0$ and Γ_0 are the initial mean and variance of the state vector (which are assumed to be known); then, compute the weights

$$w_j = \frac{\gamma_0(z_j)}{\sum_{j'=1}^M \gamma_0(z_{j'})}, \quad j \in \{1, \dots, M\}, \quad (39)$$

and draw $\{I_i, i \in \{1, \dots, N\}\}$ in such a way that, for $j \in \{1, \dots, M\}$, $\mathbb{E}[N_j] = N w_j$, where $N_j = \sum_{i=1}^N \delta_{I_i, j}$. Then, set $\Lambda^{(i,0)} = z_{I_i}$, $i \in \{1, \dots, N\}$.

At time $t \geq 1$, assume that we have N trajectories $\Lambda^{(i,t-1)} = (\Lambda_0^{(i,t-1)}, \dots, \Lambda_{t-1}^{(i,t-1)})$ and that, for each trajectory, we have stored the filtered mean $\vec{\mu}^{(i,t-1)}$ and covariance $\Gamma^{(i,t-1)}$ defined in (36) and (37), respectively.

- (1) For $i \in \{1, \dots, N\}$ and $j \in \{1, \dots, M\}$, compute the predictive mean $\vec{\mu}_{t|t-1}[\Lambda^{(i,t-1)}, z_j]$ and covariance $\Gamma_{t|t-1}[\Lambda^{(i,t-1)}, z_j]$ using (32) and (33), respectively. Then, compute the innovation covariance $\Sigma_t[\Lambda^{(i,t-1)}, z_j]$ using (34) and evaluate the likelihood $\gamma^{(i,j,t)}$ of the particle $[\Lambda^{(i,t-1)}, z_j]$ using (31). Finally, compute the filtered mean and covariance $\vec{\mu}_t([\Lambda^{(i,t-1)}, z_j])$ and $\Gamma_t([\Lambda^{(i,t-1)}, z_j])$.
- (2) Compute the weights

$$w^{(i,j,t)} = \frac{\gamma^{(i,j,t)}}{\sum_{i'=1}^N \sum_{j'=1}^M \gamma^{(i',j',t)}}, \quad (40)$$

$$i \in \{1, \dots, N\}, j \in \{1, \dots, M\}.$$

- (3) Draw $\{(I_k, J_k), k \in \{1, \dots, N\}\}$ using an unbiased sampling procedure (see (26)) with weights $\{w^{(i,j,t)}\}$, $i \in \{1, \dots, N\}$, $j \in \{1, \dots, M\}$; set, for $k \in \{1, \dots, N\}$, $\Lambda^{(k,t)} = (\Lambda^{(I_k, t-1)}, z_{J_k})$. Store the filtered mean and covariance $\vec{\mu}_t([\Lambda^{(k,t)}])$ and $\Gamma_t([\Lambda^{(k,t)}])$ using (36) and (37), respectively.

Remark 1. From the trajectories and the computed weights it is possible to evaluate, for any $\delta \geq 0$ and $t \geq \delta$, the posterior probability of $\Lambda_{t-\delta}$ given $\vec{Y}_{0:t} = \vec{y}_{0:t}$ as

$$\begin{aligned} \hat{\mathbb{P}}[\Lambda_{t-\delta} = z_k \mid \vec{Y}_{0:t} = \vec{y}_{0:t}] \\ \propto \begin{cases} \sum_{i=1}^N w^{(i,k,t)}, & \delta = 0, \text{ filtering,} \\ \sum_{i=1}^N \left(\sum_{j=1}^M w^{(i,j,t)} \right) \delta_{\Lambda_{t-\delta}, z_k}, & \delta > 0, \text{ fixed-lag smoothing.} \end{cases} \end{aligned} \quad (41)$$

Similarly, we can approximate the filtering and the smoothing distribution of the state variable as a mixture of Gaussians. For example, we can estimate the filtered mean and variance of the state as follows:

(i) *filtered mean*:

$$\sum_{i=1}^N \sum_{j=1}^M w^{(i,j,t)} \vec{\mu}_t([\Lambda^{(i,t-1)}, z_j]); \tag{42}$$

(ii) *filtered covariance*:

$$\sum_{i=1}^N \sum_{j=1}^M w^{(i,j,t)} \Gamma_t([\Lambda^{(i,t-1)}, z_j]). \tag{43}$$

3.3. Fixed-lag smoothing

Since the state process is correlated, the future observations contain information about the current value of the state; therefore, whenever it is possible to delay the decision, fixed-lag smoothing estimates yield more reliable information on the indicator process than filtering estimates.

As pointed out above, it is possible to determine an estimate of the fixed-lag smoothing distribution for any delay δ from the trajectories and the associated weights produced by the SISR or GS method described above; nevertheless, we should be aware that this estimate can be rather poor when the delay δ is large, as a consequence of the impoverishment of the system of particles (the system of particle “forgets” its past). To address this well-known problem in all particle methods, it has been proposed by several authors (see [11, 16, 17, 18]) to sample at time t from the conditional distribution of Λ_t given $\vec{Y}_{0:t+\Delta} = \vec{y}_{0:t+\Delta}$ for some $\Delta > 0$. The computation of fixed-lag smoothing distribution is also amenable to GS approximation.

Consider the distribution of the indicator variables $\Lambda_{0:t}$ conditional to the observations $\vec{Y}_{0:t+\Delta} = \vec{y}_{0:t+\Delta}$, where Δ is a positive integer. Denote by $\{F_t^\Delta\}_{t \geq 0}$ this sequence of probability measures; the dependence on the observations $\vec{y}_{0:t+\Delta}$ being, as in the previous section, implicit. This sequence of distributions also satisfies (3), that is, there exists a finite transition kernel $Q_t^\Delta : (\mathbf{Z}^t, \mathcal{P}(\mathbf{Z})^{\otimes t}) \times (\mathbf{Z}, \mathcal{P}(\mathbf{Z}))$ such that $F_t^\Delta = F_{t-1}^\Delta \otimes Q_t^\Delta$ for all $t \geq 1$. Elementary conditional probability calculations exploiting the conditional independence structure of (28) show that the transition kernel Q_t^Δ can be determined, up to a normalization constant, by the relation

$$Q_t^\Delta(\lambda_{0:t-1}; \lambda_t) \propto \frac{\sum_{\lambda_{t+1:t+\Delta}} \prod_{\tau=t}^{t+\Delta} f(\vec{y}_\tau | \vec{y}_{0:\tau-1}, \lambda_{0:\tau}) f(\lambda_\tau | \lambda_{0:\tau-1})}{\sum_{\lambda_{t:t+\Delta-1}} \prod_{\tau=t}^{t+\Delta-1} f(\vec{y}_\tau | \vec{y}_{0:\tau-1}, \lambda_{0:\tau}) f(\lambda_\tau | \lambda_{0:\tau-1})}, \tag{44}$$

where, for all $\lambda_{0:t-1} \in \mathbf{Z}^t$, the terms $f(\vec{y}_\tau | \vec{y}_{0:\tau-1}, \lambda_{0:\tau})$ can be determined recursively using Kalman filter fixed-lag smoothing update formula.

Below, we describe a straightforward implementation of the GS method to approximate the smoothing distribution by the delayed sampling procedure; more sophisticated techniques, using early pruning of the possible prolonged trajectories, are currently under investigation. For any $t \in \mathbb{N}$ and for any $\lambda_{0:t} \in \mathbf{Z}^{t+1}$, denote

$$D_t^\Delta(\lambda_{0:t}) \stackrel{\text{def}}{=} \sum_{\lambda_{t+1:t+\Delta}} \prod_{\tau=t+1}^{t+\Delta} \gamma_\tau(\lambda_{0:\tau}), \tag{45}$$

where the function γ_τ is defined in (38). With this notation, (44) may be rewritten as

$$Q_t^\Delta(\lambda_{0:t-1}; \lambda_t) \propto \gamma_t(\lambda_{0:t}) \frac{D_t^\Delta(\lambda_{0:t})}{D_{t-1}^\Delta(\lambda_{0:t-1})}. \tag{46}$$

We now describe one iteration of the algorithm. Assume that for some time instant $t \geq 1$, we have N trajectories $\Lambda^{(j,t-1)} = (\Lambda_0^{(j,t-1)}, \dots, \Lambda_{t-1}^{(j,t-1)})$; in addition, for each trajectory $\Lambda^{(j,t-1)}$, the following quantities are stored:

- (1) the factor $D_{t-1}^\Delta(\Lambda^{(j,t-1)})$ defined in (45);
- (2) for each prolongation $\lambda_{t:\tau} \in \mathbf{Z}^{\tau-t+1}$ with $\tau \in \{t, t+1, \dots, t+\Delta-1\}$, the conditional likelihood $\gamma_\tau(\Lambda^{(j,t-1)}, \lambda_{t:\tau})$ given in (38);
- (3) for each prolongation $\lambda_{t:t+\Delta-1} \in \mathbf{Z}^\Delta$, the filtering conditional mean $\vec{\mu}_{t+\Delta-1}([\Lambda^{(j,t-1)}, \lambda_{t:t+\Delta-1}])$ and covariance $\Gamma_{t+\Delta-1}(\Lambda^{(j,t-1)}, \lambda_{t:t+\Delta-1})$.

One iteration of the algorithm is then described below.

- (1) For each $i \in \{1, \dots, N\}$ and for each $\lambda_{t:t+\Delta} \in \mathbf{Z}^{\Delta+1}$, compute the predictive conditional mean and covariance of the state, $\vec{\mu}_{t+\Delta|t+\Delta-1}([\Lambda^{(i,t-1)}, \lambda_{t:t+\Delta}])$ and $\Gamma_{t+\Delta|t+\Delta-1}([\Lambda^{(i,t-1)}, \lambda_{t:t+\Delta}])$, using (32) and (33), respectively. Then compute the innovation covariance $\Sigma_{t+\Delta}([\Lambda^{(i,t-1)}, \lambda_{t:t+\Delta}])$ using (34) and the likelihood $\gamma_{t+\Delta}(\Lambda^{(i,t-1)}, \lambda_{t:t+\Delta})$ using (31).
- (2) For each $i \in \{1, \dots, N\}$ and $j \in \{1, \dots, M\}$, compute

$$D_t^\Delta(\Lambda^{(i,t-1)}, z_j) = \sum_{\lambda_{t+1:t+\Delta}} \prod_{\tau=t+1}^{t+\Delta} \gamma_\tau([\Lambda^{(i,t-1)}, z_j, \lambda_{t+1:t+\tau}]),$$

$$y^{(i,j,t)} = \gamma_t(\Lambda^{(i,t-1)}, z_j) \frac{D_t^\Delta(\Lambda^{(i,t-1)}, z_j)}{D_{t-1}^\Delta(\Lambda^{(i,t-1)})},$$

$$w^{(i,j,t)} = \frac{y^{(i,j,t)}}{\sum_{i'=1}^M \sum_{j=1}^N y^{(i',j,t)}}. \tag{47}$$

- (3) Update the trajectory of particles using an unbiased sampling procedure $\{(I_k, J_k), k \in \{1, \dots, N\}\}$ with weights $\{w^{(i,j,t)}\}$, $i \in \{1, \dots, N\}$, $j \in \{1, \dots, M\}$, and set $\Lambda^{(k,t)} = (\Lambda^{(k,t-1)}, z_{J_k})$, $k \in \{1, \dots, N\}$.

4. SOME EXAMPLES

4.1. Autoregressive model with jumps

To illustrate how the GS method works, we consider the state-space model

$$\begin{aligned} X_t &= a_{\Lambda_t} X_{t-1} + \sigma_{\Lambda_t} \epsilon_t, \\ Y_t &= X_t + \rho \eta_t, \end{aligned} \quad (48)$$

where $\{\epsilon_t\}_{t \geq 0}$ and $\{\eta_t\}_{t \geq 0}$ are i.i.d. unit-variance Gaussian noise. We assume that $\{\Lambda_t\}_{t \geq 0}$ is an i.i.d. sequence of random variables taking their values in $\mathbf{Z} \stackrel{\text{def}}{=} \{1, 2\}$, which is independent from both $\{\epsilon_t\}_{t \geq 0}$ and $\{\eta_t\}_{t \geq 0}$, and such that $\mathbb{P}[\Lambda_0 = i] = \pi_i$, $i \in \mathbf{Z}$. This can easily be extended to deal with the Markovian case. This simple model has been dealt with, among others, in [19] and [20, Section 5.1]. We focus in this section on the filtering problem, that is, we approximate the distribution of the hidden state X_t given the observations up to time t , $Y_{0:t} = y_{0:t}$. For this model, we can carry out the computations easily. The transition kernel q_t defined in (30) is given, for all $\lambda_{0:t-1} \in \mathbf{Z}^t$, $\lambda_t \in \mathbf{Z}$, by

$$q_t(\lambda_{0:t-1}, \lambda_t) \propto \frac{\pi_{\lambda_t}}{\sqrt{2\pi\Sigma_t[\lambda_{0:t}]}} \exp\left(-\frac{(y_t - \mu_{t|t-1}[\lambda_{0:t}])^2}{2\Sigma_t[\lambda_{0:t}]}\right), \quad (49)$$

where the mean $\mu_{t|t-1}[\lambda_{0:t}]$ and covariance $\Sigma_t[\lambda_{0:t}]$ are computed recursively from the filtering mean $\mu_{t-1}([\lambda_{0:t-1}])$ and covariance $\Gamma_{t-1}([\lambda_{0:t-1}])$ according to the following one-step Kalman update equations derived from (32), (33), and (34):

(i) predictive mean:

$$\mu_{t|t-1}[\lambda_{0:t}] = a_{\lambda_t} \mu_{t-1}[\lambda_{0:t}]; \quad (50)$$

(ii) predictive covariance:

$$\Gamma_{t|t-1}[\lambda_{0:t}] = a_{\lambda_t}^2 \Gamma_{t-1}[\lambda_{0:t}] + \sigma_{\lambda_t}^2; \quad (51)$$

(iii) innovation covariance:

$$\Sigma_t[\lambda_{0:t}] = \Gamma_{t|t-1}[\lambda_{0:t}] + \rho^2; \quad (52)$$

(iv) filtered mean:

$$\begin{aligned} \mu_t[\lambda_{0:t}] &= \mu_{t|t-1}[\lambda_{0:t-1}] + \frac{\Gamma_{t|t-1}([\lambda_{0:t}])}{\Gamma_{t|t-1}([\lambda_{0:t}]) + \rho^2} \\ &\quad \times (y_t - \mu_{t|t-1}[\lambda_{0:t-1}]); \end{aligned} \quad (53)$$

(v) filtered covariance:

$$\Gamma_t[\lambda_{0:t}] = \frac{\rho^2 \Gamma_{t|t-1}([\lambda_{0:t}])}{\Gamma_{t|t-1}([\lambda_{0:t}]) + \rho^2}. \quad (54)$$

We have used the parameters (used in the experiments carried out in [20, Section 5.1]): $a_i = 0.9$ ($i = 1, 2$), $\sigma_1 = 0.5$,

$\sigma_2 = 1.5$, $\pi_1 = 1.7$, and $\rho = 0.3$, and applied the GS and the SISR algorithm for online filtering. We compare estimates of the filtered state mean using the GS and the SIS with systematic resampling. In both case, we use the estimator (42) of the filtered mean. Two different unbiased sampling strategies are used: multinomial sampling and the modified stratified sampling (detailed in the appendix).¹ In Figure 1, we have displayed the box and whisker plot² of the difference between the filtered mean estimate (42) and the true value of the state variables for $N = 5, 10, 50$ particles using multinomial sampling (Figure 1a) and the modified stratified sampling (Figure 1b). These results are obtained from 100 hundred independent Monte Carlo experiments where, for each experiment, a new set of the observations and state variables are simulated. These simulations show that, for the autoregressive model, the filtering algorithm performed reasonably well even when the number of particles is small (the difference between $N = 5$ and $N = 50$ particles is negligible; $N = 50$ particles is suggested in the literature for the same simulation setting [20]). There are no noticeable differences between the standard SISR implementation and the GS implementation of the SISR. Note that the error in the estimate is dominated by the filtered variance $\mathbb{E}[(X_t - \mathbb{E}[X_t | Y_{0:t}])^2]$; the additional variations induced by the fluctuations of the particle estimates are an order of magnitude lower than this quantity.

To visualize the difference between the different sampling schemes, it is more appropriate to consider the fluctuation of the filtered mean estimates around their sample mean for a given value of the time index and of the observations. In Figure 2, we have displayed the box and whisker plot of the error at time index 25 between the filtered mean estimates and their sample mean at each time instant; these results have been obtained from 100 independent particles (this time, the set of observations and of states are held fixed over all the Monte Carlo simulations). As above, we have used $N = 5, 10, 50$ of particles and two sampling methods: multinomial sampling (Figure 2a) and modified stratified sampling (Figure 2b). This figure shows that the GS estimate of the sampled mean has a lower standard deviation than any other estimators included in this comparison, independently of the number of particles which are used. The differences between these estimators are however small compared to the filtering variance.

4.2. Joint channel equalization and symbol detection on a flat Rayleigh-fading channel

4.2.1. Model description

We consider in this section a problem arising in transmission over a Rayleigh-fading channel. Consider a communi-

¹The Matlab code to reproduce these experiments is available at <http://www.tsi.enst.fr/~moulines/>.

²The lower and upper limits of the box are the quartiles; the horizontal line in the box is the sample median; the upper and lower whiskers are at 3/2 times interquartiles.

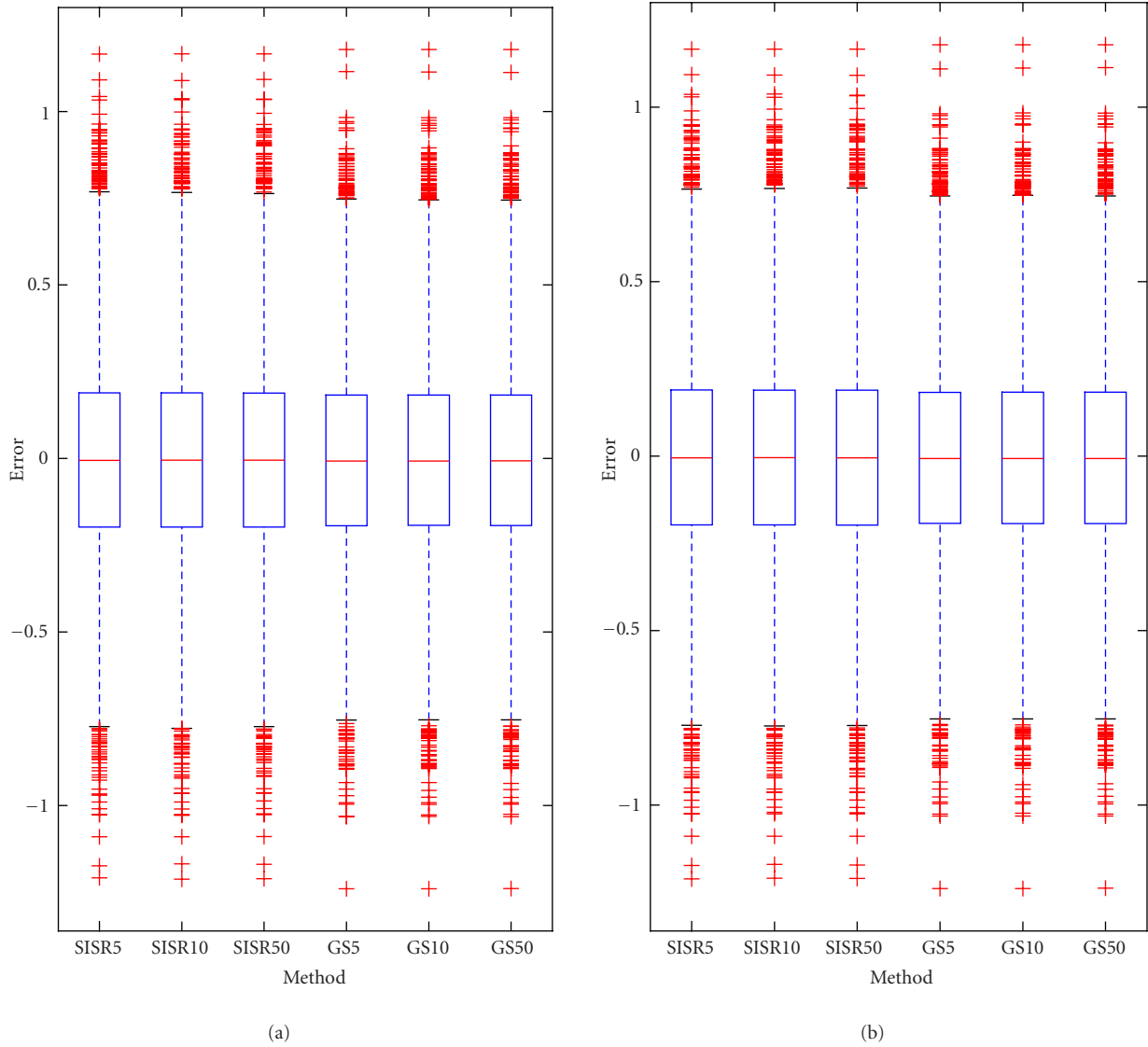


FIGURE 1: Box and whisker plot of the difference between the filtered mean estimates and the actual value of the state estimate for 100 independent Monte Carlo experiments. (a) Multinomial sampling. (b) Residual sampling with the modified stratified sampling.

cation system signaling through a flat-fading channel with additive noise. In this context, the indicator variables $\{\Lambda_t\}$ in the representation (28) are the input bits which are transmitted over the channel and $\{S_t\}_{t \geq 0}$ are the symbols generally taken into an M -ary complex alphabet. The function Ψ_t is thus the function which maps the stream of input bits into a stream of complex symbols: this function combines *channel encoding* and *symbol mapping*. In the simple example considered below, we assume binary phase shift keying (BPSK) modulation with differential encoding: $S_t = S_{t-1}(2\Lambda_t - 1)$. The input-output relationship of the flat-fading channel is described by

$$Y_t = \alpha_t S_t + V_t, \tag{55}$$

where Y_t , α_t , S_t , and V_t denote the received signal, the fading channel coefficient, the transmitted symbol, and the additive noise at time t , respectively. It is assumed in the sequel that

- (i) the processes $\{\alpha_t\}_{t \geq 0}$, $\{\Lambda_t\}_{t \geq 0}$, and $\{V_t\}_{t \geq 0}$ are mutually independent;
- (ii) the noise $\{V_t\}$ is a sequence of i.i.d. zero-mean complex random variables $V_t \sim \mathcal{N}_c(0, \sigma_V^2)$.

It is further assumed that the channel fading process is Rayleigh, that is, $\{\alpha_t\}$ is a zero-mean complex Gaussian process; here modelled as an ARMA(L, L),

$$\alpha_t - \phi_1 \alpha_{t-1} - \dots - \phi_L \alpha_{t-L} = \theta_0 \eta_t + \theta_1 \eta_{t-1} + \dots + \theta_L \eta_{t-L}, \tag{56}$$

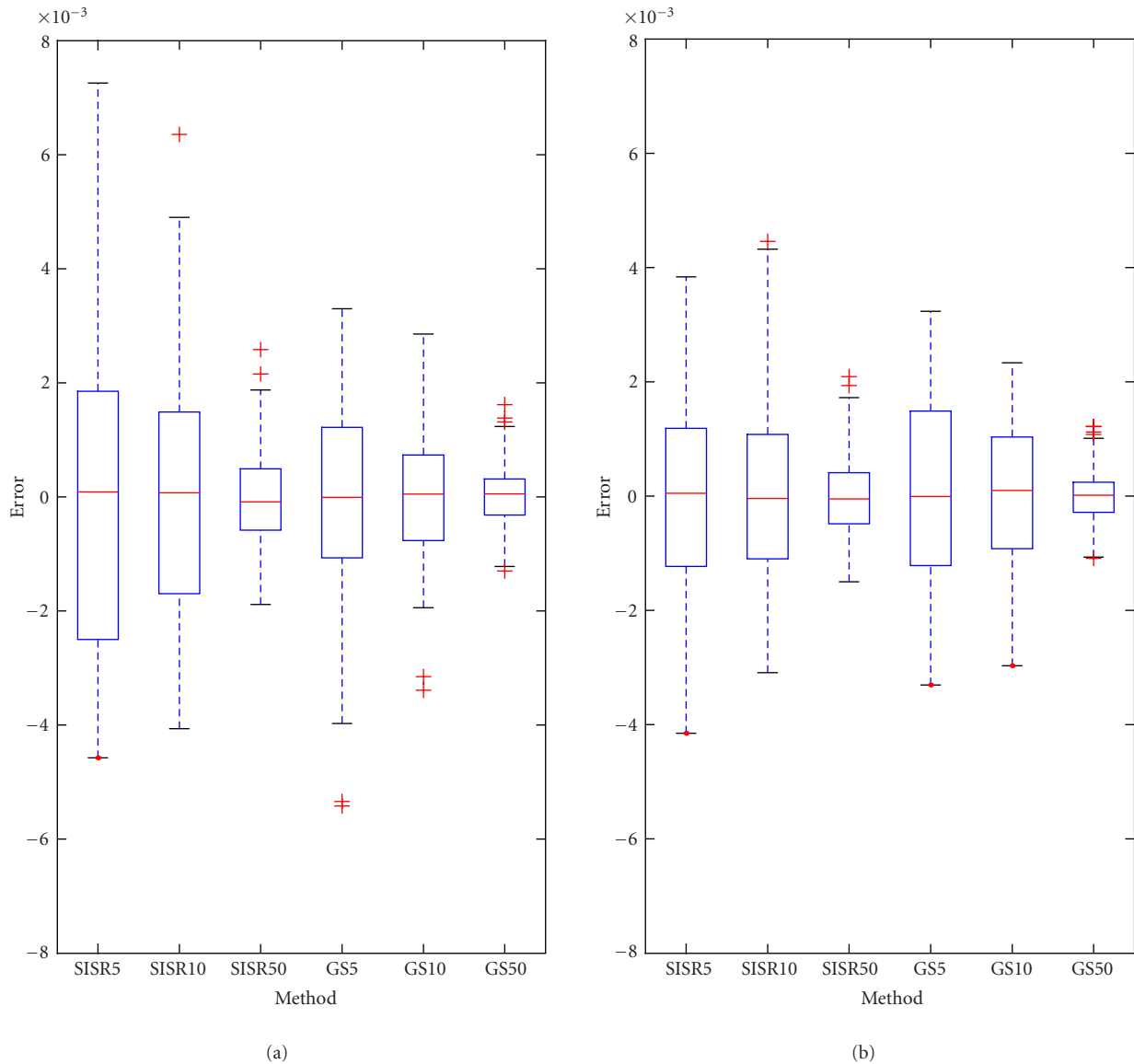


FIGURE 2: Box and whisker plot of the difference between the filtered mean estimates and their sample mean for 100 independent particles for a given value of the time index 25 and of the observations. (a) Multinomial sampling. (b) Residual sampling with the modified stratified sampling.

where ϕ_1, \dots, ϕ_L and $\theta_0, \dots, \theta_L$ are the autoregressive and the moving average (ARMA) coefficients, respectively, and $\{\eta_t\}$ is a white complex Gaussian noise with zero mean and unit variance. This model can be written in state-space form as follows:

$$\vec{X}_{t+1} = \begin{bmatrix} 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \phi_L & \phi_{L-1} & \dots & \phi_1 & 0 \end{bmatrix} \vec{X}_t + \begin{bmatrix} \psi_1 \\ \psi_2 \\ \vdots \\ \psi_L \end{bmatrix} \eta_t, \quad (57)$$

$$\alpha_t = [10 \quad \dots \quad 0] \vec{X}_t + \eta_t,$$

where $\{\psi_k\}_{1 \leq k \leq m}$ are the coefficients of the expansion of $\theta(z)/\phi(z)$, for $|z| \leq 1$, with

$$\begin{aligned} \phi(z) &= 1 - \phi_1 z - \dots - \phi_p z^p, \\ \theta(z) &= 1 + \theta_1 z + \dots + \theta_q z^q. \end{aligned} \quad (58)$$

This particular problem has been considered, among others, in [10, 16, 18, 21, 22].

4.2.2. Simulation results

To allow comparison with previously reported work, we consider the example studied in [16, Section VIII]. In this

example, the fading process is modelled by the output of a Butterworth filter of order $L = 3$ whose cutoff frequency is 0.05, corresponding to a normalized Doppler frequency $f_d T = 0.05$ with respect to the symbol rate $1/T$, which is a fast-fading scenario. More specifically, the fading process is modelled by the ARMA(3, 3) process

$$\begin{aligned} \alpha_t &- 2.37409\alpha_{t-1} + 1.92936\alpha_{t-2} - 0.53208\alpha_{t-3} \\ &= 10^{-2}(0.89409\eta_t + 2.68227\eta_{t-1} \\ &\quad + 2.68227\eta_{t-2} + 0.89409\eta_{t-3}), \end{aligned} \quad (59)$$

where $\eta_t \sim \mathcal{N}_c(0, 1)$. It is assumed that a BPSK modulation is used, that is, $S_t \in \{-1, +1\}$, with differential encoding and no channel code; more precisely, we assume that $S_t = S_{t-1}\Lambda_t$, where $\Lambda_t \in \{-1, +1\}$ is the bit sequence, assumed to be i.i.d. Bernoulli random variables with probability of success $\mathbb{P}(\Lambda_t = 1) = 1/2$.

The performance of the GS receiver (using the modified residual sampling algorithm) has been compared with the following receiver schemes.

- (1) *Known channel lower bound.* We assume that the true fading coefficients α_t are known to the receiver and we calculate the optimal coherent detection rule $\hat{S}_t = \text{sign}(\Re\{\alpha_t^* \tilde{Y}_t\})$ and $\hat{\Lambda}_t = \hat{S}_t \hat{S}_{t-1}$.
- (2) *Genie-aided lower bound.* We assume that a genie allows the receiver to observe $\tilde{Y}_t = \alpha_t + \tilde{V}_t$, with $\tilde{V}_t \sim \mathcal{N}_c(0, \sigma_V^2)$. We use \tilde{Y}_t to calculate an estimate $\hat{\alpha}_t$ of the fading coefficients via a Kalman filter and we then evaluate the optimal coherent detection $\hat{S}_t = \text{sign}(\Re\{\hat{\alpha}_t^* \tilde{Y}_t\})$ and $\hat{\Lambda}_t = \hat{S}_t \hat{S}_{t-1}$ using the filtered fading process.
- (3) *Differential detector.* In this scenario, no attempt is made to estimate the fading process and the input bits are estimated using incoherent differential detection: $\hat{\Lambda}_t = \text{sign}(\Re\{Y_t^* \tilde{Y}_{t-1}\})$.
- (4) *MKF detector.* The SMC filter described in [16, Sections IV and V] is used to estimate Λ_t . The MKF detector uses the SISR algorithm to draw samples in the indicator space and implements a Kalman filter for each trajectory in order to compute its trial sampling density and its importance weight. Resampling is performed when the ratio between the effective sample size defined in [16, equation (45)] and the actual sample size N is lower than a threshold β . The delayed weight method is used to obtain an estimate of Λ_t with a delay δ .

In all the simulations below, we have used only the concurrent sampling method because in the considered simulation scenarios, the use of the delayed sampling method did not bring significant improvement. This is mainly due to the fact that we have only considered, due to space limitations, the uncoded communication scenario.

Figure 3 shows the BER performance of each receiver versus the SNR. The SNR is defined as $\text{var}(\alpha_t)/\text{var}(V_t)$ and the BER is obtained by averaging the error rate over 10^6 symbols. The first 50 symbols were not taken into account in

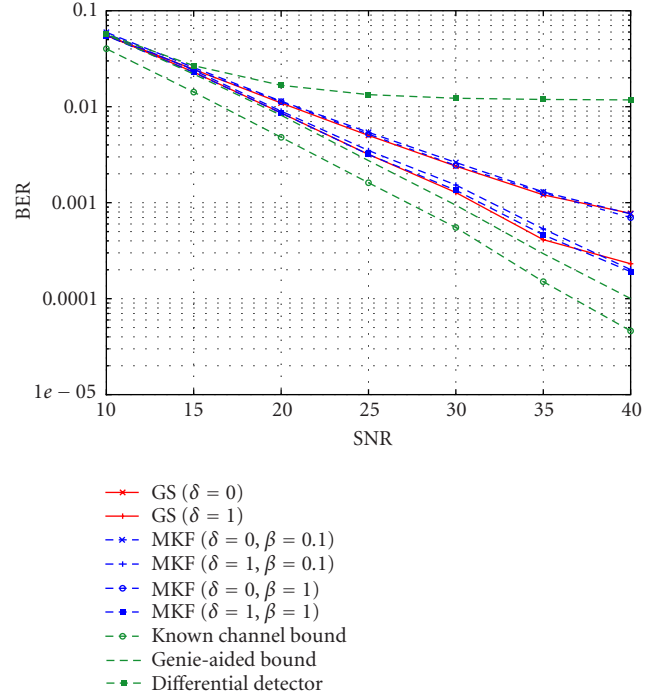


FIGURE 3: BER performance of the GS receiver versus the SNR. The BER corresponding to delays $\delta = 0$ and $\delta = 1$ are shown. Also shown in this figure are the BER curves for the MKF detector ($\delta = 0$ and $\delta = 1$), the known channel lower bound, the genie-aided lower bound, and the differential detector. The number of particles for the GS receiver and the MKF detector is 50.

counting the BER. The BER performance of the GS receiver is shown for estimation delays $\delta = 0$ (concurrent estimation) and $\delta = 1$. Also shown are the BER curves for the known channel lower bound, the genie-aided lower bound, the differential detector, and the MKF detector with estimation delays $\delta = 0$ and $\delta = 1$ and resampling thresholds $\beta = 0.1$ and $\beta = 1$ (systematic resampling). The number of particles for both the GS receiver and the MKF detector is set to 50. From this figure, it can be seen that with 50 particles, there is no significant performance difference between the proposed receiver and the MKF detector with the same estimation delay and $\beta = 0.1$ or $\beta = 1$. Note that, as observed in [16], the performance of the receiver is significantly improved by the delayed-weight method with $\delta = 1$ compared with concurrent estimate; there is no substantial improvement when increasing further the delay; the GS receiver achieves essentially the genie-aided bound over the considered SNR.

Figure 4 shows the BER performance of the GS receiver versus the number of particles at SNR = 20 dB and $\delta = 1$. Also shown in this figure is the BER performance for the MKF detector with $\beta = 0.1$ and $\beta = 1$, respectively. It can be seen from this plot that when the number of particles is decreased from 50 to 10, the BER of the MKF receiver with $\beta = 0.1$ increases by 67%, whereas the BER of the GS receiver increases by 11% only. In fact, Figure 4 also shows that, for this particular example, the BER performance of the GS receiver is identical to the BER performance of an MKF with

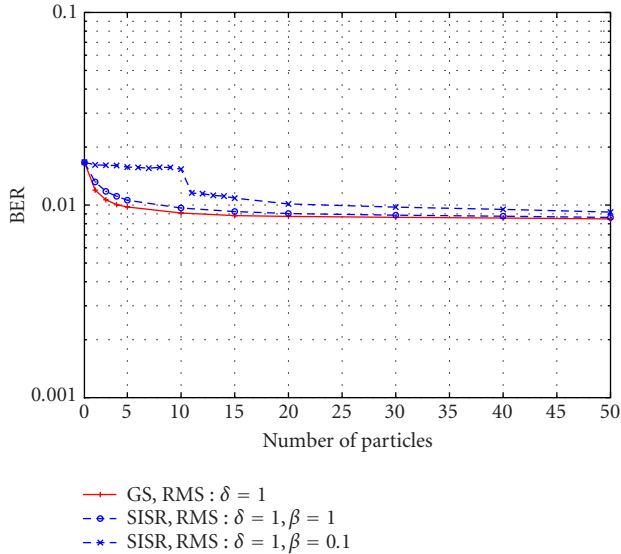


FIGURE 4: BER performance of the GS receiver versus the number of particles at SNR = 20 dB and $\delta = 1$. Also shown in this figure are the BER curves for the MKF detector with $\beta = 0.1$ and $\beta = 1$.

the same number of particles and a resampling threshold set to $\beta = 1$ (systematic resampling). This suggests that, contrary to what is usually argued in the literature [5, 16], systematic resampling of the particle seems to be, for reasons which remain yet unclear from a theoretical standpoint, more robust when the number of particles is decreased to meet the constraints of real-time implementation.

Figure 5 shows the BER performance of each receiver versus the SNR when the number of particles for both the GS receiver and the MKF detector is set to 5. For these simulations, the BER is obtained by averaging the error rate over 10^5 symbols. From this figure, it can be seen that with 5 particles, there is a significant performance difference between the proposed receiver and the MKF detector with the same estimation delay and a $\beta = 0.1$ resampling threshold. This difference remains significant even for SNR values close to 10 dB. Figure 5 also shows that, for this particular example, the BER performance of the GS receiver is identical to the BER performance of an MKF with the same estimation delay and a resampling threshold β set to 1.

5. CONCLUSION

In this paper, a sampling algorithm for conditionally linear Gaussian state-space models has been introduced. This algorithm exploits the particular structure of the flow of probability measures and the fact that, at each time instant, a global exploration of all possible offsprings of a given trajectory of indicator variables can be considered. The number of trajectories is kept constant by sampling from this set (selection step).

The global sampling algorithm appears, in the example considered here, to be robust even when a very limited number of particles is used, which is a basic requirement for

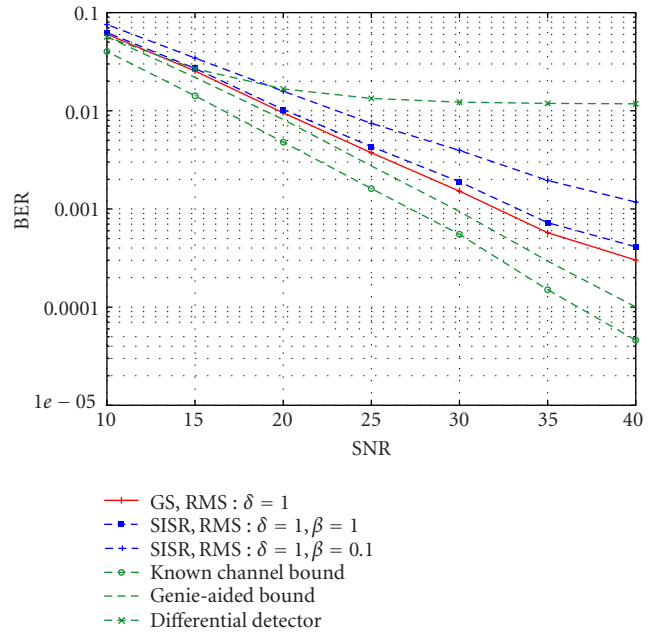


FIGURE 5: BER performance of the GS receiver versus the SNR. The BER corresponding to delay $\delta = 1$ is shown. Also shown in this figure are the BER curves for the MKF detector ($\delta = 1, \beta = 0.1$), the known channel lower bound, the genie-aided lower bound, and the differential detector. The number of particles for the GS receiver and the MKF detector is 5.

the implementation of such a solution in real-world applications: the global sampling algorithm is close to the optimal genie-aided bound with as few as 5 particles and thus provides a realistic alternative to the joint channel equalization and symbol detection algorithms reported earlier in the literature.

APPENDIX

MODIFIED STRATIFIED SAMPLING

In this appendix, we present the so-called modified stratified sampling strategy. Let M and N be integers and (w_1, \dots, w_M) be nonnegative weights such that $\sum_{i=1}^M w_i = 1$. A sampling procedure is said to be unbiased if the random vector (N_1, \dots, N_M) (where N_i is the number of times the index i is drawn) satisfies

$$\sum_{i=1}^M N_i = N, \quad \mathbb{E}[N_i] = Nw_i, \quad i \in \{1, \dots, M\}. \quad (\text{A.1})$$

The modified stratified sampling is summarized as follows.

- (1) For $i \in \{1, \dots, M\}$, compute $[Nw_i]$, where $[x]$ is the integer part of x ; then compute the residual number $\tilde{N} = N - \sum_{i=1}^M [Nw_i]$ and the residual weights

$$\tilde{w}_i = \frac{Nw_i - [Nw_i]}{\tilde{N}}, \quad i \in \{1, \dots, M\}. \quad (\text{A.2})$$

- (2) Draw \tilde{N} i.i.d. random variables $U_1, \dots, U_{\tilde{N}}$ with a uniform distribution on $[0, 1/\tilde{N}]$ and compute, for $k \in \{1, \dots, \tilde{N}\}$,

$$\tilde{U}_k = \frac{k-1}{\tilde{N}} + U_k. \quad (\text{A.3})$$

- (3) For $i \in \{1, \dots, M\}$, set \tilde{N}_i as the number of indices $k \in \{1, \dots, \tilde{N}\}$ satisfying

$$\sum_{j=1}^{i-1} w_j < \tilde{U}_k \leq \sum_{j=1}^i w_j. \quad (\text{A.4})$$

REFERENCES

- [1] T. Kailath, A. Sayed, and B. Hassibi, *Linear Estimation*, Prentice Hall, Englewood Cliffs, NJ, USA, 1st edition, 2000.
- [2] I. MacDonald and W. Zucchini, *Hidden Markov and Other Models for Discrete-Valued Time Series*, vol. 70 of *Monographs on Statistics and Applied Probability*, Chapman & Hall, London, UK, 1997.
- [3] A. Doucet, N. de Freitas, and N. Gordon, "An introduction to sequential Monte Carlo methods," in *Sequential Monte Carlo Methods in Practice*, A. Doucet, N. de Freitas, and N. Gordon, Eds., pp. 3–13, Springer, New York, NY, USA, January 2001.
- [4] C. Carter and R. Kohn, "Markov chain Monte Carlo in conditionally Gaussian state space models," *Biometrika*, vol. 83, no. 3, pp. 589–601, 1996.
- [5] A. Doucet, S. Godsill, and C. Andrieu, "On sequential Monte Carlo sampling methods for Bayesian filtering," *Statistics and Computing*, vol. 10, no. 3, pp. 197–208, 2000.
- [6] N. Shephard, "Partial non-Gaussian state space," *Biometrika*, vol. 81, no. 1, pp. 115–131, 1994.
- [7] J. Liu and R. Chen, "Sequential Monte Carlo methods for dynamic systems," *Journal American Statistical Association*, vol. 93, no. 444, pp. 1032–1044, 1998.
- [8] R. Chen and J. Liu, "Mixture Kalman filter," *Journal Royal Statistical Society Series B*, vol. 62, no. 3, pp. 493–508, 2000.
- [9] G. Rigal, *Filtrage non-linéaire, résolution particulière et applications au traitement du signal*, Ph.D. dissertation, Université Paul Sabatier, Toulouse, France, 1993.
- [10] J. Liu and R. Chen, "Blind deconvolution via sequential imputations," *Journal American Statistical Association*, vol. 430, no. 90, pp. 567–576, 1995.
- [11] E. Punsakaya, C. Andrieu, A. Doucet, and W. Fitzgerald, "Particle filtering for multiuser detection in fading CDMA channels," in *Proc. 11th IEEE Signal Processing Workshop on Statistical Signal Processing*, pp. 38–41, Orchid Country Club, Singapore, August 2001.
- [12] V. Zaritskii, V. Svetnik, and L. Shimelevich, "Monte-Carlo technique in problems of optimal information processing," *Automation and Remote Control*, vol. 36, no. 12, pp. 95–103, 1975.
- [13] H. Akashi and H. Kumamoto, "Random sampling approach to state estimation in switching environments," *Automatica*, vol. 13, no. 4, pp. 429–434, 1977.
- [14] J. Tugnait, "Adaptive estimation and identification for discrete systems with markov jump parameters," *IEEE Trans. Automatic Control*, vol. 27, no. 5, pp. 1054–1065, 1982.
- [15] A. Doucet, N. Gordon, and V. Krishnamurthy, "Particle filters for state estimation of jump Markov linear systems," *IEEE Trans. Signal Processing*, vol. 49, no. 3, pp. 613–624, 2001.
- [16] R. Chen, X. Wang, and J. Liu, "Adaptive joint detection and decoding in flat-fading channels via mixture kalman filtering," *IEEE Transactions on Information Theory*, vol. 46, no. 6, pp. 2079–2094, 2000.
- [17] X. Wang, R. Chen, and D. Guo, "Delayed-pilot sampling for mixture Kalman filter with application in fading channels," *IEEE Trans. Signal Processing*, vol. 50, no. 2, pp. 241–254, 2002.
- [18] E. Punsakaya, C. Andrieu, A. Doucet, and W. J. Fitzgerald, "Particle filtering for demodulation in fading channels with non-Gaussian additive noise," *IEEE Transactions on Communications*, vol. 49, no. 4, pp. 579–582, 2001.
- [19] G. Kitagawa, "Monte Carlo filter and smoother for non-Gaussian non-linear state space models," *Journal of Computational and Graphical Statistics*, vol. 5, no. 1, pp. 1–25, 1996.
- [20] J. Liu, R. Chen, and T. Logvinenko, "A theoretical framework for sequential importance sampling and resampling," in *Sequential Monte Carlo Methods in Practice*, A. Doucet, N. de Freitas, and N. Gordon, Eds., Springer, New York, NY, USA, 2001.
- [21] F. Ben Salem, "Récepteur particulière pour canaux mobiles évanescents," in *Journées Doctorales d'Automatique (JDA '01)*, pp. 25–27, Toulouse, France, September 2001.
- [22] F. Ben Salem, *Réception particulière pour canaux multi-trajets évanescents en communication radiomobile*, Ph.D. dissertation, Université Paul Sabatier, Toulouse, France, 2002.

Pascal Cheung-Mon-Chan graduated from the Ecole Normale Supérieure de Lyon in 1994 and received the Diplôme d'Ingénieur from the École Nationale Supérieure des Télécommunications (ENST) in Paris in the same year. After working for General Electric Medical Systems as a Research and Development Engineer, he received the Ph.D. degree from the ENST in Paris in 2003. He is currently a member of the research staff at France Telecom Research and Development.



Eric Moulines was born in Bordeaux, France, in 1963. He received the M.S. degree from Ecole Polytechnique in 1984, the Ph.D. degree from École Nationale Supérieure des Télécommunications (ENST) in 1990 in signal processing, and an "Habilitation à Diriger des Recherches" in applied mathematics (probability and statistics) from Université René Descartes (Paris V) in 1995. From 1986 until 1990, he was a member of the technical staff at Centre National de Recherche des Télécommunications (CNET). Since 1990, he was with ENST, where he is presently a Professor (since 1996). His teaching and research interests include applied probability, mathematical and computational statistics, and signal processing.

Multilevel Mixture Kalman Filter

Dong Guo

Department of Electrical Engineering, Columbia University, New York, NY 10027, USA
Email: guodong@ee.columbia.edu

Xiaodong Wang

Department of Electrical Engineering, Columbia University, New York, NY 10027, USA
Email: wangx@ee.columbia.edu

Rong Chen

Department of Information and Decision Sciences, University of Illinois at Chicago, Chicago, IL 60607-7124, USA
Email: rongchen@uic.edu

Department of Business Statistics & Econometrics, Peking University, Beijing 100871, China

Received 30 April 2003; Revised 18 December 2003

The mixture Kalman filter is a general sequential Monte Carlo technique for conditional linear dynamic systems. It generates samples of some indicator variables recursively based on sequential importance sampling (SIS) and integrates out the linear and Gaussian state variables conditioned on these indicators. Due to the marginalization process, the complexity of the mixture Kalman filter is quite high if the dimension of the indicator sampling space is high. In this paper, we address this difficulty by developing a new Monte Carlo sampling scheme, namely, the multilevel mixture Kalman filter. The basic idea is to make use of the multilevel or hierarchical structure of the space from which the indicator variables take values. That is, we draw samples in a multilevel fashion, beginning with sampling from the highest-level sampling space and then draw samples from the associate subspace of the newly drawn samples in a lower-level sampling space, until reaching the desired sampling space. Such a multilevel sampling scheme can be used in conjunction with the delayed estimation method, such as the delayed-sample method, resulting in delayed multilevel mixture Kalman filter. Examples in wireless communication, specifically the coherent and noncoherent 16-QAM over flat-fading channels, are provided to demonstrate the performance of the proposed multilevel mixture Kalman filter.

Keywords and phrases: sequential Monte Carlo, mixture Kalman filter, multilevel mixture Kalman filter, delayed-sample method.

1. INTRODUCTION

Recently there have been significant interests in the use of the sequential Monte Carlo (SMC) methods to solve on-line estimation and prediction problems in dynamic systems. Compared with the traditional filtering methods, the simple, flexible—yet powerful—SMC provides effective means to overcome the computational difficulties in dealing with nonlinear dynamic models. The basic idea of the SMC technique is the recursive use of the sequential importance sampling (SIS). There also have been many recent modifications and improvements on the SMC methodology [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12].

Among these SMC methods, the mixture Kalman filter (MKF) [3] is a powerful tool to deal with conditional dynamic linear models (CDLMs) and finds important applications in digital wireless communications [3, 13, 14]. A similar method is also discussed in [15] for CDLM system. The CDLM is a direct generalization of the dynamic linear model

(DLM) [16] and it can be generally described as follows:

$$\begin{aligned}\mathbf{x}_t &= \mathbf{F}_{\lambda_t} \mathbf{x}_{t-1} + \mathbf{G}_{\lambda_t} \mathbf{u}_t, \\ \mathbf{y}_t &= \mathbf{H}_{\lambda_t} \mathbf{x}_t + \mathbf{K}_{\lambda_t} \mathbf{v}_t,\end{aligned}\quad (1)$$

where $\mathbf{u}_t \sim \mathcal{N}(0, \mathbf{I})$ and $\mathbf{v}_t \sim \mathcal{N}(0, \mathbf{I})$ 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 \mathbf{u}_t and \mathbf{v}_t and the past \mathbf{x}_s and \mathbf{y}_s , $s < t$. The matrices \mathbf{F}_{λ_t} , \mathbf{G}_{λ_t} , \mathbf{H}_{λ_t} , and \mathbf{K}_{λ_t} are known, given λ_t .

An important feature of CDLM is that, given the trajectory of the indicator $\{\lambda_t\}$, the system becomes Gaussian and linear, for which the Kalman filter can be used. Thus, by using the marginalization technique for Monte Carlo computation [17], the MKF focuses on the sampling of the indicator variable λ_t other than the whole state variable $\{\mathbf{x}_t, \lambda_t\}$. This method can drastically reduce Monte Carlo variances associated with a standard sequential importance sampler applied directly to the space of the state variable.

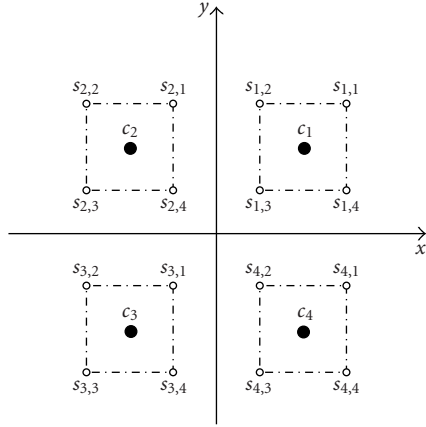


FIGURE 1: The multilevel structure of the 16-QAM modulation used in digital communications. The set of transmitted symbol $\mathcal{A}_1 = \{s_{i,j}, i = 1, \dots, 4, j = 1, \dots, 4\}$ is the original sampling space, and the centers $\mathcal{A}_2 = \{c_1, c_2, c_3, c_4\}$ constitute a higher-level sampling space.

However, the computational complexity of the MKF can be quite high, especially in the case of high-dimensional indicator space, due to the need of marginalizing out the indicator variables. Fortunately, often the space from which the indicator variables take values exhibits multilevel or hierarchical structures, which can be exploited to reduce the computational complexity of the MKF. For example, a multilevel structure of the 16-QAM modulation used in digital communications is shown in Figure 1. The set of transmitted symbols $\mathcal{A}_1 = \{s_{i,j}, i = 1, \dots, 4, j = 1, \dots, 4\}$ is the original sampling space, and the centers $\mathcal{A}_2 = \{c_1, c_2, c_3, c_4\}$ constitute a higher-level sampling space. Thus, based on the observed data, for every sample stream, we first draw a sample (say c_1) from the higher-level sampling space \mathcal{A}_2 and then draw a new sample from the associated subspaces $s_{1,1}, s_{1,2}, s_{1,3}, s_{1,4}$ of c_1 in the original sampling space \mathcal{A}_1 . In this way, we need not sample from the entire original sampling space, and many Kalman filter update steps associated with the standard MKF can be saved.

This kind of hierarchical structure imposed on the indicator space is also employed in the partitioned sampling strategy [18], which greatly improved the efficiency and the accuracy for multiple target tracking over the original SMC methods. However, in this paper, the hierarchical structure is employed to reduce the computational load associated with MKF, especially for high-dimensional indicator space, while retaining the desirable properties of MKF.

Dynamic systems often possess strong memories, that is, future observations can reveal substantial information about the current state. Therefore, it is often beneficial to make use of these future observations in sampling the current state. However, an MKF method usually does not go back to regenerate past samples in view of the new observations, although the past estimation can be adjusted by using the new importance weights. To overcome this difficulty, a *delayed-sample* method is developed [13]. It makes use of future observations in generating samples of the current state. It is seen there that this method is especially effective in improving

the performance of the MKF. However, the computational complexity of the delayed-sample method is very high. For example, for a Δ -step delayed-sample method, the algorithmic complexity is $\mathcal{O}(|\mathcal{A}_1|^\Delta)$, where $|\mathcal{A}_1|$ is the cardinality of the original sampling space. Here, we also provide a delayed multilevel MKF by exploring the multilevel structure of the indicator space. Instead of exploring the original entire space of future states, we only sample the future states in a higher-level sampling space, thus significantly reducing the dimension of the search space and the computational complexity.

In recent years, the SMC methods have been successfully employed in several important problems in communications, such as the detection in flat-fading channels [13, 19, 20], space-time coding [21, 22], OFDM system [23], and so on. To show the good performance of the proposed novel receivers, we apply them into the problem of adaptive detection in flat-fading channels in the presence of Gaussian noise.

The remainder of the paper is organized as follows. Section 2 briefly reviews the MKF algorithm and its variants. In Section 3, we present the multilevel MKF algorithm. In Section 4, we treat the delayed multilevel MKF algorithm. In Section 5, we provide simulation examples. Section 6 concludes the paper.

2. BACKGROUND OF MIXTURE KALMAN FILTER

2.1. Mixture Kalman filter

Consider again the CDLMs defined by (1). The MKF exploits the conditional Gaussian property conditioned on the indicator variable and utilizes a marginalization operation to improve the algorithmic efficiency. Instead of dealing with both \mathbf{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 SMC method, the MKF is substantially more efficient (e.g., it produces more accurate results with the same computational resources).

First we define an important concept that is used throughout the paper. A set of random samples and the corresponding weights $\{(\eta^{(i)}, w^{(i)})\}_{i=1}^m$ is said to be *properly weighted* with respect to the distribution $\pi(\cdot)$ if, for any measurable function h , we have

$$\frac{\sum_{j=1}^m h(\eta^{(j)}) w^{(j)}}{\sum_{j=1}^m w^{(j)}} \rightarrow E_\pi\{h(\eta)\} \quad \text{as } m \rightarrow \infty. \quad (2)$$

In particular, if $\eta^{(j)}$ is sampled from a trial distribution $g(\cdot)$ which has the same support as π , and if $w^{(j)} = \pi(\eta^{(j)})/g(\eta^{(j)})$, then $\{(\eta^{(j)}, w^{(j)})\}_{j=1}^m$ is properly weighted with respect to $\pi(\cdot)$.

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

$$\sum_{j=1}^m w_t^{(j)} \mathcal{N}_c(\boldsymbol{\mu}_t^{(j)}, \boldsymbol{\Sigma}_t^{(j)}), \quad (3)$$

where

$$\boldsymbol{\mu}_t^{(j)} = \boldsymbol{\mu}_t(\boldsymbol{\Lambda}_t^{(j)}), \quad \boldsymbol{\Sigma}_t^{(j)} = \boldsymbol{\Sigma}_t(\boldsymbol{\Lambda}_t^{(j)}) \quad (4)$$

are obtained with a Kalman filter on the system (1) for the given indicator trajectory $\boldsymbol{\Lambda}_t^{(j)}$. Denote

$$\boldsymbol{\kappa}_t^{(j)} \triangleq [\boldsymbol{\mu}_t^{(j)}, \boldsymbol{\Sigma}_t^{(j)}]. \quad (5)$$

Thus, a key step in the MKF is the production at time t of the weighted samples of indicators $\{(\boldsymbol{\Lambda}_t^{(j)}, \boldsymbol{\kappa}_t^{(j)}, w_t^{(j)})\}_{j=1}^m$ based on the set of samples $\{(\boldsymbol{\Lambda}_{t-1}^{(j)}, \boldsymbol{\kappa}_{t-1}^{(j)}, w_{t-1}^{(j)})\}_{j=1}^m$ at the previous time $(t-1)$. Suppose that the indicator λ_t takes values from a finite set \mathcal{A}_1 . The MKF algorithm is as follows.

Algorithm 1 (MKF). Suppose at time $(t-1)$, a set of property weighted samples $\{(\boldsymbol{\Lambda}_{t-1}^{(j)}, \boldsymbol{\kappa}_{t-1}^{(j)}, w_{t-1}^{(j)})\}_{j=1}^m$ is available with respect to $p(\boldsymbol{\Lambda}_{t-1} | \mathbf{Y}_{t-1})$. Then at time t , as the new data \mathbf{y}_t becomes available, the following steps are implemented to update each weighted sample.

For $j = 1, \dots, m$, the following steps are applied.

- (i) Based on the new data \mathbf{y}_t , for each $a_i \in \mathcal{A}_1$, run a one-step Kalman filter update assuming $\lambda_t = a_i$ to obtain

$$\boldsymbol{\kappa}_{t-1}^{(j)} \xrightarrow{\mathbf{y}_t, \lambda_t = a_i} \boldsymbol{\kappa}_{t,i}^{(j)} \triangleq [\boldsymbol{\mu}_t(\boldsymbol{\Lambda}_{t-1}^{(j)}, \lambda_t = a_i), \boldsymbol{\Sigma}_t(\boldsymbol{\Lambda}_{t-1}^{(j)}, \lambda_t = a_i)]. \quad (6)$$

- (ii) For each $a_i \in \mathcal{A}_1$, compute the sampling density

$$\begin{aligned} \rho_{t,i}^{(j)} &\triangleq P(\lambda_t = a_i | \boldsymbol{\Lambda}_{t-1}^{(j)}, \mathbf{Y}_t) \\ &\propto p(\mathbf{y}_t | \lambda_t = a_i, \boldsymbol{\Lambda}_{t-1}^{(j)}, \mathbf{Y}_{t-1}) P(\lambda_t = a_i | \boldsymbol{\Lambda}_{t-1}^{(j)}). \end{aligned} \quad (7)$$

Note that by the model (1), the first density in (7) is Gaussian and can be computed based on the Kalman filter update (6). Draw a sample $\lambda_t^{(j)}$ according to the above sampling density. Append $\lambda_t^{(j)}$ to $\boldsymbol{\Lambda}_{t-1}^{(j)}$ and obtain $\boldsymbol{\Lambda}_t^{(j)}$. If $\lambda_t^{(j)} = a_i$, then set $\boldsymbol{\kappa}_t^{(j)} = \boldsymbol{\kappa}_{t,i}^{(j)}$.

- (iii) Compute the importance weight

$$\begin{aligned} w_t^{(j)} &= w_{t-1}^{(j)} \cdot p(\mathbf{y}_t | \boldsymbol{\Lambda}_{t-1}^{(j)}, \mathbf{Y}_{t-1}) \\ &\propto w_{t-1}^{(j)} \cdot \sum_{i=1}^{|\mathcal{A}_1|} \rho_{t,i}^{(j)}. \end{aligned} \quad (8)$$

The new sample $\{\boldsymbol{\Lambda}_t^{(j)}, \boldsymbol{\kappa}_t^{(j)}, w_t^{(j)}\}$ is then properly weighted with respect to $p(\boldsymbol{\Lambda}_t | \mathbf{Y}_t)$.

- (iv) Perform a resampling step as discussed below.

2.2. Resampling procedure

The importance sampling weight $w_t^{(j)}$ measures the “quality” of the corresponding imputed indicator sequence $\boldsymbol{\Lambda}_t^{(j)}$. A relatively small weight implies that the sample is drawn far from the main body of the posterior distribution and has a small contribution in the final estimation. Such a sample is said

to be ineffective. If there are too many ineffective samples, the Monte Carlo procedure becomes inefficient. To avoid the degeneracy, a useful resampling procedure, which was suggested in [7, 11], may be used. Roughly speaking, resampling is to multiply the streams with the larger importance weights, while eliminating the ones with small importance weights. A simple, but efficient, resampling procedure consists of the following two steps.

- (1) Sample a new set of streams $\{\tilde{\boldsymbol{\Lambda}}_t^{(j)}, \tilde{\boldsymbol{\mu}}_t^{(j)}, \tilde{\boldsymbol{\Sigma}}_t^{(j)}\}_{j=1}^m$ from $\{\boldsymbol{\Lambda}_t^{(j)}, \boldsymbol{\mu}_t^{(j)}, \boldsymbol{\Sigma}_t^{(j)}\}_{j=1}^m$ with probability proportional to the importance weights $\{w_t^{(j)}\}_{j=1}^m$.
- (2) To each stream in $\{\tilde{\boldsymbol{\Lambda}}_t^{(j)}, \tilde{\boldsymbol{\mu}}_t^{(j)}, \tilde{\boldsymbol{\Sigma}}_t^{(j)}\}_{j=1}^m$, assign equal weight, that is, $\tilde{w}_t^{(j)} = 1/m$, $j = 1, \dots, m$.

Resampling can be done at every fixed-length time interval (say, every five steps) or it can be conducted dynamically. The *effective sample size* can be used to monitor the variation of the importance weights of the sample streams and to decide when to resample as the system evolves. The effective sample size is defined as in [13]:

$$\tilde{m}_t \triangleq \frac{m}{1 + v_t^2}, \quad (9)$$

where v_t , the coefficient of variation, is given by

$$v_t^2 = \frac{1}{m} \sum_{j=1}^m \left(\frac{w_t^{(j)}}{\bar{w}_t} - 1 \right)^2, \quad (10)$$

with $\bar{w}_t = \sum_{j=1}^m w_t^{(j)}/m$. In dynamic resampling, a resampling step is performed once the effective sample size \tilde{m}_t is below a certain threshold.

Heuristically, resampling can provide chances for good sample streams to amplify themselves and hence “rejuvenate” the sampler to produce a better result for future states as the system evolves. It can be shown that the samples drawn by the above resampling procedure are also indeed properly weighted with respect to $p(\boldsymbol{\Lambda}_t | \mathbf{Y}_t)$, provided that m is sufficiently large. In practice, when small to modest m is used (we use $m = 50$ in this paper), the resampling procedure can be seen as a tradeoff between the bias and the variance. That is, the new samples with their weights resulting from the resampling procedure are only approximately proper, and this introduces small bias in the Monte Carlo estimation. On the other hand, however, resampling significantly reduces the Monte Carlo variance for future samples.

2.3. Delayed estimation

Model (1) often exhibits strong memory. As a result, future observations often contain information about the current state. Hence a delayed estimate is usually more accurate than the concurrent estimate. In delayed estimation, instead of making inference on $\boldsymbol{\Lambda}_t$ instantaneously with the posterior distribution $p(\boldsymbol{\Lambda}_t | \mathbf{Y}_t)$, we delay this inference to a later time $(t + \Delta)$, $\Delta > 0$, with the distribution $p(\boldsymbol{\Lambda}_t | \mathbf{Y}_{t+\Delta})$.

As discussed in [13], there are primarily two approaches to delayed estimation, namely, the delayed-weight method and the delayed-sample method.

2.3.1. Delayed-weight method

In the concurrent MKF algorithm, if the set $\{(\Lambda_{t+\delta}^{(j)}, w_{t+\delta}^{(j)})\}_{j=1}^m$ is properly weighted with respect to $p(\Lambda_{t+\delta} | \mathbf{Y}_{t+\delta})$, then when we focus our attention on λ_t at time $(t + \delta)$, we have that $\{(\lambda_t^{(j)}, w_{t+\delta}^{(j)})\}_{j=1}^m$ is properly weighted with respect to $p(\lambda_t | \mathbf{Y}_{t+\delta})$. Then any inference about the indicator λ_t , $E\{h(\lambda_t) | \mathbf{Y}_{t+\delta}\}$, can be approximated by

$$\begin{aligned} E\{h(\lambda_t) | \mathbf{Y}_{t+\delta}\} \\ \cong \frac{1}{W_{t+\delta}} \sum_{j=1}^m h(\lambda_t^{(j)}) w_{t+\delta}^{(j)}, \quad W_{t+\delta} = \sum_{j=1}^m w_{t+\delta}^{(j)}. \end{aligned} \quad (11)$$

Since the weights $\{w_{t+\delta}^{(j)}\}_{j=1}^m$ contain information about the future observations $(\mathbf{y}_{t+1}, \dots, \mathbf{y}_{t+\delta})$, the estimate in (11) is usually more accurate than the concurrent estimate. Note that such a delayed estimation method incurs no additional computational cost (i.e., CPU time), but it requires some extra memory for storing $\{\lambda_t^{(j)}, \dots, \lambda_{t+\delta}^{(j)}\}_{j=1}^m$. For most systems, this simple delayed-weight method is quite effective for improving the performance over the concurrent method. However, if this method is not sufficient for exploiting the constraint structures of the indicator variable, we must resort to the delayed-sample method, which is described next.

2.3.2. Delayed-sample method

An alternative method of delayed estimation is to generate both the delayed samples and the weights $\{(\lambda_t^{(j)}, w_t^{(j)})\}_{j=1}^m$ based on the observations $\mathbf{Y}_{t+\Delta}$, hence making $p(\Lambda_t | \mathbf{Y}_{t+\Delta})$ the target distribution at time $(t + \Delta)$. The procedure will provide better Monte Carlo samples since it utilizes the future observations $(\mathbf{y}_{t+1}, \dots, \mathbf{y}_{t+\Delta})$ in generating the current samples of λ_t . But the algorithm is also more demanding both analytically and computationally because of the need of marginalizing out $\lambda_{t+1}, \dots, \lambda_{t+\Delta}$.

For each possible "future" (relative to time $t - 1$) symbol sequence at time $(t + \Delta - 1)$, that is,

$$(\lambda_t, \lambda_{t+1}, \dots, \lambda_{t+\Delta-1}) \in \mathcal{A}_1^\Delta, \quad (12)$$

we keep the value of a Δ -step Kalman filter $\{\kappa_{t+\tau}^{(j)}(\lambda_t^{t+\tau})\}_{\tau=0}^{\Delta-1}$, where

$$\begin{aligned} \kappa_{t+\tau}^{(j)}(\lambda_t^{t+\tau}) \\ \triangleq [\boldsymbol{\mu}_{t+\tau}(\Lambda_{t-1}^{(j)}, \lambda_t^{t+\tau}), \boldsymbol{\Sigma}_{t+\tau}(\Lambda_{t-1}^{(j)}, \lambda_t^{t+\tau})], \quad \tau = 0, \dots, \Delta-1, \end{aligned} \quad (13)$$

with $\lambda_a^b \triangleq (\lambda_a, \lambda_{a+1}, \dots, \lambda_b)$. Denote

$$\kappa_{t-1}^{(j)} \triangleq \left\{ \kappa_{t-1}^{(j)}, \{\kappa_{t+\tau}^{(j)}(\lambda_t^{t+\tau})\}_{\tau=0}^{\Delta-1} : \lambda_t^{t+\tau} \in \mathcal{A}_1^{t+\tau+1} \right\}. \quad (14)$$

The delayed-sample MKF algorithm recursively propagates the samples properly weighted for $p(\Lambda_{t-1} | \mathbf{Y}_{t+\Delta-1})$ to those for $p(\Lambda_t | \mathbf{Y}_{t+\Delta})$ and is summarized as follows.

Algorithm 2 (delayed-sample MKF). Suppose, at time $(t+\Delta-1)$, a set of properly weighted samples $\{(\Lambda_{t-1}^{(j)}, \kappa_{t-1}^{(j)}, w_{t-1}^{(j)})\}_{j=1}^m$ is available with respect to $p(\Lambda_{t-1} | \mathbf{Y}_{t+\Delta-1})$. Then at time $(t + \Delta)$ as the new data $\mathbf{y}_{t+\Delta}$ becomes available, the following steps are implemented to update each weighted sample.

For $j = 1, 2, \dots, m$, the following steps are performed.

- (i) For each $\lambda_{t+\Delta} = a_i \in \mathcal{A}_1$, and for each $\lambda_t^{t+\Delta-1} \in \mathcal{A}_1^\Delta$, perform a one-step update on the corresponding Kalman filter $\kappa_{t+\Delta-1}^{(j)}(\lambda_t^{t+\Delta-1})$, that is,

$$\kappa_{t+\Delta-1}^{(j)}(\lambda_t^{t+\Delta-1}) \xrightarrow{\mathbf{y}_{t+\Delta}, \lambda_{t+\Delta}=a_i} \kappa_{t+\Delta}^{(j)}(\lambda_t^{t+\Delta-1}, \lambda_{t+\Delta} = a_i). \quad (15)$$
- (ii) For each $a_i \in \mathcal{A}_1$, compute the sampling density

$$\begin{aligned} \rho_{t,i}^{(j)} &\triangleq P(\lambda_t = a_i | \Lambda_{t-1}^{(j)}, \mathbf{Y}_{t+\Delta}) \\ &= P(\lambda_t = a_i | \Lambda_{t-1}^{(j)}) \\ &\quad \times \sum_{\lambda_t^{t+\Delta-1} \in \mathcal{A}_1^\Delta} \left[\prod_{\tau=0}^{\Delta} p(\mathbf{y}_{t+\tau} | \mathbf{Y}_{t+\tau-1}, \Lambda_{t-1}^{(j)}, \lambda_t = a_i, \lambda_{t+1}^{t+\tau}) \right. \\ &\quad \left. \times \prod_{\tau=1}^{\Delta} P(\lambda_{t+\tau} | \Lambda_{t-1}^{(j)}, \lambda_t = a_i, \lambda_{t+1}^{t+\tau-1}) \right]. \end{aligned} \quad (16)$$

Note that the second density in (16) is Gaussian and can be computed based on the results of the Kalman filter updates in (15). Draw a sample $\lambda_t^{(j)}$ according to the above sampling density. Append $\lambda_t^{(j)}$ to $\Lambda_{t-1}^{(j)}$ and obtain $\Lambda_t^{(j)}$. Based on this sample, form $\kappa_t^{(j)}$ using the results from the previous step.

- (iii) Compute the importance weight. If $\lambda_{t-1}^{(j)} = a_k$ and $\lambda_t^{(j)} = a_i$, then

$$\begin{aligned} w_t^{(j)} &= w_{t-1}^{(j)} \frac{p(\Lambda_t^{(j)} | \mathbf{Y}_{t+\Delta})}{p(\Lambda_{t-1}^{(j)} | \mathbf{Y}_{t+\Delta-1}) p(\lambda_t^{(j)} | \Lambda_{t-1}^{(j)}, \mathbf{Y}_{t+\Delta})} \\ &\propto w_{t-1}^{(j)} \frac{\sum_{\lambda_t^{t+\Delta} \in \mathcal{A}_1^{\Delta+1}} \left[\prod_{\tau=0}^{\Delta} p(\mathbf{y}_{t+\tau} | \mathbf{Y}_{t+\tau-1}, \Lambda_{t-1}^{(j)}, \lambda_t^{t+\tau}) \right]}{\sum_{\lambda_t^{t+\Delta-1} \in \mathcal{A}_1^\Delta} \left[\prod_{\tau=0}^{\Delta-1} p(\mathbf{y}_{t+\tau} | \mathbf{Y}_{t+\tau-1}, \Lambda_{t-1}^{(j)}, \lambda_t^{t+\tau}) \right]} \\ &\quad \times \frac{\prod_{\tau=0}^{\Delta} P(\lambda_{t+\tau} | \Lambda_{t-1}^{(j)}, \lambda_t^{t+\tau-1})}{\prod_{\tau=0}^{\Delta-1} P(\lambda_{t+\tau} | \Lambda_{t-1}^{(j)}, \lambda_t^{t+\tau-1})} \\ &\propto \frac{w_{t-1}^{(j)}}{\rho_{t-1,k}^{(j)}} p(\mathbf{y}_{t-1} | \mathbf{Y}_{t-2}, \Lambda_{t-1}^{(j)}) \\ &\quad \times P(\lambda_{t-1} = a_k | \Lambda_{t-2}^{(j)}) \sum_{i=1}^{|\mathcal{A}_1|} \rho_{t,i}^{(j)}. \end{aligned} \quad (17)$$

- (iv) Resample if the effective sample size is below a certain threshold, as discussed in Section 2.2.

Finally, as noted in [13], we can use the above delayed-sample method in conjunction with the delayed-weight method. For example, using the delayed-sample method, we generate delayed samples and weights $\{(\Lambda_t^{(j)}, w_t^{(j)})\}_{j=1}^m$ based on observations $\mathbf{Y}_{t+\Delta}$. Then with an additional delay δ , we can use the following delayed-weight method to approximate any inference about the indicator λ_t :

$$\begin{aligned} & E\{h(\lambda_t) \mid \mathbf{Y}_{t+\Delta+\delta}\} \\ & \cong \frac{1}{W_{t+\delta}} \sum_{j=1}^m h(\lambda_t^{(j)}) w_{t+\delta}^{(j)}, \quad W_{t+\delta} = \sum_{j=1}^m w_{t+\delta}^{(j)}. \end{aligned} \quad (18)$$

3. MULTILEVEL MIXTURE KALMAN FILTER

Suppose that the indicator space has a multilevel structure. For example, consider the following scenario of a two-level sampling space. The first level is the original sampling space Ω with $\Omega = \{\lambda_i, i = 1, 2, \dots, N\}$. We also have a higher-level sampling space $\tilde{\Omega}$ with $\tilde{\Omega} = \{c_i, i = 1, 2, \dots, \tilde{N}\}$. The higher-level sampling space can be obtained as follows. Define the elements (say c_i) in the higher-level sampling space as the centers of subset ω_i in the original sampling space. That is,

$$c_i = \frac{1}{|\omega_i|} \sum_j \lambda_j \mathbb{1}(\lambda_j \in \omega_i) \quad (i = 1, 2, \dots, \tilde{N}), \quad (19)$$

where $\omega_i, i = 1, 2, \dots, \tilde{N}$, is the subset in the original sampling space

$$\Omega = \bigcup_i \omega_i, \quad \omega_i \cap \omega_j = \emptyset, \quad i, j = 1, \dots, \tilde{N}, \quad i \neq j. \quad (20)$$

We call c_i the parent of the elements in subset ω_i and ω_i the child set of c_i . We can also iterate the above merging procedure on the newly created higher-level sampling space to get an even higher-level sampling space.

For example, we consider the 16-QAM modulation system often used in digital communications. The values of the symbols are taken from the set

$$\Omega = \{(a, b) : a, b = \pm 0.5, \pm 2.5\}. \quad (21)$$

As shown in Figure 1, the sampling space Ω can be divided into four disjoint subsets:

$$\begin{aligned} \omega_1 &= \{(0.5, 0.5), (0.5, 2.5), (2.5, 0.5), (2.5, 2.5)\}, \\ \omega_2 &= \{(-0.5, 0.5), (-0.5, 2.5), (-2.5, 0.5), (-2.5, 2.5)\}, \\ \omega_3 &= \{(-0.5, -0.5), (-0.5, -2.5), (-2.5, -0.5), (-2.5, -2.5)\}, \\ \omega_4 &= \{(0.5, -0.5), (0.5, -2.5), (2.5, -0.5), (2.5, -2.5)\}. \end{aligned} \quad (22)$$

Moreover, the centers of these four subspaces are $c_1 = (1.5, 1.5)$, $c_2 = (-1.5, 1.5)$, $c_3 = (-1.5, -1.5)$, and $c_4 = (1.5, -1.5)$. Thus, we have obtained a higher-level sampling space composed of four elements. Then the MKF can draw samples first from the highest-level sampling space and then from the associated child set in the next lower-level sampling space. The procedure is iterated until reaching the original sampling space.

For simplicity, we will use $c_{i,l}$ to represent the i th symbol value in the l th-level sampling space. Assume that there are, in total, L levels of sampling space and the number of elements at the l th level is $|\mathcal{A}_l|$. The original sampling space is defined as the first level. Then the multilevel MKF is summarized as follows.

Algorithm 3 (multilevel MKF). Suppose, at time $(t-1)$, a set of properly weighted samples $\{(\Lambda_{t-1}^{(j)}, \kappa_{t-1}^{(j)}, w_{t-1}^{(j)})\}_{j=1}^m$ is available with respect to $p(\Lambda_{t-1} \mid \mathbf{Y}_{t-1})$. Then at time t , as the new data \mathbf{y}_t becomes available, the following steps are implemented to update each weighted sample.

For $j = 1, \dots, m$, perform the following steps.

- (A1) Draw the sample $c_{t,L}^{(j)}$ in the L th-level sampling space.
 (a) Based on the new data \mathbf{y}_t , for each $c_{i,L} \in \mathcal{A}_L$, perform a one-step update on the corresponding Kalman filter $\kappa_{t-1}^{(j)}(\lambda_{t-1})$ assuming $c_{t,L} = c_{i,L}$ to obtain

$$\kappa_{t-1}^{(j)}(\lambda_{t-1}) \xrightarrow{\mathbf{y}_t, c_{i,L}} \kappa_{t,L}^{(j)}(\lambda_{t-1}, c_{i,L}). \quad (23)$$

- (b) For each $c_{i,L} \in \mathcal{A}_L$, compute the L th-level sampling density

$$\begin{aligned} \rho_{t,L}^{(i,j)} &\triangleq P(c_{t,L} = c_{i,L} \mid \Lambda_{t-1}^{(j)}, \mathbf{Y}_t) \\ &= p(\mathbf{y}_t \mid c_{i,L}, \Lambda_{t-1}^{(j)}, \mathbf{Y}_{t-1}) P(c_{i,L} \mid \Lambda_{t-1}^{(j)}, \mathbf{Y}_{t-1}). \end{aligned} \quad (24)$$

Note that by the model (1), the first density in (24) is Gaussian and can be computed based on the Kalman filter update (23).

- (c) Draw a sample $c_{t,L}^{(j)}$ from the L th-level sampling space \mathcal{A}_L according to the above sampling density, that is,

$$P(c_{t,L}^{(j)} = c_{i,L}) \propto \rho_{t,L}^{(i,j)}, \quad c_{i,L} \in \mathcal{A}_L. \quad (25)$$

- (A2) Draw the sample $c_{t,l}^{(j)}$ in the l th-level sampling space. First, find the child set $\omega_l^{(j)}$ in the current l th-level sampling space for the drawn sample $c_{t,l+1}^{(j)}$ in the $(l+1)$ th-level sampling space, then proceed with three more steps.

- (a) For each $c_{i,l}^{(j)}, i = 1, 2, \dots, |\omega_l^{(j)}|$ in the child set $\omega_l^{(j)}$, perform a one-step update on the corresponding Kalman filter $\kappa_{t-1}^{(j)}(\lambda_{t-1})$ to obtain

$$\kappa_{t-1}^{(j)}(\lambda_{t-1}) \xrightarrow{\mathbf{y}_t, c_{i,l}^{(j)}} \kappa_{t,l}^{(j)}(\lambda_{t-1}, c_{i,l}^{(j)}). \quad (26)$$

(b) For each $c_{i,l}^{(j)}$, compute the sampling density

$$\begin{aligned} \rho_{t,l}^{(i,j)} &\triangleq P(c_{t,l} = c_{i,l}^{(j)} \mid \Lambda_{t-1}^{(j)}, \mathbf{Y}_t) \\ &= p(\mathbf{y}_t \mid c_{i,l}^{(j)}, \Lambda_{t-1}^{(j)}, \mathbf{Y}_{t-1}) P(c_{i,l}^{(j)} \mid \Lambda_{t-1}^{(j)}, \mathbf{Y}_{t-1}). \end{aligned} \quad (27)$$

Note that the first density in (27) is also Gaussian and can be computed based on the Kalman filter update (26).

(c) Draw a sample $c_{t,l}^{(j)}$ according to the sampling density, that is,

$$P(c_{t,l}^{(j)} = c_{i,l}^{(j)}) \propto \rho_{t,l}^{(i,j)}, \quad c_{i,l}^{(j)} \in \omega_l^{(j)}. \quad (28)$$

Repeat the above steps for the next level until we draw a sample $\lambda_t^{(j)} = c_{t,1}^{(j)}$ from the original sampling space.

(A3) Append the symbol $\lambda_t^{(j)}$ to $\Lambda_{t-1}^{(j)}$ and obtain $\Lambda_t^{(j)}$.

(A4) Compute the trial sampling probability. Assuming the drawn sample $c_{t,l}^{(j)} = c_{i,l}^{(j)}$ in the l th-level sampling space and the associated sampling probability is $\rho_{t,l}^{(i,j)}$, then the effective sampling probability $\tilde{P}(\lambda_t^{(j)} \mid \Lambda_{t-1}^{(j)}, \mathbf{Y}_t)$ can be computed as follows:

$$\tilde{P}(\lambda_t^{(j)} \mid \Lambda_{t-1}^{(j)}, \mathbf{Y}_t) = \prod_{l=1}^L \rho_{t,l}^{(i,j)}. \quad (29)$$

(A5) Compute the importance weight

$$\begin{aligned} w_t^{(j)} &= w_{t-1}^{(j)} \frac{p(\Lambda_{t-1}^{(j)}, \lambda_t^{(j)} \mid \mathbf{Y}_t)}{p(\Lambda_{t-1}^{(j)} \mid \mathbf{Y}_{t-1}) \tilde{P}(\lambda_t^{(j)} \mid \Lambda_{t-1}^{(j)}, \mathbf{Y}_t)} \\ &= w_{t-1}^{(j)} \frac{p(\mathbf{y}_t \mid \lambda_t^{(j)}, \mathbf{Y}_{t-1}) P(\lambda_t^{(j)} \mid \Lambda_{t-1}^{(j)}, \mathbf{Y}_{t-1})}{\prod_{l=1}^L \rho_{t,l}^{(i,j)}}. \end{aligned} \quad (30)$$

(A6) Resample if the effective sample size is below a certain threshold, as discussed in Section 2.2.

Remark 1 (complexity). Note that the dominant computation required for the above multilevel MKF is the update of the Kalman filter in (23) and (26). Denote $J \triangleq |\mathcal{A}_1|$ and $K_l \triangleq |\omega_l|$. The number of one-step Kalman filter updates in the multilevel MKF is $N = \sum_{l=1}^L K_l$. Consider the 16-QAM and its corresponding two-level sampling space shown in Figure 1. There are $N = 8$ one-step Kalman filter updates needed by the multilevel MKF, whereas, the original MKF requires $J = 16$ Kalman updates for each Markov stream. Hence, the computation complexity is reduced by half by the multilevel MKF.

Remark 2 (properties of the weighted samples). From (30), we have

$$w_{t-1}^{(j)} = w_{t-2}^{(j)} \frac{p(\Lambda_{t-2}^{(j)}, \lambda_{t-1}^{(j)} \mid \mathbf{Y}_{t-1})}{p(\Lambda_{t-2}^{(j)} \mid \mathbf{Y}_{t-2}) \tilde{P}(\lambda_{t-1}^{(j)} \mid \Lambda_{t-2}^{(j)}, \mathbf{Y}_{t-1})}. \quad (31)$$

Substituting it into (30), and repeating the procedure with $w_{t-2}^{(j)}, \dots, w_1^{(j)}$, respectively, we finally have

$$\begin{aligned} w_t^{(j)} &= \frac{p(\Lambda_{t-1}^{(j)}, \lambda_t^{(j)} \mid \mathbf{Y}_t)}{\tilde{P}(\lambda_1^{(j)} \mid \Lambda_0^{(j)}, \mathbf{Y}_1) \cdots \tilde{P}(\lambda_{t-1}^{(j)} \mid \Lambda_{t-2}^{(j)}, \mathbf{Y}_{t-1}) \tilde{P}(\lambda_t^{(j)} \mid \Lambda_{t-1}^{(j)}, \mathbf{Y}_t)}. \end{aligned} \quad (32)$$

Consequently, the samples $\{\Lambda_t^{(j)}, w_t^{(j)}\}_{j=1}^m$ drawn by the above procedure are properly weighted with respect to $p(\Lambda_t \mid \mathbf{Y}_t)$ provided that $\{\Lambda_{t-1}^{(j)}, w_{t-1}^{(j)}\}_{j=1}^m$ are properly weighted with respect to $p(\Lambda_{t-1} \mid \mathbf{Y}_{t-1})$.

4. DELAYED MULTILEVEL MIXTURE KALMAN FILTER

The delayed-sample method is used to generate samples $\{(\lambda_t^{(j)}, w_t^{(j)})\}_{j=1}^m$ based on the observations $\mathbf{Y}_{t+\Delta}$, hence making $p(\Lambda_t \mid \mathbf{Y}_{t+\Delta})$ the target distribution at time $(t + \Delta)$. The procedure will provide better Monte Carlo samples since it utilizes the future observations $(\mathbf{y}_{t+1}, \dots, \mathbf{y}_{t+\Delta})$ in generating the current samples of λ_t . But the algorithm is also more demanding both analytically and computationally because of the need of marginalizing out $\lambda_{t+1}, \dots, \lambda_{t+\Delta}$.

Instead of exploring the “future” symbol sequences in the original sampling space, our proposed delayed multilevel MKF will marginalize out the future symbols in a higher-level sampling space. That is, for each possible “future” (relative to time $t - 1$) symbol sequence at time $(t + \Delta)$, that is,

$$(\lambda_t, c_{t+1,l}, \dots, c_{t+\Delta,l}) \in \mathcal{A}_1 \times \mathcal{A}_l^\Delta \quad (l > 1), \quad (33)$$

where $c_{t,l}$ is the symbol in the l th-level sampling space, we compute the value of a Δ -step Kalman filter $\{\kappa_{t+\tau}^{(j)}(\lambda_t, c_{t+1,l}^{t+\tau})\}_{\tau=1}^\Delta$, where

$$\begin{aligned} \kappa_{t+\tau}^{(j)}(\lambda_t, c_{t+1,l}^{t+\tau}) &\triangleq \left[\boldsymbol{\mu}_{t+\tau}^{(j)}(\Lambda_{t-1}^{(j)}, \lambda_t, c_{t+1,l}^{t+\tau}), \boldsymbol{\Sigma}_{t+\tau}^{(j)}(\Lambda_{t-1}^{(j)}, \lambda_t, c_{t+1,l}^{t+\tau}) \right], \quad \tau = 1, \dots, \Delta, \end{aligned} \quad (34)$$

with $c_{a,l}^b \triangleq (c_{a,l}, c_{a+1,l}, \dots, c_{b,l})$. The delayed multilevel MKF recursively propagates the samples properly weighted for $p(\Lambda_{t-1} \mid \mathbf{Y}_{t+\Delta-1})$ to those for $p(\Lambda_t \mid \mathbf{Y}_{t+\Delta})$ and is summarized as follows.

Algorithm 4 (delayed multilevel MKF). Suppose, at time $(t - 1)$, a set of properly weighted samples $\{(\Lambda_{t-1}^{(j)}, \kappa_{t-1}^{(j)}, w_{t-1}^{(j)})\}_{j=1}^m$ is available with respect to $p(\Lambda_{t-1} \mid \mathbf{Y}_{t+\Delta-1})$. Then at time t , as the new data $\mathbf{y}_{t+\Delta}$ becomes available, the following steps are implemented to update each weighted sample.

For $j = 1, \dots, m$, apply the following steps.

(A1) For each $\lambda_t = a_i \in \mathcal{A}_1$, and for each $c_{t+1,l}^{t+\Delta} \in \mathcal{A}_l^\Delta$, perform the update on the corresponding Kalman filter $\kappa_{t+\Delta-1}^{(j)}(\lambda_t, c_{t+1,l}^{t+\Delta-1})$, that is,

$$\kappa_{t+\tau-1}^{(j)}(\lambda_t, c_{t+1,l}^{t+\tau-1}) \xrightarrow{y_{t+\tau}, c_{t+\tau,l}} \kappa_{t+\tau}^{(j)}(\lambda_t, c_{t+1,l}^{t+\tau}). \quad (35)$$

(A2) For each $a_i \in \mathcal{A}_1$, compute the sampling density

$$\begin{aligned} \rho_{t,i}^{(j)} &\triangleq P(\lambda_t = a_i \mid \Lambda_{t-1}^{(j)}, \mathbf{Y}_{t+\Delta}) \\ &= \sum_{c_{t+1,l}^{t+\Delta} \in \mathcal{A}_l^\Delta} p(\mathbf{y}_t^{t+\Delta}, \lambda_t, c_{t+1,l}^{t+\Delta} \mid \Lambda_{t-1}^{(j)}, \mathbf{Y}_{t-1}^{(j)}) \\ &\propto P(\lambda_t = a_i \mid \Lambda_{t-1}^{(j)}) \\ &\quad \times \sum_{c_{t+1,l}^{t+\Delta} \in \mathcal{A}_l^\Delta} \left[\prod_{\tau=0}^{\Delta} p(\mathbf{y}_{t+\tau} \mid \mathbf{Y}_{t+\tau-1}, \Lambda_{t-1}^{(j)}, \lambda_t = a_i, c_{t+1,l}^{t+\tau}) \right. \\ &\quad \left. \times \prod_{\tau=1}^{\Delta} P(c_{t+\tau,l} \mid c_{t+1,l}^{t+\tau-1}, \lambda_t = a_i, \Lambda_{t-1}^{(j)}) \right]. \end{aligned} \quad (36)$$

(A3) Draw a sample $\lambda_t^{(j)}$ according to the above sampling density, that is,

$$P(\lambda_t^{(j)} = \lambda_i) \propto \rho_{t,i}^{(j)}. \quad (37)$$

(A4) Append the sample $\lambda_t^{(j)}$ to $\Lambda_{t-1}^{(j)}$ and obtain $\Lambda_t^{(j)}$.

(A5) Compute the importance weight

$$\begin{aligned} w_t^{(j)} &= w_{t-1}^{(j)} \frac{p(\Lambda_{t-1}^{(j)}, \lambda_t^{(j)} \mid \mathbf{Y}_{t+\Delta})}{p(\Lambda_{t-1}^{(j)} \mid \mathbf{Y}_{t+\Delta-1})P(\lambda_t^{(j)} \mid \Lambda_{t-1}^{(j)}, \mathbf{Y}_{t+\Delta})} \\ &\propto w_{t-1}^{(j)} \frac{\sum_{c_{t+1,l}^{t+\Delta} \in \mathcal{A}_l^\Delta} p(\mathbf{y}_t^{t+\Delta}, c_{t+1,l}^{t+\Delta}, \lambda_t^{(j)} \mid \Lambda_{t-1}^{(j)}, \mathbf{Y}_{t-1}^{(j)})}{\sum_{c_{t,l}^{t+\Delta-1} \in \mathcal{A}_l^{\Delta-1}} p(\mathbf{y}_t^{t+\Delta-1}, c_{t,l}^{t+\Delta-1} \mid \Lambda_{t-1}^{(j)}, \mathbf{Y}_{t-1}^{(j)})\rho_{t,i}^{(j)}} \\ &= w_{t-1}^{(j)} \frac{\sum_{c_{t+1,l}^{t+\Delta} \in \mathcal{A}_l^\Delta} \left[\prod_{\tau=1}^{\Delta} p(\mathbf{y}_{t+\tau} \mid \mathbf{Y}_{t+\tau-1}, C_{t+1,l}^{t+\tau}, \lambda_t^{(j)}, \Lambda_{t-1}^{(j)}) \right]}{\sum_{c_{t,l}^{t+\Delta-1} \in \mathcal{A}_l^{\Delta-1}} \left[\prod_{\tau=0}^{\Delta-1} p(\mathbf{y}_{t+\tau} \mid \mathbf{Y}_{t+\tau-1}, C_{t,l}^{t+\tau}, \Lambda_{t-1}^{(j)}) \right]} \\ &\quad \times \frac{P(\lambda_t^{(j)} \mid \Lambda_{t-1}^{(j)}, \mathbf{Y}_{t-1}^{(j)}) \prod_{\tau=1}^{\Delta} P(c_{t+\tau,l} \mid c_{t+1,l}^{t+\tau-1}, \Lambda_{t-1}^{(j)})}{\prod_{\tau=0}^{\Delta-1} P(c_{t+\tau,l} \mid C_{t,l}^{t+\tau-1}, \Lambda_{t-1}^{(j)})\rho_{t,i}^{(j)}}. \end{aligned} \quad (38)$$

(A6) Do resampling if the effective sample size is below a certain threshold, as discussed in Section 2.2.

Remark 3 (properties of the weighted samples). Similar to the multilevel sampling algorithm in Section 3, it can be shown that the samples $\{\Lambda_t^{(j)}, w_t^{(j)}\}_{j=1}^m$ drawn by the above procedure are properly weighted with respect to $p(\Lambda_t \mid \mathbf{Y}_{t+\Delta})$ provided that $\{\Lambda_{t-1}^{(j)}, w_{t-1}^{(j)}\}_{j=1}^m$ are properly weighted with respect to $p(\Lambda_{t-1} \mid \mathbf{Y}_{t+\Delta-1})$. However, the likelihood function $p(\mathbf{y}_{t+\tau} \mid \mathbf{Y}_{t+\tau-1}, C_{t,l}^{t+\tau}, \Lambda_{t-1}^{(j)})$ is not simply a Gaussian distribution any more, but a mixture of Gaussian components. Since the mixture of Gaussian distribution is implausible to be achieved within the rough sampling space, it has to be approximated with a Gaussian distribution computed by the

Kalman filter assuming that the elements in the higher-level sampling space are transmitted. Therefore, some bias will be introduced into the computation of the weight. On the other hand, we make use of more information in the approximation of better distribution $p(\Lambda_{t-1}^{(j)}, \lambda_t^{(j)} \mid \mathbf{Y}_{t+\Delta})$, which makes the algorithm more efficient than the original MKF.

Remark 4 (properly weighted samples). To mitigate the bias problem introduced in the weight computation, instead of (38), we can use the following importance weight:

$$\begin{aligned} w_t^{(j)} &= w_{t-1}^{(j)} \frac{p(\Lambda_{t-1}^{(j)}, \lambda_t^{(j)} \mid \mathbf{Y}_t)}{p(\Lambda_{t-1}^{(j)} \mid \mathbf{Y}_{t-1})\tilde{P}(\lambda_t^{(j)} \mid \Lambda_{t-1}^{(j)}, \mathbf{Y}_{t+\Delta})} \\ &= w_{t-1}^{(j)} \frac{p(\mathbf{y}_t \mid \lambda_t^{(j)}, \mathbf{Y}_{t-1})P(\lambda_t^{(j)} \mid \Lambda_{t-1}^{(j)}, \mathbf{Y}_{t-1})}{\rho_{t,i}^{(j)}}. \end{aligned} \quad (39)$$

Similar to the multilevel sampling algorithm in Section 3, it is easily seen that the samples $\{\Lambda_t^{(j)}, w_t^{(j)}\}_{j=1}^m$ drawn by the above procedure are properly weighted with respect to $p(\Lambda_t \mid \mathbf{Y}_t)$ provided that $\{\Lambda_{t-1}^{(j)}, w_{t-1}^{(j)}\}_{j=1}^m$ are properly weighted with respect to $p(\Lambda_{t-1} \mid \mathbf{Y}_{t-1})$. Since the whole procedure is just to get better samples based on the future information, delayed weight may be very effective. Furthermore, there is no bias anymore in the weight computation although we still approximate the mixture Gaussian distribution with a Gaussian distribution as in Remark 3.

Remark 5 (complexity). Note that the dominant computation required for the above delayed multilevel MKF is mainly in the first step. Denote $J \triangleq |\mathcal{A}_1|$ and $M \triangleq |\mathcal{A}_l|$. The number of one-step Kalman filter updates in the delayed multilevel MKF can be computed as follows:

$$N = J + JM + \dots + JM^\Delta = J \frac{M^{\Delta+1} - 1}{M - 1}. \quad (40)$$

Note that the delayed-sample MKF requires $J^{\Delta+1}$ Kalman updates at each time. Since usually $M \ll J$, compared with the delayed-sample MKF, the computational complexity of the delayed multilevel MKF is significantly reduced.

Remark 6 (alternative sampling method). To further reduce the computational complexity in the first step, we can take the alternative sampling method composed of the following steps.

Algorithm 5 (alternative sampling method). (1) At time $t+\Delta$, for each $c_{t+1,l}^{t+\Delta} \in \mathcal{A}_l^\Delta$, perform the update on the corresponding Kalman filter $\kappa_{t+\Delta-1}^{(j)}(c_{t+1,l}^{t+\Delta-1})$, that is,

$$\kappa_{t+\tau-1}^{(j)}(c_{t+1,l}^{t+\tau-1}) \xrightarrow{y_{t+\tau}, c_{t+\tau,l}} \kappa_{t+\tau}^{(j)}(c_{t+1,l}^{t+\tau}). \quad (41)$$

(2) Select K paths from $c_{t+1,l}^{t+\Delta}$ based on the computed weight

$$\prod_{\tau=0}^{\Delta} p(\mathbf{y}_{t+\tau} \mid \mathbf{Y}_{t+\tau-1}, \Lambda_{t-1}^{(j)}, c_{t+1,l}^{t+\tau})P(c_{t+\tau,l} \mid c_{t,l}^{t+\tau-1}, \Lambda_{t-1}^{(j)}). \quad (42)$$

(3) For each $\lambda_t = a_i \in \mathcal{A}_1$, and for each path k in the selected K paths, perform the update on the corresponding Kalman filter $\kappa_{t+\Delta-1}^{(j)}(\lambda_t, c_{t+1,l}^{t+\Delta-1,k})$, that is,

$$\kappa_{t+\tau-1}^{(j)}(\lambda_t, c_{t+1,l}^{t+\tau-1,k}) \xrightarrow{y_{t+\tau}, c_{t+\tau,l}^k} \kappa_{t+\tau}^{(j)}(\lambda_t, c_{t+1,l}^{t+\tau,k}). \quad (43)$$

(4) For each $a_i \in \mathcal{A}_1$, compute the sampling density

$$\begin{aligned} \rho_{t,i}^{(j)} &\triangleq P(\lambda_t = a_i | \mathbf{\Lambda}_{t-1}^{(j)}) \\ &\times \sum_{k=1}^K \left[\prod_{\tau=0}^{\Delta} p(\mathbf{y}_{t+\tau} | \mathbf{Y}_{t+\tau-1}, \mathbf{\Lambda}_{t-1}^{(j)}, \lambda_t = a_i, c_{t+1,l}^{t+\tau,k}) \right. \\ &\quad \left. \times \prod_{\tau=1}^{\Delta} P(c_{t+\tau,l}^k | c_{t+1,l}^{t+\tau-1,k}, \lambda_t = a_i, \mathbf{\Lambda}_{t-1}^{(j)}) \right]. \end{aligned} \quad (44)$$

The weight calculation can be computed by (40) for the target distribution $p(\lambda_t | \mathbf{Y}_t)$ or (45) for the target distribution $p(\lambda_t | \mathbf{Y}_{t+\Delta})$. Besides the bias that resulted from the Gaussian approximation in Remark 3, the latter also introduces new bias because of the summation over K selected paths, other than the whole sampling space in higher level. However, the first one does not introduce any bias as in Remark 4. Denote $J \triangleq |\mathcal{A}_1|$ and $M \triangleq |\mathcal{A}_l|$. The number of one-step Kalman filter updates in the delayed multilevel MKF can be computed as $N = JK + M^\Delta$.

Remark 7 (the choice of the parameters). Note that the performance of the delayed multilevel MKF is mainly dominated by the two important parameters: the number of prediction steps Δ and the specific level of the sampling space \mathcal{A}_l . With the same computation, the delayed multilevel MKF can see further “future” steps (larger Δ) with a coarser-level sampling space (larger l), whereas it can see the “future” samples in a finer sampling space (smaller l), but with smaller Δ steps.

Remark 8 (multiple sampling space). In Algorithm 5, we can also use different sampling spaces for different delay steps. That is, we can gradually increase the sampling space from the lower level to the higher level with an increase in the delay step.

5. SIMULATIONS

We consider the problem of adaptive detection in flat-fading channels in the presence of Gaussian noise. This problem is of fundamental importance in communication theory and an array of methodologies have been developed to tackle this problem. Specifically, the optimal detector for flat-fading channels with known channel statistics is studied in [24, 25], which has a prohibitively high complexity. Suboptimal receivers in flat-fading channels employ a two-stage structure, with a channel estimation stage followed by a sequence detection stage. Channel estimation is typically implemented by a Kalman filter or a linear predictor, and is facilitated by per-survivor processing (PSP) [26], decision feedback [27], pilot symbols [28], or a combination of the above [29].

Other suboptimal receivers for flat-fading channels include the method based on a combination of a hidden Markov model and a Kalman filtering [30], and the approach based on the expectation-maximization (EM) algorithm [31].

In the communication system, the transmitted data symbols $\{s_t\}$ take values from a finite alphabet set $\mathcal{A}_1 = \{a_1, \dots, a_{|\mathcal{A}_1|}\}$, and each symbol is transmitted over a Rayleigh fading channel. As shown in [32, 33], the fading process is adequately represented by an ARMA model, whose parameters are chosen to match the spectral characteristics of the fading process. That is, the fading process is modelled by the output of a lowpass Butterworth filter of order r driven by white Gaussian noise

$$\{\alpha_t\} = \frac{\Theta(D)}{\Phi(D)} \{u_t\}, \quad (45)$$

where D is the back-shift operator $D^k, u_t \triangleq u_{t-k}$; $\Phi(z) \triangleq \phi_r z^r + \dots + \phi_1 z + 1$; $\Theta(z) \triangleq \theta_r z^r + \dots + \theta_1 z + \theta_0$; and $\{u_t\}$ is a white complex Gaussian noise sequence with unit variance and independent real and complex components. The coefficients $\{\phi_i\}$ and $\{\theta_i\}$, as well as the order r of the Butterworth filter, are chosen so that the transfer function of the filter matches the power spectral density of the fading process, which in turn is determined by the channel Doppler frequency. On the other hand, a simpler method, which uses a two-path model to build ARMA process, can be found in [34]; the results there closely approximate more complex path models. Then such a communication system has the following state-space model form:

$$\mathbf{x}_t = \mathbf{F}\mathbf{x}_{t-1} + \mathbf{g}u_t, \quad (46)$$

$$y_t = s_t \mathbf{h}^H \mathbf{x}_t + \sigma v_t, \quad (47)$$

where

$$\mathbf{F} \triangleq \begin{pmatrix} -\phi_1 & -\phi_2 & \dots & -\phi_r & 0 \\ 1 & 0 & \dots & 0 & 0 \\ 0 & 1 & \dots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & 1 & 0 \end{pmatrix}, \quad \mathbf{g} \triangleq \begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix}. \quad (48)$$

The fading coefficient sequence $\{\alpha_t\}$ can then be written as

$$\alpha_t = \mathbf{h}^H \mathbf{x}_t, \quad \mathbf{h} \triangleq [\theta_0 \ \theta_1 \ \dots \ \theta_r]. \quad (49)$$

In the state-space model, $\{u_t\}$ in (46) and $\{v_t\}$ in (47) are the white complex Gaussian noise sequences with unit variance and independent real and imaginary components:

$$u_t \stackrel{\text{i.i.d.}}{\sim} \mathcal{N}_c(0, 1), \quad v_t \stackrel{\text{i.i.d.}}{\sim} \mathcal{N}_c(0, 1). \quad (50)$$

In our simulations, the fading process is specifically modelled by the output of a Butterworth filter of order $r = 3$ driven by a complex white Gaussian noise process. The cut-off frequency of this filter is 0.05, corresponding to a normalized Doppler frequency (with respect to the symbol rate $1/T$) $f_d T = 0.05$, which is a fast-fading scenario. That is, the fading

coefficients $\{\alpha_t\}$ are modeled by the following ARMA(3,3) process:

$$\begin{aligned} \alpha_t - 2.37409\alpha_{t-1} + 1.92936\alpha_{t-2} - 0.53208\alpha_{t-3} \\ = 10^{-2}(0.89409u_t + 2.68227u_{t-1} + 2.68227u_{t-2} \\ + 0.89409u_{t-3}), \end{aligned} \quad (51)$$

where $u_t \sim \mathcal{N}_c(0, 1)$. The filter coefficients in (51) are chosen such that $\text{Var}\{\alpha_t\} = 1$. However, a simpler method, which uses a two-path model to build ARMA process, can be found in [34].

We then apply the proposed multilevel MKF methods to the problem of online estimation of the a posteriori probability of the symbol s_t based on the received signals up to time t . That is, at time t , we need to estimate

$$P(s_t = a_i | \mathbf{Y}_t), \quad a_i \in \mathcal{A}_1. \quad (52)$$

Then a hard maximum a posteriori (MAP) decision on symbol s_t is given by

$$\hat{s}_t = \arg \max_{a_i \in \mathcal{A}_1} P(s_t = a_i | \mathbf{Y}_t). \quad (53)$$

If we obtain a set of Monte Carlo samples of the transmitted symbols $\{(S_t^{(j)}, w_t^{(j)})\}_{j=1}^m$, properly weighted with respect to $p(S_t | \mathbf{Y}_t)$, then the a posteriori symbol probability in (52) is approximated by

$$p(s_t = a_i | \mathbf{Y}_t) \cong \frac{1}{W_t} \sum_{j=1}^m \mathbf{1}(s_t^{(j)} = a_i) w_t^{(j)}, \quad a_i \in \mathcal{A}_1, \quad (54)$$

with $W_t \triangleq \sum_{j=1}^m w_t^{(j)}$.

In this paper, we use the 16-QAM modulation. Note that the 16-QAM modulation has much more phase ambiguities than the BPSK in [13]. We have provided the following two schemes to resolve phase ambiguities.

Scheme 1 (pilot-assisted). Pilot symbols are inserted periodically every fixed length of symbols; the similar scheme was used in [20]. In this paper, 10% and 20% pilot symbols are studied.

Scheme 2 (differential 16-QAM). We view the 16-QAM as a pair of QPSK symbols. Then two differential QPSKs will be used to resolve the phase ambiguity. Given the transmitted symbol s_{t-1} and information symbol d_t , they can be represented by the QPSK symbol pair as $(r_{s_{t-1},1}, r_{s_{t-1},2})$ and $(r_{d_t,1}, r_{d_t,2})$, respectively. Then the differential modulation scheme is given by

$$r_{s_t,1} = r_{s_{t-1},1} \cdot r_{d_t,1}, \quad r_{s_t,2} = r_{s_{t-1},2} \cdot r_{d_t,2}. \quad (55)$$

The two-QPSK symbol pair $(r_{s_t,1}, r_{s_t,2})$ will be mapped to the 16-QAM symbols and then transmitted through the fading channel. The traditional differential receiver performs the following steps:

$$r_{d_t,1} = r_{s_t,1} \cdot r_{s_{t-1},1}^*, \quad r_{d_t,2} = r_{s_t,2} \cdot r_{s_{t-1},2}^*. \quad (56)$$

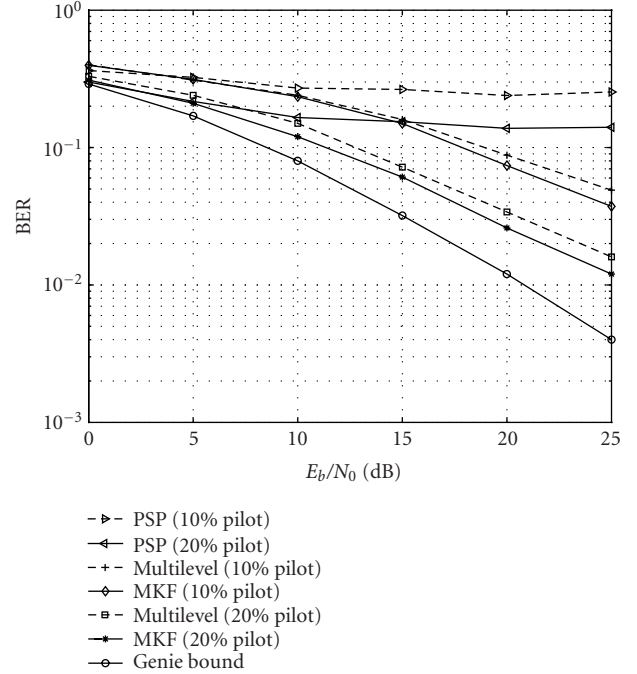


FIGURE 2: The BER performance of the PSP, MKF, and multilevel MKF for pilot-aided 16-QAM over flat-fading channels.

We next show the performance of the multilevel MKF algorithm for detecting 16-QAM over flat-fading channels. The receiver implements the decoding algorithms discussed in Section 3 in combination with the delayed-weight method under Schemes 1 and 2 discussed above. In our simulations, we take $m = 50$ Markov streams. The length of the symbol sequence is 256. We first show the bit error rate (BER) performance versus the signal-to-noise (SNR) by the PSP, the multilevel MKF, and the MKF receiver under different pilot schemes without delayed weight in Figure 2 and with delayed weight in Figure 3. The numbers of bit errors were collected over 50 independent simulations at low SNR or more at high SNR. In these figures, we also plotted the “genie-aided lower bound.”¹ The BER performance of PSP is far from the genie bound at SNR higher than 10 dB. But the performance of the multilevel MKF with 20% pilot symbol is very close to the genie bound. Furthermore, with the delayed-weight method, the performance of the multilevel MKF can be significantly improved. We next show the BER performance in Figure 4 under the differential coding scheme. As in the pilot-assisted scheme, the BER performance of PSP is far from the genie bound at SNR higher than 15 dB. On the contrary, the

¹The genie-aided lower bound is obtained as follows. We assume that at each time t , a genie provides the receiver with an observation of the modulation-free channel coefficient corrupted by additive white Gaussian noise with the same variance, that is, $\tilde{y}_t = \alpha_t + \tilde{n}_t$, where $\tilde{n}_t \sim \mathcal{N}_c(0, \sigma^2)$. The receiver employs a Kalman filter to track the fading process based on the information provided by the genie, that is, it computes $\hat{\alpha}_t = E\{\alpha_t | \tilde{\mathbf{y}}_t\}$. An estimate of the transmitted 16-QAM is obtained by slicing $(\mathbf{y}_t \hat{\alpha}_t^*)$.

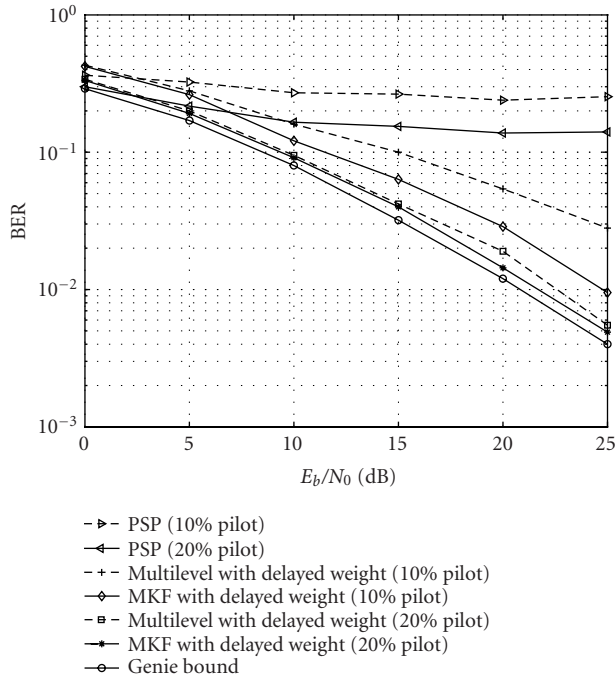


FIGURE 3: The BER performance of the PSP, the MKF with delayed weight ($\delta = 10$), and the multilevel MKF with delayed weight ($\delta = 10$) for pilot-aided 16-QAM over flat-fading channels.

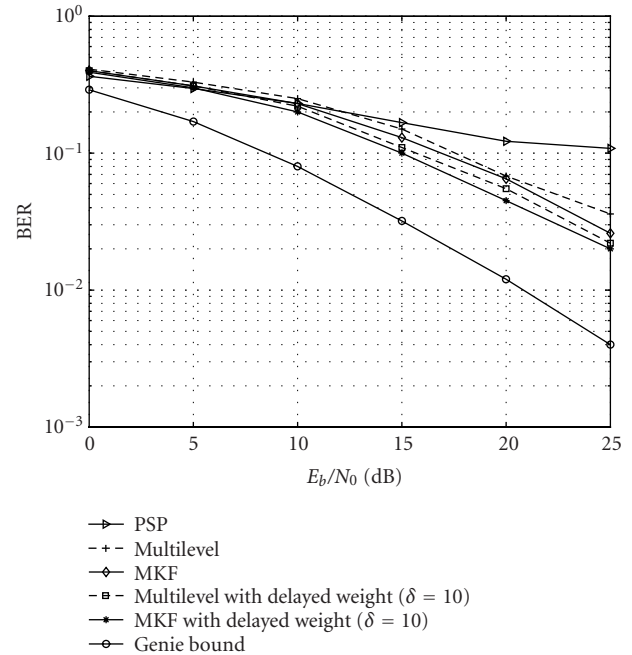


FIGURE 4: The BER performance of the PSP, the MKF, and the multilevel MKF for differential 16-QAM over flat-fading channels. The delayed-weight method is used with $\delta = 10$.

performance of the multilevel MKF is very close to that of the original MKF although there is a 5 dB gap between the genie bound and the performance of the multilevel MKF.

The BER performance of the MKF and the multilevel MKF with or without delayed weight versus the number of Markov streams is shown in Figure 5 under the differential coding scheme. The BER is gradually improved from the value 0.16 to the value about 0.08 for multilevel MKF, 0.07 for MKF, 0.065 for multilevel MKF with delayed weight, and 0.062 for MKF with delayed weight with 25 Markov streams. However, the BER performance can not be improved anymore with more than 25 streams. Therefore, the optimal number of Markov streams will be 25 in this example.

Next, we illustrate the performance of the delayed multilevel MKF. The receiver implements the decoding algorithms discussed in Section 4. We show the BER performance versus SNR in Figure 6, computed by the delayed-sample or the delayed multilevel MKF with one delayed step ($\Delta = 1$). The BER performance of the MKF and the MKF with delayed weight is also plotted in the same figure. In the delayed multilevel MKF, we implement two schemes for choosing the sampling space for the “future” symbols. In the first scheme, we choose the second-level ($l = 2$) sampling space $\mathcal{A}_2 = \{c_1, c_2, c_3, c_4\}$, where $c_i, i = 1, \dots, 4$, are the solid circles shown in Figure 1; and in the second scheme, we choose the third-level ($l = 3$) sampling space $\mathcal{A}_3 = \{c_1 + c_2, c_3 + c_4\}$. It is seen that the BER of the multilevel MKF is very close to that of the delayed-sample algorithm. It is also seen that the performance of the multilevel MKF method is conditioned on the specific level sampling space. The performance of the

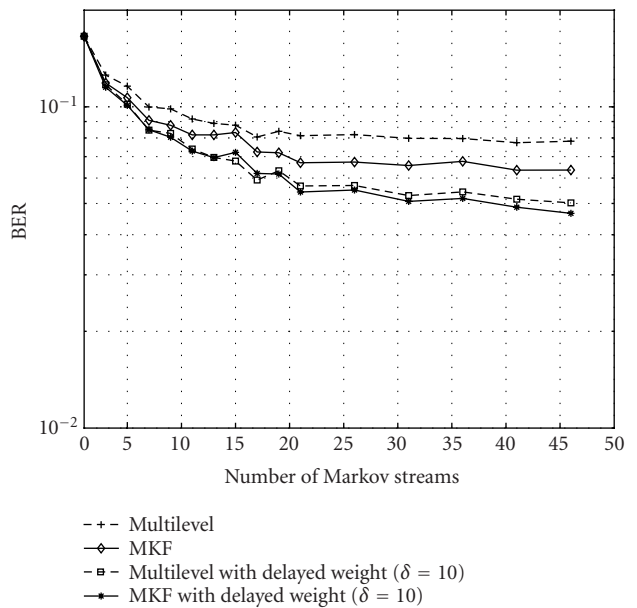


FIGURE 5: The BER performance of the MKF, and the multilevel MKF for differential 16-QAM over flat-fading channels versus the number of Markov streams. The delayed-weight method is used with $\delta = 10$.

delayed multilevel MKF based on the second-level sampling space \mathcal{A}_2 is nearly 2 dB better than that based on the third-level sampling space \mathcal{A}_3 .

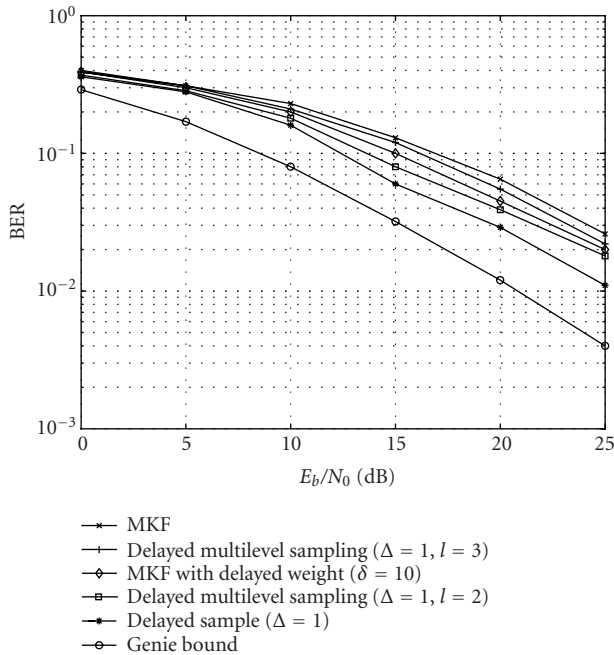


FIGURE 6: The BER performance of the delayed multilevel MKF with the second ($l = 2$)-level or the third ($l = 3$)-level sampling space for the differential 16-QAM system over flat-fading channels. The BER performance of the delayed-sample method is also shown.

6. CONCLUSION

In this paper, we have developed a new sequential Monte Carlo (SMC) sampling method—the multilevel mixture Kalman filter (MKF)—under the framework of MKF for conditional dynamic linear systems. This new scheme generates random streams using sequential importance sampling (SIS), based on the multilevel or hierarchical structure of the indicator random variables. This technique can also be used in conjunction with the delayed estimation methods, resulting in a delayed multilevel MKF. Moreover, the performance of both the multilevel MKF and the delayed multilevel MKF can be further enhanced when employed in conjunction with the delayed-weight method.

We have also applied the multilevel MKF algorithm and the delayed multilevel MKF algorithm to solve the problem of adaptive detection in flat-fading communication channels. It is seen that compared with the receiver algorithm based on the original MKF, the proposed multilevel MKF techniques offer very good performance. It is also seen that the receivers based on the delayed multilevel MKF can achieve similar performance as that based on the delayed-sample MKF, but with a much lower computational complexity.

ACKNOWLEDGMENT

This work was supported in part by the US National Science Foundation (NSF) under Grants DMS-0225692 and CCR-0225826, and by the US Office of Naval Research (ONR) under Grant N00014-03-1-0039.

REFERENCES

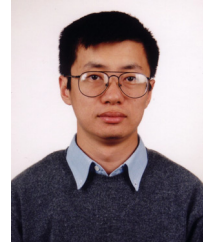
- [1] C. Andrieu, N. de Freitas, and A. Doucet, "Robust full Bayesian learning for radial basis networks," *Neural Computation*, vol. 13, no. 10, pp. 2359–2407, 2001.
- [2] E. Beadle and P. Djuric, "A fast-weighted Bayesian bootstrap filter for nonlinear model state estimation," *IEEE Trans. on Aerospace and Electronics Systems*, vol. 33, no. 1, pp. 338–343, 1997.
- [3] R. Chen and J. S. Liu, "Mixture Kalman filters," *Journal of the Royal Statistical Society: Series B*, vol. 62, no. 3, pp. 493–508, 2000.
- [4] C.-M. Cho and P. Djuric, "Bayesian detection and estimation of cisoids in colored noise," *IEEE Trans. Signal Processing*, vol. 43, no. 12, pp. 2943–2952, 1995.
- [5] P. M. Djuric and J. Chun, "An MCMC sampling approach to estimation of nonstationary hidden Markov models," *IEEE Trans. Signal Processing*, vol. 50, no. 5, pp. 1113–1123, 2002.
- [6] A. Doucet, N. de Freitas, and N. Gordon, Eds., *Sequential Monte Carlo in Practice*, Springer-Verlag, New York, NY, USA, 2001.
- [7] A. Doucet, S. J. Godsill, and C. Andrieu, "On sequential Monte Carlo sampling methods for Bayesian filtering," *Statistics and Computing*, vol. 10, no. 3, pp. 197–208, 2000.
- [8] W. Fong, S. J. Godsill, A. Doucet, and M. West, "Monte Carlo smoothing with application to audio signal enhancement," *IEEE Trans. Signal Processing*, vol. 50, no. 2, pp. 438–449, 2002.
- [9] Y. Huang and P. Djuric, "Multiuser detection of synchronous code-division multiple-access signals by perfect sampling," *IEEE Trans. Signal Processing*, vol. 50, no. 7, pp. 1724–1734, 2002.
- [10] A. Kong, J. S. Liu, and W. H. Wong, "Sequential imputations and Bayesian missing data problems," *Journal of the American Statistical Association*, vol. 89, no. 425, pp. 278–288, 1994.
- [11] J. S. Liu and R. Chen, "Sequential Monte Carlo methods for dynamic systems," *Journal of the American Statistical Association*, vol. 93, no. 443, pp. 1032–1044, 1998.
- [12] M. K. Pitt and N. Shephard, "Filtering via simulation: auxiliary particle filters," *Journal of the American Statistical Association*, vol. 94, no. 446, pp. 590–599, 1999.
- [13] R. Chen, X. Wang, and J. S. Liu, "Adaptive joint detection and decoding in flat-fading channels via mixture Kalman filtering," *IEEE Transactions on Information Theory*, vol. 46, no. 6, pp. 2079–2094, 2000.
- [14] X. Wang, R. Chen, and D. Guo, "Delayed-pilot sampling for mixture Kalman filter with application in fading channels," *IEEE Trans. Signal Processing*, vol. 50, no. 2, pp. 241–254, 2002.
- [15] A. Doucet, N. J. Gordon, and V. Kroshnamurthy, "Particle filters for state estimation of jump Markov linear systems," *IEEE Trans. Signal Processing*, vol. 49, no. 3, pp. 613–624, 2001.
- [16] M. West and J. Harrison, *Bayesian Forecasting and Dynamic Models*, Springer-Verlag, New York, NY, USA, 1989.
- [17] R. Y. Rubinstein, *Simulation and the Monte Carlo Method*, John Wiley & Sons, New York, NY, USA, 1981.
- [18] J. MacCormick and A. Blake, "A probabilistic exclusion principle for tracking multiple objects," *International Journal of Computer Vision*, vol. 39, no. 1, pp. 57–71, 2000.
- [19] D. Guo, X. Wang, and R. Chen, "Wavelet-based sequential Monte Carlo blind receivers in fading channels with unknown channel statistics," *IEEE Trans. Signal Processing*, vol. 52, no. 1, pp. 227–239, 2004.
- [20] E. Puskaya, C. Andrieu, A. Doucet, and W. Fitzgerald, "Particle filtering for demodulation in fading channels with non-Gaussian additive noise," *IEEE Trans. Communications*, vol. 49, no. 4, pp. 579–582, 2001.

- [21] D. Guo and X. Wang, "Blind detection in MIMO systems via sequential Monte Carlo," *IEEE Journal on Selected Areas in Communications*, vol. 21, no. 3, pp. 464–473, 2003.
- [22] J. Zhang and P. Djurić, "Joint estimation and decoding of space-time trellis codes," *EURASIP Journal on Applied Signal Processing*, vol. 2002, no. 3, pp. 305–315, 2002.
- [23] Z. Yang and X. Wang, "A sequential Monte Carlo blind receiver for OFDM systems in frequency-selective fading channels," *IEEE Trans. Signal Processing*, vol. 50, no. 2, pp. 271–280, 2002.
- [24] R. Haeb and H. Meyr, "A systematic approach to carrier recovery and detection of digitally phase modulated signals of fading channels," *IEEE Trans. Communications*, vol. 37, no. 7, pp. 748–754, 1989.
- [25] D. Makrakis, P. T. Mathiopoulos, and D. P. Boursas, "Optimal decoding of coded PSK and QAM signals in correlated fast fading channels and AWGN: a combined envelope, multiple differential and coherent detection approach," *IEEE Trans. Communications*, vol. 42, no. 1, pp. 63–75, 1994.
- [26] G. M. Vitetta and D. P. Taylor, "Maximum likelihood decoding of uncoded and coded PSK signal sequences transmitted over Rayleigh flat-fading channels," *IEEE Trans. Communications*, vol. 43, no. 11, pp. 2750–2758, 1995.
- [27] Y. Liu and S. D. Blostein, "Identification of frequency non-selective fading channels using decision feedback and adaptive linear prediction," *IEEE Trans. Communications*, vol. 43, no. 2/3/4, pp. 1484–1492, 1995.
- [28] H. Kong and E. Shwedyk, "On channel estimation and sequence detection of interleaved coded signals over frequency nonselective Rayleigh fading channels," *IEEE Trans. Vehicular Technology*, vol. 47, no. 2, pp. 558–565, 1998.
- [29] G. T. Irvine and P. J. McLane, "Symbol-aided plus decision-directed reception for PSK/TCM modulation on shadowed mobile satellite fading channels," *IEEE Journal on Selected Areas in Communications*, vol. 10, no. 8, pp. 1289–1299, 1992.
- [30] I. B. Collings and J. B. Moore, "An adaptive hidden Markov model approach to FM and M-ary DPSK demodulation in noisy fading channels," *Signal Processing*, vol. 47, no. 1, pp. 71–84, 1995.
- [31] C. N. Georghiades and J. C. Han, "Sequence estimation in the presence of random parameters via the EM algorithm," *IEEE Trans. Communications*, vol. 45, no. 3, pp. 300–308, 1997.
- [32] G. L. Stüber, *Principles of Mobile Communication*, Kluwer Academic Publishers, Boston, Mass, USA, 2000.
- [33] R. A. Ziegler and J. M. Cioffi, "Estimation of time-varying digital radio channels," *IEEE Trans. Vehicular Technology*, vol. 41, no. 2, pp. 134–151, 1992.
- [34] A. Duel-Hallen, "Decorrelating decision-feedback multiuser detector for synchronous code-division multiple-access channel," *IEEE Trans. Communications*, vol. 41, no. 2, pp. 285–290, 1993.

Dong Guo received the B.S. degree in geophysics and computer science from China University of Mining and Technology (CUMT), Xuzhou, China, in 1993; the M.S. degree in geophysics from the Graduate School, Research Institute of Petroleum Exploration and Development (RIPED), Beijing, China, in 1996. In 1999, he received the Ph.D. degree in applied mathematics from Beijing University, Beijing, China. He is now pursuing a second Ph.D. degree in the Department of Electrical Engineering, Columbia University, New York. His research interests are in the area of statistical signal processing and communications.



Xiaodong Wang received the B.S. degree in electrical engineering and applied mathematics from Shanghai Jiao Tong University, Shanghai, China, in 1992; the M.S. degree in electrical and computer engineering from Purdue University in 1995; and the Ph.D. degree in electrical engineering from Princeton University in 1998. From 1998 to 2001, he was an Assistant Professor in the Department of Electrical Engineering, Texas A&M University. In 2002, he joined the Department of Electrical Engineering, Columbia University, as an Assistant Professor. Dr. Wang's research interests fall in the general areas of computing, signal processing, and communications. He has worked in the areas of wireless communications, statistical signal processing, parallel and distributed computing, and bioinformatics. He has published extensively in these areas. Among his publications is a recent book entitled *Wireless Communication Systems: Advanced Techniques for Signal Reception*. Dr. Wang has received the 1999 NSF CAREER Award and the 2001 IEEE Communications Society and Information Theory Society Joint Paper Award. He currently serves as an Associate Editor for the IEEE Transactions on Signal Processing, the IEEE Transactions on Communications, the IEEE Transactions on Wireless Communications, and IEEE Transactions on Information Theory.



Rong Chen is a Professor at the Department of Information and Decision Sciences, College of Business Administration, University of Illinois at Chicago (UIC). Before joining UIC in 1999, he was with the Department of Statistics, Texas A&M University. Dr. Chen received his B.S. degree in mathematics from Peking University, China, in 1985; his M.S. and Ph.D. degrees in statistics from Carnegie Mellon University in 1987 and 1990, respectively. His main research interests are in time series analysis, statistical computing and Monte Carlo methods in dynamic systems, and statistical applications in engineering and business. He is an Associate Editor for the Journal of American Statistical Association, Journal of Business and Economic Statistics, Statistica Sinica, and Computational Statistics.



Resampling Algorithms for Particle Filters: A Computational Complexity Perspective

Miodrag Bolić

*Department of Electrical and Computer Engineering, State University of New York at Stony Brook,
Stony Brook, NY 11794-2350, USA
Email: mbolic@ece.sunysb.edu*

Petar M. Djurić

*Department of Electrical and Computer Engineering, State University of New York at Stony Brook,
Stony Brook, NY 11794-2350, USA
Email: djuric@ece.sunysb.edu*

Sangjin Hong

*Department of Electrical and Computer Engineering, State University of New York at Stony Brook,
Stony Brook, NY 11794-2350, USA
Email: snjhong@ece.sunysb.edu*

Received 30 April 2003; Revised 28 January 2004

Newly developed resampling algorithms for particle filters suitable for real-time implementation are described and their analysis is presented. The new algorithms reduce the complexity of both hardware and DSP realization through addressing common issues such as decreasing the number of operations and memory access. Moreover, the algorithms allow for use of higher sampling frequencies by overlapping in time the resampling step with the other particle filtering steps. Since resampling is not dependent on any particular application, the analysis is appropriate for all types of particle filters that use resampling. The performance of the algorithms is evaluated on particle filters applied to bearings-only tracking and joint detection and estimation in wireless communications. We have demonstrated that the proposed algorithms reduce the complexity without performance degradation.

Keywords and phrases: particle filters, resampling, computational complexity, sequential implementation.

1. INTRODUCTION

Particle filters (PFs) are very suitable for nonlinear and/or non-Gaussian applications. In their operation, the main principle is recursive generation of random measures, which approximate the distributions of the unknowns. The random measures are composed of particles (samples) drawn from relevant distributions and of importance weights of the particles. These random measures allow for computation of all sorts of estimates of the unknowns, including minimum mean square error (MMSE) and maximum a posteriori (MAP) estimates. As new observations become available, the particles and the weights are propagated by exploiting Bayes theorem and the concept of sequential importance sampling [1, 2].

The main goals of this paper are the development of resampling methods that allow for increased speeds of PFs, that require less memory, that achieve fixed timings regardless of the statistics of the particles, and that are computationally less complex. Development of such algorithms is extremely

critical for practical implementations. The performance of the algorithms is analyzed when they are executed on a digital signal processor (DSP) and specially designed hardware. Note that resampling is the only PF step that does not depend on the application or the state-space model. Therefore, the analysis and the algorithms for resampling are general.

From an algorithmic standpoint, the main challenges include development of algorithms for resampling that are suitable for applications requiring temporal concurrency.¹ A possibility of overlapping PF operations is considered because it directly affects hardware performance, that is, it increases speed and reduces memory access. We investigate sequential resampling algorithms and analyze their computational complexity metrics including the number of operations as well as the class and type of operation by performing behavioral profiling [3]. We do not consider fixed point

¹Temporal concurrency quantifies the expected number of operations that are simultaneously executed, that is, are overlapped in time.

precision issues, where a hardware solution of resampling suitable for fixed precision implementation has already been presented [4].

The analysis in this paper is related to the sample importance resampling (SIR) type of PFs. However, the analysis can be easily extended to any PF that performs resampling, for instance, the auxiliary SIR (ASIR) filter. First, in Section 2 we provide a brief review of the resampling operation. We then consider random and deterministic resampling algorithms as well as their combinations. The main feature of the random resampling algorithm, referred to as residual-systematic resampling (RSR) and described in Section 3, is to perform resampling in fixed time that does not depend on the number of particles at the output of the resampling procedure. The deterministic algorithms, discussed in Section 4, are threshold-based algorithms, where particles with moderate weights are not resampled. Thereby significant savings can be achieved in computations and in the number of times the memories are accessed. We show two characteristic types of deterministic algorithms: a low-complexity algorithm and an algorithm that allows for overlapping of the resampling operation with the particle generation and weight computation. The performance and complexity analysis are presented in Section 5 and the summary of our contributions is outlined in Section 6.

2. OVERVIEW OF RESAMPLING IN PFs

PFs are used for tracking states of dynamic state-space models described by the set of equations

$$\begin{aligned}\mathbf{x}_n &= f(\mathbf{x}_{n-1}) + \mathbf{u}_n, \\ \mathbf{y}_n &= g(\mathbf{x}_n) + \mathbf{v}_n,\end{aligned}\quad (1)$$

where \mathbf{x}_n is an evolving state vector of interest, \mathbf{y}_n is a vector of observations, \mathbf{u}_n and \mathbf{v}_n are independent noise vectors with known distributions, and $f(\cdot)$ and $g(\cdot)$ are known functions. The most common objective is to estimate \mathbf{x}_n as it evolves in time.

PFs accomplish tracking of \mathbf{x}_n by updating a random measure $\{\mathbf{x}_{1:n}^{(m)}, w_n^{(m)}\}_{m=1}^M$,² which is composed of M particles $\mathbf{x}_n^{(m)}$ and their weights $w_n^{(m)}$ defined at time instant n , recursively in time [5, 6, 7]. The random measure approximates the posterior density of the unknown trajectory $\mathbf{x}_{1:n}$, $p(\mathbf{x}_{1:n} | \mathbf{y}_{1:n})$, where $\mathbf{y}_{1:n}$ is the set of observations.

In the implementation of PFs, there are three important operations: particle generation, weight computation, and resampling. Resampling is a critical operation in particle filtering because with time, a small number of weights dominate the remaining weights, thereby leading to poor approximation of the posterior density and consequently to inferior estimates. With resampling, the particles with large weights are replicated and the ones with negligible weights are removed.

After resampling, the future particles are more concentrated in domains of higher posterior probability, which entails improved estimates.

The PF operations are performed according to

- (1) generation of particles (samples) $\mathbf{x}_n^{(m)} \sim \pi(\mathbf{x}_n | \mathbf{x}_{n-1}^{(i_{n-1}^{(m)})}, \mathbf{y}_{1:n})$, where $\pi(\mathbf{x}_n | \mathbf{x}_{n-1}^{(i_{n-1}^{(m)})}, \mathbf{y}_{1:n})$ is an importance density and $i_n^{(m)}$ is an array of indexes, which shows that the particle m should be reallocated to the position $i_n^{(m)}$;
- (2) computation of weights by

$$w_n^{*(m)} = \frac{w_{n-1}^{(i_{n-1}^{(m)})} p(\mathbf{y}_n | \mathbf{x}_n^{(m)}) p(\mathbf{x}_n^{(m)} | \mathbf{x}_{n-1}^{(i_{n-1}^{(m)})})}{a_{n-1}^{(i_{n-1}^{(m)})} \pi(\mathbf{x}_n^{(m)} | \mathbf{x}_{n-1}^{(i_{n-1}^{(m)})}, \mathbf{y}_{1:n})} \quad (2)$$

followed by normalization $w_n^{(m)} = w_n^{*(m)} / \sum_{j=1}^M w_n^{*(j)}$;

- (3) resampling $i_n^{(m)} \sim a_n^{(m)}$, where $a_n^{(m)}$ is a suitable resampling function whose support is defined by the particle $\mathbf{x}_n^{(m)}$ [8].

The above representation of the PF algorithm provides a certain level of generality. For example, the SIR filter with a stratified resampling is implemented by choosing $a_n^{(m)} = w_n^{(m)}$ for $m = 1, \dots, M$. When $a_n^{(m)} = 1/M$, there is no resampling and $i_n^{(m)} = m$. The ASIR filter can be implemented by setting $a_n^{(m)} = w_n^{(m)} p(\mathbf{y}_{n+1} | \boldsymbol{\mu}_{n+1}^{(m)})$ and $\pi(\mathbf{x}_n) = p(\mathbf{x}_n | \mathbf{x}_{n-1}^{(m)})$, where $\boldsymbol{\mu}_n^{(m)}$ is the mean, the mode, or some other likely value associated with the density $p(\mathbf{x}_n | \mathbf{x}_{n-1}^{(m)})$.

3. RESIDUAL SYSTEMATIC RESAMPLING ALGORITHM

In this section, we consider stratified random resampling algorithms, where $a_n^{(m)} = w_n^{(m)}$ [9, 10, 11]. Standard algorithms used for random resampling are different variants of stratified sampling such as residual resampling (RR) [12], branching corrections, [13] and systematic resampling (SR) [6]. SR is the most commonly used since it is the fastest resampling algorithm for computer simulations.

We propose a new resampling algorithm which is based on stratified resampling, and we refer to it as RSR [14]. Similar to RR, RSR calculates the number of times each particle is replicated except that it avoids the second iteration of RR when residual particles need to be resampled. Recall that in RR, the number of replications of a specific particle is determined in the first loop by truncating the product of the number of particles and the particle weight. In RSR instead, the updated uniform random number is formed in a different fashion, which allows for only one iteration loop and processing time that is independent of the distribution of the weights at the input. The RSR algorithm for N input and M output (resampled) particles is summarized by Algorithm 1.

Figure 1 graphically illustrates the SR and RSR methods for the case of $N = M = 5$ particles with weights given in Table 1. SR calculates the cumulative sum of the weights $C^{(m)} = \sum_{i=1}^m w_n^{(i)}$ and compares $C^{(m)}$ with the updated uniform number $U^{(m)}$ for $m = 1, \dots, N$. The uniform number

²The notation $\mathbf{x}_{1:n}$ signifies $\mathbf{x}_{1:n} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n\}$.

Purpose: generation of an array of indexes $\{i\}_1^N$ at time instant n , $n > 0$.
 Input: an array of weights $\{w_n\}_1^N$, input and output number of particles, N and M , respectively.
 Method:
 $(i) = \text{RSR}(N, M, w)$
 Generate a random number $\Delta U^{(0)} \sim \mathcal{U}[0, 1/M]$
 for $m = 1-N$
 $i^{(m)} = \lfloor (w_n^{(m)} - \Delta U^{(m-1)}) \cdot M \rfloor + 1$
 $\Delta U^{(m)} = \Delta U^{(m-1)} + i^{(m)}/M - w_n^{(m)}$
 end

ALGORITHM 1: Residual systematic resampling algorithm.

$U^{(0)}$ is generated by drawing from the uniform distribution $\mathcal{U}[0, 1/M]$ and updated by $U^{(m)} = U^{(m-1)} + 1/M$. The number of replications for particle m is determined as the number of times the updated uniform number is in the range $[C^{(m-1)}, C^{(m)})$. For particle 1, $U^{(0)}$ and $U^{(1)}$ belong to the range $[0, C^{(1)})$, so that this particle is replicated twice, which is shown with two arrows that correspond to particle 1. Particles 2 and 3 are replicated once. Particle 4 is discarded ($i^{(4)} = 0$) because no $U^{(m)}$ for $m = 1, \dots, N$ appears in the range $[C^{(3)}, C^{(4)})$.

The RSR algorithm draws the uniform random number $U^{(0)} = \Delta U^{(0)}$ in the same way but updates it by $\Delta U^{(m)} = \Delta U^{(m-1)} + i^{(m)}/M - w_n^{(m)}$. In the figure, we display both $U^{(m)} = \Delta U^{(m-1)} + i^{(m)}/M$ and $\Delta U^{(m)} = U^{(m)} - w_n^{(m)}$. Here, the uniform number is updated with reference to the *origin of the currently considered weight*, while in SR, it is propagated with reference to the *origin of the coordinate system*. The difference $\Delta U^{(m)}$ between the updated uniform number and the current weight is propagated. Figure 1 shows that $i^{(1)} = 2$ and that $\Delta U^{(1)}$ is calculated and then used as the initial uniform random number for particle 2. Particle 4 is discarded because $\Delta U^{(3)} = U^{(4)} > w^{(4)}$, so that $\lfloor (w_n^{(4)} - \Delta U^{(3)}) \cdot M \rfloor = -1$ and $i^{(4)} = 0$. If we compare $\Delta U^{(1)}$ with the relative position of the $U^{(2)}$ and $C^{(1)}$ in SR, $\Delta U^{(2)}$ in RSR with the relative position of $U^{(3)}$ and $C^{(2)}$ in SR, and so on, we see that they are equal. Therefore, SR and RSR produce identical resampling result.

3.1. Particle allocation and memory usage

We call particle allocation the way in which particles are placed in their new memory locations as a result of resampling. With proper allocation, we want to reduce the number of memory accesses and the size of state memory. The allocation is performed through index addressing, and its execution can be overlapped in time with the particle generation step. In Figure 2, three different outputs of resampling for the input weights from Figure 1 are considered. In Figure 2a, the indexes represent positions of the replicated particles. For example, $i^{(2)} = 1$ means that particle 1 replaces particle 2. Particle allocation is easily overlapped with particle generation using $\tilde{x}^{(m)} = x^{(i^{(m)})}$ for $m = 1, \dots, M$, where $\{\tilde{x}^{(m)}\}_{m=1}^M$ is the set of resampled particles. The randomness of the resampling output makes it difficult to realize in-place storage so that additional temporary memory for storing resampled

particles $\tilde{x}^{(m)}$ is necessary. In Figure 2a, particle 1 is replicated twice and occupies the locations of particles 1 and 2. Particle 2 is replicated once and must be stored in the memory of $\tilde{x}^{(m)}$ or it would be rewritten. We refer to this method as *particle allocation with index addressing*.

In Figure 2b, the indexes represent the number of times each particle is replicated. For example, $i^{(1)} = 2$ means that particle 1 is replicated twice. We refer to this method as *particle allocation with replication factors*. This method still requires additional memory for particles and memory for storing indexes.

The additional memory for storing the particles $\tilde{x}^{(m)}$ is not necessary if the particles are replicated to the positions of the discarded particles. We call this method *particle allocation with arranged indexes* of positions and replication factors (Figure 2c). Here, the addresses of both replicated particles and discarded particles as well as the number of times they are replicated (replication factor) are stored. The indexes are arranged in a way so that the replicated particles are placed in the upper and the discarded particles in the lower part of the index memory. In Figure 2c, the replicated particles take the addresses 1 – 4 and the discarded particle is on the address 5. When one knows in advance the addresses of the discarded particles, there is no need for additional memory for storing the resampled particles $\tilde{x}^{(m)}$ because the new particles are placed on the addresses occupied by the particles that are discarded. It is useful for PFs applied to multidimensional models since it avoids need for excessive memory for storing temporary particles.

For the RSR method, it is natural to use particle allocation with replication factor and arranged indexes because the RSR produces replication factors. In the particle generation step, the *for* loop with the number of iterations that corresponds to the replication factors is used for each replicated particle. The difference between the SR and the RSR methods is in the way the inner loop in the resampling step for SR and particle generation step for RSR is performed. Since the number of replicated particles is random, the *while* loop in SR has an unspecified number of operations. To allow for an unspecified number of iterations, complicated control structures in hardware are needed [15]. The main advantage of our approach is that the *while* loop of SR is replaced with a *for* loop with known number of iterations.

4. DETERMINISTIC RESAMPLING

4.1. Overview

In the literature, threshold-based resampling algorithms are based on the combination of RR and rejection control and they result in nondeterministic timing and increased complexity [8, 16]. Here, we develop threshold-based algorithms whose purpose is to reduce complexity and processing time. We refer to these methods as partial resampling (PR) because only a part of the particles is resampled.

In PR, the particles are grouped in two separate classes: one composed of particles with moderate weights and another with dominating and negligible weights. The particles

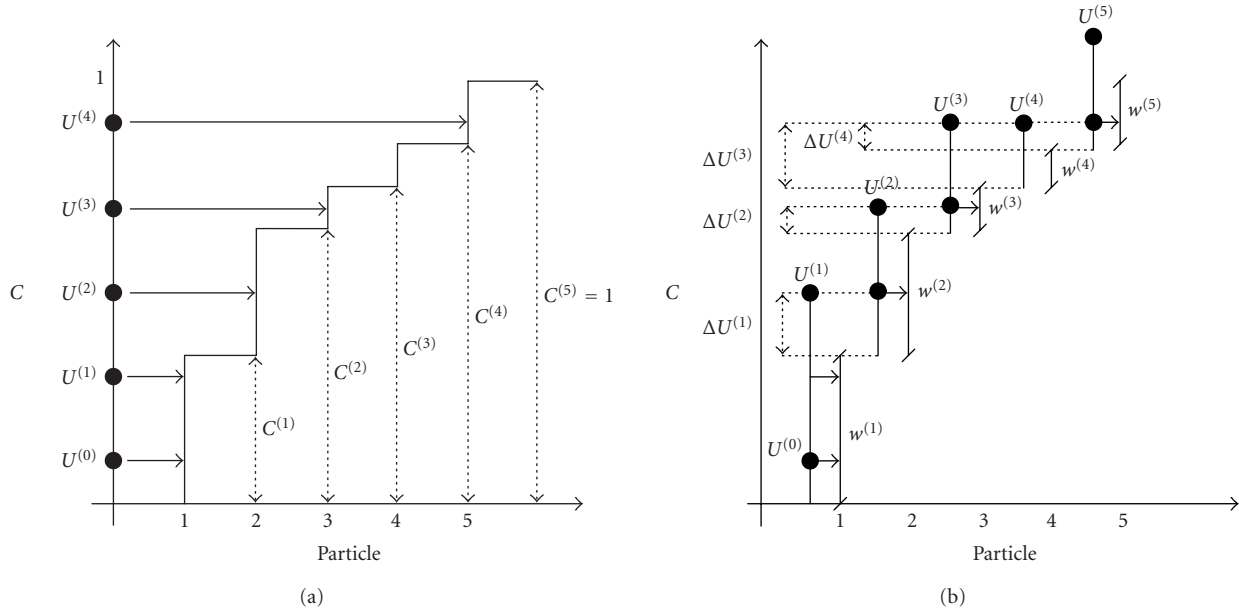


FIGURE 1: (a) Systematic and (b) residual systematic resampling for an example with $M = 5$ particles.

TABLE 1: Weights of particles.

m	$w^{(m)}$	$i^{(m)}$
1	7/20	2
2	6/20	1
3	2/20	1
4	2/20	0
5	3/20	1

with moderate weights are not resampled, whereas the negligible and dominating particles are resampled. It is clear that on average, resampling would be performed much faster because the particles with moderate weights are not resampled. We propose several PR algorithms which differ in the resampling function.

4.2. Partial resampling: suboptimal algorithms

PR could be seen as a way of a partial correction of the variance of the weights at each time instant. PR methods consist of two steps: one in which the particles are classified as moderate, negligible, or dominating and the other in which one determines the number of times each particle is replicated. In the first step of PR, the weight of each particle is compared with a high and a low threshold, T_h and T_l , respectively, where $T_h > 1/M$ and $0 < T_l < T_h$. Let the number of particles with weights greater than T_h and less than T_l be denoted by N_h and N_l , respectively. A sum of the weights of resampled particles is computed as a sum of dominating $W_h = \sum_{m=1}^{N_h} w_n^{(m)}$ for $w_n^{(m)} > T_h$ and negligible weights $W_l = \sum_{m=1}^{N_l} w_n^{(m)}$ for $w_n^{(m)} < T_l$. We define three different types of resampling with distinct resampling functions $a_n^{(m)}$.

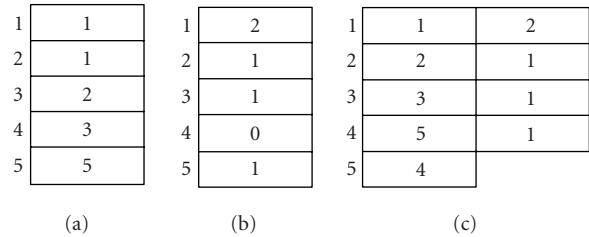


FIGURE 2: Types of memory usages: (a) indexes are positions of the replicated particles, (b) indexes are replication factors, and (c) indexes are arranged as positions and replication factors.

The resampling function of the first PR algorithm (PR1) is shown in Figure 3a and it corresponds to the stratified resampling case. The number of particles at the input and at the output of the resampling procedure is the same and equal to $N_h + N_l$. The resampling function is given by

$$a_n^{(m)} = \begin{cases} w_n^{(m)} & \text{for } w_n^{(m)} > T_h \text{ or } w_n^{(m)} < T_l, \\ \frac{1 - W_h - W_l}{M - N_h - N_l} & \text{otherwise.} \end{cases} \quad (3)$$

The second step can be performed using any resampling algorithm. For example, the RSR algorithm can be called using $(i) = \text{RSR}(N_h + N_l, N_h + N_l, w_n^{(m)} / (W_h + W_l))$, where the RSR is performed on the $N_h + N_l$ particles with negligible and dominating weights. The weights have to be normalized before they are processed by the RSR method.

The second PR algorithm (PR2) is shown in Figure 3b. The assumption that is made here is that most of the negligible particles will be discarded after resampling, and

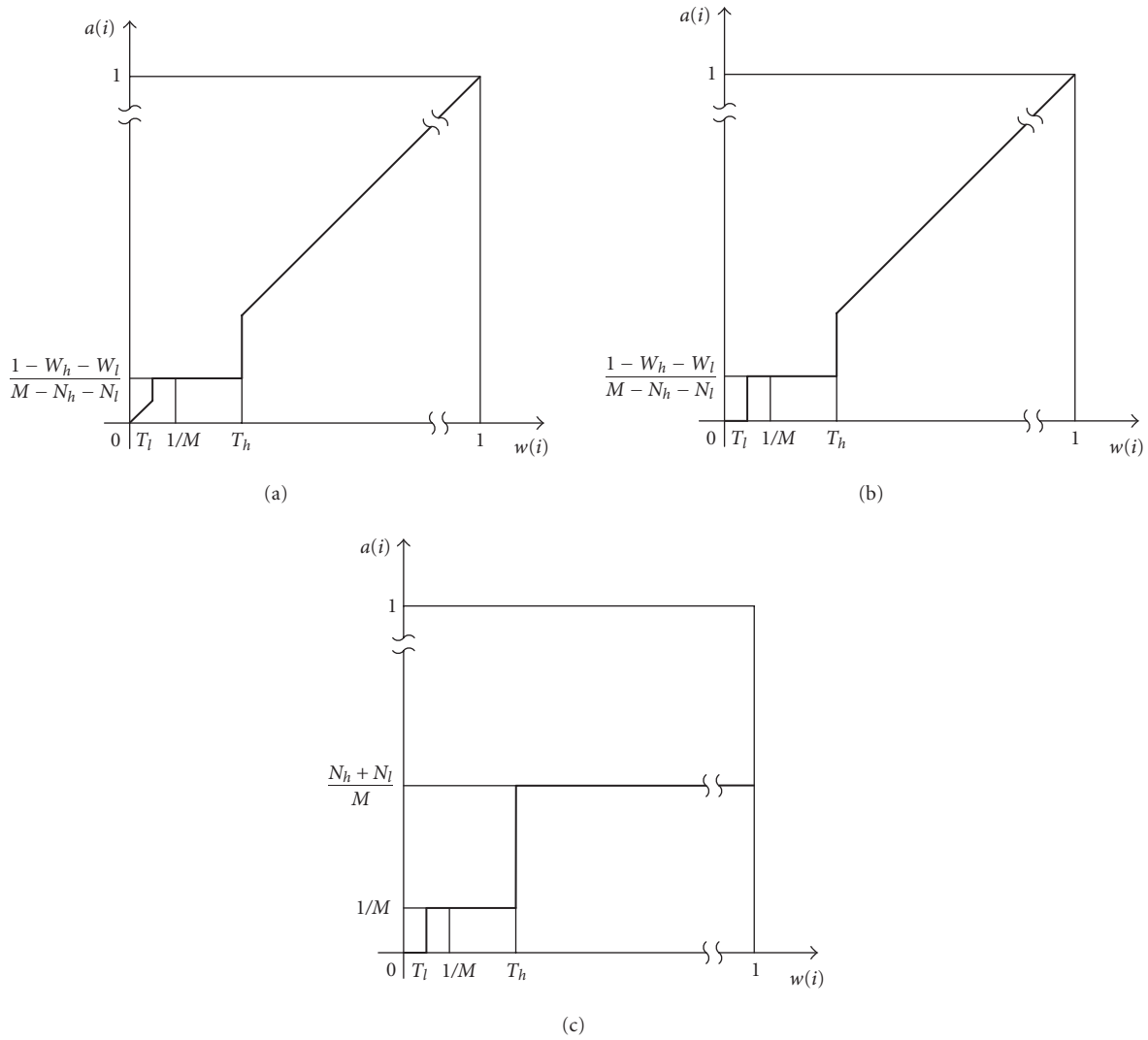


FIGURE 3: Resampling functions for the PR algorithms (a) PR1, (b) PR2, and (c) PR3.

consequently, particles with negligible weights are not used in the resampling procedure. Particles with dominating weights replace those with negligible weights with certainty. The resampling function is given as

$$a_n^{(m)} = \begin{cases} w_n^{(m)} + \frac{W_l}{N_h} & \text{for } w_n^{(m)} > T_h, \\ \frac{1 - W_h - W_l}{M - N_h - N_l} & \text{for } T_l < w_n^{(m)} < T_h, \\ 0 & \text{otherwise.} \end{cases} \quad (4)$$

The number of times each particle is replicated can be found using $(i) = \text{RSR}(N_h, N_h + N_l, (w_n^{(m)} + W_l/N_h)/(W_h + W_l))$, where the weights satisfy the condition $w_n^{(m)} > T_h$. There are only N_h input particles and $N_h + N_l$ particles are produced at the output.

The third PR algorithm (PR3) is shown in Figure 3c. The weights of all the particles above the threshold T_h are scaled with the same number. So, PR3 is a deterministic algorithm

whose resampling function is given as

$$a_n^{(m)} = \begin{cases} \frac{N_h + N_l}{M} & \text{for } w_n^{(m)} > T_h, \\ \frac{1}{M} & \text{for } T_l < w_n^{(m)} < T_h, \\ 0 & \text{otherwise.} \end{cases} \quad (5)$$

The number of replications of each dominating particle may be less by one particle than necessary because of the rounding operation. One way of resolving this problem is to assign that the first $N_t = N_l - \lfloor N_l/N_h \rfloor N_h$ dominating particles are replicated $r = \lfloor N_l/N_h \rfloor + 2$ times, while the rest of $N_h - N_t$ dominating particles are replicated $r = \lfloor N_l/N_h \rfloor + 1$ times. The weights are calculated as $w^{*(m)} = w^{(m)}$, where m represents positions of particles with moderate weights, and as $w^{*(l)} = w^{(m)}/r + W_l/(N_h + N_l)$, where m are positions of particles with dominating weights and l of particles with both dominating and negligible weights.

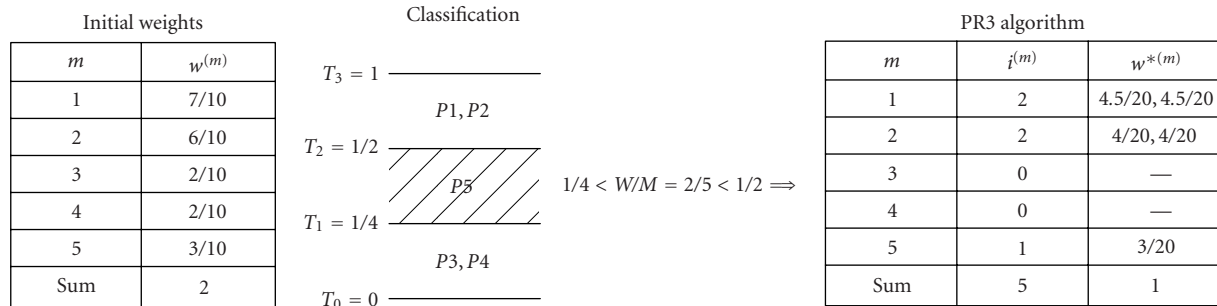


FIGURE 4: OPR method combined with the PR3 method used for final computation of weights and replication factors.

Another way of performing PR is to use a set of thresholds. The idea is to perform initial classification of the particles while the weights are computed and then to carry out the actual resampling together with the particle generation step. So, the resampling consists of two steps as in the PR2 algorithm, where classification of the particles is overlapped with the weight computation. We refer to this method as overlapped PR (OPR).

A problem with the classification of the particles is the necessity of knowing the overall sum of nonnormalized weights in advance. The problem can be resolved as follows. The particles are partitioned according to their weights. The thresholds for group i are defined as T_{i-1} , T_i for $i = 1, \dots, K$, where K is the number of groups, $T_{i-1} < T_i$ and $T_0 = 0$. The selection of thresholds is problem dependent. The thresholds that define the moderate group of particles satisfy $T_{k-1} < W/M < T_k$. The particles that have weights greater than T_k are dominant particles, and the ones with weights less than T_{k-1} , negligible particles.

In Figure 4, we provide a simple example of how this works. There are four thresholds (T_0 to T_3) and non-normalized particles are compared with the thresholds and properly grouped. After obtaining the sum of weights W , the second group for which $T_1 < W/M < T_2$, is the group of particles with moderate weights. The first group contains particles with negligible weights, and the third group is composed of particles with dominating weights. An additional loop is necessary to determine the number of times each of the dominating particles is replicated. However, the complexity of this loop is of order $O(K)$, which is several orders of magnitude lower than the complexity of the second step in the PR1 algorithm ($O(M)$). Because the weights are classified, it is possible to apply similar logic for the second resampling step as in the PR2 and PR3 algorithms. In the figure, the particles P1 and P2 are replicated twice and their weights are calculated using the formulae for weights for the PR3 method.

4.3. Discussion

In the PR1, PR2, and PR3 algorithms, the first step requires a loop of M iterations for the worst case (of number of computations) with two comparisons per each iteration (classification in three groups). Resampling in the PR1 algorithm is performed on $N_l + N_h$ particles. The worst case for the PR1 algorithm occurs when $N_l + N_h = M$, which means

that all the particles must be resampled, thereby implying that there cannot be improvements from an implementation standpoint. The main purpose of the PR2 algorithm is to improve the worst-case timing of the PR1 algorithm. Here, only N_h dominating particles are resampled. So, the input number of particles in the resampling procedure is N_h , while the output number of particles is $N_h + N_l$. If the RSR algorithm is used for resampling, then the complexity of the second step is $O(N_h)$.

PR1 and PR2 contain two loops and their timings depend on the weight statistics. As such, they do not have advantages for real-time implementation in comparison with RSR, which has only one loop of M iterations and whose processing time does not depend on the weight statistics. In the PR3 algorithm, there is no stratified resampling. The number of times each dominating particle is replicated is calculated after the first step and it depends on the current distribution of particle weights and of the thresholds. This number is calculated in $O(1)$ time, which means that there is no need for another loop in the second step. Thus, PR3 has simpler operations than the RSR algorithm.

The PR algorithms have the following advantages from the perspective of hardware implementation: (1) the resampling is performed faster on average because it is done on a much smaller number of particles; (2) there is a possibility of overlapping the resampling with the particle generation and weight computation; and (3) if the resampling is used in a parallel implementation [17], the number of exchanged particles among the processing elements is smaller because there are less particles to be replicated and replaced. There are also problems with the three algorithms. When $N_l = 0$ and $N_h = 0$, resampling is not necessary. However, when $N_l = 0$ or $N_h = 0$ but not at same time, the PR algorithms would not perform resampling even though it could be useful.

Application of the OPR algorithm requires a method for fast classification. For hardware and DSP implementation, it is suitable to define thresholds that are a power of two. So, we take that $T_i = 1/2^{K-i}$ for $i = 1, \dots, K$ and $T_0 = 0$. The group is determined by the position of the most significant "one" in the fixed point representation of weights. Memory allocation for the groups could be static or dynamic. Static allocation requires K memory banks, where the size of each bank is equal to the number of particles because all the particles could be located in one of the groups. Dynamic allocation is

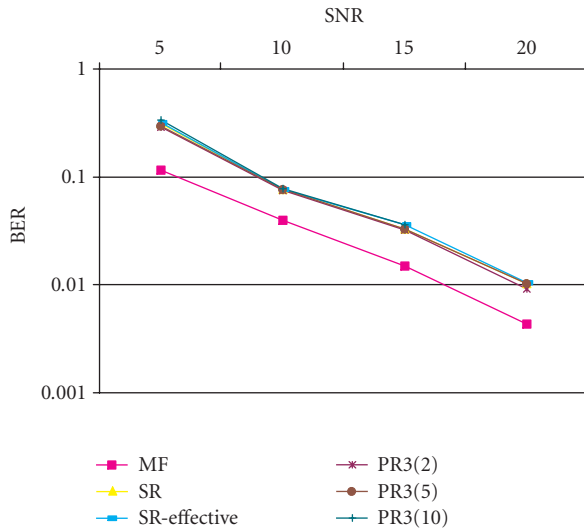


FIGURE 5: Performance of the PR3 algorithm for different threshold values applied to joint detection and estimation.

more efficient and it could be implemented using ways similar to the linked lists, where the element in a group contains two fields: the field with the address of the particle and the field that points out to the next element on the list. Thus, dynamic allocation requires memory with capacity of $2M$ words. As expected, overlapping increases the resources.

5. PARTICLE FILTERING PERFORMANCE AND COMPLEXITY

5.1. Performance analysis

The proposed resampling algorithms are applied and their performance is evaluated for the joint detection and estimation problem in communication [18, 19] and for the bearings-only tracking problem [7].

5.1.1. Joint detection and estimation

The experiment considered a Rayleigh fading channel with additive Gaussian noise with a differentially encoded BPSK modulation scheme. The detector was implemented for a channel with normalized Doppler spreads given by $B_d = 0.01$, which corresponds to fast fading. An AR(3) process was used to model the channel. The AR coefficients were obtained from the method suggested in [20]. The proposed detectors were compared with the *clairvoyant* detector, which performs matched filtering and detection assuming that the channel is known exactly by the receiver. The number of particles was $N = 1000$.

In Figure 5, the bit error rate (BER) versus signal-to-noise ratio (SNR) is depicted for the PR3 algorithm with different sets of thresholds, that is, $T_h = \{2M, 5M, 10M\}$ and $T_l = \{1/(2M), 1/(5M), 1/(10M)\}$. In the figure, the PR3 algorithm with the thresholds $2M$ and $1/2M$ is denoted as PR3(2), the one with thresholds $5M$ and $1/5M$ as PR3(5), and so on. The BERs for the matched filter (MF) and for

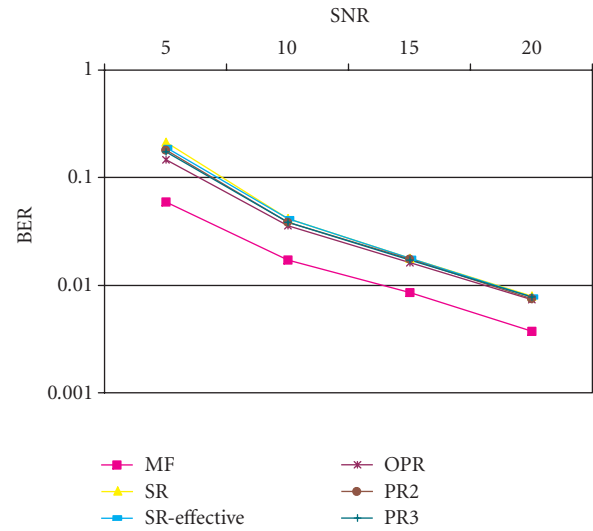


FIGURE 6: Comparison of the PR2, PR3, and OPR algorithms with SR applied to the joint detection and estimation problem.

the case when the SR is performed are shown as well. It is observed that the BER is similar for all types of resampling. However, the best results are obtained when the thresholds $2M$ and $1/2M$ were used. Here, the effective number of particles that is used is the largest in comparison with the PR3 algorithm with greater T_h and smaller T_l . This is a logical result because according to PR3, all the particles are concentrated in the narrower area between the two thresholds producing in this way a larger effective sample size. PR3 with thresholds $2M$ and $1/2M$ slightly outperforms the SR algorithm which is a bit surprising. The reason for this could be that the particles with moderate weights are not unnecessarily resampled in the PR3 algorithm. The same result is obtained even with different values of Doppler spread.

In Figure 6, BER versus SNR is shown for different resampling algorithms: PR2, PR3, OPR, and SR. The thresholds that are used for the PR2 and PR3 are $2M$ and $1/2M$. The OPR uses $K = 24$ groups and thresholds which are power of two. Again, all the results are comparable. The OPR and PR2 algorithms slightly outperform the other algorithms.

5.1.2. Bearings-only tracking

We tested the performance of PFs by applying the resampling algorithms to bearings-only tracking [7] with different initial conditions. In the experiment, PR2 and PR3 are used with two sets of threshold values, that is, $T_h = \{2M, 10M\}$ and $T_l = \{1/(2M), 1/(10M)\}$. In Figure 7, we show the number of times when the track is lost versus number of particles, for two different pairs of thresholds. We consider that the track is lost if all the particles have zero weights. In the figure, the PR3 algorithm with thresholds $2M$ and $1/2M$ is denoted as PR3(2) and the one $10M$ and $1/10M$ as PR3(10). The used algorithms are SR, SR performed after every 5th observation, PR2, and PR3. The resampling algorithms show again similar performances. The best results for PR2 and PR3 are obtained when the thresholds $10M$ and $1/10M$ are used.

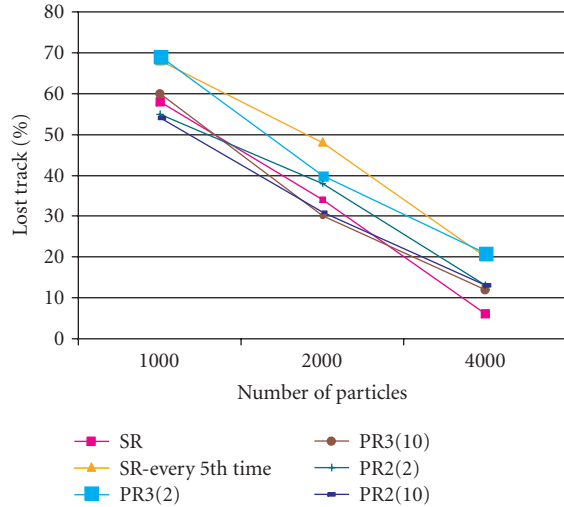


FIGURE 7: Number of times when the track is lost for the PR2, PR3, and SR applied to the bearings-only tracking problem.

5.2. Complexity analysis

The complexity of the proposed resampling algorithms is evaluated. We consider both computation complexity as well as memory requirements. We also present benefits of the proposed algorithms when concurrency in hardware is exploited.

5.2.1. Computational complexity

In Table 2, we provide a comparison of the different resampling algorithms. The results for RR are obtained for the worst-case scenario. The complexity of the RR, RSR, and PR algorithms is of $O(N)$, and the complexity of the SR algorithm is of $O(\max(N, M))$, where N and M are the input and output numbers of particles of the resampling procedure.

When the number of particles at the input of the resampling algorithm is equal to the number of particles at the output, the RR algorithm is by far the most complex. While the number of additions for the SR and RSR algorithms are the same, the RSR algorithm performs M multiplications. Since multiplication is more complex than addition, we can view that the SR is a less complex algorithm. However, when N is a power of two such that the multiplications by N are avoided, the RSR algorithm is the least complex.

The resampling algorithms SR, RSR, and PR3 were implemented on the Texas Instruments (TI) floating-point DSP TMS320C67xx. Several steps of profiling brought about five-fold speed-up when the number of resampled particles was 1000. The particle allocation step was not considered. The number of clock cycles per particle was around 18 for RSR and 4.1 for PR3. The SR algorithm does not have fixed timing. The mean duration was 24.125 cycles per particle with standard deviation of 5.17. On the processor TMS320C6711C whose cycle time is 5 nanoseconds, the processing of RSR with 1000 particles took 90 microseconds.

TABLE 2: Comparison of the number of operations for different resampling algorithms.

Operation	SR	RR	RSR	PR3
Multiplications	0	N	N	0
Additions	$2M + N$	$6N$	$3N$	$2N$
Comparisons	$N + M$	$3N$	0	$2N$

5.2.2. Memory requirements

In our analysis, we considered the memory requirement not only for resampling, but also for the complete PF. The memory size of the weights and the memory access during weight computation do not depend on the resampling algorithm. We consider particle allocation without indexes and with index addressing for the SR algorithm, and with arranged indexing for RSR, PR2, PR3, and OPR. For both particle allocation methods, the SR algorithm has to use two memories for storing particles. In Table 3, we can see the memory capacity for the RSR, PR2, and PR3 algorithms. The difference among these methods is only in the size of the index memory. For the RSR algorithm which uses particle allocation with arranged indexes, the index memory has a size of $2M$, where M words are used for storing the addresses of the particles that are replicated or discarded. The other M words represent the replication factors.

The number of resampled particles for the worst case of the PR2 algorithm corresponds to the number of particles in the RSR algorithm. Therefore, their index memories are of the same size. From an implementation standpoint, the most promising algorithm is the PR3 algorithm. It is the simplest one and it requires the smallest size of memory. The replication factor of the dominating particles is the same and of the moderate particles is one. So, the size of the index memory of PR3 is M , and it requires only one additional bit to represent whether a particle is dominant or moderate.

The OPR algorithm needs the largest index memory. When all the PF steps are overlapped, it requires a different access pattern than the other deterministic algorithms. Due to possible overwriting of indexes that are formed during the weight computation step with the ones that are read during particle generation, it is necessary to use two index-memory banks. Furthermore, particle generation and weight computation should access these memories alternately. Writing to the first memory is performed in the resampling step in one time instance whereas in the next one, the same memory is used by particle generation for reading. The second memory bank is used alternately. If we compare the memory requirements of the OPR algorithm with that of the PR3 algorithm, it is clear that OPR requires four times more memory for storing indexes for resampling.

5.2.3. PF speed improvements

The PF sampling frequency can be increased in hardware by exploiting temporal concurrency. Since there are no data dependencies among the particles in the particle generation and weight computation, the operations of these two steps

TABLE 3: Memory capacity for different resampling algorithms.

	SR without indexes	SR with indexes	RSR	PR2	PR3	OPR
States	$2N_sM$	$2N_sM$	N_sM	N_sM	N_sM	N_sM
Weights	M	M	M	M	M	M
Indexes	0	M	$2M$	$2M$	M	$4M$

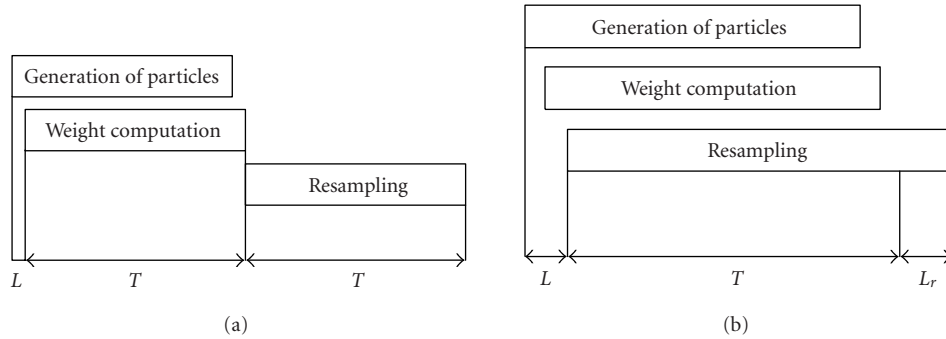


FIGURE 8: The timing of the PF with the (a) RSR or PR methods and (b) with the OPR method.

can be overlapped. Furthermore, the number of memory accesses is reduced because during weight computation, the values of the states do not need to be read from the memory since they are already in the registers.

The normalization step requires the use of an additional loop of M iterations as well as M divisions per observation. It has been noted that the normalization represents an unnecessary step which can be merged with the resampling and/or the computation of the importance weights. Avoidance of normalization requires additional changes which depend on whether resampling is carried out at each time instant and on the type of resampling. For PFs which perform SR or RSR at each time instant, the uniform random number in the resampling algorithm should be drawn from $[0, W_M/M]$ and updated with W_M/M , where W_M is the sum of the weights. Normalization in the PR methods could be avoided by including information about the sum W_M in the thresholds by using $T_{hn} = T_h W_M$ and $T_{ln} = T_l W_M$. With this approach, dynamic range problems for fixed precision arithmetics that appear usually with division are reduced. The computational burden is decreased as well because the number of divisions is reduced from M to 1.

The timing operations for a hardware implementation, where all the blocks are fine-grain pipelined are shown in Figure 8a. Here, the particle generation and weight calculation operations are overlapped in time and normalization is avoided. The symbol L is the constant hardware latency defined by the depth of pipelining in the particle generation and weight computation, T_{clk} is the clock period, M is the number of particles, and T is the minimum processing time of any of the basic PF operations. The SR is not suitable for hardware implementations, where fixed and minimal timings are required, because its processing time depends on the weight distribution and it is longer than MT_{clk} . So, in order to have

resampling operation performed in M clock cycles, RSR or PR3 algorithms with particle allocation with arranged indexes must be used. The minimum PF sampling period that can be achieved is $(2MT_{\text{clk}} + L)$.

OPR in combination with the PR3 algorithm allows for higher sampling frequencies. In the OPR, the classification of the particles is overlapped with the weight calculation as shown in Figure 8b. The symbol L_r is the constant latency of the part of the OPR algorithm that determines which group contains moderate, and which contains negligible and dominating particles. The latency L_r is proportional to the number of OPR groups. The speed of the PF can almost be increased twice if we consider pipelined hardware implementation. In Figure 8b, it is obvious that the PF processing time is reduced to $(MT_{\text{clk}} + L + L_r)$.

5.3. Final remarks

We summarize the impact of the proposed resampling algorithms on the PF speed and memory requirements.

- (1) The RSR is an improved RR algorithm with higher speed and fixed processing time. As such, besides for hardware implementations, it is a better algorithm for resampling that is executed on standard computers.
- (2) Memory requirements are reduced. The number of memory access and the size of the memory are reduced when RSR or any of PR algorithms are used for multidimensional state-space models. These methods can be appropriate for both hardware and DSP applications, where the available memory is limited. When the state-space model is one dimensional, then there is no purpose of adding an index memory and introducing a more complex control. In this case, the SR algorithm is recommended.

- (3) In hardware implementation and with the use of temporal concurrency, the PF sampling frequency can be considerably improved. The best results are achieved for the OPR algorithm at the expense of hardware resources.
- (4) The average amount of operations is reduced. This is true for PR1, PR2, and PR3 since they perform resampling on a smaller number of particles. This is desirable in PC simulations and some DSP applications.

6. CONCLUSION

Resampling is a critical step in the hardware implementation of PFs. We have identified design issues of resampling algorithms related to execution time and storage requirement. We have proposed new resampling algorithms whose processing time is not random and that are more suitable for hardware implementation. The new resampling algorithms reduce the number of operations and memory access or allow for overlapping the resampling step with weight computation and particle generation. While these algorithms minimize performance degradation, their complexity is reduced remarkably. We have also provided performance analysis of PFs that use our resampling algorithms when applied to joint detection and estimation in wireless communications and bearings-only tracking. Even though the algorithms are developed with the aim of improving the hardware implementation, these algorithms should also be considered as resampling methods in simulations on standard computers.

ACKNOWLEDGMENT

This work has been supported under the NSF Awards CCR-0082607 and CCR-0220011.

REFERENCES

- [1] J. M. Bernardo and A. F. M. Smith, *Bayesian Theory*, John Wiley & Sons, New York, NY, USA, 1994.
- [2] J. Geweke, "Antithetic acceleration of Monte Carlo integration in Bayesian inference," *Journal of Econometrics*, vol. 38, no. 1-2, pp. 73-89, 1988.
- [3] A. Gerstlauer, R. Domer, J. Peng, and D. Gajski, *System Design: A Practical Guide with SpecC*, Kluwer Academic Publishers, Boston, Mass, USA, 2001.
- [4] S. Hong, M. Bolić, and P. M. Djurić, "An efficient fixed-point implementation of residual resampling scheme for high-speed particle filters," *IEEE Signal Processing Letters*, vol. 11, no. 5, pp. 482-485, 2004.
- [5] S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking," *IEEE Trans. Signal Processing*, vol. 50, no. 2, pp. 174-188, 2002.
- [6] A. Doucet, N. de Freitas, and N. Gordon, Eds., *Sequential Monte Carlo Methods in Practice*, Springer, New York, NY, USA, 2001.
- [7] N. J. Gordon, D. J. Salmond, and A. F. M. Smith, "Novel approach to nonlinear/non-Gaussian Bayesian state estimation," *IEE Proceedings Part F: Radar and Signal Processing*, vol. 140, no. 2, pp. 107-113, 1993.
- [8] J. S. Liu, R. Chen, and T. Logvinenko, "A theoretical framework for sequential importance sampling and resampling," in *Sequential Monte Carlo Methods in Practice*, A. Doucet, N. de Freitas, and N. Gordon, Eds., pp. 225-246, Springer, New York, NY, USA, January 2001.
- [9] C. Berzuini, N. G. Best, W. R. Gilks, and C. Larizza, "Dynamic conditional independence models and Markov chain Monte Carlo methods," *Journal of the American Statistical Association*, vol. 92, no. 440, pp. 1403-1412, 1997.
- [10] A. Kong, J. S. Liu, and W. H. Wong, "Sequential imputations and Bayesian missing data problems," *Journal of American Statistical Association*, vol. 89, no. 425, pp. 278-288, 1994.
- [11] J. S. Liu and R. Chen, "Blind deconvolution via sequential imputations," *Journal of American Statistical Association*, vol. 90, no. 430, pp. 567-576, 1995.
- [12] E. R. Beadle and P. M. Djurić, "A fast-weighted Bayesian bootstrap filter for nonlinear model state estimation," *IEEE Trans. on Aerospace and Electronic Systems*, vol. 33, no. 1, pp. 338-343, 1997.
- [13] D. Crisan, P. Del Moral, and T. J. Lyons, "Discrete filtering using branching and interacting particle systems," *Markov Processes and Related Fields*, vol. 5, no. 3, pp. 293-318, 1999.
- [14] M. Bolić, P. M. Djurić, and S. Hong, "New resampling algorithms for particle filters," in *Proc. IEEE Int. Conf. Acoustics, Speech, Signal Processing (ICASSP '03)*, vol. 2, pp. 589-592, Hong Kong, China, April 2003.
- [15] K. Compton and S. Hauck, "Reconfigurable computing: a survey of systems and software," *ACM Computing Surveys*, vol. 34, no. 2, pp. 171-210, 2002.
- [16] J. S. Liu, R. Chen, and W. H. Wong, "Rejection control and sequential importance sampling," *Journal of American Statistical Association*, vol. 93, no. 443, pp. 1022-1031, 1998.
- [17] M. Bolić, P. M. Djurić, and S. Hong, "Resampling algorithms and architectures for distributed particle filters," submitted to *IEEE Trans. Signal Processing*.
- [18] R. Chen, X. Wang, and J. S. Liu, "Adaptive joint detection and decoding in flat-fading channels via mixture Kalman filtering," *IEEE Transactions on Information Theory*, vol. 46, no. 6, pp. 2079-2094, 2000.
- [19] P. M. Djurić, J. H. Kotecha, J. Zhang, et al., "Particle filtering," *IEEE Signal Processing Magazine*, vol. 20, no. 5, pp. 19-38, 2003.
- [20] H. Zamiri-Jafarian and S. Pasupathy, "Adaptive MLSD receiver with identification of flat fading channels," in *Proc. IEEE Vehicular Technology Conference*, vol. 2, pp. 695-699, 1997.

Miodrag Bolić received the B.S. and M.S. degrees in electrical engineering and computer sciences (EECS) from the University of Belgrade, Yugoslavia, in 1996 and 2001, respectively. He is now a Ph.D. student at the State University of New York (SUNY), Stony Brook, USA, and he is working part-time for Symbol Technologies, Holtsville, NY. Before joining SUNY, he worked at the Institute of Nuclear Science Vinča, Yugoslavia. His research is related to the development and modification of the statistical signal processing algorithms for more efficient hardware implementation and VLSI architectures for digital signal processing.



Petar M. Djurić received his B.S. and M.S. degrees in electrical engineering from the University of Belgrade in 1981 and 1986, respectively, and his Ph.D. degree in electrical engineering from the University of Rhode Island in 1990. From 1981 to 1986, he was a Research Associate with the Institute of Nuclear Sciences, Vinča, Belgrade. Since 1990, he has been with State University of New York (SUNY), Stony Brook, where he is a Professor in the Department of Electrical and Computer Engineering. He works in the area of statistical signal processing, and his primary interests are in the theory of modeling, detection, estimation, and time series analysis and its application to a wide variety of disciplines including wireless communications and biomedicine.



Sangjin Hong received the B.S. and M.S. degrees in electrical engineering and computer sciences (EECS) from the University of California, Berkeley. He received his Ph.D. degree in EECS from the University of Michigan, Ann Arbor. He is currently with the Department of Electrical and Computer Engineering at State University of New York (SUNY), Stony Brook. Before joining SUNY, he worked at Ford Aerospace Corp. Computer Systems Division as a Systems Engineer. He also worked at Samsung Electronics in Korea as a Technical Consultant. His current research interests are in the areas of low-power VLSI design of multimedia wireless communications and digital signal processing systems, reconfigurable SoC design and optimization, VLSI signal processing, and low-complexity digital circuits. Prof. Hong served on numerous technical program committees for IEEE conferences. Prof. Hong is a Member of IEEE.



A New Class of Particle Filters for Random Dynamic Systems with Unknown Statistics

Joaquín Míguez

*Departamento de Electrónica e Sistemas, Universidade da Coruña, Facultade de Informática, Campus de Elviña s/n,
15071 A Coruña, Spain
Email: jmiguez@udc.es*

Mónica F. Bugallo

*Department of Electrical and Computer Engineering, State University of New York at Stony Brook, Stony Brook,
NY 11794-2350, USA
Email: monica@ece.sunysb.edu*

Petar M. Djurić

*Department of Electrical and Computer Engineering, State University of New York at Stony Brook, Stony Brook,
NY 11794-2350, USA
Email: djuric@ece.sunysb.edu*

Received 4 May 2003; Revised 29 January 2004

In recent years, particle filtering has become a powerful tool for tracking signals and time-varying parameters of random dynamic systems. These methods require a mathematical representation of the dynamics of the system evolution, together with assumptions of probabilistic models. In this paper, we present a new class of particle filtering methods that do not assume explicit mathematical forms of the probability distributions of the noise in the system. As a consequence, the proposed techniques are simpler, more robust, and more flexible than standard particle filters. Apart from the theoretical development of specific methods in the new class, we provide computer simulation results that demonstrate the performance of the algorithms in the problem of autonomous positioning of a vehicle in a 2-dimensional space.

Keywords and phrases: particle filtering, dynamic systems, online estimation, stochastic optimization.

1. INTRODUCTION

Many problems in signal processing can be stated in terms of the estimation of an unobserved discrete-time random signal in a dynamic system of the form

$$\mathbf{x}_t = f_x(\mathbf{x}_{t-1}) + \mathbf{u}_t, \quad t = 1, 2, \dots, \quad (1)$$

$$\mathbf{y}_t = f_y(\mathbf{x}_t) + \mathbf{v}_t, \quad t = 1, 2, \dots, \quad (2)$$

where

- (a) $\mathbf{x}_t \in \mathbb{R}^{L_x}$ is the signal of interest, which represents the system state at time t ;
- (b) $f_x : \mathbb{R}^{L_x} \rightarrow I_x \subseteq \mathbb{R}^{L_x}$ is a (possibly nonlinear) state transition function;
- (c) $\mathbf{u}_t \in \mathbb{R}^{L_x}$ is the state perturbation or system noise at time t ;
- (d) $\mathbf{y}_t \in \mathbb{R}^{L_y}$ is the vector of observations collected at time t , which depends on the system state;

(e) $f_y : \mathbb{R}^{L_x} \rightarrow I_y \subseteq \mathbb{R}^{L_y}$ is a (possibly nonlinear) transformation of the state;

(f) $\mathbf{v}_t \in \mathbb{R}^{L_y}$ is the observation noise vector at time t , assumed statistically independent from the system noise \mathbf{u}_t .

Equation (1) describes the dynamics of the system state vector and, hence, it is usually termed *state equation* or *system equation*, whereas (2) is commonly referred to as *observation equation* or *measurement equation*. It is convenient to distinguish the structure of the dynamic system defined by the functions f_x and f_y from the associated probabilistic model, which depends on the probability distribution of the noise signals and the a priori distribution of the state, that is, the statistics of \mathbf{x}_0 .

We denote the a priori probability density function (pdf) of a random signal s as $p(s)$. If the signal s is statistically dependent on some observation z , then we write the conditional (or a posteriori) pdf as $p(s|z)$. From the Bayesian point of view, all the information relevant for the estimation

of the state at time t is contained in the so-called *filtering pdf*, that is, the a posteriori density of the system state given the observations up to time t ,

$$p(\mathbf{x}_t | \mathbf{y}_{1:t}), \quad (3)$$

where $\mathbf{y}_{1:t} = \{\mathbf{y}_1, \dots, \mathbf{y}_t\}$. The density (3) usually involves a multidimensional integral which does not have a closed-form solution for an arbitrary choice of the system structure and the probabilistic model. Indeed, analytical solutions can only be obtained for particular setups. The most classical example occurs when f_x and f_y are linear functions and the noise processes are Gaussian with known parameters. Then the filtering pdf of \mathbf{x}_t is itself Gaussian, with mean \mathbf{m}_t and covariance matrix \mathbf{C}_t , which we denote as

$$p(\mathbf{x}_t | \mathbf{y}_{1:t}) = \mathcal{N}(\mathbf{m}_t, \mathbf{C}_t), \quad (4)$$

where the posterior parameters \mathbf{m}_t and \mathbf{C}_t can be recursively computed, as time evolves, using the elegant algorithm known as Kalman filter (KF) [1]. Unfortunately, the assumptions of linearity and Gaussianity do not hold for most real-world problems. Although modified Kalman-like solutions that account for general nonlinear and non-Gaussian settings have been proposed, including the extended KF (EKF) [2] and the unscented KF (UKF) [3], such techniques are based on simplifications of the system dynamics and suffer from severe degradation when the true dynamic system departs from the linear and Gaussian assumptions.

Since general analytical solutions are intractable, Bayesian estimation in nonlinear, non-Gaussian systems must be addressed using numerical techniques. Deterministic approaches, such as classical numerical integration procedures, turn out ineffective or too demanding except for very low-dimensional systems and, as a consequence, methods based on the Monte Carlo methodology have progressively gained momentum. Monte Carlo methods are simulation-based techniques aimed at estimating the a posteriori pdf of the state signal given the available observations. The pdf estimate consists of a random grid of weighted points in the state space \mathbb{R}^{L_x} . These points, usually termed *particles*, are Monte Carlo samples of the system state that are assigned nonnegative *weights*, which can be interpreted as probabilities of the particles.

The collection of particles and their weights yield an empirical measure which approximates the continuous a posteriori pdf of the system state [4]. The recursive update of this measure whenever a new observation is available is known as *particle filtering* (PF). Although there are other popular Monte Carlo methods based on the idea of producing empirical measures with random support, for example, Markov Chain Monte Carlo (MCMC) techniques [5], PF algorithms have recently received a lot of attention because they are particularly suitable for real-time estimation. The sequential importance sampling (SIS) algorithm [6, 7] and the bootstrap filter (BF) [8, 9] are the most successful members of the PF class of methods [10]. Existing PF techniques rely on

- (i) the knowledge of the probabilistic model of the dynamic system (1)-(2), which includes the densities $p(\mathbf{x}_0)$, $p(\mathbf{u}_t)$, and $p(\mathbf{v}_t)$,
- (ii) the ability to numerically evaluate the likelihood $p(\mathbf{y}_t | \mathbf{x}_t)$ and to sample from the transition density $p(\mathbf{x}_t | \mathbf{x}_{t-1})$.

Therefore, the practical performance of PF algorithms in real-world problems heavily depends on how accurate the underlying probabilistic model of choice is. Although this may seem irrelevant in engineering problems where it is relatively straightforward to associate the observed signals with realistic and convenient probability distributions, in many situations, this is not the case. Many times, it is very hard to find an adequate model using the information available in practice. In other occasions, the working models obtained after a lot of effort (involving, e.g., time-series analysis techniques) are so complicated that they render any subsequent signal processing algorithm impractical due to its high complexity.

In this paper, we introduce a new PF approach to deal with uncertainty in the probabilistic modeling of the dynamic system (1)-(2). We start with the requirement that the ultimate objective of PF is to yield an estimate of the signals of interest $\mathbf{x}_{0:t}$, given the observations $\mathbf{y}_{1:t}$. If a suitable probabilistic model is at hand, good signal estimates can be computed from the filtering pdf (3) induced by the model, and a PF algorithm can be employed to recursively build up a random grid that approximates the posterior distribution. However, it is often possible to use signal estimation methods that do not explicitly rely on the a posteriori pdf, for example, blind detection in digital communications can be performed according to several criteria, such as the constrained minimization of the received signal power [11] or the constant modulus method [12, 13]. Such approaches are very popular because they are based on a simple figure of merit, and this simplicity leads to robust and easy-to-design algorithms.

The same advantages in robustness and easy design can be gained in PF whenever a complex (yet possibly not strongly tied to physical reality) probabilistic model can be substituted by a simpler reference for signal estimation. Contrary to initial intuition, estimation techniques based on ad hoc, heuristic, or, simply, alternative references, different from the state posterior distribution, are not precluded by the use of the PF methodology. It is important to realize that the procedure for sequential build-up of the random grid is not tied to the concept of a posteriori pdf. We will show that, by simply specifying a stochastic mechanism for generating particles, the PF methodology can be successfully used to build an approximation of any function of the system state that admits a recursive decomposition. Specifically, we propose a new family of PF algorithms in which the statistical reference of the a posteriori state pdf is substituted by a user-defined cost function that measures the quality of the state signal estimates according to the available observations. Hence, methods within the new class are termed cost-reference particle filters (CRPFs), in contrast to conventional statistical-reference particle filters (SRPFs).

As long as a recursive decomposition of the cost function is found, a PF algorithm, similar to the SIS and bootstrap methods, can be used to construct a random-grid approximation of the cost function in the vicinity of its minima. For this reason, CRPFs yield *local* representations of the cost function specifically built to facilitate the computation of minimum-cost estimates of the state signal.

The remainder of this paper is organized as follows. The fundamentals of the CRPF family are introduced in Section 2. This includes three basic building blocks: the *cost* and *risk* functions, which provide a measure of the quality of the particles, and the stochastic mechanism for particle generation and sequential update of the random grid. Since the usual tools for PF algorithm design (e.g., proposal distributions, auxiliary variables, etc.) do not necessarily extend to the new framework, this section also contains a discussion on design issues. In particular, we identify the factors on which the choice of the cost and risk functions will usually depend, and derive consequent design guidelines, including a useful choice of parameters that leads to a simple interpretation of the algorithm and its connection with the theory of stochastic approximation (SA) [14].

Due to the change in the reference, convergence results regarding SRPFs are not valid for CRPFs. Section 3 is devoted to the problem of identifying sufficient conditions for *asymptotically optimal propagation* (AOP) of particles. The stochastic procedure for drawing new samples of the state signal and propagating the existing particles using the new samples is the key for the convergence of the algorithm. We term this particle propagation step as asymptotically optimal when the increment in the average cost of the particles in the filter after propagation is minimal. A set of sufficient conditions for optimal propagation, related to the properties of the sampling and weighting methods, is provided.

Section 4 is devoted to the discussion of *resampling* in the new family of PF techniques. We argue that the objective of this important algorithmic step is different from its usual role in conventional PF algorithms, and exploit this difference to propose a *local resampling* scheme suitable for a straightforward implementation using parallel VLSI hardware (note that resampling is a major bottleneck for the parallel implementation of PF methods [15]).

Computer simulation results that illustrate the validity of our approach are presented in Section 5. In particular, we tackle the problem of positioning a vehicle that moves along a 2-dimensional space. An instance of the proposed CRPF class of methods that employs a simple cost function is compared with the standard auxiliary BF [9] technique. Finally, Section 6 contains conclusions.

2. COST-REFERENCE PARTICLE FILTERING

The basics of the new family of PF methods are introduced in this section. We start with a general description of the CRPF technique, where key concepts, namely, the cost and risk functions, particle propagation, and particle selection, are introduced. The second part of the section is devoted to practical design issues. We suggest guidelines for the design

of CRPFs and propose a simple choice of the algorithm parameters that lead to a straightforward interpretation of the CRPF technique.

2.1. Sequential algorithm

The ultimate aim of the method is the online estimation of the sequence of system states from the available observations, that is, we intend to estimate $\mathbf{x}_t | \mathbf{y}_{1:t}$, $t = 0, 1, 2, \dots$, according to some reference function that yields a quantitative measure of quality. In particular, we propose the use of a real *cost* function with a recursive additive structure, that is,

$$\mathcal{C}(\mathbf{x}_{0:t} | \mathbf{y}_{1:t}, \lambda) = \lambda \mathcal{C}(\mathbf{x}_{0:t-1} | \mathbf{y}_{1:t-1}, \lambda) + \Delta \mathcal{C}(\mathbf{x}_t | \mathbf{y}_t), \quad (5)$$

where $0 < \lambda < 1$ is a forgetting factor, $\Delta \mathcal{C} : \mathbb{R}^{L_x} \times \mathbb{R}^{L_y} \rightarrow \mathbb{R}$ is the *incremental cost* function, and $\mathcal{C}(\mathbf{x}_{0:t} | \mathbf{y}_{1:t}, \lambda)$ complies with the definition

$$\mathcal{C} : \mathbb{R}^{(t+1)L_x} \times \mathbb{R}^{tL_y} \times \mathbb{R} \rightarrow \mathbb{R}. \quad (6)$$

We should remark that (5) is not the only recursive decomposition that can be employed. A straightforward alternative is to choose a cost function which is built at time t as the convex sum

$$\mathcal{C}(\mathbf{x}_{0:t} | \mathbf{y}_{1:t}, \lambda) = \lambda \mathcal{C}(\mathbf{x}_{0:t-1} | \mathbf{y}_{1:t-1}, \lambda) + (1 - \lambda) \Delta \mathcal{C}(\mathbf{x}_t | \mathbf{y}_t). \quad (7)$$

This form of cost function is perfectly valid for the definition and construction of CRPFs and choosing it would not affect (or would affect trivially) the arguments presented in the rest of this paper, including the asymptotic convergence results in Section 3. However, we will constrain ourselves to the familiar form of (5) for simplicity.

A high value of $\mathcal{C}(\mathbf{x}_{0:t} | \mathbf{y}_{1:t}, \lambda)$ means that the state sequence $\mathbf{x}_{0:t}$ is not a good estimate given the sequence of observations $\mathbf{y}_{1:t}$, while a low value of $\mathcal{C}(\mathbf{x}_{0:t} | \mathbf{y}_{1:t}, \lambda)$ indicates that $\mathbf{x}_{0:t}$ is close to the true state signal. The sequence $\mathbf{x}_{0:t}$ is said to have a high cost, in the former case, or a low cost, in the latter case. Particularly notice the recursive structure in (5), where the cost of a sequence up to time $t - 1$ can be updated by solely looking at the state and observation vectors at time t , \mathbf{x}_t , and \mathbf{y}_t , respectively, which are used to compute the cost increment $\Delta \mathcal{C}(\mathbf{x}_t | \mathbf{y}_t)$. The forgetting factor λ avoids attributing an excessive weight to old observations when a long series of data are collected, hence allowing for potential adaptivity.

We also introduce a one-step *risk* function of the form

$$\begin{aligned} \mathcal{R} : \mathbb{R}^{L_x} \times \mathbb{R}^{L_y} &\rightarrow \mathbb{R}, \\ \mathbf{x}_{t-1}, \mathbf{y}_t &\rightsquigarrow \mathcal{R}(\mathbf{x}_{t-1} | \mathbf{y}_t) \end{aligned} \quad (8)$$

that measures the adequacy of the state at time $t - 1$ given the new observation \mathbf{y}_t . It is convenient to view the risk function $\mathcal{R}(\mathbf{x}_{t-1} | \mathbf{y}_t)$ as a prediction of the cost increment $\Delta \mathcal{C}(\mathbf{x}_t | \mathbf{y}_t)$ that can be obtained before \mathbf{x}_t is actually propagated. Hence, a natural choice of the risk function is

$$\mathcal{R}(\mathbf{x}_{t-1} | \mathbf{y}_t) = \Delta \mathcal{C}(f_x(\mathbf{x}_{t-1}) | \mathbf{y}_t). \quad (9)$$

The proposed estimation technique proceeds sequentially in a similar manner as the BF. Given a set of M state samples and associated costs up to time t , that is, the weighted-particle set (wps)

$$\Xi_t = \{\mathbf{x}_t^{(i)}, \mathcal{C}_t^{(i)}\}_{i=1}^M, \quad (10)$$

where $\mathcal{C}_t^{(i)} = \mathcal{C}(\mathbf{x}_{0:t}^{(i)} | \mathbf{y}_{1:t}, \lambda)$, the grid of state trajectories is randomly propagated when \mathbf{y}_{t+1} is observed in order to build an updated wps Ξ_{t+1} . The state and observation signals are those described in the dynamic system (1)-(2). We only add the following mild assumptions:

- (1) the initial state is known to lie in a bounded interval $I_{\mathbf{x}_0} \subset \mathbb{R}^{L_x}$;
- (2) the system and observation noise are both zero mean.

Assumption (1) is needed to ensure that we initially draw a set of samples that is not infinitely far from the true state \mathbf{x}_0 . Notice that this is a structural assumption, not a probabilistic one. Assumption (2) is made for the sake of simplicity because zero-mean noise is the rule in most systems.

The sequential CRPF algorithm based on the structure of system (1)-(2), the definitions of cost and risk functions given by (5) and (8), respectively, and assumptions (1) and (2), is described below.

(1) *Time $t = 0$ (initialization)*. Draw M particles from the uniform distribution in the interval $I_{\mathbf{x}_0}$,

$$\mathbf{x}_0^{(i)} \sim \mathcal{U}(I_{\mathbf{x}_0}), \quad (11)$$

and assign them a zero cost. The initial wps

$$\Xi_0 = \{\mathbf{x}_0^{(i)}, \mathcal{C}_0^{(i)} = 0\}_{i=1}^M \quad (12)$$

is obtained.

(2) *Time $t + 1$ (selection of the most promising trajectories)*. The goal of the selection step is to replicate those particles with a low cost while high-cost particles are discarded. As usual in PF methods, selection is implemented by a resampling procedure [7]. We point out that, differently from the standard BF, resampling in CRPFs does not produce equally weighted particles. Instead, each particle preserves its own cost. Notice that the equal weighting of resampled particles in standard PF algorithms comes from the use of a statistical reference. In CRPF, preserving the particle costs after resampling actually shifts the random grid representation of the cost function toward its local minima. Such a behavior is sound, as we are interested in *minimum cost* signal estimates. Further issues related to resampling are discussed in Section 4.

For $i = 1, 2, \dots, M$, compute the one-step risk of particle i and let

$$\mathcal{R}_{t+1}^{(i)} = \lambda \mathcal{C}_t^{(i)} + \mathcal{R}(\mathbf{x}_t^{(i)} | \mathbf{y}_{t+1}) \quad (13)$$

which yields a predictive cost of the trajectory $\mathbf{x}_{0:t}$ according to the new observation \mathbf{y}_t . Define a probability mass function (pmf) of the form

$$\hat{\pi}_{t+1}^{(i)} \propto \mu(\mathcal{R}_{t+1}^{(i)}), \quad (14)$$

where $\mu : \mathbb{R} \rightarrow [0, +\infty)$ is a monotonically decreasing function. An intermediate wps is obtained by resampling the trajectories $\{\mathbf{x}_t^{(i)}\}_{i=1}^M$ according to the pmf $\hat{\pi}_{t+1}^{(i)}$. Specifically, we select $\hat{\mathbf{x}}_t^{(i)} = \mathbf{x}_t^{(k)}$ with probability $\hat{\pi}_{t+1}^{(k)}$, and build the set $\hat{\Xi}_{t+1} = \{\hat{\mathbf{x}}_t^{(i)}, \hat{\mathcal{C}}_t^{(i)}\}_{i=1}^M$, where $\hat{\mathcal{C}}_t^{(i)} = \mathcal{C}_t^{(k)}$ if and only if $\hat{\mathbf{x}}_t^{(i)} = \mathbf{x}_t^{(k)}$.

(3) *Time $t + 1$ (random particle propagation)*. Select an arbitrary conditional pdf of the state $p_{t+1}(\mathbf{x}_{t+1} | \mathbf{x}_t)$ with the constraint that

$$E_{p_{t+1}(\mathbf{x}_{t+1} | \mathbf{x}_t)}[\mathbf{x}_{t+1}] = f_x(\mathbf{x}_t), \quad (15)$$

where $E_{p(s)}[\cdot]$ denotes expected value with respect to the pdf in the subindex. Using the selected propagation density, draw new particles

$$\mathbf{x}_{t+1}^{(i)} \sim p_{t+1}(\mathbf{x}_{t+1} | \hat{\mathbf{x}}_t^{(i)}) \quad (16)$$

and update the associated costs

$$\mathcal{C}_{t+1}^{(i)} = \lambda \hat{\mathcal{C}}_t^{(i)} + \Delta \mathcal{C}_{t+1}^{(i)}, \quad (17)$$

where

$$\Delta \mathcal{C}_{t+1}^{(i)} = \Delta \mathcal{C}(\mathbf{x}_{t+1}^{(i)} | \mathbf{y}_{t+1}) \quad (18)$$

for $i = 1, 2, \dots, M$.

As a result, the updated wps $\Xi_{t+1} = \{\mathbf{x}_{t+1}^{(i)}, \mathcal{C}_{t+1}^{(i)}\}_{i=1}^M$ is obtained.

(4) *Time $t + 1$ (estimation of the state)*. Estimation procedures are better understood if a pmf $\pi_{t+1}^{(i)}$, $i = 1, 2, \dots, M$, is assigned to the particles in Ξ_{t+1} . The natural way to define this pmf is according to the particle costs, that is,

$$\pi_{t+1}^{(i)} \propto \mu(\mathcal{C}_{t+1}^{(i)}), \quad (19)$$

where μ is a monotonically decreasing function.

The minimum cost estimate at time $t + 1$ is trivially computed as

$$i_0 = \arg \max_i \{\pi_{t+1}^{(i)}\}, \quad (20)$$

$$\hat{\mathbf{x}}_{0:t+1}^{\min} = \mathbf{x}_{t+1}^{(i_0)}$$

and its physical meaning is obvious. An equally useful estimate can be computed as the mean value of $\mathbf{x}_{t+1}^{(i)}$ according to the pmf $\pi_{t+1}^{(i)}$, that is,

$$\bar{\mathbf{x}}_{t+1}^{\text{mean}} = \sum_{i=1}^M \pi_{t+1}^{(i)} \mathbf{x}_{t+1}^{(i)}. \quad (21)$$

Note that $\bar{\mathbf{x}}_{t+1}^{\text{mean}}$ can also be regarded as a minimum cost estimate because the particle set Ξ_{t+1} is a random-grid local representation of the cost function in the vicinity of its minima. In fact, estimator (21) has slight advantages over (20). Namely, the averaging of particles according to the pmf $\pi_{t+1}^{(i)}$ yields an estimated state trajectory which is smoother than the one resulting from simply choosing the particle with the least cost at each time step. Besides, computing the mean of the particles under $\pi_{t+1}^{(i)}$ may result in an estimate with a slightly smaller cost than the least cost particle, since $\bar{\mathbf{x}}_{t+1}^{\text{mean}}$ is obtained by interpolation of particles around the least cost state.

Sufficient conditions for the mean estimate (21) to attain an asymptotically minimum cost are given in Section 3.

We will refer to the general procedure described above as a CRPF algorithm. It is apparent that many implementations are possible for a single problem, so in the next section, we discuss the choice of the functions and parameters involved in the method.

2.2. Design issues

An instance of the CRPF class of algorithms is selected by choosing

- (i) the cost function $\mathcal{C}(\cdot|\cdot)$,
- (ii) the risk function $\mathcal{R}(\cdot|\cdot)$,
- (iii) the monotonically decreasing function $\mu : \mathbb{R} \rightarrow [0, +\infty)$ that maps costs and risks into the resampling and estimation pmfs, as indicated in (14) and (19), respectively,
- (iv) the sequence of pdfs $p_{t+1}(\mathbf{x}_{t+1}|\mathbf{x}_t)$ for particle generation.

The cost and risk functions measure the quality of the particles in the filter. Recall that the risk is conveniently interpreted as a prediction of the cost of a particle, given a new observation, before random propagation is actually carried out (see the selection step in Section 2.1). Therefore, the cost and the risk should be closely related, and we suggest to choose $\mathcal{R}(\cdot|\cdot)$ according to (9). Whenever possible, both the cost and risk functions should be

- (i) strictly convex in the range of values of \mathbf{x}_t , where the state is expected to lie, in order to avoid ambiguities in the estimators (20) and (21) as well as in the selection (resampling) step,
- (ii) easy to compute in order to facilitate the practical implementation of the algorithm,
- (iii) dependent on the complete state and observation signals, that is, it should involve all the elements of \mathbf{x}_t and \mathbf{y}_t .

A simple, yet useful and physically meaningful, choice of $\mathcal{C}(\cdot|\cdot)$, $\mathcal{R}(\cdot|\cdot)$ that will be used in the numerical examples of Section 5 is given by

$$\mathcal{C}(\mathbf{x}_0) = 0, \quad (22)$$

$$\Delta \mathcal{C}(\mathbf{x}_t | \mathbf{y}_t) = \|\mathbf{y}_t - f_y(\mathbf{x}_t)\|^q, \quad (23)$$

$$\mathcal{R}(\mathbf{x}_t | \mathbf{y}_{t+1}) = \|\mathbf{y}_{t+1} - f_y(f_x(\mathbf{x}_t))\|^q, \quad (24)$$

where $q \geq 1$ and $\|\mathbf{v}\| = \sqrt{\mathbf{v}^T \mathbf{v}}$ denotes the norm of \mathbf{v} . Given a fixed and bounded sequence of observations $\mathbf{y}_{1:t}$, the optimal (minimum cost) sequence of state vectors is

$$\begin{aligned} \mathbf{x}_{0:t}^{\text{opt}} &= \arg \min_{\mathbf{x}_{0:t}} \{ \mathcal{C}(\mathbf{x}_{0:t} | \mathbf{y}_{1:t}, \lambda) \} \\ &= \arg \min_{\mathbf{x}_{0:t}} \left\{ \sum_{k=0}^t \lambda^{(t-k)} \Delta \mathcal{C}(\mathbf{x}_k | \mathbf{y}_k) \right\}. \end{aligned} \quad (25)$$

We call $\mathbf{x}_{0:t}^{\text{opt}}$ optimal because it is obtained by minimization of the continuous cost function, and it is in general different from the minimum cost estimate obtained by CRPF, which we have denoted as $\bar{\mathbf{x}}_{0:t}^{\text{min}}$ in Section 2.1.

With the assumed choice of cost and risk functions given by (22)–(24), the invertible observation function $f_y : \mathbb{R}^{L_x} \rightarrow I_y \subseteq \mathbb{R}^{L_y}$, and $\mathbf{y}_t \in I_y$, for all $t \geq 1$, it is straightforward to derive a pointwise solution of the form¹

$$\mathbf{x}_t^{\text{opt}} = \arg \min_{\mathbf{x}_t} \{ \Delta \mathcal{C}(\mathbf{x}_t | \mathbf{y}_t) \} = f_y^{-1}(\mathbf{y}_t). \quad (26)$$

Therefore, as the CRPF algorithm randomly selects and propagates the sample states with the least cost, it can be understood (again, under assumption of (22)–(24)) as a numerical stochastic method for approximately solving the set of (possibly nonlinear) equations

$$\mathbf{y}_t - f_y(\mathbf{x}_t) = \mathbf{0}, \quad t = 1, 2, \dots \quad (27)$$

Furthermore, setting $q = 1$ in (23) and (24), we obtain a Monte Carlo estimate of the mean absolute deviation solution of the above set of equations, while $q = 2$ results in a stochastic optimization of the least squares type.

This interpretation of the CRPF algorithm as a method for numerically solving (27) allows to establish a connection between the proposed methodology and the theory of SA [14], which is briefly commented upon in Appendix A.

The function $\mu : \mathbb{R} \rightarrow [0, +\infty)$ should be selected to guarantee an adequate discrimination of low-cost particles from those presenting higher costs (recall that we are interested in computing a local representation of the cost function in the vicinity of its minima). As shown in Section 5, the choice of μ has a direct impact on the algorithm performance. Specifically, notice that the accuracy of the selection step is highly dependent on the ability of μ to assign large probability masses to lower-cost particles.

¹Note, however, that additional solutions may exist at $\nabla_{\mathbf{x}} f_y(\mathbf{x}) = \mathbf{0}$ depending on $f_y(\cdot)$.

A straightforward choice of this function is

$$\mu_1(\mathcal{C}_t^{(i)}) = \mathcal{C}_t^{(i)-1}, \quad \mathcal{C}_t^{(i)} \in \mathbb{R} \setminus \{0\}, \quad (28)$$

which is simple to compute and potentially useful in many systems. It has a serious drawback, however, in situations where the range of the costs, that is, $\max_i \{\mathcal{C}_t^{(i)}\} - \min_i \{\mathcal{C}_t^{(i)}\}$, is much smaller than the average cost $(1/M) \sum_{i=1}^M \mathcal{C}_t^{(i)}$. In such scenarios, μ_1 yields nearly uniform probability masses and the algorithm performance degrades. Better discrimination properties can be achieved with an adequate modification of μ_1 , for example, with

$$\mu_2(\mathcal{C}_t^{(i)}) = \frac{1}{(\mathcal{C}_t^{(i)} - \min_k \{\mathcal{C}_t^{(k)}\} + \delta)^\beta}, \quad (29)$$

where $0 < \delta < 1$ and $\beta > 1$. When compared with μ_1 , μ_2 assigns larger masses to low-cost particles and much smaller masses to higher-cost samples. The discrimination ability of μ_2 is enhanced by reducing the value of δ (i.e., $\delta \simeq 0$) and/or increasing β . The relative merit of μ_2 over μ_1 is experimentally illustrated in Section 5.

The last selection to be made is the pdf for particle propagation, $p_{t+1}(\mathbf{x}_{t+1}|\mathbf{x}_t)$, in step 3 of the CRPF algorithm. The theoretical properties required for *optimal* propagation are explored in Section 3. From a practical and intuitive² point of view, it is desirable to use easy-to-sample pdfs with a large enough variance to avoid losing tracks of the state signal, but not too large, to prevent the generation of too dispersed particles. A simple strategy implemented in the simulations of Section 5 consists of using zero-mean Gaussian densities with adaptively selected variance. Specifically, the particle i is propagated from time t to time $t+1$ as

$$\mathbf{x}_{t+1}^{(i)} \sim \mathcal{N}(f_x(\hat{\mathbf{x}}_{t-1}^{(i)}), \sigma_t^{2,(i)} \mathbf{I}_{L_x}), \quad (30)$$

where \mathbf{I}_{L_x} is the $L_x \times L_x$ identity function and the variance $\sigma_t^{2,(i)}$ is recursively computed as

$$\sigma_t^{2,(i)} = \frac{t-1}{t} \sigma_{t-1}^{2,(i)} + \frac{\|\mathbf{x}_t^{(i)} - f_x(\hat{\mathbf{x}}_{t-1}^{(i)})\|^2}{tL_x}. \quad (31)$$

This adaptive-variance technique has appeared useful and efficient in our simulations, as illustrated in Section 5, but alternative approaches (including the simple choice of a fixed variance) can also be successfully exploited.

3. CONVERGENCE OF CRPF ALGORITHMS

In this section, we assess the convergence of the proposed CRPF algorithm. In particular, we seek sufficient conditions

for AOP of the particles from time $t-1$ to time t . Let $\Xi_t = \{\mathbf{x}_t^{(i)}, \mathcal{C}_t^{(i)}\}_{i=1}^M$ be the wps computed at time t . We say that Ξ_t has been obtained by AOP from Ξ_{t-1} if and only if

$$\lim_{M \rightarrow \infty} |\Delta \mathcal{C}(\mathbf{x}_t^{\text{opt}} | \mathbf{y}_t) - \overline{\Delta \mathcal{C}}_t| = 0 \quad (\text{in some sense}), \quad (32)$$

where $\mathbf{x}_t^{\text{opt}}$ is the optimal state according to (26) and

$$\overline{\Delta \mathcal{C}}_t = \sum_{i=1}^M \hat{\omega}_t^{(i)} \Delta \mathcal{C}_t^{(i)}, \quad (33)$$

with a pmf $\hat{\omega}_t^{(i)} \propto \mu(\Delta \mathcal{C}_t^{(i)})$, is the mean incremental cost at time t . The results presented in this section prove that AOP can be ensured by adequately choosing the propagation density and function $\mu : \mathbb{R} \rightarrow [0, \infty)$ that relates the cost to the pmf's $\hat{\pi}_t^{(i)}$ and $\pi_t^{(i)}$. Notice that $\pi_t^{(i)} = \hat{\omega}_t^{(i)}$ when $\lambda = 0$.

A corollary of the AOP convergence theorem is also established that provides sufficient conditions for the mean state estimate given by (21), for the case $\lambda = 0$, to be asymptotically optimal in terms of its incremental cost.

3.1. Preliminary definitions

Some preliminary definitions are necessary before stating and proving sufficient AOP conditions. If the selection and propagation steps of the CRPF method are considered jointly, it turns out that, at time t , M particles are sampled as

$$\mathbf{x}_t^{(i)} \sim p_t^{M'}(\mathbf{x}), \quad (34)$$

where $M' < \infty$ denotes the number of particles available at time $t-1$ and sampling the pdf $p_t^{M'}(\mathbf{x})$ amounts to resampling M times in $\Xi_{t-1} = \{\mathbf{x}_{t-1}^{(i)}, \mathcal{C}_{t-1}^{(i)}\}_{i=1}^{M'}$ and then propagating the resulting particles and updating the costs to build the new wps $\Xi_t = \{\mathbf{x}_t^{(i)}, \mathcal{C}_t^{(i)}\}_{i=1}^M$ (note that we explicitly allow $M \neq M'$). Although other possibilities exist, for example, as described in Section 4, when multinomial resampling is used in the selection step of the CRPF algorithm, the pdf in (34) is a finite mixture of the form

$$p_t^{M'}(\mathbf{x}) = \sum_{k=1}^{M'} \hat{\pi}_t^{(k)} p_t(\mathbf{x} | \mathbf{x}_{t-1}^{(k)}). \quad (35)$$

We also introduce the following notation for a ball centered at $\mathbf{x}_t^{\text{opt}}$ with radius $\varepsilon > 0$:

$$S\{\mathbf{x}_t^{\text{opt}}, \varepsilon\} = \{\mathbf{x} \in \mathbb{R}^{L_x} : \|\mathbf{x} - \mathbf{x}_t^{\text{opt}}\| < \varepsilon\}, \quad (36)$$

and we write

$$S^M\{\mathbf{x}_t^{\text{opt}}, \varepsilon\} = \{\mathbf{x} \in \{\mathbf{x}_t^{(i)}\}_{i=1}^M : \|\mathbf{x} - \mathbf{x}_t^{\text{opt}}\| < \varepsilon\} \quad (37)$$

for its discrete counterpart built from the particles in Ξ_t .

²Part of this intuition is confirmed by the convergence theorem in Section 3.

3.2. Convergence theorem

Lemma 1. Let $\{\mathbf{x}_t^{(i)}\}_{i=1}^M$ be a set of particles drawn at time t using the propagation pdf $p_t^{M'}(\mathbf{x})$ as defined by (34), let $\mathbf{y}_{1:t}$ be a fixed bounded sequence of observations, and let $\Delta\mathcal{C}(\mathbf{x}|\mathbf{y}_t) \geq 0$ be a continuous cost function, bounded in $S\{\mathbf{x}_t^{\text{opt}}, \varepsilon\}$, with a minimum at $\mathbf{x} = \mathbf{x}_t^{\text{opt}}$.

If the three following conditions are met:

- (1) any ball with center at $\mathbf{x}_t^{\text{opt}}$ has a nonzero probability under the propagation density, that is,

$$\int_{S\{\mathbf{x}_t^{\text{opt}}, \varepsilon\}} p_t^{M'}(\mathbf{x}) d\mathbf{x} = \gamma > 0 \quad \forall \varepsilon > 0, \quad (38)$$

- (2) the supremum of the function $\mu(\Delta\mathcal{C}(\cdot|\cdot))$ for points outside $S(\mathbf{x}_t^{\text{opt}}, \varepsilon)$ is a finite constant, that is,

$$\mathbf{S}_{\text{out}} = \sup_{\mathbf{x}_t \in \mathbb{R}^{L_x} \setminus S(\mathbf{x}_t^{\text{opt}}, \varepsilon)} \{\mu(\Delta\mathcal{C}(\mathbf{x}_t|\mathbf{y}_t))\} < \infty, \quad (39)$$

- (3) the supremum of the function $\mu(\Delta\mathcal{C}(\cdot|\cdot))$ for points inside $S^M(\mathbf{x}_t^{\text{opt}}, \varepsilon)$ converges to infinity faster than the identity function, that is,

$$\lim_{M \rightarrow \infty} \frac{M}{\mathbf{S}_{\text{in}}} = 0, \quad (40)$$

where

$$\mathbf{S}_{\text{in}} = \sup_{\mathbf{x}_t \in S^M(\mathbf{x}_t^{\text{opt}}, \varepsilon)} \{\mu(\Delta\mathcal{C}(\mathbf{x}_t|\mathbf{y}_t))\}, \quad (41)$$

then the set function $\mu_t : A \subseteq \{\mathbf{x}_t^{(i)}\}_{i=1}^M \rightarrow [0, \infty)$ defined as

$$\mu_t(A \subseteq \{\mathbf{x}_t^{(i)}\}_{i=1}^M) = \sum_{\mathbf{x} \in A} \mu(\Delta\mathcal{C}(\mathbf{x}|\mathbf{y}_t)) \quad (42)$$

is an infinite discrete measure (see definition in, e.g., [16]) that satisfies

$$\lim_{M \rightarrow \infty} \Pr \left[1 - \frac{\mu_t(S^M(\mathbf{x}_t^{\text{opt}}, \varepsilon))}{\mu_t(\{\mathbf{x}_t^{(i)}\}_{i=1}^M)} \geq \delta \right] = 0 \quad \forall \delta > 0, \quad (43)$$

where $\Pr[\cdot]$ denotes probability, that is,

$$\lim_{M \rightarrow \infty} \frac{\mu_t(S^M(\mathbf{x}_t^{\text{opt}}, \varepsilon))}{\mu_t(\{\mathbf{x}_t^{(i)}\}_{i=1}^M)} = 1 \quad (i.p.), \quad (44)$$

where *i.p.* stands for “in probability.”

See Appendix B for a proof.

Theorem 1. If conditions (38), (39), and (40) in Lemma 1 hold true, then the mean incremental cost at time t ,

$$\overline{\Delta\mathcal{C}_t} = \sum_{i=1}^M \omega_t^{(i)} \Delta\mathcal{C}(\mathbf{x}_t^{(i)}|\mathbf{y}_t), \quad (45)$$

converges to the minimal incremental cost as $M \rightarrow \infty$,

$$\lim_{M \rightarrow \infty} |\Delta\mathcal{C}(\mathbf{x}_t^{\text{opt}}|\mathbf{y}_t) - \overline{\Delta\mathcal{C}_t}| = 0 \quad (i.p.). \quad (46)$$

See Appendix C for a proof.

Finally, an interesting corollary that justifies the use of the mean estimate (21) can be easily derived from Lemma 1 and Theorem 1.

Corollary 1. Assuming (38), (39), and (40) in Lemma 1, and forgetting factor $\lambda = 0$, the mean cost estimate is asymptotically optimal, that is,

$$\lim_{M \rightarrow \infty} |\Delta\mathcal{C}(\bar{\mathbf{x}}_t^{\text{mean}}|\mathbf{y}_t) - \Delta\mathcal{C}_t(\mathbf{x}_t^{\text{opt}}|\mathbf{y}_t)| = 0 \quad (i.p.), \quad (47)$$

where

$$\bar{\mathbf{x}}_t^{\text{mean}} = \sum_{i=1}^M \pi_t^{(i)} \mathbf{x}_t^{(i)}. \quad (48)$$

See Appendix D for a proof.

3.3. Discussion

Theorem 1 states that conditions (38)–(40) are sufficient to achieve AOP (*i.p.*). The validity of this result clearly depends on the existence of a propagation pdf, $p_t^{M'}[\cdot]$, and a measure μ_t with *good* properties in order to meet the required conditions.

It is impossible to guarantee that condition (38) holds true in general, as the value of $\mathbf{x}_t^{\text{opt}}$ is a priori unknown, but if the number of particles is large enough and they are evenly distributed on the state space, it is reasonable to expect that the region around $\mathbf{x}_t^{\text{opt}}$ has a nonzero probability. Intuitively, if the wps is locked to the system state at time $t - 1$, using the system dynamics to propagate the particles to time t should keep the filtering algorithm locked to the state trajectory. Indeed, our computer simulation experiments give evidence that the propagation pdf is not a critical weakness, and the proposed sequence of Gaussian densities given by (30) and (31) yields a remarkably good performance.

Conditions (39) and (40) are related to the choice of μ or, equivalently, the measure μ_t . For the proposed cost model given by (22) and (23), it is simple to show that condition (39) holds true, both for $\mu = \mu_1$ and $\mu = \mu_2$, as defined in (28) and (29), respectively. The analysis of condition (40) is more demanding and will not be addressed here. An educated intuition, also supported by the computer simulation results in Section 5, points in the direction of selecting $\mu = \mu_2$ with a small enough value of δ .

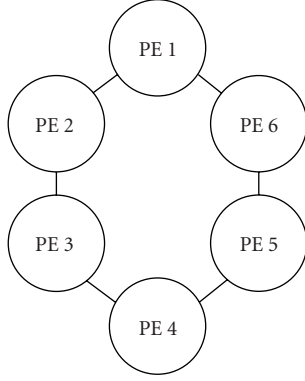


FIGURE 1: $M = 6$ processors in a ring configuration for parallel implementation of the local resampling algorithm.

4. RESAMPLING AND PARALLEL IMPLEMENTATION

Resampling is an indispensable algorithmic component in sequential methods for statistical reference PF, which, otherwise, suffer from weight degeneracy and do not converge to useful solutions [4, 7, 15]. However, resampling also becomes a major obstacle for efficient implementation of PF algorithms in parallel VLSI hardware devices because it creates full data dependencies among processing units [15]. Although some promising methods have been recently proposed [15, 17], parallelization of resampling algorithms remains an open problem.

The selection step in CRPFs (see Section 2.1) is much less restrictive than resampling in conventional SRPFs. Specifically, while resampling methods in SRPFs must ensure that the probability distribution of the resampled population is an unbiased and unweighted approximation of the original distribution of the particles [4], selection in CRPFs is only aimed at ensuring that the particles are close to the locations that produce cost function minima. We have found evidence of state estimates obtained by CRPF being better when the random grid of particles comprises small regions of the state space around these minima. Therefore, selection algorithms can be devised with the only and mild constraint that they do not increase the average cost of particles.

Now we briefly describe a simple resampling technique for CRPFs that lends itself to a straightforward parallelization. Figure 1 shows an array of independent processors connected in a ring configuration. We assume, for simplicity, that the number of processors is equal to the number of particles M , although the algorithm is easily generalized to a smaller number of processing elements (PEs). The i th PE (PE _{i}) contains the triple $\{\mathbf{x}_t^{(i)}, \mathcal{C}_t^{(i)}, \mathcal{R}_{t+1}^{(i)}\}$ in its memory. The proposed *local resampling* technique proceeds in two steps.

- (i) PE _{i} transmits $\{\mathbf{x}_t^{(i)}, \mathcal{C}_t^{(i)}, \mathcal{R}_{t+1}^{(i)}\}$ to PE _{$i+1$} and PE _{$i-1$} and receives the corresponding information from its neighbors. This communication step can be typically carried out in a single cycle and, when complete, PE _{i} contains three particles $\{\mathbf{x}_t^{(k)}, \mathcal{C}_t^{(k)}, \mathcal{R}_{t+1}^{(k)}\}_{k=i-1}^{i+1}$.

- (ii) Each PE draws a single particle with probabilities according to the risks, that is, for the i th PE:

$$\hat{\mathbf{x}}_t^{(i)} = \mathbf{x}_t^{(k)}, \quad \hat{\mathcal{C}}_t^{(i)} = \mathcal{C}_t^{(k)}, \quad k \in \{i-1, i, i+1\}, \quad (49)$$

with probability $\hat{\pi}_t^{(k)} = \mu(\mathcal{R}_{t+1}^{(k)}) / \sum_{l=i-1}^{i+1} \mu(\mathcal{R}_{t+1}^{(l)})$.

Note that, in two simple steps, the algorithm stochastically selects those particles with smaller risks. It is apparent that the method lends easily to parallelization, with very limited communication requirements. The term *local resampling* comes from the observation that low-risk particles are only locally spread by the method, that is, a PE containing a high-risk particle can only get a low-risk sample from its two neighbors.

5. COMPUTER SIMULATIONS

In this section, we present computer simulations that illustrate the validity of our approach. We have considered the problem of autonomous positioning of a vehicle moving along a 2-dimensional space. The vehicle is assumed to have means to estimate its current speed every T_s seconds and it also measures, with the same frequency, the power of three radio signals emitted from known locations and with known attenuation coefficients. This information can be used by a particle filter to estimate the actual vehicle position.

Following [18], we model the positioning problem by the state-space system

- (i) state equation:

$$\mathbf{x}_t = \mathbf{G}_x \mathbf{x}_{t-1} + \mathbf{G}_v \mathbf{v}_t + \mathbf{G}_u \mathbf{u}_t; \quad (50)$$

- (ii) observation equation:

$$y_{i,t} = 10 \log_{10} \left(\frac{P_{i,0}}{\|\mathbf{r}_i - \mathbf{x}_t\|^{\alpha_i}} \right) + w_{i,t}, \quad (51)$$

where $\mathbf{x}_t \in \mathbb{R}^2$ indicates the position of the vehicle in the 2-dimensional reference set, $\mathbf{G}_x = \mathbf{I}_2$ and $\mathbf{G}_v = \mathbf{G}_u = T_s \mathbf{I}_2$ are known transition matrices, $\mathbf{v}_t \in \mathbb{R}^2$ is the observable vehicle speed, which is assumed constant during the interval $((t-1)T_s, tT_s)$, and \mathbf{u}_t is a noise process that accounts for measurement errors of the speed. The vector $\mathbf{y}_t = [y_{1,t}, y_{2,t}, y_{3,t}]^T$ collects the received power from three emitters located at known reference locations $\mathbf{r}_i \in \mathbb{R}^2$, $i = 1, 2, 3$, that transmit their signals with initial power $P_{i,0}$ through a fading channel with attenuation coefficient α_i , and, finally, $\mathbf{w}_t = [w_{1,t}, w_{2,t}, w_{3,t}]^T$ is the observation noise. Each time step represents T_s seconds, the position vectors \mathbf{x}_t and \mathbf{r}_i have units of meters (m), the speed is given in m/s, and the received power is measured in dB. The initial vehicle position \mathbf{x}_0 is drawn from a standard 2-dimensional Gaussian distribution, that is, $\mathbf{x}_0 \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_2)$.

We have applied the proposed CRPF methodology for solving the positioning problem and, for comparison and benchmarking purposes, we have also implemented the popular auxiliary BF [9], which has an algorithmic structure (resampling, importance sampling, and state estimation) very

Parameters. For all i ,
 $\lambda = 0.95$; $q = 1, 2$; $\delta = 0.01$; $\beta = 2$; $M = 50$; $\sigma_0^{2,(i)} = 10$.

Initialization. For $i = 1, \dots, M$,
 $\mathbf{x}_0^{(i)} \sim \mathcal{U}(-8, +8)$,
 $\mathcal{C}_0^{(i)} = 0$.

Recursive update. For $i = 1, \dots, M$,
 $\mathcal{R}_{t+1}^{(i)} = \lambda \mathcal{C}_t^{(i)} + \|\mathbf{y}_{t+1} - f_y(\mathbf{G}_x \mathbf{x}_t^{(i)} + \mathbf{G}_v \mathbf{v}_{t+1})\|^q$.

Multinomial selection (resampling).

$$\text{pmf} : \hat{\pi}_{t+1}^{(i)} = \begin{cases} \frac{(\mathcal{R}_{t+1}^{(i)})^{-1}}{\sum_{l=1}^M (\mathcal{R}_{t+1}^{(l)})^{-1}} & \text{(function } \mu_1), \\ \frac{(\mathcal{R}_{t+1}^{(i)} - \min_{j \in \{1, \dots, M\}} \mathcal{R}_{t+1}^{(j)} + \delta)^{-\beta}}{\sum_{l=1}^M (\mathcal{R}_{t+1}^{(l)} - \min_{j \in \{1, \dots, M\}} \mathcal{R}_{t+1}^{(j)} + \delta)^{-\beta}} & \text{(function } \mu_2). \end{cases}$$

Selection. $(\hat{\mathbf{x}}_t^{(i)}, \hat{\mathcal{C}}_t^{(i)}) = (\mathbf{x}_t^{(k)}, \mathcal{C}_t^{(k)})$, $k \in \{1, \dots, M\}$, with probability $\hat{\pi}_{t+1}^{(k)}$.

Variance update.
 $t \leq 10$: $\sigma_{t+1}^{2,(i)} = \sigma_t^{2,(i)}$,
 $t > 10$: $\sigma_t^{2,(i)} = \frac{t-1}{t} \sigma_{t-1}^{2,(i)} + \frac{\|\mathbf{x}_t^{(i)} - f_x(\hat{\mathbf{x}}_{t-1}^{(i)})\|^2}{tL_x}$.

Let $\mathbf{x}_{t+1}^{(i)} \sim p_{t+1}(\mathbf{x}_{t+1} | \hat{\mathbf{x}}_t^{(i)})$, where
 $E_{p_{t+1}(\mathbf{x}_{t+1} | \hat{\mathbf{x}}_t^{(i)})}[\mathbf{x}_{t+1}] = f_x(\hat{\mathbf{x}}_t^{(i)})$,
 $\text{Cov}_{p_{t+1}(\mathbf{x}_{t+1} | \hat{\mathbf{x}}_t^{(i)})}[\mathbf{x}_{t+1}] = \sigma_{t+1}^{2,(i)} \mathbf{I}_2$,
 $\mathbf{x}_{0:t+1}^{(i)} = \{\hat{\mathbf{x}}_{0:t}^{(i)}, \mathbf{x}_{t+1}^{(i)}\}$,
 $\mathcal{C}_{t+1}^{(i)} = \lambda \hat{\mathcal{C}}_t^{(i)} + \|\mathbf{y}_{t+1} - f_y(\mathbf{x}_{t+1}^{(i)})\|^q$.

State estimation.
 $\pi_t^{(i)} \propto \mu_1(\mathcal{C}_t^{(i)})$ or $\pi_t^{(i)} \propto \mu_2(\mathcal{C}_t^{(i)})$,
 $\bar{\mathbf{x}}_t^{\text{mean}} = \sum_{i=1}^M \pi_t^{(i)} \mathbf{x}_t^{(i)}$.

ALGORITHM 1: CRPF algorithm with multinomial resampling for the 2-dimensional positioning problem.

similar to the proposed CRPF family. Algorithm 1 summarizes the details of the CRPF algorithm with multinomial selection, including the alternatives in the choice of function μ . The selection step can be substituted by the local resampling procedure shown in Algorithm 2. A pseudocode for the auxiliary BF is also provided in Algorithm 3.

In the following subsections, we describe different computer experiments that were carried out using synthetic data generated according to model (50)–(51). Two types of plots are presented, both for CRPF and BF algorithms. Vehicle trajectories in the 2-dimensional space, resulting from a single simulation of the dynamic system, are shown to illustrate the ability of the algorithms to remain locked to the state trajectory. We chose the mean absolute deviation as a performance figure of merit. It was measured between the true vehicle trajectory in \mathbb{R}^2 and the trajectory estimated by the particle filters and its unit was meter. All mean-deviation plots were obtained by averaging 50 independent simulations. Both the BF and the CRPF type of algorithms were run with $M = 50$ particles.

5.1. Mixture Gaussian noise processes

In the first experiment, we modeled the system and observation noise processes \mathbf{u}_t and \mathbf{w}_t , respectively, as independent

and temporally white, with the mixture Gaussian pdfs:

$$\begin{aligned} \mathbf{u}_t &\sim 0.3\mathcal{N}(\mathbf{0}, \sqrt{0.2}\mathbf{I}_2) \\ &\quad + 0.4\mathcal{N}(\mathbf{0}, \mathbf{I}_2) + 0.3\mathcal{N}(\mathbf{0}, \sqrt{10}\mathbf{I}_2), \\ w_{l,t} &\sim 0.3\mathcal{N}(0, 0.2) + 0.4\mathcal{N}(0, 1) \\ &\quad + 0.3\mathcal{N}(0, 10), \quad l = 1, 2, 3. \end{aligned} \quad (52)$$

In Figure 2, we compare the auxiliary BF with perfect knowledge of the noise distributions, and several CRPF algorithms that use the cost and risk functions proposed in Section 2.2 (see (22)–(24)). For all CRPF methods, the forgetting factor was $\lambda = 0.95$, but we ran algorithms with different values of q , $q = 1, 2$, and functions μ_1 and μ_2 (see (28) and (29)). For the latter function μ_2 , we set $\delta = 0.01$ and $\beta = 2$. The propagation mechanism for the CRPF methods consisted of the sequence of Gaussian densities given by (30) and (31), with initial value $\sigma_0^{2,(i)} = 10$ for all i .

Figure 2a shows the system trajectory in a single run and the estimates corresponding to the BF and CRPF algorithms. The trajectory started in an unknown position close to (0, 0) and evolved for one hour, with sampling period $T_s = 2$ seconds. It is apparent that all the algorithms remained locked to the vehicle position during the whole simulation interval.

Local selection (resampling) at the i th PE.

$$\text{For } k = i - 1, i, i + 1, \hat{\pi}_{t+1}^{(k)} = \begin{cases} \frac{(\mathcal{R}_{t+1}^{(k)})^{-1}}{\sum_{l=i-1}^{i+1} (\mathcal{R}_{t+1}^{(l)})^{-1}} & (\mu_1), \\ \frac{(\mathcal{R}_{t+1}^{(k)} - \min_{j \in \{i-1, i, i+1\}} \mathcal{R}_{t+1}^{(j)} + \delta)^{-\beta}}{\sum_{l=i-1}^{i+1} (\mathcal{R}_{t+1}^{(l)} - \min_{j \in \{i-1, i, i+1\}} \mathcal{R}_{t+1}^{(j)} + \delta)^{-\beta}} & (\mu_2). \end{cases}$$

Selection. $(\hat{\mathbf{x}}_t^{(i)}, \hat{\mathbf{c}}_t^{(i)}) = (\mathbf{x}_t^{(l)}, \mathbf{c}_t^{(l)})$, $l \in \{i - 1, i, i + 1\}$, with probability $\hat{\pi}_{t+1}^{(l)}$.

ALGORITHM 2: Local resampling for the CRPF algorithm.

Initialization. For $i = 1, \dots, M$,

$$\mathbf{x}_0^{(i)} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_2),$$

$$w_0^{(i)} = \frac{1}{M}.$$

Recursive update.

For $t = 1, \dots, K$,

For $i = 1, \dots, M$,

$$\hat{\mathbf{x}}_t^{(i)} = f_x(\mathbf{x}_{t-1}^{(i)}),$$

$$\kappa_i = k, \text{ with probability } p(\mathbf{y}_t | \hat{\mathbf{x}}_t^{(k)}) w_{t-1}^{(k)},$$

$$\mathbf{x}_t^{(i)} \sim p[\mathbf{x}_t | \mathbf{x}_{t-1}^{(\kappa_i)}].$$

Weight update. $\tilde{w}_t^{(i)} = \frac{p(\mathbf{y}_t | \mathbf{x}_t^{(i)})}{p(\mathbf{y}_t | \hat{\mathbf{x}}_t^{(\kappa_i)})}$.

Weight normalization. $w_t^{(i)} = \frac{\tilde{w}_t^{(i)}}{\sum_{k=1}^M \tilde{w}_t^{(k)}}$.

ALGORITHM 3: Auxiliary BF for the 2-dimensional positioning problem.

The latter observation is confirmed by the mean absolute deviation plot in Figure 2b. The deviation signal was computed as

$$e_t = \frac{1}{50} \frac{1}{2} \sum_{j=1}^{50} \left| x_{1,t,j} - x_{1,t,j}^{\text{est}} \right| + \left| x_{2,t,j} - x_{2,t,j}^{\text{est}} \right|, \quad (53)$$

where j is the simulation number, $\mathbf{x}_{t,j} = [x_{1,t,j}, x_{2,t,j}]^T$ is the true position at time t , and $\mathbf{x}_{t,j}^{\text{est}} = [x_{1,t,j}^{\text{est}}, x_{2,t,j}^{\text{est}}]^T$ is the corresponding estimate obtained with the particle filter. We observe that the CRPF algorithms with μ_2 attained the lowest deviation and outperformed the auxiliary BF. Although it is not shown here, the auxiliary BF improved its performance as the sampling period was decreased,³ and achieved a lower deviation than the CRPFs for $T_s \leq 0.5$ second. The reason is that, as T_s decreases, the correlation of the states increases due to the variation of \mathbf{G}_u , and the BF exploits this statistical information better. Therefore, we can conclude that the BF can be more accurate when strong statistical information is available, and that the proposed CRPFs are more robust and

³Obviously, the BF will also deliver a better performance as the number of particles M grows. In fact, it can be shown [4] that the estimate of the posteriori pdf and its moments obtained from the BF converge uniformly to the true density and the true values of the moments. This means that, as $M \rightarrow \infty$, the state estimates given by the BF become optimal (in the mean square error sense) and that for large M , the BF will outperform the CRPF algorithm.

steadily attain a good performance for a wider range of scenarios. This conclusion is easily confirmed with the remaining experiments presented in this section.

Figures 3 and 4 show the trajectories and the mean absolute deviations for the BF and CRPF algorithms when the sampling period was increased to $T_s = 5$ seconds and $T_s = 10$ seconds, respectively. Note that increasing T_s also increases the speed measurement error. As before, the CRPF techniques with μ_2 outperformed the BF in the long term.

Because of its better performance, we also checked the behavior of the CRPF method that uses μ_2 for different values of parameter δ . Figure 5a shows the true position and the estimates obtained using three different values of δ , namely, 0.1, 0.01, and 0.001, with fixed $\beta = 2$. All the algorithms appear to perform similarly for the considered range of values. This is confirmed with the results presented in Figure 5b in terms of the mean absolute deviation. They also illustrate the robustness and stability of the method.

In the following, unless it is stated differently, the CRPF algorithm was always implemented with μ_2 and parameters $q = 2$, $\delta = 0.01$, and $\beta = 2$. The sampling period was also fixed and was $T_s = 5$ seconds.

5.2. Mixture Gaussian system and observation noise—Gaussian BF

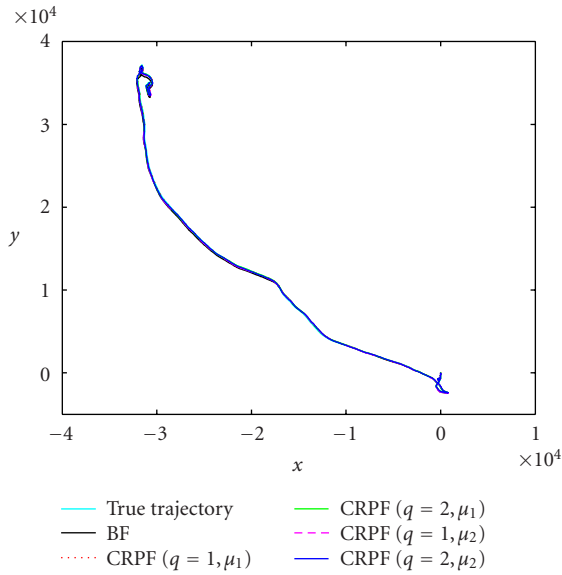
Figure 6 shows the results (trajectory and mean deviation) obtained with the same system and observation noise distributions as in Section 5.1 when the auxiliary BF (labeled as BF (Gaussian)) is mismatched with the dynamical system and models the noise processes with Gaussian densities:

$$\begin{aligned} p[\mathbf{u}_t] &= \mathcal{N}(\mathbf{0}, \sqrt{0.2}\mathbf{I}_2), \\ p[w_{l,t}] &= \mathcal{N}(0, 0.2), \quad l = 1, 2, 3. \end{aligned} \quad (54)$$

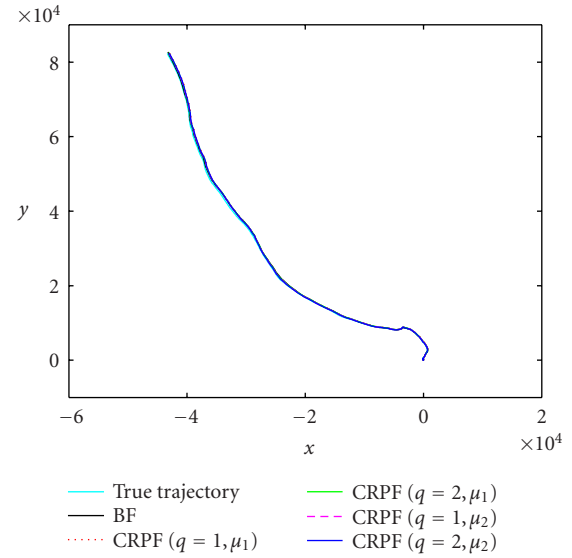
It is apparent that the use of the correct statistical information is critical for the bootstrap algorithm (in the figure, we also plotted the result obtained when the BF used the true mixture Gaussian density—labeled as BF (M-Gaussian)). Note that the CRPF algorithm also drew the state particles from a Gaussian sequence of densities (see Section 2.2), but it attained a superior performance compared to the BF.

5.3. Local versus multinomial resampling

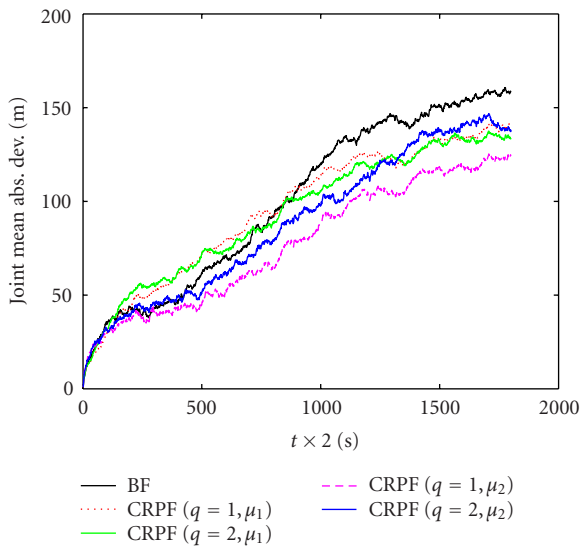
We have verified the performance of the CRPF that uses the new resampling scheme proposed in Section 4. The results



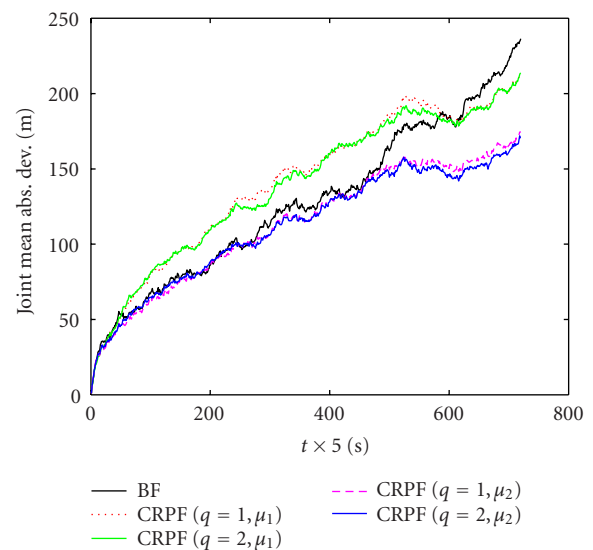
(a)



(a)



(b)



(b)

FIGURE 2: Mixture Gaussian noise processes. $T_s = 2$ seconds. (a) Trajectory. (b) Mean absolute deviation.

FIGURE 3: Mixture Gaussian noise processes. $T_s = 5$ seconds. (a) Trajectory. (b) Mean absolute deviation.

can be observed in Figure 7. The CRPF with local resampling shows approximately the same performance as the BF with perfect knowledge of the noise statistics. Although it presents a slight degradation with respect to the CRPF with multinomial resampling, the feasibility of a simple parallel implementation makes the local resampling method extremely appealing.

5.4. Different estimation criteria

Figure 8 compares the trajectory and mean deviation of two CRPF algorithms that used different criteria to obtain the estimates of the state: the minimum cost estimate $\hat{\mathbf{x}}_t^{\min}$

(see (20)) and the mean cost estimate $\hat{\mathbf{x}}_t^{\text{mean}}$ (see (21)). It is clear that both algorithms performed similarly and outperformed the BF in the long term.

5.5. Laplacian noise

Finally, we have repeated our experiment by modeling the noises using Laplacian distributions, that is,

$$p[\mathbf{u}_t] = \mathcal{L}(\mathbf{0}, \sqrt{0.5}\mathbf{I}_2) = \frac{1}{0.5} e^{-|\mathbf{u}_t|/0.5},$$

$$p[w_{l,t}] = 0.3\mathcal{L}(0, 0.5) = \frac{1}{0.5} e^{-|w_{l,t}|/0.5}, \quad l = 1, 2, 3. \quad (55)$$

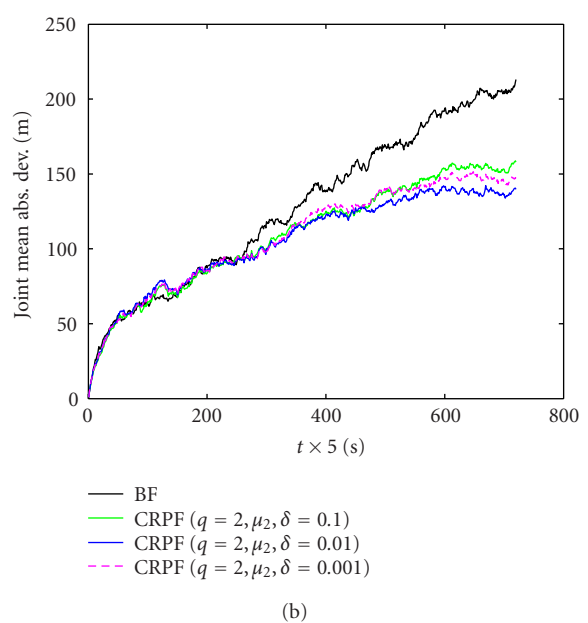
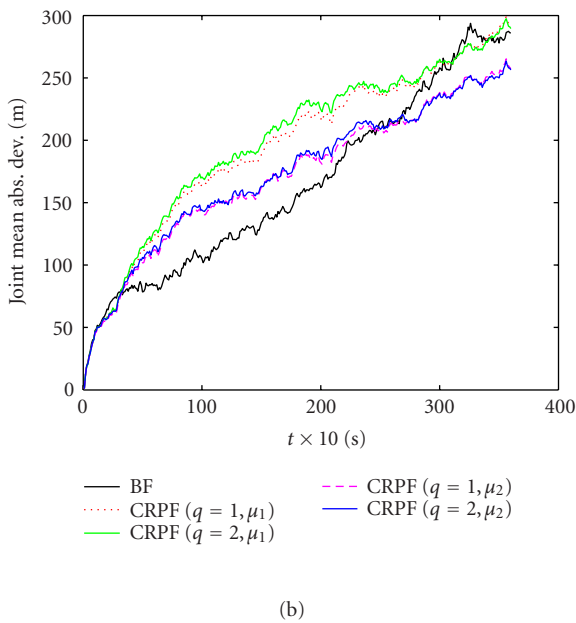
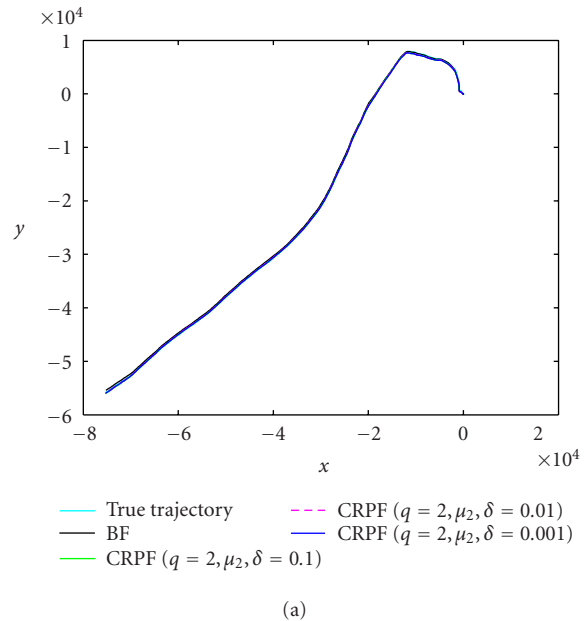
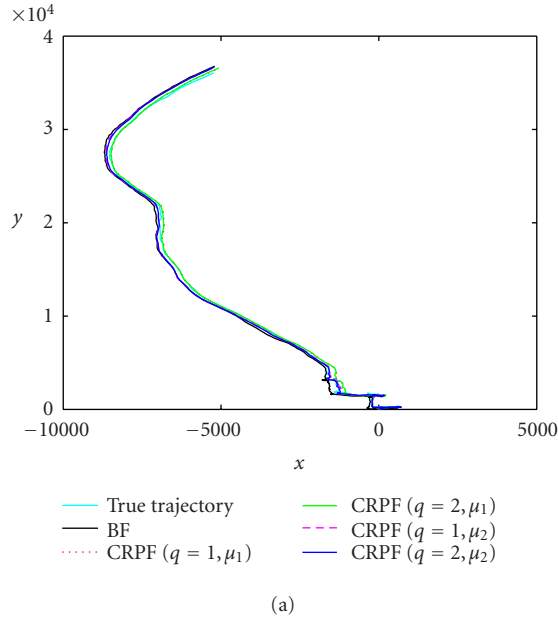


FIGURE 4: Mixture Gaussian noise processes. $T_s = 10$ seconds. (a) Trajectory. (b) Mean absolute deviation.

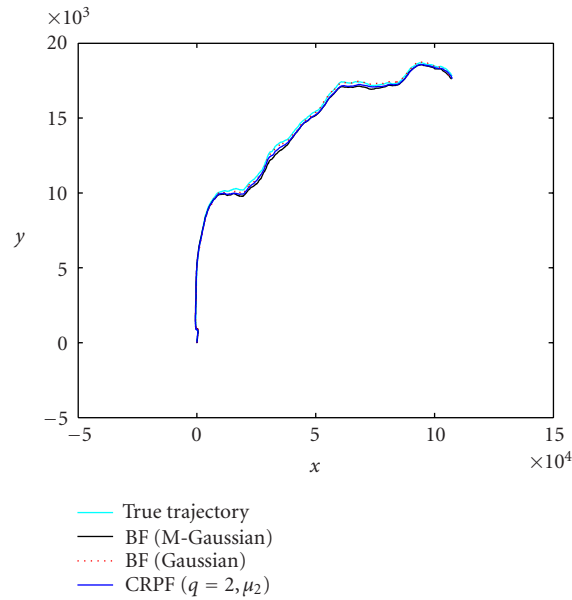
FIGURE 5: Different δ values. $T_s = 5$ seconds. (a) Trajectory. (b) Mean absolute deviation.

Figure 9 depicts the results obtained for the BF with perfect knowledge of the probability distribution of the noise and the CRPF algorithm. Again, the proposed method attained better performance in terms of mean absolute deviation.

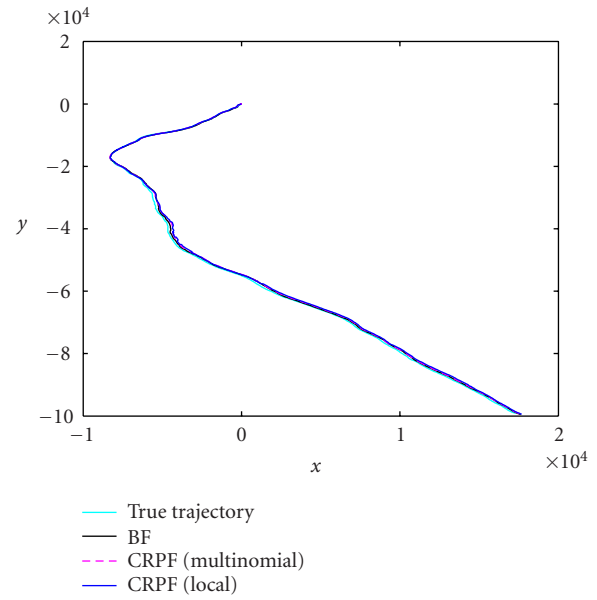
6. CONCLUSIONS

Particle filters provide optimal numerical solutions in problems that amount to estimation of unobserved time-varying states of dynamic systems. Such methods rely on the knowledge of prior probability distributions of the initial state

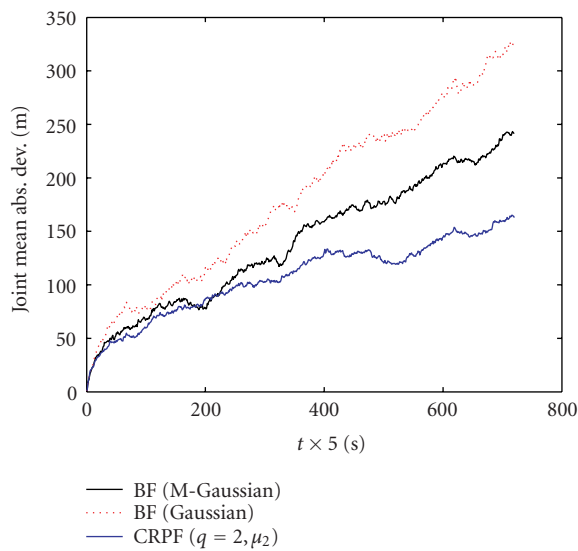
and noise processes that affect the system, and require the ability to evaluate likelihood functions and the state transition densities. Under these assumptions, different methods have been proposed that recursively estimate posterior densities by generating a collection of samples and associated importance weights. In this paper, we introduced a new class of particle filtering methods that aim at the estimation of system states from available observations *without a priori knowledge of any probability density functions*. The proposed method is based on cost functions that measure the quality of the state signal estimates given the available observations.



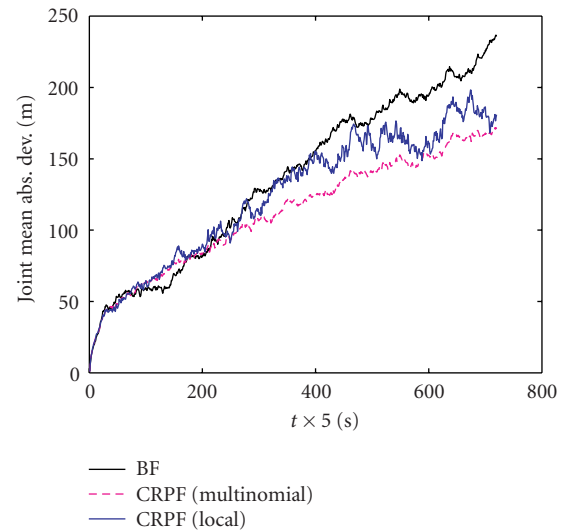
(a)



(a)



(b)



(b)

FIGURE 6: Mixture Gaussian system and observation noise—Gaussian BF. $T_s = 5$ seconds. (a) Trajectory. (b) Mean absolute deviation.

Since they do not assume explicit probabilistic models for the dynamic system, the proposed techniques, which have been termed CRPFs, are more robust than standard particle filters in problems where there is uncertainty (or a mismatch with physical phenomena) in the probabilistic model of the dynamic system. The basic concepts related to the formulation and design of these new algorithms, as well as theoretical results concerning their convergence, were provided. We also proposed a *local resampling* scheme that allows for simple implementations of the CRPF techniques with parallel VLSI hardware. Computer simulation results illustrate the

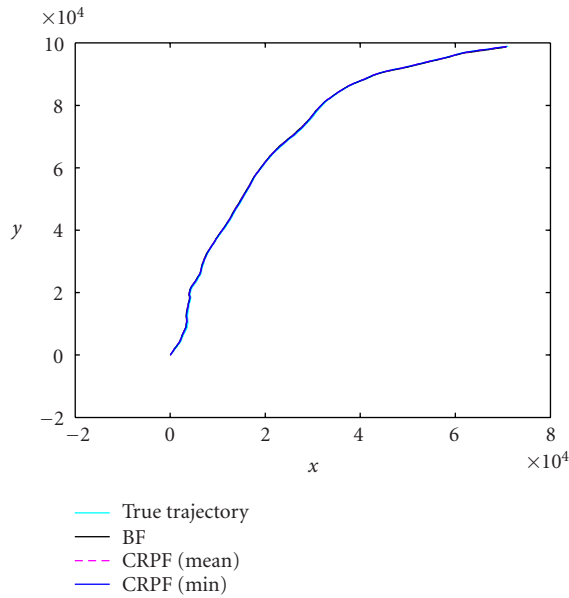
FIGURE 7: Local versus multinomial resampling. $T_s = 5$ seconds. (a) Trajectory. (b) Mean absolute deviation.

robustness and the excellent performance of the proposed algorithms when compared to the popular auxiliary BF.

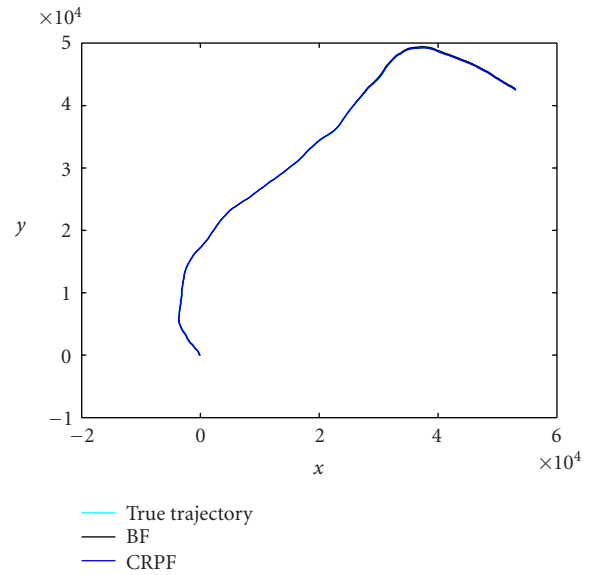
APPENDICES

A. CRPF AND STOCHASTIC APPROXIMATION

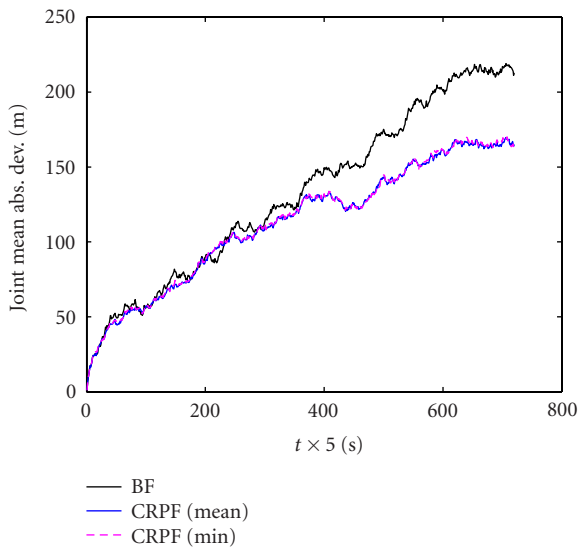
It is interesting to compare the CRPF method with the SA algorithm. The subject of SA can be traced back to the 1951 paper of Robbins and Monro [19], and a recent tutorial review can be found in [14].



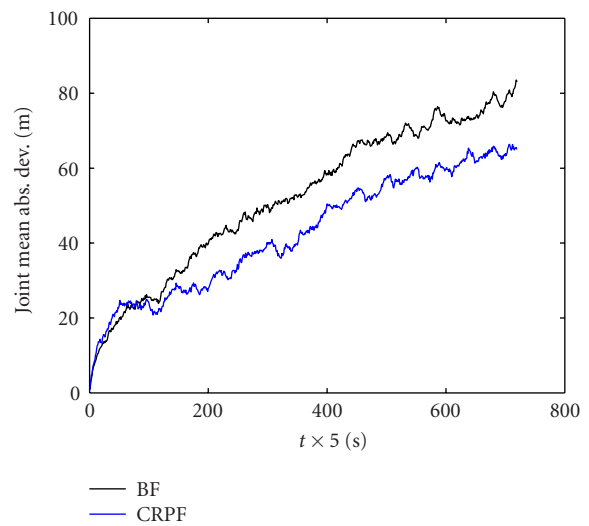
(a)



(a)



(b)



(b)

FIGURE 8: Different estimation criteria. $T_s = 5$ seconds. (a) Trajectory. (b) Mean absolute deviation.

FIGURE 9: Laplacian. $T_s = 5$ s. (a) Trajectory. (b) Mean absolute deviation.

In a typical problem addressed by SA, an objective function that has to be minimized involves expectations, for example, the minimization of $E(Q(x, \psi_t))$, where $Q(\cdot)$ is a function of the unknown x and random variables ψ_t . The problem is that the distributions of the random variables are unknown and the expectation of the function cannot be analytically found. To make the problem tractable, one approximates the expectation by simply dropping the expectation operator, and proceeding as if $E(Q(x, \psi_t)) = Q(x, \psi_t)$. Robbins and Monro proposed the following scheme that solves

for x_t :

$$\hat{x}_t = \hat{x}_{t-1} + \gamma_t Q(\hat{x}_{t-1}, \psi_t), \tag{A.1}$$

where γ_t is a sequence of positive scalars that have to satisfy the conditions $\sum_t \gamma_t = \infty$, $\sum_t \gamma_t^2 < \infty$. In the signal processing literature, the best known SA method is the LMS algorithm.

The CRPF method also attempts to estimate the unknown x_t without probabilistic assumptions. In doing so, it actually aims at *inverting* the dynamic model and, therefore,

it performs SA similarly to RM though by other means.⁴ In CRPF, the dynamics of the state are taken into account both through the propagation step and by recursively solving the optimization problem (25). Further research in CRPF from the perspective of SA can probably yield new and deeper insight of this new class of algorithms.

B. PROOF OF LEMMA 1

The proof is carried out in two steps. First, we prove the implication

$$\frac{1 - \delta}{\delta} \mathbf{S}_{\text{out}} \lim_{M \rightarrow \infty} \frac{E_{p_t^{M'}} n_M}{\mu_t(S^M(\mathbf{x}_t^{\text{opt}}, \varepsilon))} = 0 \tag{B.1}$$

$$\downarrow$$

$$\lim_{M \rightarrow \infty} \Pr \left[1 - \frac{\mu_t(S^M(\mathbf{x}_t^{\text{opt}}, \varepsilon))}{\mu_t(\{\mathbf{x}_t^{(i)}\}_{i=1}^M)} \geq \delta \right] = 0 \tag{B.2}$$

for any $\varepsilon, \delta > 0$. Then, we only need to show that (B.1) holds true under conditions (38)–(40) in order to complete the proof.

Straightforward manipulation of the inequality in (43) leads to the following equivalence chain that holds true for any $\varepsilon, \delta > 0$:

$$\lim_{M \rightarrow \infty} \Pr \left[1 - \frac{\mu_t(S^M(\mathbf{x}_t^{\text{opt}}, \varepsilon))}{\mu_t(\{\mathbf{x}_t^{(i)}\}_{i=1}^M)} \geq \delta \right] = 0 \tag{B.3}$$

$$\downarrow$$

$$\lim_{M \rightarrow \infty} \Pr [\mu_t(S^M(\mathbf{x}_t^{\text{opt}}, \varepsilon)) \leq (1 - \delta)\mu_t(\{\mathbf{x}_t^{(i)}\}_{i=1}^M)] = 0 \tag{B.4}$$

$$\downarrow$$

$$\lim_{M \rightarrow \infty} \Pr \left[\mu_t(S^M(\mathbf{x}_t^{\text{opt}}, \varepsilon)) \leq \frac{1 - \delta}{\delta} \mu_t(\{\mathbf{x}_t^{(i)}\}_{i=1}^M \setminus S^M(\mathbf{x}_t^{\text{opt}}, \varepsilon)) \right] = 0, \tag{B.5}$$

where we have exploited that

$$\mu_t(\{\mathbf{x}_t^{(i)}\}_{i=1}^M) = \mu_t(S^M(\mathbf{x}_t^{\text{opt}}, \varepsilon)) + \mu_t(\{\mathbf{x}_t^{(i)}\}_{i=1}^M \setminus S^M(\mathbf{x}_t^{\text{opt}}, \varepsilon)). \tag{B.6}$$

Using the notation

$$1_A(x) = \begin{cases} 1 & \text{if } x \in A, \\ 0 & \text{otherwise,} \end{cases} \tag{B.7}$$

for the indicator function, we can write

$$\begin{aligned} & \mu_t(\{\mathbf{x}_t^{(i)}\}_{i=1}^M \setminus S^M(\mathbf{x}_t^{\text{opt}}, \varepsilon)) \\ &= \sum_{i=1}^M \mu(\Delta C(\mathbf{x}_t^{(i)} | \mathbf{y}_t)) 1_{\{\mathbf{x}: \|\mathbf{x} - \mathbf{x}_t^{\text{opt}}\| \geq \varepsilon\}}(\mathbf{x}_t^{(i)}) \\ &\leq \mathbf{S}_{\text{out}} \sum_{i=1}^M 1_{\{\mathbf{x}: \|\mathbf{x}_t^{(i)} - \mathbf{x}_t^{\text{opt}}\| \geq \varepsilon\}}(\mathbf{x}_t^{(i)}) = \mathbf{S}_{\text{out}} n_M, \end{aligned} \tag{B.8}$$

where n_M is the cardinality of the discrete set $\{\mathbf{x}_t^{(i)}\}_{i=1}^M \setminus S^M(\mathbf{x}_t^{\text{opt}}, \varepsilon)$. Therefore, using (B.8) and the equivalence between (B.3) and (B.5), we arrive at the implication

$$\lim_{M \rightarrow \infty} \Pr \left[1 - \frac{\mu_t(S^M(\mathbf{x}_t^{\text{opt}}, \varepsilon))}{\mu_t(\{\mathbf{x}_t^{(i)}\}_{i=1}^M)} \geq \delta \right] = 0 \tag{B.9}$$

$$\uparrow$$

$$\lim_{M \rightarrow \infty} \Pr \left[\mu_t(S^M(\mathbf{x}_t^{\text{opt}}, \varepsilon)) \leq \frac{1 - \delta}{\delta} \mathbf{S}_{\text{out}} n_M \right] = 0.$$

Since both $\mu(\Delta C(\mathbf{x}_t | \mathbf{y}_t)) \geq 0$ and $(1 - \delta)\mathbf{S}_{\text{out}} n_M / \delta > 0$, we can use the relationship [16, equation 4.4-5] to obtain

$$\begin{aligned} & \Pr \left[\mu_t(S^M(\mathbf{x}_t^{\text{opt}}, \varepsilon)) \leq \frac{1 - \delta}{\delta} \mathbf{S}_{\text{out}} n_M \right] \\ &\leq \frac{1 - \delta}{\delta} \mathbf{S}_{\text{out}} \frac{E_{\text{Pr}[n_M]}[n_M]}{\mu_t(S^M(\mathbf{x}_t^{\text{opt}}, \varepsilon))}, \end{aligned} \tag{B.10}$$

where we have used the fact that the supremum \mathbf{S}_{out} does not depend on M or n_M . When jointly considered, (B.9) and (B.10) yield the implication (B.1) \Rightarrow (B.2) and we only have to show that (B.1) holds true in order to complete the proof.

The expectation on the left-hand side of (B.1) can be computed by resorting to assumption (38), which yields, after straightforward manipulations,

$$\lim_{M \rightarrow \infty} E_{\text{Pr}[n_M]}[n_M] = (1 - \gamma) \lim_{M \rightarrow \infty} M \quad (\text{i.p.}) \tag{B.11}$$

Substituting (B.11) into (B.1) yields

$$\begin{aligned} & \frac{1 - \delta}{\delta} \mathbf{S}_{\text{out}} \lim_{M \rightarrow \infty} \frac{E_{\text{Pr}[n_M]}[n_M]}{\mu_t(S^M(\mathbf{x}_t^{\text{opt}}, \varepsilon))} \\ &= \frac{1 - \delta}{\delta} \mathbf{S}_{\text{out}} (1 - \gamma) \lim_{M \rightarrow \infty} \frac{M}{\mu_t(S^M(\mathbf{x}_t^{\text{opt}}, \varepsilon))} \\ &\leq \frac{1 - \delta}{\delta} \mathbf{S}_{\text{out}} (1 - \gamma) \lim_{M \rightarrow \infty} \frac{M}{\mathbf{S}_{\text{in}}} = 0, \end{aligned} \tag{B.12}$$

where the last equality is obtained from assumptions (39) and (40). The proof is complete by going back to implication (B.1) \Rightarrow (B.2).

⁴Note, however, that the notion of inversion must be understood in a broad sense, since f_y may not necessarily be invertible and, even if f_y^{-1} exists, it may happen that \mathbf{y}_t does not belong to its domain.

C. PROOF OF THEOREM 1

Using Lemma 1, we obtain that the set $S^M(\mathbf{x}_t^{\text{opt}}, \varepsilon)$ has (asymptotically) a unit probability mass after the propagation step, that is,

$$\lim_{M \rightarrow \infty} \sum_{i: \mathbf{x}_t^{(i)} \in S^M(\mathbf{x}_t^{\text{opt}}, \varepsilon)} \bar{\omega}_t^{(i)} = 1 \quad (\text{i.p.}) \quad \forall \varepsilon > 0. \quad (\text{C.1})$$

Therefore,

$$\begin{aligned} & \lim_{M \rightarrow \infty} \left| \Delta \mathcal{C}(\mathbf{x}_t^{\text{opt}} | \mathbf{y}_t) - \overline{\Delta \mathcal{C}_t} \right| \\ &= \lim_{M \rightarrow \infty} \left| \Delta \mathcal{C}(\mathbf{x}_t^{\text{opt}} | \mathbf{y}_t) \right. \\ & \quad \left. - \sum_{i: \mathbf{x}_t^{(i)} \in S^M(\mathbf{x}_t^{\text{opt}}, \varepsilon)} \bar{\omega}_t^{(i)} \Delta \mathcal{C}(\mathbf{x}_t^{(i)} | \mathbf{y}_t) \right| \quad (\text{i.p.}), \end{aligned} \quad (\text{C.2})$$

$$\begin{aligned} & \lim_{M \rightarrow \infty} \left| \Delta \mathcal{C}(\mathbf{x}_t^{\text{opt}} | \mathbf{y}_t) - \sum_{i: \mathbf{x}_t^{(i)} \in S^M(\mathbf{x}_t^{\text{opt}}, \varepsilon)} \bar{\omega}_t^{(i)} \Delta \mathcal{C}(\mathbf{x}_t^{(i)} | \mathbf{y}_t) \right| \\ & \leq \lim_{M \rightarrow \infty} \left| \Delta \mathcal{C}(\mathbf{x}_t^{\text{opt}} | \mathbf{y}_t) \right. \\ & \quad \left. - \sup_{\mathbf{x}_t \in S^M(\mathbf{x}_t^{\text{opt}}, \varepsilon)} \{ \Delta \mathcal{C}(\mathbf{x}_t | \mathbf{y}_t) \} \right| \quad (\text{i.p.}) \end{aligned} \quad (\text{C.3})$$

for all $\varepsilon > 0$.

We write the upper bound on the right-hand side of (C.3) as a function of the radius ε :

$$B(\varepsilon) = \lim_{M \rightarrow \infty} \left| \Delta \mathcal{C}(\mathbf{x}_t^{\text{opt}} | \mathbf{y}_t) - \sup_{\mathbf{x}_t \in S^M(\mathbf{x}_t^{\text{opt}}, \varepsilon)} \{ \Delta \mathcal{C}(\mathbf{x}_t | \mathbf{y}_t) \} \right|. \quad (\text{C.4})$$

It can be easily proved that $\lim_{M \rightarrow \infty} \#S^M(\mathbf{x}_t^{\text{opt}}, \varepsilon = 1/\sqrt{M}) = \infty$, where $\#$ denotes the number of elements in a discrete set. Since $\lim_{M \rightarrow \infty} 1/\sqrt{M} = 0$ and it is assumed that $\Delta \mathcal{C}(\cdot | \mathbf{y}_t)$ is continuous and bounded in $S(\mathbf{x}_t^{\text{opt}}, \varepsilon)$ for all $\varepsilon > 0$, it follows that

$$B\left(\varepsilon = \frac{1}{\sqrt{M}}\right) = 0 \quad (\text{i.p.}) \quad (\text{C.5})$$

and, by exploiting the fact that the left-hand side of (C.2) does not depend on ε , we can readily use (C.5) to obtain

$$\begin{aligned} & \lim_{M \rightarrow \infty} \left| \Delta \mathcal{C}(\mathbf{x}_t^{\text{opt}} | \mathbf{y}_t) - \overline{\Delta \mathcal{C}_t} \right| \\ & \leq B\left(\varepsilon = \frac{1}{\sqrt{M}}\right) = 0 \quad (\text{i.p.}), \end{aligned} \quad (\text{C.6})$$

which concludes the proof.

D. PROOF OF COROLLARY 1

When $\lambda = 0$, $\bar{\omega}_t^{(i)} = \pi_t^{(i)}$ and, according to Lemma 1,

$$\lim_{M \rightarrow \infty} \sum_{i: \mathbf{x}_t^{(i)} \in S^M(\mathbf{x}_t^{\text{opt}}, \varepsilon)} \pi_t^{(i)} = 1 \quad (\text{i.p.}) \quad (\text{D.1})$$

for all $\varepsilon > 0$. Hence, we can write the mean state estimate (in the limit $M \rightarrow \infty$) as

$$\lim_{M \rightarrow \infty} \bar{\mathbf{x}}_t^{\text{mean}} = \lim_{M \rightarrow \infty} \sum_{\mathbf{x}_t^{(i)} \in S^M(\mathbf{x}_t^{\text{opt}}, \varepsilon)} \pi_t^{(i)} \mathbf{x}_t^{(i)} \quad (\text{D.2})$$

and, therefore, the incremental cost of the mean state estimate can be upper bounded as

$$\lim_{M \rightarrow \infty} \Delta \mathcal{C}(\bar{\mathbf{x}}_t^{\text{mean}} | \mathbf{y}_t) \leq \lim_{M \rightarrow \infty} \sup_{\mathbf{x}_t \in S^M(\mathbf{x}_t^{\text{opt}}, \varepsilon)} \Delta \mathcal{C}(\mathbf{x}_t | \mathbf{y}_t). \quad (\text{D.3})$$

Using inequality (D.3) and the obvious fact that $\Delta \mathcal{C}(\mathbf{x}_t^{\text{opt}} | \mathbf{y}_t)$ is minimal by definition, we find that

$$\begin{aligned} & \lim_{M \rightarrow \infty} \left| \Delta \mathcal{C}(\bar{\mathbf{x}}_t^{\text{mean}} | \mathbf{y}_t) - \Delta \mathcal{C}(\mathbf{x}_t^{\text{opt}} | \mathbf{y}_t) \right| \\ & \leq \lim_{M \rightarrow \infty} \left| \sup_{\mathbf{x}_t \in S^M(\mathbf{x}_t^{\text{opt}}, \varepsilon)} \Delta \mathcal{C}(\mathbf{x}_t | \mathbf{y}_t) - \Delta \mathcal{C}(\mathbf{x}_t^{\text{opt}} | \mathbf{y}_t) \right| \\ & = B(\varepsilon), \end{aligned} \quad (\text{D.4})$$

where $B(\varepsilon)$ is the same as defined in (C.4). Therefore, we can apply the same technique as in the proof of Theorem 1 and, taking $\varepsilon = 1/\sqrt{M}$, we obtain

$$\begin{aligned} & \lim_{M \rightarrow \infty} \left| \Delta \mathcal{C}(\bar{\mathbf{x}}_t^{\text{mean}} | \mathbf{y}_t) - \Delta \mathcal{C}_t(\mathbf{x}_t^{\text{opt}} | \mathbf{y}_t) \right| \\ & \leq B\left(\varepsilon = \frac{1}{\sqrt{M}}\right) = 0 \quad (\text{i.p.}), \end{aligned} \quad (\text{D.5})$$

which concludes the proof of the corollary.

ACKNOWLEDGMENTS

This work has been supported by Ministerio de Ciencia y Tecnología of Spain and Xunta de Galicia (Project TIC2001-0751-C04-01), and the National Science Foundation (Award CCR-0082607). The authors wish to thank the anonymous reviewers of this paper for their very constructive comments that assisted us in improving the original draft. J. Míguez would also like to thank Professor Ricardo Cao for intellectually stimulating discussions which certainly contributed to the final form of this work.

REFERENCES

- [1] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Transactions of the ASME—Journal of Basic Engineering*, vol. 82, pp. 35–45, March 1960.
- [2] B. D. O. Anderson and J. B. Moore, *Optimal Filtering*, Prentice-Hall, Englewood Cliffs, NJ, USA, 1979.
- [3] S. Julier, J. Uhlmann, and H. F. Durrant-Whyte, "A new method for the nonlinear transformation of means and covariances in filters and estimators," *IEEE Trans. on Automatic Control*, vol. 45, no. 3, pp. 477–482, 2000.
- [4] D. Crisan and A. Doucet, "A survey of convergence results on particle filtering methods for practitioners," *IEEE Trans. Signal Processing*, vol. 50, no. 3, pp. 736–746, 2002.
- [5] W. J. Fitzgerald, "Markov chain Monte Carlo methods with applications to signal processing," *Signal Processing*, vol. 81, no. 1, pp. 3–18, 2001.
- [6] J. S. Liu and R. Chen, "Sequential Monte Carlo methods for dynamic systems," *Journal of the American Statistical Association*, vol. 93, no. 443, pp. 1032–1044, 1998.
- [7] A. Doucet, S. Godsill, and C. Andrieu, "On sequential Monte Carlo sampling methods for Bayesian filtering," *Statistics and Computing*, vol. 10, no. 3, pp. 197–208, 2000.
- [8] N. Gordon, D. Salmond, and A. F. M. Smith, "Novel approach to nonlinear and non-Gaussian Bayesian state estimation," *IEE Proceedings Part F: Radar and Signal Processing*, vol. 140, no. 2, pp. 107–113, 1993.
- [9] M. K. Pitt and N. Shephard, "Auxiliary variable based particle filters," in *Sequential Monte Carlo Methods in Practice*, A. Doucet, N. de Freitas, and N. Gordon, Eds., pp. 273–293, Springer, New York, NY, USA, 2001.
- [10] A. Doucet, N. de Freitas, and N. Gordon, Eds., *Sequential Monte Carlo Methods in Practice*, Springer, New York, NY, USA, 2001.
- [11] M. L. Honig, U. Madhow, and S. Verdú, "Blind adaptive multiuser detection," *IEEE Transactions on Information Theory*, vol. 41, no. 4, pp. 944–960, 1995.
- [12] J. R. Treichler, M. G. Larimore, and J. C. Harp, "Practical blind demodulators for high-order QAM signals," *Proceedings of the IEEE*, vol. 86, no. 10, pp. 1907–1926, 1998.
- [13] J. Míguez and L. Castedo, "A linearly constrained constant modulus approach to blind adaptive multiuser interference suppression," *IEEE Communications Letters*, vol. 2, no. 8, pp. 217–219, 1998.
- [14] T. L. Lai, "Stochastic approximation," *The Annals of Statistics*, vol. 31, no. 2, pp. 391–406, 2003.
- [15] M. Bolić, P. M. Djurić, and S. Hong, "New resampling algorithms for particle filters," in *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP '03)*, vol. 2, pp. 589–592, Hong Kong, April 2003.
- [16] H. Stark and J. W. Woods, *Probability and Random Processes with Applications to Signal Processing*, Prentice-Hall, Upper Saddle River, NJ, USA, 3rd edition, 2002.
- [17] M. Bolić, P. M. Djurić, and S. Hong, "Resampling algorithms for particle filters suitable for parallel VLSI implementation," in *Proc. 37th Annual Conference on Information Sciences and Systems (CISS '03)*, Baltimore, Md, March 2003.
- [18] F. Gustafsson, F. Gunnarsson, N. Bergman, U. Forssell, J. Jansson, R. Karlsson, and P.-J. Nordlund, "Particle filters for positioning, navigation, and tracking," *IEEE Transactions on Signal Processing*, vol. 50, no. 2, pp. 425–437, 2002.
- [19] H. Robbins and S. Monro, "A stochastic approximation method," *Annals of Mathematical Statistics*, vol. 22, pp. 400–407, 1951.

Joaquín Míguez was born in Ferrol, Galicia, Spain, in 1974. He obtained the Licenciado en Informática (M.S.) and Doctor en Informática (Ph.D.) degrees from Universidade da Coruña, Spain, in 1997 and 2000, respectively. Late in 2000, he joined Departamento de Electrónica e Sistemas, Universidade da Coruña, where he became an Associate Professor in July 2003. From April to December 2001, he was a Visiting Scholar in the Department of Electrical and Computer Engineering, Stony Brook University. His research interests are in the field of statistical signal processing, with emphasis on the topics of Bayesian analysis, sequential Monte Carlo methods, adaptive filtering, stochastic optimization, and their applications to multiuser communications, smart antenna systems, sensor networks, target tracking, and vehicle positioning and navigation.



Mónica F. Bugallo received the Ph.D. degree in computer engineering from Universidade da Coruña, Spain, in 2001. From 1998 to 2000, she was with the Departamento de Electrónica e Sistemas, the Universidade da Coruña, where she worked in interference cancellation applied to multiuser communication systems. In 2001, she joined the Department of Electrical and Computer Engineering, Stony Brook University, where she is currently an Assistant Professor, and teaches courses in digital communications and information theory. Her research interests lie in the area of statistical signal processing and its applications to different disciplines including communications and biology.



Petar M. Djurić received his B.S. and M.S. degrees in electrical engineering from the University of Belgrade in 1981 and 1986, respectively, and his Ph.D. degree in electrical engineering from the University of Rhode Island in 1990. From 1981 to 1986, he was a Research Associate with the Institute of Nuclear Sciences, Vinča, Belgrade. Since 1990, he has been with Stony Brook University, where he is a Professor in the Department of Electrical and Computer Engineering. He works in the area of statistical signal processing, and his primary interests are in the theory of modeling, detection, estimation, and time series analysis and its application to a wide variety of disciplines including wireless communications and biomedicine.



A Particle Filtering Approach to Change Detection for Nonlinear Systems

Babak Azimi-Sadjadi

*Electrical, Computer, and Systems Engineering Department, Rensselaer Polytechnic Institute, Troy, NY 12180-3590, USA
Email: babak@ecse.rpi.edu*

The Institute for Systems Research, University of Maryland, College Park, MD 20742, USA

P. S. Krishnaprasad

*The Institute for Systems Research, University of Maryland, College Park, MD 20742, USA
Email: krishna@isr.umd.edu*

Received 13 September 2003; Revised 22 March 2004

We present a change detection method for nonlinear stochastic systems based on particle filtering. We assume that the parameters of the system before and after change are known. The statistic for this method is chosen in such a way that it can be calculated recursively while the computational complexity of the method remains constant with respect to time. We present simulation results that show the advantages of this method compared to linearization techniques.

Keywords and phrases: nonlinear filtering, generalized likelihood ratio test, CUSUM algorithm, online change detection.

1. INTRODUCTION

Page states the change detection problem as follows [1]: “Whenever observations are taken in order it can happen that the whole set of observations can be divided into subsets, each of which can be regarded as a random sample from a common distribution, each subset corresponding to a different parameter value of the distribution. The problems to be considered in this paper are concerned with the identification of the subsamples and the detection of changes in the parameter value.”

We refer to a change or an abrupt change as any change in the parameters of the system that happens either instantaneously or much faster than any change that the nominal bandwidth of the system allows.

The key difficulty of all change detection methods is that of detecting intrinsic changes that are not necessarily directly observed but are measured together with other types of perturbations [2].

The change detection could be offline or online. In online change detection, we are only interested in detecting the change as quickly as possible (e.g., to minimize the detection delay with fixed mean time between false alarms), and the estimate of the time when the change occurs is not of importance. In offline change detection, we assume that the whole observation sequence is available at once and finding the estimate of the time of change could be one of the goals of the detection method. In this paper, we limit our concern to online detection of abrupt changes.

The change detection methods that we consider here can

be classified under the general name of likelihood ratio (LR) methods. Cumulative sum (CUSUM) and generalized LR (GLR) tests are among these methods. CUSUM was first proposed by Page [1]. The most basic CUSUM algorithm assumes that the observation signal is a sequence of stochastic variables which are independent and identically distributed (i.i.d.) with known common probability density function before the change time, and i.i.d. with another known probability density function after the change time. In the CUSUM algorithm, the log-likelihood ratio for the observation from time i to time k is calculated and its difference with its current minimum is compared with a certain threshold. If this difference exceeds the threshold, an alarm is issued.

Properties of the CUSUM algorithm have been studied extensively. Its most important property is the asymptotic optimality, which was first proven in [3]. More precisely, CUSUM is optimal, with respect to the worst mean delay, when the mean time between false alarms goes to infinity. This asymptotic point of view is convenient in practice because a low rate of false alarms is always desirable.

In the case of unknown system parameters after change, the GLR algorithm can be used as a generalization of the CUSUM algorithm. Since, in this algorithm, the exact information of the change pattern is not known, the LR is maximized over all possible change patterns.¹

¹If the maximum does not exist, the supremum of the LR should be calculated.

For stochastic systems with linear dynamics and linear observations, the observation sequence is not i.i.d. Therefore, the regular CUSUM algorithm cannot be applied for detection of changes in such systems. However, if such systems are driven by Gaussian noise, the innovation process associated with the system is known to be a sequence of independent random variables. The regular CUSUM algorithm or its more general counterpart, GLR, can be applied to this innovation process [2, 4].

In this paper, we are interested in the change detection problem for stochastic systems with nonlinear dynamics and observations. We show that for such systems, the complexity of the CUSUM algorithm grows with respect to time. This growth in complexity cannot be tolerated in practical problems. Therefore, instead of the statistic used in the CUSUM algorithm, we introduce an alternative statistic. We show that with this statistic, the calculation of the LR can be done recursively and the computational complexity of the method stays constant with respect to time.

Unlike the linear case, change detection for nonlinear stochastic systems has not been investigated in any depth. In the cases where a nonlinear system experiences a sudden change, linearization and change detection methods for linear systems are the main tools for solving the change detection problem (see, e.g., [5]). The reason this subject has not been pursued is clear; even when there is no change, the estimation of the state of the system, given the observations, results in an infinite-dimensional nonlinear filter [6], and the change in the system can only make the estimation harder.

In the last decade, there has been an increasing interest in simulation-based nonlinear filtering methods. These filtering methods are based on a gridless approximation of the conditional density of the state, given the observations. Gridless simulation-based filtering, now known by many different names such as particle filtering (PAF) [7, 8], the condensation algorithm [9], the sequential Monte Carlo (SMC) method [10], and Bayesian bootstrap filtering [11], was first introduced by Gordon et al. [11] and then it was rediscovered independently by Isard and Blake [9] and Kitagawa [12].

The theoretical results regarding the convergence of the approximate conditional density given by PAF to the true conditional density (in some proper sense) suggest that this method is a strong alternative for nonlinear filtering [7]. The advantage of this method over the nonlinear filter is that PAF is a finite-dimensional filter. The authors believe that PAF and its modifications are a starting point to study change detection for nonlinear stochastic systems. In this paper, we use the results in [13] and we develop a new change detection method for nonlinear stochastic systems.

In [13], we showed that when the number of satellites is below a critical number, linearization methods such as extended Kalman filtering (EKF) result in an unacceptable position error for an integrated inertial navigation system/global positioning system (INS/GPS). We also showed that the approximate nonlinear filtering methods, the projection particle filter [13], in particular, are capable of providing an acceptable estimate of the position in the same situation.

If the carrier phase is used for position information in an integrated INS/GPS, one sudden change that happens rather often is the cycle slip. A cycle slip happens when the phase of the received signal estimated by the phase lock loop in the receiver has a sudden jump. An integrated INS/GPS with carrier phase receiver is used as an application for the method introduced in this paper for detection of a cycle slip with known strength.

In Section 2, we state the approximate nonlinear filtering method used in this paper. In Section 3, we briefly define the change detection problem. In Section 4, we review the CUSUM algorithm for linear systems with additive changes. Then, in Sections 5 and 6, we present a new change detection method for nonlinear stochastic systems. In Section 7, we lay out the formulation for an integrated INS/GPS. In Section 8, we present some simulation results. In Section 9, we summarize the results and lay out the future work.

2. APPROXIMATE NONLINEAR FILTERING

Consider the dynamical system

$$\begin{aligned}\mathbf{x}_{k+1} &= \mathbf{f}_k(\mathbf{x}_k) + G_k(\mathbf{x}_k)\mathbf{w}_k, \\ \mathbf{y}_k &= \mathbf{h}_k(\mathbf{x}_k) + \mathbf{v}_k,\end{aligned}\quad (1)$$

where the distribution of \mathbf{x}_0 is given, $\mathbf{x}_k \in \mathbb{R}^n$, $\mathbf{y}_k \in \mathbb{R}^d$, and $\mathbf{w}_k \in \mathbb{R}^q$, and $\mathbf{v}_k \in \mathbb{R}^d$ are white noise processes with known statistics, and the functions $\mathbf{f}_k(\cdot)$ and $\mathbf{h}_k(\cdot)$ and the matrix $G_k(\cdot)$ have the proper dimensions. The noise processes \mathbf{w}_k , \mathbf{v}_k , $k = 0, 1, \dots$, and the initial condition \mathbf{x}_0 are assumed independent.

We assume the initial distribution for \mathbf{x}_0 is given. The goal is to find the conditional distribution of the state, given the observation, that is, $P_k(d\mathbf{x}_k | \mathcal{Y}_1^k)$, where $\mathcal{Y}_1^k = \{\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_k\}$ is the observation up to and including time k . The propagation of the conditional distribution, at least conceptually, can be expressed as follows [6].

Step (1). Initialization:

$$P_0(d\mathbf{x}_0 | \mathbf{y}_0) = P(d\mathbf{x}_0). \quad (2)$$

Step (2). Diffusion:

$$\begin{aligned}P_{(k+1)^-}(d\mathbf{x}_{k+1} | \mathcal{Y}_1^k) \\ = \int P(d\mathbf{x}_{k+1} | \mathbf{x}_k) P_k(d\mathbf{x}_k | \mathcal{Y}_1^k).\end{aligned}\quad (3)$$

Step (3). Bayes' rule update:

$$\begin{aligned}P_{(k+1)}(d\mathbf{x}_{k+1} | \mathcal{Y}_1^{k+1}) \\ = \frac{p(\mathbf{y}_{k+1} | \mathbf{x}_{k+1}) P_{(k+1)^-}(d\mathbf{x}_{k+1} | \mathcal{Y}_1^k)}{\int p(\mathbf{y}_{k+1} | \mathbf{x}_{k+1}) P_{(k+1)^-}(d\mathbf{x}_{k+1} | \mathcal{Y}_1^k)}.\end{aligned}\quad (4)$$

Step (4). $k \leftarrow k + 1$; go to Step (2).

We have assumed that $P(d\mathbf{y}_{k+1} | \mathbf{x}_{k+1}) = p(\mathbf{y}_{k+1} | \mathbf{x}_{k+1}) d\mathbf{y}_{k+1}$ and $p(\mathbf{y}_{k+1} | \mathbf{x}_{k+1})$ is the conditional density of the observation, given the state at time $k + 1$.

The conditional distribution given by the above steps is exact, but in general, it can be viewed as an infinite-dimensional filter, thus not implementable. PAF, in brief,

Step (1). Initialization.
Sample N i.i.d. random vectors $\mathbf{x}_0^1, \dots, \mathbf{x}_0^N$ with the initial distribution $P_0(d\mathbf{x})$.

Step (2). Diffusion.
Find $\hat{\mathbf{x}}_{k+1}^1, \dots, \hat{\mathbf{x}}_{k+1}^N$ from the given $\mathbf{x}_k^1, \dots, \mathbf{x}_k^N$, using the dynamic rule:

$$\mathbf{x}_{k+1} = \mathbf{f}_k(\mathbf{x}_k) + G_k(\mathbf{x}_k)\mathbf{v}_k.$$

Step (3). Use Bayes' rule and find the empirical distribution

$$P_{k+1}^N(d\mathbf{x}) = \sum_{j=1}^N \frac{p(\mathbf{y}_{k+1} | \hat{\mathbf{x}}_{k+1}^j)}{\sum_{i=1}^N p(\mathbf{y}_{k+1} | \hat{\mathbf{x}}_{k+1}^i)} \delta_{\hat{\mathbf{x}}_{k+1}^j}(d\mathbf{x}).$$

Step (4). Resampling.
Sample $\mathbf{x}_{k+1}^1, \dots, \mathbf{x}_{k+1}^N$ according to $P_{k+1}^N(d\mathbf{x})$.

Step (5). $k \leftarrow k + 1$; go to Step (2).

ALGORITHM 1: Particle filtering.

is an approximation method that mimics the above calculations with a finite number of operations using the Monte Carlo method. Algorithm 1 shows one manifestation of PAF [7, 11].

It is customary to call $\mathbf{x}_k^1, \dots, \mathbf{x}_k^N$ particles. The key idea in PAF is to eliminate the particles that have low importance weights $p(\mathbf{y}_k | \mathbf{x}_k)$ and to multiply particles having high importance weights [11, 14]. The surviving particles are thus approximately distributed according to $P_k^N(d\mathbf{x})$. This automatically makes the approximation one of better resolution in the areas where the probability is higher.

In the simulations in this paper, we use a modified version of the classical PAF method called projection PAF. For completeness sake, we repeat the algorithm that was given in [13]. In projection PAF, we assume that the conditional density of the state of the system, given the observation, is close to an exponential family of densities \mathcal{F} defined as follows.²

Definition 1 (Brigo [15]). Let $\{c_1(\cdot), \dots, c_p(\cdot)\}$ be affinely independent³ scalar functions defined on k dimensional Euclidean space \mathbb{R}^k . Assume that

$$\Theta_0 = \left\{ \theta \in \mathbb{R}^p : \Upsilon(\theta) = \log \int \exp(\theta^T \mathbf{c}(\mathbf{x})) d\mathbf{x} < \infty \right\} \quad (5)$$

is a convex set with a nonempty interior, where $\mathbf{c}(\mathbf{x}) = [c_1(\mathbf{x}), \dots, c_p(\mathbf{x})]^T$. Then \mathcal{F} , defined as

$$\begin{aligned} \mathcal{F} &= \{p(\cdot, \theta), \theta \in \Theta\}, \\ p(\mathbf{x}, \theta) &:= \exp[\theta^T \mathbf{c}(\mathbf{x}) - \Upsilon(\theta)], \end{aligned} \quad (6)$$

where $\Theta \subseteq \Theta_0$ is open, is called an exponential family of probability densities.

²For details of the assumptions and the convergence results for the projection PAF, see [13].

³ $\{c_1, \dots, c_p\}$ are affinely independent if for distinct points $\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_{p+1}$, $\sum_{i=1}^{p+1} \lambda_i c(\mathbf{x}_i) = 0$ and $\sum_{i=1}^{p+1} \lambda_i = 0$ imply $\lambda_1 = \lambda_2 = \dots = \lambda_{p+1} = 0$ [16].

Step (1). Initialization.
Sample N i.i.d. random vectors $\mathbf{x}_0^1, \dots, \mathbf{x}_0^N$ with the density $p_0(\mathbf{x})$.

Step (2). Diffusion.
Find $\hat{\mathbf{x}}_{k+1}^1, \dots, \hat{\mathbf{x}}_{k+1}^N$ from the given $\mathbf{x}_k^1, \dots, \mathbf{x}_k^N$, using the dynamic rule:

$$\mathbf{x}_{k+1} = \mathbf{f}_k(\mathbf{x}_k) + G_k(\mathbf{x}_k)\mathbf{v}_k.$$

Step (3). Find the MLE of $\hat{\theta}_{(k+1)^-}$, given $\hat{\mathbf{x}}_{k+1}^1, \dots, \hat{\mathbf{x}}_{k+1}^N$:

$$\hat{\theta}_{(k+1)^-} = \arg \max_{\theta} \prod_{i=1}^N \exp(\theta^T \mathbf{c}(\hat{\mathbf{x}}_{k+1}^i) - \Upsilon(\theta)).$$

Step (4). Use Bayes' rule

$$\begin{aligned} p(\mathbf{x}, \hat{\theta}_{(k+1)}) &= \frac{\exp(\hat{\theta}_{(k+1)}^T \mathbf{c}(\mathbf{x}) - \Upsilon(\hat{\theta}_{(k+1)})) p(\mathbf{y}_{k+1} | \mathbf{x})}{\int \exp(\hat{\theta}_{(k+1)}^T \mathbf{c}(\mathbf{x}) - \Upsilon(\hat{\theta}_{(k+1)})) p(\mathbf{y}_{k+1} | \mathbf{x}) d\mathbf{x}}. \end{aligned}$$

Step (5). Resampling.
Sample $\mathbf{x}_{k+1}^1, \dots, \mathbf{x}_{k+1}^N$ according to $p(\mathbf{x}, \hat{\theta}_{(k+1)})$.

Step (6). $k \leftarrow k + 1$; go to Step (2).

ALGORITHM 2: Projection particle filtering for an exponential family of densities.

With this definition for the exponential family of densities, the projection PAF algorithm is stated as in Algorithm 2.

3. CHANGE DETECTION: PROBLEM DEFINITION

Online detection of a change can be formulated as follows [2]. Let \mathcal{Y}_1^k be a sequence of observed random variables with conditional densities $p_{\theta}(\mathbf{y}_k | \mathbf{y}_{k-1}, \dots, \mathbf{y}_1)$. Before the unknown change time t_0 , the parameter of the conditional density θ is constant and equal to θ_0 . After the change, this parameter is equal to θ_1 . In online change detection, one is interested in detecting the occurrence of such a change. The exact time and the estimation of the parameters before and after the change are not required. In the case of multiple changes, we assume that the changes are detected fast enough so that in each time instance, only one change can be considered. Online change detection is performed by a stopping rule [2]

$$t_a = \inf \{k : g_k(\mathcal{Y}_1^k) \geq \lambda\}, \quad (7)$$

where λ is a threshold, $\{g_k\}_{k \geq 1}$ is a family of functions, and t_a is the alarm time, that is, the time when change is detected.

If $t_a < t_0$, then a false alarm has occurred. The criterion for choosing the parameter λ and the family of functions $\{g_k\}_{k \geq 1}$ is to minimize the detection delay for the fixed mean time between false alarms.

4. ADDITIVE CHANGES IN LINEAR DYNAMICAL SYSTEMS

Consider the system

$$\begin{aligned} \mathbf{x}_{k+1} &= F_k \mathbf{x}_k + G_k \mathbf{w}_k + \Gamma_k \Upsilon_{\mathbf{x}}(k, t_0), \\ \mathbf{y}_k &= H_k \mathbf{x}_k + \mathbf{v}_k + \Xi_k \Upsilon_{\mathbf{y}}(k, t_0), \end{aligned} \quad (8)$$

where $\mathbf{x}_k \in \mathbb{R}^n$, $\mathbf{y}_k \in \mathbb{R}^d$, and $\mathbf{w}_k \in \mathbb{R}^q$ and $\mathbf{v}_k \in \mathbb{R}^d$ are white noise processes with known statistics. F_k , G_k , H_k , Γ_k , and Ξ_k are matrices of proper dimensions, and $Y_x(k, t_0)$ and $Y_y(k, t_0)$ are the dynamic profiles of the assumed changes, of dimensions $\tilde{n} \leq n$ and $\tilde{d} \leq d$, respectively. \mathbf{w}_k and \mathbf{v}_k are white Gaussian noise processes, independent of the initial condition \mathbf{x}_0 . It is assumed that $Y_x(k, t_0) = 0$ and $Y_y(k, t_0) = 0$ for $k < t_0$, but we do not necessarily have the exact knowledge of the dynamic profile and the gain matrices Γ_k and Ξ_k .

For the case of known parameters before and after change, the CUSUM [2] algorithm can be used, and it is well known that the change detection method has the following form:

$$\begin{aligned} t_a &= \min \{k \geq 1 \mid g_k \geq \lambda\}, \\ g_k &= \max_{1 \leq j \leq k} S_j^k, \\ S_j^k &= \ln \frac{\prod_{i=j}^k P_{\rho(i,j)}(\epsilon_i)}{\prod_{i=j}^k p_0(\epsilon_i)}, \end{aligned} \quad (9)$$

where ϵ_i is the innovation process calculated using Kalman filtering assuming that no change occurred, and $\rho(i, j)$ is the mean of the innovation process at time j conditioned on the change occurring at the time i . p_0 and $p_{\rho(\cdot, \cdot)}$ are Gaussian densities with means 0 and $\rho(\cdot, \cdot)$, respectively. The covariance matrix for these two densities is the same and is calculated using Kalman filtering. S_j^k is the LR between two hypotheses: change occurrence at j and no change occurrence.

When the parameter after change is not known, GLR can be used to calculate g_k [4]:

$$g_k = \max_{1 \leq j \leq k} \sup_{Y_x, Y_y} S_j^k. \quad (10)$$

The solution for (10) is well known and can be found in many references (see [2]).

Similar to nonlinear filtering, change detection for nonlinear stochastic systems results in an algorithm that is infinite dimensional. Linearization techniques, whenever applicable, are the main approximation tool for studying the change detection problem for nonlinear systems. Although linearization techniques are computationally efficient, they are not always applicable. In the sections to come, we propose a new method based on nonlinear PAF that can be used for change detection for nonlinear stochastic systems.

5. NONLINEAR CHANGE DETECTION: PROBLEM SETUP

Consider the nonlinear system

$$\begin{aligned} \mathbf{x}_{k+1} &= \mathbf{f}_k^{i_k}(\mathbf{x}_k) + \mathbf{G}_k^{i_k}(\mathbf{x}_k)\mathbf{w}_k, \\ \mathbf{y}_k &= \mathbf{h}_k^{i_k}(\mathbf{x}_k) + \mathbf{v}_k, \end{aligned} \quad (11)$$

where

$$i_k = \begin{cases} 0, & k < t_0, \\ 1, & k \geq t_0, \end{cases} \quad (12)$$

and the functions $\mathbf{f}_k^0(\cdot)$, $\mathbf{f}_k^1(\cdot)$, $\mathbf{h}_k^0(\cdot)$, $\mathbf{h}_k^1(\cdot)$ and the matrices $\mathbf{G}_k^0(\cdot)$, $\mathbf{G}_k^1(\cdot)$ have the proper dimensions. The sudden change occurs when i_k changes from 0 to 1.

In this setup, S_j^k can be written as follows:

$$S_j^k = \ln \frac{p(\mathcal{Y}_j^k | \mathcal{Y}_1^{j-1}, t_0 = j)}{p(\mathcal{Y}_j^k | \mathcal{Y}_1^{j-1}, t_0 > k)}. \quad (13)$$

Writing (13) in a recursive form, we get

$$p(\mathcal{Y}_j^k | \mathcal{Y}_1^{j-1}, t_0 = j) = \prod_{i=j}^k p(\mathbf{y}_i | \mathcal{Y}_1^{i-1}, t_0 = j), \quad (14)$$

where $p(\mathbf{y}_i | \mathcal{Y}_1^{i-1}, t_0 = j)$ can be written as follows:

$$p(\mathbf{y}_i | \mathcal{Y}_1^{i-1}, t_0 = j) = \int p(\mathbf{y}_i | \mathbf{x}_i) P(d\mathbf{x}_i | \mathcal{Y}_1^{i-1}, t_0 = j). \quad (15)$$

To find $P(d\mathbf{x}_i | \mathcal{Y}_1^{i-1}, t_0 = j)$ or equivalently to find the density $p(\mathbf{x}_i | \mathcal{Y}_1^{i-1}, t_0 = j)$ ⁴ in (15), one needs to find an approximation for the corresponding nonlinear filter. We assume that this approximation is done using either PAF or projection PAF [13].

To calculate the LR in (13), we must calculate the conditional densities of the state, given the observation for two hypotheses (change occurrence at j and change occurrence after k). This means that two nonlinear filters should be implemented just to compare these two hypotheses. Therefore, it is clear that to use an algorithm similar to (9), k parallel nonlinear filters should be implemented. In Figure 1, we see that the computational complexity of the CUSUM algorithm grows linearly with respect to time. In most applications, this growth is not desirable. One possible way to approximate the CUSUM algorithm is to truncate the branches that fork from the main branch in Figure 1. We will explain this truncation procedure and its technical difficulties in the next few lines.

Recall that the main branch (horizontal) and the branches forked from it in Figure 1 are representing a series of nonlinear filters with specific assumptions on the change time. The dynamic and observation equations for all forked branches are the same and the only difference is the initial density. If the conditional density of the state, given the observation for a nonlinear system, with the wrong initial density converges (in some meaningful way) to the true conditional density (initialized by the true initial density), we say that the corresponding nonlinear filter is *asymptotically stable* [17].

For asymptotically stable nonlinear filters, the forked branches in Figure 1 converge to a single branch, therefore, there is no need to implement several parallel nonlinear filters. In other words, after each branching, the independent nonlinear filter is used for a period of time and then this branch converges to the branches that have forked earlier,

⁴If the density exists.

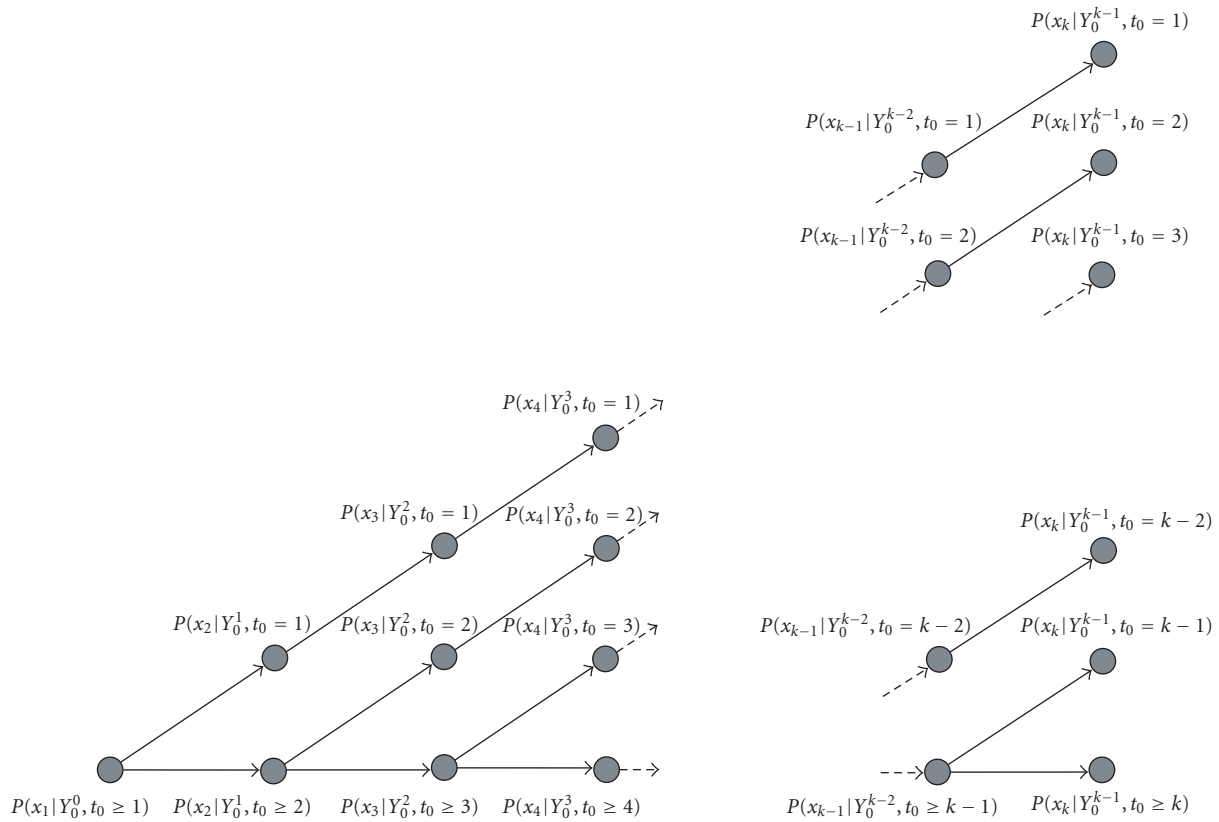


FIGURE 1: Combination of nonlinear filters used in the CUSUM change detection algorithm.

that is, joins them. The time needed for the branch of the independent nonlinear filter to join the other forked branches depends on the convergence rate and the target accuracy of the approximation.

Although the procedure mentioned above can be used for asymptotically stable nonlinear filters, there are several problems associated with this method. The known theoretical results for identifying asymptotically stable filters are limited to either requiring ergodicity and the compactness of the state space [18, 19, 20], or very special cases of the observation equation [17]. The rate of convergence of the filters in different branches is another potential shortcoming of the mentioned procedure. If the convergence rate is low in comparison to the rate of parameter change in the system, then the algorithm cannot take advantage of this convergence.

6. NONLINEAR CHANGE DETECTION: NONGROWING COMPUTATIONAL COMPLEXITY

In this section, we introduce a new statistic to overcome the problem of growing computational complexity for the change detection method. We emphasize that the parameters of the system before and after change are assumed to be known. Therefore, the conditional density of the state of the system, given the observation, can be calculated using a nonlinear filter. We show that this statistic can be calculated recursively.

Consider the following statistic:

$$T_j^k = \ln \frac{p(\mathcal{Y}_j^k | \mathcal{Y}_1^{j-1}, t_0 \in \{j, \dots, k\})}{p(\mathcal{Y}_j^k | \mathcal{Y}_1^{j-1}, t_0 > k)}. \tag{16}$$

The change detection algorithm based on statistic T_j^k can be presented as

$$t_a = \min \{k \geq j \mid T_j^k \geq \lambda \text{ or } T_j^k \leq -\alpha\}, \tag{17}$$

where j is the last time when $T_j^k \geq \lambda$ or $T_j^k \leq -\alpha$, and $\lambda > 0$ and $\alpha > 0$ are chosen such that the detection delay is minimized for a fixed mean time between two false alarms.

For the rest of this paper, we assume that the probability of change at time i condition on no change before i is q , that is,

$$P(t_0 = i \mid t_0 \geq i) = q. \tag{18}$$

Without loss of generality and for simplifying the notation, we assume that $j = 1$. To calculate the statistic T_1^k , it is sufficient to find $P(d\mathbf{x}_k, t_0 \leq k | \mathcal{Y}_1^k)$ and $P(d\mathbf{x}_k, t_0 > k | \mathcal{Y}_1^k)$. Then T_1^k is given by

$$\begin{aligned} T_1^k &= \ln \left(\frac{(1-q)^k \int P(d\mathbf{x}_k, t_0 \leq k | \mathcal{Y}_1^k)}{(1-(1-q)^k) \int P(d\mathbf{x}_k, t_0 > k | \mathcal{Y}_1^k)} \right) \\ &= \ln \left(\frac{(1-q)^k}{(1-(1-q)^k)} \frac{W_k^1}{W_k^0} \right), \end{aligned} \tag{19}$$

where $W_k^0 = \int P(d\mathbf{x}_k, t_0 > k | \mathcal{Y}_1^k)$ and $W_k^1 = \int P(d\mathbf{x}_k, t_0 \leq k | \mathcal{Y}_1^k)$. Therefore, to calculate T_1^k recursively, it is sufficient to

calculate $P(d\mathbf{x}_k, t_0 \leq k | \mathcal{Y}_1^k)$ and $P(d\mathbf{x}_k, t_0 > k | \mathcal{Y}_1^k)$ recursively. $P(d\mathbf{x}_{k+1}, t_0 \leq k+1 | \mathcal{Y}_1^{k+1})$ can be written as follows:

$$\begin{aligned}
 & P(d\mathbf{x}_{k+1}, t_0 \leq k+1 | \mathcal{Y}_1^{k+1}) \\
 &= \frac{P(d\mathbf{x}_{k+1}, t_0 \leq k+1, \mathbf{y}_{k+1} | \mathcal{Y}_1^k)}{\int P(d\mathbf{x}_{k+1}, t_0 \leq k+1, \mathbf{y}_{k+1} | \mathcal{Y}_1^k) + \int P(d\mathbf{x}_{k+1}, t_0 > k+1, \mathbf{y}_{k+1} | \mathcal{Y}_1^k)} \\
 &= \frac{p(\mathbf{y}_{k+1} | \mathbf{x}_{k+1}, t_0 \leq k+1)P(d\mathbf{x}_{k+1}, t_0 \leq k+1 | \mathcal{Y}_1^k)}{\int p(\mathbf{y}_{k+1} | \mathbf{x}_{k+1}, t_0 \leq k+1)P(d\mathbf{x}_{k+1}, t_0 \leq k+1 | \mathcal{Y}_1^k) + \int p(\mathbf{y}_{k+1} | \mathbf{x}_{k+1}, t_0 > k+1)P(d\mathbf{x}_{k+1}, t_0 > k+1 | \mathcal{Y}_1^k)} \\
 &= \frac{p_1(\mathbf{y}_{k+1} | \mathbf{x}_{k+1})P(d\mathbf{x}_{k+1}, t_0 \leq k+1 | \mathcal{Y}_1^k)}{\int p_1(\mathbf{y}_{k+1} | \mathbf{x}_{k+1})P(d\mathbf{x}_{k+1}, t_0 \leq k+1 | \mathcal{Y}_1^k) + \int p_0(\mathbf{y}_{k+1} | \mathbf{x}_{k+1})P(d\mathbf{x}_{k+1}, t_0 > k+1 | \mathcal{Y}_1^k)},
 \end{aligned} \tag{20}$$

where $p_1(\mathbf{y}_{k+1} | \mathbf{x}_{k+1})$ is the conditional density of the state \mathbf{x}_{k+1} , given the observation \mathbf{y}_{k+1} , under the hypothesis that change has occurred and $p_0(\mathbf{y}_{k+1} | \mathbf{x}_{k+1})$ is the same condi-

tional density under the hypothesis that no change has occurred. Similarly, for $P(d\mathbf{x}_{k+1}, t_0 > k+1 | \mathcal{Y}_1^{k+1})$, we have the following:

$$\begin{aligned}
 & P(d\mathbf{x}_{k+1}, t_0 > k+1 | \mathcal{Y}_1^{k+1}) \\
 &= \frac{p_0(\mathbf{y}_{k+1} | \mathbf{x}_{k+1})P(d\mathbf{x}_{k+1}, t_0 > k+1 | \mathcal{Y}_1^k)}{\int p_1(\mathbf{y}_{k+1} | \mathbf{x}_{k+1})P(d\mathbf{x}_{k+1}, t_0 \leq k+1 | \mathcal{Y}_1^k) + \int p_0(\mathbf{y}_{k+1} | \mathbf{x}_{k+1})P(d\mathbf{x}_{k+1}, t_0 > k+1 | \mathcal{Y}_1^k)}.
 \end{aligned} \tag{21}$$

Also, we have

$$\begin{aligned}
 & P(d\mathbf{x}_{k+1}, t_0 \leq k+1 | \mathcal{Y}_1^k) \\
 &= \int P(d\mathbf{x}_{k+1}, t_0 \leq k+1 | \mathbf{x}_k, t_0 \leq k, \mathcal{Y}_1^k)P(d\mathbf{x}_k, t_0 \leq k | \mathcal{Y}_1^k) \\
 &\quad + \int P(d\mathbf{x}_{k+1}, t_0 \leq k+1 | \mathbf{x}_k, t_0 > k, \mathcal{Y}_1^k)P(d\mathbf{x}_k, t_0 > k | \mathcal{Y}_1^k) \\
 &= W_k^1 \int P_1^{k+1}(d\mathbf{x}_{k+1} | \mathbf{x}_k)P(d\mathbf{x}_k | t_0 \leq k, \mathcal{Y}_1^k) \\
 &\quad + qW_k^0 \int P(d\mathbf{x}_{k+1} | \mathbf{x}_k, t_0 = k+1)P(d\mathbf{x}_k | t_0 > k, \mathcal{Y}_1^k),
 \end{aligned} \tag{22}$$

$$\begin{aligned}
 & P(d\mathbf{x}_{k+1}, t_0 > k+1 | \mathcal{Y}_1^k) \\
 &= (1-q)W_k^0 \int P_0^{k+1}(d\mathbf{x}_{k+1} | \mathbf{x}_k)P(d\mathbf{x}_k | t_0 > k, \mathcal{Y}_1^k).
 \end{aligned} \tag{23}$$

In (22) and (23), $P_0^{k+1}(d\mathbf{x}_{k+1} | \mathbf{x}_k)$ and $P_1^{k+1}(d\mathbf{x}_{k+1} | \mathbf{x}_k)$ are the Markov transition kernel under the hypothesis that no change has occurred before $k+1$ and change has occurred before k , respectively. $P(d\mathbf{x}_{k+1} | \mathbf{x}_k, t_0 = k+1)$ is the Markov transition kernel under the hypothesis that the change has occurred at $k+1$. For the dynamics in (11), we have $P(d\mathbf{x}_{k+1} | \mathbf{x}_k, t_0 = k+1) = P_0^{k+1}(d\mathbf{x}_{k+1} | \mathbf{x}_k)$.

Equations (19)–(23) show that the statistic T_1^k can be calculated recursively. They also show that in the prediction step of the nonlinear filter at each time instance, only two conditional distributions should be calculated, $P(d\mathbf{x}_{k+1}, t_0 \leq k+1 | \mathcal{Y}_1^k)$ and $P(d\mathbf{x}_{k+1}, t_0 > k+1 | \mathcal{Y}_1^k)$. Therefore, if a PAF method is used to approximate (22) and (23), we only need two sets of particles to approximate these two conditional distributions. In the Bayes' rule update step and the resampling step of the PAF, (20) and (21) are used. One possible way of implementing (19) in (23) using a PAF method is as follows.

In Algorithm 3, the same number of particles is used to find the conditional distribution before and after change. This guarantees that always enough numbers of particles are available for approximating the conditional densities before and after change.

In the remaining sections, we use the introduced statistic T_j^k for detecting a sudden change in the phase measurement of an integrated INS/GPS.

7. INTEGRATED INS/GPS

GPS provides worldwide accurate positioning if four or more satellites are in view of the receiver. Although the satellite

Step (1). Initialization.

Sample $2N$ i.i.d. random vectors $\mathbf{x}_0^1, \dots, \mathbf{x}_0^N$ with the weight W_0^0/N , and $\mathbf{x}_0^{N+1}, \dots, \mathbf{x}_0^{2N}$ with the weight W_0^1/N and $W_0^1 = 1 - W_0^0 = \epsilon$, with the initial distribution $P_0(d\mathbf{x}, t_0 > 0)$. (Since we start with the assumption that no change has happened, ϵ is either 0 or a very small number.)

Step (2). Diffusion.

Find $\hat{\mathbf{x}}_{k+1}^1, \dots, \hat{\mathbf{x}}_{k+1}^N$ from the given $\mathbf{x}_k^1, \dots, \mathbf{x}_k^N$ using the dynamic rule:

$$\mathbf{x}_{k+1} = \mathbf{f}_k^0(\mathbf{x}_k) + \mathbf{G}_k^0(\mathbf{x}_k)\mathbf{v}_k,$$

and find $\hat{\mathbf{x}}_{k+1}^{N+1}, \dots, \hat{\mathbf{x}}_{k+1}^{2N}$ from the given $\mathbf{x}_k^{N+1}, \dots, \mathbf{x}_k^{2N}$ using the dynamic rule:

$$\mathbf{x}_{k+1} = \mathbf{f}_k^1(\mathbf{x}_k) + \mathbf{G}_k^1(\mathbf{x}_k)\mathbf{v}_k.$$

Step (3). Find an estimate for $P(d\mathbf{x}_{k+1} | t_0 > k, \mathcal{Y}_1^k)$ and $P(d\mathbf{x}_{k+1} | t_0 \leq k, \mathcal{Y}_1^k)$ either by using empirical distributions

$$P_{(k+1)^-}^N(d\mathbf{x}_{k+1} | t_0 > k, \mathcal{Y}_1^k) = \frac{1}{N} \sum_{j=1}^N \delta_{\hat{\mathbf{x}}_{k+1}^j}(d\mathbf{x}_{k+1}),$$

$$P_{(k+1)^-}^N(d\mathbf{x}_{k+1} | t_0 \leq k, \mathcal{Y}_1^k) = \frac{1}{N} \sum_{j=N+1}^{2N} \delta_{\hat{\mathbf{x}}_{k+1}^j}(d\mathbf{x}_{k+1}),$$

or by using Step (3) of Algorithm 2. Then

$$\begin{aligned} P_{(k+1)^-}^N(d\mathbf{x}_{k+1}, t_0 > k + 1 | \mathcal{Y}_1^k) \\ &= (1 - q) W_k^0 P_{(k+1)^-}^N(d\mathbf{x}_{k+1} | t_0 > k, \mathcal{Y}_1^k), \\ P_{(k+1)^-}^N(d\mathbf{x}_{k+1}, t_0 \leq k + 1 | \mathcal{Y}_1^k) \\ &= q W_k^0 P_{(k+1)^-}^N(d\mathbf{x}_{k+1} | t_0 > k, \mathcal{Y}_1^k) \\ &\quad + W_k^1 P_{(k+1)^-}^N(d\mathbf{x}_{k+1} | t_0 \leq k, \mathcal{Y}_1^k). \end{aligned}$$

Step (4). Use Bayes' rule in (20) and (21) to calculate $P_{k+1}^N(d\mathbf{x}_{k+1}, t_0 > k + 1 | \mathcal{Y}_1^{k+1})$ and $P_{k+1}^N(d\mathbf{x}_{k+1}, t_0 \leq k + 1 | \mathcal{Y}_1^{k+1})$. Then set $W_{k+1}^0 = \int P_{k+1}^N(d\mathbf{x}_{k+1}, t_0 > k + 1 | \mathcal{Y}_1^{k+1})$ and $W_{k+1}^1 = \int P_{k+1}^N(d\mathbf{x}_{k+1}, t_0 \leq k + 1 | \mathcal{Y}_1^{k+1})$.

Step (5). Resampling.

Sample $\mathbf{x}_{k+1}^1, \dots, \mathbf{x}_{k+1}^N$ according to $P_{k+1}^N(d\mathbf{x}_{k+1}, t_0 > k + 1 | \mathcal{Y}_1^{k+1})/W_{k+1}^0$.

Sample $\mathbf{x}_{k+1}^{N+1}, \dots, \mathbf{x}_{k+1}^{2N}$ according to $P_{k+1}^N(d\mathbf{x}_{k+1}, t_0 \leq k + 1 | \mathcal{Y}_1^{k+1})/W_{k+1}^1$.

Step (6). $k \leftarrow k + 1$; go to Step (2).

ALGORITHM 3: Change detection using particle filtering.

constellation guarantees availability of four or more satellites worldwide, natural or man-made obstacles can block the satellite signals easily. Integrating dead reckoning or INS with GPS [21, 22, 23, 24] is a method to overcome this vulnerability. Here, INS or the dead reckoning provides positioning that is calibrated by the GPS. In this section, we consider the case of an integrated INS/GPS. In [25], we showed that using nonlinear filtering for positioning is essential. We compared the proposed PAF with regular PAF and EKF.

TABLE 1: Definition of the parameters for WGS84 reference frame.

a	6378137.0 m	Semimajor axis
b	6356752.3142 m	Semiminor axis
ω_{ie}	7.292115×10^{-5}	Earth angular velocity
e	$(\sqrt{b(a-b)})/a$	Ellipsoid eccentricity

Using carrier phase measurements enables the differential GPS to reach centimeter-level accuracy. A phase lock loop cannot measure the full-cycle part of the carrier phase. This unmeasured part is called integer ambiguity that requires to be resolved through other means. In this paper, we assume that the integer ambiguity is resolved (see, e.g., [26]). However, the measured phase can experience a sudden change undetected by the phase lock loop. This sudden change is called the cycle slip and if it is undetected by the integrated INS/GPS, it results in an error in the estimated position. We will use the method introduced in this paper to detect such changes.

We consider the observation equation provided by the i th GPS satellite at time k to have the form

$$y_k^i = \rho^i(p_x, p_y, p_z) - \rho^i(b_x, b_y, b_z) + n_i(t, t_0) + c\delta + v_k^i, \quad (24)$$

where $[p_x, p_y, p_z]^T$ and $[b_x, b_y, b_z]^T$ are the rover and (known) base coordinates at time k , respectively, $\rho^i(x_1, x_2, x_3)$ is the distance from point $[x_1, x_2, x_3]^T$ to satellite i , δ is a combination of the receiver clock bias in the base and the rover, c is the speed of light, v_k^i is the measurement noise for the i th satellite signal, t_0 is the unknown moment of the cycle slip, and $n_i(t, t_0) = 0$ for $t < t_0$ and $n_i(t, t_0) = n_i$ for $t \geq t_0$, where n_i is the change in phase measurement due to the cycle slip.

The main goal in the simulations is to detect the change in the phase measurement as soon as it happens.

Here we point out that the nonlinearity in measurement is not solely due to the function ρ . Integrated INS/GPS requires coordinate transformations between INS parameters and GPS parameters, which contributes to the nonlinearity of the measurement.

Parameters of an integrated INS/GPS are expressed in different coordinate systems. The GPS measurements are given in an earth-centered earth-fixed (ECEF) frame [27, 28]. The GPS position is given either in the rectangular coordinate system (see (24)) or in the geodetic coordinate system with the familiar latitude, longitude, and height coordinate vector $[p_\lambda, p_\phi, p_h]^T$. For the latter, the earth's geoid is approximated by an ellipsoid. Table 1 shows the defining parameters for the geoid according to the WGS84 reference frame. The parameters and measurements of INS are given in the local geographical frame or in the body frame system, where the velocity is given by the north-east-down velocity vector $[v_N, v_E, v_D]^T$. The transformation matrix from the body frame to the local geographical frame is given by the matrix R_{b2g} . In this paper, we assume the estimation problem for the gyro measurements is solved, hence R_{b2g} is known.

The GPS clock drift and the INS equations constitute key dynamics in integrated INS/GPS. The INS dynamic equation can be expressed as follows (see [25] for details):

$$d \begin{pmatrix} p_\lambda \\ p_\phi \\ p_h \end{pmatrix} = \begin{pmatrix} \frac{1}{R_\lambda + p_h} & 0 & 0 \\ 0 & \frac{1}{(R_\phi + p_h) \cos(p_\lambda)} & 0 \\ 0 & 0 & -1 \end{pmatrix} \begin{pmatrix} v_N \\ v_E \\ v_D \end{pmatrix} dt,$$

$$d \begin{pmatrix} v_N \\ v_E \\ v_D \end{pmatrix} = \begin{pmatrix} -\frac{v_E^2}{R_\phi + p_h} \tan(p_\lambda) - 2\omega_{ie} \sin(p_\lambda) v_E \\ \frac{v_E v_N}{R_\lambda + p_h} \tan(\lambda) + \omega_{ie} \sin(p_\lambda) v_N \\ \frac{v_E v_D}{R_\phi + p_h} + 2\omega_{ie} \cos(p_\lambda) v_D \\ -\frac{v_N^2}{R_\lambda + p_h} - \frac{v_E^2}{R_\phi + p_h} - 2\omega_{ie} \cos(p_\lambda) v_E \end{pmatrix} dt + R_{b2g} \left(\begin{pmatrix} \tilde{a}_u \\ \tilde{a}_v \\ \tilde{a}_w \end{pmatrix} + \begin{pmatrix} b_u \\ b_v \\ b_w \end{pmatrix} \right) + \begin{pmatrix} 0 \\ 0 \\ g \end{pmatrix} dt + d\mathbf{w}_t^v, \quad (25)$$

where $g = 9.780327 \text{ m/s}^2$ is the gravitational acceleration, $R_\lambda = a(1 - e^2)/(1 - e^2 \sin^2(p_\lambda))^{3/2}$, $R_\phi = a/(1 - e^2 \sin^2(p_\lambda))^{1/2}$, $[\tilde{a}_u, \tilde{a}_v, \tilde{a}_w]^T$ and $[b_u, b_v, b_w]^T$ are the accelerometer measurement and the accelerometer measurement bias, respectively, both expressed in the body frame, and \mathbf{w}^v is a vector-valued Brownian motion process with zero mean and known covariance matrix. The measurement bias $\mathbf{b} = [b_u, b_v, b_w]^T$ has the dynamics

$$d\mathbf{b} = -a_b \mathbf{b} dt + d\mathbf{w}_t^b, \quad (26)$$

where \mathbf{w}_t^b is a vector-valued Brownian motion with zero mean and known covariance matrix, and a_b is a small positive constant. The receiver clock drift δ_t is represented by the integration of an exponentially correlated random process ρ_t [24]:

$$d\rho_t = -\frac{1}{500} \rho_t dt + d\mathbf{w}_t^\rho, \quad (27)$$

$$d\delta_t = \rho_t dt,$$

where \mathbf{w}_t^ρ is a zero-mean Brownian motion process with variance $\sigma_\rho^2 = 10^{-24}$. This dynamic model is typical for a quartz TCXO with frequency drift rate 10^{-9} s/s [24].

8. SIMULATIONS AND RESULTS

The dimension of the dynamical system in the simulation is eleven. The state of the dynamical system \mathbf{x} is given as follows:

$$\mathbf{x} = [p_\lambda, p_\phi, p_h, v_N, v_E, v_D, b_u, b_v, b_w, \rho, \delta]^T. \quad (28)$$

The differential equation describing the dynamics of the system is the combination of the differential equations in (25), (26), and (27). Here we assume that $a_b = 0.001$ and that the covariance matrices for the Brownian motions in the INS dynamic equations Σ_b and Σ_v are diagonal. To be more specific, $\Sigma_b = 10^{-6} I$ and $\Sigma_v = 10^{-4} I$, where I is the identity matrix of the proper size. The observation equation is given in (24), where y_i is one component of the observation vector. The dimension of the observation vector is the same as the number of available satellites. In (24), the observation is given as a function of the position in the ECEF rectangular coordinate system that is then transformed to the geodetic coordinate system [25].

For the simulation, we simply chose an eleven-dimensional Gaussian density for the projection PAF. This choice of density makes the random vector generation easy and computationally affordable. To be able to use the projection PAF, we used the maximum likelihood criteria to estimate the parameters of the Gaussian density before and after Bayes' correction.

We used two Novatel GPS receivers to collect navigation data (April 2, 2000). From this data, we extracted position information for the satellites, the pseudorange, and the carrier phase measurement noise powers for the L1 frequency. From the collected information we generated the pseudorange and the carrier phase data for one static and one moving receiver (base and rover, respectively). We assume that for the carrier phase measurement, the integer ambiguity problem is already solved (see [26] for details). The movement of the INS/GPS platform was simulation based and it was the basis for the measurement data measured by the accelerometers, the gyros, the GPS pseudorange, and the GPS carrier phase data.

As a precursor, we note that in the simulation in [13] we showed that for an integrated INS/GPS when the number of satellites is less than a critical number, projection PAF provides a very accurate estimate of the position, while the position solution given by EKF is unacceptable. In Figures 2 and 3, a comparison of the position estimation error in the rectangular coordinate system for *one typical run* of each method is shown. For that simulation, we assumed that the GPS receiver starts with six satellites. At time $t = 100$, the receiver loses the signal from three satellites, and it gains one satellite signal back at $t = 400$. From these two figures, it is clear that when the number of satellites in view is below a certain number (here four satellites), the EKF is unable to provide a reasonable estimate of the position for the integrated INS/GPS. Since the error of the position estimate of linearization methods is unacceptable even when no change in the phase measurement occurs, using these methods in the presence of an abrupt change is fruitless as well.

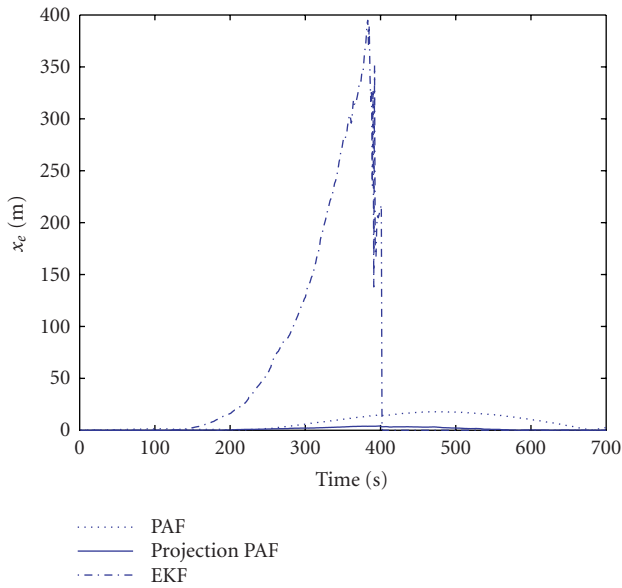


FIGURE 2: Position estimation error (= Euclidean distance between Cartesian position vector and its estimate) for three methods: EKF, PAF, and projection PAF. The system starts with six satellites in view. At $t = 100$ seconds, the signals from three satellites are lost. At $t = 400$ seconds, the system regains the signal from one satellite.

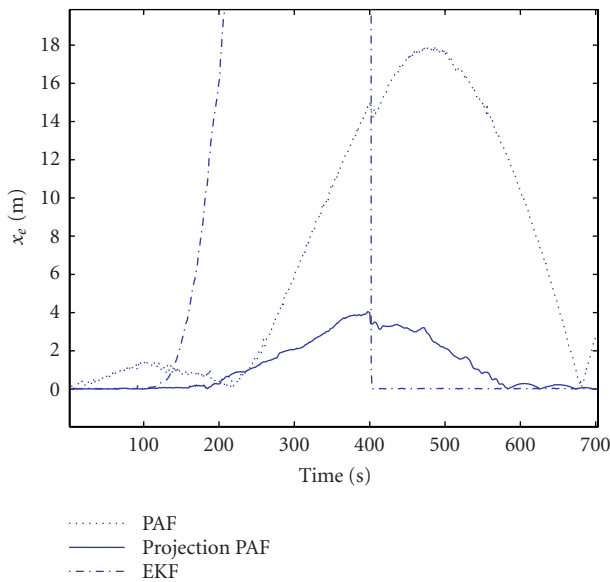


FIGURE 3: Details of Figure 2, where the difference between the projection PAF method and the PAF method is clear.

To apply our method described in (17) for sudden phase change detection in an integrated INS/GPS, we use projection PAF as our nonlinear filtering method. We use the CUSUM algorithm to evaluate the proposed changed detection scheme. We compare the statistic in (17) with that of the CUSUM algorithm. We wish to emphasize that in the example given in this section, we assume that the parameter of change, before and after the change, is known and the only

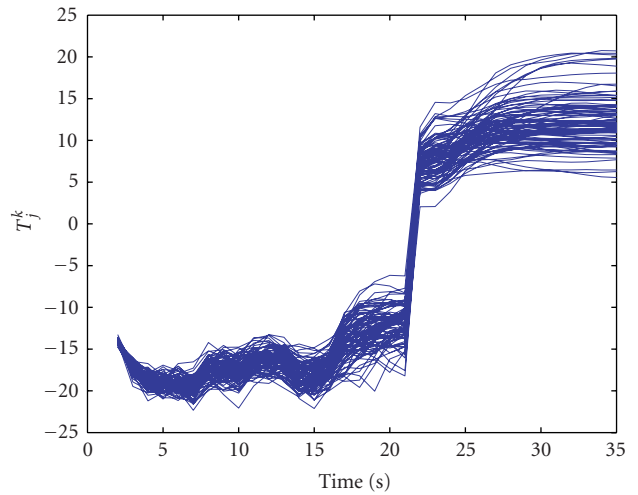


FIGURE 4: The plot of T_j^k with respect to time for 100 independent runs of an INS/GSP system. At time $t = 15$, the receiver loses three satellites. The cycle slip in channel one occurred at $t = 20$ seconds.

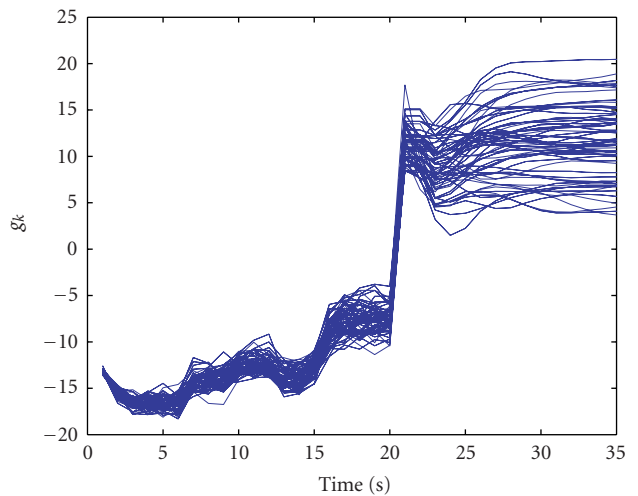


FIGURE 5: The plot of g_k with respect to time for 100 independent runs of an INS/GSP system. g_k is the statistics used in the CUSUM algorithm. At time $t = 15$, the receiver loses three satellites. The cycle slip in channel one occurred at $t = 20$ seconds.

unknown parameter is the change time. In the future, we will address the more general problem of unknown change parameters.

For the simulation in this paper, we assumed that the phase lock loop associated to satellite one experiences a cycle slip at time $t = 20$ and the phase changes suddenly. The size of the change is one cycle. We assumed that the GPS receiver starts with six satellites. At time $t = 15$, the receiver loses three satellites (we eliminate them from the data). We used Algorithm 2 to calculate the statistic in the CUSUM algorithm g_k and Algorithm 3 and projection PAF to calculate T_j^k . In Figure 4, we have plotted T_j^k with respect to time for

100 independent runs. In Figure 5, we have plotted the statistic g_k for the same 100 independent runs. The figures show that there are sudden changes both in T_j^k and g_k when a cycle slip occurs, and this is true for all 100 runs. These figures also show that for this simulation, the performance of the algorithm given in this paper is comparable to the performance of the CUSUM algorithm. This simulation shows that T_j^k can be used successfully to detect the cycle slip with known strength.

9. CONCLUSION AND FUTURE WORK

In this paper, we developed a new method for the detection of abrupt changes for known parameters after change. We showed that unlike the CUSUM algorithm, the statistic in this method can be calculated recursively for nonlinear stochastic systems. In the future, we intend to extend our results to the case where the parameters after change are unknown. The major obstacle in this extension is the complexity of the change detection method.

ACKNOWLEDGMENT

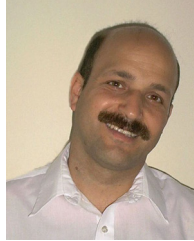
This research was supported in part by Army Research Office under ODDR&E MURI97 Program Grant no. DAAG55-97-1-0114 to the Center for Dynamics and Control of Smart Structures (through Harvard University), under ODDR&E MURI01 Program Grant no. DAAD19-01-1-0465 to the Center for Communicating Networked Control Systems (through Boston University), and by the National Science Foundation Learning and Intelligent Systems Initiative Grant no. CMS9720334.

REFERENCES

- [1] E. S. Page, "Continuous inspection schemes," *Biometrika*, vol. 41, pp. 100–115, 1954.
- [2] M. Basseville and I. V. Nikiforov, *Detection of Abrupt Changes: Theory and Application*, Prentice-Hall, Englewood Cliffs, NJ, USA, 1993.
- [3] G. Lorden, "Procedures for reacting to a change in distribution," *Annals of Mathematical Statistics*, vol. 42, no. 6, pp. 1897–1908, 1971.
- [4] A. S. Willsky and H. L. Jones, "A generalized likelihood ratio approach to the detection and estimation of jumps in linear systems," *IEEE Trans. Automatic Control*, vol. 21, no. 1, pp. 108–112, 1976.
- [5] I. V. Nikiforov, "New optimal approach to global positioning system/differential global positioning system integrity monitoring," *Journal of Guidance, Control and Dynamics*, vol. 19, no. 5, pp. 1023–1033, 1996.
- [6] P. S. Maybeck, *Stochastic Models, Estimation, and Control*, vol. 2, Academic Press, New York, NY, USA, 1982.
- [7] P. Del Moral, "Nonlinear filtering: interacting particle solution," *Markov Processes and Related Fields*, vol. 2, no. 4, pp. 555–580, 1996.
- [8] F. Le Gland, C. Musso, and N. Oudjane, "An analysis of regularized interacting particle methods in nonlinear filtering," in *Proc. IEEE 3rd European Workshop on Computer-Intensive Methods in Control and Signal Processing*, J. Rojicek, M. Valeckova, M. Kárný, and K. Warwick, Eds., vol. 1, pp. 167–174, Prague, Czech Republic, September 1998.
- [9] M. Isard and A. Blake, "Contour tracking by stochastic propagation of conditional density," in *Proc. European Conference on Computer Vision (ECCV '96)*, vol. 1, pp. 343–356, Cambridge, UK, April 1996.
- [10] P. Fearnhead, *Sequential Monte Carlo methods in filter theory*, Ph.D. thesis, Merton College, University of Oxford, Oxford, UK, 1998.
- [11] N. J. Gordon, D. J. Salmond, and A. F. M. Smith, "Novel approach to nonlinear/non-Gaussian Bayesian state estimation," *IEE Proceedings Part F: Radar and Signal Processing*, vol. 140, no. 2, pp. 107–113, 1993.
- [12] G. Kitagawa, "Monte Carlo filter and smoother for non-Gaussian nonlinear state space models," *Journal of Computational and Graphical Statistics*, vol. 5, no. 1, pp. 1–25, 1996.
- [13] B. Azimi-Sadjadi and P. S. Krishnaprasad, "Approximate nonlinear filtering and its applications for an integrated INS/GPS," to appear in *Automatica*.
- [14] A. Doucet, N. de Freitas, and N. Gordon, "An introduction to sequential Monte Carlo methods," in *Sequential Monte Carlo Methods in Practice*, A. Doucet, N. de Freitas, and N. Gordon, Eds., pp. 1–14, Springer, New York, NY, USA, 2001.
- [15] D. Brigo, *Filtering by projection on the manifold of exponential densities*, Ph.D. thesis, Department of Economics and Econometrics, Vrije Universiteit, Amsterdam, The Netherlands, October 1996.
- [16] B. R. Crain, "Exponential models, maximum likelihood estimation, and the Haar condition," *Journal of the American Statistical Association*, vol. 71, no. 355, pp. 737–740, 1976.
- [17] A. Budhiraja and D. Ocone, "Exponential stability in discrete-time filtering for non-ergodic signals," *Stochastic Processes and Their Applications*, vol. 82, no. 2, pp. 245–257, 1999.
- [18] R. Atar and O. Zeitouni, "Exponential stability for nonlinear filtering," *Annales de l'Institut Henri Poincaré. Probabilités et Statistiques*, vol. 33, no. 6, pp. 697–725, 1997.
- [19] R. Atar, "Exponential stability for nonlinear filtering of diffusion processes in a noncompact domain," *The Annals of Probability*, vol. 26, no. 4, pp. 1552–1574, 1998.
- [20] F. LeGland and N. Oudjane, "Stability and approximation of nonlinear filters in the Hilbert metric, and application to particle filters," in *Proc. IEEE 39th Conference on Decision and Control*, vol. 2, pp. 1585–1590, Sydney, Australia, December 2000.
- [21] E. Abbott and D. Powell, "Land-vehicle navigation using GPS," *Proceedings of the IEEE*, vol. 87, no. 1, pp. 145–162, 1999.
- [22] G. Elkaim, M. O'Connor, T. Bell, and B. Parkinson, "System identification of a farm vehicle using carrier-phase differential GPS," in *Proc. of 9th International Technical Meeting of the Satellite Division of the Institute of Navigation (ION GPS-96)*, pp. 485–494, Kansas City, Mo, USA, September 1996.
- [23] R. Zickel and N. Nehemia, "GPS aided dead reckoning navigation," in *Proc. 1994 National Technical Meeting*, pp. 577–586, San Diego, Calif, USA, January 1994.
- [24] H. Carvalho, P. Del Moral, A. Monin, and G. Salut, "Optimal nonlinear filtering in GPS/INS integration," *IEEE Trans. on Aerospace and Electronics Systems*, vol. 33, no. 3, pp. 835–850, 1997.
- [25] B. Azimi-Sadjadi, *Approximate nonlinear filtering with applications to navigation*, Ph.D. thesis, University of Maryland at College Park, College Park, Md, USA, August 2001.
- [26] B. Azimi-Sadjadi and P. S. Krishnaprasad, "Integer ambiguity resolution in GPS using particle filtering," in *Proc. American Control Conference*, vol. 5, pp. 3761–3766, Arlington, Va, USA, June 2001.

- [27] B. Hofmann-Wellenhof, H. Lichtenegger, and J. Collins, *Global Positioning System: Theory and Practice*, Springer, New York, NY, USA, 2nd edition, 1993.
- [28] J. A. Farrell and M. Barth, *The Global Positioning System and Inertial Navigation*, McGraw-Hill, New York, NY, USA, 1998.

Babak Azimi-Sadjadi received his B.S. degree from Sharif University of Technology in 1989, his M.S. degree from Tehran University in 1992, and his Ph.D. degree from the University of Maryland at College Park in 2001, all in electrical engineering. He is currently with the Department of Electrical, Computer, and Systems Engineering, Rensselaer Polytechnic Institute, where he is a Research Assistant Professor. He also has a joint appointment with the Institute for Systems Research, University of Maryland, as a Research Associate. His research interests include nonlinear filtering, networked control systems, and wireless networks.



P. S. Krishnaprasad received the Ph.D. degree from Harvard University in 1977. He was in the faculty of the Systems Engineering Department at Case Western Reserve University from 1977 to 1980. He has been with the University of Maryland since August 1980, where he has held the position of Professor of electrical engineering since 1987, and has a joint appointment with the Institute for Systems Research since 1988.



He is also a member of the Applied Mathematics faculty. He has held visiting positions with Erasmus University (Rotterdam); the Department of Mathematics at University of California, Berkeley; the Department of Mathematics at University of Groningen (The Netherlands); the Mathematical Sciences Institute at Cornell University; and the Mechanical and Aerospace Engineering Department at Princeton University. Krishnaprasad's interests include geometric control theory, filtering and signal processing theory, robotics, acoustics, and biologically-inspired approaches to control, sensing, and computation. He is currently actively engaged in the study of collectives, that is, communicating networked control systems. P. S. Krishnaprasad was elected a Fellow of the IEEE in 1990.

Particle Filtering for Joint Symbol and Code Delay Estimation in DS Spread Spectrum Systems in Multipath Environment

Elena Punskeya

*Signal Processing Laboratory, Department of Engineering, University of Cambridge, Trumpington Street, Cambridge CB2 1PZ, UK
Email: op205@eng.cam.ac.uk*

Arnaud Doucet

*Signal Processing Laboratory, Department of Engineering, University of Cambridge, Trumpington Street, Cambridge CB2 1PZ, UK
Email: ad2@eng.cam.ac.uk*

William J. Fitzgerald

*Signal Processing Laboratory, Department of Engineering, University of Cambridge, Trumpington Street, Cambridge CB2 1PZ, UK
Email: wjf@eng.cam.ac.uk*

Received 19 June 2003; Revised 18 June 2004

We develop a new receiver for joint symbol, channel characteristics, and code delay estimation for DS spread spectrum systems under conditions of multipath fading. The approach is based on particle filtering techniques and combines sequential importance sampling, a selection scheme, and a variance reduction technique. Several algorithms involving both deterministic and randomized schemes are considered and an extensive simulation study is carried out in order to demonstrate the performance of the proposed methods.

Keywords and phrases: direct sequence spread spectrum system, multipath fading, particle filters, signal detection, synchronization.

1. INTRODUCTION

Direct sequence (DS) spread spectrum systems are robust to many channel impairments, allow multiuser CDMA and low-detectability signal transmission, and, therefore, are widely used in different areas of digital communications. Unlike many other communication systems, however, spread spectrum receivers require additional code synchronization, which can be a rather challenging task under conditions of multipath fading, when severe amplitude and phase variations take place.

The problem of joint symbol, delay, and multipath estimation has been addressed in the literature before (see e.g., [1, 2]), and proved to be a difficult one due to its inherited nonlinearity. The previously proposed approaches were mainly based on the use of the extended Kalman filter (EKF). However, many of them concentrated on the channel parameters and delay estimation only; moreover, in a number of

cases, when EKF methods were applied, the estimated parameters were divergent [1]. Joint signal detection and channel estimation was performed using deterministic maximum likelihood (DML) methods [3, 4]. However, since the unknown parameters of interest were assumed deterministic in this case, a serious drawback of DML-type approaches was the phenomenon of error propagation. Later, a stochastic maximum likelihood (ML) approach for the estimation of channel parameters was adopted with consequent symbol detection using Viterbi algorithms [5]. The space-alternating generalized expectation maximization (SAGE) scheme for maximum a posteriori (MAP) estimation was presented in [6].

In this paper, we propose to estimate the channel parameters, code delays, and symbols jointly using particle filtering techniques—a set of powerful and versatile simulation-based methods recently appeared in the literature (see [7] for a survey). The idea is to approximate the posterior distribution of

interest by swarms of N ($N \gg 1$) weighted points in the sample space, called particles, which evolve randomly in time in correlation with each other and either give birth to offspring particles or die according to their ability to represent the different zones of interest of the state space.

The methods have already been successfully applied to problems arising in digital communications, in particular, demodulation in fading channels [7, 8, 9] and detection in synchronous CDMA [10]. In all this work, the unknown fading channel characteristics were integrated out and only the symbols needed to be imputed. The algorithm, thus, made use of the structure of the model, and the unknown state involved discrete parameters only. Later investigation [10, 11], however, revealed some concerns regarding the efficiency of the standard randomized particle filtering techniques in this context. It has been shown that, for a fixed computational complexity, more efficient deterministic schemes could be designed leading to an improved performance of the receiver. We attempt here to study these results further, and compare various randomized and nonrandomized approaches. Itis [12] has recently developed a particle filtering method to address a problem closely related to ours. However, in his approach, the unknown symbol sequence is obtained through a standard algorithm, and only channel parameters and code delays are estimated using particle filtering. The problem we are dealing with is more complex, since it involves both discrete (symbols) and continuous-valued (delays) unknowns. The deterministic particle method, unfortunately, is not applicable directly in this case. However, in view of the recent results, we propose to combine it with sequential importance sampling for the mixed, discrete and continuous-valued parameter case, followed by an appropriate selection procedure. The resulting algorithm explores the state space in a more systematic way at little or no extra cost in comparison with the standard particle filtering, employing a suboptimal importance distribution. We develop and test this approach against other deterministic and stochastic schemes, and demonstrate its performance by means of an extensive simulation study.

The remainder of the paper is organized as follows. The model specifications and estimation objectives are stated in Section 2. In Section 3, a particle filtering method is developed for joint symbol/channel coefficients/code delay estimation. This section also introduces and reviews several alternative deterministic and stochastic schemes, with simulation results and comparisons presented in Section 4. Some conclusions are drawn at the end of the paper.

2. PROBLEM STATEMENT AND ESTIMATION OBJECTIVES

Transmitted waveform

We denote, for any generic sequence κ_t , $\kappa_{i:j} \triangleq (\kappa_i, \kappa_{i+1}, \dots, \kappa_j)^T$, and let d_n be the n th information symbol, and $s_{\text{tx}}(\tau)$ the corresponding analog bandpass spread spectrum signal

waveform transmitted in the symbol interval of duration T :

$$s_{\text{tx}}(\tau) = \text{Re} [s_n(d_{1:n})u(\tau) \exp(j2\pi f_0\tau)] \quad (1)$$

for $(n-1)T < \tau \leq nT$,

where $s_n(\cdot)$ performs the mapping from the digital sequence to waveforms and corresponds to the modulation technique employed, f_0 denotes the carrier frequency, and $u(\tau)$ is a wideband pseudonoise (PN) waveform defined by

$$u(\tau) = \sum_{k=1}^K c_k \eta(\tau - kT_c). \quad (2)$$

Here, $c_{1:K}$ is a spreading code sequence consisting of K chips (with values $\{\pm 1\}$) per symbol, $\eta(\tau - kT_c)$ is a rectangular pulse of unit height and duration T_c , transmitted at $(k-1)T_c < \tau \leq kT_c$, and T_c is the chip interval satisfying the relation $T_c = T/K$.

Channel model

The signal is passed through a noisy multipath fading channel which causes random amplitude and phase variations on the signal. The channel can be represented by a time-varying tapped-delayed line with taps spaced T_s seconds apart, where T_s is the Nyquist sampling rate for the transmitted waveform; $T_s = T_c/2$ due to the PN bandwidth being approximately $1/T_c$ (see sampling theorem [13]). The equivalent discrete-time impulse response of the channel is given by

$$h_t = \sum_{l=0}^{L-1} f_t^{(l)} \delta_{t,l}, \quad (3)$$

where t is a discrete time index, $t = 1, 2, \dots$. By L we understand the maximum number of paths (nonzero coefficients of h_t) of the channel [5], $f_t^{(l)}$ are the complex-valued time-varying multipath coefficients arranged into the vector \mathbf{f}_t , and $\delta_{t,l}$ denotes the Kronecker delta, which is 1 if $t = l$, and 0 otherwise.

We assume here that the channel coefficients \mathbf{f}_t and code delay θ_t propagate according to the first-order autoregressive (AR) model:

$$\mathbf{f}_t = \mathbf{A}\mathbf{f}_{t-1} + \mathbf{B}\mathbf{v}_t, \quad \mathbf{v}_t \stackrel{\text{i.i.d.}}{\sim} \mathcal{N}(\mathbf{0}, \mathbf{I}_L), \quad (4)$$

$$\theta_t = \gamma\theta_{t-1} + \sigma_\theta \vartheta_t, \quad \vartheta_t \stackrel{\text{i.i.d.}}{\sim} \mathcal{N}(0, 1), \quad (5)$$

which corresponds to a Rayleigh uncorrelated scattering channel model; here $\mathbf{A} \triangleq \text{diag}(\alpha^{(0)}, \dots, \alpha^{(L-1)})$, $\mathbf{B} \triangleq \text{diag}(\sigma_f^{(0)}, \dots, \sigma_f^{(L-1)})$, with $\sigma_f^{(l)}$ being the standard deviation, and $\alpha^{(l)}$ accounting for the Doppler spread (see [2, 14] for details and discussion on the use of higher-order AR). In this paper, matrices \mathbf{A} , \mathbf{B} , and parameters γ and σ_θ are assumed known. Directions on the choice of these parameters are given in [2, 14].

Received signal

The complex output of the channel sampled at the Nyquist rate, (in which case, $t = 2K(n-1) + 1, \dots, 2Kn$ samples correspond to the n th symbol transmitted, that is, $d_n \leftrightarrow y_{2K(n-1)+1:2Kn}$) can, thus, be expressed as

$$y_t = \mathbf{C}(d_{1:n}, \theta_t) + \sigma \epsilon_t, \quad \epsilon_t \stackrel{\text{i.i.d.}}{\sim} \mathcal{N}_c(0, 1), \quad (6)$$

where $\mathbf{C}(d_{1:n}, \theta_t) = \sum_{l=0}^{L-1} f_t^{(l)} s_{rx}((t-l)T_s - \theta_t)$ and σ^2 is the noise variance.¹ The noise sequences ϑ_t , ϵ_t , and $v_t^{(l)}$, $l = 0, \dots, L-1$ are assumed mutually independent and independent of the initial states $\mathbf{f}_0 \sim \mathcal{N}_c(\hat{\mathbf{f}}_0, \Sigma_{\mathbf{f},0})$, $\theta_0 \sim \mathcal{N}(\hat{\theta}_0, \Sigma_{\theta,0})$. The received waveform $s_{rx}(\tau)$ is obtained after ideal lowpass filtering of rectangular pulses and is given by [2]

$$\begin{aligned} s_{rx}(\tau) &= s_n(d_{1:n}) \sum_{k=1}^K c_k \frac{1}{\pi} \left[\text{Si} \left(2\pi \frac{\tau - (k-1)T_c}{T_c} \right) - \text{Si} \left(2\pi \frac{\tau - kT_c}{T_c} \right) \right], \\ &\quad \text{for } (n-1)T < \tau \leq nT, \end{aligned} \quad (7)$$

where

$$\text{Si}(\phi) = \int_0^\phi \frac{\sin(\varphi)}{\varphi} d\varphi. \quad (8)$$

Estimation objectives

The symbols d_n , which are assumed i.i.d., the channel characteristics \mathbf{f}_t , and the code delay θ_t are unknown for $n, t > 0$. Our aim is to obtain sequentially in time an estimate of the joint posterior probability density of these parameters $p(d_{1:n}, \mathbf{f}_{0:2Kn}, \theta_{0:2Kn} | y_{1:2Kn})$, and some of its characteristics, such as the MAP estimates of the symbols

$$\hat{d}_{1:n} = \arg \max_{d_{1:n}} p(d_{1:n} | y_{1:2Kn}), \quad (9)$$

and the minimum mean square error (MMSE) estimates of the channel characteristics $\mathbb{E}(\mathbf{f}_{0:2Kn} | y_{1:2Kn})$ and the delays $\mathbb{E}(\theta_{0:2Kn} | y_{1:2Kn})$. This problem, unfortunately, does not admit any analytical solution and, thus, approximate methods must be employed. One of the methods that has proved to be useful in practice is particle filtering, and, in the next section, we propose a receiver based on the use of these techniques.

3. PARTICLE FILTERING RECEIVER

Particle filtering receivers have already been designed in [7, 8, 9], although for a much simpler case including symbols estimation only. The problem considered here is more complicated since an additional continuous parameter is involved, and, in this section, the particle filtering algorithm

for the joint estimation of all unknown parameters is detailed. We begin our treatment with incorporating a variance reduction technique, namely, Rao-Blackwellisation, and then proceed with the derivation of the particle filtering equations for the estimation of the required posterior distribution. The alternative deterministic and stochastic approaches are considered at the end of the section.

3.1. Rao-Blackwellisation

In this paper, we follow a Bayesian approach and, given the measurements $y_{1:2Kn}$, base our inference on the joint posterior distribution $p(d_{1:n}, \mathbf{f}_{0:2Kn}, \theta_{0:2Kn} | y_{1:2Kn})$. A straightforward application of particle filtering would, thus, focus on the estimation of this joint probability distribution, and, consequently, obtain the estimates of $d_{1:n}$, $\mathbf{f}_{0:2Kn}$, and $\theta_{0:2Kn}$ sequentially in time. It is beneficial, however, to improve the standard approach by making most of the structure of the model and applying the variance reduction techniques.

Indeed, similar to [7, 8, 9, 15], the problem of estimating $p(d_{1:n}, \mathbf{f}_{0:2Kn}, \theta_{0:2Kn} | y_{1:2Kn})$ can be reduced to a one of sampling from a lower-dimensional posterior $p(d_{1:n}, \theta_{0:2Kn} | y_{1:2Kn})$. If the approximation of $p(d_{1:n}, \theta_{0:2Kn} | y_{1:2Kn})$ could be obtained, say, via particle filtering:

$$\begin{aligned} \hat{p}_N(d_{1:n}, \theta_{0:2Kn} | y_{1:2Kn}) &= \sum_{i=1}^N \tilde{w}_n^{(i)} \delta(\{d_{1:n}, \theta_{0:2Kn}\} - \{d_{1:n}^{(i)}, \theta_{0:2Kn}^{(i)}\}), \end{aligned} \quad (10)$$

one could compute the probability density $p(\mathbf{f}_{0:2Kn} | y_{1:2Kn}, d_{1:n}, \theta_{0:2Kn})$ using the Kalman filter associated with (4) and (6). As a result, the posterior $p(\mathbf{f}_{0:2Kn} | y_{1:2Kn})$ could be approximated by a random mixture of Gaussians

$$\begin{aligned} \hat{p}_N(\mathbf{f}_{0:2Kn} | y_{1:2Kn}) &= \int_{\theta_{0:2Kn}} \sum_{d_{1:n}} p(\mathbf{f}_{0:2Kn} | y_{1:2Kn}, d_{1:n}, \theta_{0:2Kn}) \\ &\quad \times \hat{p}_N(d_{1:n}, \theta_{0:2Kn} | y_{1:2Kn}) d\theta_{0:2Kn} \\ &= \sum_{i=1}^N \tilde{w}_n^{(i)} p(\mathbf{f}_{0:2Kn} | y_{1:2Kn}, d_{1:n}^{(i)}, \theta_{0:2Kn}^{(i)}) \end{aligned} \quad (11)$$

leading to lower variance of the estimates and, therefore, increased algorithm efficiency [15].

Strictly speaking, we are interested in estimating the information symbols only with the tracking of the channel being naturally incorporated into the proposed algorithm. However, following this approach, the MMSE (conditional mean) estimates of fading coefficients can, of course, be obtained if necessary as follows:

$$\begin{aligned} \hat{\mathbb{E}}_N[\mathbf{f}_{2K(n-1)+1:2Kn} | y_{1:2Kn}] &= \int \mathbf{f}_{2K(n-1)+1:2Kn} \hat{p}_N(\mathbf{f}_{0:2Kn} | y_{1:2Kn}) d\mathbf{f}_{0:2Kn} \\ &= \sum_{i=1}^N \tilde{w}_n^{(i)} \mathbb{E}[\mathbf{f}_{2K(n-1)+1:2Kn} | y_{1:2Kn}, d_{1:n}^{(i)}, \theta_{0:2Kn}^{(i)}], \end{aligned} \quad (12)$$

¹The case of non-Gaussian noise can be easily treated using the techniques presented in [9].

with $\mathbb{E}[\mathbf{f}_{2K(n-1)+1:2Kn} | \mathbf{y}_{1:2Kn}, \mathbf{d}_{1:n}^{(i)}, \boldsymbol{\theta}_{0:2Kn}^{(i)}]$ being computed by the Kalman filter, with $2K$ steps required for each symbol transmitted.

3.2. Particle filtering algorithm

We can now proceed with the estimation of $p(\mathbf{d}_{1:n}, \boldsymbol{\theta}_{0:2Kn} | \mathbf{y}_{1:2Kn})$ using particle filtering techniques. The method is based on the following remark. Suppose N particles $\{\mathbf{d}_{1:n}^{(i)}, \boldsymbol{\theta}_{0:n}^{(i)}\}_{i=1}^N$, where $\boldsymbol{\theta}_n$ denotes

$$\boldsymbol{\theta}_n = \boldsymbol{\theta}_{2K(n-1)+1:2Kn} \quad \text{for } n = 1, 2, \dots, \quad (13)$$

can be easily simulated according to an arbitrary convenient importance distribution $\pi(\mathbf{d}_{1:n}, \boldsymbol{\theta}_{0:n} | \mathbf{y}_{1:n})$ (such that $p(\mathbf{d}_{1:n}, \boldsymbol{\theta}_{0:n} | \mathbf{y}_{1:n}) > 0$ implies $\pi(\mathbf{d}_{1:n}, \boldsymbol{\theta}_{0:n} | \mathbf{y}_{1:n}) > 0$).

Then, using the importance sampling identity, an estimate of $p(\mathbf{d}_{1:n}, \boldsymbol{\theta}_{0:n} | \mathbf{y}_{1:n})$ is given by the following point mass approximation:

$$\hat{p}_N(\mathbf{d}_{1:n}, \boldsymbol{\theta}_{0:n} | \mathbf{y}_{1:n}) = \sum_{i=1}^N \tilde{w}_n^{(i)} \delta(\{\mathbf{d}_{1:n}, \boldsymbol{\theta}_{0:n}\} - \{\mathbf{d}_{1:n}^{(i)}, \boldsymbol{\theta}_{0:n}^{(i)}\}), \quad (14)$$

where $\tilde{w}_n^{(i)}$ are the so-called normalized *importance weights*,

$$\tilde{w}_n^{(i)} = \frac{w_n^{(i)}}{\sum_{j=1}^N w_n^{(j)}}, \quad w_n^{(i)} \propto \frac{p(\mathbf{d}_{1:n}, \boldsymbol{\theta}_{0:n}^{(i)} | \mathbf{y}_{1:n})}{\pi(\mathbf{d}_{1:n}, \boldsymbol{\theta}_{0:n}^{(i)} | \mathbf{y}_{1:n})}, \quad (15)$$

and \mathbf{y}_n denotes

$$\mathbf{y}_n = \mathbf{y}_{2K(n-1)+1:2Kn} \quad (16)$$

for $n = 1, 2, \dots$. The distribution $\pi(\mathbf{d}_{1:n}, \boldsymbol{\theta}_{0:n}^{(i)} | \mathbf{y}_{1:n})$ has to admit $\pi(\mathbf{d}_{1:n-1}, \boldsymbol{\theta}_{0:n-1}^{(i)} | \mathbf{y}_{1:n-1})$ as a marginal distribution so that one could propagate this estimate sequentially in time without subsequently modifying the past simulated trajectories. The weights $w_n^{(i)}$ could also be updated online in this case:

$$w_n^{(i)} \propto w_{n-1}^{(i)} p(\mathbf{y}_n | \mathbf{d}_{1:n}, \boldsymbol{\theta}_{1:n}^{(i)}, \mathbf{y}_{1:n-1}) \times \frac{p(\mathbf{d}_n^{(i)}, \boldsymbol{\theta}_n^{(i)} | \mathbf{d}_{1:n-1}, \boldsymbol{\theta}_{n-1}^{(i)})}{\pi(\mathbf{d}_n^{(i)}, \boldsymbol{\theta}_n^{(i)} | \mathbf{d}_{1:n-1}, \boldsymbol{\theta}_{n-1}^{(i)}, \mathbf{y}_{1:n})}. \quad (17)$$

The sequential importance sampling described above is combined with a selection procedure when the effective sample size \widehat{N}_{eff}

$$\widehat{N}_{\text{eff}} = \left[\sum_{i=1}^N (\tilde{w}_n^{(i)})^2 \right]^{-1} \quad (18)$$

falls below some fraction of N , say N_{thres} (see [15] for details). This helps to avoid the degeneracy of the algorithm by discarding particles with low normalized importance weights and multiplying those with high ones.

Given for the $(n-1)$ th symbol N particles $\{\mathbf{d}_{1:n-1}^{(i)}, \boldsymbol{\theta}_{0:n-1}^{(i)}\}_{i=1}^N$ distributed approximately according to $p(\mathbf{d}_{1:n-1}, \boldsymbol{\theta}_{0:n-1} | \mathbf{y}_{1:n-1})$, the general particle filtering receiver, proceeds as in Algorithm 1.

Sequential importance sampling step

For $i = 1, \dots, N$, sample

$$(\tilde{\mathbf{d}}_n^{(i)}, \tilde{\boldsymbol{\theta}}_n^{(i)}) \sim \pi(\mathbf{d}_n, \boldsymbol{\theta}_n | \mathbf{d}_{1:n-1}, \boldsymbol{\theta}_{0:n-1}, \mathbf{y}_{1:n}).$$

For $i = 1, \dots, N$, evaluate the importance weights $w_n^{(i)}$ up to a normalizing constant.

For $i = 1, \dots, N$, normalize $w_n^{(i)}$ to obtain $\tilde{w}_n^{(i)}$.

Selection step

If $\widehat{N}_{\text{eff}} < N_{\text{thres}}$, multiply/discard particles

$$\{\tilde{\mathbf{d}}_n^{(i)}, \tilde{\boldsymbol{\theta}}_n^{(i)}\}_{i=1}^N \text{ with respect to high/low}$$

$\tilde{w}_n^{(i)}$ to obtain N unweighted particles

$$\{\mathbf{d}_{1:n}^{(i)}, \boldsymbol{\theta}_{1:n}^{(i)}\}_{i=1}^N.$$

ALGORITHM 1: Particle filtering algorithm.

For $i = 1, \dots, N$,

sample $\tilde{\mathbf{d}}_n^{(i)} \sim p(\mathbf{d}_n)$, set $w_n^{(i)} = 1$.

For $t = 2K(n-1) + 1, \dots, 2Kn$,

sample $\tilde{\boldsymbol{\theta}}_t^{(i)} \sim p(\boldsymbol{\theta}_t | \boldsymbol{\theta}_{t-1}^{(i)})$,

perform one-step Kalman filter update

$$(w_n^{(i)} = w_n^{(i)} p(\mathbf{y}_t | \mathbf{d}_{1:n}, \boldsymbol{\theta}_{0:t-1}, \tilde{\boldsymbol{\theta}}_t^{(i)}, \mathbf{y}_{1:t-1})).$$

For $i = 1, \dots, N$, normalize $w_n^{(i)}$ to obtain $\tilde{w}_n^{(i)}$.

ALGORITHM 2: Sequential importance sampling (prior as importance distribution).

3.3. Implementation issues

The choice of importance distribution and selection scheme is discussed in [16]; depending on these choices, the computational complexity of the algorithm varies.

3.3.1. Importance density

Prior density

The simplest solution is to take the prior as an importance distribution, that is,

$$\begin{aligned} \pi(\mathbf{d}_n, \boldsymbol{\theta}_n | \mathbf{d}_{1:n-1}, \boldsymbol{\theta}_{0:n-1}, \mathbf{y}_{1:n}) &= p(\mathbf{d}_n) p(\boldsymbol{\theta}_n | \boldsymbol{\theta}_{n-1}) \\ &= p(\mathbf{d}_n) \prod_{t=2K(n-1)+1}^{2Kn} p(\boldsymbol{\theta}_t | \boldsymbol{\theta}_{t-1}), \end{aligned} \quad (19)$$

then w_n becomes

$$\begin{aligned} w_n &\propto p(\mathbf{y}_n | \mathbf{y}_{1:n-1}, \mathbf{d}_{1:n}, \boldsymbol{\theta}_{0:n}) \\ &= \prod_{t=2K(n-1)+1}^{2Kn} p(\mathbf{y}_t | \mathbf{d}_{1:n}, \boldsymbol{\theta}_{0:t}, \mathbf{y}_{1:t-1}), \end{aligned} \quad (20)$$

and requires evaluation of $2K$ one-step Kalman filter updates for each symbol as shown in Algorithm 2.

If K is long, it is useful to resample the particles at intermediate steps between $t = 2K(n-1) + 1$ and $t = 2Kn$. One can also use Markov chain Monte Carlo (MCMC) steps to rejuvenate the particles and in particular \mathbf{d}_n .

Suboptimal importance density

Of course, using the prior distribution in our case can be inefficient, as no information carried by the observations is used to explore the state space. The optimal choice, in a sense of minimizing the conditional variance of the importance weights [15], would consist of taking

$$\pi(d_n, \theta_n | d_{1:n-1}, \theta_{0:n-1}, \mathbf{y}_{1:n}) = p(d_n, \theta_n | d_{1:n-1}, \theta_{0:n-1}, \mathbf{y}_{1:n}), \quad (21)$$

as an importance density. From Bayes' rule, $p(d_n, \theta_n | d_{1:n-1}, \theta_{0:n-1}, \mathbf{y}_{1:n})$ may be expressed as

$$\begin{aligned} p(d_n, \theta_n | d_{1:n-1}, \theta_{0:n-1}, \mathbf{y}_{1:n}) \\ = \frac{p(\mathbf{y}_n | \mathbf{y}_{1:n-1}, d_{1:n-1}, d_n, \theta_{0:n-1}, \theta_n) p(d_n) p(\theta_n | \theta_{n-1})}{p(\mathbf{y}_n | \mathbf{y}_{1:n-1}, d_{1:n-1}, \theta_{0:n-1})}, \end{aligned} \quad (22)$$

in which case,

$$\begin{aligned} w_n &= p(\mathbf{y}_n | \mathbf{y}_{1:n-1}, d_{1:n-1}, \theta_{0:n-1}) \\ &= \int_{\check{\theta}_n} \sum_{m=1}^M [p(\mathbf{y}_n | \mathbf{y}_{1:n-1}, d_{1:n-1}, d_n = m, \theta_{0:n-1}, \check{\theta}_n) \\ &\quad \times p(d_n = m) p(\check{\theta}_n | \theta_{n-1}) d\check{\theta}_n] \end{aligned} \quad (23)$$

cannot be computed analytically. Our aim then is to develop a suboptimal importance density "closest" to $p(d_n, \theta_n | d_{1:n-1}, \theta_{0:n-1}, \mathbf{y}_{1:n})$.

The probability density $p(d_n, \theta_n | d_{1:n-1}, \theta_{0:n-1}, \mathbf{y}_{1:n})$ can be factorized as

$$\begin{aligned} p(d_n, \theta_n | d_{1:n-1}, \theta_{0:n-1}, \mathbf{y}_{1:n}) \\ = p(d_n | d_{1:n-1}, \theta_{0:n}, \mathbf{y}_{1:n}) p(\theta_n | d_{1:n-1}, \theta_{0:n-1}, \mathbf{y}_{1:n}), \end{aligned} \quad (24)$$

where $p(d_n | d_{1:n-1}, \theta_{0:n}, \mathbf{y}_{1:n})$ would be an optimal importance function if θ_n were fixed given by

$$p(d_n | d_{1:n-1}, \theta_{0:n}, \mathbf{y}_{1:n}) = \frac{p(\mathbf{y}_n | \mathbf{y}_{1:n-1}, d_{1:n-1}, d_n, \theta_{0:n}) p(d_n)}{p(\mathbf{y}_n | \mathbf{y}_{1:n-1}, d_{1:n-1}, \theta_{0:n})}. \quad (25)$$

The second term in (24), $p(\theta_n | d_{1:n-1}, \theta_{0:n-1}, \mathbf{y}_{1:n})$, unfortunately presents a problem since the integral in (23) cannot be evaluated in closed form. As a solution, we propose to use the prior density $p(\theta_n | \theta_{n-1})$ instead of $p(\theta_n | d_{1:n-1}, \theta_{0:n-1}, \mathbf{y}_{1:n})$ and, thus, employ the following suboptimal importance function (see [17] for a similar approach developed independently):

$$\begin{aligned} \pi(d_n, \theta_n | d_{1:n-1}, \theta_{0:n-1}, \mathbf{y}_{1:n}) \\ = p(d_n | d_{1:n-1}, \theta_{0:n}, \mathbf{y}_{1:n}) p(\theta_n | \theta_{n-1}). \end{aligned} \quad (26)$$

For $i = 1, \dots, N$,
 For $m = 1, \dots, M$, $w_n^{(i,m)} = 1$,
 For $t = 2Kn + 1, \dots, 2K(n+1)$,
 sample $\check{\theta}_t^{(i)} \sim p(\theta_t | \theta_{t-1}^{(i)})$,
 for $m = 1, \dots, M$, perform one-step Kalman
 filter update
 $(w_n^{(i,m)} = w_n^{(i,m)} p(y_t | d_{1:n-1}^{(i)}, d_n = m, \theta_{0:t-1}^{(i)},$
 $\check{\theta}_t^{(i)}, \mathbf{y}_{1:t-1}^{(i)}).$
 Evaluate the importance weight $w_n^{(i)}$ up to a
 normalizing constant:

$$w_n^{(i)} \propto \sum_{m=1}^M w_n^{(i,m)} p(d_n = m).$$

 For $i = 1, \dots, N$, normalize $w_n^{(i)}$ to obtain $\tilde{w}_n^{(i)}$.

ALGORITHM 3: Evaluation of importance weights (suboptimal importance distribution).

The importance weights in this case can be calculated as

$$\begin{aligned} w_n &\propto p(\mathbf{y}_n | \mathbf{y}_{1:n-1}, d_{1:n-1}, \theta_{0:n}) \\ &= \sum_{m=1}^M p(\mathbf{y}_n | \mathbf{y}_{1:n-1}, d_{1:n-1}, d_n = m, \theta_{0:n-1}, \theta_n) p(d_n = m), \end{aligned} \quad (27)$$

where θ_n is drawn from the prior Gaussian distribution with mean $\gamma\theta_{n-1}$ and variance σ_θ^2 :

$$\theta_n^{(i)} \sim \mathcal{N}(\gamma\theta_{n-1}^{(i)}, \sigma_\theta^2) \quad \text{for } i = 1, \dots, N. \quad (28)$$

The importance weight $w_n^{(i)}$ in (27) does not actually depend on $d_n^{(i)}$, and the weights evaluation and selection steps can be done prior to the sampling of $d_n^{(i)}$ as in Algorithm 3.

For each symbol detection, this procedure requires the evaluation of the M $2K$ -step-ahead Kalman filters, which is quite computationally expensive. Further research should, therefore, concentrate on development of other more efficient suboptimal importance distributions on a case by case basis.

3.3.2. Selection

As far as the selection step is concerned, a stratified sampling scheme [18] is employed in this paper since it has the minimum variance one can achieve in the class of unbiased schemes [19]. The algorithm is based on generating N points equally spaced in the interval $[0, 1]$, with the number of offspring N_i for each particle being equal to the number of points lying between the partial sums of weights q_{i-1} and q_i , where $q_i = \sum_{j=1}^i \tilde{w}_t^{(j)}$. The procedure can be implemented in $O(N)$ operations.

3.4. Deterministic particle filter

The use of the suboptimal importance distribution described in Section 3.3.1 increases the efficiency of the algorithm in comparison with the standard approach using the prior.

However, as shown in [11], if one already opts for (27), and all the calculations have to be performed anyway, it might be better to base our approximation of $p(d_{1:n}, \theta_{0:n} | \mathbf{y}_{1:n})$ directly on

$$\begin{aligned} & \hat{p}_{N \times M}(d_{1:n}, \theta_{0:n} | \mathbf{y}_{1:n}) \\ &= \sum_{i=1}^N \sum_{m=1}^M \tilde{w}_n^{(i,m)} \delta(\{d_{1:n}, \theta_{0:n}\} - \{d_{1:n-1}^{(i)}, d_n = m, \theta_{0:n-1}^{(i)}, \theta_n^{(i)}\}), \end{aligned} \quad (29)$$

where corresponding weights $\tilde{w}_n^{(i,m)}$ are equal to

$$w_n^{(i,m)} \propto \tilde{w}_{n-1}^{(i)} p(\mathbf{y}_n | \mathbf{y}_{1:n-1}, d_{1:n-1}^{(i)}, d_n = m, \theta_{0:n-1}^{(i)}, \theta_n^{(i)}) p(d_n = m), \quad (30)$$

and $\theta_n^{(i)}$ is still drawn from its prior (28). Indeed, all possible “extensions” of the existing state sequences at each step n are considered in this case, and one does not discard unnecessarily any information by selecting randomly one path out of the M available. In the above expression, $\tilde{w}_{n-1}^{(i)}$ is the weight of the “parent” particle, which has M “offspring” instead of the usual one, resulting in a total number of $N \times M$ particles at each stage. This number increases exponentially with time, and, therefore, a selection procedure has to be employed at each step n .

The simplest way to perform such selection is just to choose the N most likely offspring and discard the others (as, e.g., in [20]). The superiority of this approach over other methods in the fully discrete framework is shown in [10, 11]. A more complicated procedure involves preserving the particles with high weights and resampling the ones with low weights, thus reducing their total number to N . An algorithm of this type is presented in [21] but other selection schemes can be designed. Contrary to the case involving the discrete parameters only, in this scenario, a resampling scheme with replacement could be employed, since $\theta_n^{(i)}$ is chosen randomly. Therefore, stratified resampling could be used in order to select N particles from $N \times M$ particles available.

Whether we choose to preserve the most likely particles, employ the selection scheme proposed in [21], or stratified resampling, the computational load of the resulting algorithms at each time step n is that of $N \times M \times 2K$ Kalman filters, and the selection step in the first two cases is implemented in $O(N \times M \log N \times M)$ operations. Of course, if M is large, which is the case in many applications, all these methods are too computationally expensive to be used, and one should employ a standard particle filter.

4. SIMULATION RESULTS

In the following experiments the bit error rate (BER) and the tracking delay error ($\theta_t - \hat{\theta}_t$) were evaluated by means of computer simulations. Gray-encoded M -ary differential phase shift keyed (MDPSK) signals were employed, with mapping function

$$s_n = \exp(j\phi_n), \quad \phi_n = \sum_{j=1}^n \sum_{m=1}^M \frac{2\pi m}{M} \delta(d_j - d_m). \quad (31)$$

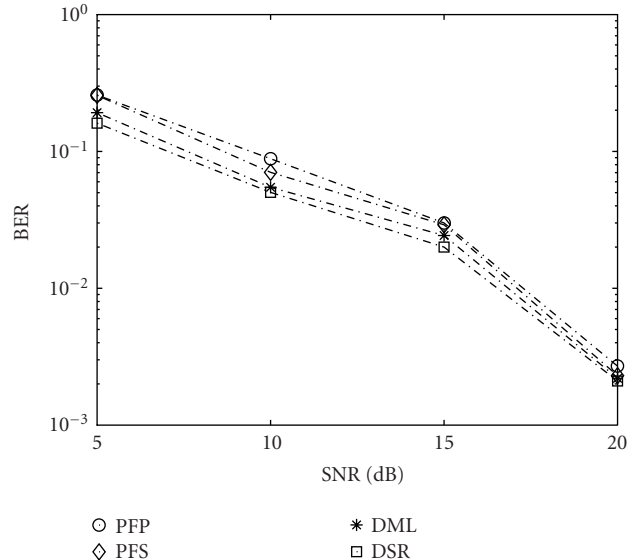


FIGURE 1: Bit error rate performance (4DPSK signal, $N = 100$).

In order to assess the performance of the proposed approaches we first applied them to a simpler case of synchronization in flat fading conditions, $L = 1$, for a system with no spectrum spreading employed, $c_1 = 1$, $K = 1$. In the first experiment, 4DPSK signals were considered with the average signal-to-noise ratio (SNR) varying from 5 to 20 dB. The AR coefficients for the channel, (4), were set to $\alpha^{(0)} = 0.999$, $\sigma_f^{(0)} = 0.01$, and the delay model parameters in (5) were chosen to be the same, $\gamma = 0.999$ and $\sigma_\theta = 0.01$. The BER obtained by the particle filtering receiver employing prior (PFP) and suboptimal (PFS) importance distributions, and the deterministic receiver preserving N most likely particles (DML) and using stratified resampling (DSR) is presented in Figure 1. The marginal maximum a posteriori estimate (MMAP)

$$\hat{d}_n = \arg \max_{d_n} p(d_n | \mathbf{y}_{1:2Kn}) \quad (32)$$

was employed to obtain the symbols. The number of particles used in these algorithms was equal to $N = 100$, and little or no improvement in BER was gained by increasing this number for deterministic schemes. For the randomized approaches, the number of particles required to achieve the BER of DSR algorithm was equal to $N = 1200$. In Figure 2, the mean square delay error (MSE) is presented as a function of the number of particles N for SNR = 10 dB:

$$\hat{\theta}_{\text{MSE}} = \frac{1}{2KL_d} \sum_{n=1}^{2KL_d} (\theta_n - \hat{\theta}_n)^2, \quad (33)$$

where L_d is a length of the symbol sequence, $L_d = 1000$. The results for the different SNRs are given in Figure 3. As one can see, the deterministic particle filter with stratified resampling slightly outperforms the receiver selecting most likely

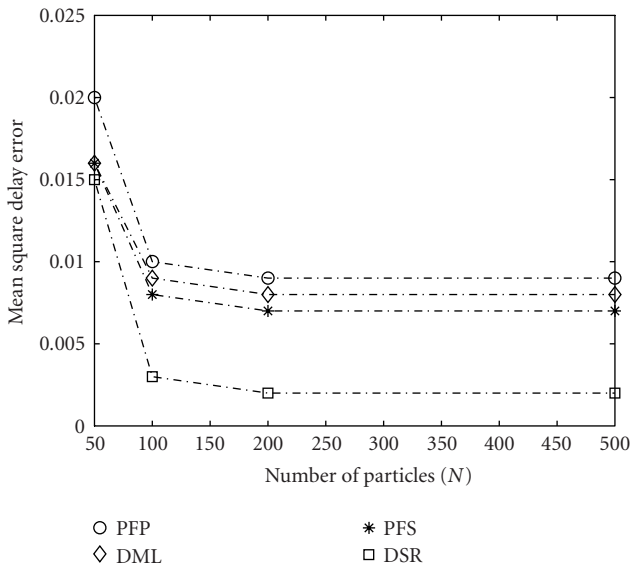


FIGURE 2: Mean square delay error for the different number of particles (4DPSK, SNR = 10 dB).

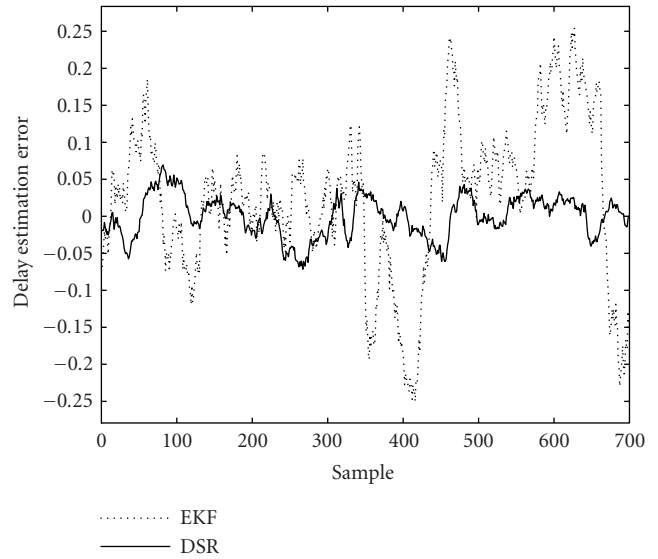


FIGURE 4: The error in delay estimation.

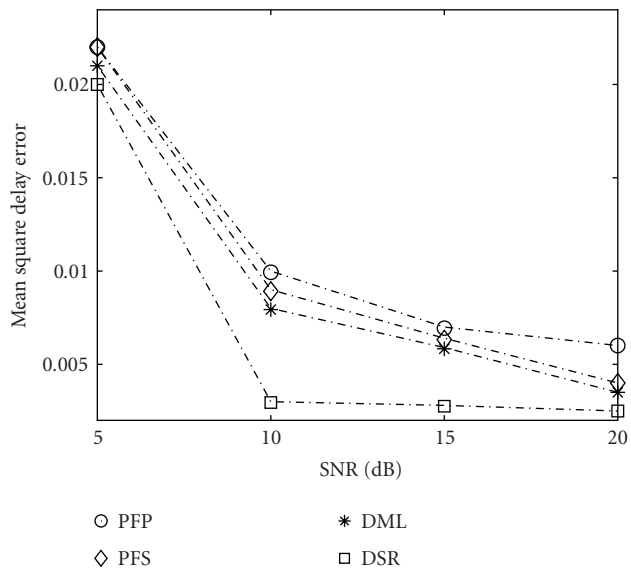


FIGURE 3: Mean square delay error via SNR (4DPSK $N = 100$).

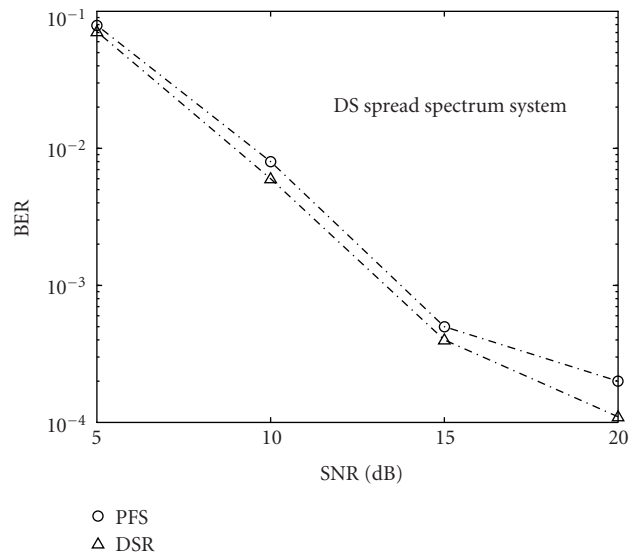


FIGURE 5: Bit error rate.

particles, and is more efficient than both standard particle filtering schemes.

The particle filtering approach was also compared with the EKF method [2] in the second experiment. The algorithm was simplified to consider channel and code delay estimation only (the transmitted symbols were assumed known as, e.g., with pilot symbols being used). Otherwise, simulation set-up was the same ($N = 100$ particles were employed). The results for the SNR = 10 dB presented in Figure 4 demonstrate good performance of the particle filtering method. Please note that deterministic particle filter is in

a sense a variant of the conventional per-survivor processing (PSP) algorithm combined with the randomized particle filtering procedure for the channel and delay estimation. Thus, this procedure has proved more efficient, even in the situation which is more favorable for EKF, that is, which does not involve the uncertainty associated with the unknown transmitted symbols. Simulations demonstrating the superiority of the particle filtering approach over RAKE receiver for related problems are presented in [22]; for symbol detection, the comparison of the particle filtering methods with other well-known approaches could be found in [8, 9, 23].

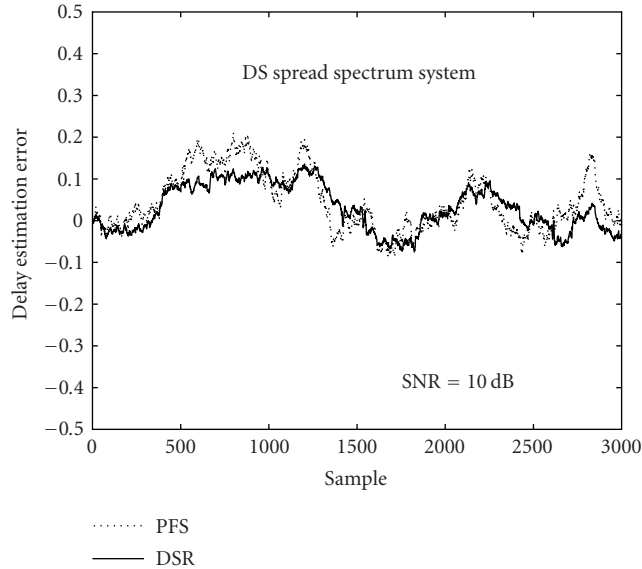


FIGURE 6: The error in delay estimation.

Finally, we applied the proposed algorithm to perform joint symbols/channel coefficients/code delay estimation for DS spread spectrum systems with $K = 15$, $L = 4$. A binary DPSK modulation scheme was employed with the multipath channel response and AR coefficients chosen as in Channel B in [2]. As shown in Figure 5, the algorithm employing 100 particles exhibits good BER performance. A tracking error trajectory for 100 information symbols (corresponding to 1500 chips and 3000 channel samples) and an average SNR equal to 10 dB is presented in Figure 6. Figure 7 also illustrates the mean square delay error as a function of SNR for $L_d = 1000$.

5. CONCLUSION

In this paper, we propose the application of particle filtering techniques to a challenging problem of joint symbols, channel coefficients, and code delay estimation for DS spread spectrum systems in multipath fading. The algorithm is designed to make use of the structure of the model, and incorporates a variance reduction technique. The work is based on the recent results on the superiority of the DML approach in a fully discrete environment [10, 11]. The method cannot be applied straightforwardly, however, and several procedures combining both deterministic and randomized schemes are considered. The algorithms are tested and compared. Although computer simulations show that all methods are capable of providing good performance, in this particular case involving additional continuous-valued parameters, the deterministic scheme employing stratified resampling turns out to be the most efficient one. The choice of the algorithm might, however, be application-dependent, so further investigation is necessary. The receiver can be extended to address multiuser DS-CDMA transmission, using the techniques proposed in [24], for example, or simplified

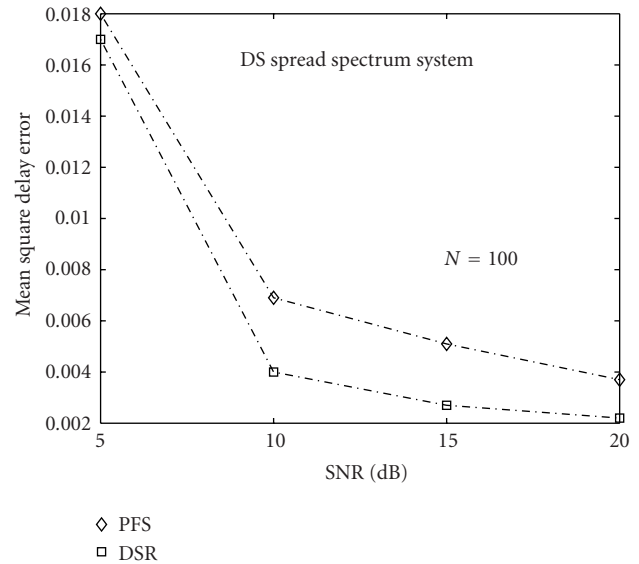


FIGURE 7: Mean square delay error via SNR.

to consider channel tracking only since it is naturally incorporated in the proposed algorithm. Future research should concentrate on the development of suboptimal importance distributions and selection schemes capable of increasing the algorithm efficiency.

REFERENCES

- [1] R. A. Iltis, "A DS-CDMA tracking mode receiver with joint channel/delay estimation and MMSE detection," *IEEE Trans. Communications*, vol. 49, no. 10, pp. 1770–1779, 2001.
- [2] R. A. Iltis, "Joint estimation of PN code delay and multipath using the extended Kalman filter," *IEEE Trans. Communications*, vol. 38, no. 10, pp. 1677–1685, 1990.
- [3] U. Fawer and B. Aazhang, "A multiuser receiver for code division multiple access communications over multipath channels," *IEEE Trans. Communications*, vol. 43, no. 2/3/4, pp. 1556–1565, 1995.
- [4] H. Zamiri-Jafarian and S. Pasupathy, "Adaptive MLSDE using the EM algorithm," *IEEE Trans. Communications*, vol. 47, no. 8, pp. 1181–1193, 1999.
- [5] J. Míguez and L. Castedo, "Semiblind maximum-likelihood demodulation for CDMA systems," *IEEE Trans. Vehicular Technology*, vol. 51, no. 4, pp. 775–781, 2002.
- [6] A. Logothetis and C. Carlemalm, "SAGE algorithms for multipath detection and parameter estimation in asynchronous CDMA systems," *IEEE Trans. Signal Processing*, vol. 48, no. 11, pp. 3162–3174, 2000.
- [7] A. Doucet, J. F. G. de Freitas, and N. J. Gordon, Eds., *Sequential Monte Carlo Methods in Practice*, Springer, New York, NY, USA, 2001.
- [8] R. Chen, X. Wang, and J. S. Liu, "Adaptive joint detection and decoding in flat-fading channels via mixture Kalman filtering," *IEEE Transactions on Information Theory*, vol. 46, no. 6, pp. 2079–2094, 2000.
- [9] E. Punsakaya, C. Andrieu, A. Doucet, and W. J. Fitzgerald, "Particle filtering for demodulation in fading channels with non-Gaussian additive noise," *IEEE Trans. Communications*, vol. 49, no. 4, pp. 579–582, 2001.
- [10] E. Punsakaya, C. Andrieu, A. Doucet, and W. J. Fitzgerald,

“Particle filtering for multiuser detection in fading CDMA channels,” in *Proc. IEEE 11th Signal Processing Workshop on Statistical Signal Processing (SSP '01)*, pp. 38–41, Singapore, August 2001.

- [11] E. Punsakaya, A. Doucet, and W. J. Fitzgerald, “On the use and misuse of particle filtering in digital communications,” in *Proceedings XI European Signal Processing Conference (EUSIPCO '02)*, Toulouse, France, September 2002.
- [12] R. A. Iltis, “A sequential Monte Carlo filter for joint linear/nonlinear state estimation with application to DS-CDMA,” *IEEE Trans. Signal Processing*, vol. 51, no. 2, pp. 417–426, 2003.
- [13] J. G. Proakis, *Digital Communications*, McGraw-Hill, New York, NY, USA, 3rd edition, 1995.
- [14] C. Kominakis, C. Fragouli, A. H. Sayed, and R. D. Wesel, “Channel estimation and equalization in fading,” in *Proc. IEEE 33rd Asilomar Conference on Signals, Systems, and Computers*, vol. 2, pp. 1159–1163, Pacific Grove, Calif, USA, October 1999.
- [15] A. Doucet, S. Godsill, and S. Andrieu, “On sequential Monte Carlo sampling methods for Bayesian filtering,” *Statistics and Computing*, vol. 10, no. 3, pp. 197–208, 2000.
- [16] A. Doucet, N. J. Gordon, and V. Kroschnamurthy, “Particle filters for state estimation of jump Markov linear systems,” *IEEE Trans. Signal Processing*, vol. 49, no. 3, pp. 613–624, 2001.
- [17] T. Ghirmai, M. F. Bugallo, J. Míguez, and P. M. Djuric, “Joint symbol detection and timing estimation using particle filtering,” in *Proc. IEEE Int. Conf. Acoustics, Speech, Signal Processing (ICASSP '03)*, vol. 4, pp. 596–599, Hong Kong, April 2003.
- [18] G. Kitagawa, “Monte Carlo filter and smoother for non-Gaussian nonlinear state space models,” *Journal of Computational and Graphical Statistics*, vol. 5, no. 1, pp. 1–25, 1996.
- [19] D. Crisan, “Particle filters—a theoretical perspective,” in *Sequential Monte Carlo Methods in Practice*, A. Doucet, J. F. G. de Freitas, and N. J. Gordon, Eds., Springer, New York, NY, USA, 2001.
- [20] J. K. Tugnait and A. H. Haddad, “A detection-estimation scheme for state estimation in switching environments,” *Automatica*, vol. 15, no. 4, pp. 477–481, 1979.
- [21] P. Fearnhead, *Sequential Monte Carlo methods in filter theory*, Ph.D. thesis, University of Oxford, Oxford, UK, 1998.
- [22] T. Bertozzi, *Applications of particle filtering to digital communications*, Ph.D. thesis, CNAM, Paris, France, December 2003.
- [23] E. Punsakaya, *Sequential Monte Carlo methods for digital communications*, Ph.D. thesis, Engineering Department, Cambridge University, Cambridge, UK, June 2003.
- [24] B. Dong, X. Wang, and A. Doucet, “A new class of soft MIMO demodulation algorithms,” *IEEE Trans. Signal Processing*, vol. 51, no. 11, pp. 2752–2763, 2003.

Arnaud Doucet was born in France on November 2, 1970. He received the M.S. degree from the Telecom INT in 1993 and the Ph.D. degree from University Paris XI Orsay in December 1997. From 1998 to 2000, he was a Research Associate in the Signal Processing group of Cambridge University. From 2001 to 2002 he was a Senior Lecturer in the Department of Electrical and Electronic Engineering at Melbourne University, Australia. Since September 2002, he has been university Lecturer in information engineering at Cambridge University. He is a Member of the IEEE Signal Processing Theory and Methods Committee and an Associate Editor for The Annals of The Institute of Statistical Mathematics and Test. His research interests include Bayesian statistics and simulation-based methods for estimation and control. He has coedited with J. F. G. de Freitas and N. J. Gordon *Sequential Monte Carlo Methods in Practice*, 2001.



William J. Fitzgerald received the B.S., M.S., and Ph.D. degrees in physics from the University of Birmingham, Birmingham, UK. He is the Professor of applied statistics and signal processing and the Head of Research at the Signal Processing Laboratory, Department of Engineering, University of Cambridge, UK. Prior to joining Cambridge, he worked at the Institut Laue Langevin, Grenoble, France, for seven years and was then a Professor of physics at the Swiss Federal Institute of Technology (ETH), Zürich, for five years. He is a Fellow of Christ's College, Cambridge, where he is Director of studies in engineering. He works on Bayesian inference applied to signal and data modeling. He is also interested in nonlinear signal processing, image restoration and medical imaging, extreme value statistics, bioinformatics, data mining, and data classification.



Elena Punsakaya received the B.S. and M.S. degrees in mechanical engineering from the Bauman Moscow State Technical University, Moscow, Russia, in 1997, and the M.Phil. and Ph.D. degrees in information engineering from the University of Cambridge, Cambridge, UK, in 1999 and 2003, respectively. She is currently with the Signal Processing Laboratory, Department of Engineering, University of Cambridge, UK, and is a Research Fellow of Homerton College, Cambridge, UK, where she is a Director of Studies in Engineering. Her research interests focus on Bayesian inference, simulation-based methods, in particular, Markov chain Monte Carlo and particle filtering, and their application to digital communications.



Particle Filtering Equalization Method for a Satellite Communication Channel

Stéphane Sénécal

*The Institute of Statistical Mathematics, 4-6-7 Minami Azabu, Minato-ku, Tokyo 106-8569, Japan
Email: steph@ism.ac.jp*

Pierre-Olivier Amblard

*Groupe Non Linéaire, Laboratoire des Images et des Signaux (LIS), ENSIEG, BP 46, 38402 Saint Martin d'Hères Cedex, France
Email: bidou.amblard@lis.inpg.fr*

Laurent Cavazzana

*École Nationale Supérieure d'Informatique et de Mathématiques Appliquées de Grenoble (ENSIMAG), BP 72,
38402 Saint Martin d'Hères Cedex, France
Email: laurent.cavazzana@ensimag.imag.fr*

Received 23 April 2003; Revised 23 March 2004

We propose the use of particle filtering techniques and Monte Carlo methods to tackle the in-line and blind equalization of a satellite communication channel. The main difficulties encountered are the nonlinear distortions caused by the amplifier stage in the satellite. Several processing methods manage to take into account these nonlinearities but they require the knowledge of a training input sequence for updating the equalizer parameters. Blind equalization methods also exist but they require a Volterra modelization of the system which is not suited for equalization purpose for the present model. The aim of the method proposed in the paper is also to blindly restore the emitted message. To reach this goal, a Bayesian point of view is adopted. *Prior* knowledge of the emitted symbols and of the nonlinear amplification model, as well as the information available from the received signal, is jointly used by considering the *posterior* distribution of the input sequence. Such a probability distribution is very difficult to study and thus motivates the implementation of Monte Carlo simulation methods. The presentation of the equalization method is cut into two parts. The first part solves the problem for a simplified model, focusing on the nonlinearities of the model. The second part deals with the complete model, using sampling approaches previously developed. The algorithms are illustrated and their performance is evaluated using bit error rate versus signal-to-noise ratio curves.

Keywords and phrases: traveling-wave-tube amplifier, Bayesian inference, Monte Carlo estimation method, sequential simulation, particle filtering.

1. INTRODUCTION

Telecommunication has been taking on increasing importance in the past decades and thus led to the use of satellite-based means for transmitting information. A major implementation task to deal with such an approach is the attenuation of emitted communication signals during their trip through the atmosphere. Indeed, one of the most important roles devoted to telecommunication satellites is to amplify the received signal before sending it back to Earth. Severe technical constraints, due to the lack of space and energy available on board, can be solved thanks to special devices, namely, traveling-wave-tube (TWT) amplifiers [1]. A common model for such a satellite transmission chain is depicted in Figure 1.

Although efficient for amplifying tasks, TWT devices suffer from nonlinear behaviors in their characteristics, thus implying complex modeling and processing methods for equalizing the transmission channel.

The very first approaches for solving the equalization problem of models similar to the one depicted in Figure 1 were developed in the framework of neural networks. These methods are based on a modelization of the nonlinearities using layers of perceptrons [2, 3, 4, 5, 6]. Most of these approaches require a learning or training input sequence for adapting the parameters of the equalization algorithm. However, the knowledge or the use of such sequences is sometimes impossible: if the signal is intensely corrupted by noise at the receiver stage or for noncooperative applications, for instance.

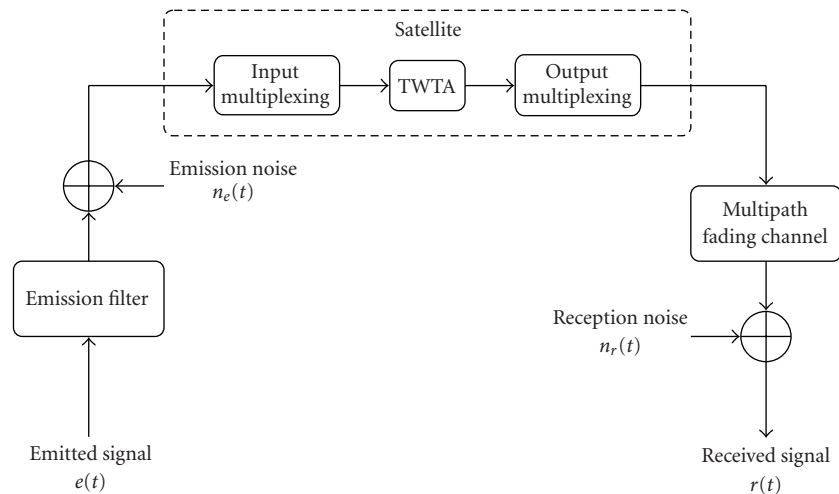


FIGURE 1: Satellite communication channel.

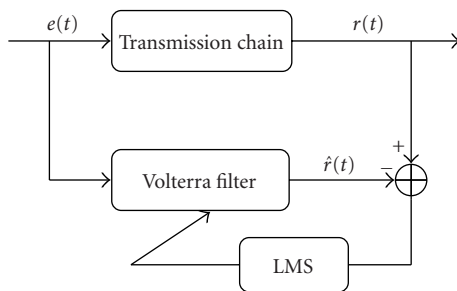


FIGURE 2: Identification of the model depicted in Figure 1 with a Volterra filter.

Blind equalization methods have thus to be considered. These methods often need precise hypothesis with the emitted signals: Gaussianity or circularity properties of the probability density function of the signal, for instance [7]. Recently, some methods make it possible to identify [8] or equalize [9, 10, 11] blindly nonlinear communication channels under general hypothesis. These blind equalization methods assume that the transfer function of the system can be modeled as a Volterra filter [12, 13].

However, for the transmission model considered here, a Volterra modelization happens to be only suitable for the task of identification and not for a direct equalization. For instance, a method based on a Volterra modelization of the TWT amplifier and a Viterbi algorithm at the receiver stage is considered in [14]. Such an identification method can be easily implemented through a recursive adaptation rule of the filter parameters with a least mean squares approach (cf. Figure 2). The mean of the quadratic error (straight line) and its standard deviation (dotted lines) are depicted in Figure 3 for 100 realizations of binary phase shift keying (BPSK) symbol sequences, each composed of 200 samples. Similarly, the equalization problem of the transmission chain 1 can be considered with a Volterra filter scheme, adapted with a recursive least squares algorithm as depicted in Figure 4. However, in

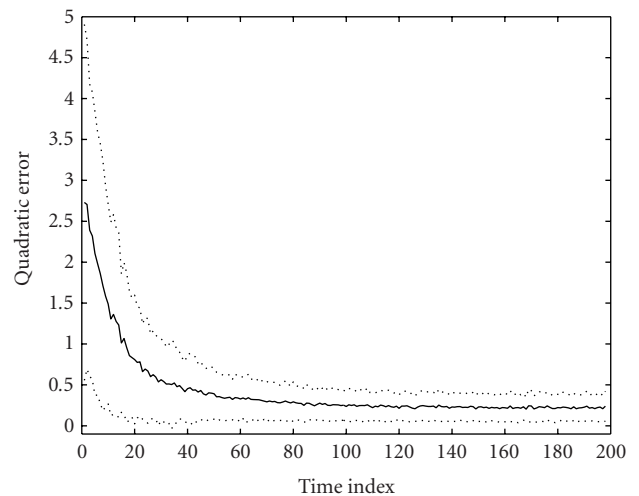


FIGURE 3: Identification error, scheme of Figure 2.

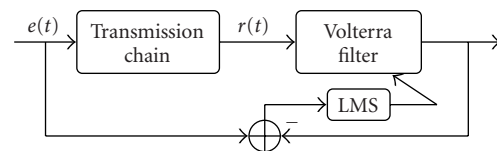


FIGURE 4: Equalization of the model depicted in Figure 1 with a Volterra filter.

this case, the error function happens not to converge showing that the Volterra filter is unstable and that the system is not invertible with such modelization.

It is then necessary to consider a different approach for realizing blindly the equalization of this communication model. The aim of this paper is thus to introduce a blind and sequential equalization method based on particle filtering (sequential Monte Carlo techniques) [15, 16]. For realizing the equalization of the communication channel, it

seems interesting to fully exploit the analytical properties of the nonlinearities induced by TWT amplifiers through parametric models of these devices [1]. Sequential Monte Carlo methods, originally developed for the recursive estimation of nonlinear and/or non-Gaussian state space models [17, 18], are well suited for reaching this goal. The field of communication seems to be particularly propitious for applying particle filtering techniques, as shown in the recent literature of the signal processing community [15, 19] and this issue and of the statistics community (see [20, 21], [16, Section 4]). Such Monte Carlo approaches were successfully applied to blind deconvolution [22], equalization of flat-fading channels [23], and phase tracking problems [24], for instance.

The paper is organized as follows. Firstly, models for the emitted signal, the TWT amplification stage, and the other parts of the transmission chain are introduced in Section 2. A procedure for estimating the emitted signal is considered in a Bayesian framework. Monte Carlo estimation techniques are then proposed in Section 3 for implementing the computation of the estimated signal under the assumption of a simpler communication model, focusing on the nonlinear part of the channel. This approach uses analytical formulae of the TWT amplifier model described in Section 2 and sampling methods for estimating integral expressions. The method is then generalized in Section 4 for building a blind and recursive equalization scheme of the complete transmission chain. The sequential simulation algorithm proposed is based on particle filtering techniques. This approach makes it possible to process the data in-line and without the help of a learning input sequence. The performance of the algorithm is illustrated by numerical experiments in Section 5. Finally, some conclusions are drawn in Section 6. Details of the Monte Carlo approach are given in Appendix A.

2. MODELING OF THE TRANSMISSION MODEL

The model of the satellite communication channel depicted in Figure 1 is roughly the same as the one considered for various problems dealing with TWT amplifiers devices (cf., e.g., [2]). The different stages of this communication channel are detailed below.

2.1. Emission stage

The information signal to transmit is denoted by $e(t)$. It is usually a digital signal composed of a sequence of N_e symbols $(e_k)_{1 \leq k \leq N_e}$. The signal is transmitted under the analog form

$$e(t) = \sum_{k=1}^{N_e} e_k \mathbb{1}_{[(k-1)T, kT]}(t), \quad (1)$$

where T denotes the symbol rate and $\mathbb{1}_{\Omega}(\cdot)$ is the indicator function of set Ω . Symbols e_k are generated from classical modulations used in the field of digital telecommunication, like PSK or quadratic amplitude modulation (QAM), for instance. In the following, the case of 4-QAM symbols is considered. Each symbol can be written as

$$e_k = \exp(i\phi_k), \quad (2)$$

where the sequence of samples $(\phi_k)_{1 \leq k \leq N_e}$ is independently and identically distributed from

$$\mathcal{U}_{\{\pi/4, 3\pi/4, 5\pi/4, 7\pi/4\}}, \quad (3)$$

where \mathcal{U}_{Ω} denotes the uniform distribution on the set Ω . The signal is emitted through the atmosphere to the satellite. The emission process is modeled by a Chebyshev filter. This class of filters admits an IIR representation and their parameters, particularly their cutoff frequency, depend on the value of symbol rate T [2]. A detailed introduction to Chebyshev filters is given in [25], for instance. In the present case, the emission filter is assumed to be modeled with a 3 dB bandwidth equal to $1.66/T$. The emitted signal is altered during its trip in the atmosphere by disturbance signals. These phenomena are modeled by an additive corrupting noise $n_e(t)$, which is assumed to be Gaussianly, independently, and identically distributed:

$$n_e(t) \sim \mathcal{N}_{CC}(0, \sigma_e^2), \quad (4)$$

where $\mathcal{N}_{CC}(0, \sigma_e^2)$ is a complex circular Gaussian distribution, with zero-mean and variance equal to σ^2 .

Remark 1. The amplitude of signal (1) is adjusted in practice at the emission stage in order to reach a signal-to-noise ratio (SNR) roughly equal to 15 dB during the transmission.

2.2. Amplification

After being received by the satellite, the signal is amplified and sent back to Earth. This amplification stage is processed by a TWT device. A simple model for TWT amplifier is an instantaneous nonlinear filter defined by

$$z = r \exp(i\phi) \mapsto Z = A(r) \exp(i(\phi + \Phi(r))), \quad (5)$$

where r denotes the modulus of input signal. Amplitude gain and phase wrapping can be modeled by the following expressions:

$$A(r) = \frac{\alpha_a r}{1 + \beta_a r^2}, \quad (6)$$

$$\Phi(r) = \frac{\alpha_p r^2}{1 + \beta_p r^2}. \quad (7)$$

These formulae have been shown to model various types of TWT amplifier device with accuracy [1]. Figures 5 and 6 represent functions (6) and (7) for two sets of parameters estimated in [1, Table 1] from real data and duplicated in Table 1. Curves with straight lines represent functions obtained with the set of parameters of the first row of Table 1. The ones with dashed lines represent functions obtained with the other set of parameters.

A drawback of model (5) is that it is not invertible in a strict theoretical sense, as drawn in Figure 5. However, only the amplificative and invertible part of the system, represented above the dotted line on Figure 5, will be considered.

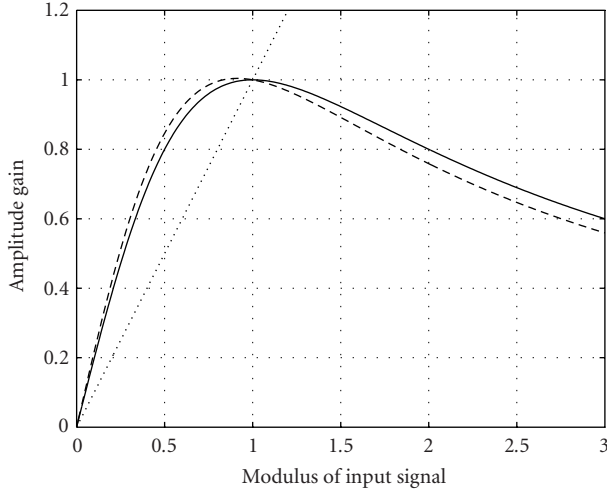


FIGURE 5: Amplitude gain (6) of TWT models.

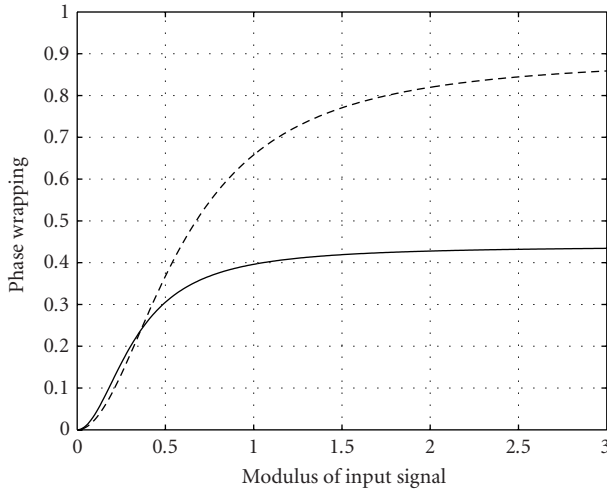


FIGURE 6: Phase wrapping (7) of TWT models.

Signal processing in the satellite also performs the task of multiplexing. The devices used for this purpose are modeled by Chebyshev filters. Tuning of their parameters is given in [2], for instance. In the present case, filters at the input and at the output of the amplifier are assumed to have bandwidths equal, respectively, to $2/T$ and $3.3/T$.

2.3. Reception

The transmission of the signal back to Earth is much less powerful than at the emission stage. This is mainly due to severe technical constraints because of the satellite design. The influence of the atmospheric propagation medium is then modeled by a multipath fading channel [26, Section 11], with one reflected path representing an attenuation of 10 dB in this case: $z(t) \rightarrow z(t) + \alpha z(t - \Delta)$. Moreover, the signal is still corrupted by disturbance signals, modeled by an additive noise signal $n_r(t)$, Gaussianly, independently, and identically

TABLE 1: Parameters of (6) and (7) measured in practice.

α_a	β_a	α_p	β_p
2	1	4	9.1
2.2	1.2	2.5	2.8

distributed:

$$n_r(t) \sim \mathcal{N}_{CC}(0, \sigma_r^2). \quad (8)$$

This noise is always much more intense than at the emission stage. This is mainly due to the weak emission power available in the satellite. The received signal, denoted as $r(t)$, is sampled at rate T_s .

2.4. Equalization

The goal of equalization is to recover emitted sequence $(e_k)_{1 \leq k \leq N_e}$ from the knowledge of sampled sequence $(r(jT_s))_{1 \leq j \leq N_r}$. The equalization method proposed in this paper consists in estimating symbol sequence $(\phi_k)_{1 \leq k \leq N_e}$ by considering its *posterior* distribution conditionally to sequence $(r(jT_s))_{1 \leq j \leq N_r}$ of samples of the received signal:

$$p\left((\phi_k)_{1 \leq k \leq N_e} \mid (r(jT_s))_{1 \leq j \leq N_r}\right). \quad (9)$$

To reach this goal, a Bayesian estimation procedure is considered with the computation of *maximum a posteriori* (MAP) estimates [27].

Remark 2. Bayesian approaches have already been successfully applied in digital signal processing in the field of mobile communication. In [28], for instance, autoregressive models and discrete-valued signals are considered.

The computation of the estimates is implemented via Monte Carlo simulation methods [16, 29]. As the complete transmission chain is a complex system, a simpler model focusing on the nonlinear part of the channel is considered in the following section, where Monte Carlo estimation approaches are introduced. These estimation techniques will be used in the equalization algorithm for the global transmission chain in Section 4.

3. MONTE CARLO ESTIMATION METHODS

As a first approximation, to focus on the nonlinearity of the model, only a TWT amplifier is considered in a transmission channel corrupted with noises at its input and output parts as shown in Figure 7.

The received signal $r(t)$ is assumed to be sampled at symbol rate T . The problem is then to estimate a 4-QAM symbol ϕ *a priori* distributed from (3) with the knowledge of the model depicted in Figure 7 (cf. relations (4), (6), (7) and (8)), and information of a received sample r . A Bayesian approach is developed [27] by considering the *posterior* distribution

$$p(\phi|r) \quad (10)$$

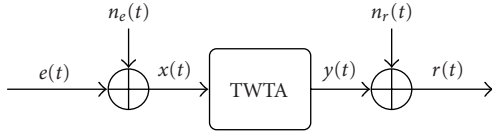


FIGURE 7: Simple communication channel with TWT amplifier.

and its classical MAP estimate. The method proposed in the following consists in estimating values of distribution (10) thanks to Monte Carlo simulation schemes [29] using relations (6) and (7) which model nonlinearities of the TWT amplifier in a parametric manner.

3.1. Estimation with known parameters

In order to further simplify the study, parameters of the transmission channel depicted in Figure 7 are firstly assumed to be known. In the sequel, the coefficients of expressions (6) and (7) are denoted by the symbol TWT. This information is taken into account in the *posterior* distribution (10) thus becoming

$$p(\phi | A, \sigma_e, \text{TWT}, \sigma_r, r), \quad (11)$$

where A denotes the amplitude of the emitted signal. From Bayes' formula, the probability density function of this distribution is proportional to

$$p(r | A, \phi, \sigma_e, \text{TWT}, \sigma_r) \times p(\phi | A, \sigma_e, \text{TWT}, \sigma_r). \quad (12)$$

The *prior* distribution at the right-hand side of the above expression reduced to $p(\phi)$, which is given by (3). The problem is then to compute the likelihood

$$p(r | A, \phi, \sigma_e, \text{TWT}, \sigma_r). \quad (13)$$

Indeed, this formula can be viewed (cf. Appendix A) as the following expectation:

$$\mathbb{E} \left\{ \exp \left(- \frac{1}{\sigma_r^2} |r - \text{TWT}(x)|^2 \right) \right\} \quad (14)$$

with respect to the random variable x which is Gaussianly distributed:

$$x \sim \mathcal{N}_{CC}(A \exp(i\phi), \sigma_e^2). \quad (15)$$

Considering a sequence of samples $(x_\ell)_{1 \leq \ell \leq N}$ independently and identically distributed from (15), a Monte Carlo approximation of (14) is given by

$$\frac{1}{N} \sum_{\ell=1}^N \exp \left(- \frac{1}{\sigma_r^2} |r - \text{TWT}(x_\ell)|^2 \right) \quad (16)$$

which is accurate for a number N of samples large enough. References [29, 30] provide detailed ideas and references about Monte Carlo methods. To illustrate such an approach, approximation (16) is computed for the emitted symbol $\phi = \pi/4$ and the values of TWT amplifier parameters given by

TABLE 2: Estimates of (11), $\text{SNR}_e = 10$ dB, $\text{SNR}_r = 3$ dB, 100 realizations.

ϕ	$\hat{p}(\phi r)$
$\frac{\pi}{4}$	0.69 ± 0.28
$\frac{3\pi}{4}$	0.13 ± 0.21
$\frac{5\pi}{4}$	0.02 ± 0.05
$\frac{7\pi}{4}$	0.16 ± 0.24

the first row of Table 1. Amplitude A of the emitted signal equals 0.5 and variances of transmission noises are such that $\text{SNR}_e = 10$ dB and $\text{SNR}_r = 3$ dB. One hundred realizations are simulated and, for each, a sequence (15) of 100 samples is considered. Table 2 gives mean values obtained from (16) and their standard deviations.

The error of the estimated values (16) of probabilities (11) might seem quite large as the standard deviations can be reduced providing a larger number of samples. In the sequel, we are only interested in obtaining rough estimates of (11), enabling comparison of mean values for different ϕ as shown in Table 2. Thus, even with a reduced number of samples (15), it is possible to estimate accurately the MAP estimate of (11).

Performance of the Monte Carlo estimation method is then considered with respect to SNR at the input and output of the amplifier (cf. Figure 7). The bit error rate (BER) is computed by averaging the results obtained with a MAP approach. Statistics of the Monte Carlo estimates (16) of distribution (11) are computed with 100 realizations of symbol sequences composed of 1,000 samples each. For each estimate, sequences (15) composed of 100 samples are considered. The results of these simulations for SNR_e taking values 10, 12, and 15 dB are depicted in Figure 8 and curves from the bottom to the top are associated to decreasing SNR_e .

The Bayesian approach and its Monte Carlo implementation make it possible to estimate the emitted signal with accuracy for a wide range of noise variances (cf. Figure 8). However, the estimation method described previously requires the knowledge of the model parameters. For many applications in the field of telecommunication, it is necessary to assume these parameters unknown. It is the case for non-stationary transmission models and for communication in noncooperative contexts like passive listening, for instance. The equalization problem of the simplified model depicted in Figure 7 is now tackled in the case where the parameters $(A, \sigma_e, \text{TWT}, \sigma_r)$ of the transmission channel are assumed to be unknown.

3.2. Estimation with unknown parameters

If the parameters are unknown, there are, at least, two Bayesian estimation approaches to be considered with respect to *posterior* distribution (10). A first method consists

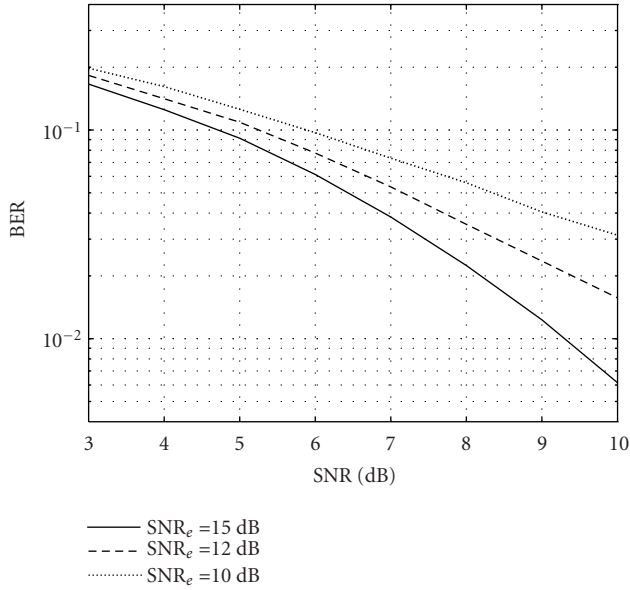


FIGURE 8: Mean BER values for MAP estimates of signals versus SNR_r in dB, model of Figure 7 with known parameters $(A, \sigma_e, \text{TWT}, \sigma_r)$, for various SNR_e values.

in dealing with the joint distribution of all the parameters of the model

$$p(\phi, A, \sigma_e, \text{TWT}, \sigma_r | r). \quad (17)$$

This method makes it possible theoretically to jointly estimate the emitted symbols and the parameters of the transmission channel by implementing MAP and/or *posterior* mean approaches. The probability density function of distribution (17) being generally very complex, Markov chain Monte Carlo (MCMC) simulation methods [29, 30] can be used to perform these estimation tasks. Such an approach is developed in [31] particularly for equalizing the complete transmission chain depicted in Figure 1. However, results obtained with this method happen not to give accurate estimates of the model parameters in practice. Indeed, MCMC methods are generally useful for estimating various models in the field of telecommunication [28, 32].

Another approach consists in considering a marginalized version of distribution (10) with respect to the parameters of the model:

$$p(\phi) = \int p(\phi, A, \sigma_e, \text{TWT}, \sigma_r | r) d(A, \sigma_e, \text{TWT}, \sigma_r). \quad (18)$$

Such a technique, called Rao-Blackwellization in the statistics literature, for example, [33, 34], makes it possible to improve the efficiency of sampling schemes (see [20, 21], [16, Section 24]). From Bayes' formula, the integrand of expression (18) is proportional to

$$p(r | \phi, A, \sigma_e, \text{TWT}, \sigma_r) \times p(\phi, A, \sigma_e, \text{TWT}, \sigma_r). \quad (19)$$

Assuming that symbols and the model parameters are independent, expression (18) is proportional to

$$p(\phi) \times \int p(r | \phi, A, \sigma_e, \text{TWT}, \sigma_r) p(A, \sigma_e, \text{TWT}, \sigma_r) \times d(A, \sigma_e, \text{TWT}, \sigma_r). \quad (20)$$

From the study of the previous case, the likelihood term in the integrand can be computed via a Monte Carlo estimate of expression (14) with a sequence of samples (15). An approach to estimate (18) is then to consider the integral expression in (20) as the expectation

$$E_{p(A, \sigma_e, \text{TWT}, \sigma_r)} \{p(r | \phi, A, \sigma_e, \text{TWT}, \sigma_r)\} \quad (21)$$

which is estimated via a Monte Carlo approximation of the following form:

$$\frac{1}{N_p} \sum_{k=1}^{N_p} p(r | \phi, A_k, \sigma_e(k), \text{TWT}_k, \sigma_r(k)), \quad (22)$$

where $(A_k, \sigma_e(k), \text{TWT}_k, \sigma_r(k))_{k=1, \dots, N_p}$ is a sequence of samples independently and identically distributed from the *prior* distribution

$$p(A, \sigma_e, \text{TWT}, \sigma_r). \quad (23)$$

Remark 3. The algorithm for sampling distribution (17) introduced in [31] requires also the setting of *prior* distribution (23).

The model of the parameters includes generally *prior* information thanks to physical constraints. For instance, the TWT amplifier is assumed to work in the amplificative part of its characteristic (cf. Figure 5). Thus, as a first rough approximation, it can be assumed that $A \sim \mathcal{U}_{[0,1]}$ *a priori*. This parameter is also tuned such that SNR_e equals 15 dB during the emission process (cf. Remark 1) implying the constraint $\sigma_e = 0.2A$. In a less strict case, it is sufficient to assume that $\sigma_e \sim \mathcal{U}_{[0.01, 0.5]}$. The parameters $(\alpha_a, \beta_a, \alpha_p, \beta_p)$ of the TWT amplifier are supposed to be independent of other variables of the system and also to be mutually independent. From the values introduced in Table 1, an adequate *prior* distribution is

$$(\alpha_a, \beta_a, \alpha_p, \beta_p) \sim \mathcal{U}_{[1,3]} \times \mathcal{U}_{[0,2]} \times \mathcal{U}_{[1,5]} \times \mathcal{U}_{[2,10]}. \quad (24)$$

The extremal values of the downlink transmission noise variance σ_r can be estimated with respect to *prior* ranges of values defined above. A uniform $\mathcal{U}_{[0.1, 1.1]}$ *prior* distribution for σ_r is thus chosen. Once all *prior* distributions have been defined, it is possible to implement a Monte Carlo estimation procedure for (20) with the help of approximations (22) and (16). Such an approach is tested for the computation of values of *posterior* distribution for an emitted symbol $\phi = \pi/4$ considering that the values of the TWT amplifier are given by the first row of Table 1, that the amplitude of the emitted signal is given by $A = 0.5$, and that noise variances are such that

TABLE 3: Estimates of (10), $\text{SNR}_e = 10$ dB, $\text{SNR}_r = 3$ dB, 100 realizations.

ϕ	$\hat{p}(\phi r)$
$\frac{\pi}{4}$	0.61 ± 0.21
$\frac{3\pi}{4}$	0.23 ± 0.20
$\frac{5\pi}{4}$	0.05 ± 0.03
$\frac{7\pi}{4}$	0.12 ± 0.14

$\text{SNR}_e = 10$ dB and $\text{SNR}_r = 3$ dB. One hundred estimations are simulated and for each realization, sequences of 100 samples

$$(A_k, \sigma_e(k), \text{TWT}_k, \sigma_r(k))_{1 \leq k \leq N_p} \quad (25)$$

are drawn from distribution (23). For each sequence, as in the case where the model parameters are known, approximations (16) are computed from sequences (25) composed of 100 samples each. Table 3 shows the mean values of the estimated (22) and their standard deviations computed from these simulations.

As for the previous case, where parameters ($A, \sigma_e, \text{TWT}, \sigma_r$) are known, we are only interested in obtaining rough mean values of Monte Carlo estimates of the MAP expression (10) and thus do not consider larger sample sizes for reducing the standard deviation of these estimates.

Performance of this Monte Carlo estimation method is now considered with respect to uplink and downlink SNR. As previously, BERs are computed by averaging results obtained with a MAP approach for the Monte Carlo estimate (22) of *posterior* distribution (10). One hundred realizations of 1000-symbol sequences are considered for each value of SNR. For every Monte Carlo estimate, sequences (25) and (15) are composed of 100 samples. The results of simulations for SNR_e equal to 10, 12, and 15 dB are depicted in Figure 9. Curves from the bottom to the top are associated to a decreasing uplink SNR. As a comparison, the estimated mean values of BER in the case where the parameters of the TWT amplifier are known are represented with dashed lines.

Performance is not much corrupted in the case where model parameters are unknown. Thus, considering the *posterior* distribution of interest (18), marginalized with respect to these parameters, seems to be a good strategy for tackling the equalization problem. An in-line simulation method based on the Monte Carlo estimation techniques previously developed is proposed hereinafter for realizing the equalization of the complete transmission chain depicted in Figure 1.

4. PARTICLE FILTERING EQUALIZATION METHOD

4.1. Transmission model

Equalizing the complete satellite communication channel depicted in Figure 1 is a difficult problem as it requires taking

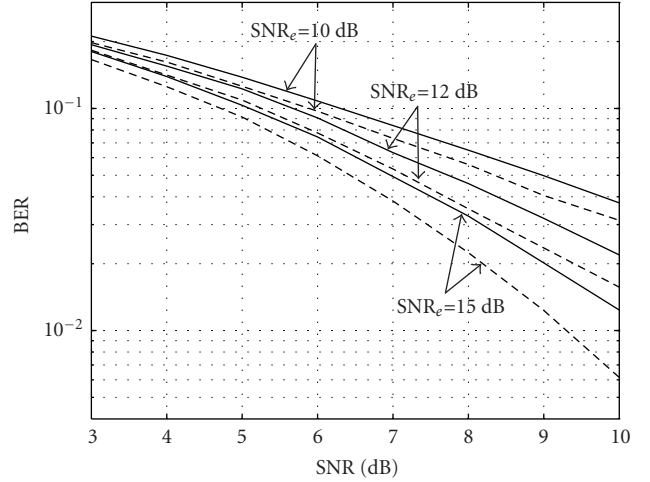


FIGURE 9: Mean BER values for MAP estimates of signals versus SNR_r in dB, model of Figure 7 with unknown/known parameters ($A, \sigma_e, \text{TWT}, \sigma_r$) (straight/dashed lines), for various SNR_e values.

into account several phenomena:

- (1) effects of the filters modeling, emission, and multiplexing stages;
- (2) attenuation of the received signal mainly due to multiple paths during the downlink transmission;
- (3) correlation induced by filters and emission and fading models.

An equalization method is proposed for this model within a Bayesian estimation framework [27]. It consists in considering the *posterior* distribution of the sampled symbols conditionally to the sequence of the received samples:

$$p\left((e(jT_s))_{1 \leq j \leq N_r} \mid (r(jT_s))_{1 \leq j \leq N_r}\right). \quad (26)$$

An estimation procedure is then implemented by computing the MAP estimate of distribution (26). Monte Carlo estimation methods developed in the previous paragraphs can be slightly modified in order to take into account the parameters of the complete transmission chain (cf. points (1) and (2) above).

The correlation of the samples at the receiver stage mainly comes from the linear filters in the channel. In fact, this problem yields to the estimation of parameter p : the number of received samples per symbol rate $p = T/T_s$, as parameters of Chebyshev filters at the emission and multiplexing stages depend on its value.

Computing the correlation of the received samples makes it possible to give an estimate of p [31] in the case where this quantity is an integer, and thus to estimate the parameters of the filters. In the sequel, we consider that this is the case, assuming that a proper synchronization processing has been performed at the receiver stage. This task can also be achieved via Monte Carlo simulation methods [24]. This parameter p will be used in an explicit manner in the recursive equalization algorithm introduced in Section 4.3.

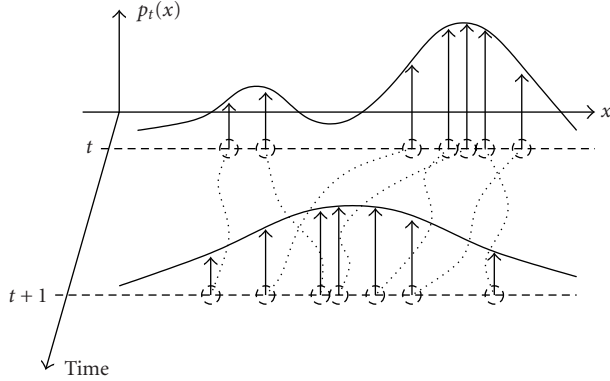


FIGURE 10: Formal updating scheme of a particle filter.

An MCMC simulation scheme [29, 30], for the batch processing of received data, was studied in [31]. A sequential simulation method for sampling the distribution (26) is now introduced, as many applications in the field of telecommunication require in-line processing methods when data is available sequentially.

4.2. Sequential simulation method

A sequential method for sampling distribution (26) can be implemented via particle filtering techniques [15, 16]. The wide scope of this approach, originally developed for the recursive estimation of nonlinear state space models [17, 18, 20, 21], is well suited for the sampling task of this equalization problem. The basic idea of particle filtering is to generate iteratively sequences of the variables of interest, each of them denoted as a “particle,” here written as

$$(x_0(i), x_1(i), \dots, x_t(i))_{1 \leq i \leq M} \quad (27)$$

such that particles $(x_t(1), \dots, x_t(M))$ at time t are distributed from the desired distribution, denoted as $p_t(x)$. This goal can be reached with the use of two “tuning factors” in the algorithm:

- (i) the way the particles are propagated or diffused, $x_t(i) \rightarrow x_{t+1}(i)$, in the sampling space, namely, the choice of a proposal or candidate distribution;
- (ii) the way the distribution of particles $(x_t(1), \dots, x_t(M))$ approximates the target distribution $p_t(x)$: by affecting a weight $w_t(i)$ to each particle depending on the proposal distribution, and updating these weights with an appropriate recursive scheme.

These two tasks are illustrated in Figure 10, where each “ball” stands for a particle $x_t(i)$ whose weight is represented by the length of an associated arrow.

Such recursive simulation algorithm is referred to as sequential importance sampling or particle filtering in the literature [16, 18, 20, 21] of Monte Carlo methods. A good choice of the candidate distribution generally makes it possible to reduce the computational time of the sampling scheme, as

- (1) Initialization. Sample $\phi_0(i) \sim \mathcal{U}_{\{\pi/4, 3\pi/4, 5\pi/4, 7\pi/4\}}$, set the weights $w_0(i) = 1/M$ for $i = 1, \dots, M$, set $j = 1$.

- (2) Importance sampling. Diffuse, propagate the particles by drawing

$$\tilde{\phi}_j(i) \sim p(\phi_j | \phi_{j-1}(i)) \quad (28)$$

for $i = 1, \dots, M$, and actualize the paths

$$[\tilde{\phi}_0(i), \dots, \tilde{\phi}_j(i)] = [\phi_0(i), \dots, \phi_{j-1}(i), \tilde{\phi}_j(i)].$$

- (3) Compute, update the weights

$$\tilde{w}_j(i) = p(r(jT_{\text{ech}}) | \tilde{\phi}_j(i)) \times w_{j-1}(i), \quad (29)$$

and normalize them: $w_j(i) = \tilde{w}_j(i) / \sum_{k=1}^M \tilde{w}_j(k)$.

- (4) Selection/actualization of particles. Resample M particles $(\phi_0(i), \dots, \phi_j(i))$ from the set $(\tilde{\phi}_0(i), \dots, \tilde{\phi}_j(i))_{1 \leq i \leq M}$ according to their weights $(w_j(i))_{1 \leq i \leq M}$ and set the weights equal to $1/M$.
- (5) $j \leftarrow j + 1$ and go to (2).

ALGORITHM 1: Equalization algorithm.

explained in the next paragraph. Such Monte Carlo simulation scheme is now proposed to tackle the sequential sampling of distribution (26).

4.3. Equalization algorithm

In the present case, phase samples $\phi_j = \phi(jT_s)$ of the emitted signal are directly sampled. The simulation scheme which is considered is the bootstrap filter [15, 16, 17, 18] and is given in Algorithm 1.

The important sampling and computation steps (28) and (29) are detailed hereinafter.

The information brought by parameter p , number of received samples per symbol duration, is taken into account via the proposal distribution (28). Indeed, candidates for particles $\tilde{\phi}_j(i)$ can be naturally sampled from the following scheme:

- (i) Set $\tilde{\phi}_j(i) = \tilde{\phi}_{j-1}(i)$ with probability $1 - 1/p$;
- (ii) Sample $\tilde{\phi}_j(i) \sim \mathcal{U}_{\{\pi/4, 3\pi/4, 5\pi/4, 7\pi/4\}}$ with probability $1/p$.

This sampling scheme is very simple and can easily be improved by considering $\tilde{\phi}_{kp}(i) \sim \mathcal{U}_{\{\pi/4, 3\pi/4, 5\pi/4, 7\pi/4\}}$ and $\tilde{\phi}_{k+p}(i) = \tilde{\phi}_{kp}(i)$ for $1 \leq k \leq p - 1$, for instance. However, the scheme above gives sufficiently accurate results as a first approximation, due to its flexibility (if a false symbol is chosen, there is probability to switch to other symbols again) and its ability to deal with possible uncertainty on the value of parameter p . This scheme is also efficient to limit the negative effect of sample impoverishment due to the resampling step, as detailed hereinafter. The proposed scheme, however, does not take into account completely the information coming from emission and received signals and if some coding techniques are used to generate the symbols, this knowledge should be introduced in the sampling scheme (28) if possible.

The computation of weights (29) is realized by using similar Monte Carlo approaches to the ones introduced previously, including filters and their parameters in expressions (14) and (18). In this respect, Algorithm 1 can be seen as a Rao-Blackwellized particle filter [20, 21] where the parameters of the channel, considered here as nuisance parameters, are integrated out. This generally helps to lead to more robust estimates in practice [16, 33, 34].

A crucial point in the implementation of particle filtering techniques lies in the resampling stage, step (4) in Algorithm 1. As the computations for sampling the candidates (28) and updating the weights (29) can be performed in parallel, the resampling step gives the main contribution in the computing time of the algorithm as its achievement needs the interaction of all the particles. This stage is compulsory in practice if one wants the sampler to work efficiently. This is mainly due to the fact that the sequential importance sampling algorithm without resampling increases naturally the variance of the weights $(w_j(i))_{1 \leq i \leq M}$ with time [20, 22, 35]. In such case, only a few particles are affected nonnegligible weights after several iterations, implying a poor approximation of the target distribution and a waste of computation.

To limit this effect, several approaches can be considered [15]. One consists in using very large numbers of particles M and/or in performing the resampling step for each iteration [17, 18]. However, resampling too many times often leads to severe sample impoverishment [16, 20, 21]. Other methods, also aiming at minimizing computational and memory costs, consist in using efficient sampling schemes for diffusing the particles [20] and performing occasionally the resampling stage when it seems to be needed [15, 21]. When to perform resampling can be decided by measuring the variance of weights via the computation of the effective sample size $M/(1 + \text{var}(\hat{w}_j(i)))$, whose one estimate is given by $1/\sum_{i=1}^M w_i^2(i)$ [15, 16, 21, 35]. In this case, the resampling stage can be performed each time the estimated effective sample size is small, measuring how the propagation of the particles in the sampling space is efficient. This quantity equals M for uniformly weighted particles and equals 1 for degenerated cases where all the particles have zero weights except one.

It is also possible to compute the entropy of the weights, describing “how far” the distribution of the weights is from the uniform distribution. Indeed, the entropy is maximized for uniform weights and minimized for the degenerated configurations as mentioned above. In this sense, the entropy of weights quantifies the information of the samples and measures the efficiency of representation for a given population of particles. This approach is adopted in [24, 36], for instance, and also in our algorithm as follows. Step (4) of Algorithm 1 is therefore replaced by the computing of entropy of the weights

$$H(w_t(1), \dots, w_t(M)) = - \sum_{i=1}^M w_t(i) \log(w_t(i)) \quad (30)$$

and a resampling/selection step is processed only if the con-

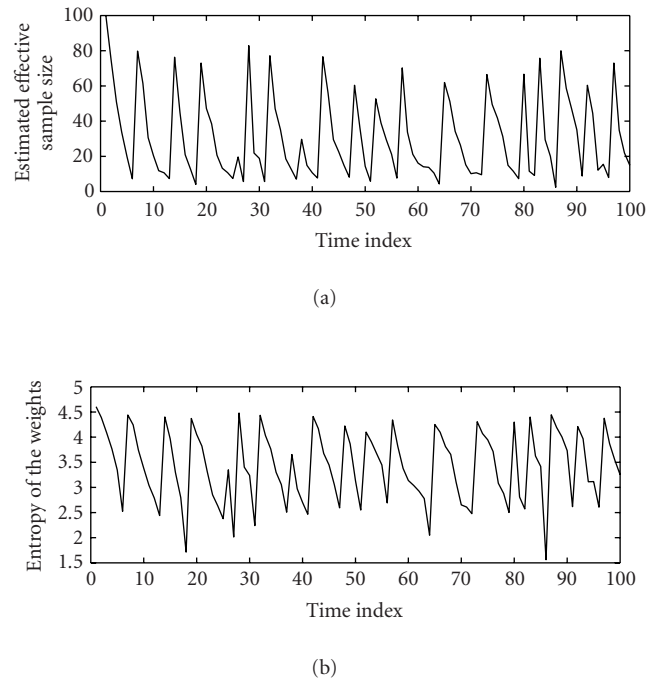


FIGURE 11: (a) Estimated effective sample size (ESS) and (b) entropy (H) of the weights for one realization of the particle filtering algorithm, $M=100$ particles, resampling performed when $\text{ESS} \leq M/10$ or $H \leq \log M/2$.

dition

$$H(w_t(1), \dots, w_t(M)) \leq \lambda \times \max_{w(1), \dots, w(M)} H(w(1), \dots, w(M)) = \lambda \log M \quad (31)$$

holds, assuming that λ is a threshold value set by the user. To show that the estimated effective sample size and entropy lead to similar results for the resampling task, their values for one realization of the algorithm are depicted in Figure 11.

Also, the resampling step can be performed via different techniques [18, 21]. In the sequel, we use the general multinomial sampling procedure [16, 18].

As the variable of interest ϕ is distributed from a set of discrete values, using large numbers of particles M happened to be unnecessary. Simulations have shown that several dozens are sufficient for the problem considered here. This is also the case for other Monte Carlo simulation methods used for estimating telecommunication models [37]. Moreover, the degeneracy phenomenon is not really a nuisance in the present case as the simulation algorithm Algorithm 1 is only used in a MAP estimation purpose and not for computing mean integral estimates as (18) and (A.4), for instance.

Some simulations concerning performance of the equalization algorithm with this sequential sampling scheme are now presented.

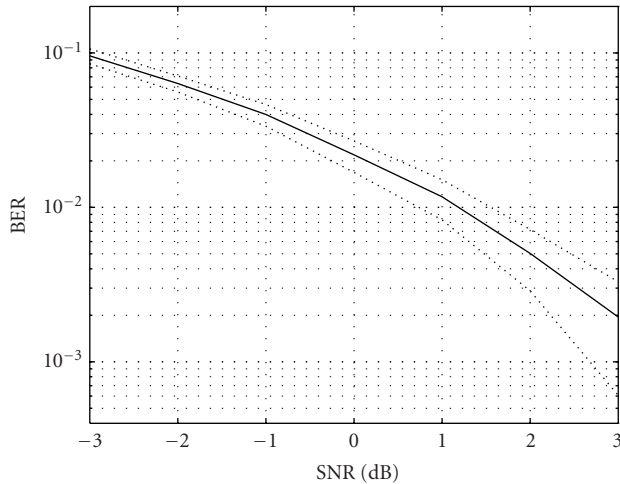


FIGURE 12: Mean \pm standard deviation (straight/dotted lines) of the BER values for MAP estimates of signals versus SNR_r , in dB, model of Figure 1 for $\text{SNR}_e = 12$ dB.

5. NUMERICAL EXPERIMENTS

The equalization method was run for 100 realizations of sequences of 1 000 samples for each value of SNR at the receiver stage and for SNR equal to 12 dB at the emission stage. The number of received samples per symbol rate is $p = 8$. The channel model is depicted in Figure 1.

- The amplifier model is described in Section 2.2 where parameters are set as the first row of Table 1.
- The fading channel is composed of one delayed trajectory of 3 samples ($\Delta = 3T_s$; cf. Section 2.3) and 10 dB attenuated in comparison with the principal trajectory.

For each realization, the emitted symbol sequence is estimated by considering the MAP trajectory computed from a Monte Carlo approximation of the distribution (26) with $M = 50$ particles (27). Weights (29) are computed with the help of Monte Carlo estimation techniques introduced in Section 3.2, considering sequences of 100 samples. In our simulations, the value for threshold (31) $\lambda = 0.1$ gave acceptable estimation results. The mean values (straight lines) and their associated variances (dotted lines) of the BER of MAP estimates are depicted in Figure 12.

The equalization algorithm was also run for different values of uplink SNR: 10 dB and 15 dB. The mean values of the BER computed from the estimated phases are depicted in Figure 13. Curves from the bottom to the top are associated to a decreasing SNR_e .

One of the advantages of the proposed equalization method is its robustness with respect to nonstationarities of the transmission channel. This property comes from the MAP estimation procedure considering the distribution (18) marginalized with respect to channel parameters. Simulations including perturbations of the parameters of the transmission chain lead to similar results to those presented in Figures 12 and 13. On the other hand, in case of dysfunc-

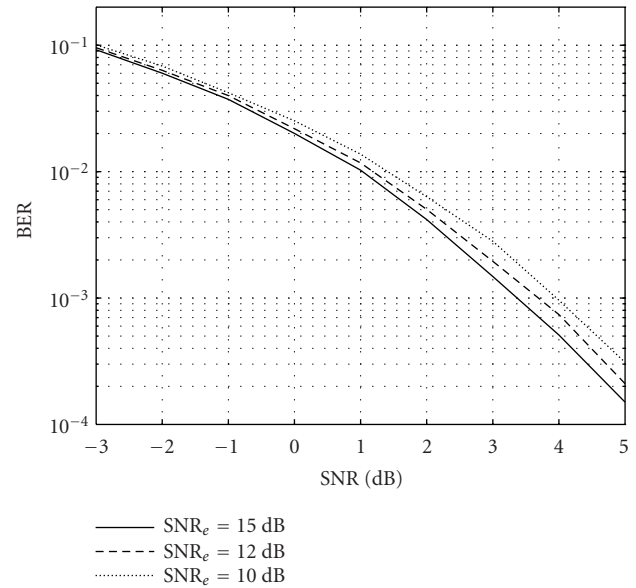


FIGURE 13: Mean BER values for MAP estimates of signals versus SNR_r , in dB, model of Figure 1 for various SNR_e values.

tion in devices of the amplifier and/or of the filters or if sudden change of noise intensities happens during the transmission, the estimation method remains almost insensitive to these changes. The approach currently developed cannot thus be used for diagnostic purposes, as it is the case for certain methods based on neural networks [4]. An interesting hybrid approach is proposed in the conclusion to cope with this task.

In order to compare the performance of the equalization method, BERs computed from the MAP estimates of the symbols are compared with ones obtained with signals estimated by an equalizer built from a 2-10-4 multilayer neural network, using hyperbolic tangents as activation functions [3]. The mean values (straight lines) and standard deviations (dotted lines) of BER computed from Monte Carlo MAP estimates are represented in Figures 14 and 15. The mean values of BER computed from signals estimated with the neural network method are depicted in dashed lines.

The two methods give similar results for this configuration. Nevertheless, an important and interesting characteristic of the sequential Monte Carlo estimation method is that it does not require any learning sequence for equalizing the transmission chain, contrary to approaches based on neural networks. The proposed method is thus efficient in the context of blind communication. A calibration step is at least necessary in order to estimate the number of received samples per symbol rate. This tuning can be realized in a simple manner by computing the correlation of the received samples.

6. CONCLUSION

The particle filtering equalization method proposed in this paper makes it possible to estimate sequentially digital signals

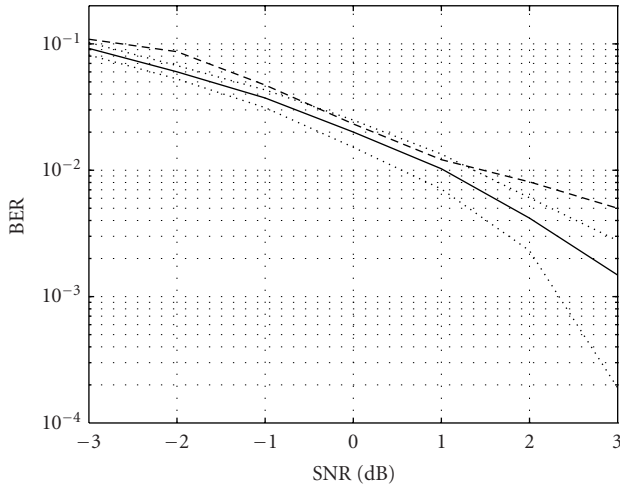


FIGURE 14: Mean \pm standard deviation (straight/dotted lines) of the BER values for MAP estimates of signals versus SNR_r in dB, model of Figure 1 for SNR_e= 15 dB. Means of the BER values for the signals estimated with neural networks are depicted in dashed lines.

transmitted through a satellite communication channel. The approach takes explicitly into account the nonlinearities induced by the amplification stage in the satellite thanks to a Bayesian estimation framework implemented via Monte Carlo simulation methods. This approach enables to estimate recursively the distribution of interest, namely, the *posterior* distribution of the variables, marginalized with respect to the parameters of the transmission chain.

An advantage of this approach is that it is robust to non-stationarities of the channel. On the contrary, the method cannot detect these nonstationarities and thus be employed to predict some dysfunction of transmission devices, as it is the case for certain neural networks approach. Therefore, it seems interesting to use Markov chain and/or sequential Monte Carlo methods to train appropriate neural networks models. This approach was successfully applied in the field of statistical learning, for instance [16, Section 17].

As for many particle-filtering-based methods, the implementation of the sampling algorithm is generally demanding in terms of computing time, if compared to classical nonsimulation approaches, especially for tackling problems arising in the field of digital communication [19].

However, an advantage of the proposed Monte Carlo equalization method is that it does not require the knowledge of any learning input sequence in order to update the equalizer parameters. To the best of our knowledge, it seems at the moment the only method for achieving directly the blind equalization task of such transmission channel. This blind property can be of premium importance in the framework of communication in noncooperative context. This is the case in passive listening, for instance, or when transmission of learning sequences cannot be completed correctly because of intense noises during the transmission.

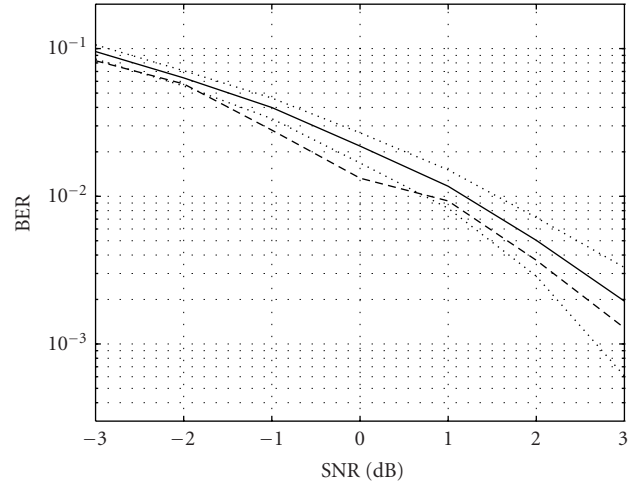


FIGURE 15: Mean \pm standard deviation (straight/dotted lines) of the BER values for MAP estimates of signals versus SNR_r in dB, model of Figure 1 for SNR_e= 15 dB. Means of the BER values for the signals estimated with neural networks are depicted in dashed lines.

APPENDICES

A. MONTE CARLO ESTIMATION OF POSTERIOR DISTRIBUTION $p(\phi|A, \sigma_e, \text{TWT}, \sigma_r, r)$

As stated in Section 3.1, the problem is to compute likelihood (13). A solution consists in integrating this expression with respect to y , the amplified signal (cf. Figure 7):

$$\int p(r, y|A, \phi, \sigma_e, \text{TWT}, \sigma_r) dy. \quad (\text{A.1})$$

From Bayes' formula, this expression is proportional to

$$\int p(r|y, A, \phi, \sigma_e, \text{TWT}, \sigma_r) p(y|A, \phi, \sigma_e, \text{TWT}, \sigma_r) dy. \quad (\text{A.2})$$

As downlink transmission noise is assumed to be Gaussian (cf. (8)), the likelihood is yielding

$$p(r|y, \sigma_r) \propto \exp\left(-\frac{|r-y|^2}{\sigma_r^2}\right). \quad (\text{A.3})$$

The right probability density function in integral (A.2) can be computed by marginalizing it with respect to x , the signal to amplify (cf. Figure 7):

$$\begin{aligned} & p(y|A, \phi, \sigma_e, \text{TWT}, \sigma_r) \\ &= \int p(y, x|A, \phi, \sigma_e, \text{TWT}, \sigma_r) dx \\ &\propto \int p(y|x, A, \phi, \sigma_e, \text{TWT}, \sigma_r) \\ &\quad \times p(x|A, \phi, \sigma_e, \text{TWT}, \sigma_r) dx. \end{aligned} \quad (\text{A.4})$$

The signal y is entirely determined by the signal x from relations (6) and (7). The left probability density function in integral (A.4) thus equals

$$p(y|x, A, \phi, \sigma_e, \text{TWT}, \sigma_r) = \delta(y - \text{TWT}(x)). \quad (\text{A.5})$$

The right probability density function in expression (A.4) yields

$$p(x|A, \phi, \sigma_e) \propto \exp\left(-\frac{|x - A \exp(i\phi)|^2}{\sigma_e^2}\right) \quad (\text{A.6})$$

as the uplink transmission noise is assumed to be Gaussian (cf. (4)). Expression (13) is thus proportional to

$$\int \exp\left(-\frac{1}{\sigma_r^2} |r - \text{TWT}(x)|^2\right) \exp\left(-\frac{|x - A \exp(i\phi)|^2}{\sigma_e^2}\right) dx \quad (\text{A.7})$$

and the above formula can be viewed as an expectation

$$\mathbb{E}\left\{\exp\left(-\frac{1}{\sigma_r^2} |r - \text{TWT}(x)|^2\right)\right\}, \quad (\text{A.8})$$

where

$$x \sim \mathcal{N}_{\text{CC}}(A \exp(i\phi), \sigma_e^2). \quad (\text{A.9})$$

ACKNOWLEDGMENTS

This work was partly supported by the research program BLISS (IST-1999-14190) from the European Commission. The first author is grateful to the Japan Society for the Promotion of Science and the Grant-in-Aid for Scientific Research in Japan for their funding. The authors thank P. Comon and C. Jutten for helpful comments and are grateful to the anonymous reviewers for their helpful suggestions which have greatly improved the presentation of this paper.

REFERENCES

- [1] A. A. M. Saleh, "Frequency-independent and frequency-dependent nonlinear models of TWT amplifiers," *IEEE Trans. Communications*, vol. 29, no. 11, pp. 1715–1720, 1981.
- [2] S. Bouchired, M. Ibnkahla, D. Roviras, and F. Castanié, "Equalization of satellite mobile communication channels using RBF networks," in *Proc. IEEE Workshop on Personal Indoor and Mobile Radio Communication (PIMRC '98)*, Boston, Mass, USA, September 1998.
- [3] G. J. Gibson, S. Siu, and C. F. N. Cowen, "Multilayer perceptron structures applied to adaptive equalisers for data communications," in *Proc. IEEE Int. Conf. Acoustics, Speech, Signal Processing (ICASSP '89)*, vol. 2, pp. 1183–1186, Glasgow, UK, May 1989.
- [4] M. Ibnkahla, N. J. Bershad, J. Sombrin, and F. Castanié, "Neural network modeling and identification of nonlinear channels with memory: algorithms, applications, and analytic models," *IEEE Trans. Signal Processing*, vol. 46, no. 5, pp. 1208–1220, 1998.
- [5] M. Ibnkahla, J. Sombrin, F. Castanié, and N. J. Bershad, "Neural networks for modeling nonlinear memoryless communication channels," *IEEE Trans. Communications*, vol. 45, no. 7, pp. 768–771, 1997.
- [6] C. You and D. Hong, "Blind equalization techniques using the complex multi-layer perceptron," *Proc. IEEE Global Telecommunications Conference (GLOBECOM '96)*, pp. 1340–1344, November 1996.
- [7] S. Prakriya and D. Hatzinakos, "Blind identification of LTI-ZMNL-LTI nonlinear channel models," *IEEE Trans. Signal Processing*, vol. 43, no. 12, pp. 3007–3013, 1995.
- [8] S. Prakriya and D. Hatzinakos, "Blind identification of linear subsystems of LTI-ZMNL-LTI models with cyclostationary inputs," *IEEE Trans. Signal Processing*, vol. 45, no. 8, pp. 2023–2036, 1997.
- [9] G. B. Giannakis and E. Serpedin, "Linear multichannel blind equalizers of nonlinear FIR Volterra channels," *IEEE Trans. Signal Processing*, vol. 45, no. 1, pp. 67–81, 1997.
- [10] G. Kechriotis, E. Zervas, and E. S. Manolakos, "Using recurrent neural networks for adaptive communication channel equalization," *IEEE Transactions on Neural Networks*, vol. 5, no. 2, pp. 267–278, 1994.
- [11] G. M. Raz and B. D. Van Veen, "Blind equalization and identification of nonlinear and IIR systems—a least squares approach," *IEEE Trans. Signal Processing*, vol. 48, no. 1, pp. 192–200, 2000.
- [12] H. Kantz and T. Schreiber, *Nonlinear Time Series Analysis*, vol. 7 of *Cambridge Nonlinear Science Series*, Cambridge University Press, Cambridge, UK, 1997.
- [13] H. Tong, *Non-linear Time Series: a Dynamical System Approach*, vol. 6 of *Oxford Statistical Science Series*, Oxford University Press, Oxford, UK, 1990.
- [14] J. Y. Huang, "On the design of nonlinear satellite CDMA receiver," M.S. thesis, Institute of Communication Engineering, College of Engineering and Computer Science, National Chiao Tung University, Hsinchu, Taiwan, June 1998.
- [15] M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking," *IEEE Trans. Signal Processing*, vol. 50, no. 2, pp. 174–188, 2002.
- [16] A. Doucet, N. De Freitas, and N. Gordon, Eds., "Statistics for engineering and information science," in *Sequential Monte Carlo Methods in Practice*, Springer, New York, NY, USA, 2001.
- [17] N. Gordon, D. Salmond, and A. F. M. Smith, "Novel approach to nonlinear/non-Gaussian Bayesian state estimation," *IEEE proceedings F, Radar and Signal Processing*, vol. 140, no. 2, pp. 107–113, 1993.
- [18] G. Kitagawa, "Monte Carlo filter and smoother for non-Gaussian nonlinear state space models," *Journal of Computational and Graphical Statistics*, vol. 5, no. 1, pp. 1–25, 1996.
- [19] E. Punskeya, A. Doucet, and W. J. Fitzgerald, "On the use and misuse of particle filtering in digital communications," *Proc. European Signal Processing Conference (EUSIPCO '02)*, September 2002.
- [20] A. Doucet, S. Godsill, and C. Andrieu, "On sequential Monte Carlo sampling methods for Bayesian filtering," *Statistics and Computing*, vol. 10, no. 3, pp. 197–208, 2000.
- [21] J. S. Liu and R. Chen, "Sequential Monte Carlo methods for dynamic systems," *Journal of the American Statistical Association*, vol. 93, no. 443, pp. 1032–1044, 1998.

- [22] J. S. Liu and R. Chen, "Blind deconvolution via sequential imputations," *Journal of the American Statistical Association*, vol. 90, no. 430, pp. 567–576, 1995.
- [23] R. Chen, X. Wang, and J. S. Liu, "Adaptive joint detection and decoding in flat-fading channels via mixture kalman filtering," *IEEE Transactions on Information Theory*, vol. 46, no. 6, pp. 2079–2094, 2000.
- [24] P.-O. Amblard, J.-M. Brossier, and E. Moisan, "Phase tracking: what do we gain from optimality? Particle filtering vs. phase-locked loops," *Signal Processing*, vol. 83, no. 1, pp. 151–167, 2003.
- [25] A. V. Oppenheim and R. W. Schaffer, *Discrete-Time Signal Processing*, Prentice-Hall Signal Processing Series, Prentice-Hall, Englewood Cliffs, NJ, USA, 1989.
- [26] C. Meyr, M. Moeneclaey, and S. A. Fechtel, *Digital Communication Receivers*, Wiley Series in Telecommunications and Signal Processing, John Wiley & Sons, New York, NY, USA, 1995.
- [27] J. M. Bernardo and A. F. M. Smith, *Bayesian Theory*, Wiley Series in Probability and Mathematical Statistics, John Wiley & Sons, Chichester, UK, 1994.
- [28] T. Clapp and S. J. Godsill, "Bayesian blind deconvolution for mobile communications," in *Proc. IEE Colloquium on Adaptive Signal Processing for Mobile Communication Systems*, vol. 9, pp. 9/1–9/6, London, UK, October 1997.
- [29] C. P. Robert and G. Casella, *Monte Carlo Statistical Methods*, Springer-Verlag, New York, NY, USA, 1999.
- [30] W. R. Gilks, S. Richardson, and D. J. Spiegelhalter, *Markov Chain Monte Carlo in Practice*, Interdisciplinary Statistics, Chapman & Hall, London, UK, 1996.
- [31] S. Sénécal and P.-O. Amblard, "Blind equalization of a nonlinear satellite system using MCMC simulation methods," *EURASIP Journal on Applied Signal Processing*, vol. 2002, no. 1, pp. 46–57, 2002.
- [32] R. Chen and T. H. Li, "Blind restoration of linearly degraded discrete signals by Gibbs sampling," *IEEE Trans. Signal Processing*, vol. 43, no. 10, pp. 2410–2413, 1995.
- [33] G. Casella and C. Robert, "Rao-blackwellisation of sampling schemes," *Biometrika*, vol. 83, no. 1, pp. 81–94, 1996.
- [34] J. S. Liu, W. H. Wong, and A. Kong, "Covariance structure of the Gibbs sampler with applications to the comparisons of estimators and augmentation schemes," *Biometrika*, vol. 81, no. 1, pp. 27–40, 1994.
- [35] A. Kong, J. S. Liu, and W. H. Wong, "Sequential imputations and Bayesian missing data problems," *Journal of the American Statistical Association*, vol. 89, no. 425, pp. 278–288, 1994.
- [36] D. T. Pham, "Stochastic methods for sequential data assimilation in strongly nonlinear systems," *Monthly Weather Review*, vol. 129, no. 5, pp. 1194–1207, 2001.
- [37] P. Cheung-Mon-Chan and E. Moulines, "Global sampling for sequential filtering over discrete state space," *EURASIP Journal on Applied Signal Processing*, vol. 2004, no. 15, pp. 2242–2254, 2004.

Pierre-Olivier Amblard was born in Dijon, France, in 1967. He received the Ingénieur degree in electrical engineering in 1990 from the École Nationale Supérieure d'Ingénieurs Electriciens de Grenoble, Institut National Polytechnique de Grenoble (ENSIEG, INPG). He received the DEA degree and the Ph.D. degree in signal processing in 1990 and 1994, respectively, both from the INPG. Since 1994, he has been Chargé de Recherches with the Centre National de la Recherche Scientifique (CNRS), and he is working in the Laboratoire des Images et des Signaux (LIS, UMR 5083), where he is in charge of the Groupe Non Linéaire. His research interests include higher-order statistics, nonstationary and nonlinear signal processing, and applications of signal processing in physics.



Laurent Cavazzana graduated from the École Nationale Supérieure d'Informatique et de Mathématiques Appliquées de Grenoble (France) and received a DEA degree in applied mathematics from the Joseph Fourier University (Grenoble, France) in 2003. His areas of interest lie in stochastic modeling, neural networks, and speech processing.



Stéphane Sénécal received the M.S. degree in electrical engineering in 1999 and the Ph.D. degree in statistical signal processing in 2002 from Institut National Polytechnique de Grenoble, France. He is now a Research Associate at the Institute of Statistical Mathematics, Japan. His research interests include stochastic simulation methods and their applications in signal processing and modeling.



Channel Tracking Using Particle Filtering in Unresolvable Multipath Environments

Tanya Bertozzi

Diginext, 45 Impasse de la Draille, 13857 Aix-en-Provence Cedex 3, France
Email: bertozzi@diginext.fr

Conservatoire National des Arts et Métiers (CNAM), 292 rue Saint-Martin, 75141 Paris Cedex 3, France

Didier Le Ruyet

Conservatoire National des Arts et Métiers (CNAM), 292 rue Saint-Martin, 75141 Paris Cedex 3, France
Email: leruyet@cnam.fr

Cristiano Panazio

Conservatoire National des Arts et Métiers (CNAM), 292 rue Saint-Martin, 75141 Paris Cedex 3, France
Email: panazio.cristiano@cnam.fr

Han Vu Thien

Conservatoire National des Arts et Métiers (CNAM), 292 rue Saint-Martin, 75141 Paris Cedex 3, France
Email: vu-thien@cnam.fr

Received 1 May 2003; Revised 9 June 2004

We propose a new timing error detector for timing tracking loops inside the Rake receiver in spread spectrum systems. Based on a particle filter, this timing error detector jointly tracks the delays of each path of the frequency-selective channels. Instead of using a conventional channel estimator, we have introduced a joint time delay and channel estimator with almost no additional computational complexity. The proposed scheme avoids the drawback of the classical early-late gate detector which is not able to separate closely spaced paths. Simulation results show that the proposed detectors outperform the conventional early-late gate detector in indoor scenarios.

Keywords and phrases: sequential Monte Carlo, multipath channels, importance sampling, timing estimation.

1. INTRODUCTION

In wireless communications, direct-sequence spread spectrum (DS-SS) techniques have received an increasing interest, especially for the third generation of mobile systems. In DS-SS systems, the adapted filter typically employed is the Rake receiver. This receiver is efficient to counteract the effects of frequency-selective channels. It is composed of fingers, each assigned to one of the most significant channel paths. The outputs of the fingers are combined proportionally to the power of each path for estimating the transmitted symbols (maximum-ratio combining). Unfortunately, the performance of the Rake receiver strongly depends on the quality of the estimation of the parameters associated with the channel paths. As a consequence, we have to estimate the delay of each path using a timing error detector (TED). This goal is generally achieved in two steps: acquisition and

tracking. During the acquisition phase, the number and the delays of the most significant paths are determined. These delays are estimated within one half chip from the exact delays. Then, the tracking module refines the first estimation and follows the delay variations during the permanent phase. The conventional TED used during the tracking phase is the early-late gate-TED (ELG-TED) associated with each path. It is well known that the ELG-TED works very well in the case of a single fading path. However, in the presence of multipath propagation, the interference between the different paths can degrade its performance. In fact, the ELG-TED cannot separate the individual paths when they are closer than one chip period from the other paths, whereas a discrimination up to $T_c/4$ can still increase the diversity of the receiver (T_c denotes the chip time) [1]. When the difference between the delays of two paths is contained in the interval $0-1.5 T_c$, we are in the presence of unresolvable multipaths. This scenario

corresponds, for example, to the indoor scenario. The problem of unresolvable multipaths has recently been analyzed in [2, 3, 4].

Particle filtering (PF) or sequential Monte Carlo (SMC) methods [5] represent the most powerful approach for the sequential estimation of the hidden state of a nonlinear dynamic model. The solution to this problem depends on the knowledge of the posterior probability density (PPD) of the hidden state given the observations. Except in a few special cases including linear Gaussian system models, it is impossible to analytically calculate a sequential expression of this PPD. It is necessary to adopt numerical approximations. The PF methods give a discrete approximation of the PPD of the hidden state by weighted points or particles which can be recursively updated as new observations become available.

The first main application of the PF methods was target tracking. More recently, these techniques have been successfully applied in communications, including blind equalization in Gaussian [6] and non-Gaussian [7, 8] noises and joint symbol and timing estimation [9]. For a complete survey of the communication problems dealt with using PF methods, see [10].

In this paper we propose to use the PF methods for estimating the delays of the paths in multipath fading channels. Since these methods are based on a joint approach, they provide optimal estimates of the different channel delays. In this way, we can overcome the problem of the adjacent paths which causes the failure of the conventional single-path-tracking approaches in the presence of unresolvable multipaths. Moreover, we will combine the PF-based TED (PF-TED) with a conventional estimator for estimating the amplitudes of the channel coefficients. We will also apply the PF methods to the estimation of the channel coefficients in order to jointly estimate the delays and the coefficients.

This paper is organized as follows. In Section 2, we will introduce the system model. Then in Section 3, we will describe the conventional ELG-TED and the PF-TED. In Section 4, we will present the conventional estimators of the channel coefficients and the application of the PF methods to the joint estimation of the delays and the channel coefficients. In Section 5, we will give simulation results. Finally, we will draw a conclusion in Section 5.

2. SYSTEM MODEL

We consider a DS-SS system sending a complex data sequence $\{s_n\}$. The data symbols are spread by a spreading sequence $\{d_m\}_{m=0}^{N_s-1}$ where N_s is the spreading factor.

The resulting baseband equivalent transmitted signal is given by

$$e(t) = \sum_n s_n \sum_{m=0}^{N_s-1} d_m g(t - mT_c - nT), \quad (1)$$

where T_c and T are respectively the chip and symbol period and $g(t)$ is the impulse response of the root-raised cosine filter with a rolloff factor equal to 0.22 in the case of the universal mobile telecommunications system (UMTS) [11].

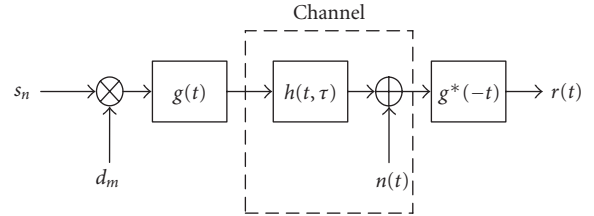


FIGURE 1: Equivalent lowpass transmission system model.

$h(t, \tau)$ denotes the overall impulse response of the multipath propagation channel with L_h independent paths (wide-sense stationary uncorrelated scatterers (WSSUS) model):

$$h(t, \tau) = \sum_{l=1}^{L_h} h_l(t) \delta(\tau - \tau_l(t)). \quad (2)$$

Each path is characterized by its time-varying delay $\tau_l(t)$ and channel coefficient $h_l(t)$.

The signal at the output of the matched filter is given by

$$r(t) = \sum_{l=1}^{L_h} h_l(t) \sum_n s_n \sum_{m=0}^{N_s-1} d_m R_g(t - mT_c - nT - \tau_l(t)) + \tilde{n}(t), \quad (3)$$

where $\tilde{n}(t)$ represents the additive white gaussian noise (AWGN) $n(t)$ filtered by the matched filter and

$$R_g(t) = \int_{-\infty}^{+\infty} g^*(\tau) g(t + \tau) d\tau \quad (4)$$

is the total impulse response of the transmission and receiver filters.

Figure 1 shows the equivalent lowpass transmission model considered in this paper.

The output of the matched filter is used as the input of the Rake receiver. The Rake receiver model is shown in Figure 2. The Rake receiver is composed of L branches corresponding to the L most significant paths. In the l th branch, the received and filtered signal $r(t)$ is sampled at time $mT_c + nT + \hat{\tau}_l$ in order to compensate the timing delay τ_l of the associated path with the estimate $\hat{\tau}_l$. The outputs of each branch are combined to estimate the transmitted symbols. The output of the Rake receiver is given as

$$\hat{s}_n = \hat{s}(nT) = \frac{1}{N_s} \sum_{l=1}^L \hat{h}_l^* \sum_{m=0}^{N_s-1} d_m^* r(mT_c + nT + \hat{\tau}_l). \quad (5)$$

3. THE TIMING ERROR DETECTION

3.1. The conventional TED

The Rake receiver needs good timing delays and channel estimators for each path to extract the most signal power from the received signal and to maximize the signal-to-noise ratio at the output of Rake receiver.

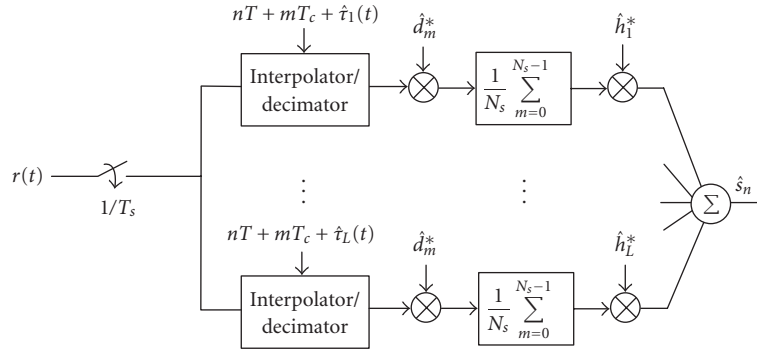


FIGURE 2: Rake receiver model.

The conventional TED for DS-SS systems is the ELG-TED. The ELG-TED is devoted to the tracking of the delay of one path. It is composed of the early and late branches. The signal $r(t)$ is sampled at time $mT_c + nT + \hat{\tau}_l \pm \Delta$. In this paper, we will use $\Delta = T_c/2$. We will restrict ourselves to the coherent ELG-TED where the algorithm uses an estimation of the transmitted data or the pilots when they are available. The output of a coherent ELG-TED associated with the l th path is given by

$$\begin{aligned} x_n &= x(nT) \\ &= \text{Re} \left\{ \hat{s}_n^* h_l^* \sum_{m=nN_s}^{(n+1)N_s-1} \left(r \left(mT_c + \hat{\tau}_l + \frac{T_c}{2} \right) \right. \right. \\ &\quad \left. \left. - r \left(mT_c + \hat{\tau}_l - \frac{T_c}{2} \right) \right) \hat{d}_m^* \right\}. \end{aligned} \quad (6)$$

The main limitation of the ELG-TED is its discrimination capability. Indeed, when the paths are unresolvable (separated by less than T_c), the ELG-TED is not able to correctly distinguish and track the path. This scenario corresponds for example to the indoor case.

These drawbacks motivated the proposed PF-TED.

3.2. The PF-TED

We propose to use the PF methods in order to jointly track the delay of each individual path of the channel. We assume that the acquisition phase has allowed us to determine the number of the most significant paths and to roughly estimate their delay.

The PF methods are used to sequentially estimate time-varying quantities from measures provided by sensors. In general, the physical phenomenon is represented by a state space model composed of two equations: the first describes the evolution of the unknown quantities called hidden state (*evolution equation*) and the second the relation between the measures called observations and the hidden state (*observation equation*). Given the initial distribution of the hidden state, the estimation of the hidden state at time t based on the observations until time t is known as Bayesian inference or Bayesian filtering. This estimation can be obtained through

the knowledge of two distributions: the PPD of the sequence of hidden states from time 1 to time t given the corresponding sequence of observations and the marginal distribution of the hidden state at time t given the sequence of the observations until time t . Except in a few special cases including linear Gaussian state space models, it is impossible to analytically calculate these distributions. The PF methods provide a discrete and sequential approximation of the distributions. It can be updated when a new observation is available, without reprocessing the previous observations. The support of the distributions is discretized by particles, which are weighted samples evolving in time.

Tracking the delay of the individual channel paths can be interpreted as a Bayesian inference. The delays are the hidden state of the system and the model (3) of the received samples relating the observations to the delays represents the observation equation. We notice that this equation is nonlinear with respect to the delays and as a consequence, we cannot analytically estimate the delays. To overcome this nonlinearity, we propose to apply the PF methods.

The PF methods have previously been applied for the delay estimation in DS-CDMA systems [12, 13]. In [12], the PF methods are used to jointly estimate the data, the channel coefficients, and the propagation delay. In [13], the PF methods are combined with a Kalman filter (KF) to respectively estimate the delay propagation and the channel coefficients; the information symbols are assumed known, provided by a Rake receiver. In both papers, the delays of each channel path are considered known and multiple of the sampling time; therefore, only the propagation delay is estimated. In this paper, the approach is different. We suppose that each channel path has a slow time-varying delay, unknown at the receiver. This environment can represent an indoor wireless communication. We assume that the information symbols are known or have been estimated essentially for three reasons:

- (i) the computational complexity of the receiver should be reduced;
- (ii) the channel estimation is typically performed transmitting known pilot symbols, for example using a specific channel as the common pilot channel (CPICH) of the UMTS;

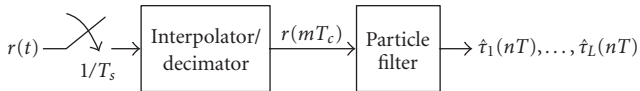


FIGURE 3: Structure of the proposed PF-TED.

- (iii) the PF methods applied to the estimation of the information symbols perform slightly worse than simple deterministic algorithms [12, 14].

Firstly, we will apply the PF methods only to the estimation of the delays of each channel path, considering that the channel coefficients are known. In the next paragraph, we will introduce the estimation of the channel coefficients.

The structure of the proposed PF-TED is shown in Figure 3. This estimator operates on samples from the matched filter output taken at an arbitrary sampling rate $1/T_s$ (at least Nyquist sampling). Then, the samples are processed by means of interpolation and decimation in order to obtain intermediate samples at the chip rate $1/T_c$. These samples are the input of the particle filter. In order to reduce the computational complexity of the PF-TED and since the time variation of the delays is slow with respect to the symbol duration, we choose that the particle filter works at the symbol rate $1/T$. Moreover, in order to exploit all the information contained in the chips of a symbol period, the equations of the PF algorithm are modified. The PF algorithm proposed in this paper is thus the adaptation of the PF methods to a DS-SS system.

Following [15], the evolution of the delays of the channel paths can be described as a first-order autoregressive (AR) process:

$$\begin{aligned} \tau_{1,n} &= \alpha_1 \tau_{1,n-1} + \nu_{1,n}, \\ &\vdots \\ \tau_{L,n} &= \alpha_L \tau_{L,n-1} + \nu_{L,n}, \end{aligned} \quad (7)$$

where $\tau_{l,n}$ for $l = 1, \dots, L$ denotes the delay of the l th channel path at time n , $\alpha_1, \dots, \alpha_L$ express the possible time variation of the delays from a time to the next one, and ν_1, \dots, ν_L are AWGN with zero mean and variance σ_ν^2 . Note that the time index n is an integer multiple of the symbol duration.

The estimation of the delays can be achieved using the minimum mean square error (MMSE) method or the maximum a posteriori (MAP) method. The MMSE solution is given by the following expectation:

$$\hat{\tau}_n = E[\tau_n | r_{1:n}], \quad (8)$$

where $\tau_n = \{\tau_{1,n}, \dots, \tau_{L,n}\}$ and $r_{1:n}$ is the sequence of received samples from time 1 to n . The calculation of (8) involves the knowledge of the marginal distribution $p(\tau_n | r_{1:n})$. Unlike the MMSE solution that yields an estimate of the delays at each time, the MAP method provides the estimate of the hidden state sequence $\tau_{1:n} = \{\tau_1, \dots, \tau_n\}$:

$$\hat{\tau}_{1:n} = \arg \max_{\tau_{1:n}} p(\tau_{1:n} | r_{1:n}). \quad (9)$$

The calculation of (9) requires the knowledge of the PPD $p(\tau_{1:n} | r_{1:n})$.

The simulations give similar results for the MMSE method and the MAP method. Hence, we choose to adopt the MMSE solution as in [9]. In order to obtain samples from the marginal distribution, we use the sequential importance sampling (SIS) approach [16]. Applying the definition of the expectation, (8) can be expressed as follows:

$$\hat{\tau}_n = \int \tau_n p(\tau_n | r_{1:n}) d\tau_n. \quad (10)$$

The aim of the SIS technique is to approximate the marginal distribution $p(\tau_n | r_{1:n})$ by means of weighted particles:

$$p(\tau_n | r_{1:n}) \approx \sum_{i=1}^{N_p} \tilde{w}_n^{(i)} \delta(\tau_n - \tau_n^{(i)}), \quad (11)$$

where N_p is the number of particles, $\tilde{w}_n^{(i)}$ is the normalized importance weight at time n associated with the particle i , and $\delta(\tau_n - \tau_n^{(i)})$ denotes the Dirac delta centered in $\tau_n = \tau_n^{(i)}$.

The phases of the PF-TED based on the SIS approach are summarized below.

(1) *Initialization*. In this paper, we apply the PF methods for the tracking phase, assuming that the number of the channel paths and the initial value of the delay for each path have been estimated during the acquisition phase [17]. We assume that the error on the delay estimated by the acquisition phase belongs to the interval $(-T_c/2, T_c/2)$. Hence, the a priori probability density $p(\tau_0)$ can be considered uniformly distributed in $(\hat{\tau}_0 - T_c/2, \hat{\tau}_0 + T_c/2)$, where $\hat{\tau}_0$ is the delay provided by the acquisition phase. Note that the PF methods can be used also for the acquisition phase. However, the number of particles has to be increased, because we have no a priori information on the initial value of the delays.

(2) *Importance sampling*. The time evolution of the particles is achieved with an importance sampling distribution. When r_n is observed, the particles are drawn according to the importance function. In general, the importance function is chosen to minimize the variance of the importance weights associated with each particle. In fact, it can be shown that the variance of the importance weights can only increase stochastically over time [16]. This means that, after a few iterations of the SIS algorithm, only one particle has a normalized weight almost equal to 1 and the other weights are very close to zero. Therefore, a large computational effort is devoted to updating paths with almost no contribution to the final estimate. In order to avoid this behavior, a resampling phase of the particles is inserted among the recursions of the SIS algorithm. To limit this degeneracy phenomenon, we need to use the optimal importance function [16], given by

$$\pi(\tau_n^{(i)} | \tau_{1:n-1}^{(i)}, r_{1:n}) = p(\tau_n^{(i)} | \tau_{n-1}^{(i)}, r_n). \quad (12)$$

Unfortunately, the optimal importance function can be analytically calculated only in a few cases, including the class of models represented by a Gaussian state space model with linear observation equation. In this case, the observation equation (3) is nonlinear and thus, the optimal importance function cannot be analytically determined. We can consider two solutions to this problem [16]:

- (i) the a priori importance function $p(\tau_n^{(i)} | \tau_{n-1}^{(i)})$;
- (ii) an approximated expression of the optimal importance function by linearization of the observation equation about $\tau_{l,n}^{(i)} = \alpha_l \tau_{l,n-1}^{(i)}$ for $l = 1, \dots, L$.

Since the second solution involves the derivative calculation of the nonlinear observation equation, and hence very complex operations, we choose the a priori importance function as in [9]. Considering that the noises $v_{l,n}$ for $l = 1, \dots, L$ in (7) are Gaussian, the importance function for each delay l is a Gaussian distribution with mean $\alpha_l \tau_{l,n-1}^{(i)}$ and variance σ_v^2 .

(3) *Weight update.* The evaluation of the importance function for each particle at time n enables the calculation of the importance weights [16]:

$$w_n^{(i)} = w_{n-1}^{(i)} \frac{p(r_n | \tau_n^{(i)}) p(\tau_n^{(i)} | \tau_{n-1}^{(i)})}{\pi(\tau_n^{(i)} | \tau_{1:n-1}^{(i)}, r_{1:n})}. \quad (13)$$

This expression represents the calculation of the importance weights if we only consider the samples of the received signal at the symbol rate. However, in a DS-SS system we have additional information provided by N_s samples for each symbol period due to the spreading sequence. Consequently, we modify (13) taking into account the presence of a spreading sequence. Indeed, observing that the received samples are independent, the probability density $p(r_n | \tau_n^{(i)})$ at the symbol rate can be written as

$$p(r_n | \tau_n^{(i)}) = \prod_{m=nN_s}^{(n+1)N_s-1} p(r_m | \tau_n^{(i)}). \quad (14)$$

Considering (3) at the chip rate and recalling the assumptions of known symbols, the probability density $p(r_m | \tau_n^{(i)})$ is Gaussian. Typically, the received sample r_m is complex. For the calculation of the Gaussian distribution, we can write r_m as a bidimensional vector with components being the real part and the imaginary part of r_m . The probability density $p(r_m | \tau_n^{(i)})$ is thus given by

$$p(r_m | \tau_n^{(i)}) = \frac{1}{\pi \sigma_n^2} \exp \left\{ -\frac{1}{\sigma_n^2} |r_m - \mu_m^{(i)}|^2 \right\}, \quad (15)$$

where σ_n^2 is the variance of the AWGN $\tilde{n}(t)$ in (3) and the mean $\mu_m^{(i)}$ is obtained by

$$\mu_m^{(i)} = \sum_{l=1}^L h_{l,n} s_n \sum_{k=m-3}^{m+3} d_k R_g(mT_c - kT_c - nT - \tau_{l,n}^{(i)}). \quad (16)$$

In order to reduce the computational complexity of the PF-TED, in (16) we have assumed that the contribution of the raised cosine filter R_g to the sum on the spreading sequence is limited to the previous 3 and next 3 samples. By substitution of (15) in (14), the latter becomes

$$p(r_n | \tau_n^{(i)}) = \left(\frac{1}{\pi \sigma_n^2} \right)^{N_s} \exp \left\{ -\frac{1}{\sigma_n^2} \sum_{m=nN_s}^{(n+1)N_s-1} |r_m - \mu_m^{(i)}|^2 \right\}. \quad (17)$$

Assuming the a priori importance function, (13) yields

$$\begin{aligned} w_n^{(i)} &= w_{n-1}^{(i)} p(r_n | \tau_n^{(i)}) \\ &= w_{n-1}^{(i)} \left(\frac{1}{\pi \sigma_n^2} \right)^{N_s} \exp \left\{ -\frac{1}{\sigma_n^2} \sum_{m=nN_s}^{(n+1)N_s-1} |r_m - \mu_m^{(i)}|^2 \right\}. \end{aligned} \quad (18)$$

Finally, the importance weights in (18) are normalized using the following expression:

$$\tilde{w}_n^{(i)} = \frac{w_n^{(i)}}{\sum_{j=1}^{N_p} w_n^{(j)}}. \quad (19)$$

(4) *Estimation.* By substitution of (11) into (10), we obtain at each time the MMSE estimate:

$$\hat{\tau}_n = \sum_{i=1}^{N_p} \tilde{w}_n^{(i)} \tau_n^{(i)}. \quad (20)$$

(5) *Resampling.* This algorithm presents a degeneracy phenomenon. After a few iterations of the algorithm, only one particle has a normalized weight almost equal to 1 and the other weights are very close to zero. This problem of the SIS method can be eliminated with a resampling of the particles. A measure of the degeneracy is the effective sample size N_{eff} , estimated by

$$\hat{N}_{\text{eff}} = \frac{1}{\sum_{i=1}^{N_p} (\tilde{w}_n^{(i)})^2}. \quad (21)$$

When \hat{N}_{eff} is below a fixed threshold N_{thres} , the particles are resampled according to the weight distribution [16]. After each resampling task, the normalized weights are initialized to $1/N_p$.

4. THE ESTIMATION OF THE CHANNEL COEFFICIENTS

4.1. The conventional estimators

Channel estimation is performed using the known pilot symbols. If we suppose that the channel remains almost unchanged during the slot, the conventional estimator of the channel coefficients of the l th path is obtained by correlation using the known symbols [18]:

$$\hat{h}_l = \frac{1}{N_{\text{pilot}} N_s} \sum_{n=0}^{N_{\text{pilot}}-1} \sum_{m=0}^{N_s-1} s_n^* d_m^* r(mT_c + nT + \hat{\tau}_{l,n}), \quad (22)$$

where N_{pilot} is the number of pilots in a slot. For each path, the received signal is sampled at time $mT_c + nT + \hat{\tau}_{l,n}$ in order to compensate its delay. Then the samples are multiplied by the despread sequence and summed on the whole sequence of pilot symbols. The problem of this estimator is that when the delays are unresolvable, the estimation becomes biased. To eliminate this bias, we can use an estimator based on the maximum likelihood (ML) criterion. In [1, 19], a simplified version of the ML estimation is proposed. The channel coefficients which maximize the ML criterion are given by

$$\hat{\mathbf{h}} = \mathbf{P}^{-1} \mathbf{a}, \quad (23)$$

where $\hat{\mathbf{h}} = (\hat{h}_1, \dots, \hat{h}_L)$, \mathbf{P} is an $L \times L$ matrix with elements $P_{ij} = R_g(\tau_{i,n} - \tau_{j,n})$, and \mathbf{a} is the vector of the channel coefficients calculated using (22).

4.2. The PF-based joint estimation of the delays and the channel coefficients

We can apply the PF methods to jointly estimate the delays of each path and the channel coefficients with a very low additional cost in terms of computational complexity. This is a suboptimal solution, since the observation equation (3) is linear and Gaussian with respect to the channel coefficients. The optimal solution is represented by a KF. However, combining the PF methods and the KF to jointly estimate the delays and the channel coefficients involves the implementation of a KF. It is better to use the particles employed for the delay estimation and to associate to each particle the estimation of the channel coefficients.

In this case, the hidden state is composed of the L delays and the L channel coefficients of each individual path. When a particle evolves in time, its new position is thus determined by the evolution of the delays and the evolution of the channel coefficients. The delays evolve as described for the PF-TED. For the channel coefficients, we assume that the time variations are slow as, for example, in indoor environments. Hence, the evolution of the channel coefficients can be expressed by the following first-order AR model:

$$\begin{aligned} h_{1,n} &= \beta_1 h_{1,n-1} + z_{1,n}, \\ &\vdots \\ h_{L,n} &= \beta_L h_{L,n-1} + z_{L,n}, \end{aligned} \quad (24)$$

where β_1, \dots, β_L describe the possible time variation of the channel coefficients from a time to the next one and z_1, \dots, z_L are AWGN with zero mean and variance σ_z^2 . The parameters of the channel AR model (24) are chosen according to the Doppler spread of the channel [20]. Notice that this joint estimator operates at the symbol rate as the PF-TED.

As for the delays, we only consider the MMSE method for the estimation of the channel coefficients and we use the a priori importance function:

$$\pi(h_n^{(i)} | h_{1:n-1}^{(i)}, r_{1:n}) = p(h_n^{(i)} | h_{n-1}^{(i)}), \quad (25)$$

where $h_n = \{h_{1,n}, \dots, h_{L,n}\}$. Considering that the noises $z_{l,n}$ for $l = 1, \dots, L$ in (24) are Gaussian, the importance function for the channel coefficients is a Gaussian distribution with mean $\beta_l h_{l,n-1}^{(i)}$ and variance σ_z^2 . To determine the positions of the particles at time n from the positions at time $n-1$, each particle is drawn according to $p(\tau_n^{(i)} | \tau_{n-1}^{(i)})$ and (25).

The calculation of the importance weights is very similar to the case of the PF-TED. The only difference is that the channel coefficients $h_{l,n}$ are replaced by the support of the particles $h_{l,n}^{(i)}$ to calculate the mean (16).

5. SIMULATION RESULTS

In this section, we will compare the performance of the conventional ELG-TED and the PF-TED. In order to demonstrate the gain achieved using the latter, we will consider different indoor scenarios with a two-path Rayleigh channel with the same average power on each path and a maximum Doppler frequency of 19 Hz corresponding to a mobile speed of 10 Km/h for a carrier frequency of 2 GHz. The simulation setup is compatible with the UMTS standard. In these conditions, the time variations of the channel delays can be expressed by the model (7), with $\alpha_1 = \dots = \alpha_L = 0.99999$ and $\sigma_v^2 = 10^{-5}$ [15]. Moreover, the time variations of the channel coefficients can be represented by the model (24), $\beta_1 = \dots = \beta_L = 0.999$ and $\sigma_z^2 = 10^{-3}$.

In these simulations, a CPICH is used. In each slot of CPICH, 40 pilot symbols equal to 1 are expanded into a chip level by a spreading factor of 64. The spreading sequence is a PN sequence changing at each symbol.

5.1. Tracking performance

We assume that the channel coefficients are known to evaluate the TED's tracking capacity and the simulation time is equal to 0.333 second, corresponding to 500 slots. We have firstly considered the delays of the two paths varying according to the following model:

$$\begin{aligned} \tau_{1,n} &= \alpha_1 \tau_{1,n-1} + \nu_{1,n}, \\ \tau_{2,n} &= \alpha_2 \tau_{2,n-1} + \nu_{2,n}, \end{aligned} \quad (26)$$

where $\alpha_1 = \alpha_2 = 0.999$, $\sigma_{\nu,1}^2 = \sigma_{\nu,2}^2 = 0.001$, $\tau_{1,0} = 0$, and $\tau_{2,0} = 1$.

Figure 4 shows one realization of the considered delays and the tracking performance of two ELG-TEDs used for the estimation of the two delays. We assume that $E_s/N_0 = 10$ dB, where E_s is the energy per symbol and N_0 is the unilateral spectral power density. The classical ELG-TED presents difficulties to follow the time variation of the two delays, especially when the delay separation becomes less than $1 T_c$.

However, it is very important for the TED to distinguish the different paths of the channel to enable the Rake receiver to exploit the diversity contained in the multipath nature of the channel. In [1], it has been shown that the gain in diversity decreases as the separation between the paths decreases. In particular, a loss of 2.5 dB in the performance of the matched filter bound for a BER equal to 10^{-2} , passing

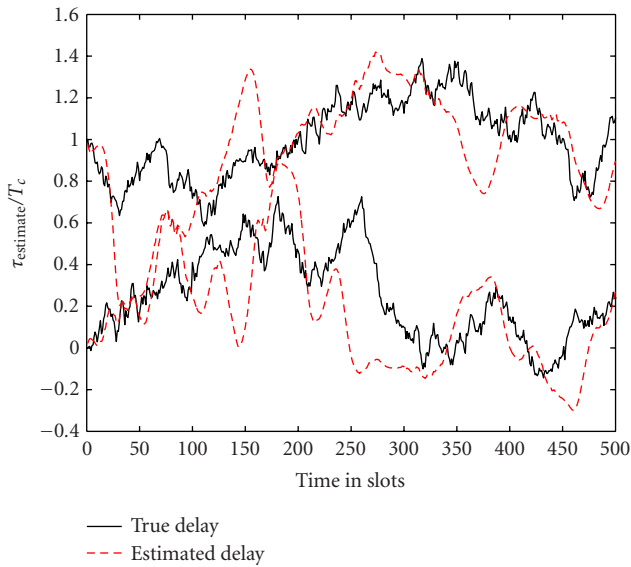


FIGURE 4: Delay tracking with the conventional ELG-TED.

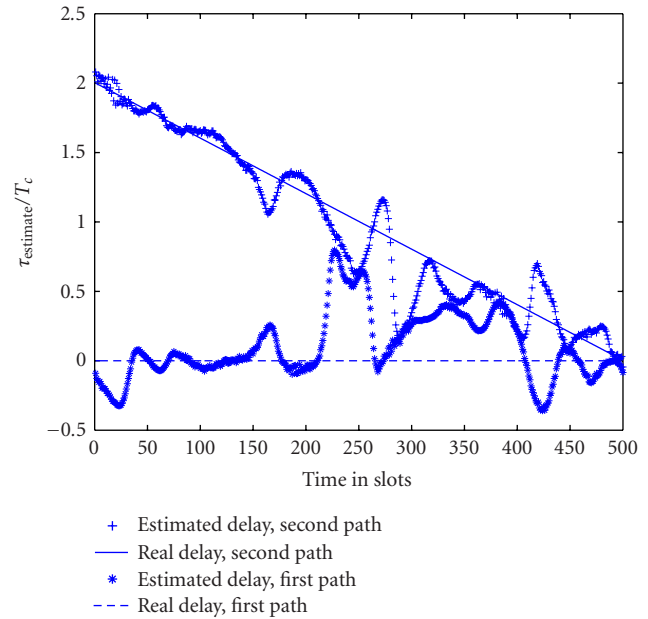


FIGURE 6: Delay tracking with the conventional ELG-TED.

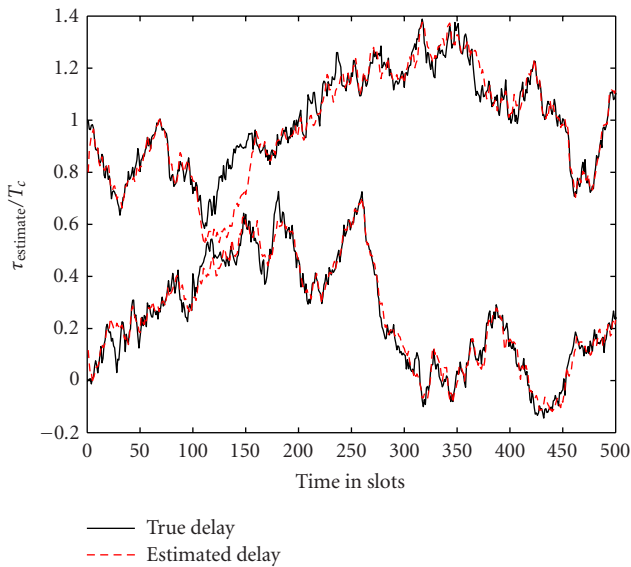


FIGURE 5: Delay tracking with the PF-TED.

from T_c to $T_c/4$, has been observed. Moreover, it has been noted that an interesting gain in diversity occurs if the TED distinguishes paths separated by more than $T_c/4$. On the other hand, it has been found that the performance of the matched filter bound for a separation of $T_c/8$ is very close to the one obtained with only one path. Consequently, the TED discrimination capacity has to be equal to $T_c/4$. Unfortunately, the ELG-TED fails to distinguish all the paths with a delay separation less than $1 T_c$. In Figure 5, we can observe how the discrimination capacity of the TED can be improved using the PF methods.

In order to better highlight this behavior, we have fixed the delay of the first path at 0 and the delay of the second path is decreasing linearly from $2T_c$ to 0 over a simulation time of 0.333 second corresponding to 500 slots. We assume that $E_s/N_0 = 10$ dB, where E_s is the energy per symbol and N_0 is the unilateral spectral power density.

Firstly, we consider that the channel coefficients are known to evaluate the TED's tracking capacity. Figure 6 gives a representative example of the evolution of the two estimated delays using two ELG-TEDs. As soon as the difference between the two delays is lower than $1 T_c$, due to the correlation between the two paths, the estimated delays tend to oscillate around each real delay. The ELG-TEDs are no longer able to perform the correct tracking of the delays. On the other hand, as shown in Figure 7, the proposed PF-TED is able to track almost perfectly the two paths. These results have been obtained using a particle filter with only 10 particles.

Then, we have introduced the estimation of the channel coefficients into the TED. Figure 8 shows the results obtained with two ELG-TEDs combined with the conventional estimator based on the correlation. As soon as the difference between the two delays is lower than $1 T_c$, the detectors no longer recognize the two paths: the weaker path merges with the stronger one.

In Figure 9, the PF-TED is also associated with the conventional estimator of the channel coefficients based on the correlation. When the delay of the second path becomes less than $1 T_c$, the channel estimator decreases its capacity to track the time variations of the channel coefficients and the PF-TED cannot track the delays of the two paths. To improve the channel estimation, we associate the PF-TED with the ML estimator, as shown in Figure 10. In this case, the PF-TED can track the delay of the second path up to $T_c/2$.

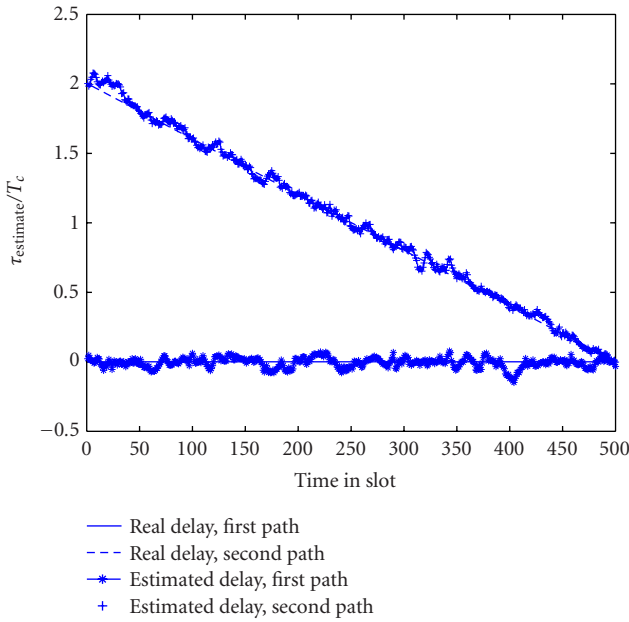


FIGURE 7: Delay tracking with the PF-TED.

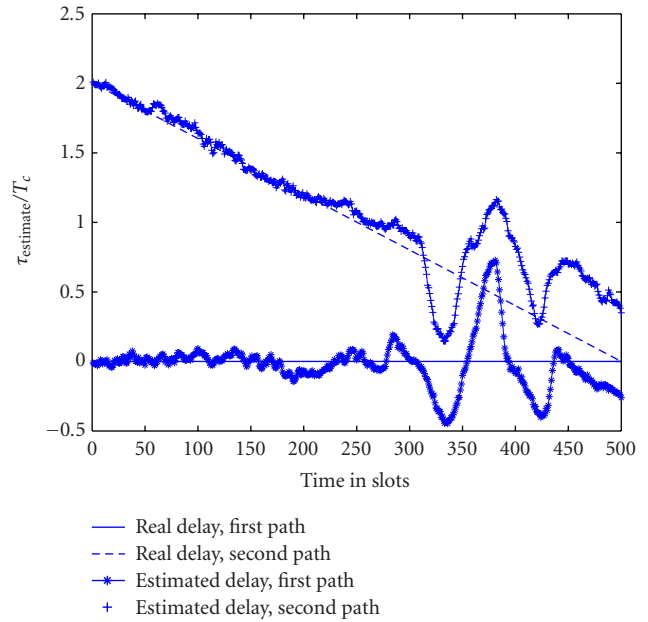


FIGURE 9: Delay tracking with the PF-TED associated with a conventional channel coefficient estimator based on the correlation.

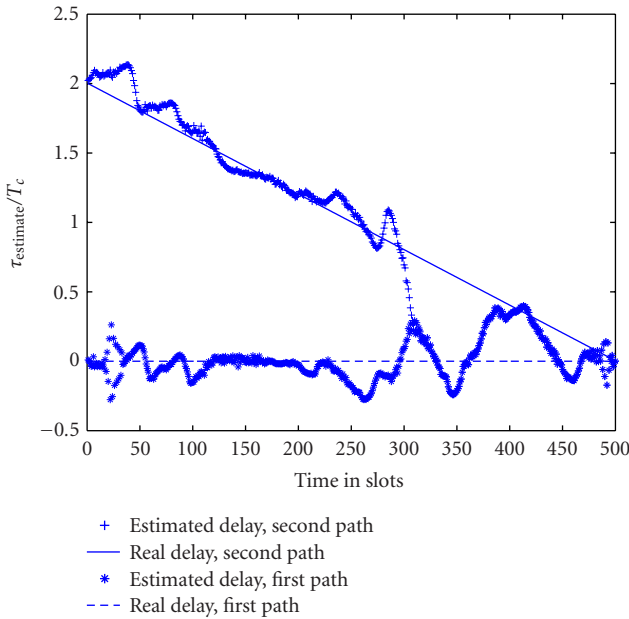


FIGURE 8: Delay tracking with the conventional ELG-TED associated with a conventional channel coefficient estimator based on the correlation.

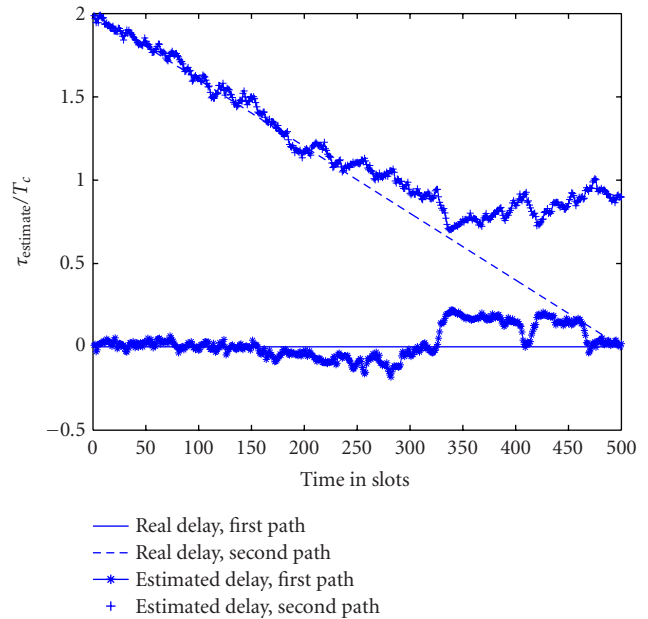


FIGURE 10: Delay tracking with the PF-TED associated with a conventional channel coefficient estimator based on the ML.

For smaller delays, the PF-TED continues to distinguish the two paths, but it cannot follow the time variations of the second delay. The delay of the second path remains close to the values estimated at $T_c/2$.

Using the PF methods to jointly estimate the delays and the channel coefficients, we can notice in Figure 11 that the PF-TED can track the time variations of the second path. This solution implies only a low additional cost in terms of

computational complexity with respect to the PF-TED, since it exploits the set of particles used for the delay estimation for the channel coefficient estimation.

5.2. Mean square error of the delay estimators

In this section, we will compare the estimation of the mean square error (MSE) estimating τ_n of the ELG-TED and the PF-TED with the lower posterior Cramer-Rao bound

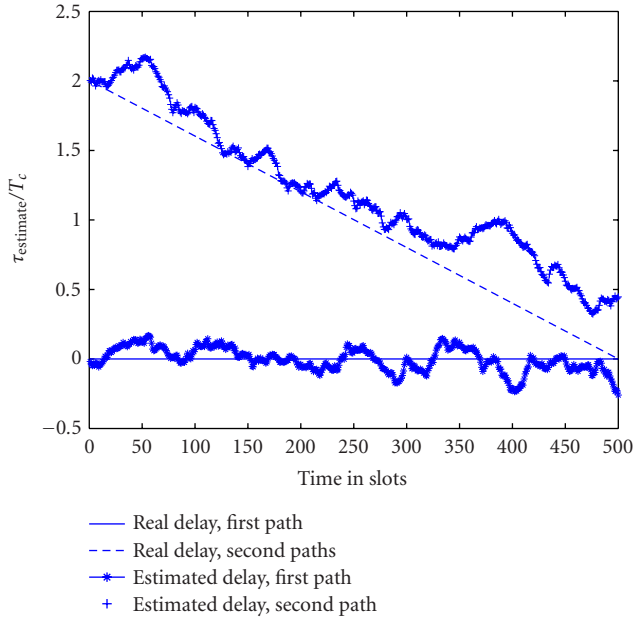


FIGURE 11: Delay tracking with a joint delay and channel coefficient estimator based on the PF methods.

(PCRB). In the Bayesian context of this paper, the PCRB [21] is more suitable than the Cramer-Rao bound [22] to evaluate the MSE of varying unknown parameters.

The PCRB for estimating τ_n using $r_{1:n}$ has the form

$$E(\hat{\tau}_n - \tau_n)^2 \geq J_{n,n}^{-1}, \quad (27)$$

where $J_{n,n}$ is the right lower element of the $n \times n$ Fisher information matrix.

In [21], the authors have shown how to recursively evaluate $J_{n,n}$. For our application, the nonlinear filtering system is

$$\begin{aligned} \tau_{n+1} &= \alpha \tau_n + v_n, \\ r_n &= z_n(\tau_n) + \tilde{n}_n, \end{aligned} \quad (28)$$

where the second relation represents the nonlinear observation equation (3) at chip rate.

Since the spreading sequence is different at each chip time, we have to evaluate $z_n(\tau_n)$ at this rate.

From the general recursive equation given in [21], the sequence $\{J_{n,n}\}$ can be obtained as follows:

$$\begin{aligned} J_{n+1,n+1} &= \sigma_v^{-1} + E[\nabla_{\tau_{n+1}} z_{n+1}(\tau_{n+1})]^2 \sigma_n^{-1} \\ &\quad - (\alpha \sigma_v^{-1})^2 (J_{n,n} + \alpha^2 \sigma_v^{-1})^{-1}. \end{aligned} \quad (29)$$

In order to calculate $E[\nabla_{\tau_{n+1}} z_{n+1}(\tau_{n+1})]$, we have applied a Monte Carlo evaluation. We generate M i.i.d. state trajectories of a given length N_t $\{\tau_0^i, \tau_1^i, \dots, \tau_{N_t}^i\}$ with $1 \leq i \leq M$ by simulating the system model defined in (28) starting from an initial state τ_0 drawn from the a priori probability density $p(\tau_0)$. For the calculation, we fixed $M = 100$.

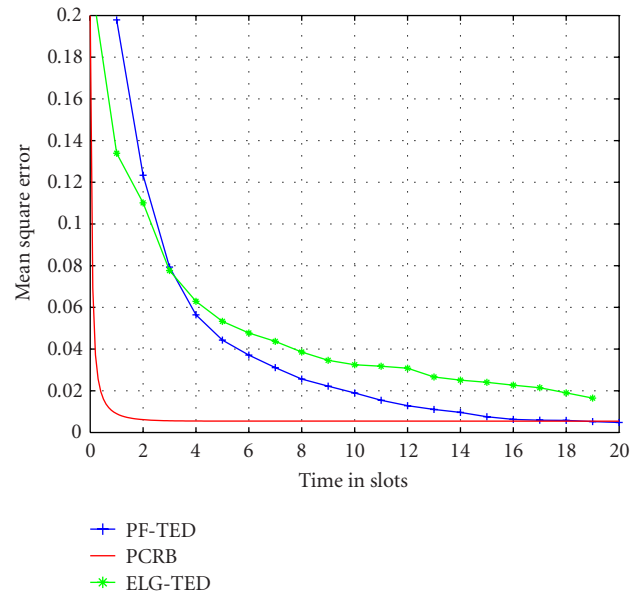


FIGURE 12: Comparison of the PCRB with the MSE estimating τ_n of ELG-TED and PF-TED.

In Figure 12, we show the comparison of the PCRB with the MSE estimating τ_n of the ELG-TED and PF-TED. For both algorithms, we use a uniform initial pdf $p(\tau_0)$. For the PF-TED, the 10 particles were initialized uniformly in the interval $\{-T_c/2, T_c/2\}$. The signal-to-noise ratio E_s/N_0 was fixed to 10 dB. We can see in Figure 12 that the PF-TED outperforms the ELG-TED and reaches the PCRB bound after 15 slots. The slow convergence of the ELG-TED and PF-TED compared to the PCRB can be explained since the two TEDs are updated at each symbol while the PCRB bound is calculated for each chip.

5.3. Performance evaluation

Figure 13 shows the BERs versus E_s/N_0 considering a two-path channel with the same average power on each path. The delays of the first and second paths were respectively fixed at 0 and $1 T_c$. The same maximum Doppler frequency as above was used. The BER values have been averaged over 50 000 bits.

When using two ELG-TEDs, except when the channel is known, the performance is very poor compared to the maximum achievable performance (known delays and channel coefficients). On the other hand, the PF-TED with channel coefficients known or estimated reaches the optimal performance. We can conclude that the considered TED must be able to separate the different paths of the channel, otherwise the performance of the Rake receiver breaks down.

6. CONCLUSIONS

In this paper we have proposed to use the PF methods in order to track the delay of the different channel paths. We have assumed that an acquisition phase has already provided an initial estimation of these delays.

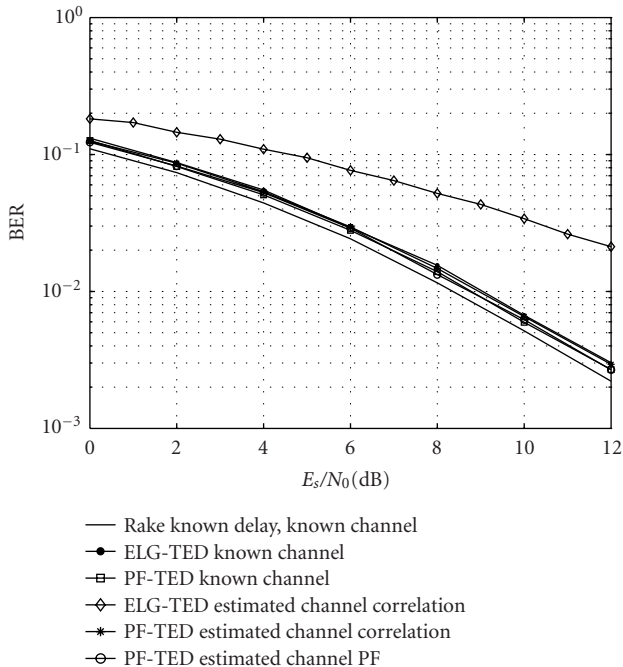


FIGURE 13: Performance comparison of the ELG-TED and the PF-TED.

We have firstly considered that the channel coefficients are known. We have compared the tracking capacity of the conventional ELG-TED and the proposed PF-TED. We have shown that when the delays of the channel paths become very close (less than $1 T_c$), the ELG-TED is unable to track the time variations of the delays. However, the PF-TED continues to track the delays.

We have introduced the channel coefficient estimation to the TED. We have considered two classical methods: the estimation based on the correlation using pilot symbols and the estimation based on the ML criterion. We have shown that the ELG-TED with estimation of the channel coefficients loses the capacity to distinguish the paths when the delays are very closed. On the other hand, the PF-TED associated with the classical two-channel estimator is able to separate the different paths. However, for very close delays the channel estimators prevent the PF-TED from tracking the time variations of the delays. We have proposed to estimate jointly the delays and the channel coefficients using the PF methods to avoid this loss of tracking. We have found that the joint estimation enables a better tracking of the delays.

Finally, we have seen that it is very important for the Rake receiver that the TED can distinguish the different paths of the channel. We have observed that in the case of unresolvable paths, the ELG-TED confuses the paths and the performance of the Rake receiver is very poor.

As a conclusion, we can say that the PF-TED based on the joint estimation of the delays and the channel coefficients can be a good substitute of the classical ELG-TED, specially for indoor wireless communications. Moreover, the computational complexity of the PF-TED is very limited, since we have used only 10 particles.

REFERENCES

- [1] H. Boujemaa and M. Siala, "Rake receivers for direct sequence spread spectrum systems," *Ann. Telecommun.*, vol. 56, no. 5-6, pp. 291-305, 2001.
- [2] V. Aue and G. P. Fettweis, "A non-coherent tracking scheme for the RAKE receiver that can cope with unresolvable multipath," in *Proc. IEEE International Conference on Communications (ICC '99)*, pp. 1917-1921, Vancouver, BC, Canada, June 1999.
- [3] R. De Gaudenzi, "Direct-sequence spread-spectrum chip tracking in the presence of unresolvable multipath components," *IEEE Trans. Vehicular Technology*, vol. 48, no. 5, pp. 1573-1583, 1999.
- [4] G. Fock, J. Baltersee, P. Schulz-Rittich, and H. Meyr, "Channel tracking for RAKE receivers in closely spaced multipath environments," *IEEE Journal on Selected Areas in Communications*, vol. 19, no. 12, pp. 2420-2431, 2001.
- [5] A. Doucet, J. F. G. de Freitas, and N. J. Gordon, Eds., *Sequential Monte Carlo Methods in Practice*, Springer, New York, NY, USA, 2001.
- [6] J. S. Liu and R. Chen, "Blind deconvolution via sequential imputations," *Journal of the American Statistical Association*, vol. 90, no. 430, pp. 567-576, 1995.
- [7] R. Chen, X. Wang, and J. S. Liu, "Adaptive joint detection and decoding in flat-fading channels via mixture Kalman filtering," *IEEE Transactions on Information Theory*, vol. 46, no. 6, pp. 2079-2094, 2000.
- [8] E. Punskeya, C. Andrieu, A. Doucet, and W. J. Fitzgerald, "Particle filtering for demodulation in fading channels with non-Gaussian additive noise," *IEEE Trans. Communications*, vol. 49, no. 4, pp. 579-582, 2001.
- [9] T. Ghirmai, M. F. Bugallo, J. Míguez, and P. M. Djurić, "Joint symbol detection and timing estimation using particle filtering," in *Proc. IEEE Int. Conf. Acoustics, Speech, Signal Processing (ICASSP '03)*, vol. 4, pp. 596-599, Hong Kong, April 2003.
- [10] P. M. Djurić, J. H. Kotecha, J. Zhang, et al., "Particle filtering," *IEEE Signal Processing Magazine*, vol. 2, no. 5, pp. 19-38, September 2003.
- [11] Third Generation Partnership Project, "Spreading and modulation (FDD) (Release 1999)," 3G Tech. Spec. (TS) 25.213, v. 3.3.0, Technical Specification Group Radio Access Network, December 2000.
- [12] E. Punskeya, A. Doucet, and W. J. Fitzgerald, "On the use and misuse of particle filtering in digital communications," in *Proc. 11th European Signal Processing Conference (EUSIPCO '02)*, Toulouse, France, September 2002.
- [13] R. A. Iltis, "A sequential Monte Carlo filter for joint linear/nonlinear state estimation with application to DS-SS-CDMA," *IEEE Trans. Signal Processing*, vol. 51, no. 2, pp. 417-426, 2003.
- [14] T. Bertozzi, D. Le Ruyet, G. Rigal, and H. Vu-Thien, "On particle filtering for digital communications," in *Proc. IEEE Workshop on Signal Processing Advances in Wireless Communications (SPAWC '03)*, pp. 570-574, Rome, Italy, June 2003.
- [15] R. A. Iltis, "Joint estimation of PN code delay and multipath using the extended Kalman filter," *IEEE Trans. Communications*, vol. 38, no. 10, pp. 1677-1685, 1990.
- [16] A. Doucet, S. Godsill, and C. Andrieu, "On sequential Monte Carlo sampling methods for Bayesian filtering," *Statistics and Computing*, vol. 10, no. 3, pp. 197-208, 2000.
- [17] A. J. Viterbi, *CDMA: Principles of Spread Spectrum Communication*, Addison-Wesley Wireless Communications, Prentice Hall, Englewood Cliffs, NJ, USA, 1995.

- [18] H. Meyr, M. Moeneclaey, and S. Fechtel, *Digital Communication Receivers: Synchronization, Channel Estimation, and Signal Processing*, Wiley Series in Telecommunications and Signal Processing, John Wiley & Sons, New York, NY, USA, 2nd edition, 1998.
- [19] E. Sourour, G. Bottomley, and R. Ramesh, "Delay tracking for direct sequence spread spectrum systems in multipath fading channels," in *IEEE 49th Vehicular Technology Conference (VTC '99)*, vol. 1, pp. 422–426, Houston, Tex, USA, July 1999.
- [20] C. Komminakis, C. Fragouli, A. H. Sayed, and R. D. Wesel, "Multi-input multi-output fading channel tracking and equalization using Kalman estimation," *IEEE Trans. Signal Processing*, vol. 50, no. 5, pp. 1065–1076, 2002.
- [21] P. Tichavsky, C. H. Muravchik, and A. Nehorai, "Posterior Cramer-Rao bounds for discrete-time nonlinear filtering," *IEEE Trans. Signal Processing*, vol. 46, no. 5, pp. 1386–1396, 1998.
- [22] U. Mengali and A. N. D'Andrea, *Synchronization Techniques for Digital Receivers*, Plenum Press, New York, NY, USA, 1997.

Tanya Bertozzi received the Dr. Eng. degree in electrical engineering with a specialization in telecommunications from University of Parma, Italy, in 1998. From October 1998 to February 1999, she was a Research Engineer in the Information Department, University of Parma, Italy. In March 1999, she joined Diginext, Aix-en-Provence, France, as a Research Engineer. She is currently a Project Manager. In 2003, she received the Ph.D. degree in digital communications from the Conservatoire National des Arts et Métiers (CNAM), Paris, France. Her research interests are in the areas of digital communications and signal processing including particle filtering applications, channel estimation, multiuser detection, and space-time coding.



Didier Le Ruyet received the M.Eng. degree, the M.S. degree, and the Ph.D. degree from the Conservatoire National des Arts et Métiers (CNAM), Paris, France, in 1994 and 2001, respectively. From 1988 to 1995 he was a Research Engineer in the image processing and telecommunication Departments of Sagem, Cergy, France. He joined the Department of Electrical and Computer Engineering at CNAM, Paris, in 1996, where he became an Assistant Professor in 2002. His main research interests lie in the areas of digital communications and signal processing including channel estimation, channel coding, iterative decoding, and space-time coding.



Cristiano Panazio was born in Brasília, Brazil, in 1977. He received his B.S. and his M.S. degrees in electrical engineering from University of Campinas (UNICAMP), in 1999 and 2001, respectively. Since 2002, he is pursuing a Ph.D. degree at the Conservatoire National des Arts et Métiers (CNAM), Paris, France. His interests include adaptive filtering, synchronization, and nonlinear signal processing.



Han Vu Thien received the M. Eng. degree in telecommunications from the École Nationale Supérieure des Télécommunications (ENST), Paris, France, in 1967 and the Ph.D. in physical science from University of Paris XI, Orsay, France, in 1972. He has spent his entire career with the Conservatoire National des Arts et Métiers (CNAM), where he became a Professor of electrical engineering in 1982. Since 1984, he is also the Director of the Laboratoire des Signaux et systèmes. His main research interests lie in image and signal processing for medical applications, Digital television, and HF communication.



Joint Tracking of Manoeuvring Targets and Classification of Their Manoeuvrability

Simon Maskell

*QinetiQ Ltd, St. Andrews Road, Malvern, Worcestershire WR14 3PS, UK
Email: smaskell@signal.qinetiq.com*

Department of Engineering, University of Cambridge, Cambridge CB2 1PZ, UK

Received 30 May 2003; Revised 23 January 2004

Semi-Markov models are a generalisation of Markov models that explicitly model the state-dependent sojourn time distribution, the time for which the system remains in a given state. Markov models result in an exponentially distributed sojourn time, while semi-Markov models make it possible to define the distribution explicitly. Such models can be used to describe the behaviour of manoeuvring targets, and particle filtering can then facilitate tracking. An architecture is proposed that enables particle filters to be both robust and efficient when conducting joint tracking and classification. It is demonstrated that this approach can be used to classify targets on the basis of their manoeuvrability.

Keywords and phrases: tracking, classification, manoeuvring targets, particle filtering.

1. INTRODUCTION

When tracking a manoeuvring target, one needs models that can cater for each of the different regimes that can govern the target's evolution. The transitions between these regimes are often (either explicitly or implicitly) taken to evolve according to a Markov model. At each time epoch there is a probability of being in one discrete state given that the system was in another discrete state. Such Markov switching models result in an exponentially distributed sojourn time, the time for which the system remains in a given discrete state. Semi-Markov models (also known as renewal processes [1]) are a generalisation of Markov models that explicitly model the (discrete-state-dependent) distribution over sojourn time. At each time epoch there is a probability of being in one discrete state given that the system was in another discrete state and how long it has been in that discrete state. Such models offer the potential to better describe the behaviour of manoeuvring targets.

However, it is believed that the full potential of semi-Markov models has not yet been realised. In [2], sojourns were restricted to end at discrete epochs and filtered mode probabilities were used to deduce the parameters of the time-varying Markov process, equivalent to the semi-Markov process. In [3], the sojourns were taken to be gamma-distributed with integer-shape parameters such that the gamma variate could be expressed as a sum of exponential variates; the semi-Markov model could then be expressed as a (potentially highly dimensional) Markov model. This paper

proposes an approach that does not rely on the sojourn time distribution being of a given form, and so is capable of capitalising on all available model fidelity regarding this distribution. The author asserts that the restrictions of the aforementioned approaches currently limit the use of semi-Markov models in tracking systems and that the improved modelling (and so estimation) accuracy that semi-Markov models make possible has not been realised up to now.

This paper further considers the problem of both tracking and classifying targets. As discussed in [4], joint tracking and classification is complicated by the fact that sequentially updating a distribution over class membership necessarily results in an accumulation of errors. This is because, when tracking, errors are forgotten. In this context, the capacity to not forget, memory, is a measure of how rapidly the distribution over states becomes increasingly diffuse, making it difficult to predict where the target will be given knowledge of where it was. Just as the system forgets where it was, so any algorithm that mimics the system forgets any errors that are introduced. So, if the algorithm forgets any errors, it must converge. In the case of classification, this diffusion does not take place; if one knew the class at one point, it would be known for all future times. As a result, when conducting joint tracking and classification, it becomes not just pragmatically attractive but essential that the tracking process introduces as few errors as possible. This means that the accumulation of errors that necessarily takes place has as little impact as possible on the classification process.

There have been some previous approaches to solving the problem of joint tracking and identification that have been based on both grid-based approximations [5] and particle filters [6, 7]. An important failing of these implementations is that target classes with temporarily low likelihoods can end up being permanently lost. As a consequence of this same feature of the algorithms, these implementations cannot recover from any miscalculations and are not robust. This robustness issue has been addressed by stratifying the classifier [4]; one uses separate filters to track the target for each class (i.e., one might use a particle filter for one class and a Kalman filter for another) and then combines the outputs to estimate the class membership probabilities and so classification of the target. This architecture does enable different state spaces and filters to be used for each class, but has the deficiency that this choice could introduce biases and so systematic errors. So, the approach taken here is to adopt a single state space common to all the classes and a single (particle) filter, but to then attempt to make the filter as efficient as possible while maintaining robustness. This ability to make the filter efficient by exploiting the structure of the problem in the structure of the solution is the motivation for the use of a particle filter specifically.

This paper demonstrates this methodology by considering the challenging problem of classifying targets which differ only in terms of their similar sojourn time distributions; the set of dynamic models used to model the different regimes are taken to be the same for all the classes. Were one using a Markov model, all the classes would have the same mean sojourn time and so the same best-fitting Markov model. Hence, it is only possible to classify the targets because semi-Markov models are being used.

Since the semi-Markov models are nonlinear and non-Gaussian, the particle-filtering methodology [8] is adopted for solving this joint tracking and classification problem. The particle-filter represents uncertainty using a set of samples. Here, each of the samples represent different hypotheses for the sojourns times and state transitions. Since there is uncertainty over both how many transitions occurred and when they occurred, the particles represent the diversity over the number of transitions and their timing. Hence, the particles differ in dimensionality. This is different from the usual case for which the dimensionality of all the particles is the same. Indeed, this application of the particle filter is a special case of the generic framework developed concurrently by other researchers [9]. The approach described here exploits the specifics of the semi-Markov model, but the reader interested in the more generic aspects of the problem is referred to [9].

Since, if the sojourn times are known, the system is linear and Gaussian, the Kalman filter is used to deduce the parameters of the uncertainty over target state given the hypothesised history of sojourns. So, the particle filter is only used for the difficult part of the problem—that of deducing the timings of the sojourn ends—and the filter operates much like a multiple hypothesis tracker, with hypotheses in the (continuous) space of transition times. To make this more explicit, it should be emphasised that the complexity of the particle

filter is not being increased by using semi-Markov models, but rather particle filters are being applied to the problem associated with semi-Markov models. The resulting computational cost is roughly equivalent to one Kalman filter per particle and in the example considered in Section 6 just 25 particles were used for each of the three classes.¹ The author believes that this computational cost is not excessive and that, in applications for which it is beneficial to capitalise on the use of semi-Markov models—which the author believes to be numerous—the approach is practically useful. However, this issue of the trade-off between the computational cost and the resulting performance for specific applications is not the focus of this paper; here the focus is on proposing the generic methodology. For this reason, a simple yet challenging, rather than necessarily practically useful, example is used to demonstrate that the methodology has merit.

A crucial element of the particle filter is the proposal distribution, the method by which each new sample is proposed from the old samples. Expedient choice of proposal distribution can make it possible to drastically reduce the number of particles necessary to achieve a certain level of performance. Often, the trade-off between complexity and performance is such that this reduction in the number of particles outweighs any additional computation necessary to use the more expedient proposal distributions. So, the choice of proposal distribution can be motivated as a method for reducing computational expense. Here, however, if as few errors as possible, are to be introduced as is critically important when conducting joint tracking and classification, it is crucial that the proposal distribution is well matched to the true system. Hence, the set of samples is divided into a number of strata, each of which had a proposal that was well matched to one of the classes. Whatever the proposal distribution, it is possible to calculate the probability of every class. So, to minimise the errors introduced, for each particle (and so hypothesis for the history of state transitions and sojourn times), the probability of all the classes is calculated. So each particle uses a proposal matched to one class, but calculates the probability of the target being a member of every class. Note that this calculation is not computationally expensive, but provides information that can be used to significantly improve the efficiency of the filter.

So, the particles are used to estimate the manoeuvres and a Kalman filter is used to track the target. The particles are split into strata each of which is well suited to tracking one of the classes and the strata of particles used to classify the target on the basis of the target's manoeuvrability. The motivation for this architecture is the need to simultaneously achieve robustness and efficiency.

This paper is structured as follows: Section 2 begins by introducing the notation and the semi-Markov model

¹This number is small and one might use more in practical situations, but the point is that the number of particles is not large and so the computational expense is roughly comparable to other existing algorithms.

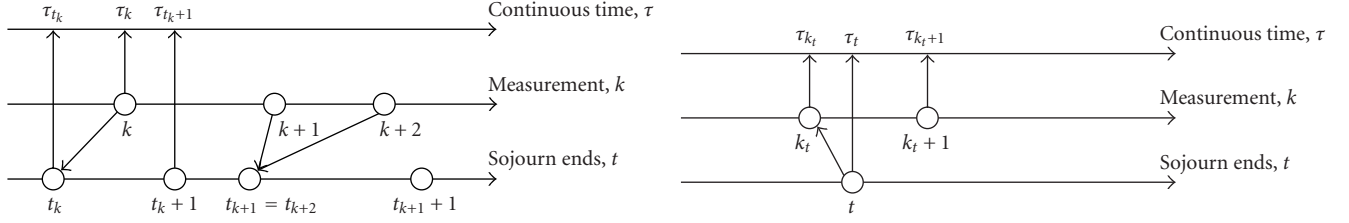


FIGURE 1: Diagram showing the relationship between continuous time, the time when measurements were received, and the time of sojourn ends. The circles represent the receipt of measurements or the start of a sojourn.

structure that is used. Section 3 describes how a particle filter can be applied to the hard parts of the problem, the estimation of the semi-Markov process' states. Some theoretical concerns relating to robust joint tracking and identification are discussed in Section 4. Then, in Section 5, efficient and robust particle-filter architectures are proposed as solutions for the joint tracking and classification problem. Finally, an exemplar problem is considered in Section 6 and some conclusions are drawn in Section 7.

2. MODEL

When using semi-Markov models, there is a need to distinguish between continuous time, the indexing of the measurements, and the indexing of the sojourns. Here, continuous time is taken to be τ , measurements are indexed by k , and manoeuvre regimes (or *sojourns*) are indexed by t . The continuous time when the k th measurement was received is τ_k . The time of the onset of the sojourn is τ_t ; t_k is then the index of the sojourn during which the k th measurement was received. Similarly, k_t is the most recent measurement prior to the onset of the t th sojourn. This is summarised in Table 1 while Figure 1 illustrates the relationship between such quantities as $(t_k + 1)$ and t_{k+1} .

The model corresponding to sojourn t is s_t . s_t is a discrete semi-Markov process with transition probabilities $p(s_t | s_{t-1})$ that are known; note that since, at the sojourn end, a transition must occur, so $p(s_t | s_{t-1}) = 0$ if $s_t = s_{t-1}$;

$$p(s_t | s_{t-1}) = p(s_t | s_{1:t-1}), \quad (1)$$

where $s_{1:t-1}$ is the history of states for the first to the $(t-1)$ th regime and similarly, $y_{1:k}$ will be used to denote the history of measurements up to the k th measurement.

For simplicity, the transition probabilities are here considered invariant with respect to time once it has been determined that a sojourn is to end; that is, $p(s_t | s_{t-1})$ is not a function of τ . The sojourn time distribution that determines the length of time for which the process remains in state s_t is distributed as $g(\tau - \tau_t | s_t)$:

$$p(\tau_{t+1} | \tau_t, s_t) \triangleq g(\tau - \tau_t | s_t). \quad (2)$$

The s_t process governs a continuous time process, x_τ , which given s_t and a state at a time after the start of the sojourn $x_{\tau_{t+1}} > x_{\tau'} > x_{\tau_t}$ has a distribution $f(x_\tau | x_{\tau'}, s_t)$. So, the

TABLE 1: Definition of notation.

Notation	Definition
τ_k	Continuous time relating to k th measurement
τ_t	Continuous time relating to t th sojourn time
t_k	Sojourn prior to k th measurement; so that $\tau_{t_k} \leq \tau_k \leq \tau_{t_{k+1}}$
k_t	Measurement prior to t th sojourn; so that $\tau_{k_t} \leq \tau_t \leq \tau_{k_t+1}$
s_t	Manoeuvre regime for $\tau_t < \tau < \tau_{t+1}$

distribution of x_τ given the initial state at the start of the sojourn and the fact that the sojourn continues to time τ is

$$p(x_\tau | x_{\tau_t}, s_t, \tau_{t+1} > \tau) \triangleq f(x_\tau | x_{\tau_t}, s_t). \quad (3)$$

If x_k is the history of states (in continuous time), then a probabilistic model exists for how each measurement, y_k , is related to the state at the corresponding continuous time:

$$p(y_k | x_k) = p(y_k | x_{\tau_t; \tau_k}) = p(y_k | x_{\tau_k}). \quad (4)$$

This formulation makes it straightforward to then form a dynamic model for $s_{1:t_k}$ process and $\tau_{1:t_k}$ as follows:

$$p(s_{1:t_k}, \tau_{1:t_k}) = \left(\prod_{t'=2}^{t_k} p(s_{t'} | s_{t'-1}) p(\tau_{t'} | \tau_{t'-1}, s_{t'-1}) \right) p(s_1) p(\tau_1), \quad (5)$$

where $p(s_1)$ is the initial prior on the state of the sojourn time (which we later assume to be uniform) and $p(\tau_1)$ is the prior on the time of the first sojourn end (which we later assume to be a delta function). This can then be made conditional on $s_{1:t_{k-1}}$ and $\tau_{1:t_{k-1}}$, which makes it possible to sample the semi-Markov process' evolution between measurements:

$$\begin{aligned} & p(\{s_{1:t_k}, \tau_{1:t_k}\} \setminus \{s_{1:t_{k-1}}, \tau_{1:t_{k-1}}\} | s_{1:t_{k-1}}, \tau_{1:t_{k-1}}) \\ & \propto \frac{p(s_{1:t_k}, \tau_{1:t_k})}{p(s_{1:t_{k-1}}, \tau_{1:t_{k-1}})} \\ & = \frac{\left(\prod_{t'=2}^{t_k} p(s_{t'} | s_{t'-1}) p(\tau_{t'} | \tau_{t'-1}, s_{t'-1}) \right) p(s_1) p(\tau_1)}{\left(\prod_{t'=2}^{t_{k-1}} p(s_{t'} | s_{t'-1}) p(\tau_{t'} | \tau_{t'-1}, s_{t'-1}) \right) p(s_1) p(\tau_1)} \\ & = \prod_{t'=t_{k-1}+1}^{t_k} p(s_{t'} | s_{t'-1}) p(\tau_{t'} | \tau_{t'-1}, s_{t'-1}), \end{aligned} \quad (6)$$

where $A \setminus B$ is the set A without the elements of the set B . Note that in this case $\{s_{1:t_k}, \tau_{1:t_k}\} \setminus \{s_{1:t_{k-1}}, \tau_{1:t_{k-1}}\}$ could be the empty set in which case, $p(\{s_{1:t_k}, \tau_{1:t_k}\} \setminus \{s_{1:t_{k-1}}, \tau_{1:t_{k-1}}\} | s_{1:t_{k-1}}, \tau_{1:t_{k-1}}) = 1$.

So, it is possible to write the joint distribution of the s_t and x_τ processes and the times of the sojourns, $\tau_{1:t_k}$, up to the time of the k th measurement, τ_k , as

$$\begin{aligned}
& p(s_{1:t_k}, x_k, \tau_{1:t_k} | y_{1:k}) \\
& \propto p(s_{1:t_k}, \tau_{1:t_k}) p(x_k, y_{1:k} | s_{1:t_k}, \tau_{1:t_k}) \\
& = p(s_{1:t_k}, \tau_{1:t_k}) p(x_k | s_{1:t_k}, \tau_{1:t_k}) p(y_{1:k} | x_k) \\
& = p(s_{1:t_k}, \tau_{1:t_k}) p(x_{\tau_k} | x_{\tau_{t_k}}, s_{t_k}) \left(\prod_{t'=2}^{t_k} p(x_{\tau_{t'}} | x_{\tau_{t'-1}}, s_{t'-1}) \right) \\
& \quad \times p(x_{\tau_1}) \prod_{k'=1}^k p(y_{k'} | x_{\tau_{k'}}) \\
& \propto \underbrace{p(s_{1:t_{k-1}}, x_{k-1}, \tau_{1:t_{k-1}} | y_{1:k-1})}_{\text{The posterior at } k-1} \\
& \quad \times \underbrace{p(\{s_{1:t_k}, \tau_{1:t_k}\} \setminus \{s_{1:t_{k-1}}, \tau_{1:t_{k-1}}\} | s_{1:t_{k-1}}, \tau_{1:t_{k-1}})}_{\text{Evolution of semi-Markov model}} \\
& \quad \times \underbrace{p(y_k | x_{\tau_k})}_{\text{Likelihood}} \underbrace{\frac{p(x_{\tau_k} | x_{\tau_{t_k}}, s_{t_k})}{p(x_{\tau_{t_k-1}} | x_{\tau_{t_k-1}}, s_{t_{k-1}})}}_{\text{Effect on } x_\tau \text{ of incomplete regimes}} \\
& \quad \times \underbrace{\left(\prod_{t'=t_{k-1}+1}^{t_k} p(x_{\tau_{t'}} | x_{\tau_{t'-1}}, s_{t'-1}) \right)}_{\text{Effect on } x_\tau \text{ of sojourns between } k-1 \text{ and } k}.
\end{aligned} \tag{7}$$

This is a recursive formulation of the problem. The annotations indicate the individual terms' relevance.

3. APPLICATION OF PARTICLE FILTERING

Here, an outline of the form of particle filtering used is given so as to provide some context for the subsequent discussion and introduce notation. The reader who is unfamiliar with the subject is referred to the various tutorials (e.g., [8]) and books (e.g., [10]) available on the subject.

A particle filter is used to deduce the sequence of sojourn times, $\tau_{1:t_k}$, and the sequence of transitions, $s_{1:t_k}$, as a set of measurements are received. This is achieved by sampling N times from a proposal distribution of a form that extends the existing set of sojourn times and the s_t process with samples of the sojourns that took place between the previous and the current measurements:

$$\begin{aligned}
& \{ \{s_{1:t_k}, \tau_{1:t_k}\} \setminus \{s_{1:t_{k-1}}, \tau_{1:t_{k-1}}\} \}^i \\
& \sim q(\{s_{1:t_k}, \tau_{1:t_k}\} \setminus \{s_{1:t_{k-1}}, \tau_{1:t_{k-1}}\} | \{s_{1:t_{k-1}}, \tau_{1:t_{k-1}}\}^i, y_k), \\
& \quad i = 1, \dots, N.
\end{aligned} \tag{8}$$

A weight is then assigned according to the principle of importance sampling:

$$\begin{aligned}
\bar{w}_k^i &= w_{k-1}^i \frac{p(\{s_{1:t_k}, \tau_{1:t_k}\}^i \setminus \{s_{1:t_{k-1}}, \tau_{1:t_{k-1}}\}^i | \{s_{1:t_{k-1}}, \tau_{1:t_{k-1}}\}^i)}{q(\{s_{1:t_k}, \tau_{1:t_k}\}^i \setminus \{s_{1:t_{k-1}}, \tau_{1:t_{k-1}}\}^i | \{s_{1:t_{k-1}}, \tau_{1:t_{k-1}}\}^i, y_k)} \\
& \quad \times p(y_k | \{s_{1:t_k}, \tau_{1:t_k}\}^i).
\end{aligned} \tag{9}$$

These *unnormalised* weights are then normalised:

$$w_k^i = \frac{\bar{w}_k^i}{\sum_{i'=1}^N \bar{w}_k^{i'}} \tag{10}$$

and estimates of expectations calculated using the (normalised) weighted set of samples. When the weights become skewed, some of the samples dominate these expectations, so the particles are resampled; particles with low weights are probabilistically discarded and particles with high weights are probabilistically replicated in such a way that the expected number of offspring resulting from a given particle is proportional to the particle's weight. This resampling can introduce unnecessary errors. So, it should be used as infrequently as possible. To this end, a threshold can be put on the approximate effective sample size, so that when this effective sample size falls below a predefined threshold, the resampling step is performed. This approximate effective sample can be calculated as follows:

$$N_{\text{eff}} \approx \frac{1}{\sum_{i=1}^N (\bar{w}_k^i)^2}. \tag{11}$$

It is also possible to calculate the incremental likelihood:

$$p(y_k | y_{1:k-1}) \approx \sum_{i=1}^N \bar{w}_k^i, \tag{12}$$

which can be used to calculate the likelihood of the entire data sequence, which will be useful in later sections:

$$p(y_{1:k}) = p(y_1) \prod_{k'=2}^k p(y_{k'} | y_{1:k'-1}), \tag{13}$$

where $p(y_1) \triangleq p(y_1 | y_{1:0})$, so can be calculated using (12).

4. THEORETICAL CONCERNS RELATING TO JOINT TRACKING AND CLASSIFICATION

The proofs of convergence for particle filters rely on the ability of the dynamic models used to forget, the errors introduced by the Monte Carlo integration [11, 12]. If errors are forgotten, then the errors cannot accumulate and so the algorithm must converge on the true uncertainty relating to the path through the state space.

Conversely, if the system does not forget, then errors will accumulate and this will eventually cause the filter to diverge. This applies to sequential algorithms in general, including Kalman filters,² which accumulate finite precision errors, though such errors are often sufficiently small that such problems rarely arise and have even less rarely been noticed.

For a system to forget, its model needs to involve the states changing with time; it must be *ergodic*. There is then a finite probability of the system being in any state given that it was in any other state at some point in the past; so, it is not possible for the system to get stuck in a state. Models for classification do not have this ergodic property since the class is constant for all time; such models have infinite memory. Approaches to classification (and other long memory problems) have been proposed in the past based on both implicit and explicit modifications of the model that reduce the memory of the system by introducing some dynamics. Here, the emphasis is on using the models in their true form.

However, if the model's state is discrete, as is the case with classification, there is a potential solution described in this context in [4]. The idea is to ensure that all probabilities are calculated based on the classes remaining constant and to run a filter for each class; these filters cannot be reduced in number when the probability passes a threshold if the system is to be robust. In such a case, the overall filter is conditionally ergodic. The approach is similar to that advocated for classification alone whereby different classifiers are used for different classes [13].

The preceding argument relates to the way that the filter forgets errors. This enables the filter to always be able to visit every part of the state space; and the approach advocated makes it possible to recover from a misclassification. However, this does not guarantee that the filter can calculate classification probabilities with any accuracy. The problem is the variation resulting from different realisations of the errors caused in the inference process. In a particle-filter context, this variation is the Monte Carlo variation and is the result of having sampled one of many possible different sets of particles at a given time. Put more simply; performing the sampling step twice would not give the same set of samples.

Equation (13) means that, if each iteration of the tracker introduces errors, the classification errors necessarily accumulate. There is nothing that can be done about this. All that can be done is to attempt to minimise the errors that are introduced such that the inevitable accumulation of errors will not impact performance on a time scale that is of interest.

So, to be able to classify targets based on their dynamic behaviour, all estimates of probabilities must be based on the classes remaining constant for all time and the errors introduced into the filter must be minimised. As a result, classification performance is a good test of algorithmic performance.

²It is well documented that extended Kalman filters can accumulate linearisation errors which can cause filter divergence, but here the discussion relates to Kalman filtering with linear Gaussian distributions such that the Kalman filter is an analytic solution to the problem of describing the pdf.

5. EFFICIENT AND ROBUST CLASSIFICATION

The previous section asserts that to be robust, it is essential to estimate probabilities based on all the classes always remaining constant. However, to be efficient, the filter should react to the classification estimates and focus its effort on the most probable classes (this could equally be the class with the highest expected cost according to some nonuniform cost function but this is not considered here).

To resolve these two seemingly contradictory requirements of robustness twinned with efficiency, the structure of the particle filter can be capitalised upon. The particle filter distinguishes between the proposal used to sample the particles' paths and the weights used to reflect the disparity between the proposal and the true posterior. So, it is possible for the proposal to react to the classification probabilities and favour proposals well suited to the more probable classes while calculating the weights for the different classes; this is equivalent to Rao-Blackwellising the discrete distribution over class for each particle.

One could enable the system to react to the classification probabilities while remaining robust to misclassification by each particle sampling the importance function from a set of importance samplers according to the classification probabilities. Each importance sampler would be well suited to the corresponding class and each particle would calculate the weights with respect to all the classes given its sampled values of the state.

However, here a different architecture is advocated; the particles are divided into strata, such that the different strata each use an importance function well suited to one of the classes. For any particle in the j th stratum, S_j , and in the context of the application of particle filtering to semi-Markov models, the importance function is then of the form $q(\{s_{1:t_k}, \tau_{1:t_k}\} \setminus \{s_{1:t_{k-1}}, \tau_{1:t_{k-1}}\} \mid \{s_{1:t_{k-1}}, \tau_{1:t_{k-1}}\}, y_k, S_j)$. The strata then each have an associated weight and these weights sum to unity across the strata. If each particle calculates the probability of all the classes given its set of hypotheses, then the architecture will be robust. It is then possible to make the architecture efficient by adding a decision logic that reacts to the weights on the strata; one might add and remove strata on the basis of the classification probabilities. The focus here is not on designing such a decision logic, but to propose an architecture that permits the use of such logic.

To use this architecture, it is necessary to manipulate strata of particles and so to be able to calculate the total weight on a class or equally on a stratum. To this end, the relations that enable this to happen are now outlined.

The classes are indexed by c , particles by i , and the strata by j . The model used to calculate the weights is M and the stratum is S . So, the unnormalised weight for the i th particle in stratum S_j , using model M_c , is $\tilde{w}_k^{(i,j,c)}$.

The weight on a stratum, $p(S_j | y_{1:k})$, can be deduced from

$$p(S_j | y_{1:k}) \propto p(y_{1:k} | S_j) p(S_j), \quad (14)$$

where $p(S_j)$ is the (probably uniform) prior across the strata.

This leads to the following recursion:

$$p(S_j | y_{1:k}) \propto p(y_k | y_{1:k-1}, S_j) p(S_j | y_{1:k-1}), \quad (15)$$

where $p(y_k | y_{1:k-1}, S_j)$ can be estimated using a minor modification of (12) as follows:

$$p(y_k | y_{1:k-1}, S_j) \approx \sum_{i,c} \bar{w}_k^{(i,j,c)}. \quad (16)$$

Similarly, for the classes,

$$p(M_c | y_{1:k}) \propto p(y_k | y_{1:k-1}, M_c) p(M_c | y_{1:k-1}), \quad (17)$$

where

$$\begin{aligned} p(M_c | y_{1:k}) &= \sum_j p(S_j, M_c | y_{1:k}) \\ &= \sum_j p(S_j | y_{1:k}) p(M_c | S_j, y_{1:k}), \quad (18) \\ p(M_c | S_j, y_{1:k}) &\propto \sum_i \bar{w}_k^{(i,j,c)}. \end{aligned}$$

To implement this recursion, the weights of the classes are normalised such that they sum to unity over the particle in the strata:

$$w_k^{(c|i,j)} \triangleq \frac{\bar{w}_k^{(i,j,c)}}{\bar{w}_k^{(i,j)}}, \quad (19)$$

where $\bar{w}_k^{(i,j)}$ is the total unnormalised weight of the particle:

$$\bar{w}_k^{(i,j)} \triangleq \sum_c \bar{w}_k^{(i,j,c)}. \quad (20)$$

These weights are then normalised such that they sum to unity within each strata:

$$w_k^{(i,j)} \triangleq \frac{\bar{w}_k^{(i,j)}}{\bar{w}_k^{(j)}}, \quad (21)$$

where $\bar{w}_k^{(j)}$ is the total unnormalised weight of the stratum:

$$\bar{w}_k^{(j)} \triangleq \sum_i \bar{w}_k^{(i,j)}. \quad (22)$$

These weights are also normalised such that they sum to unity across the strata:

$$w_k^{(j)} \triangleq \frac{\bar{w}_k^{(j)}}{\sum_j \bar{w}_k^{(j)}}. \quad (23)$$

The skewness of each stratum is then used to assess whether that stratum has degenerated and so if resampling is necessary for the set of particles in that stratum. This means that the weight relating to M_c for the i th particle within the j th stratum is

$$w_k^{(i,j,c)} \propto w_k^{(j)} w_k^{(i,j)} w_k^{(c|i,j)}. \quad (24)$$

```

For j = 1 : N_M
  Initialise: w_0^{(j)} = 1/N_M
  For i = 1 : N_P
    Initialise: w_0^{(i,j)} = 1/N_P
    Initialise: x_0^{(i,j)} ~ p(x_0)
    For c = 1 : N_M
      Initialise: w_0^{(c|i,j)} = 1/N_M
    End For
  End For
End For
For k = 1 : N_K
  Implement recursion
End For

```

ALGORITHM 1

So, with N_P particles and N_M classes (and so N_M strata), running the algorithm over N_K steps can be summarised as follows in Algorithm 1. $p(x_0)$ is the initial prior on the state and Implement Recursion is conducted as in Algorithm 2 where V_j is the reciprocal of the sum of the squared weights, on the basis of which one can decide whether or not it is necessary to Resample. N_T is then the threshold on the approximate effective sample size which determines when to resample; $N_T \approx (1/2)N_P$ might be typical. Note that the resampling operation will result in replicants of a subset of some of the particles within the j th stratum, but that for each copy of the i th particle in the j th stratum, $w_k^{(c|i,j)}$ is left unmodified.

6. EXAMPLE

6.1. Model

The classification of targets which differ solely in terms of the semi-Markov model governing the s_t process is considered. The classes have different gamma distributions for their sojourn times but all have the same mean value for the sojourn time, and so the same best-fitting Markov model. As stated in the introduction, this example is intended to provide a difficult to analyse, yet simple to understand, exemplar problem. The author does intend the reader to infer that the specific choice of models and parameters are well suited to any specific application.

The x_τ process is taken to be a constant velocity model; an integrated diffusion process

$$f(x_{\tau+\Delta} | x_\tau, s) = \mathcal{N}(x_{\tau+\Delta}; A(\Delta)x_\tau, Q_s(\Delta)), \quad (25)$$

where $\mathcal{N}(x; m, C)$ denotes a Gaussian distribution for x , with mean, m , and covariance, C , and where

$$\begin{aligned} A(\Delta) &= \begin{bmatrix} 1 & \Delta \\ 0 & 1 \end{bmatrix}, \\ Q_s(\Delta) &= \begin{bmatrix} \frac{\Delta^3}{3} & \frac{\Delta^2}{2} \\ \frac{\Delta^2}{2} & \Delta \end{bmatrix} \sigma_s^2, \end{aligned} \quad (26)$$


```

Initialise  $\bar{w}_k = 0$ 
For  $j = 1 : N_M$ 
    Initialise  $V_j = 0$ 
    Initialise output classification probabilities:  $\bar{P}_k^c = 0$ 
    Initialise  $\bar{w}_k^{(j)} = 0$ 
    For  $i = 1 : N_P$ 
        Initialise  $\bar{w}_k^{(i,j)} = 0$ 
        Sample  $x_k^{(i,j)} \sim q(x_k | x_{k-1}^{(i,j)}, y_k, S_j)$ 
        For  $c = 1 : N_M$ 
             $w_k^{(i,j,c)} = w_{k-1}^{(j)} w_{k-1}^{(i,j)} w_{k-1}^{(c|i,j)}$ 
             $\bar{w}_k^{(i,j,c)} = w_k^{(i,j,c)} (p(y_k | x_k^{(i,j)}, M_c) \times p(x_k^{(i,j,c)} | x_{k-1}^{(i,j)}, M_c) / q(x_k^{(i,j)} | x_{k-1}^{(i,j)}, y_k, S_j))$ 
             $\bar{w}_k^{(i,j)} = \bar{w}_k^{(i,j)} + \bar{w}_k^{(i,j,c)}$ 
             $\bar{w}_k^{(j)} = \bar{w}_k^{(j)} + \bar{w}_k^{(i,j)}$ 
             $\bar{w}_k = \bar{w}_k + \bar{w}_k^{(i,j,c)}$ 
             $\bar{P}_k^c = \bar{P}_k^c + \bar{w}_k^{(i,j,c)}$ 
        End For
    End For
End For
For  $c = 1 : N_M$ 
     $P_k^c = \bar{P}_k^c / \bar{w}_k$ , which can be output as necessary
End For
For  $j = 1 : N_M$ 
     $w_k^{(j)} = \bar{w}_k^{(j)} / \bar{w}_k$ 
    For  $i = 1 : N_P$ 
         $w_k^{(i,j)} = \bar{w}_k^{(i,j)} / \bar{w}_k^{(j)}$ 
        For  $c = 1 : N_M$ 
             $w_k^{(c|i,j)} = \bar{w}_k^{(c|i,j)} / \bar{w}_k^{(i,j)}$ 
        End For
         $V_j = V_j + (w_k^{(i,j)})^2$ 
    End For
    Resample  $j$ th stratum if  $1/V_j < N_T$ 
End For
    
```

ALGORITHM 2

where the discrete state, s_t , takes one of two values which differ in terms of σ_s^2 ; $\sigma_1^2 = 0.001$ and $\sigma_2^2 = 100$.

The data are linear Gaussian measurements of position

$$p(y_k | x_{\tau_k}) = \mathcal{N}(y_k; Hx_{\tau_k}, R), \quad (27)$$

where

$$H = \begin{bmatrix} 1 & 0 \end{bmatrix}, \quad (28)$$

and $R = 0.1$. The measurements are received at regular intervals such that $\tau_k - \tau_{k-1} = 0.5$ for all $k > 1$.

The three classes' sojourn distributions are

$$g(\tau - \tau_t | s_t, M_c) = \begin{cases} \mathcal{G}(\tau - \tau_t; 2, 5), & s_t = 1, c = 1, \\ \mathcal{G}(\tau - \tau_t; 10, 1), & s_t = 1, c = 2, \\ \mathcal{G}(\tau - \tau_t; 50, 0.2), & s_t = 1, c = 3, \\ \mathcal{G}(\tau - \tau_t; 10, 0.1), & s_t = 2, \forall c, \end{cases} \quad (29)$$

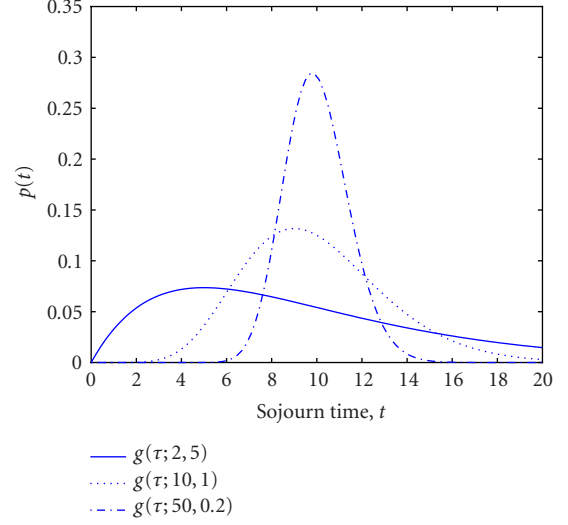


FIGURE 2: Sojourn time distributions for $s_t = 1$ for the different classes.

where $\mathcal{G}(x; \alpha, \beta)$ is a gamma distribution over x , with shape parameter α and scale parameter β . Figure 2 shows these different sojourn time distributions. Note that since the mean of the gamma distribution is $\alpha\beta$, all the sojourn distributions for $s_t = 1$ have the same mean. Hence, the exponential distribution (which only has a single parameter that defines the mean) for all three classes would be the same.

Since there are only two discrete states, the state transition probabilities are simple:

$$p(s_t | s_{t-1}) = \begin{cases} 0, & s_t = s_{t-1}, \\ 1, & s_t \neq s_{t-1}. \end{cases} \quad (30)$$

This means that, given the initial discrete state, the sojourn ends define the discrete-state sequence.

$p(s_1)$ is taken to be uniform across the two models and $p(\tau_1) = \delta(\tau_1 - 0)$, so it assumed known that there was a transition at time 0. x_0 is initialised at zero as follows:

$$x_0 = \begin{bmatrix} 0 \\ 0 \end{bmatrix}. \quad (31)$$

6.2. Tracking of manoeuvring targets

A target from the first class is considered. A Rao-Blackwellised particle filter is used. The particle filter samples the sojourn ends and then, conditional on the sampled sojourn ends and state transitions, uses a Kalman filter to exactly describe the uncertainty relating to x_{τ} and a discrete distribution over class to exactly describe the classification probabilities (as described previously).

For the proposal in the particle filter, (6), the dynamic prior for the s_t process is used, with a minor modification:

$$\begin{aligned} q(\{s_{1:t_k}, \tau_{1:t_k}\} \setminus \{s_{1:t_{k-1}}, \tau_{1:t_{k-1}}\} | \{s_{1:t_{k-1}}, \tau_{1:t_{k-1}}\}, y_k) \\ \triangleq p(\{s_{1:t_k}, \tau_{1:t_k}\} \setminus \{s_{1:t_{k-1}}, \tau_{1:t_{k-1}}\} | s_{1:t_{k-1}}, \tau_{1:t_{k-1}}, \tau_{t_{k-1}} > \tau_k, M_j) \\ = \int p(\{s_{1:t_k}, \tau_{1:t_{k+1}}\} \setminus \{s_{1:t_{k-1}}, \tau_{1:t_{k-1}}\} \\ | s_{1:t_{k-1}}, \tau_{1:t_{k-1}}, \tau_{t_{k+1}} > \tau_k, M_j) d\tau_{t_{k+1}}, \end{aligned} \quad (32)$$

that is, when sampling up to time τ_k , the s_t process is extended to beyond τ_k , but the sample of the final sojourn time is integrated out (so forgotten); the proposal simply samples that the next sojourn is after the time of the measurement, not what time it actually took place. This exploits some structure in the problem since $\tau_{t_{k+1}}$ has no impact on the estimation up to time τ_k and so classification on the basis of $y_{1:k}$. The weight update equation simplifies since the dynamics are used as the proposal:

$$\bar{w}_k^i = w_{k-1}^i p(y_k | \{s_{1:t_k}, \tau_{1:t_k}\}^i), \quad (33)$$

where $p(y_k | \{s_{1:t_k}, \tau_{1:t_k}\}^i)$ can straightforwardly be calculated by a Kalman filter with a time-varying process model (with model transitions at the sojourn ends) and measurement updates at the times of the measurements.

Having processed the k measurement, the i th particle then needs to store the time of the hypothesised last sojourn, $\tau_{t_k}^{(i)}$, the current state, $s_{t_k}^{(i)}$, a mean and covariance for x_{τ_k} , and a weight, $w_k^{(i)}$.

Just $N_P = 25$ particles are used and initialised with samples from $p(s_1)$ and $p(\tau_1)$ (so all the same τ_1). Each particles' initial value for the Kalman filter's mean is the true initial state, m . The initial value for the covariance is then defined as C :

$$C = \begin{bmatrix} 100 & 0 \\ 0 & 10 \end{bmatrix}. \quad (34)$$

The weights are all initialised as equal for all the particles. Resampling takes place if the approximate effective sample size given in (11) falls below $N_T = 12.5$. Since each particle needs to calculate the parameters of a Kalman filter, the computational cost is roughly equivalent to that of a multiple hypothesis tracker [14] with 25 hypotheses; here the hypotheses (particles) are in the continuous space of the times of the sojourn ends rather than the discrete space of the associations of measurements with the track. The computational cost is therefore relatively low and the algorithm is therefore amenable to practical real-time implementation.

With N_P particles and N_K iterations, the algorithm is implemented as in Algorithm 3.

The true trajectory through the discrete space is given in Figure 3. The hypothesis for the trajectory through the discrete space for some of the particles is shown in Figure 4. Note that, as a result of the resampling, all the particles have the same hypothesis for the majority of the trajectory through the discrete space, which is well matched (for the

```

For  $i = 1 : N_P$ 
  Initialise  $w_0^i = 1/N_P$ 
  Initialise  $\tau_1^i = 0$ 
  Initialise  $s_1^i$  as 1 if  $i > N_P/N_M$  or 2 with otherwise
  Initialise Kalman filter mean  $m_0^i = m$ 
  Initialise Kalman filter covariance  $C_0^i = C$ 
End For
For  $k = 1 : N_K$ 
  Initialise  $V = 0$ 
  Initialise  $\bar{w}_k = 0$ 
  For  $i = 1 : N_P$ 
    Sample  $\{s_{1:t_k}, \tau_{1:t_k}\}^i \setminus \{s_{1:t_{k-1}}, \tau_{1:t_{k-1}}\}^i$ 
     $\sim p(\{s_{1:t_k}, \tau_{1:t_k}\} \setminus \{s_{1:t_{k-1}}, \tau_{1:t_{k-1}}\} | \{s_{1:t_{k-1}}, \tau_{1:t_{k-1}}\}^i)$ 
    Calculate  $m_k^i$  and  $C_k^i$  from  $m_{k-1}^i$  and  $C_{k-1}^i$ 
    using  $s_{1:t_k}^i \setminus s_{1:t_{k-1}}^i$ 
    Calculate  $p(y_k | \{s_{1:t_k}, \tau_{1:t_k}\}^i)$  from  $y_k, m_k^i,$ 
    and  $C_k^i$ 
     $\bar{w}_k^i = w_{k-1}^i p(y_k | \{s_{1:t_k}, \tau_{1:t_k}\}^i)$ 
     $\bar{w}_k = \bar{w}_k + \bar{w}_k^i$ 
  End For
  For  $i = 1 : N_P$ 
     $w_k^i = \bar{w}_k^i / \bar{w}_k$ 
     $V = V + (w_k^i)^2$ 
  Resample if  $1/V < N_T$ 
End For

```

ALGORITHM 3

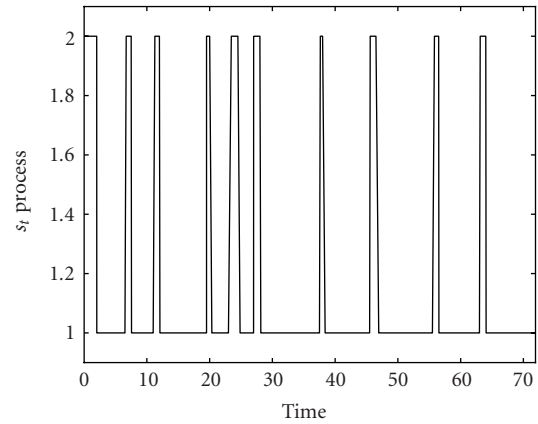


FIGURE 3: True trajectory for target through s_t state space.

most part) to the true trajectory. The diversity of the particles represents the uncertainty over the later part of the state sequence with the particles representing different hypothesised times and numbers of recent regime switches.

6.3. Classification on the basis of manoeuvrability

The proposals that are well suited to each class each use the associated class' prior as their proposal:

$$\begin{aligned} q(\{s_{1:t_k}, \tau_{1:t_k}\} \setminus \{s_{1:t_{k-1}}, \tau_{1:t_{k-1}}\} | \{s_{1:t_{k-1}}, \tau_{1:t_{k-1}}\}, y_k, S_j) \\ \triangleq p(\{s_{1:t_k}, \tau_{1:t_k}\} \setminus \{s_{1:t_{k-1}}, \tau_{1:t_{k-1}}\} | \{s_{1:t_{k-1}}, \tau_{1:t_{k-1}}\}, M_j). \end{aligned} \quad (35)$$

The weight update equation is then

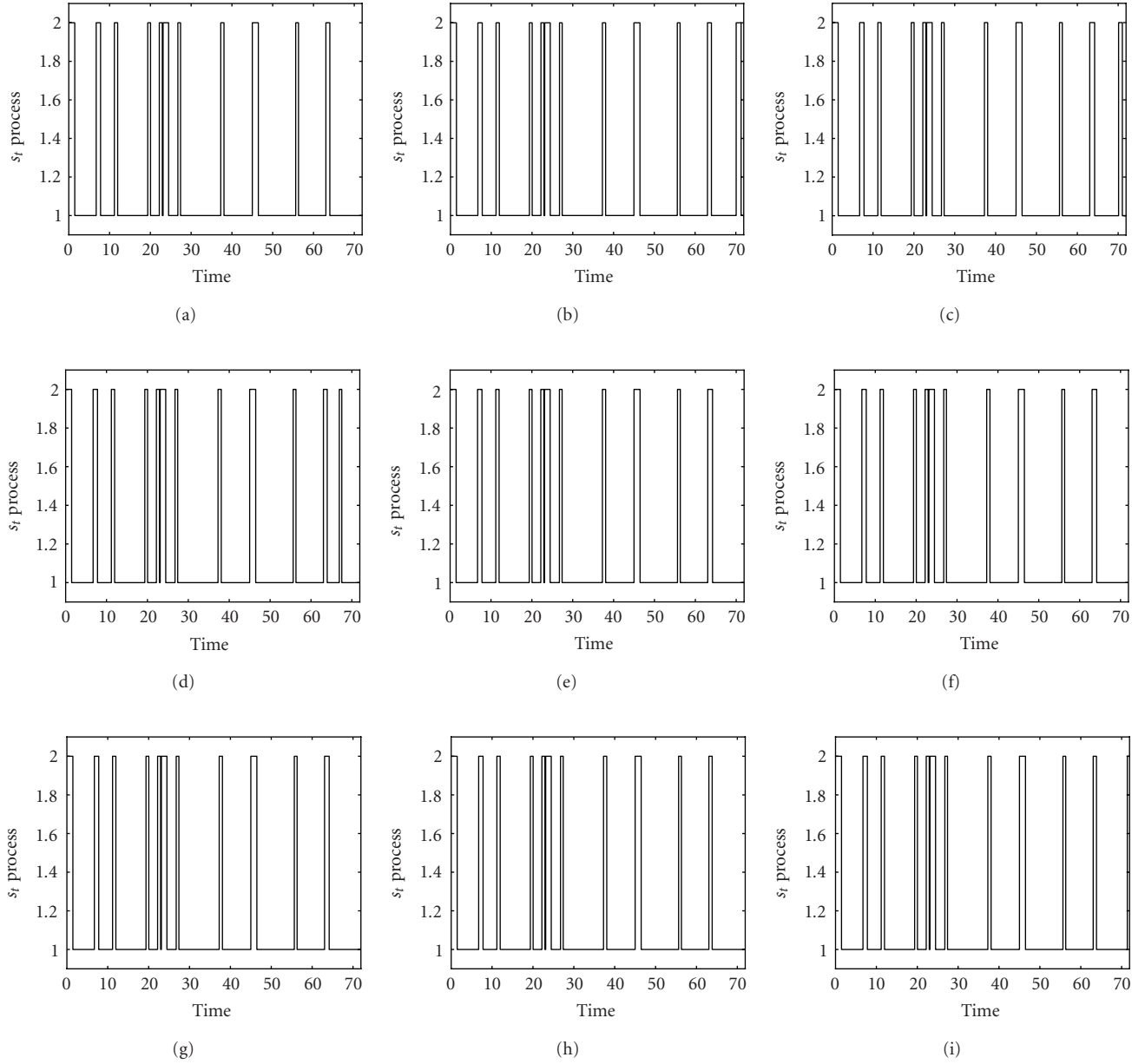


FIGURE 4: A subset of the particles' hypothesised trajectories through s_t space. (a) Particle 1. (b) Particle 2. (c) Particle 3. (d) Particle 4. (e) Particle 5. (f) Particle 6. (g) Particle 7. (h) Particle 8. (i) Particle 9.

$$\tilde{w}_k^{(i,j,c)} = w_{k-1}^{(i,j,c)} \frac{P(y_k | \{s_{1:t_k}, \tau_{1:t_k}\}^{(i,j)}) P(\{s_{1:t_k}, \tau_{1:t_k}\}^{(i,j)} \setminus \{s_{1:t_{k-1}}, \tau_{1:t_{k-1}}\}^{(i,j)} | \{s_{1:t_{k-1}}, \tau_{1:t_{k-1}}\}^{(i,j)}, M_c)}{P(\{s_{1:t_k}, \tau_{1:t_k}\}^{(i,j)} \setminus \{s_{1:t_{k-1}}, \tau_{1:t_{k-1}}\}^{(i,j)} | \{s_{1:t_{k-1}}, \tau_{1:t_{k-1}}\}^{(i,j)}, M_j)}. \quad (36)$$

Having processed the k measurement, the i th particle in the j th stratum stores the time of the hypothesised last so-

jour, $\tau_{t_k}^{(i,j)}$, the current state, $s_{t_k}^{(i,j)}$, a mean and covariance for x_{t_k} , a weight for each class, $w_k^{(c|i,j)}$, and a weight, $w_k^{(i|j)}$.

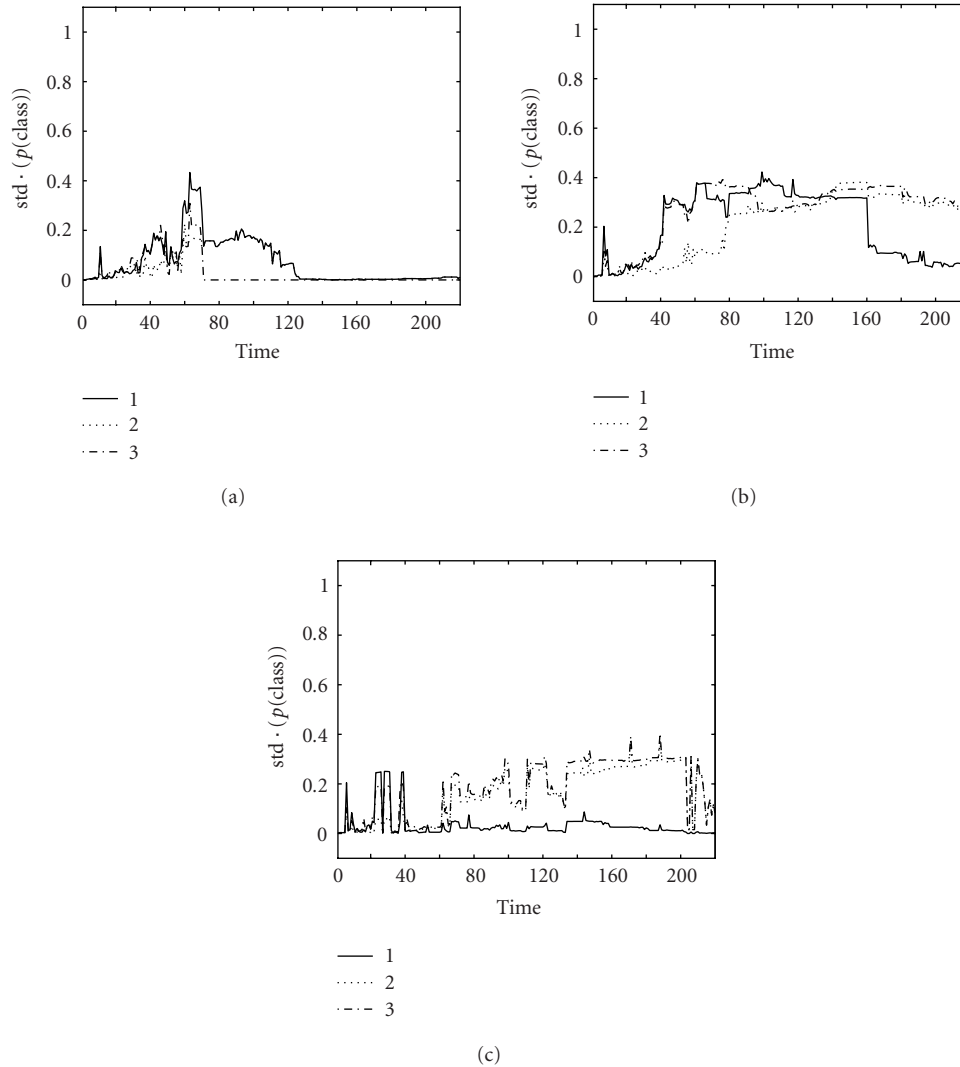


FIGURE 5: Standard deviation (std) of estimated classification probabilities, ($p(\text{class})$), across ten filter runs for simulations according to each of the three models, labelled as 1, 2, and 3. (a) Data simulated from class 1. (b) Data simulated from class 2. (c) Data simulated from class 3.

Each stratum also stores $w_k^{(j)}$. The reader is referred to the preceding sections' summaries of the algorithms for the implementation details.

$N_P = 25$ particles are used per stratum, each is initialised as described previously with a uniform distribution over the classes and with the weights on the strata initialised as being equal. Resampling for a given stratum takes place if the approximate effective sample size given in (11) for the stratum falls below $N_T = 12.5$. Since each of the $N_M = 3$ strata has $N_P = 25$ particles, the computational cost is approximately that of a multiple hypothesis tracker which maintains 75 hypotheses; the algorithm is practicable in terms of its computational expense.

However, it should be noted that, for this difficult problem of joint tracking and classification using very similar models, the number of particles used is small. This is intentional and is motivated by the need to look at the difference

between the variance in the class membership probabilities and the variance of the strata weights.

Ten runs were conducted with data simulated according to each of the three models. The number of particles used is deliberately sufficiently small that the inevitable accumulation of errors causes problems in the time frame considered. This enables a comparison between the time variation in the variance across the runs of the classification probabilities and the variance across the runs of the strata weights. So, Figures 5 and 6 show the time variation in the variance across the runs of these two quantities. It is indeed evident that there is significant variation across the runs; the errors are indeed accumulating with time. It is also evident that this accumulation is faster for the importance weights than for the classification probabilities. This implies that the choice of importance function is less important, in terms of robustness of the estimation of the classification probabilities, than

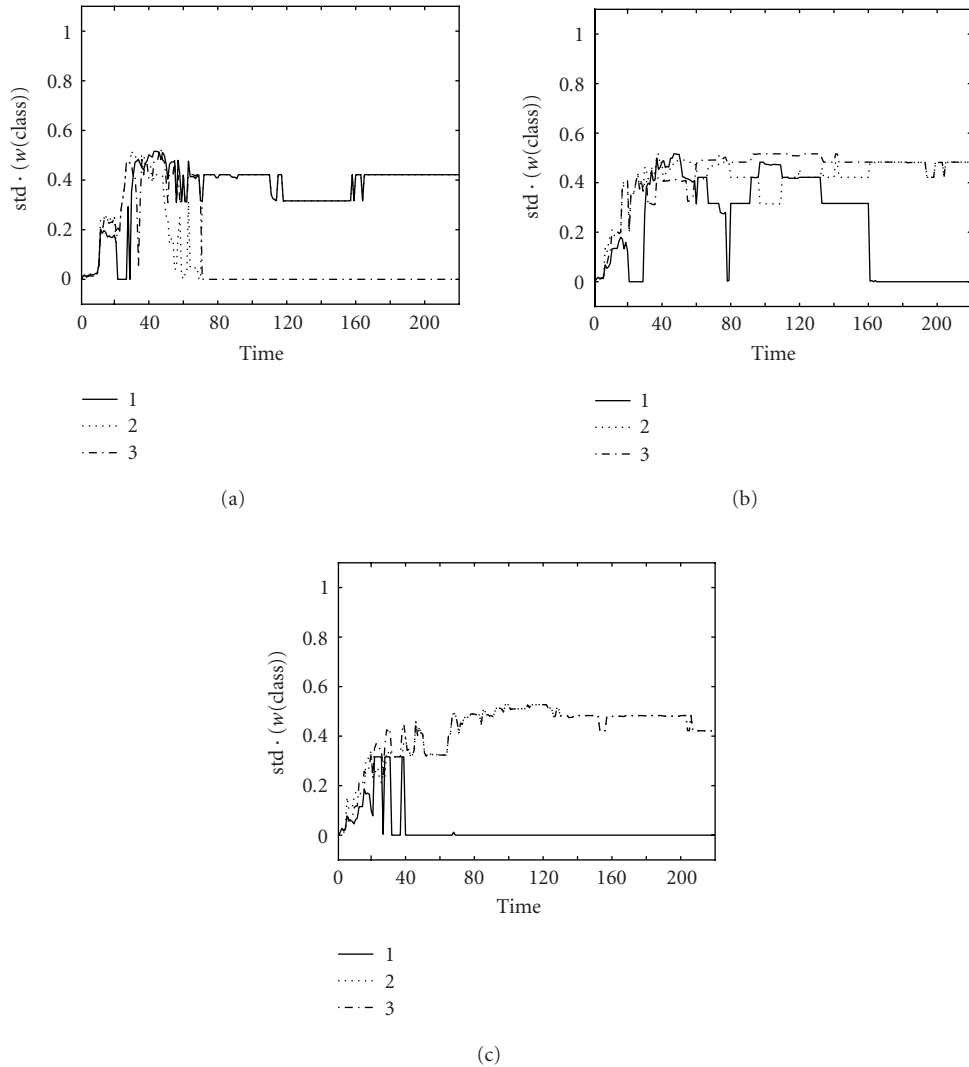


FIGURE 6: Variance of strata weights across ten filter runs for simulations according to each of the three models. (a) Data simulated from class 1. (b) Data simulated from class 2. (c) Data simulated from class 3.

calculating the probabilities of all the classes for every sample.

It is difficult to draw many conclusions from the variations across the true class. Since such issues are quite specific to the models and parameters, which are not the focus of this paper, this is not further investigated or discussed.

7. CONCLUSIONS

Particle filtering has been applied to the use of semi-Markov models for tracking manoeuvring targets. An architecture has been proposed that enables particle filters to be both robust and efficient when classifying targets on the basis of their dynamic behaviour. It has been demonstrated that it is possible to jointly track such manoeuvring targets and classify their manoeuvrability.

ACKNOWLEDGMENTS

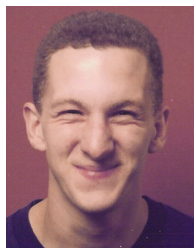
This work was funded by the UK MoD Corporate Research Programme. The author also gratefully acknowledges the award of his Industrial Fellowship by the Royal Commission for the Exhibition of 1851. The author would like to thank John Boyd and Dave Sworder for useful discussions on the subject of joint tracking and classification associated with the use of semi-Markov models and Arnaud Doucet and Matthew Orton for discussions of how particle filters can be used in such problems. The author also thanks the reviewers for their comments.

REFERENCES

- [1] D. R. Cox and V. Isham, *Point Processes*, Chapman and Hall, New York, NY, USA, 1980.
- [2] L. Campo, P. Mookerjee, and Y. Bar-Shalom, "State estimation for systems with sojourn-time-dependent Markov model

- switching,” *IEEE Trans. Automatic Control*, vol. 36, no. 2, pp. 238–243, 1991.
- [3] D. D. Sworder, M. Kent, R. Vojak, and R. G. Hutchins, “Renewal models for maneuvering targets,” *IEEE Trans. on Aerospace and Electronics Systems*, vol. 31, no. 1, pp. 138–150, 1995.
- [4] N. J. Gordon, S. Maskell, and T. Kirubarajan, “Efficient particle filters for joint tracking and classification,” in *Signal and Data Processing of Small Targets*, vol. 4728 of *Proceedings SPIE*, pp. 439–449, Orlando, Fla, USA, April 2002.
- [5] S. Challa and G. Pulford, “Joint target tracking and classification using radar and ESM sensors,” *IEEE Trans. on Aerospace and Electronics Systems*, vol. 37, no. 3, pp. 1039–1055, 2001.
- [6] Y. Boers and H. Driessen, “Integrated tracking and classification: an application of hybrid state estimation,” in *Signal and Data Processing of Small Targets*, vol. 4473 of *Proceedings SPIE*, pp. 198–209, San Diego, Calif, USA, July 2001.
- [7] S. Herman and P. Moulin, “A particle filtering approach to FM-band passive radar tracking and automatic target recognition,” in *Proc. IEEE Aerospace Conference*, vol. 4, pp. 1789–1808, BigSky, Mont, USA, March 2002.
- [8] M. S. Arulampalam, S. Maskell, N. J. Gordon, and T. Clapp, “A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking,” *IEEE Trans. Signal Processing*, vol. 50, no. 2, pp. 174–188, 2002.
- [9] J. Vermaak, S. J. Godsill, and A. Doucet, “Radial basis function regression using trans-dimensional sequential Monte Carlo,” in *Proc. 12th IEEE Workshop on Statistical Signal Processing*, pp. 525–528, St. Louis, Mo, USA, October 2003.
- [10] A. Doucet, J. F. G. de Freitas, and N. J. Gordon, Eds., *Sequential Monte Carlo Methods in Practice*, Springer, New York, NY, USA, 2001.
- [11] F. Le Gland and N. Oudjane, “Stability and uniform approximation of nonlinear filters using the Hilbert metric, and application to particle filters,” Tech. Rep. RR-4215, INRIA, Chesnay Cedex France, 2001.
- [12] D. Crisan and A. Doucet, “Convergence of sequential Monte Carlo methods,” Tech. Rep. CUED/F-INFENG/TR381, Department of Engineering, University of Cambridge, Cambridge, UK, 2000.
- [13] P. M. Baggenstoss, “The PDF projection theorem and the class-specific method,” *IEEE Trans. Signal Processing*, vol. 51, no. 3, pp. 672–685, 2003.
- [14] Y. Bar-Shalom and X. R. Li, *Multitarget-Multisensor Tracking: Principles and Techniques*, YBS Publishing, Storrs, Conn, USA, 1995.

Simon Maskell is a Ph.D. degree graduand and holds a first-class degree with Distinction from Department of Engineering, University of Cambridge. His Ph.D. degree was funded by one of six prestigious Royal Commission for the Exhibition of 1851 Industrial Fellowships awarded on the basis of outstanding excellence to researchers working in British industry. At QinetiQ, he is a Lead Researcher for tracking in the Advanced Signal and Information Processing Group (ASIP). As such, he leads several projects and coordinates ASIP’s tracking research while also supervising a number of other researchers. Simon has authored a number of papers, as well as several technical reports and a patent. He has also been the leading force behind the development of a QinetiQ product to provide a generic solution to all of QinetiQ’s tracking problems.



Bearings-Only Tracking of Manoeuvring Targets Using Particle Filters

M. Sanjeev Arulampalam

Maritime Operations Division, Defence Science and Technology Organisation (DSTO), Edinburgh, South Australia 5111, Australia
Email: sanjeev.arulampalam@dsto.defence.gov.au

B. Ristic

Intelligence, Surveillance & Reconnaissance Division, Defence Science and Technology Organisation (DSTO), Edinburgh, South Australia 5111, Australia
Email: branko.ristic@dsto.defence.gov.au

N. Gordon

Intelligence, Surveillance & Reconnaissance Division, Defence Science and Technology Organisation (DSTO), Edinburgh, South Australia 5111, Australia
Email: neil.gordon@dsto.defence.gov.au

T. Mansell

Maritime Operations Division, Defence Science and Technology Organisation (DSTO), Edinburgh, South Australia 5111, Australia
Email: todd.mansell@dsto.defence.gov.au

Received 2 June 2003; Revised 17 December 2003

We investigate the problem of bearings-only tracking of manoeuvring targets using particle filters (PFs). Three different (PFs) are proposed for this problem which is formulated as a multiple model tracking problem in a jump Markov system (JMS) framework. The proposed filters are (i) multiple model PF (MMPF), (ii) auxiliary MMPF (AUX-MMPF), and (iii) jump Markov system PF (JMS-PF). The performance of these filters is compared with that of standard interacting multiple model (IMM)-based trackers such as IMM-EKF and IMM-UKF for three separate cases: (i) single-sensor case, (ii) multisensor case, and (iii) tracking with hard constraints. A conservative CRLB applicable for this problem is also derived and compared with the RMS error performance of the filters. The results confirm the superiority of the PFs for this difficult nonlinear tracking problem.

Keywords and phrases: bearings-only tracking, manoeuvring target tracking, particle filter.

1. INTRODUCTION

The problem of bearings-only tracking arises in a variety of important practical applications. Typical examples are submarine tracking (using a passive sonar) or aircraft surveillance (using a radar in a passive mode or an electronic warfare device) [1, 2, 3]. The problem is sometimes referred to as target motion analysis (TMA), and its objective is to track the kinematics (position and velocity) of a moving target using noise-corrupted bearing measurements. In the case of autonomous TMA (single observer only), which is the focus of a major part of this paper, the observation platform needs to manoeuvre in order to estimate the target range [1, 3]. This need for ownship manoeuvre and its impact on target state observability have been explored extensively in [4, 5].

Most researchers in the field of bearings-only tracking have concentrated on tracking a nonmanoeuvring target. Due to inherent nonlinearity and observability issues, it is difficult to construct a finite-dimensional optimal Bayesian filter even for this relatively simple problem. As for the bearings-only tracking of a manoeuvring target, the problem is much more difficult and so far, very limited research has been published in the open literature. For example, interacting multiple model (IMM)-based trackers were proposed in [6, 7] for this problem. These algorithms employ a constant velocity (CV) model along with manoeuvre models to capture the dynamic behaviour of a manoeuvring target scenario. Le Cadre and Tremois [8] modelled the manoeuvring target using the CV model with process noise and developed a tracking filter in the hidden Markov model framework.

This paper presents the application of particle filters (PFs) [9, 10, 11] for bearings-only tracking of manoeuvring targets and compares its performance with traditional IMM-based filters. This work builds on the investigation carried out by the authors in [12] for the same problem. The additional features considered in this paper include (a) use of different manoeuvre models, (b) two additional PFs, and (c) tracking with hard constraints. The error performance of the developed filters is analysed by Monte Carlo (MC) simulations and compared to the theoretical Cramér-Rao lower bounds (CRLBs). Essentially, the manoeuvring target problem is formulated in a jump Markov system (JMS) framework and these filters provide suboptimal solutions to the target state, given a sequence of bearing measurements and the particular JMS framework. In the JMS framework considered in this paper, the target motion is modelled by three switching dynamics models whose evolution follows a Markov chain. One of these models is the standard CV model while the other two correspond to coordinated turn (CT) models that capture the manoeuvre dynamics.

Three different PFs are proposed for this problem: (i) multiple model PF (MMPF), (ii) auxiliary MMPF (AUX-MMPF), and (iii) JMS-PF. The MMPF [12, 13] and AUX-MMPF [14] represent the target state and the mode at every time by a set of paired particles and construct the joint posterior density of the target state and mode, given all measurements. The JMS-PF, on the other hand, involves a hybrid scheme where it uses particles to represent only the distribution of the modes, while mode-conditioned state estimation is carried out using extended Kalman filters (EKFs).

The performance of the above algorithms is compared with two conventional schemes: (i) IMM-EKF and (ii) IMM-UKF. These filters represent the posterior density at each time epoch by a finite Gaussian mixture, and they merge and mix these Gaussian mixture components at every step to avoid the exponential growth in the number of mixture components. The IMM-EKF uses EKFs while the IMM-UKF utilises unscented Kalman filters (UKFs) [15] to compute the mode-conditioned state estimates.

In addition to the autonomous bearings-only tracking problem, two further cases are investigated in the paper: multisensor bearings-only tracking, and tracking with hard constraints. The multisensor bearings-only problem involves a slight modification to the original problem, where a second static sensor sends its target bearing measurements to the original platform. The problem of tracking with hard constraints involves the use of prior knowledge, such as speed constraints, to improve tracker performance.

The organisation of the paper is as follows. Section 2 presents the mathematical formulation for the bearings-only tracking problem for each of the three different cases investigated: (i) single-sensor case, (ii) multisensor case, and (iii) tracking with hard constraints. In Section 3 the relevant CRLBs are derived for all but case (iii) for which the analytic derivation is difficult (due to the non-Gaussian prior and process noise vectors). The tracking algorithms for each case are then presented in Section 4 followed by simulation results in Section 5.

2. PROBLEM FORMULATION

2.1. Single-sensor case

Conceptually, the basic problem in bearings-only tracking is to estimate the trajectory of a target (i.e., position and velocity) from noise-corrupted sensor bearing data. For the case of a single-sensor problem, these bearing data are obtained from a single-moving observer (ownship). The target state vector is

$$\mathbf{x}^t = \begin{bmatrix} x^t & y^t & \dot{x}^t & \dot{y}^t \end{bmatrix}^T, \quad (1)$$

where (x, y) and (\dot{x}, \dot{y}) are the position and velocity components, respectively. The ownship state vector \mathbf{x}^o is similarly defined. We now introduce the relative state vector defined by

$$\mathbf{x} \triangleq \mathbf{x}^t - \mathbf{x}^o = \begin{bmatrix} x & y & \dot{x} & \dot{y} \end{bmatrix}^T \quad (2)$$

for which the discrete-time state equation will be written. The dynamics of a manoeuvring target is modelled by multiple switching regimes, also known as a JMS. We make the assumption that at any time in the observation period, the target motion obeys one of $s = 3$ dynamic behaviour models: (a) CV motion model, (b) clockwise CT model, and (c) anticlockwise CT model. Let $S \triangleq \{1, 2, 3\}$ denote the set of three models for the dynamic motion, and let r_k be the regime variable in effect in the interval $(k - 1, k]$, where k is the discrete-time index. Then, the target dynamics can be mathematically written as

$$\mathbf{x}_{k+1} = \mathbf{f}^{(r_{k+1})}(\mathbf{x}_k, \mathbf{x}_k^o, \mathbf{x}_{k+1}^o) + \mathbf{G}\mathbf{v}_k \quad (r_{k+1} \in S), \quad (3)$$

where

$$\mathbf{G} = \begin{bmatrix} \frac{T^2}{2} & 0 \\ 0 & \frac{T^2}{2} \\ T & 0 \\ 0 & T \end{bmatrix}, \quad (4)$$

T is the sampling time, and \mathbf{v}_k is a 2×1 i.i.d. process noise vector with $\mathbf{v}_k \sim \mathcal{N}(\mathbf{0}, \mathbf{Q})$. The process noise covariance matrix is chosen to be $\mathbf{Q} = \sigma_a^2 \mathbf{I}$, where \mathbf{I} is the 2×2 identity matrix and σ_a is a process noise parameter. Note that $\mathbf{G}\mathbf{v}_k$ in (3) corresponds to a piecewise constant white acceleration noise model [16] which is adequate for the large sampling time chosen in our paper. The mode-conditioned transition function $\mathbf{f}^{(r_{k+1})}(\cdot, \cdot, \cdot)$ in (3) is given by

$$\mathbf{f}^{(r_{k+1})}(\mathbf{x}_k, \mathbf{x}_k^o, \mathbf{x}_{k+1}^o) = \mathbf{F}^{(r_{k+1})}(\mathbf{x}_k) \cdot (\mathbf{x}_k + \mathbf{x}_k^o) - \mathbf{x}_{k+1}^o. \quad (5)$$

Here $\mathbf{F}^{(r_{k+1})}(\cdot)$ is the transition matrix corresponding to mode r_{k+1} , which, for the particular problem of interest, can be specified as follows. $\mathbf{F}^{(1)}(\cdot)$ corresponds to CV motion and is thus given by the standard CV transition matrix:

$$\mathbf{F}^{(1)}(\mathbf{x}_k) = \begin{bmatrix} 1 & 0 & T & 0 \\ 0 & 1 & 0 & T \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \quad (6)$$

The next two transition matrices correspond to CT transitions (clockwise and anticlockwise, respectively). These are given by

$$\mathbf{F}^{(j)}(\mathbf{x}_k) = \begin{bmatrix} 1 & 0 & \frac{\sin(\Omega_k^{(j)}T)}{\Omega_k^{(j)}} & -\frac{(1-\cos(\Omega_k^{(j)}T))}{\Omega_k^{(j)}} \\ 0 & 1 & \frac{(1-\cos(\Omega_k^{(j)}T))}{\Omega_k^{(j)}} & \frac{\sin(\Omega_k^{(j)}T)}{\Omega_k^{(j)}} \\ 0 & 0 & \cos(\Omega_k^{(j)}T) & -\sin(\Omega_k^{(j)}T) \\ 0 & 0 & \sin(\Omega_k^{(j)}T) & \cos(\Omega_k^{(j)}T) \end{bmatrix}, \quad j = 2, 3, \quad (7)$$

where the mode-conditioned turning rates are

$$\Omega_k^{(2)} = \frac{a_m}{\sqrt{(\dot{x}_k + \dot{x}_k^o)^2 + (\dot{y}_k + \dot{y}_k^o)^2}}, \quad (8)$$

$$\Omega_k^{(3)} = \frac{-a_m}{\sqrt{(\dot{x}_k + \dot{x}_k^o)^2 + (\dot{y}_k + \dot{y}_k^o)^2}}.$$

Here $a_m > 0$ is a typical manoeuvre acceleration. Note that the turning rate is expressed as a function of target speed (a nonlinear function of the state vector \mathbf{x}_k) and thus models 2 and 3 are clearly nonlinear transitions.

We model the mode r_k in effect at $[k-1, k]$ by a time-homogeneous 3-state first-order Markov chain with known transition probability matrix $\mathbf{\Pi}$, whose elements are

$$\pi_{ij} \triangleq \mathbb{P}(r_k = j | r_{k-1} = i), \quad i, j \in S, \quad (9)$$

such that

$$\pi_{ij} \geq 0, \quad \sum_j \pi_{ij} = 1. \quad (10)$$

The initial probabilities are denoted by $\pi_i \triangleq \mathbb{P}(r_1 = i)$ for $i \in S$ and they satisfy

$$\pi_i \geq 0, \quad \sum_i \pi_i = 1. \quad (11)$$

The available measurement at time k is the angle from the observer's platform to the target, referenced (clockwise positive) to the y -axis and is given by

$$z_k = h(\mathbf{x}_k) + w_k, \quad (12)$$

where w_k is a zero-mean independent Gaussian noise with variance σ_w^2 and

$$h(\mathbf{x}_k) = \arctan\left(\frac{x_k}{y_k}\right) \quad (13)$$

is the true bearing angle. The state variable of interest for estimation is the hybrid state vector $\mathbf{y}_k = (\mathbf{x}_k^T, r_k)^T$. Thus, given a set of measurements $\mathbf{Z}_k = \{z_1, \dots, z_k\}$ and the jump-Markov model (3), the problem is to obtain estimates of the hybrid state vector \mathbf{y}_k . In particular, we are interested in computing the kinematic state estimate $\hat{\mathbf{x}}_{k|k} = \mathbb{E}[\mathbf{x}_k | \mathbf{Z}_k]$ and mode probabilities $\mathbb{P}(r_k = j | \mathbf{Z}_k)$, for every $j \in S$.

2.2. Multisensor case

Suppose there is a possibility of the ownship receiving additional (secondary) bearing measurements from a sensor located at (x_k^s, y_k^s) whose measurement errors are independent to that of the ownship sensor. For simplicity, we assume that (a) additional measurements are synchronous to the primary sensor measurements that (b) there is a zero transmission delay from the sensor at (x_k^s, y_k^s) to the ownship at (x_k^o, y_k^o) . The secondary measurement can be modelled as

$$z'_k = h'(\mathbf{x}_k) + w'_k, \quad (14)$$

where

$$h'(\mathbf{x}_k) = \arctan\left(\frac{x_k + x_k^o - x_k^s}{y_k + y_k^o - y_k^s}\right) \quad (15)$$

and w'_k is a zero-mean white Gaussian noise sequence with variance $\sigma_{w'}^2$. If the additional bearing measurement is not received at time k , we set $z'_k = \emptyset$. The bearings-only tracking problem for this multisensor case is then to estimate the state vector \mathbf{x}_k given a sequence of measurements $\mathbf{Z}_k = \{z_1, z'_1, \dots, z_k, z'_k\}$.

2.3. Tracking with constraints

In many tracking problems, one has some hard constraints on the state vector which can be a valuable source of information in the estimation process. For example, we may know the minimum and maximum speeds of the target given by the constraint

$$s_{\min} \leq \sqrt{(\dot{x}_k + \dot{x}_k^o)^2 + (\dot{y}_k + \dot{y}_k^o)^2} \leq s_{\max}. \quad (16)$$

Suppose some constraint (such as the speed constraint) is imposed on the state vector, and denote the set of constrained state vectors by Ψ . Let the initial distribution of the state vector in the absence of constraints be $\mathbf{x}_0 \sim p(\mathbf{x}_0)$. With constraints, this initial distribution becomes a truncated density $\tilde{p}(\mathbf{x}_0)$, that is,

$$\tilde{p}(\mathbf{x}_0) = \begin{cases} \frac{p(\mathbf{x}_0)}{\int_{\mathbf{x}_0 \in \Psi} p(\mathbf{x}_0) d\mathbf{x}_0}, & \mathbf{x}_0 \in \Psi, \\ 0 & \text{otherwise.} \end{cases} \quad (17)$$

Likewise, the dynamics model should be modified in such a way that \mathbf{x}_k is always constrained to Ψ . In the absence of hard constraints, suppose that the process noise $\mathbf{v}_k \sim g(\mathbf{v})$ is used in the filter. With constraints, the pdf of \mathbf{v}_k becomes a state-dependent truncated density given by

$$\tilde{g}(\mathbf{v}; \mathbf{x}_k) = \begin{cases} \frac{g(\mathbf{v})}{\int_{\mathbf{v} \in G(\mathbf{x}_k)} g(\mathbf{v}) d\mathbf{v}}, & \mathbf{v} \in G(\mathbf{x}_k), \\ 0 & \text{otherwise,} \end{cases} \quad (18)$$

where $G(\mathbf{x}_k) = \{\mathbf{v} : \mathbf{x}_k \in \Psi\}$.

For the bearings-only tracking problem, we will consider hard constraints in target dynamics only. The measurement model remains the same as that for the unconstrained problem. Given a sequence of measurements \mathbf{Z}_k and some constraint Ψ on the state vector, the aim is to obtain estimates of the state vector \mathbf{x}_k , that is,

$$\begin{aligned} \hat{\mathbf{x}}_{k|k} &= \mathbb{E}[\mathbf{x}_k | \mathbf{Z}_k, \Psi] \\ &= \int \mathbf{x}_k p(\mathbf{x}_k | \mathbf{Z}_k, \Psi) d\mathbf{x}_k, \end{aligned} \quad (19)$$

where $p(\mathbf{x}_k | \mathbf{Z}_k, \Psi)$ is the posterior density of the state, given the measurements and hard constraints.

3. CRAMÉR-RAO LOWER BOUNDS

We follow the approach taken in [12] for the development of a conservative CRLB for the manoeuvring target tracking problem. This bound assumes that the true model history of the target trajectory

$$\mathcal{H}_k^* = \{r_1^*, r_2^*, \dots, r_k^*\} \quad (20)$$

is known a priori. Then, a bound on the covariance of $\hat{\mathbf{x}}_k$ was shown to be

$$\begin{aligned} &\mathbb{E}[(\hat{\mathbf{x}}_k - \mathbf{x}_k)(\hat{\mathbf{x}}_k - \mathbf{x}_k)^T] \\ &\geq \mathbb{E}[(\hat{\mathbf{x}}_k - \mathbf{x}_k)(\hat{\mathbf{x}}_k - \mathbf{x}_k)^T | \mathcal{H}_k^*] \\ &\geq [\mathbf{J}_k^*]^{-1}, \end{aligned} \quad (21)$$

where the mode-history-conditioned information matrix \mathbf{J}_k^* is

$$\mathbf{J}_k^* = \mathbb{E}[(\nabla_{\mathbf{x}_k} \log p(\mathbf{x}_k, \mathbf{Z}_k))(\nabla_{\mathbf{x}_k} \log p(\mathbf{x}_k, \mathbf{Z}_k))^T | \mathcal{H}_k^*]. \quad (22)$$

Following [17], a recursion for \mathbf{J}_k^* can be written as

$$\mathbf{J}_{k+1}^* = \mathbf{D}_k^{22} - \mathbf{D}_k^{21}(\mathbf{J}_k^* + \mathbf{D}_k^{11})^{-1}\mathbf{D}_k^{12}, \quad (23)$$

where, in the case of additive Gaussian noise models applicable to our problem, matrices \mathbf{D}_k^{ij} are given by

$$\begin{aligned} \mathbf{D}_k^{11} &= \mathbb{E}\left\{\left(\tilde{\mathbf{F}}_k^{(r_{k+1}^*)}\right)^T \mathbf{Q}_k^{-1} \tilde{\mathbf{F}}_k^{(r_{k+1}^*)}\right\}, \\ \mathbf{D}_k^{12} &= -\mathbb{E}\left\{\left(\tilde{\mathbf{F}}_k^{(r_{k+1}^*)}\right)^T\right\} \mathbf{Q}_k^{-1} = (\mathbf{D}_k^{21})^T, \\ \mathbf{D}_k^{22} &= \mathbf{Q}_k^{-1} + \mathbb{E}\left\{\tilde{\mathbf{H}}_{k+1}^T \mathbf{R}_{k+1}^{-1} \tilde{\mathbf{H}}_{k+1}\right\}, \end{aligned} \quad (24)$$

where

$$\begin{aligned} \tilde{\mathbf{F}}_k^{(r_{k+1}^*)} &= \left[\nabla_{\mathbf{x}_k}(\mathbf{f}^{(r_{k+1}^*)}(\mathbf{x}_k))\right]^T, \\ \tilde{\mathbf{H}}_{k+1} &= \left[\nabla_{\mathbf{x}_{k+1}} h_{k+1}^T(\mathbf{x}_{k+1})\right]^T, \end{aligned} \quad (25)$$

$\mathbf{R}_{k+1} = \sigma_\beta^2 \rightarrow \mathbf{R}_{k+1} = \sigma_\theta^2$ is the variance of the bearing measurements, and \mathbf{Q}_k is the process noise covariance matrix. The Jacobian $\tilde{\mathbf{F}}_k^{(1)}$ for the case of $r_{k+1}^* = 1$ is simply the transition matrix given in (6). For $r_{k+1}^* \in \{2, 3\}$, the Jacobian $\tilde{\mathbf{F}}_k^{(r_{k+1}^*)}$ can be computed as

$$\tilde{\mathbf{F}}_k^{(j)} = \begin{bmatrix} 1 & 0 & \frac{\partial f_1^{(j)}}{\partial \dot{x}_k} & \frac{\partial f_1^{(j)}}{\partial \dot{y}_k} \\ 0 & 1 & \frac{\partial f_2^{(j)}}{\partial \dot{x}_k} & \frac{\partial f_2^{(j)}}{\partial \dot{y}_k} \\ 0 & 0 & \frac{\partial f_3^{(j)}}{\partial \dot{x}_k} & \frac{\partial f_3^{(j)}}{\partial \dot{y}_k} \\ 0 & 0 & \frac{\partial f_4^{(j)}}{\partial \dot{x}_k} & \frac{\partial f_4^{(j)}}{\partial \dot{y}_k} \end{bmatrix}, \quad j = 2, 3, \quad (26)$$

where $f_i^{(j)}(\cdot)$ denotes the i th element of the dynamics model function $\mathbf{f}^{(j)}(\cdot)$. The detailed evaluation of $\tilde{\mathbf{F}}_k^{(j)}$ is given in the appendix.

Likewise, the Jacobian of the measurement function is given by

$$\tilde{\mathbf{H}}_{k+1} = \begin{bmatrix} \frac{\partial h}{\partial x_{k+1}} & \frac{\partial h}{\partial y_{k+1}} & \frac{\partial h}{\partial \dot{x}_{k+1}} & \frac{\partial h}{\partial \dot{y}_{k+1}} \end{bmatrix}, \quad (27)$$

where

$$\begin{aligned} \frac{\partial h}{\partial x_{k+1}} &= \frac{y_{k+1}}{x_{k+1}^2 + y_{k+1}^2}, & \frac{\partial h}{\partial y_{k+1}} &= \frac{-x_{k+1}}{x_{k+1}^2 + y_{k+1}^2}, \\ \frac{\partial h}{\partial \dot{x}_{k+1}} &= \frac{\partial h}{\partial \dot{y}_{k+1}} = 0. \end{aligned} \quad (28)$$

For the case of additional measurements from a secondary sensor, the only change required will be in the computation of \mathbf{D}_k^{22} . In particular, \mathbf{R}_{k+1} and $\tilde{\mathbf{H}}_{k+1}$ will be replaced by \mathbf{R}'_{k+1} and $\tilde{\mathbf{H}}'_{k+1}$, corresponding to the augmented measurement vector (z_{k+1}, z'_{k+1}) . These are given by

$$\mathbf{R}'_{k+1} = \begin{bmatrix} \sigma_{\theta'}^2 & 0 \\ 0 & \sigma_{\theta'}^2 \end{bmatrix},$$

$$\tilde{\mathbf{H}}'_{k+1} = \begin{bmatrix} \frac{\partial h}{\partial x_{k+1}} & \frac{\partial h}{\partial y_{k+1}} & \frac{\partial h}{\partial \dot{x}_{k+1}} & \frac{\partial h}{\partial \dot{y}_{k+1}} \\ \frac{\partial h'}{\partial x_{k+1}} & \frac{\partial h'}{\partial y_{k+1}} & \frac{\partial h'}{\partial \dot{x}_{k+1}} & \frac{\partial h'}{\partial \dot{y}_{k+1}} \end{bmatrix}, \quad (29)$$

where $\sigma_{\theta'}^2 \rightarrow \sigma_{\theta}^2$ is the noise variance of the secondary sensor, and the first row of $\tilde{\mathbf{H}}'_{k+1}$ is identical to (27). The elements of the second row of $\tilde{\mathbf{H}}'_{k+1}$ are given by

$$\frac{\partial h'}{\partial x_{k+1}} = \frac{y_{k+1} + y_{k+1}^{01} - y_{k+1}^{02}}{(x_{k+1} + x_{k+1}^{01} - x_{k+1}^{02})^2 + (y_{k+1} + y_{k+1}^{01} - y_{k+1}^{02})^2},$$

$$\frac{\partial h'}{\partial y_{k+1}} = \frac{-(x_{k+1} + x_{k+1}^{01} - x_{k+1}^{02})}{(x_{k+1} + x_{k+1}^{01} - x_{k+1}^{02})^2 + (y_{k+1} + y_{k+1}^{01} - y_{k+1}^{02})^2},$$

$$\frac{\partial h'}{\partial \dot{x}_{k+1}} = \frac{\partial h'}{\partial \dot{y}_{k+1}} = 0. \quad (30)$$

The simulation experiments for this problem will be carried out on fixed trajectories. This means that for the corresponding CRLBs, the expectation operators in (24) vanish and the required Jacobians will be computed at the true trajectories. The recursion (23) is initialised by

$$\mathbf{J}_1^* = \mathbf{P}_1^{-1}, \quad (31)$$

where \mathbf{P}_1 is the initial covariance matrix of the state estimate. This can be computed using the expression (38), where we replace the measurement θ_1 by the true initial bearing.

4. TRACKING ALGORITHMS

This section describes five recursive algorithms designed for tracking a manoeuvring target using bearings-only measurements. Two of the algorithms are IMM-based algorithms and the other three are PF-based schemes. The algorithms considered are (i) IMM-EKF, (ii) IMM-UKF, (iii) MMPPF, (iv) AUX-MMPPF, and (v) JMS-PF. All five algorithms are applicable to both single-sensor and multisensor tracking problems, formulated in Section 2.

Sections 4.1, 4.2, 4.3, 4.4, and 4.5 will present the elements of the five tracking algorithms to be investigated. The IMM-based trackers will not be presented in detail; the interested reader is referred to [7, 12, 16] for a detailed exposition of these trackers. Section 4.6 presents the required methodology for the multisensor case while Section 4.7 discusses the modifications required in the PF-based trackers for tracking with hard constraints.

4.1. IMM-EKF algorithm

The IMM-EKF algorithm is an EKF-based routine that has been utilised for manoeuvring target tracking problems formulated in a JMS framework [7, 12]. The basic idea is that, for each dynamic model of the JMS, a separate EKF is used, and the filter outputs are weighted according to the mode probabilities to give the state estimate and covariance. At each time index, the target state pdf is characterised by a finite Gaussian mixture which is then propagated to the next time index. Ideally, this propagation results in an s -fold increase in the number of mixture components, where s is the number of modes in the JMS. However, the IMM-EKF algorithm avoids this growth by merging groups of components using mixture probabilities. The details of the IMM-EKF algorithm can be found in [7], where slightly different motion models to the one used here were proposed.

The sources of approximation in the IMM-EKF algorithm are twofold. First, the EKF approximates nonlinear transformations by linear transformations at some operating point. If the nonlinearity is severe or if the operating point is not chosen properly, the resultant approximation can be poor, leading to filter divergence. Second, the IMM approximates the exponentially growing Gaussian mixture with a finite Gaussian mixture. The above two approximations can cause filter instability in certain scenarios.

Next, we provide details of the filter initialisation for the EKF routines used in this algorithm.

4.1.1. Filter initialisation

Suppose the initial prior range is $r \sim \mathcal{N}(\bar{r}, \sigma_r^2)$, where \bar{r} and σ_r^2 are the mean and variance of the initial range. Then, given the first bearing measurement θ_1 , the position components of the relative target state vector is initialised according to standard procedure [12], that is,

$$x_1 = \bar{r} \sin \theta_1, \quad y_1 = \bar{r} \cos \theta_1, \quad (32)$$

with covariance

$$\mathbf{P}^{xy} = \begin{bmatrix} \sigma_x^2 & \sigma_{xy} \\ \sigma_{yx} & \sigma_y^2 \end{bmatrix}, \quad (33)$$

$$\sigma_x^2 = \bar{r}^2 \sigma_{\theta}^2 \cos^2 \theta_1 + \sigma_r^2 \sin^2 \theta_1, \quad (34)$$

$$\sigma_y^2 = \bar{r}^2 \sigma_{\theta}^2 \sin^2 \theta_1 + \sigma_r^2 \cos^2 \theta_1, \quad (35)$$

$$\sigma_{xy} = \sigma_{yx} = (\sigma_r^2 - \bar{r}^2 \sigma_{\theta}^2) \sin \theta_1 \cos \theta_1, \quad (36)$$

where σ_{θ} is the bearing-measurement standard deviation. We adopt a similar procedure to initialise the velocity components. The overall relative target state vector can thus be initialised as follows. Suppose we have some prior knowledge of the target speed and course given by $s \sim \mathcal{N}(\bar{s}, \sigma_s^2)$ and $c \sim \mathcal{N}(\bar{c}, \sigma_c^2)$, respectively. Then, the overall relative target state vector is initialised as

$$\hat{\mathbf{x}}_1 = \begin{bmatrix} \bar{r} \sin \theta_1 \\ \bar{r} \cos \theta_1 \\ \bar{s} \sin(\bar{c}) - \dot{x}_1^o \\ \bar{s} \cos(\bar{c}) - \dot{y}_1^o \end{bmatrix}, \quad (37)$$

where $(\dot{x}_1^o, \dot{y}_1^o)$ is the velocity of the ownship at time index 1. The corresponding initial covariance matrix is given by

$$\mathbf{P}_1 = \begin{bmatrix} \sigma_x^2 & \sigma_{xy} & 0 & 0 \\ \sigma_{yx} & \sigma_y^2 & 0 & 0 \\ 0 & 0 & \sigma_{\dot{x}}^2 & \sigma_{\dot{x}\dot{y}} \\ 0 & 0 & \sigma_{\dot{y}\dot{x}} & \sigma_{\dot{y}}^2 \end{bmatrix}, \quad (38)$$

where $\sigma_x^2, \sigma_{xy}, \sigma_{yx}, \sigma_y^2$ are given by (34)–(36), and

$$\begin{aligned} \sigma_{\dot{x}}^2 &= \bar{s}^2 \sigma_c^2 \cos^2(\bar{c}) + \sigma_s^2 \sin^2(\bar{c}), \\ \sigma_{\dot{y}}^2 &= \bar{s}^2 \sigma_c^2 \sin^2(\bar{c}) + \sigma_s^2 \cos^2(\bar{c}), \\ \sigma_{\dot{x}\dot{y}} &= \sigma_{\dot{y}\dot{x}} = (\sigma_s^2 - \bar{s}^2 \sigma_c^2) \sin(\bar{c}) \cos(\bar{c}). \end{aligned} \quad (39)$$

4.2. IMM-UKF algorithm

This algorithm is similar to the IMM-EKF with the main difference being that the model-matched EKFs are replaced by model-matched UKFs [15]. The UKF for model 1 uses the unscented transform (UT) only for the filter update (because only the measurement equation is non-linear). The UKFs for models 2 and 3 use the UT for both the prediction and the update stage of the filter. The IMM-UKF is initialised in a similar manner to that of the IMM-EKF.

4.3. MMPF

The MMPF [12, 13] has been used to solve various manoeuvring target tracking problems. Here we briefly review the basics of this filter.

The MMPF estimates the posterior density $p(\mathbf{y}_k | \mathbf{Z}_k)$, where $\mathbf{y}_k = [\mathbf{x}_k^T, r_k]^T$ is the augmented (hybrid) state vector. In order to recursively compute the PF estimates, the MC representation of $p(\mathbf{y}_k | \mathbf{Z}_k)$ has to be propagated in time. Let $\{\mathbf{y}_{k-1}^i, w_{k-1}^i\}_{i=1}^N$ denote a random measure that characterises the posterior pdf $p(\mathbf{y}_{k-1} | \mathbf{Z}_{k-1})$, where $\mathbf{y}_{k-1}^i, i = 1, \dots, N$, is a set of support points with associated weights $w_{k-1}^i, i = 1, \dots, N$. Then, the posterior density of the augmented state at $k - 1$ can be approximated as

$$p(\mathbf{y}_{k-1} | \mathbf{Z}_{k-1}) \approx \sum_{i=1}^N w_{k-1}^i \delta(\mathbf{y}_{k-1} - \mathbf{y}_{k-1}^i), \quad (40)$$

where $\delta(\cdot)$ is the Dirac delta measure. Next, the posterior pdf at k can be written as

$$\begin{aligned} p(\mathbf{y}_k | \mathbf{Z}_k) &\propto p(z_k | \mathbf{y}_k) \int p(\mathbf{y}_k | \mathbf{y}_{k-1}) p(\mathbf{y}_{k-1} | \mathbf{Z}_{k-1}) d\mathbf{y}_{k-1} \\ &\approx p(z_k | \mathbf{y}_k) \sum_{i=1}^N w_{k-1}^i p(\mathbf{y}_k | \mathbf{y}_{k-1}^i), \end{aligned} \quad (41)$$

where approximation (40) was used in (41). Now, to represent the pdf $p(\mathbf{y}_k | \mathbf{Z}_k)$ using particles, we employ the *importance sampling* method [9]. By choosing the importance density to be $p(\mathbf{y}_k | \mathbf{y}_{k-1})$, one can draw samples $\mathbf{y}_k^{*i} \sim p(\mathbf{y}_k | \mathbf{y}_{k-1}^i)$, $i = 1, \dots, N$. To draw a sample from $p(\mathbf{y}_k | \mathbf{y}_{k-1}^i)$, we first draw a sample from $p(r_k | r_{k-1}^i)$ which is a discrete probability mass

function given by the i th row of the Markov chain transition probability matrix. That is, we choose $r_k^{*i} \sim p(r_k | r_{k-1}^i)$ such that

$$\mathbb{P}(r_k^{*i} = j) = \pi_{ij}. \quad (42)$$

Next, given the mode r_k^{*i} , one can easily sample $\mathbf{x}_k^{*i} \sim p(\mathbf{x}_k | \mathbf{x}_{k-1}^i, r_k^{*i})$ by generating process noise sample $\mathbf{v}_{k-1}^{*i} \sim \mathcal{N}(\mathbf{0}, \mathbf{Q})$ and propagating $\mathbf{x}_{k-1}^i, r_k^{*i}$, and \mathbf{v}_{k-1}^{*i} through the dynamics model (3). This gives us the sample $\{\mathbf{y}_k^{*i} = [(\mathbf{x}_k^{*i})^T, r_k^{*i}]^T\}_{i=1}^N$ which can be used to approximate the posterior pdf $p(\mathbf{y}_k | \mathbf{Z}_k)$ as

$$p(\mathbf{y}_k | \mathbf{Z}_k) \approx \sum_{i=1}^N w_k^i \delta(\mathbf{y}_k - \mathbf{y}_k^{*i}), \quad (43)$$

where

$$w_k^i \propto w_{k-1}^i p(z_k | \mathbf{y}_k^{*i}). \quad (44)$$

Note that $p(z_k | \mathbf{y}_k^{*i}) = p(z_k | \mathbf{x}_k^{*i})$ which can be computed using the measurement equation (12). This completes the description of a single recursion of the MMPF. The filter is initialised by generating N samples $\{\mathbf{x}_1^i\}_{i=1}^N$ from the initial density $\mathcal{N}(\hat{\mathbf{x}}_1, \mathbf{P}_1)$, where $\hat{\mathbf{x}}_1$ and \mathbf{P}_1 were specified in (37) and (38), respectively.

A common problem with PFs is the degeneracy phenomenon, where, after a few iterations, all but one particle will have negligible weight. A measure of degeneracy is the effective sample size N_{eff} which can be empirically evaluated as

$$\hat{N}_{\text{eff}} = \frac{1}{\sum_{i=1}^N w_k^i{}^2}. \quad (45)$$

The usual approach to overcoming the degeneracy problem is to introduce resampling whenever $\hat{N}_{\text{eff}} < N_{\text{thr}}$ for some threshold N_{thr} . The resampling step involves generating a new set $\{\mathbf{y}_k^i\}_{i=1}^N$ by sampling with replacement N times from the set $\{\mathbf{y}_k^{*i}\}_{i=1}^N$ such that

$$\mathbb{P}(\mathbf{y}_k^i = \mathbf{y}_k^{*j}) = w_k^j. \quad (46)$$

4.4. AUX-MMPF

The AUX-MMPF [14] focuses on the characterisation of pdf $p(\mathbf{x}_k, i, r_k | \mathbf{Z}_k)$, where i refers to the i th particle at $k - 1$. This density is marginalised to obtain a representation of $p(\mathbf{x}_k | \mathbf{Z}_k)$.

A proportionality for the joint probability density $p(\mathbf{x}_k, i, r_k | \mathbf{Z}_k)$ can be written using Bayes' rule as

$$\begin{aligned} p(\mathbf{x}_k, i, r_k | \mathbf{Z}_k) &\propto p(z_k | \mathbf{x}_k) p(\mathbf{x}_k, i, r_k | \mathbf{Z}_{k-1}) \\ &= p(z_k | \mathbf{x}_k) p(\mathbf{x}_k | r_k, i, \mathbf{Z}_{k-1}) p(r_k | i, \mathbf{Z}_{k-1}) p(i | \mathbf{Z}_{k-1}) \\ &= p(z_k | \mathbf{x}_k) p(\mathbf{x}_k | \mathbf{x}_{k-1}^i, r_k) p(r_k | r_{k-1}^i) w_{k-1}^i, \end{aligned} \quad (47)$$

where $p(r_k|r_{k-1})$ is an element of the transition probability matrix $\mathbf{\Pi}$ defined by (9). To sample directly from $p(\mathbf{x}_k, i, r_k|\mathbf{Z}_k)$ as given by (47) is not practical. Hence, we again use importance sampling [9, 14] to first obtain a sample from a density which closely resembles (47), and then weight the samples appropriately to produce an MC representation of $p(\mathbf{x}_k, i, r_k|\mathbf{Z}_k)$. This can be done by introducing the function $q(\mathbf{x}_k, i, r_k|\mathbf{Z}_k)$ with proportionality

$$q(\mathbf{x}_k, i, r_k|\mathbf{Z}_k) \propto p(z_k|\mu_k^i(r_k))p(\mathbf{x}_k|\mathbf{x}_{k-1}^i, r_k)p(r_k|r_{k-1}^i)w_{k-1}^i, \quad (48)$$

where

$$\begin{aligned} \mu_k^i(r_k) &= \mathbb{E}\{\mathbf{x}_k|\mathbf{x}_{k-1}^i, r_k\} \\ &= \mathbf{f}^{(r_k)}(\mathbf{x}_{k-1}^i, \mathbf{x}_{k-1}^o, \mathbf{x}_k^o). \end{aligned} \quad (49)$$

Importance density $q(\mathbf{x}_k, i, r_k|\mathbf{Z}_k)$ differs from (47) only in the first factor. Now, we can write $q(\mathbf{x}_k, i, r_k|\mathbf{Z}_k)$ as

$$q(\mathbf{x}_k, i, r_k|\mathbf{Z}_k) = q(i, r_k|\mathbf{Z}_k)g(\mathbf{x}_k|i, r_k, \mathbf{Z}_k) \quad (50)$$

and define

$$g(\mathbf{x}_k|i, r_k, \mathbf{Z}_k) \triangleq p(\mathbf{x}_k|\mathbf{x}_{k-1}^i, r_k). \quad (51)$$

In order to obtain a sample from the density $q(\mathbf{x}_k, i, r_k|\mathbf{Z}_k)$, we first integrate (48) with respect to \mathbf{x}_k to get an expression for $q(i, r_k|\mathbf{Z}_k)$,

$$q(i, r_k|\mathbf{Z}_k) \propto p(z_k|\mu_k^i(r_k))p(r_k|r_{k-1}^i)w_{k-1}^i. \quad (52)$$

A random sample can now be obtained from the density $q(\mathbf{x}_k, i, r_k|\mathbf{Z}_k)$ as follows. First, a sample $\{ij, r_k^j\}_{j=1}^N$ is drawn from the discrete distribution $q(i, r_k|\mathbf{Z}_k)$ given by (52). This can be done by splitting each of the N particles at $k-1$ into s groups (s is the number of possible modes), each corresponding to a particular mode. Each of the sN particles is assigned a weight proportional to (52), and N points $\{ij, r_k^j\}_{j=1}^N$ are then sampled from this discrete distribution. From (50) and (51), it is seen that the samples $\{\mathbf{x}_k^j\}_{j=1}^N$ from the joint density $q(\mathbf{x}_k, i, r_k|\mathbf{Z}_k)$ can now be generated from $p(\mathbf{x}_k|\mathbf{x}_{k-1}^{ij}, r_k^j)$. The resultant triplet sample $\{\mathbf{x}_k^j, ij, r_k^j\}_{j=1}^N$ is a random sample from the density $q(\mathbf{x}_k, i, r_k|\mathbf{Z}_k)$. To use these samples to characterise the density $p(\mathbf{x}_k, i, r_k|\mathbf{Z}_k)$, we attach the weights w_k^j to each particle, where w_k^j is a ratio of (48) and (47), evaluated at $\{\mathbf{x}_k^j, ij, r_k^j\}$, that is,

$$\begin{aligned} w_k^j &= \frac{p(z_k|\mathbf{x}_k^j)p(\mathbf{x}_k^j|\mathbf{x}_{k-1}^{ij}, r_k^j)p(r_k^j|r_{k-1}^{ij})w_{k-1}^{ij}}{p(z_k|\mu_k^{ij}(r_k))p(\mathbf{x}_k^j|\mathbf{x}_{k-1}^{ij}, r_k^j)p(r_k^j|r_{k-1}^{ij})w_{k-1}^{ij}} \\ &= \frac{p(z_k|\mathbf{x}_k^j)}{p(z_k|\mu_k^{ij}(r_k))}. \end{aligned} \quad (53)$$

By defining the augmented vector $\mathbf{y}_k \triangleq (\mathbf{x}_k^T, i, r_k)^T$, we can write down an MC representation of the pdf $p(\mathbf{x}_k, i, r_k|\mathbf{Z}_k)$ as

$$p(\mathbf{x}_k, i, r_k|\mathbf{Z}_k) = p(\mathbf{y}_k) \approx \sum_{j=1}^N w_k^j \delta(\mathbf{y}_k - \mathbf{y}_k^j). \quad (54)$$

Observe that by omitting the $\{ij, r_k^j\}$ components in the triplet sample, we have a representation of the marginalised density $p(\mathbf{x}_k|\mathbf{Z}_k)$, that is,

$$p(\mathbf{x}_k|\mathbf{Z}_k) \approx \sum_{j=1}^N w_k^j \delta(\mathbf{x}_k - \mathbf{x}_k^j). \quad (55)$$

The AUX-MMPPF is initialised according to the same procedure as for MMPPF.

4.5. JMS-PF

The JMS-PF is based on the jump Markov linear system (JMLS) PF proposed in [18, 19] for a JMLS. Let

$$\begin{aligned} \mathbf{X}_k &= \{\mathbf{x}_1, \dots, \mathbf{x}_k\}, \\ R_k &= \{r_1, \dots, r_k\} \end{aligned} \quad (56)$$

denote the sequences of states and modes up to time index k . Standard particle filtering techniques focused on the estimation of the pdf of the state vector \mathbf{x}_k . However, in the JMS-PF, we place emphasis on the estimation of the pdf of the mode sequence R_k , given measurements $\mathbf{Z}_k = \{z_1, \dots, z_k\}$. The density $p(\mathbf{X}_k, R_k|\mathbf{Z}_k)$ can be factorised into

$$p(\mathbf{X}_k, R_k|\mathbf{Z}_k) = p(\mathbf{X}_k|R_k, \mathbf{Z}_k)p(R_k|\mathbf{Z}_k). \quad (57)$$

Given a specific mode sequence R_k and measurements \mathbf{Z}_k , the first term on the right-hand side of (57), $p(\mathbf{X}_k|R_k, \mathbf{Z}_k)$, can easily be estimated using an EKF or some other nonlinear filter. Therefore, we focus our attention on $p(R_k|\mathbf{Z}_k)$; for estimation of this density, we propose to use a PF.

Using Bayes' rule, we note that

$$p(R_k|\mathbf{Z}_k) = \frac{p(z_k|\mathbf{Z}_{k-1}, R_k)p(r_k|r_{k-1})}{p(z_k|\mathbf{Z}_{k-1})}p(R_{k-1}|\mathbf{Z}_{k-1}). \quad (58)$$

Equation (58) provides a useful recursion for the estimation of $p(R_k|\mathbf{Z}_k)$ using a PF. We describe a general recursive algorithm which generates N particles $\{R_k^i\}_{i=1}^N$ at time k which characterises the pdf $p(R_k|\mathbf{Z}_k)$. The algorithm requires the introduction of an importance function $q(r_k|\mathbf{Z}_k, R_{k-1})$. Suppose at time $k-1$, one has a set of particles $\{R_{k-1}^i\}_{i=1}^N$ that characterises the pdf $p(R_{k-1}|\mathbf{Z}_{k-1})$. That is,

$$p(R_{k-1}|\mathbf{Z}_{k-1}) \approx \frac{1}{N} \sum_{i=1}^N \delta(R_{k-1} - R_{k-1}^i). \quad (59)$$

Now draw N samples $r_k^i \sim q(r_k | \mathbf{Z}_k, R_{k-1}^i)$. Then, from (58) and the principle of importance sampling, one can write

$$p(R_k | \mathbf{Z}_k) \approx \sum_{i=1}^N w_k^i \delta(R_k - R_k^i), \quad (60)$$

where $R_k^i \equiv \{R_{k-1}^i, r_k^i\}$ and the weight

$$w_k^i \propto \frac{p(z_k | \mathbf{Z}_{k-1}, R_k^i) p(r_k^i | r_{k-1}^i)}{q(r_k^i | \mathbf{Z}_k, R_{k-1}^i)}. \quad (61)$$

From (60), we note that one can perform resampling (if required) to obtain an approximate i.i.d. sample from $p(R_k | \mathbf{Z}_k)$. The recursion can be initialised according to the specified initial state distribution of the Markov chain, $\pi_i = \mathbb{P}(r_1 = i)$.

How do we choose the importance density $q(r_k | \mathbf{Z}_k, R_{k-1})$? A sensible selection criterion is to choose a proposal that minimises the variance of the importance weights at time k , given R_{k-1} and \mathbf{Z}_k . According to this strategy, it was shown in [18] that the optimal importance density is $p(r_k | \mathbf{Z}_k, R_{k-1}^i)$. Now, it is easy to see that this density satisfies

$$p(r_k | \mathbf{Z}_k, R_{k-1}^i) = \frac{p(z_k | \mathbf{Z}_{k-1}, R_{k-1}^i, r_k) p(r_k | r_{k-1}^i)}{p(z_k | \mathbf{Z}_{k-1}, R_{k-1}^i)}. \quad (62)$$

Note that $p(r_k | \mathbf{Z}_k, R_{k-1}^i)$ is proportional to the numerator of (62) as the denominator is independent of r_k . Also, the term $p(r_k | r_{k-1}^i)$ is simply the Markov chain transition probability (specified by the transition probability matrix $\mathbf{\Pi}$). The term $p(z_k | \mathbf{Z}_{k-1}, R_k)$, which features in the numerator of (62), can be approximated by one-step-ahead EKF outputs, that is, we can write

$$p(z_k | \mathbf{Z}_{k-1}, R_k) \approx \mathcal{N}(v_k(R_k, \mathbf{Z}_{k-1}); 0, \mathbf{S}_k(R_k, \mathbf{Z}_{k-1})), \quad (63)$$

where $v_k(\cdot, \cdot)$ and $\mathbf{S}_k(\cdot, \cdot)$ are the mode-history-conditioned innovation and its covariance, respectively. Thus, $p(r_k | r_{k-1}^i)$ and (63) allow the computation of the optimal importance density.

Using (62) as the importance density $q(\cdot | \cdot, \cdot)$ in (61), we find that the weight

$$w_k^i \propto p(z_k | \mathbf{Z}_{k-1}, R_{k-1}^i). \quad (64)$$

Since $r_k \in \{1, \dots, s\}$, the importance weights given above can be computed as

$$w_k^i \propto p(z_k | \mathbf{Z}_{k-1}, R_{k-1}^i) = \sum_{j=1}^s p(z_k | \mathbf{Z}_{k-1}, R_{k-1}^i, r_k = j) p(r_k = j | r_{k-1}^i). \quad (65)$$

Note that the computation of the importance weights in (65) requires s one-step-ahead EKF innovations and their covariances.

This completes the description of the PF for estimation of the Markov chain distribution $p(R_k | \mathbf{Z}_k)$. As mentioned earlier, given a particular mode sequence, the state estimates are easily obtained using a standard EKF.

4.6. Methodology for the multisensor case

The methodology for the multisensor case is similar to that of the single-sensor case. The two main points to note for this case are that (a) the secondary measurement is processed in a sequential manner assuming a zero time delay between the primary and secondary measurements and (b) for the processing of the secondary measurement, the measurement function (15) is used in place of (13) for the computation of the necessary quantities such as Jacobians, predicted measurements, and weights.

4.7. Modifications for tracking with hard constraints

The problem of bearings-only tracking with hard constraints was described in Section 2.3. Recall that for the constraint $\mathbf{x}_k \in \Psi$, the state estimate is given by the mean of the posterior density $p(\mathbf{x}_k | \mathbf{Z}_k, \Psi)$. This density cannot be easily constructed by standard Kalman-filter-based techniques. However, since PFs make no restrictions on the prior density or the distributions of the process and measurement noise vectors, it turns out that $p(\mathbf{x}_k | \mathbf{Z}_k, \Psi)$ can be constructed using PFs. The only modifications required in the PFs for the case of constraint $\mathbf{x}_k \in \Psi$ are as follows:

- (i) the prior distribution needs to be $\tilde{p}(\mathbf{x})$ defined in (17) and the filter needs to be able to sample from this density;
- (ii) in the prediction step, samples are drawn from the constrained process noise density $\tilde{g}(\mathbf{v}; \mathbf{x}_k)$ instead of the standard process noise pdf.

Both changes require the ability to sample from a truncated density. A simple method to sample from a generic truncated density $\tilde{t}(\mathbf{x})$ defined by

$$\tilde{t}(\mathbf{x}) = \begin{cases} \frac{t(\mathbf{x})}{\int_{\mathbf{x} \in \Psi} t(\mathbf{x}) d\mathbf{x}}, & \mathbf{x} \in \Psi, \\ 0 & \text{otherwise} \end{cases} \quad (66)$$

is as follows. Suppose we can easily sample from $t(\mathbf{x})$. Then, to draw $\mathbf{x} \sim \tilde{t}(\mathbf{x})$, we can use rejection sampling from $t(\mathbf{x})$ until the condition $\mathbf{x} \in \Psi$ is satisfied. The resulting sample is then distributed according to $\tilde{t}(\mathbf{x})$. This simple technique will be adopted in the modifications required in the PF for the constrained problem.¹ With the above modifications, the PF leads to a cloud of particles that characterise the posterior density $p(\mathbf{x}_k | \mathbf{Z}_k, \Psi)$ from which the state estimate $\hat{\mathbf{x}}_{k|k}$ and its covariance $\mathbf{P}_{k|k}$ can be obtained.

¹This rejection sampling scheme can potentially be inefficient. For more efficient schemes to sample from truncated densities, see [20].

5. SIMULATION RESULTS

In this section, we present a performance comparison of the various tracking algorithms described in the previous section. The comparison will be based on a set of 100MC simulations and where possible, the CRLB will be used to indicate the best possible performance that one can expect for a given scenario and a set of parameters. Before proceeding, we give a description of the four performance metrics that will be used in this analysis: (i) RMS position error, (ii) efficiency η , (iii) root time-averaged mean square (RTAMS) error, and (iv) number of divergent tracks.

To define each of the above performance metrics, let (x_k^i, y_k^i) and $(\hat{x}_k^i, \hat{y}_k^i)$ denote the true and estimated target positions at time k at the i th MC run. Suppose M of such MC runs are carried out. Then, the RMS position error at k can be computed as

$$\text{RMS}_k = \sqrt{\frac{1}{M} \sum_{i=1}^M (\hat{x}_k^i - x_k^i)^2 + (\hat{y}_k^i - y_k^i)^2}. \quad (67)$$

Now, if $\mathbf{J}_k^{-1}[i, j]$ denotes the ij th element of the inverse information matrix for the problem at hand, then the corresponding CRLB for the metric (67) can be written as

$$\text{CRLB}(\text{RMS}_k) = \sqrt{\mathbf{J}_k^{-1}[1, 1] + \mathbf{J}_k^{-1}[2, 2]}. \quad (68)$$

The second metric stated above is the efficiency parameter η defined as

$$\eta_k \triangleq \frac{\text{CRLB}(\text{RMS}_k)}{\text{RMS}_k} \times 100\% \quad (69)$$

which indicates ‘‘closeness’’ to CRLB. Thus, $\eta_k = 100\%$ implies an efficient estimator that achieves the CRLB exactly.

For a particular scenario and parameters, the overall performance of a filter is evaluated using the third metric which is the RTAMS error. This is defined as

$$\text{RTAMS} = \sqrt{\frac{1}{(t_{\max} - \ell)M} \sum_{k=\ell+1}^{t_{\max}} \sum_{i=1}^M (\hat{x}_k^i - x_k^i)^2 + (\hat{y}_k^i - y_k^i)^2}, \quad (70)$$

where t_{\max} is the total number of observations (or time epochs) and ℓ is a time index after which the averaging is carried out. Typically ℓ is chosen to coincide with the end of the first ownship manoeuvre.

The final metric stated above is the number of divergent tracks. A track is declared divergent if its estimated position error at any time index exceeds a threshold which is set to be 20 km in our simulations. It must be noted that the first three metrics described above are computed only on nondivergent tracks.

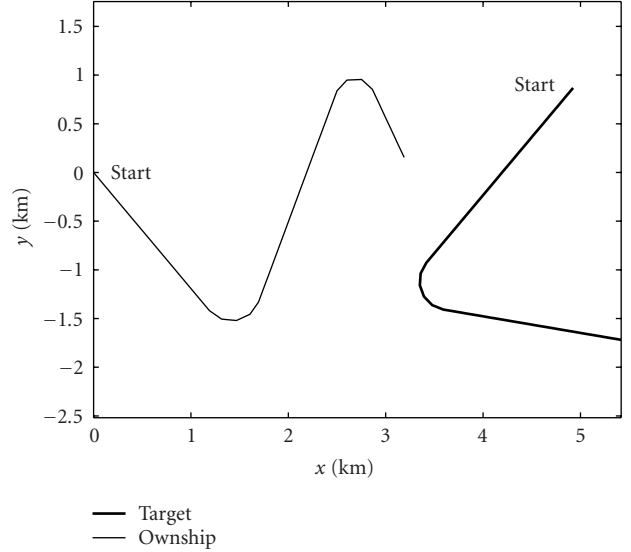


FIGURE 1: A typical bearings-only tracking scenario with a manoeuvring target.

5.1. Single-sensor case

The target-observer geometry for this case is shown in Figure 1. The target which is initially 5 km away from the ownship maintains an initial course of -140° . It executes a manoeuvre in the interval 20–25 minutes to attain a new course of 100° , and maintains this new course for the rest of the observation period. The ownship, travelling at a fixed speed of 5 knots and an initial course of 140° , executes a manoeuvre in the interval 13–17 minutes to attain a new course of 20° . It maintains this course for a period of 15 minutes at the end of which it executes a second manoeuvre and attains a new course of 155° . The final target-observer range for this case is 2.91 km. Bearing measurements with accuracy $\sigma_\theta = 1.5^\circ$ are received every $T = 1$ minute for an observation period of 40 minutes.

Unless otherwise mentioned, the following nominal filter parameters were used in the simulations. The initial range and speed prior standard deviations were set to $\sigma_r = 2$ km and $\sigma_s = 2$ knots, respectively. The initial course and its standard deviation were set to $\bar{c} = \theta_1 + \pi$ and $\sigma_c = \pi/\sqrt{12}$, where θ_1 is the initial bearing measurement. The process noise parameter was set to $\sigma_a = 1.6 \times 10^{-6}$ km/s². The MMPF and AUX-MMPF used $N = 5000$ particles while the JMS-PF used $N = 100$ particles. Resampling was carried out if $\hat{N}_{\text{eff}} < N_{\text{thr}}$, where the threshold was set to $N_{\text{thr}} = N/3$. The resampling scheme used in the simulations is an algorithm based on order statistics [21].

The transition probability matrix required for the jump Markov process was chosen to be

$$\mathbf{\Pi} = \begin{bmatrix} 0.9 & 0.05 & 0.05 \\ 0.4 & 0.5 & 0.1 \\ 0.4 & 0.1 & 0.5 \end{bmatrix} \quad (71)$$

TABLE 1: Performance comparison for the single-sensor case.

Algorithm/ CRLB	RMS error (final)			RTAMS (km)	Improvement (%)	Divergent tracks
	(km)	(%)	η			
IMM-EKF	1.18	40	22	1.07	0	0
IMM-UKF	0.80	28	32	0.72	32	1
MMPF	0.59	20	43	0.44	59	0
AUX-MMPF	0.55	19	46	0.47	56	0
JMS-PF	0.77	27	33	0.64	40	0
CRLB	0.25	9	100	0.21	80	—

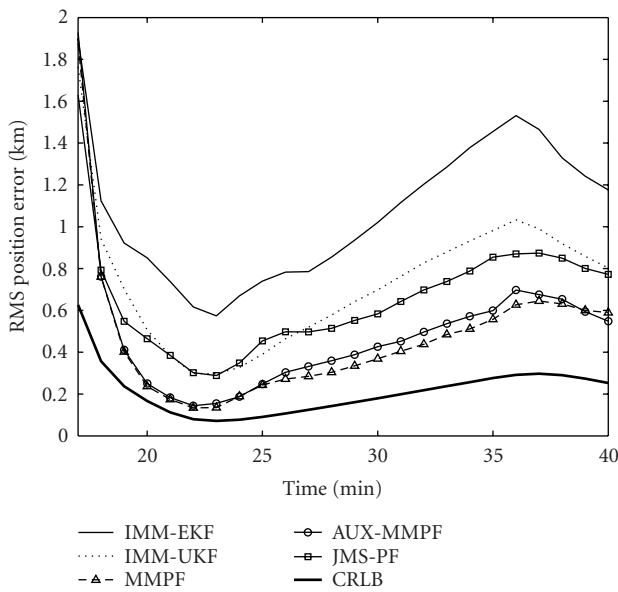


FIGURE 2: RMS position error versus time for a manoeuvring target scenario.

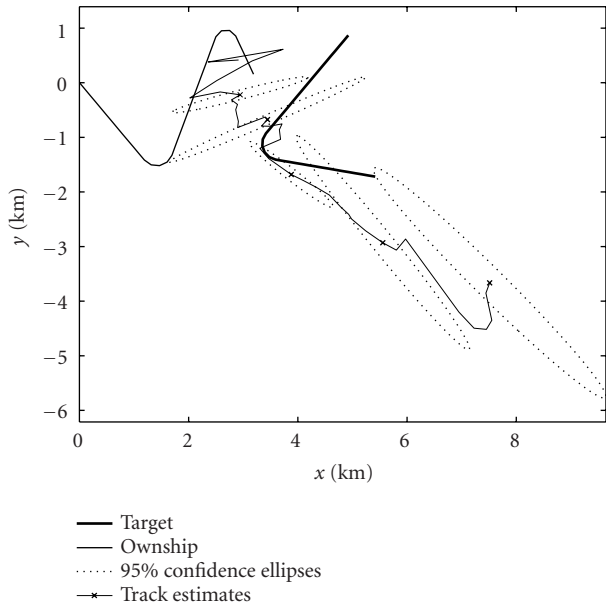
and the typical manoeuvre acceleration parameter for the filters was set to $a_m = 1.08 \times 10^{-5} \text{ km/s}^2$.

Figure 2 shows the RMS error curves corresponding to the five filters: IMM-EKF, IMM-UKF, MMPF, AUX-MMPF, and JMS-PF. A detailed comparison is also given in Table 1. Note that the column “improvement” refers to the percentage improvement in RTAMS error compared with a baseline filter which is chosen to be the IMM-EKF. From the graph and the table, it is clear that the performance of the IMM-EKF and IMM-UKF is poor compared to the other three filters. Though the final RMS error performance of the IMM-UKF is comparable to the JMS-PF, since it has one divergent track, its overall performance is considered worse than that of the JMS-PF. It is clear that the best filters for this case were the MMPF and AUX-MMPF which achieved 59% and 56% improvement, respectively, over the IMM-EKF. Also note that the JMS-PF performance is between that of IMM-EKF/IMM-UKF and MMPF/AUX-MMPF. From the simulations, it appears that the relative computational requirements (with respect to the IMM-EKF) for the IMM-UKF, MMPF, AUX-MMPF, and JMS-PF are 2.6, 23, 32, and 30, respectively.

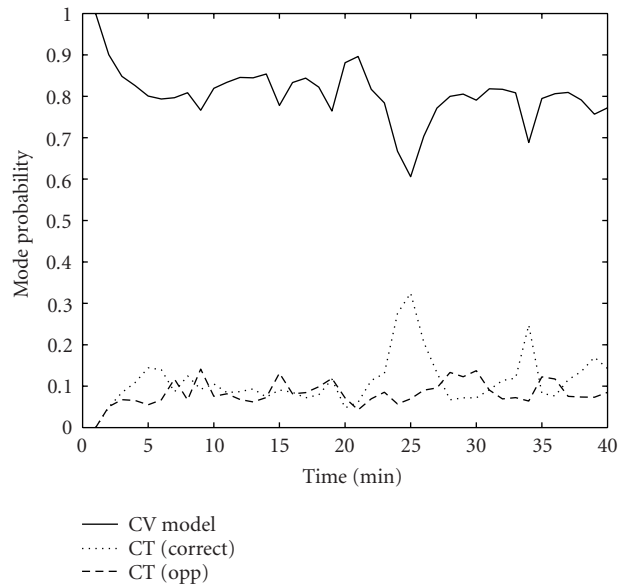
The rationale for the performance differences noted above can be explained as follows. There are two sources of approximations in both IMM-EKF and IMM-UKF. Firstly, the probability of the mode history is approximated by the IMM routine which merges mode histories. Secondly, the mode-conditioned filter estimates are obtained using an EKF and an UKF (for the IMM-EKF and IMM-UKF, respectively), both of which approximate the non-Gaussian posterior density by a Gaussian. In contrast, the MMPF and AUX-MMPF attempt to alleviate both sources of approximations: they estimate the mode probabilities with no merging of histories and they make no linearisation (as in EKF) and characterise the non-Gaussian posterior density in a near-optimal manner. Thus we observe the superior performance of the MMPF and AUX-MMPF. The JMS-PF on the other hand is worse than MMPF/AUX-MMPF but better than IMM-EKF/IMM-UKF as it attempts to alleviate only one of the sources of approximations discussed above. Specifically, while the JMS-PF attempts to compute the mode history probability exactly, it uses an EKF (a local linearisation approximation) to compute the mode-conditioned filtered estimates. Hence, note that even if the number of particles for the JMS-PF is increased, its performance can never reach that of the MMPF/AUX-MMPF. It is interesting to note from the improvement figures for the JMS-PF and MMPF that the first source of approximation is more critical than the second one. In fact, the contributions of the first and second sources of approximation appear to be in the ratio 2 : 1.

It is worth noting that in the above simulations, the performance of the AUX-MMPF is comparable to that of the MMPF. This is expected due to the low process noise used in the simulations as one would expect the performance of the AUX-MMPF to approach that of MMPF as the process noise tends to zero. However, for problems with moderate to high process noise, the AUX-MMPF is likely to outperform the MMPF.

Next, we illustrate a case where the IMM-EKF shows a tendency to diverge while the MMPF tracks the target well for the same set of measurements. Figure 3a shows the estimated track and 95% error ellipses (plotted every 8 minutes) for the IMM-EKF. Note that the IMM-EKF covariance estimate at 8 minutes is poor as it does not encapsulate the true target position. This has resulted in not only subsequent poor track estimates, but also inability to detect the target manoeuvre.

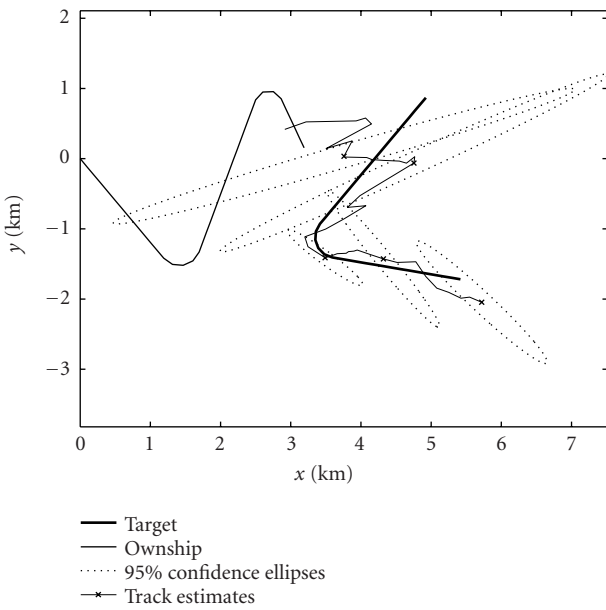


(a)

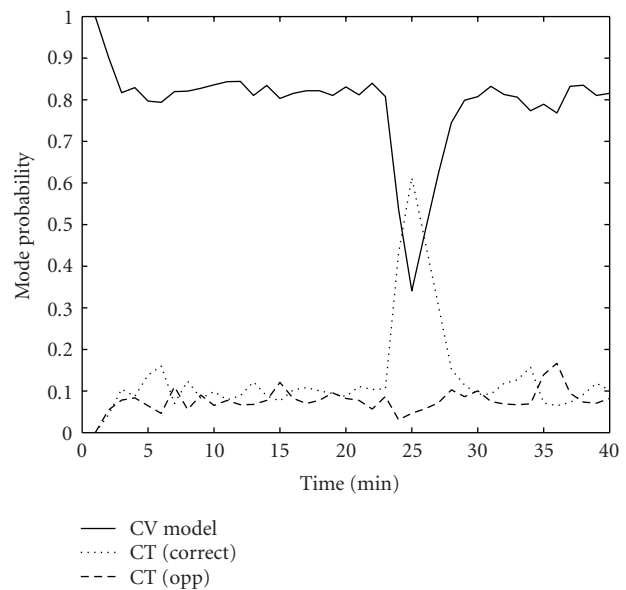


(b)

FIGURE 3: IMM-EKF tracker results. (a) Track estimates and 95% confidence ellipses. (b) Mode probabilities.



(a)



(b)

FIGURE 4: MMPF tracker results. (a) Track estimates and 95% confidence ellipses. (b) Mode probabilities.

This is clear from the mode probability curves shown in Figure 3b, where we note that though there is a slight bump in the mode probability for the correct manoeuvre model, the algorithm is unable to establish the occurrence of the manoeuvre. The overall result is a track that is showing a tendency to diverge from the true track.

For the same set of measurements, the MMPF shows excellent performance as can be seen from Figure 4. Here we note that the 95% confidence ellipse of the PF encapsulates the true target position at all times. Notice that the size of the covariance matrix shortly after the target manoeuvre is small compared to other times. The reason for this is that the

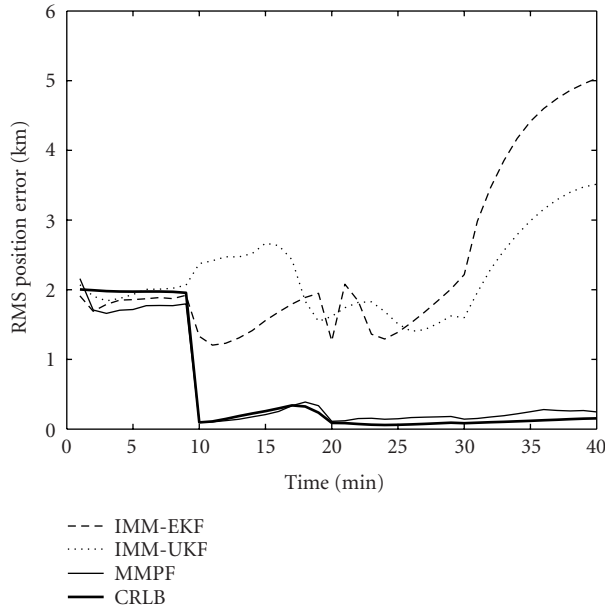


FIGURE 5: RMS position error versus time for a multisensor case.

target observability is best at that instant compared to other times. For the given scenario, both the ownship manoeuvre and the target manoeuvre have resulted in a geometry that is very observable at that instant. After the target manoeuvre, the relative position of the target increases and this leads to a slight decrease in observability and hence slight enlargement of the covariance matrix. The mode probability curves for the MMPF shows that unlike the results of IMM-EKF, the MMPF mode probabilities indicate a higher probability of occurrence of a manoeuvre. The overall result is a much better tracker performance for the same set of measurements.

5.2. Multisensor case

Here we consider the scenario identical to the one considered in Section 5.1, except that an additional static sensor, located at (5 km, -2 km), provides bearing measurements to the ownship at regular time intervals. These measurements, with accuracy $\sigma_{\theta} = 2^{\circ}$, arrive at only 3 time epochs, namely, at $k = 10, 20$, and 30 . Figure 5 shows a comparison of IMM-EKF, IMM-UKF, and MMPF for this case. It is seen that the MMPF exhibits excellent performance, with RMS error results very close to the CRLB. The detailed comparison given in Table 2 shows that MMPF achieves a final RMS error accuracy that is within 8% of the final range. By comparing with the corresponding results for the single-sensor case, we note that the final RMS error is reduced by a factor of 2.5. Interestingly, the IMM-EKF and IMM-UKF performance is very poor and is worse than their corresponding performance when no additional measurement is received. Though this may seem counterintuitive, it can be explained as follows. For the given geometry, at the time of the first arrival of the bearing measurement from the second sensor, it is possible that due to nonlinearities and low observability in the time interval 0–10 minutes, the track estimates and filter calcu-

lated covariance of the IMM-based filters are in error, leading to a large innovation for the second sensor measurement. The inaccurate covariance estimate results in an incorrect filter gain computation for the second sensor measurement. In the update equations of these filters, the large innovation gets weighted by the computed gain which does not properly reflect the contribution of the new measurement. The consequence of this is filter divergence. It turns out that for the ownship measurements-only case, even if the track and covariance estimates are in error, the errors introduced in the filter gain computation are not as severe as in the multisensor case. Furthermore, as the uncertainty is mainly along the line of bearing, the innovation for this case is not likely to be very large. Thus the severity of track and covariance error for this particular scenario is worse for the multisensor case than for the single-sensor case. Similar results have been observed in the context of an air surveillance scenario [12].

5.3. Tracking with hard constraints

In this section, we present the results for the case of bearings-only tracking with hard constraints. The scenario and parameters used for this case are identical to the ones considered in Section 5.1. This time, however, in addition to the available bearing measurements, we also impose some hard constraints on target speed. Specifically, assume that we have prior knowledge that the target speed is in the range $3.5 \leq s \leq 4.5$ knots. This type of nonstandard information is difficult to incorporate into the standard EKF-based algorithms (such as the IMM-EKF), and so in the comparison below, the IMM-EKF will not utilise the hard constraints. However, the PF-based algorithms, and, in particular, the MMPF and AUX-MMPE, can easily incorporate such nonstandard information according to the technique described in Section 4.7.

Figure 6 shows the RMS error in estimated position for the MMPF that incorporates prior knowledge of speed constraint (referred to as MMPF-C). The figure also shows the performance curves of the IMM-EKF and the standard MMPF that do not utilise knowledge of hard constraints. A detailed numerical comparison is given in Table 3. It can be seen that the MMPF-C achieves 83% improvement in RTAMS over the IMM-EKF. Also, observe that by incorporating the hard constraints, the MMPF-C achieves a 50% reduction in RTAMS error over the standard MMPF that does not utilise hard constraints (emphasising the significance of this nonstandard information). Incorporating such nonstandard information results in highly non-Gaussian posterior pdfs which the PF is effectively able to characterise.

6. CONCLUSIONS

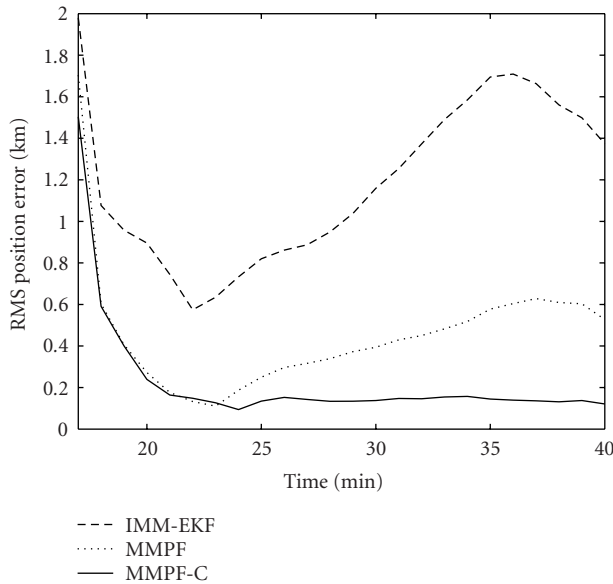
This paper presented a comparative study of PF-based trackers against conventional IMM-based routines for the problem of bearings-only tracking of a manoeuvring target. Three separate cases have been analysed: single-sensor case; multisensor case, and tracking with speed constraints. The results overwhelmingly confirm the superior performance of PF-based algorithms against the conventional IMM-based

TABLE 2: Performance comparison for the multisensor case.

Algorithm/ CRLB	RMS error (final)			RTAMS (km)	Improvement (%)	Divergent tracks
	(km)	(%)	η			
IMM-EKF	5.03	173	3	3.16	0	17
IMM-UKF	3.51	121	4	2.32	27	7
MMPF	0.25	8	63	0.22	93	1
CRLB	0.15	5	100	0.13	96	—

TABLE 3: Performance comparison for tracking with hard constraints.

Algorithm	RMS error (final)		RTAMS (km)	Improvement (%)	Divergent tracks
	(km)	(%)			
IMM-EKF	1.37	47	1.21	0	0
MMPF	0.53	18	0.44	64	0
MMPF-C	0.12	4	0.20	83	0

FIGURE 6: RMS position error versus time for the case of tracking with speed constraint $3.5 \leq s \leq 4.5$ knots.

schemes. The key strength of the PE, demonstrated in this application, is its flexibility to handle nonstandard information along with the ability to deal with nonlinear and non-Gaussian models.

APPENDIX

JACOBIANS OF THE MANOEUVRE DYNAMICS

The Jacobians $\tilde{\mathbf{F}}_k^{(r_{k+1}^*)}$, $r_{k+1}^* \in \{2, 3\}$, of the manoeuvre dynamics can be computed by taking the gradients of the respective transitions. Let $f_i^{(j)}(\cdot)$ denote the i th element of the dynamics model function $\mathbf{f}^{(j)}(\cdot)$ and let $(\dot{x}_k^t, \dot{y}_k^t)$ denote the target velocity vector. Then, by taking the gradients of $\mathbf{f}^{(j)}(\cdot)$ for

$j = 2, 3$, the required Jacobians can be computed to give

$$\tilde{\mathbf{F}}_k^{(j)} = \begin{bmatrix} 1 & 0 & \frac{\partial f_1^{(j)}}{\partial \dot{x}_k} & \frac{\partial f_1^{(j)}}{\partial \dot{y}_k} \\ 0 & 1 & \frac{\partial f_2^{(j)}}{\partial \dot{x}_k} & \frac{\partial f_2^{(j)}}{\partial \dot{y}_k} \\ 0 & 0 & \frac{\partial f_3^{(j)}}{\partial \dot{x}_k} & \frac{\partial f_3^{(j)}}{\partial \dot{y}_k} \\ 0 & 0 & \frac{\partial f_4^{(j)}}{\partial \dot{x}_k} & \frac{\partial f_4^{(j)}}{\partial \dot{y}_k} \end{bmatrix}, \quad j = 2, 3, \quad (\text{A.1})$$

where

$$\begin{aligned} \frac{\partial f_1^{(j)}}{\partial \dot{x}_k} &= \frac{\sin(\Omega_k^{(j)} T)}{\Omega_k^{(j)}} + g_1^{(j)}(k) \frac{\partial \Omega_k^{(j)}}{\partial \dot{x}_k}, \\ \frac{\partial f_1^{(j)}}{\partial \dot{y}_k} &= \frac{-(1 - \cos(\Omega_k^{(j)} T))}{\Omega_k^{(j)}} + g_1^{(j)}(k) \frac{\partial \Omega_k^{(j)}}{\partial \dot{y}_k}, \\ \frac{\partial f_2^{(j)}}{\partial \dot{x}_k} &= \frac{(1 - \cos(\Omega_k^{(j)} T))}{\Omega_k^{(j)}} + g_2^{(j)}(k) \frac{\partial \Omega_k^{(j)}}{\partial \dot{x}_k}, \\ \frac{\partial f_2^{(j)}}{\partial \dot{y}_k} &= \frac{\sin(\Omega_k^{(j)} T)}{\Omega_k^{(j)}} + g_2^{(j)}(k) \frac{\partial \Omega_k^{(j)}}{\partial \dot{y}_k}, \\ \frac{\partial f_3^{(j)}}{\partial \dot{x}_k} &= \cos(\Omega_k^{(j)} T) + g_3^{(j)}(k) \frac{\partial \Omega_k^{(j)}}{\partial \dot{x}_k}, \\ \frac{\partial f_3^{(j)}}{\partial \dot{y}_k} &= -\sin(\Omega_k^{(j)} T) + g_3^{(j)}(k) \frac{\partial \Omega_k^{(j)}}{\partial \dot{y}_k}, \\ \frac{\partial f_4^{(j)}}{\partial \dot{x}_k} &= \sin(\Omega_k^{(j)} T) + g_4^{(j)}(k) \frac{\partial \Omega_k^{(j)}}{\partial \dot{x}_k}, \\ \frac{\partial f_4^{(j)}}{\partial \dot{y}_k} &= \cos(\Omega_k^{(j)} T) + g_4^{(j)}(k) \frac{\partial \Omega_k^{(j)}}{\partial \dot{y}_k}, \end{aligned} \quad (\text{A.2})$$

with

$$\begin{aligned}
 g_1^{(j)}(k) &= \frac{T \cos(\Omega_k^{(j)} T) \dot{x}_k^t}{\Omega_k^{(j)}} - \frac{\sin(\Omega_k^{(j)} T) \dot{x}_k^t}{(\Omega_k^{(j)})^2} \\
 &\quad - \frac{T \sin(\Omega_k^{(j)} T) \dot{y}_k^t}{\Omega_k^{(j)}} \\
 &\quad + \frac{(-1 + \cos(\Omega_k^{(j)} T)) \dot{y}_k^t}{(\Omega_k^{(j)})^2}, \\
 g_2^{(j)}(k) &= \frac{T \sin(\Omega_k^{(j)} T) \dot{x}_k^t}{\Omega_k^{(j)}} - \frac{(1 - \cos(\Omega_k^{(j)} T)) \dot{x}_k^t}{(\Omega_k^{(j)})^2} \\
 &\quad + \frac{T \cos(\Omega_k^{(j)} T) \dot{y}_k^t}{\Omega_k^{(j)}} - \frac{\sin(\Omega_k^{(j)} T) \dot{y}_k^t}{(\Omega_k^{(j)})^2}, \\
 g_3^{(j)}(k) &= -\sin(\Omega_k^{(j)} T) T \dot{x}_k^t - \cos(\Omega_k^{(j)} T) T \dot{y}_k^t, \\
 g_4^{(j)}(k) &= \cos(\Omega_k^{(j)} T) T \dot{x}_k^t - \sin(\Omega_k^{(j)} T) T \dot{y}_k^t, \\
 \frac{\partial \Omega_k^{(j)}}{\partial \dot{x}_k} &= \frac{(-1)^{j+1} a_m \dot{x}_k^t}{\left[(\dot{x}_k^t)^2 + (\dot{y}_k^t)^2 \right]^{3/2}}, \\
 \frac{\partial \Omega_k^{(j)}}{\partial \dot{y}_k} &= \frac{(-1)^{j+1} a_m \dot{y}_k^t}{\left[(\dot{x}_k^t)^2 + (\dot{y}_k^t)^2 \right]^{3/2}}
 \end{aligned} \tag{A.3}$$

for $j = 2, 3$.

REFERENCES

- [1] S. Blackman and R. Popoli, *Design and Analysis of Modern Tracking Systems*, Artech House, Norwood, Mass, USA, 1999.
- [2] J. C. Hassab, *Underwater Signal and Data Processing*, CRC Press, Boca Raton, Fla, USA, 1989.
- [3] S. Nardone, A. Lindgren, and K. Gong, "Fundamental properties and performance of conventional bearings-only target motion analysis," *IEEE Trans. Automatic Control*, vol. 29, no. 9, pp. 775–787, 1984.
- [4] E. Fogel and M. Gavish, "Nth-order dynamics target observability from angle measurements," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 24, no. 3, pp. 305–308, 1988.
- [5] T. L. Song, "Observability of target tracking with bearings-only measurements," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 32, no. 4, pp. 1468–1472, 1996.
- [6] S. S. Blackman and S. H. Roszkowski, "Application of IMM filtering to passive ranging," in *Proc. SPIE Signal and Data Processing of Small Targets*, vol. 3809 of *Proceedings of SPIE*, pp. 270–281, Denver, Colo, USA, July 1999.
- [7] T. Kirubarajan, Y. Bar-Shalom, and D. Lerro, "Bearings-only tracking of maneuvering targets using a batch-recursive estimator," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 37, no. 3, pp. 770–780, 2001.
- [8] J.-P. Le Cadre and O. Tremois, "Bearings-only tracking for maneuvering sources," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 34, no. 1, pp. 179–193, 1998.
- [9] N. J. Gordon, D. J. Salmond, and A. F. M. Smith, "Novel approach to nonlinear/non-Gaussian Bayesian state estimation," *IEE Proceedings Part F: Radar and Signal Processing*, vol. 140, no. 2, pp. 107–113, 1993.

- [10] A. Doucet, N. de Freitas, and N. Gordon, Eds., *Sequential Monte Carlo Methods in Practice*, Springer, New York, NY, USA, 2001.
- [11] M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A tutorial on particle filters for online nonlinear/ non-Gaussian Bayesian tracking," *IEEE Trans. Signal Processing*, vol. 50, no. 2, pp. 174–188, 2002.
- [12] B. Ristic and M. S. Arulampalam, "Tracking a manoeuvring target using angle-only measurements: algorithms and performance," *Signal Processing*, vol. 83, no. 6, pp. 1223–1238, 2003.
- [13] S. McGinnity and G. W. Irwin, "Multiple model bootstrap filter for maneuvering target tracking," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 36, no. 3, pp. 1006–1012, 2000.
- [14] R. Karlsson and N. Bergman, "Auxiliary particle filters for tracking a maneuvering target," in *Proc. 39th IEEE Conference on Decision and Control*, vol. 4, pp. 3891–3895, Sydney, Australia, December 2000.
- [15] S. Julier, J. Uhlmann, and H. F. Durrant-Whyte, "A new method for the nonlinear transformation of means and covariances in filters and estimators," *IEEE Trans. Automatic Control*, vol. 45, no. 3, pp. 477–482, 2000.
- [16] Y. Bar-Shalom and X. R. Li, *Estimation and Tracking: Principles, Techniques and Software*, Artech House, Norwood, Mass, USA, 1993.
- [17] P. Tichavsky, C. H. Muravchik, and A. Nehorai, "Posterior Cramér-Rao bounds for discrete-time nonlinear filtering," *IEEE Trans. Signal Processing*, vol. 46, no. 5, pp. 1386–1396, 1998.
- [18] A. Doucet, N. J. Gordon, and V. Krishnamurthy, "Particle filters for state estimation of jump Markov linear systems," *IEEE Trans. Signal Processing*, vol. 49, no. 3, pp. 613–624, 2001.
- [19] N. Bergman, A. Doucet, and N. Gordon, "Optimal estimation and Cramér-Rao bounds for partial non-Gaussian state space models," *Annals of the Institute of Statistical Mathematics*, vol. 53, no. 1, pp. 97–112, 2001.
- [20] C. P. Robert and G. Casella, *Monte Carlo Statistical Methods*, Springer-Verlag, New York, NY, USA, 1999.
- [21] J. Carpenter, P. Clifford, and P. Fearnhead, "Improved particle filter for nonlinear problems," *IEE Proceedings—Radar, Sonar and Navigation*, vol. 146, no. 1, pp. 2–7, 1999.

M. Sanjeev Arulampalam received the B.S. degree in mathematics and the B.E. degree with first-class honours in electrical and electronic engineering from the University of Adelaide in 1991 and 1992, respectively. In 1993, he won a Telstra Research Labs Postgraduate Fellowship award to work toward a Ph.D. degree in electrical and electronic engineering at the University of Melbourne, which he completed in 1997. Since 1998, Dr. Arulampalam has been with the Defence Science and Technology Organisation (DSTO), Australia, working on many aspects of target tracking with a particular emphasis on nonlinear/non-Gaussian tracking problems. In March 2000, he won the Anglo-Australian Postdoctoral Research Fellowship, awarded by the Royal Academy of Engineering, London. This postdoctoral research was carried out in the UK, both at the Defence Evaluation and Research Agency (DERA) and at Cambridge University, where he worked on particle filters for nonlinear tracking problems. Currently, he is a Senior Research Scientist in the Submarine Combat Systems Group of Maritime Operations



Division, DSTO, Australia. His research interests include estimation theory, target tracking, and sequential Monte Carlo methods. Dr. Arulampalam coauthored a recent book, *Beyond the Kalman Filter: Particle Filters for Tracking Applications*, Artech House, 2004.

B. Ristic received all his degrees in electrical engineering: a Ph.D. degree from Queensland University of Technology (QUT), Australia, in 1995, an M.S. degree from Belgrade University, Serbia, in 1991, and a B.E. degree from the University of Novi Sad, Serbia, in 1984. He began his career in 1984 at the Vinča Institute, Serbia. From 1989 to 1994, he was with the University of Queensland, Brisbane, and QUT, Australia, doing research related to the automatic placement and routing of integrated circuits and the design and analysis of time-frequency and time-scale distributions. In 1995, he was with GEC Marconi Systems in Sydney, Australia, developing a concept demonstrator for noise cancellation in towed arrays. Since 1996, he has been with the Defense Science and Technology Organisation (DSTO), Australia, where he has been involved in the design and analysis of target tracking and data fusion systems. During 2003/2004, he has been on a study leave with Université Libre de Bruxelles (ULB), Belgium, doing research on reasoning under uncertainty. Dr. Ristic coauthored a recent book, *Beyond the Kalman Filter: Particle Filters for Tracking Applications*, Artech House, 2004. During his career, he has published more than 80 technical papers.



N. Gordon obtained a B.S. in mathematics and physics from Nottingham University in 1988 and a Ph.D. degree in statistics from Imperial College, University of London in 1993. He was with the Defence Evaluation and Research Agency (DERA) and QinetiQ in the UK from 1988 to 2002. In 2002, he joined the Tracking and Sensor Fusion Research Group at Defence Science and Technology Organisation (DSTO) in Australia. Neil has written approximately 65 articles in peer-reviewed journals and international conferences on tracking and other dynamic state estimation problems. He is the coauthor/coeditor of two books on particle filtering.



T. Mansell received a B.S. degree with first-class honours in physics and electronics from Deakin University in 1989. Following graduation, he joined the Defence Science and Technology Organisation (DSTO) in Melbourne as a Cadet Research Scientist, and in 1990 began a Ph.D. in artificial intelligence with The University of Melbourne. On completion of his Ph.D. in 1994, Dr. Mansell began working on the application of information fusion techniques to naval problems (including mine warfare and combat systems). In 1996, Dr. Mansell undertook a 15-month posting with Canada's Defence Research Establishment Atlantic in the sonar information management group looking at tactical decision-making and naval combat systems. On return to Australia, Dr. Mansell relocated to DSTO, Edinburgh, South Australia, to lead the submarine combat systems task. In 2000, he became Head of the Submarine Combat Systems Group (SMCS) within Maritime Operations Division. In 2003, Dr. Mansell was appointed Head of Maritime Combat Systems and the DSTO lead for



the Air Warfare Destroyer, Combat System Studies. Dr. Mansell is currently the Counsellor for Defence Science in the Australian High Commission, London. Dr. Mansell's main research interests are in combat system architectures, open system architectures, information fusion, artificial intelligence, human-machine interaction, and tactical decision support systems.

Time-Varying Noise Estimation for Speech Enhancement and Recognition Using Sequential Monte Carlo Method

Kaisheng Yao

Institute for Neural Computation, University of California, San Diego, 9500 Gilman Drive, La Jolla, CA 92093-0523, USA
Email: kyao@ucsd.edu

Te-Won Lee

Institute for Neural Computation, University of California, San Diego, 9500 Gilman Drive, La Jolla, CA 92093-0523, USA
Email: tewon@ucsd.edu

Received 4 May 2003; Revised 9 April 2004

We present a method for sequentially estimating time-varying noise parameters. Noise parameters are sequences of time-varying mean vectors representing the noise power in the log-spectral domain. The proposed sequential Monte Carlo method generates a set of particles in compliance with the prior distribution given by clean speech models. The noise parameters in this model evolve according to random walk functions and the model uses extended Kalman filters to update the weight of each particle as a function of observed noisy speech signals, speech model parameters, and the evolved noise parameters in each particle. Finally, the updated noise parameter is obtained by means of minimum mean square error (MMSE) estimation on these particles. For efficient computations, the residual resampling and Metropolis-Hastings smoothing are used. The proposed sequential estimation method is applied to noisy speech recognition and speech enhancement under strongly time-varying noise conditions. In both scenarios, this method outperforms some alternative methods.

Keywords and phrases: sequential Monte Carlo method, speech enhancement, speech recognition, Kalman filter, robust speech recognition.

1. INTRODUCTION

A speech processing system may be required to work in conditions where the speech signals are distorted due to background noise. Those distortions can drastically drop the performance of automatic speech recognition (ASR) systems, which usually perform well in quiet environments. Similarly, speech-coding systems spend much of their coding capacity encoding additional noise information.

There have been great interests in developing algorithms that achieve robustness to those distortions. In general, the proposed methods can be grouped into two approaches. One approach is based on front-end processing of speech signals, for example, speech enhancement. Speech enhancement can be done either in time-domain, for example, in [1, 2], or more widely used, in spectral domain [3, 4, 5, 6, 7]. The objective of speech enhancement is to increase signal-to-noise ratio (SNR) of the processed speech with respect to the observed noisy speech signal.

The second approach is based on statistical models of speech and/or noise. For example, parallel model combination (PMC) [8] adapts speech mean vectors according to the input noise power. In [9], code-dependent cepstral normalization (CDCN) modifies speech signals based on probabilities from speech models. Since methods in this model-based approach are devised in a principled way, for example, maximum likelihood estimation [9], they usually have better performances than methods in the first approach, particularly in applications such as noisy speech recognition [10].

However, a main shortcoming in some of the methods described above lies in their assumption that the background noise is stationary (noise statistics do not change in a given utterance). Based on this assumption, noise is often estimated from segmented noise-alone slices, for example, by voice-activity detection (VAD) [7]. Such an assumption may not hold in many real applications because the estimated noise may not be pertinent to noise in speech intervals in nonstationary environments.

Recently, methods have been proposed for speech enhancement in nonstationary noise. For example, in [11], a method based on sequential Monte Carlo method is applied to estimate time-varying autocorrelation coefficients of speech models for speech enhancement. This algorithm is more advanced in its assumption that autocorrelation coefficients of speech models are time varying. In fact, sequential Monte Carlo method is also applied to estimate noise parameters for robust speech recognition in nonstationary noise [12] through a nonlinear model [8], which was recently found to be effective for speech enhancement [13] as well.

The purpose of this paper is to present a method based on sequential Monte Carlo for estimation of noise parameter (time-varying mean vector of a noise model) with its application to speech enhancement and recognition. The method is based on a nonlinear function that models noise effects on speech [8, 12, 13]. Sequential Monte Carlo method generates particles of parameters (including speech and noise parameters) from a prior speech model that has been trained from a clean speech database. These particles approximate posterior distribution of speech and noise parameter sequences given the observed noisy speech sequence. Minimum mean square error (MMSE) estimation of the noise parameter is obtained from these particles. Once the noise parameter has been estimated, it is used in subtraction-type speech enhancement methods, for example, Wiener filter and perceptual filter,¹ and adaptation of speech mean vectors for speech recognition.

The remainder of the paper is organized as follows. The model specification and estimation objectives for the noise parameters are stated in Section 2. In Section 3, the sequential Monte Carlo method is developed to solve the noise parameter estimation problem. Section 4.3 demonstrates application of this method to speech recognition by modifying speech model parameters. Application to speech enhancement is shown in Section 4.4. Discussions and conclusions are presented in Section 5.

Notation

Sets are denoted as $\{\cdot, \cdot\}$. Vectors and sequences of vectors are denoted by uppercased letters. Time index is in the parenthesis of vectors. For example, a sequence $Y(1 : T) = (Y(1) Y(2) \cdots Y(T))$ consists of vector $Y(t)$ at time t , where its i th element is $y_i(t)$. The distribution of the vector $Y(t)$ is $p(Y(t))$. Superscript T denotes transpose.

The symbol X (or x) is exclusively used for original speech and Y (or y) is used for noisy speech in testing environments. N (or n) is used to denote noise.

By default, observation (or feature) vectors are in log-spectral domain. Superscripts *lin*, *l*, *c* denote linear spectral domain, log-spectral domain, and cepstral domain. The symbol $*$ denotes convolution.

2. PROBLEM DEFINITION

2.1. Model definitions

Consider a clean speech signal $x(t)$ at time t that is corrupted by additive background noise $n(t)$.² In time domain, the received speech signal $y(t)$ can be written as

$$y(t) = x(t) + n(t). \quad (1)$$

Assume that the speech signal $x(t)$ and noise $n(t)$ are uncorrelated. Hence, the power spectrum of the input noisy signal is the summation of the power spectra of clean speech signal and those of the noise. The output at filter bank j can be described by $y_j^{\text{lin}}(t) = \sum_m b(m) |\sum_{l=0}^{L-1} v(l) y(t-l) e^{-j2\pi lm/L}|^2$, summing the power spectra of the windowed signal $v(t) * y(t)$ with length L at each frequency m with binning weight $b(m)$. $v(t)$ is a window function (usually a Hamming window) and $b(m)$ is a triangle window.³ Similarly, we denote the filter bank output for clean speech signal $x(t)$ and noise $n(t)$ as $x_j^{\text{lin}}(t)$ and $n_j^{\text{lin}}(t)$ for j th filter bank, respectively. They are related as

$$y_j^{\text{lin}}(t) = x_j^{\text{lin}}(t) + n_j^{\text{lin}}(t), \quad (2)$$

where j is from 1 to J , and J is the number of filter banks.

The filter bank output exhibits a large variance. In order to achieve an accurate statistical model, in some applications, for example, speech recognition, logarithm compression of $y_j^{\text{lin}}(t)$ is used instead. The corresponding compressed power spectrum is called log-spectral power, which has the following relationship (derived in Appendix A) with noisy signal, clean speech signal, and noise:

$$y_j^l(t) = x_j^l(t) + \log(1 + \exp(n_j^l(t) - x_j^l(t))). \quad (3)$$

The function is plotted in Figure 1. We observed that this function is convex and continuous. For noise log-spectral power $n_j^l(t)$ that is much smaller than clean speech log-spectral power $x_j^l(t)$, the function outputs $x_j^l(t)$. This shows that the function is not “sensitive” to noise log-spectral power that is much smaller than clean speech log-spectral power.⁴

We consider the vector for clean speech log-spectral power $X^l(t) = (x_1^l(t), \dots, x_J^l(t))^T$. Suppose that the statistics of the log-spectral power sequence $X^l(1 : T)$ can be modeled by a hidden Markov model (HMM) with output density at each state s_t ($1 \leq s_t \leq S$) represented by mixtures of Gaussian $\sum_{k_t=1}^M \pi_{s_t k_t} \mathcal{N}(X^l(t); \mu_{s_t k_t}^l, \Sigma_{s_t k_t}^l)$, where M denotes the number

²Channel distortion and reverberation are not considered in this paper. In this paper, $x(t)$ can be considered as a speech signal received by a close-talking microphone, and $n(t)$ is the background noise picked up by the microphone.

³In Mel-scaled filter bank analysis [16], $b(m)$ is a triangle window centered in the Mel scale.

⁴We will discuss later in Sections 3.5 and 4.2 that such property may result in larger-than-necessary estimation of the noise log-spectral power.

¹A model for frequency masking [14, 15] is applied.

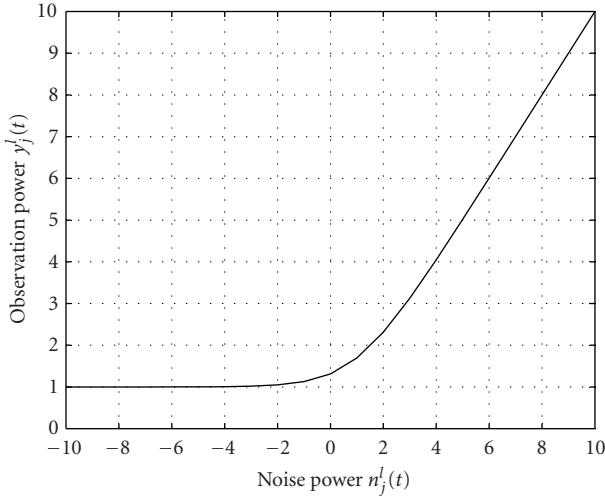


FIGURE 1: Plot of function $y_j^l(t) = x_j^l(t) + \log(1 + \exp(n_j^l(t) - x_j^l(t)))$. $x_j^l(t) = 1.0$; $n_j^l(t)$ ranges from -10.0 to 10.0 .

of Gaussian densities in each state. To model the statistics of noise log-spectral power $N^l(1 : T)$, we use a single Gaussian density with a time-varying mean vector $\mu_n^l(t)$ and a constant diagonal variance matrix V_n^l .

With the above-defined statistical models, we may plot the dependence among their parameters and observation sequence $Y^l(1 : t)$ by a graphical model [17] in Figure 2. In this figure, the rectangular boxes correspond to discrete state/mixture indexes, and the round circles correspond to continuous-valued vectors. Shaded circles denote observed noisy speech log-spectral power.

The state $s_t \in \{1, \dots, S\}$ gives the current state index at frame t . State sequence is a Markovian sequence with state transition probability $p(s_t | s_{t-1}) = a_{s_{t-1}s_t}$. At state s_t , an index $k_t \in \{1, \dots, M\}$ assigns a Gaussian density $\mathcal{N}(\cdot; \mu_{s_t k_t}^l, \Sigma_{s_t k_t}^l)$ with prior probability $p(k_t | s_t) = \pi_{s_t k_t}$. Speech parameter $\mu_{s_t k_t}^l(t)$ is thus distributed in Gaussian given s_t and k_t ; that is,

$$s_t \sim p(s_t | s_{t-1}) = a_{s_{t-1}s_t}, \quad (4)$$

$$k_t \sim p(k_t | s_t) = \pi_{s_t k_t}, \quad (5)$$

$$\mu_{s_t k_t}^l(t) \sim \mathcal{N}(\cdot; \mu_{s_t k_t}^l, \Sigma_{s_t k_t}^l). \quad (6)$$

Assuming that the variances of $X^l(t)$ and $N^l(t)$ are very small (as done in [8]) for each filter bank j , given s_t and k_t , we may relate the observed signal $Y^l(t)$ to speech mean vector $\mu_{s_t k_t}^l(t)$ and time-varying noise mean vector $\mu_n^l(t)$ with the function

$$Y^l(t) = \mu_{s_t k_t}^l(t) + \log(1 + \exp(\mu_n^l(t) - \mu_{s_t k_t}^l(t))) + w_{s_t k_t}(t), \quad (7)$$

where $w_{s_t k_t}(t)$ is distributed in $\mathcal{N}(\cdot; 0, \Sigma_{s_t k_t}^l)$, representing the possible modeling error and measurement noise in the above equation.

Furthermore, to model time-varying noise statistics, we assume that the noise parameter $\mu_n^l(t)$ follows a random walk function; that is,

$$\begin{aligned} \mu_n^l(t) &\sim p(\mu_n^l(t) | \mu_n^l(t-1)) \\ &= \mathcal{N}(\mu_n^l(t); \mu_n^l(t-1), V_n^l). \end{aligned} \quad (8)$$

We collectively denote these parameters $\{\mu_{s_t k_t}^l(t), s_t, k_t, \mu_n^l(t); \mu_{s_t k_t}^l(t) \in \mathbb{R}^J, 1 \leq s_t \leq S, 1 \leq k_t \leq M, \mu_n^l(t) \in \mathbb{R}^J\}$ as $\theta(t)$. It is clearly seen from (4)–(8) that they have the following prior distribution and likelihood at each time t :

$$\begin{aligned} p(\theta(t) | \theta(t-1)) \\ = a_{s_{t-1}s_t} \pi_{s_t k_t} \end{aligned} \quad (9)$$

$$\times \mathcal{N}(\mu_{s_t k_t}^l(t); \mu_{s_t k_t}^l, \Sigma_{s_t k_t}^l) \mathcal{N}(\mu_n^l(t); \mu_n^l(t-1), V_n^l),$$

$$\begin{aligned} p(Y^l(t) | \theta(t)) \\ = \mathcal{N}(Y^l(t); \mu_{s_t k_t}^l(t) \\ + \log(1 + \exp(\mu_n^l(t) - \mu_{s_t k_t}^l(t))), \Sigma_{s_t k_t}^l). \end{aligned} \quad (10)$$

Remark 1. In comparison with the traditional HMM, the new model shown in Figure 2 may provide more robustness to contaminating noise, because it includes explicit modeling of the time-varying noise parameters. However, probabilistic inference in the new model can no longer be done by the efficient Viterbi algorithm [18].

2.2. Estimation objective

The objective of this method is to estimate, up to time t , a sequence of noise parameters $\mu_n^l(1 : t)$ given the observed noisy speech log-spectral sequence $Y^l(1 : t)$ and the above defined graphical model, in which speech models are trained from clean speech signals. Formally, $\mu_n^l(1 : t)$ is calculated by the MMSE estimation

$$\hat{\mu}_n^l(1 : t) = \int_{\mu_n^l(1:t)} \mu_n^l(1 : t) p(\mu_n^l(1 : t) | Y^l(1 : t)) d\mu_n^l(1 : t), \quad (11)$$

where $p(\mu_n^l(1 : t) | Y^l(1 : t))$ is the posterior distribution of $\mu_n^l(1 : t)$ given $Y^l(1 : t)$.

Based on the graphical model shown in Figure 2, Bayesian estimation of the time-varying noise parameter $\mu_n^l(1 : t)$ involves construction of a likelihood function of observation sequence $Y^l(1 : t)$ given parameter sequence $\Theta(1 : t) = (\theta(1), \dots, \theta(t))$ and prior probability $p(\Theta(1 : t))$ for $t = 1, \dots, T$. The posterior distribution of $\Theta(1 : t)$ given observation sequence $Y^l(1 : t)$ is

$$p(\Theta(1 : t) | Y^l(1 : t)) \propto p(Y^l(1 : t) | \Theta(1 : t)) p(\Theta(1 : t)). \quad (12)$$

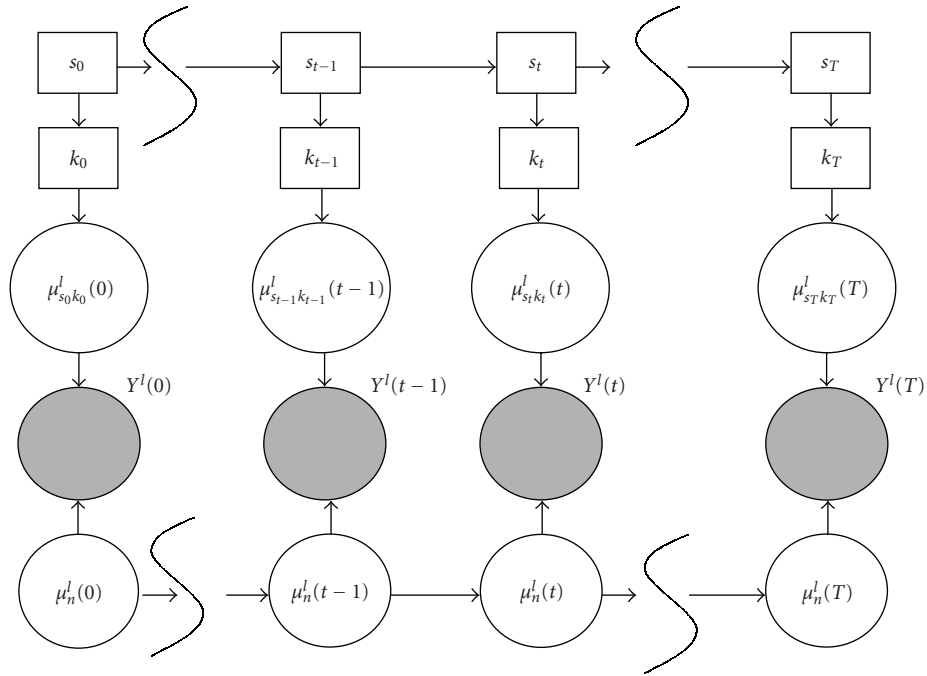


FIGURE 2: The graphical model representation of the dependence of the speech and noise model parameters. s_t and k_t denote the state and Gaussian mixture at frame t in speech model. $\mu_{s_t k_t}^l(t)$ and $\mu_n^l(t)$ denote the speech and noise parameters. $Y^l(t)$ is the observed noisy speech signal at frame t .

Due to the Markovian property shown in (9) and (10), the above posterior distribution can be written as

$$p(\Theta(1:t)|Y^l(1:t)) \propto \prod_{\tau=2}^t p(Y^l(\tau)|\theta(\tau))p(\theta(\tau)|\theta(\tau-1))p(Y^l(1)|\theta(1))p(\theta(1)). \tag{13}$$

Based on this posterior distribution, MMSE estimation in (11) can be achieved by

$$\hat{\mu}_n^l(1:t) = \int_{\mu_{1:n}^l(1:t)} \mu_{1:n}^l(1:t) \times \sum_{s_{1:t}, k_{1:t}} \int_{\mu_{s_{1:t}, k_{1:t}}^l(1:t)} p(\Theta(1:t)|Y^l(1:t)) d\mu_{s_{1:t}, k_{1:t}}^l(1:t) d\mu_n^l(1:t). \tag{14}$$

Note that there are difficulties in evaluating the MMSE estimation. The first relates to the nonlinear function in (10), and the second arises from the unseen state sequence $s_{1:t}$ and mixture sequence $k_{1:t}$. These unseen sequences, together with nodes $\{\mu_{s_t, k_t}^l(t)\}$, $\{Y^l(t)\}$, and $\{\mu_n^l(t)\}$, form loops in the graphical model. These loops in Figure 2 make exact inferences on posterior probabilities of unseen sequences $s_{1:t}$ and $k_{1:t}$, computationally intractable. In the following section, we devise a sequential Monte Carlo method to tackle these problems.

3. SEQUENTIAL MONTE CARLO METHOD FOR NOISE PARAMETER ESTIMATION

This section presents a sequential Monte Carlo method for estimating noise parameters from observed noisy signals and pretrained clean speech models. This method applies sequential Bayesian importance sampling (BIS) in order to generate particles of speech and noise parameters from a proposal distribution. These particles are selected according to their weights calculated with a function of their likelihood. It should be noted that the application here is one particular case of a more general sequential BIS method [19, 20].

3.1. Importance sampling

Suppose that there are N particles $\{\Theta^{(i)}(1:t); i = 1, \dots, N\}$. Each particle is denoted as

$$\Theta^{(i)}(1:t) = \{s_{1:t}^{(i)}, k_{1:t}^{(i)}, \mu_{s_{1:t}, k_{1:t}}^{l(i)}(1:t), \mu_n^{l(i)}(1:t)\}. \tag{15}$$

These particles are generated according to $p(\Theta(1:t)|Y^l(1:t))$. Then, these particles form an empirical distribution of $\Theta(1:t)$, given by

$$\bar{p}_N(\Theta(1:t)|Y^l(1:t)) = \frac{1}{N} \sum_{i=1}^N \delta_{\Theta^{(i)}(1:t)}(d\Theta(1:t)), \tag{16}$$

where $\delta_x(\cdot)$ is the Dirac delta measure concentrated on x .

Using this distribution, an estimate of the parameters of interests $\bar{f}_{\Theta}(1:t)$ can be obtained by

$$\begin{aligned}\bar{f}_{\Theta}(1:t) &= \int f_{\Theta}(1:t) \bar{p}_N(\Theta(1:t)|Y^l(1:t)) d\Theta(1:t) \\ &= \frac{1}{N} \sum_{i=1}^N f_{\Theta}^{(i)}(1:t),\end{aligned}\quad (17)$$

where, for example, function $f_{\Theta}(1:t)$ is $\Theta(1:t)$ and $f_{\Theta}^{(i)}(1:t) = \Theta^{(i)}(1:t)$ if $\bar{f}_{\Theta}(1:t)$ is used for estimating posterior mean of $\Theta(1:t)$. As the number of particles N goes to infinity, this estimate approaches the true estimate under mild conditions [21].

It is common to encounter the situation that the posterior distribution $p(\Theta(1:t)|Y^l(1:t))$ cannot be sampled directly. Alternatively, importance sampling (IS) method [22] implements the empirical estimate in (17) by sampling from an easier distribution $q(\Theta(1:t)|Y^l(1:t))$, whose support includes that of $p(\Theta(1:t)|Y^l(1:t))$; that is,

$$\begin{aligned}\bar{f}_{\Theta}(1:t) &= \int f_{\Theta}(1:t) \frac{p(\Theta(1:t)|Y^l(1:t))}{q(\Theta(1:t)|Y^l(1:t))} \\ &\quad \times q(\Theta(1:t)|Y^l(1:t)) d\Theta(1:t) \\ &= \frac{\sum_{i=1}^N f_{\Theta}^{(i)}(1:t) w^{(i)}(1:t)}{\sum_{i=1}^N w^{(i)}(1:t)},\end{aligned}\quad (18)$$

where $\Theta^{(i)}(1:t)$ is sampled from distribution $q(\Theta(1:t)|Y^l(1:t))$, and each particle (i) has a weight given by

$$w^{(i)}(1:t) = \frac{p(\Theta^{(i)}(1:t)|Y^l(1:t))}{q(\Theta^{(i)}(1:t)|Y^l(1:t))}. \quad (19)$$

Equation (18) can be written as

$$\bar{f}_{\Theta}(1:t) = \sum_{i=1}^N f_{\Theta}^{(1:i)}(t) \bar{w}^{(i)}(1:t), \quad (20)$$

where the normalized weight is given as $\bar{w}^{(i)}(1:t) = w^{(i)}(1:t) / \sum_{j=1}^N w^{(j)}(1:t)$.

3.2. Sequential Bayesian importance sampling

Making use of the Markovian property in (13), we can have the following sequential BIS method to approximate the posterior distribution $p(\Theta(1:t)|Y^l(1:t))$. Basically, given an estimate of the posterior distribution at the previous time $t-1$, the method updates estimate of $p(\Theta(1:t)|Y^l(1:t))$ by combining a prediction step from a proposal sampling distribution in (24) and (25), and a sampling weight updating step in (26).

Suppose that a sequence of parameters $\hat{\Theta}(1:t-1)$ up to the previous time $t-1$ is given. By Markovian property

in (13), the posterior distribution of $\Theta(1:t) = (\hat{\Theta}(1:t-1)\theta(t))$ given $Y^l(1:t)$ can be written as

$$\begin{aligned}p(\Theta(1:t)|Y^l(1:t)) &\propto p(Y^l(t)|\theta(t)) p(\theta(t)|\hat{\theta}(t-1)) \\ &\quad \times \prod_{\tau=2}^{t-1} p(Y^l(\tau)|\hat{\theta}(\tau)) p(\hat{\theta}(\tau)|\hat{\theta}(\tau-1)) \\ &\quad \times p(Y^l(1)|\hat{\theta}(1)) p(\hat{\theta}(1)).\end{aligned}\quad (21)$$

We assume that the proposal distribution is in fact given as

$$\begin{aligned}q(\Theta(1:t)|Y^l(1:t)) &= q(Y^l(t)|\theta(t)) q(\theta(t)|\hat{\theta}(t-1)) \\ &\quad \times \prod_{\tau=2}^{t-1} q(\hat{\theta}(\tau)|\hat{\theta}(\tau-1)) q(Y^l(\tau)|\hat{\theta}(\tau)) \\ &\quad \times q(Y^l(1)|\hat{\theta}(1)) q(\hat{\theta}(1)).\end{aligned}\quad (22)$$

Plugging (21) and (22) into (19), we can update weight in a recursive way; that is,

$$\begin{aligned}w^{(i)}(1:t) &= \frac{p(Y^l(t)|\theta^{(i)}(t)) p(\theta^{(i)}(t)|\hat{\theta}^{(i)}(t-1))}{q(Y^l(t)|\theta^{(i)}(t)) q(\theta^{(i)}(t)|\hat{\theta}^{(i)}(t-1))} \\ &\quad \times \frac{\prod_{\tau=2}^{t-1} p(\hat{\theta}^{(i)}(\tau)|\hat{\theta}^{(i)}(\tau-1)) p(Y^l(\tau)|\hat{\theta}^{(i)}(\tau))}{\prod_{\tau=2}^{t-1} q(\hat{\theta}^{(i)}(\tau)|\hat{\theta}^{(i)}(\tau-1)) q(Y^l(\tau)|\hat{\theta}^{(i)}(\tau))} \\ &\quad \times \frac{p(Y^l(1)|\hat{\theta}^{(i)}(1)) p(\hat{\theta}^{(i)}(1))}{q(Y^l(1)|\hat{\theta}^{(i)}(1)) q(\hat{\theta}^{(i)}(1))} \\ &= w^{(i)}(1:t-1) \frac{p(Y^l(t)|\theta^{(i)}(t)) p(\theta^{(i)}(t)|\hat{\theta}^{(i)}(t-1))}{q(Y^l(t)|\theta^{(i)}(t)) q(\theta^{(i)}(t)|\hat{\theta}^{(i)}(t-1))}.\end{aligned}\quad (23)$$

Such a time-recursive evaluation of weights can be further simplified by allowing proposal distribution to be the prior distribution of the parameters. In this paper, the proposal distribution is given as

$$\begin{aligned}q(Y^l(t)|\theta^{(i)}(t)) &= 1, \\ q(\theta^{(i)}(t)|\hat{\theta}^{(i)}(t-1)) &= a_{s_t^{(i)} s_t^{(i)}} \pi_{s_t^{(i)} k_t^{(i)}} \mathcal{N}(\mu_{s_t^{(i)} k_t^{(i)}}^{l(i)}(t); \mu_{s_t^{(i)} k_t^{(i)}}^l, \Sigma_{s_t^{(i)} k_t^{(i)}}^l).\end{aligned}\quad (24)$$

Consequently, the above weight is updated by

$$w^{(i)}(t) \propto w^{(i)}(t-1) p(Y^l(t)|\theta^{(i)}(t)) p(\mu_n^{l(i)}(t)|\hat{\mu}_n^{l(i)}(t-1)). \quad (26)$$

Remark 2. Given $\hat{\Theta}(1:t-1)$, there is an optimal proposal distribution that minimizes variance of the importance weights. This optimal proposal distribution is in fact the posterior distribution $p(\theta(t)|\hat{\Theta}(1:t-1), Y^l(1:t))$ [23, 24].

3.3. Rao-Blackwellization and the extended Kalman filter

Note that $\mu_n^{l(i)}(t)$ in particle (i) is assumed to be distributed in $\mathcal{N}(\mu_n^{l(i)}(t); \hat{\mu}_n^{l(i)}(t-1), V_n^l)$. By the Rao-Blackwell theorem [25], the variance of weight in (26) can be reduced by marginalizing out $\mu_n^{l(i)}(t)$. Therefore, we have

$$w^{(i)}(t) \propto w^{(i)}(t-1) \times \int_{\mu_n^{l(i)}(t)} p(Y^l(t)|\theta^{(i)}(t)) \times p(\mu_n^{l(i)}(t)|\hat{\mu}_n^{l(i)}(t-1)) d\mu_n^{l(i)}(t). \quad (27)$$

Referring to (9) and (10), we notice that the integrand $p(Y^l(t)|\theta^{(i)}(t))p(\mu_n^{l(i)}(t)|\hat{\mu}_n^{l(i)}(t-1))$ is a state-space model by (7) and (8). In this state-space model, given $s_t^{(i)}$, $k_t^{(i)}$, and $\mu_{s_t^{(i)}k_t^{(i)}}^{l(i)}(t)$, $\mu_n^{l(i)}(t)$ is the hidden continuous-valued vector distributed in $\mathcal{N}(\mu_n^{l(i)}(t); \hat{\mu}_n^{l(i)}(t-1), V_n^l)$, and $Y^l(t)$ is the observed signal of this model. This integral in (27) can be analytically obtained if we linearize (7) with respect to $\mu_n^{l(i)}(t)$. The linearized state-space model provides an extended Kalman filter (EKF) (see Appendix B for the detail of EKF), and the integral is $p(Y^l(t)|s_t^{(i)}, k_t^{(i)}, \mu_{s_t^{(i)}k_t^{(i)}}^{l(i)}(t), \hat{\mu}_n^{l(i)}(t-1), Y^l(t-1))$, which is the predictive likelihood shown in (B.1). An advantage of updating weight by (27) is its simplicity of implementation.

Because the predictive likelihood is obtained from EKF, the weight $w^{(i)}(t)$ may not asymptotically approach the target posterior distribution. One way to achieve asymptotically the target posterior distribution may follow a method called the *extended Kalman particle filter* in [26], where the weight is updated by

$$w^{(i)}(t) \propto w^{(i)}(t-1) \frac{p(Y^l(t)|\theta^{(i)}(t))p(\mu_n^{l(i)}(t)|\hat{\mu}_n^{l(i)}(t-1))}{q(\mu_n^{l(i)}(t)|\hat{\mu}_n^{l(i)}(t-1), s_t^{(i)}, k_t^{(i)}, \mu_{s_t^{(i)}k_t^{(i)}}^{l(i)}(t), Y^l(t))}, \quad (28)$$

and the proposal distribution for $\mu_n^{l(i)}(t)$ is from the posterior distribution of $\mu_n^{l(i)}(t)$ by EKF; that is,

$$\begin{aligned} & q(\mu_n^{l(i)}(t)|\hat{\mu}_n^{l(i)}(t-1), s_t^{(i)}, k_t^{(i)}, \mu_{s_t^{(i)}k_t^{(i)}}^{l(i)}(t), Y^l(t)) \\ &= \mathcal{N}(\mu_n^{l(i)}(t); \hat{\mu}_n^{l(i)}(t-1) + G^{(i)}(t)\alpha^{(i)}(t-1), K^{(i)}(t)), \end{aligned} \quad (29)$$

where Kalman gain $G^{(i)}(t)$, innovation vector $\alpha^{(i)}(t-1)$, and posterior variance $K^{(i)}(t)$ are respectively given in (B.7), (B.2), and (B.4).

However, for the following reasons, we did not apply the stricter *extended Kalman particle filter* to our problem. First, the scheme in (28) is not Rao-Blackwellized. The variance of sampling weights might be larger than the Rao-Blackwellized method in (27). Second, although observation function (7) is

nonlinear, it is convex and continuous. Therefore, linearization of (7) with respect to $\mu_n^{l(i)}(t)$ may not affect the mode of the posterior distribution $p(\mu_n^{l(i)}(1:t)|Y^l(1:t))$. By the asymptotic theory (see [25, page 430]), under the mild condition that the variance of noise $N^l(t)$ (parameterized by V_n^l) is finite, bias for estimating $\hat{\mu}_n^{l(i)}(t)$ by MMSE estimation via (17) with weight given by (27) may be reduced as the number of particles N grows large. (However, unbiasedness for estimating $\hat{\mu}_n^{l(i)}(t)$ may not be established since there are zero derivatives with respect to the parameter $\mu_n^{l(i)}(t)$ in (7).) Third, evaluation of (28) is computationally more expensive than (27), because (28) involves calculation processes on two state-space models. We will show some experiments in Section 4.1 to support the above considerations.

Remark 3. Working in linear spectral domain in (2) for noise estimation does not require EKF. Thus, if the noise parameter in $\Theta(t)$ and the observations are both in the linear spectral domain, the corresponding sequential BIS can achieve asymptotically the target posterior distribution (12). In practice, however, due to the large variance in the linear spectral domain, we may frequently encounter numerical problems that make it difficult to build an accurate statistical model for both clean speech and noise. Compressing linear spectral power into log-spectral domain is commonly used in speech recognition to achieve more accurate models. Furthermore, because the performance by adapting acoustic models (modifying mean and variance of acoustic models) is usually higher than enhanced noisy speech signals for noisy speech recognition [10], in the context of speech recognition, it is beneficial to devise an algorithm that works in the domain for building acoustic models. In our examples, acoustic models are trained from cepstral or log-spectral features, thus, the parameter estimation algorithm is devised in the log-spectral domain, which is linearly related to the cepstral domain. We will show later that the estimated noise parameter $\hat{\mu}_n^{l(i)}(t)$ substitutes $\hat{\mu}_n^l$ using a log-add method (36) to adapt acoustic model mean vectors. Thus, to avoid inconsistency due to transformations between different domains, the noise parameter may be estimated in log-spectral domain, instead of linear spectral domain.

3.4. Avoiding degeneracy by resampling

Since the above particles are discrete approximations of the posterior distribution $p(\Theta(1:t)|Y^l(1:t))$, in practice, after several steps of sequential BIS, the weights of not all but some particles may become insignificant. This could cause a large variance in the estimate. In addition, it is not necessary to compute particles with insignificant weights. Selection of the particles is thus necessary to reduce the variance and to make efficient use of computational resources.

Many methods for selecting particles have been proposed, including sampling-importance resampling (SIR) [27], residual resampling [28], and so forth. We apply residual resampling for its computational simplicity. This method basically avoids degeneracy by discarding those particles with insignificant weights, and in order to keep the number of the

particles constant, particles with significant weights are duplicated. The steps are as follows. Firstly, set $\tilde{N}^{(i)} = \lfloor N\tilde{w}^{(i)}(1:t) \rfloor$. Secondly, select the remaining $\tilde{N} = N - \sum_{i=1}^N \tilde{N}^{(i)}$ particles with new weights $\hat{w}^{(i)}(1:t) = \tilde{N}^{-1}(\tilde{w}^{(i)}(1:t)N - \tilde{N}^{(i)})$, and obtain particles by sampling in a distribution approximated by these new weights. Finally, add the particles to those obtained in the first step. After this residual sampling step, the weight for each particle is $1/N$. Besides computational simplicity, residual resampling is known to have smaller variance $\text{var} N^{(i)} = \tilde{N}\hat{w}^{(i)}(1:t)(1 - \hat{w}^{(i)}(1:t))$ compared to that of SIR (which is $\text{var} N^{(i)}(t) = N\tilde{w}^{(i)}(1:t)(1 - \tilde{w}^{(i)}(1:t))$). We denote the particles after the selection step as $\{\hat{\Theta}^{(i)}(1:t); i = 1 \dots N\}$.

After the selection step, the discrete nature of the approximation may lead to large bias/variance, of which the extreme case is that all the particles have the same parameters estimated. Therefore, it is necessary to introduce a resampling step to avoid such degeneracy. We apply a Metropolis-Hastings smoothing [19] step in each particle by sampling a candidate parameter given the currently estimated parameter according to the proposal distribution $q(\theta^*(t)|\hat{\theta}^{(i)}(t))$. For each particle, a value is calculated as

$$g^{(i)}(t) = g_1^{(i)}(t)g_2^{(i)}(t), \quad (30)$$

where $g_1^{(i)}(t) = p((\hat{\Theta}^{(i)}(t-1)\theta^*(t))|Y^l(1:t))/p(\hat{\Theta}^{(i)}(1:t)|Y^l(1:t))$ and $g_2^{(i)}(t) = q(\hat{\theta}^{(i)}(t)|\theta^*(t))/q(\theta^*(t)|\hat{\theta}^{(i)}(t))$. Within an acceptance possibility $\min\{1, g^{(i)}(t)\}$, the Markov chain then moves towards the new parameter $\theta^*(t)$; otherwise, it remains at the original parameter.

To simplify calculations, we assume that the proposal distribution $q(\theta^*(t)|\hat{\theta}^{(i)}(t))$ is symmetric.⁵ Note that $p(\hat{\Theta}^{(i)}(1:t)|Y^l(1:t))$ is proportional to $\tilde{w}^{(i)}(1:t)$ up to a scalar factor. With (27), (B.1), and $\tilde{w}^{(i)}(1:t-1) = 1/N$, we can obtain the acceptance possibility as

$$\min \left[1, \frac{p\left(Y^l(t)|s_t^{*(i)}, k_t^{*(i)}, \mu_{s_t^{*(i)}k_t^{*(i)}}^{l(i)}(t), \hat{\mu}_n^{l(i)}(t-1), Y^l(t-1)\right)}{p\left(Y^l(t)|\tilde{s}_t^{(i)}, \tilde{k}_t^{(i)}, \tilde{\mu}_{\tilde{s}_t^{(i)}\tilde{k}_t^{(i)}}^{l(i)}(t), \hat{\mu}_n^{l(i)}(t-1), Y^l(t-1)\right)} \right]. \quad (31)$$

Denote the obtained particles hereafter as $\{\hat{\Theta}^{(i)}(1:t); i = 1, \dots, N\}$ with equal weights.

3.5. Noise parameter estimation via the sequential Monte Carlo method

Following the above considerations, we present the implemented algorithm for noise parameter estimation. Given that, at time $t-1$, N particles $\{\hat{\Theta}^{(i)}(1:t-1); i = 1, \dots, N\}$ are

distributed approximately according to $p(\Theta(1:t-1)|Y^l(1:t-1))$, the sequential Monte Carlo method proceeds as follows at time t .

Algorithm 1.

Bayesian importance sampling step

- (1) Sampling. For $i = 1, \dots, N$, sample a proposal $\hat{\Theta}^{(i)}(1:t) = (\hat{\Theta}^{(i)}(1:t-1)\hat{\theta}^{(i)}(t))$ by
 - (a) sampling $s_t^{(i)} \sim a_{s_{t-1}s_t}$;
 - (b) sampling $k_t^{(i)} \sim \pi_{s_t k_t}$;
 - (c) sampling $\hat{\mu}_{s_t k_t}^{l(i)}(t) \sim \mathcal{N}(\mu_{s_t k_t}^l(t); \mu_{s_t k_t}^{l(i)}, \Sigma_{s_t k_t}^l)$.
- (2) Extended Kalman prediction. For $i = 1, \dots, N$, evaluate (B.2)–(B.7) for each particle by EKFs. Predict noise parameter for each particle by

$$\hat{\mu}_n^{l(i)}(t) = \hat{\mu}_n^{l(i)}(t|t-1), \quad (32)$$

where $\hat{\mu}_n^{l(i)}(t|t-1)$ is given in (B.3).

- (3) Weighting. For $i = 1, \dots, N$, evaluate the weight of each particle $\hat{\Theta}^{(i)}$ by

$$\hat{w}^{(i)}(1:t) \propto \hat{w}^{(i)}(1:t-1)p\left(Y^l(t)|s_t^{(i)}, k_t^{(i)}, \hat{\mu}_{s_t k_t}^{l(i)}(t), \hat{\mu}_n^{l(i)}(t-1), Y^l(t-1)\right), \quad (33)$$

where the second term in the right-hand side of the equation is the predictive likelihood, given in (B.1), of the EKF.

- (4) Normalization. For $i = 1, \dots, N$, the weight of the i th particle is normalized by

$$\hat{w}^{(i)}(1:t) = \frac{\hat{w}^{(i)}(1:t)}{\sum_{i=1}^N \hat{w}^{(i)}(1:t)}. \quad (34)$$

Resampling

- (1) Selection. Use residual resampling to select particles with larger normalized weights and discard those particles with insignificant weights. Duplicate particles of large weights in order to keep the number of particles as N . Denote the set of particles after the selection step as $\{\tilde{\Theta}^{(i)}(1:t); i = 1, \dots, N\}$. These particles have equal weights $\tilde{w}^{(i)}(1:t) = 1/N$.
- (2) Metropolis-Hastings smoothing. For $i = 1, \dots, N$, sample $\Theta^{*(i)}(1:t) = (\tilde{\Theta}^{(i)}(1:t-1)\theta^*(t))$ from step (1) to step (3) in the Bayesian importance sampling step with starting parameters given by $\tilde{\Theta}^{(i)}(1:t)$. For $i = 1, \dots, N$, set an acceptance possibility by (31). For $i = 1, \dots, N$, accept $\Theta^{*(i)}(1:t)$ (i.e., substitute $\tilde{\Theta}^{(i)}(1:t)$ by $\Theta^{*(i)}(1:t)$) with probability $r^{(i)}(t) \sim U(0, 1)$. The particles after the step are $\{\hat{\Theta}^{(i)}(1:t); i = 1, \dots, N\}$ with equal weights $\hat{w}^{(i)}(1:t) = 1/N$.

⁵Generating $\theta^*(t)$ involves sampling speech state s_t^* from $\tilde{s}_{1:t}^{(i)}$ according to a first-order Markovian transition probability $p(s_t^*|\tilde{s}_t^{(i)})$ in the graphical model in Figure 2. Usually, this transition probability matrix is not symmetric; that is, $p(s_t^*|\tilde{s}_t^{(i)}) \neq p(\tilde{s}_t^{(i)}|s_t^*)$. Our assumption of symmetric proposal distribution $q(\theta^*(t)|\hat{\theta}^{(i)}(t))$ is for simplicity in calculating an acceptance possibility.

TABLE 1: State estimation experiment results. The results show the mean and variance of the mean squared error (MSE) calculated over 100 independent runs.

Algorithm	MSE		Averaged execution time (s)
	Mean	Variance	
Particle filter	8.713	49.012	5.338
Extended Kalman particle filter	6.496	34.899	13.439
Rao-Blackwellized particle filter	4.559	8.096	6.810

Noise parameter estimation

- (1) Noise Parameter Estimation. With the above generated particles at each time t , estimation of the noise parameter $\mu_n^l(t)$ may be acquired by MMSE. Since each particle has the same weight, MMSE estimation of $\hat{\mu}_n^l(t)$ can be easily carried out as

$$\hat{\mu}_n^l(t) = \frac{1}{N} \sum_{i=1}^N \hat{\mu}_n^{l(i)}(t). \quad (35)$$

The computational complexity of the algorithm at each time t is $O(2N)$ and is roughly equivalent to $2N$ EKFs. These steps are highly parallel, and if resources permit, can be implemented in a parallel way. Since the sampling is based on BIS, the storage required for the calculation does not change over time. Thus the computation is efficient and fast.

Note that the estimated $\hat{\mu}_n^l(t)$ may be biased from the true physical mean vector for log-spectral noise power $N^l(t)$, because the function plotted in Figure 1 has zero derivative with respect to $n_j^l(t)$ in regions where $n_j^l(t)$ is much smaller than $x_j^l(t)$. For those $\hat{\mu}_n^{l(i)}(t)$ which are initialized with values larger than speech mean vector $\mu_{s_i, k_i}^{l(i)}$, updating by EKF may be lower bounded around the speech mean vector. As a result, the updated $\hat{\mu}_n^l(t) = 1/N \sum_{i=1}^N \hat{\mu}_n^{l(i)}(t)$ may not be the true noise log-spectral power.

Remark 4. The above problem, however, may not hurt a model-based noisy speech recognition system, since it is the modified likelihood in (10) that is used to decode speech signals.⁶ But in a speech enhancement system, noisy speech spectrum is directly processed on the estimated noise parameter. Therefore, biased estimation of the noise parameter may hurt performances more apparently than in a speech recognition system.

4. EXPERIMENTS

We first conducted synthetic experiments in Section 4.1 to compare three types of particle filters presented in Sections 3.2 and 3.3. Then, in the following sections, we present applications of the above noise parameter estimation method

⁶The likelihood of the observed signal $Y^l(t)$, given speech model parameter and a noise parameter, is the same as long as the noise parameter is much smaller than the speech parameter $\mu_{s_i, k_i}^{l(i)}(t)$.

based on Rao-Blackwellized particle filter (27). We consider particularly difficult tasks for speech processing, speech enhancement, and noisy speech recognition in nonstationary noisy environments. We show in Section 4.2 that the method can track noise dynamically. In Section 4.3, we show that the method improves system robustness to noise in an ASR system. Finally, we present results on speech enhancement in Section 4.4, where the estimated noise parameter is used in a time-varying linear filter to reduce noise power.

4.1. Synthetic experiments

This section⁷ presents some experiments⁸ to show the validity of Rao-Blackwellized filter applied to the state-space model in (7) and (8). A sequence of $\mu_n^l(1:t)$ was generated from (8), where state-process noise variance V_n^l was set to 0.75. Speech mean vector $\mu_{s_i, k_i}^l(t)$ in (7) was set to a constant 10. The observation noise variance Σ_{s_i, k_i}^l was set to 0.00005. Given only the noisy observation $Y^l(1:t)$ for $t = 1, \dots, 60$, different filters (particle filter by (26), extended Kalman particle filter by (28), and Rao-Blackwellized particle filter by (27)) were used to estimate the underlying state sequence $\mu_n^l(1:t)$. The number of particles in each type of filter was 200, and all the filters applied residual resampling [28]. The experiments were repeated for 100 times with random reinitialization of $\mu_n^l(1)$ for each run. Table 1 summarizes the mean and variance of the MSE of the state estimates, together with the averaged execution time of each filter. Figure 3 compares the estimates generated from a single run of the different filters. In terms of MSE, the extended Kalman particle filter performed better than the particle filter. However, the execution time of the extended Kalman particle filter was the longest (more than two times longer than that of particle filter (26)). Performance of the Rao-Blackwellized particle filter of (27) is clearly the best in terms of MSE. Notice that its averaged execution time was comparable to that of particle filter.

4.2. Estimation of noise parameter

Experiments were performed on the TI-Digits database downsampled to 16 kHz. Five hundred clean speech utterances from 15 speakers and 111 utterances unseen in the training set were used for training and testing, respectively.

⁷A Matlab implementation of the synthetic experiments is available by sending email to the corresponding author.

⁸All variables in these experiments are one dimensional.

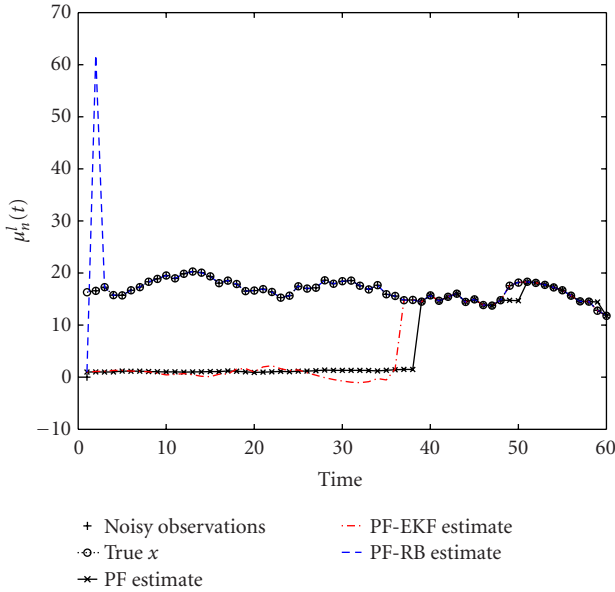


FIGURE 3: Plot of estimates generated by the different filters on the synthetic state estimation experiment versus true state. PF denotes particle filter by (26). PF-EKF denotes particle filter with EKF proposal sampling by (28). PF-RB denotes Rao-Blackwellized particle filter by (27).

Digits and silence were respectively modeled by 10-state and 3-state whole-word HMMs with 4 diagonal Gaussian mixtures in each state.

The window size was 25.0 milliseconds with a 10.0 milliseconds shift. Twenty-six filter banks were used in the binning stage; that is, $J = 26$. Speech feature vectors were Mel-scaled frequency cepstral coefficients (MFCCs), which were generated by transforming log-spectral power spectra vector with discrete Cosine transform (DCT). The baseline system had 98.7% word accuracy for speech recognition under clean conditions.

For testing, white noise signal was multiplied by a chirp signal and a rectangular signal in the time domain. The time-varying mean of the noise power as a result changed either continuously, denoted as experiment A, or dramatically, denoted as experiment B. SNR of the noisy speech ranged from 0 dB to 20.4 dB. We plotted the noise power in the 12th filter bank versus frames in Figure 4, together with the estimated noise power by the sequential method with the number of particles N set to 120 and the environment driving noise variance V_n^l set to 0.0001. As a comparison, we also plotted in Figure 5 the noise power and its estimate by the method with the same number of particles but larger driving noise variance set to 0.001.

Four seconds of contaminating noise were used to initialize $\hat{\mu}_n^l(0)$ in the noise estimation method. Initial value $\hat{\mu}_n^{l(i)}(0)$ of each particle was obtained by sampling from $\mathcal{N}(\hat{\mu}_n^l(0) + \zeta(0), 10.0)$, where $\zeta(0)$ was distributed in $U(-1.0, 9.0)$. To apply the estimation algorithm in Section 3.5, observation vectors were transformed into log-spectral domain.

Based on the results in Figures 4 and 5, we make the following observations. First, the method can track the evolution of the noise power. Second, the larger driving noise variance V_n^l will make faster convergence but larger estimation error. Third, as discussed in Section 3.5, there was large bias in the region where noise power changed from large to small. Such observation was more explicit in experiment B (noise multiplied with a rectangular signal).

4.3. Noisy speech recognition in time-varying noise

The experiment setup was the same as in the previous experiments in Section 4.2. Features for speech recognition were MFCCs plus their first- and second-order time differentials. Here, we compared three systems. The first was the baseline trained on clean speech without noise compensation (denoted as Baseline). The second was the system with noise compensation, which transformed clean speech acoustic models by mapping clean speech mean vector μ_{s_i, k_i}^l at each state s_i and Gaussian density k_i with the function [8]

$$\hat{\mu}_{s_i, k_i}^l = \mu_{s_i, k_i}^l + \log(1 + \exp(\hat{\mu}_n^l - \mu_{s_i, k_i}^l)), \quad (36)$$

where $\hat{\mu}_n^l$ was obtained by averaging noise log-spectral in noise-alone segments in the testing set. This system was denoted as stationary noise assumption (SNA). The third system used the method in Section 3.5 to estimate the noise parameter $\hat{\mu}_n^l(t)$ without training transcript. The estimated noise parameter was plugged into $\hat{\mu}_n^l$ in (36) for adapting acoustic mean vector at each time t . This system was denoted according to the number of particles and variance of the environment driving noise V_n^l .

4.3.1. Results in the simulated nonstationary noise

In terms of recognition performance in the simulated nonstationary noise described in Section 4.2, Table 2 shows that the method can effectively improve system robustness to the time-varying noise. For example, with 60 particles and the environment driving noise variance V_n^l set to 0.001, the method improved word accuracy from 75.3%, achieved by SNA, to 94.3% in experiment A. The table also shows that the word accuracies can be improved by increasing the number of particles. For example, given driving noise variance V_n^l set to 0.0001, increasing the number of particles from 60 to 120 could improve word accuracy from 77.1% to 85.8% in experiment B.

4.3.2. Speech recognition in real noise

In this experiment, speech signals were contaminated by highly nonstationary machine gun noise in different SNRs. The number of particles was set to 120, and the environment driving noise variance V_n^l was set to 0.0001. Recognition performances are shown in Table 3, together with Baseline and SNA. It is observed that, in all SNR conditions, the method in Section 3.5 further improved system performances in comparison with SNA. For example, in 8.9 dB SNR, the method improved word accuracy from 75.6% by SNA to 83.1%. As a whole, it reduced the word error rate by 39.9% more than SNA.

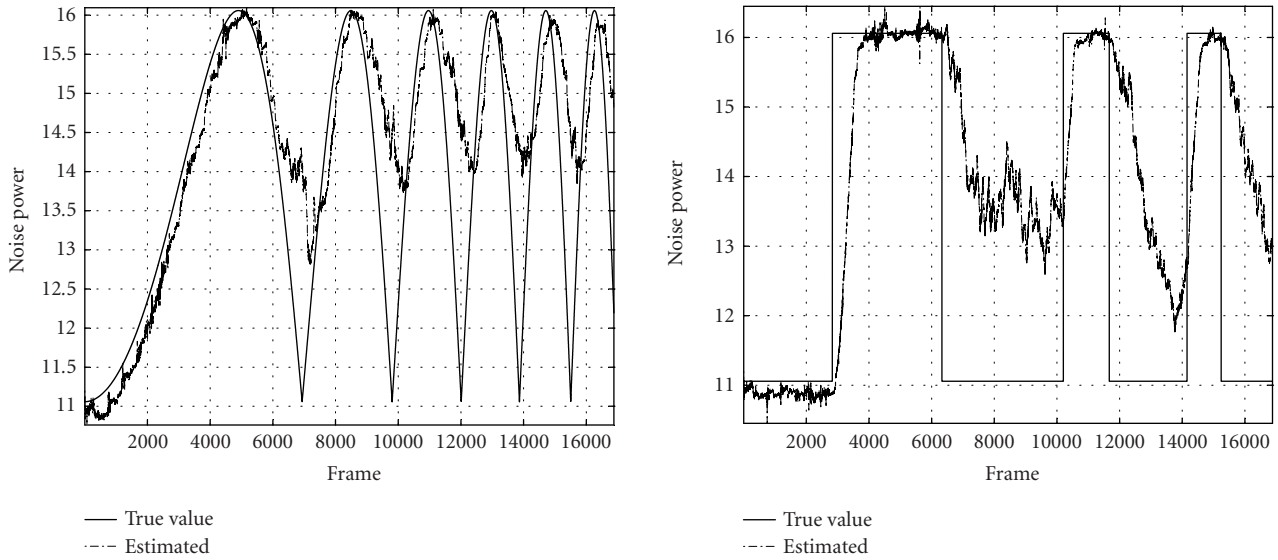


FIGURE 4: Estimation of the time-varying parameter $\mu_n^l(t)$ by the sequential Monte Carlo method at the 12th filter bank in experiment A. The number of particles is 120. The environment driving noise variance is 0.0001. The solid curve is the true noise power, whereas the dash-dotted curve is the estimated noise power.

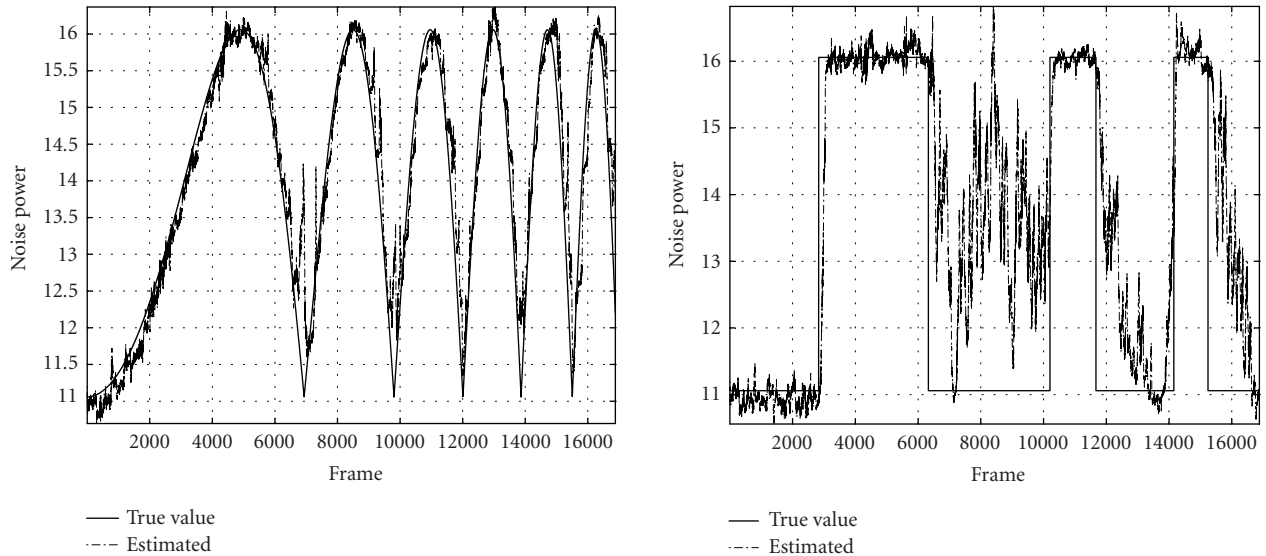


FIGURE 5: Estimation of the time-varying parameter $\mu_n^l(t)$ by the sequential Monte Carlo method at the 12th filter bank in experiment A. The number of particles is 120. The environment driving noise variance is 0.001. The solid curve is the true noise power, whereas the dash-dotted curve is the estimated noise power.

4.4. Perceptual speech enhancement

Enhanced speech $\hat{x}(t)$ is obtained by filtering the noisy speech sequence $y(t)$ via a time-varying linear filter $h(t)$; that is,

$$\hat{x}(t) = h(t) * y(t). \quad (37)$$

This process can be studied in the frequency domain as multiplication of the noisy speech power spectrum $y_j^{\text{lin}}(t)$ by a

time-varying linear coefficient at each filter bank; that is,

$$\hat{x}_j^{\text{lin}}(t) = h_j(t) \cdot y_j^{\text{lin}}(t), \quad (38)$$

where $h_j(t)$ is the gain at filter bank j at time t . Referring to (2), we can expand it as

$$\hat{x}_j^{\text{lin}}(t) = h_j(t)x_j^{\text{lin}}(t) + h_j(t)n_j^{\text{lin}}(t). \quad (39)$$

We are left with two choices for linear time-varying filters.

TABLE 2: Word accuracy (%) in simulated nonstationary noise, achieved by the sequential Monte Carlo method in comparison with baseline without noise compensation, denoted as Baseline, and noise compensation assuming stationary noise, denoted as stationary noise assumption.

Experiment	Baseline	Stationary noise assumption	No. of particles = 60		No. of particles = 120	
			V_n^l		V_n^l	
			0.001	0.0001	0.001	0.0001
A	48.2	75.3	94.3	94.0	94.3	94.6
B	53.0	78.0	82.2	77.1	85.8	85.8

TABLE 3: Word accuracy (%) in machine gun noise, achieved by the sequential Monte Carlo method in comparison with baseline without noise compensation, denoted as Baseline, and noise compensation assuming stationary noise, denoted as stationary noise assumption.

SNR (dB)	Baseline	Stationary noise assumption	No. of particles = 120, $V_n^l = 0.0001$
28.9	90.4	92.8	97.6
14.9	64.5	76.8	88.3
8.9	56.0	75.6	83.1
1.6	50.0	69.0	72.9

- (1) Wiener filter constructs the coefficient as

$$h_j(t) = 1 - \frac{\hat{n}_j^{\text{lin}}(t)}{y_j^{\text{lin}}(t)}, \quad (40)$$

where $\hat{n}_j^{\text{lin}}(t)$ is the estimate of noise power spectrum.

- (2) The criterion for perceptual filter is to construct $h_j(t)$ so that the amplitude of the filtered noise power spectra $h_j(t) \cdot n_j^{\text{lin}}(t)$ is below the masking threshold of the denoised speech; that is,

$$h_j(t) \cdot n_j^{\text{lin}}(t) \leq T_j(t), \quad (41)$$

where $T_j(t)$ is the masking threshold of the denoised speech signal. The threshold is a function of clean speech spectrum $x_j^{\text{lin}}(t)$. Since $x_j^{\text{lin}}(t)$ is not directly observed, the following equation is used instead, which makes the masking threshold a function of the estimated noise power spectra $\hat{n}_j^{\text{lin}}(t)$:

$$\hat{x}_j^{\text{lin}}(t) = y_j^{\text{lin}}(t) - \hat{n}_j^{\text{lin}}(t). \quad (42)$$

The perceptual filter exploits the masking properties of the human auditory system, and it has been employed by many researchers (e.g., [14]) in order to provide improved performance over the Wiener filter in low SNR conditions. Masking occurs because the auditory system is incapable of distinguishing two signals close in time or frequency domain. This is manifested by an evaluation of the minimum threshold of audibility due to a masker signal. Masking has been widely applied to speech and audio coding [15]. We consider frequency masking [15] when a weak signal is made inaudible by a stronger signal occurring simultaneously.

Both Wiener filter and perceptual filter require the estimated noise power spectrum $\hat{n}_j^{\text{lin}}(t)$. Under the assumption of stationary noise, the noise power spectrum can be estimated from noise-alone segments provided by explicit VAD, for example, speech enhancement scheme in [7]. However, in real applications, we encounter time-varying noise, which may change its statistics during speech utterances.

The objective of this section is to test the above devised method in Section 3.5 for speech enhancement in time-varying noise. The estimated $\hat{\mu}_n^l(t)$ is converted to linear spectral domain $\hat{\mu}_n^{\text{lin}}(t)$ by exponential operation. Corresponding j th element in $\hat{\mu}_n^{\text{lin}}(t)$ substitutes $\hat{n}_j^{\text{lin}}(t)$ in (40) and (42), respectively, to construct Wiener filter and perceptual filter. Therefore, the proposed speech enhancement algorithm is a combination of sequential noise parameter estimation by sequential Monte Carlo method and speech enhancement method with time-varying linear filtering. Diagram of the algorithm is shown in Figure 6. At each frame t , the algorithm carries out the noise parameter estimation in the log-spectral domain and perceptual enhancement of noisy speech in the time domain. Noise parameter estimation in the module “Noise parameter estimation” works in the log-spectral domain of input speech signals. Estimation of noise parameter is given by Algorithm 1. With the estimated noise parameter at the current frame, the module “wiener filter” outputs the enhanced speech spectrum in linear spectral domain, and the enhanced speech spectrum is used in “masking threshold calculation.” Perceptual filter based on masking threshold and the estimated noise parameter is constructed in the module “perceptual filter.” With the time-varying perceptual filter constructed, input noisy speech signal is filtered in time domain in the module “filtering” to obtain perceptually enhanced signal $\hat{x}(t)$. A detailed description of this algorithm is provided in the following sections.

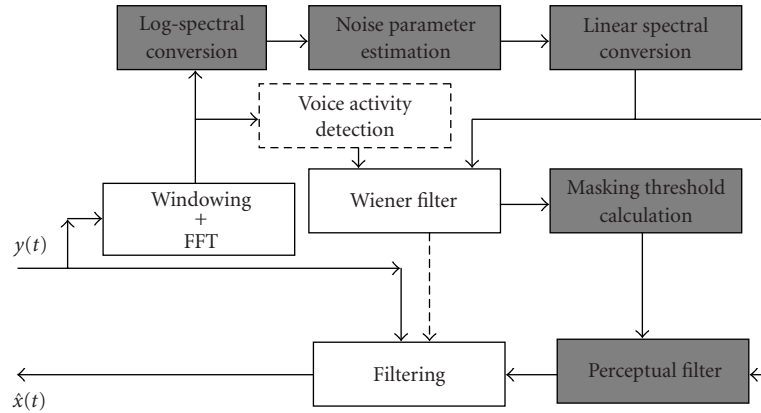


FIGURE 6: Diagram of the proposed speech enhancement method. Noisy signal $y(t)$ is converted into linear spectral amplitude in “windowing + FFT.” Noise parameter is sequentially estimated in “noise parameter estimation.” The estimated noise parameter is converted back into linear spectral domain and is fed into “Wiener filter” to obtain enhanced linear power spectrum. The enhanced spectrum is inputted to “masking threshold calculation,” and the obtained masking threshold is used in perceptual filter with the estimated noise parameter in linear spectral domain. Module “perceptual filter” outputs filter coefficients for speech enhancement in “filtering,” which outputs the enhanced signal $\hat{x}(t)$.

4.4.1. Masking threshold calculation

The masking threshold $T_j(t)$ is obtained through modeling the frequency selectivity of the human ear and its masking property. This paper applies a computational model of masking by Johnston [15].

Frequency masking threshold calculation

(1) Frequency analysis. According to a mapping between linear frequency and Bark frequency [14], power spectrum $x_j^{\text{lin}}(t)$ after short-time Fourier transform (STFT) of input speech signal is combined in each Bark bank b ($1 \leq b \leq B$) by

$$x_b^{\text{lin}}(t) = \sum_{j=b_L}^{b_H} x_j^{\text{lin}}(t), \quad (43)$$

where b_L and b_H denote the lowest and the highest frequency for the bark index b .

(2) Convolution with spreading function. The spreading function \mathbf{S} is used to estimate the effects of masking across critical bands. One example of the spreading function B_b at $b = 2$ is plotted in Figure 7. The spread Bark spectrum at bark index b is denoted as $C_b^{\text{lin}}(t) = B_b x_b^{\text{lin}}(t)$.

(3) Relative threshold calculation based on tone-like or noise-like determination. The tone-like or noise-like is determined by spectral flatness measure (SFM), which is calculated by measuring the decibel (dB) of the ratio of the geometric mean of the power spectrum to the arithmetic mean of the power spectrum.

(4) Masking threshold calculation. The relative threshold is subtracted from the spread critical band spectrum to yield the spread threshold estimate.

(5) Renormalization and including absolute threshold information [15].

(6) Converting the masking threshold from Bark frequency to linear frequency domain. The masking threshold in linear spectral domain $T_j(t)$ is obtained as a result.

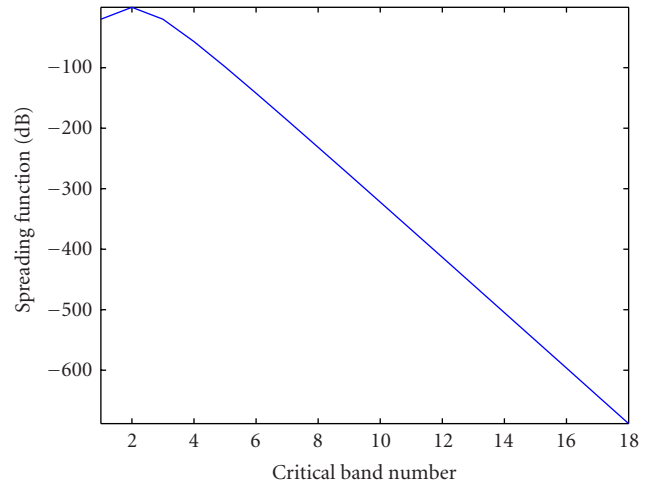


FIGURE 7: Spreading function for the noise masking threshold calculation. The plot shows the spreading function applied to critical band at 2.

An example of masking threshold in linear spectral domain for a given input spectrum is plotted in Figure 8. The sampling frequency is 8 kHz. Therefore, the total number of critical bands is $B = 18$. In the method presented above, the masking threshold is calculated from the clean speech signal.

4.4.2. Wiener filter and perceptual filter

We apply the method in Section 3.5 for time-varying noise parameter estimation. The j th element in $\hat{\mu}_n^l(t)$ is converted to linear spectral domain by exponential operation and then substitutes $\hat{n}_j^{\text{lin}}(t)$ in (40) and (42), respectively, for Wiener filter and perceptual filter. Masking threshold of the perceptual filter is obtained from Section 4.4.1.

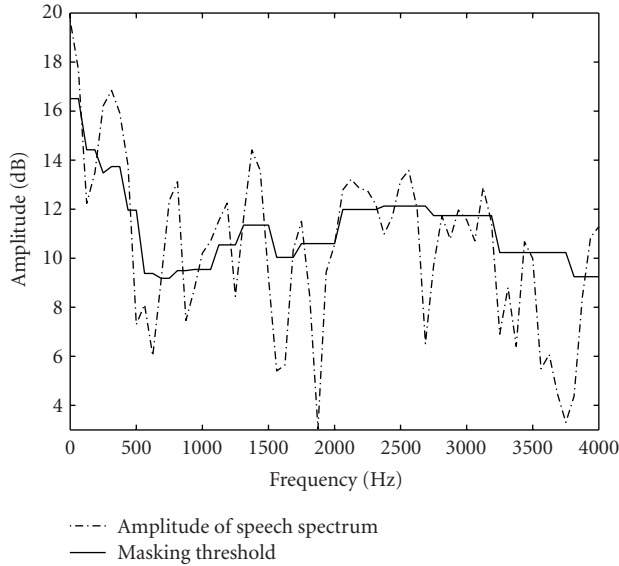


FIGURE 8: Example of the masking threshold $T_b(t)$.

4.4.3. Experimental results

Experiments were performed on Aurora 2 database. Speech models were trained on 8840 clean speech utterances. The model was an HMM with 18 states and 8 Gaussian mixtures in each state. Noise model was a single Gaussian density with time-varying mean vector. Window size was 25.0 milliseconds with a 10.0 milliseconds shift. J was set to 65.

We compared three systems. The first system, denoted as Baseline, was a speech enhancement system based on ETSI proposal [7], in which a VAD is used for decision of speech/nonspeech segments. Noise parameters were estimated from segmented noise-alone frames. The second system, denoted as Known, differs from the first system in that the Wiener filter was designed with noise parameters estimated by the proposed method. The third system, denoted as Perceptual, was a perceptual filter with noise parameter estimated by the proposed method.

VAD was initialized during the first three frames in each utterance. Driving variance V_n^i in (9) was set to 0.0003. Number of particles (N in (35)) was set to 800.

Noise signals were (1) simulated nonstationary noise, generated by multiplying white noise with a time-varying continuous factor in time domain, (2) Babble noise, and (3) Restaurant noise.

4.4.4. Performance evaluation

Spectrogram

An example of the original clean speech signals, noisy signals in the simulated nonstationary noise, and enhanced signals are shown in Figure 9. The contrast is more evident by viewing their corresponding spectrogram in Figure 10. It is observed that the noise power appeared after 0.4 seconds, which

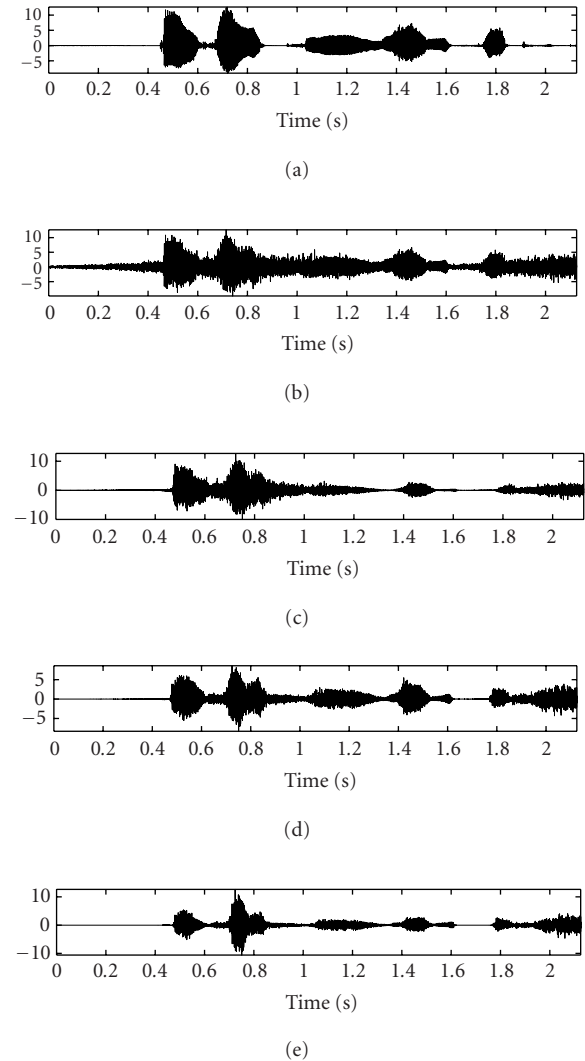


FIGURE 9: An example of signals. (a) Clean speech signal in English "Oh Oh Two One Six." (b) Noisy signal (noise is the simulated nonstationary noise and SNR is -0.2 dB). (c) Enhanced speech signal by Wiener filter (system Baseline). (d) Enhanced speech signal by Wiener filter with noise parameters estimated by the proposed method (system Known). (e) Enhanced speech signal by perceptual filter with noise parameters estimated by the proposed method (system Perceptual).

was almost at the time when the speech segments occurred. Figure 10c shows that Baseline cannot handle this nonstationarity of the noise, and the enhanced signal by the system still contains much noise power in the speech segments. On the contrary, with the proposed method, the enhanced signal by Known has reduced the noise power in speech segments (shown in Figure 10d). Perceptual reduces noise in the enhanced signal to a greater extent than the other two systems (shown in Figure 10e). An example in Babble noise is shown in Figure 11, and the corresponding spectrogram is shown in Figure 12.

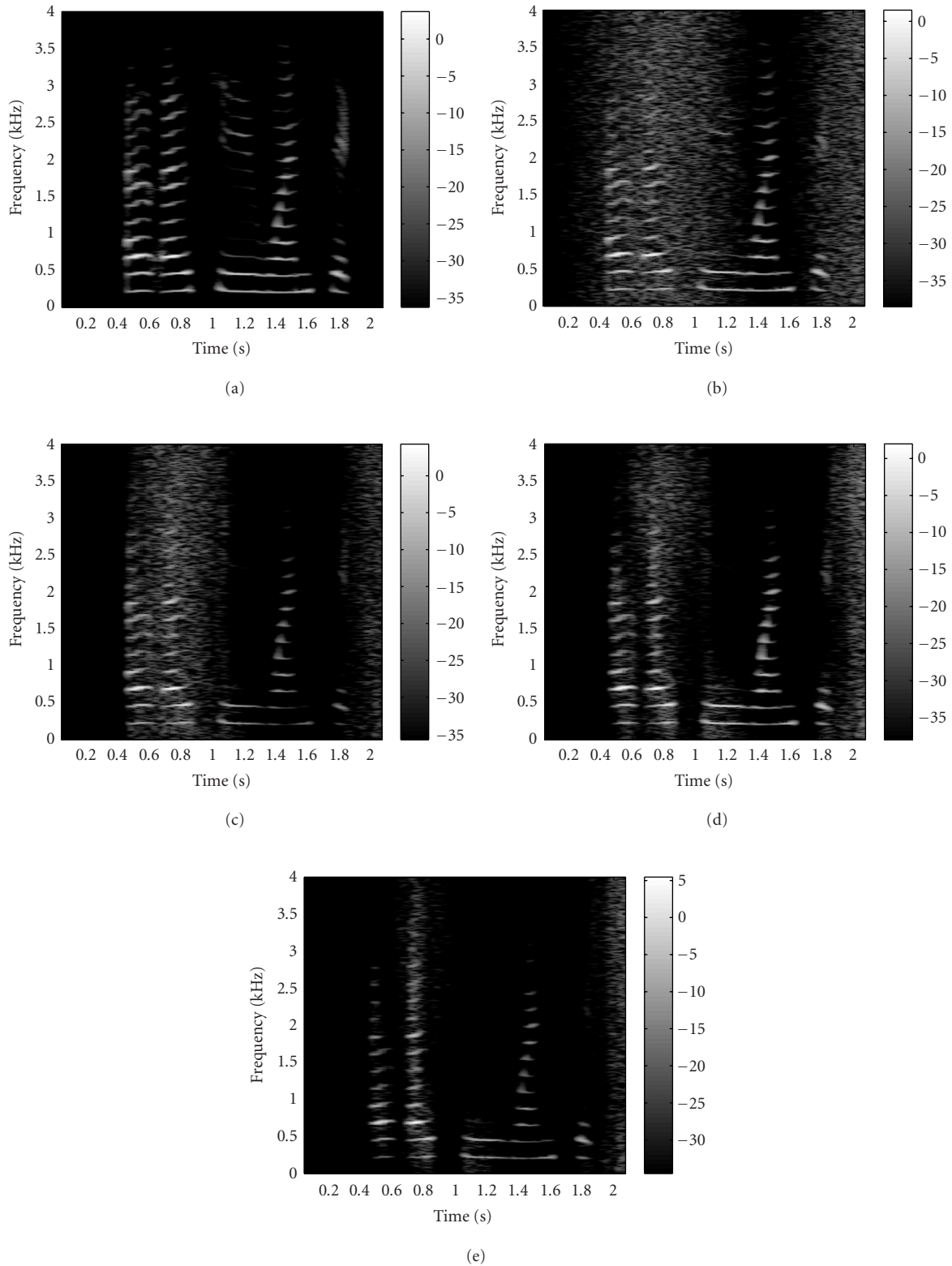


FIGURE 10: An example of the spectrum of the signals (from top to down). (a) Spectrogram of the clean speech signal in English “Oh Oh Two One Six.” (b) Spectrogram of the noisy signal (noise is the simulated nonstationary noise and SNR is -0.2 dB). (c) Spectrogram of the enhanced signal by Wiener filter (system Baseline). (d) Spectrogram of the enhanced signal by Wiener filter with noise parameters estimated by the proposed method (system Known). (e) Spectrogram of the enhanced signal by perceptual filter with noise parameters estimated by the proposed method (system Perceptual).

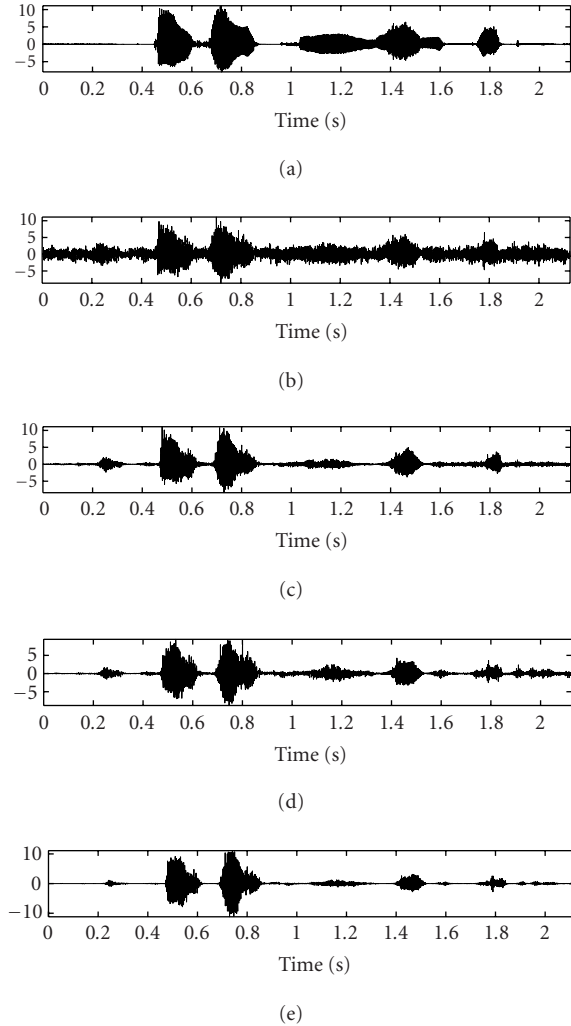


FIGURE 11: An example of signals. (a) Clean speech signal in English “Oh Oh Two One Six.” (b) Noisy signal (noise is babble and SNR is -1.86 dB). (c) Enhanced speech by Wiener filter (system Baseline). (d) Enhanced speech by Wiener filter with noise parameter estimated by the proposed method (system Known). (e) Enhanced speech by perceptual filter with noise parameter estimated by the proposed method (system Perceptual).

However, the nonstationary noise was not perfectly removed in the final part of the sequence in Figure 10. This was in part due to inefficiency in the proposal distribution. Note that the speech states and mixtures were sampled according to the proposal distribution in (25). Thus, at the end of an utterance, the proposed speech states might not yet reach the states of silence. As a result, the speech parameter $\hat{\mu}_{s_t^{(i)}k_t^{(i)}}^{l(i)}(t)$ might still mask (be larger than) the noise parameter $\hat{\mu}_n^l(t)$. In this situation, the noise parameter may not have been updated (remained small if previously estimated noise parameter was smaller than speech parameters $\hat{\mu}_{s_t^{(i)}k_t^{(i)}}^{l(i)}$) because the Kalman gain in EKF was (approaching) zero. Therefore, noise in the final part of the sequence cannot be perfectly removed in some utterances.

Another direction in which the method needs to be improved is obvious in Figure 12. In this figure, high-frequency components are attenuated more than necessary. Since the Mel scale and Bark scale are wider in higher-frequency components than those in the lower-frequency components, noise parameters may not be accurately estimated due to frequency uncertainty between linear frequency and Mel scale (or Bark scale). Constructing speech enhancement algorithms that work directly in linear spectral domain (not Bark-scaled log-spectral domain in this work) may achieve higher frequency resolution and hence better enhancement results.

SNR improvement

The amount of noise reduction is generally measured with the segmental SNR (SegSNR) improvement—the difference between input and output SegSNR:

$$G_{\text{SNR}} = \frac{1}{B} \sum_{b=0}^{B-1} 10 \cdot \log_{10} \frac{(1/D) \sum_{d=0}^{D-1} n^2(d+Db)}{(1/D) \sum_{d=0}^{D-1} [x(d+Db) - \hat{x}(d+Db)]^2}, \quad (44)$$

where B represents the number of frames in the signal. D is the number of observation samples per frame, and it is set to 256.

Figure 13 shows the SegSNR improvement obtained from various noise types and at various noise levels. We can see that the system Known with the sequential Monte Carlo method has improved SegSNR over system Baseline. Figure 13 also shows that both systems Known and Perceptual benefit from the sequential Monte Carlo method. Furthermore, Perceptual shows much greater improvement than Known, which implies that it is effective to employ human auditory properties for speech enhancement.⁹

5. CONCLUSIONS AND DISCUSSIONS

We have presented a sequential Monte Carlo method for a Bayesian estimation of time-varying noise parameters. This method is derived from the general sequential Monte Carlo method for time-varying parameter estimation, but with particular considerations on time-varying noise parameter estimation. The estimated noise parameters are used in a Wiener filter and a perceptual filter for speech enhancement in nonstationary noisy environments. We also demonstrate that, with the estimated noise parameters, a sequential modification of the time-varying mean vector of speech models can improve speech recognition performance in nonstationary noise. The results show that it is a promising approach to handle speech signal processing in nonstationary noise scenarios.

⁹However, as discussed in Section 4.4.4, because the system Perceptual attenuated higher-frequency components more than traditional Wiener filters, the subjective quality of the perceptually enhanced speech signal in human hearing was in fact no better than that by Wiener filters.

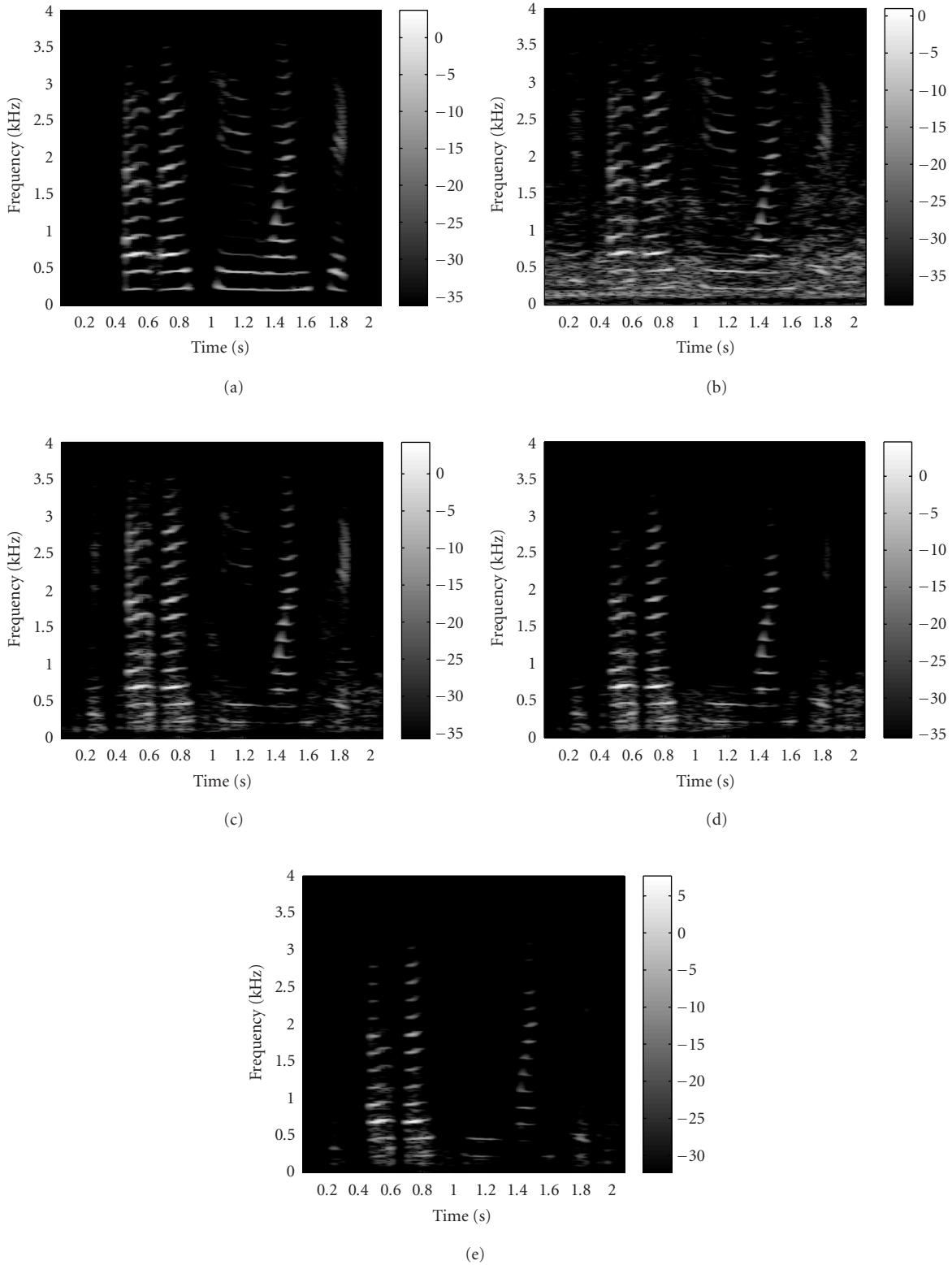
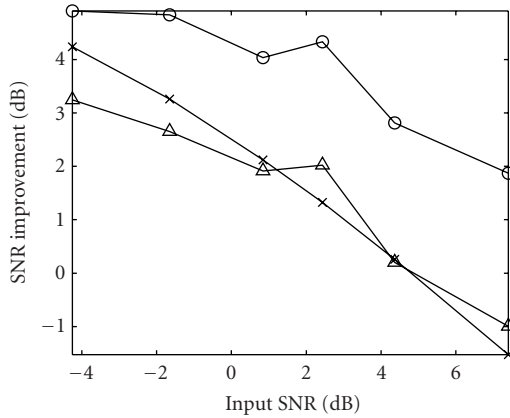
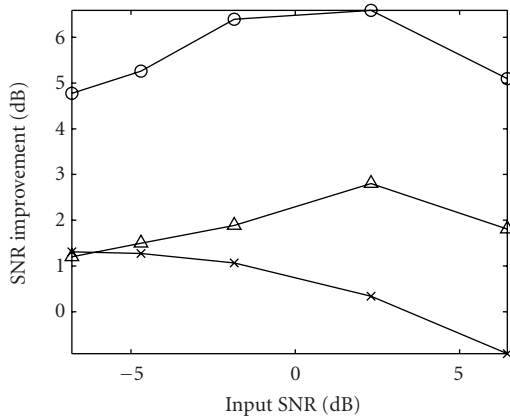


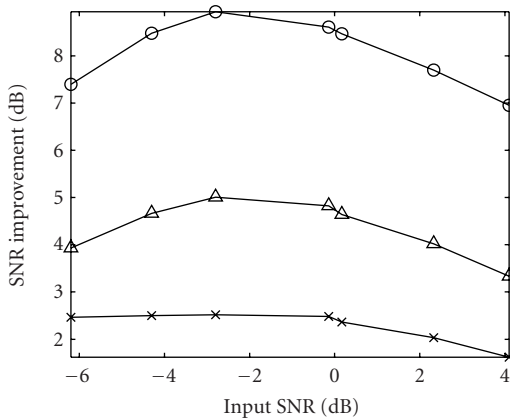
FIGURE 12: An example of the spectrum of the signals (from top to bottom). (a) Spectrogram of the clean speech signal in English “Oh Oh Two One Six.” (b) Spectrogram of the noisy signal (noise is babble and SNR is -1.86 dB). (c) Spectrogram of the enhanced speech by Wiener filter (system Baseline). (d) Spectrogram of the enhanced speech by Wiener filter with noise parameter estimated by the proposed method (system Known). (e) Spectrogram of the enhanced speech by perceptual filter with noise parameter estimated by the proposed method (system Perceptual).



(a)



(b)



(c)

FIGURE 13: Segmental SNR improvement in the following noise: (a) simulated nonstationary noise; (b) babble noise; (c) restaurant noise. The tested systems are the following: (x) Wiener filter (system Baseline); (Δ) Wiener filter with noise parameter estimated by the proposed method (system Known); (o) Perceptual filter with noise parameter estimated by the proposed method (system Perceptual).

The sequential Monte Carlo method in this paper is successfully applied to two seemingly different areas in speech processing, speech enhancement, and speech recognition. This is possible because the graphical model shown in Figure 2 is applicable to the above two areas. The graphical model incorporates two hidden state sequences: one is the speech state sequence for modeling transition of speech units, and the other is a continuous-valued state sequence for modeling noise statistics. With the sequential Monte Carlo method, noise parameter estimation can be conducted via sampling the speech state sequences and updating continuous-valued noise states with $2N$ EKFs at each time. The highly parallel scheme of the method allows an efficient parallel implementation.

We are currently considering the following steps for improved performance: (1) making use of more efficient proposal distribution, for example, auxiliary sampling [29], (2) accurate training of speech models, and (3) design of algorithms working directly in linear spectral domain for speech enhancement. Improvements may be achieved if explicit speech modeling, for example, autocorrelation modeling of speech signals [11], pitch model [30], and so forth, can be incorporated in the framework. Because there is non-linear function involved, we also believe that incorporating smoothing techniques recently proposed for nonlinear time series [31] may achieve improved performances.

APPENDICES

A. APPROXIMATION OF THE ENVIRONMENT EFFECTS ON SPEECH FEATURES

Effects of additive noise on speech power at the j th filter bank can be approximated by (2), where $y_j^{\text{lin}}(t)$, $x_j^{\text{lin}}(t)$, and $n_j^{\text{lin}}(t)$ denote noisy speech power, speech power, and additive noise power in filter bank j [8, 9].

In the log-spectral domain, this equation can be written below as

$$\begin{aligned} \log(x_j^{\text{lin}}(t) + n_j^{\text{lin}}(t)) &= \log x_j^{\text{lin}}(t) + \log\left(1 + \frac{n_j^{\text{lin}}(t)}{x_j^{\text{lin}}(t)}\right) \\ &= \log x_j^{\text{lin}}(t) + \log(1 + \exp(\log n_j^{\text{lin}}(t) - \log x_j^{\text{lin}}(t))). \end{aligned} \quad (\text{A.1})$$

Substituting $x_j^l(t) = \log x_j^{\text{lin}}(t)$, $n_j^l(t) = \log n_j^{\text{lin}}(t)$, and $y_j^l(t) = \log y_j^{\text{lin}}(t)$, we have (3).

B. EXTENDED KALMAN FILTER

The prediction likelihood of the EKF is given by [24]

$$\begin{aligned} &p(Y^l(t) | s_t^{(i)}, k_t^{(i)}, \mu_{s_t^{(i)} k_t^{(i)}}^{l(i)}(t), \hat{\mu}_n^{l(i)}(t-1), Y^l(t-1)) \\ &= \int_{\mu_n^{l(i)}(t)} p(Y^l(t) | \theta^{(i)}(t)) p(\mu_n^{l(i)}(t) | \hat{\mu}_n^{l(i)}(t-1)) d\mu_n^{l(i)}(t) \\ &\propto \exp\left(-\frac{1}{2} \alpha^{(i)}(t)^T (C^{(i)}(t) \Sigma_{s_t^{(i)} k_t^{(i)}}^l C^{(i)}(t)^T + V_n^l)^{-1} \alpha^{(i)}(t)\right), \end{aligned} \quad (\text{B.1})$$

where, respectively, the innovation vector $\alpha^{(i)}(t)$, one-step ahead prediction of noise parameter $\hat{\mu}_n^{l(i)}(t|t-1)$, correlation matrix of the error in $\hat{\mu}_n^{l(i)}(t)$, correlation matrix of the error in $\hat{\mu}_n^{l(i)}(t|t-1)$, measurement matrix at time t (obtained by the first-order differentiation of (7) with respect to $\mu_n^{l(i)}(t)$), and gain function $G^{(i)}(t)$ are given as

$$\alpha^{(i)}(t) = Y^l(t) - \mu_{s_t^{(i)}k_t^{(i)}}^{l(i)}(t) - \log\left(1 + \exp\left(\hat{\mu}_n^{l(i)}(t-1) - \mu_{s_t^{(i)}k_t^{(i)}}^{l(i)}(t)\right)\right), \quad (\text{B.2})$$

$$\hat{\mu}_n^{l(i)}(t|t-1) = \hat{\mu}_n^{l(i)}(t-1) + G^{(i)}(t)\alpha^{(i)}(t-1), \quad (\text{B.3})$$

$$K^{(i)}(t) = K^{(i)}(t, t-1) - G^{(i)}(t)C^{(i)}(t)K^{(i)}(t, t-1), \quad (\text{B.4})$$

$$K^{(i)}(t, t-1) = K^{(i)}(t-1) + V_n^l, \quad (\text{B.5})$$

$$C^{(i)}(t) = \frac{\exp\left(\hat{\mu}_n^{l(i)}(t|t-1) - \mu_{s_t^{(i)}k_t^{(i)}}^{l(i)}(t)\right)}{1 + \exp\left(\hat{\mu}_n^{l(i)}(t|t-1) - \mu_{s_t^{(i)}k_t^{(i)}}^{l(i)}(t)\right)}, \quad (\text{B.6})$$

$$G^{(i)}(t) = K^{(i)}(t, t-1)C^{(i)}(t)^T \left[C^{(i)}(t)K^{(i)}(t, t-1)C^{(i)}(t)^T + \sum_{s_t^{(i)}k_t^{(i)}}^l \right]^{-1}. \quad (\text{B.7})$$

ACKNOWLEDGMENTS

The authors thank anonymous reviewers for their helpful comments on this paper, which substantially improved its presentation. The corresponding author thanks Dr. S. Nakamura (ATR SLT) for helpful discussions. Part of the work was performed when K. Yao was with ATR SLT.

REFERENCES

- [1] J. S. Lim, *Speech Enhancement*, Prentice-Hall, Englewood Cliffs, NJ, USA, 1983.
- [2] K. K. Paliwal and A. Basu, "A speech enhancement method based on Kalman filtering," in *Proc. IEEE Int. Conf. Acoustics, Speech, Signal Processing*, vol. 12, pp. 177–180, Dallas, Tex, USA, April 1987.
- [3] J. S. Lim and A. V. Oppenheim, "All-pole modeling of degraded speech," *IEEE Trans. Acoustics, Speech, and Signal Processing*, vol. 26, no. 3, pp. 197–210, 1978.
- [4] Y. Ephraim, "A Bayesian estimation approach for speech enhancement using hidden Markov models," *IEEE Trans. Signal Processing*, vol. 40, no. 4, pp. 725–735, 1992.
- [5] Y. Ephraim and D. Malah, "Speech enhancement using a minimum-mean square error short-time spectral amplitude estimator," *IEEE Trans. Acoustics, Speech, and Signal Processing*, vol. 32, no. 6, pp. 1109–1121, 1984.
- [6] M. G. Hall, A. V. Oppenheim, and A. S. Willsky, "Time-varying parametric modeling of speech," *Signal Processing*, vol. 5, no. 3, pp. 267–285, 1983.
- [7] ETSI, "Speech processing, transmission and quality aspects (STQ); distributed speech recognition; advanced front-end feature extraction algorithm; compression algorithms," Tech. Rep. ETSI ES 202 050, European Telecommunications Standards Institute, Sophia Antipolis, France, 2002.
- [8] M. J. F. Gales and S. J. Young, "Robust speech recognition in additive and convolutional noise using parallel model combination," *Computer Speech and Language*, vol. 9, no. 4, pp. 289–307, 1995.
- [9] A. Acero, *Acoustical and environmental robustness in automatic speech recognition*, Ph.D. thesis, Carnegie Mellon University, Pittsburgh, Pa, USA, September 1990.
- [10] S. V. Vaseghi and B. P. Milner, "Noise compensation methods for hidden Markov model speech recognition in adverse environments," *IEEE Trans. Speech, and Audio Processing*, vol. 5, no. 1, pp. 11–21, 1997.
- [11] J. Vermaak, C. Andrieu, A. Doucet, and S. J. Godsill, "Particle methods for Bayesian modeling and enhancement of speech signals," *IEEE Trans. Speech, and Audio Processing*, vol. 10, no. 3, pp. 173–185, 2002.
- [12] K. Yao and S. Nakamura, "Sequential noise compensation by sequential Monte Carlo method," in *Advances in Neural Information Processing Systems*, vol. 14, pp. 1205–1212, MIT Press, Cambridge, Mass, USA, 2001.
- [13] D. Burshtein and S. Gannot, "Speech enhancement using a mixture-maximum model," *IEEE Trans. Speech, and Audio Processing*, vol. 10, no. 6, pp. 341–351, 2002.
- [14] N. Virag, "Single channel speech enhancement based on masking properties of the human auditory system," *IEEE Trans. Speech, and Audio Processing*, vol. 7, no. 2, pp. 126–137, 1999.
- [15] J. D. Johnston, "Estimation of perceptual entropy using noise masking criteria," in *Proc. IEEE Int. Conf. Acoustics, Speech, Signal Processing*, vol. 5, pp. 2524–2527, New York, NY, USA, April 1988.
- [16] L. Rabiner and B.-H. Juang, *Fundamentals of Speech Recognition*, Prentice-Hall, Englewood Cliffs, NJ, USA, 1993.
- [17] M. I. Jordan, Z. Ghahramani, T. S. Jaakkola, and L. K. Saul, "An introduction to variational methods for graphical models," in *Learning in Graphical Models*, pp. 105–162, Kluwer Academic, Dordrecht, 1998.
- [18] A. J. Viterbi, "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm," *IEEE Transactions on Information Theory*, vol. 13, no. 2, pp. 260–269, 1967.
- [19] W. K. Hastings, "Monte Carlo sampling methods using Markov chains and their applications," *Biometrika*, vol. 57, no. 1, pp. 97–109, 1970.
- [20] G. Casella and C. P. Robert, "Rao-Blackwellisation of sampling schemes," *Biometrika*, vol. 83, no. 1, pp. 81–94, 1996.
- [21] A. Doucet, N. de Freitas, and N. J. Gordon, Eds., *Sequential Monte Carlo in Practice*, Springer-Verlag, New York, NY, USA, 2001.
- [22] J. M. Bernardo and A. F. M. Smith, *Bayesian Theory*, John Wiley & Sons, New York, NY, USA, 1994.
- [23] V. S. Zariwskii, V. B. Svetnik, and L. I. Shimelevich, "Monte-Carlo techniques in problem of optimal information processing," *Automation and Remote Control*, vol. 36, no. 3, pp. 2015–2022, 1975.
- [24] A. Doucet, S. J. Godsill, and C. Andrieu, "On sequential Monte Carlo sampling methods for Bayesian filtering," *Statistics and Computing*, vol. 10, no. 3, pp. 197–208, 2000.
- [25] E. L. Lehmann and G. Casella, *Theory of Point Estimation*, Springer-Verlag, New York, NY, USA, 1998.
- [26] R. Merwe, A. Doucet, N. Freitas, and E. Wan, "The unscented particle filter," Tech. Rep. CUED/F-INFENG/TR. 380, Signal Processing Group, Engineering Department, Cambridge University, Cambridge, UK, 2000.
- [27] B. Efron and R. J. Tibshirani, *An Introduction to the Bootstrap*, Chapman & Hall, New York, NY, USA, 1993.
- [28] J. S. Liu and R. Chen, "Sequential Monte Carlo methods for dynamic systems," *Journal of the American Statistical Association*, vol. 93, no. 443, pp. 1032–1044, 1998.
- [29] M. K. Pitt and N. Shephard, "Filtering via simulation: auxiliary particle filters," *Journal of the American Statistical Association*, vol. 94, no. 446, pp. 590–699, 1999.

- [30] M. Davy and S. J. Godsill, "Bayesian harmonic models for musical pitch estimation and analysis," Tech. Rep. CUED/F-INFENG/TR.431, Signal Processing Group, Engineering Department, Cambridge University, Cambridge, UK, 2002.
- [31] S. J. Godsill, A. Doucet, and M. West, "Monte Carlo smoothing for nonlinear time series," *Journal of the American Statistical Association*, vol. 99, no. 465, pp. 156–168, 2004.

Kaisheng Yao is a Postgraduate Researcher at the Institute for Neural Computation, University of California, San Diego. Dr. Yao received his B. Eng. and M. Eng. in electrical engineering from Huazhong University of Science & Technology (HUST), Wuhan, China, in 1993 and 1996. In 2000, he received his Dr. Eng. degree for work performed jointly at the State Key Laboratory on Microwave and Digital Communications, Department of Electronic Engineering, Tsinghua University, China, and at the Human Language Technology Center (HLTC), Department of Electrical and Electronic Engineering, Hong Kong University of Science and Technology, Hong Kong. From 2000 to 2002, he was a Researcher in Spoken Language Translation Research Laboratories at the Advanced Telecommunications Research Institute International (ATR), Japan. Since August 2002, he has been with the Institute for Neural Computation, University of California, San Diego. His research interests are mainly in speech recognition in noise, acoustic modeling, dynamic Bayesian networks, statistical signal processing, and some general problems in pattern recognition. He has served as a Reviewer of IEEE Transactions on Speech and Audio Processing and ICASSP.



Te-Won Lee is an Associate Research Professor at the Institute for Neural Computation, University of California, San Diego, and a Collaborating Professor in the Biosystems Department at the Korea Advanced Institute of Science and Technology (KAIST). Dr. Lee received his diploma degree in March 1995 and his Ph.D. degree in October 1997 (summa cum laude) in electrical engineering from the University of Technology Berlin. During his studies, he received the Erwin-Stephan Prize for excellent studies from the University of Technology Berlin and the Carl-Ramhauser Prize for excellent dissertations from the DaimlerChrysler Corporation. He was a Max-Planck Institute Fellow from 1995 till 1997 and a Research Associate at the Salk Institute for Biological Studies from 1997 till 1999. Dr. Lee's research interests include machine learning algorithms with applications in signal and image processing. Recently, he has worked on variational Bayesian methods for independent component analysis, algorithms for speech enhancement and recognition, models for computational vision, and classification algorithms for medical informatics.



Particle Filtering Applied to Musical Tempo Tracking

Stephen W. Hainsworth

*Department of Engineering, University of Cambridge, Cambridge CB2 1PZ, UK
Email: swh21@cantab.net*

Malcolm D. Macleod

*QinetiQ, Malvern, WR14 3PS, UK
Email: m.macleod@signal.qinetiq.com*

Received 30 May 2003; Revised 1 May 2004

This paper explores the use of particle filters for beat tracking in musical audio examples. The aim is to estimate the time-varying tempo process and to find the time locations of beats, as defined by human perception. Two alternative algorithms are presented, one which performs Rao-Blackwellisation to produce an almost deterministic formulation while the second is a formulation which models tempo as a Brownian motion process. The algorithms have been tested on a large and varied database of examples and results are comparable with the current state of the art. The deterministic algorithm gives the better performance of the two algorithms.

Keywords and phrases: beat, tracking, particle filters, music.

1. INTRODUCTION

Musical audio analysis has been a growing area for research over the last decade. One of the goals in the area is fully automated transcription of real polyphonic audio signals, though this problem is currently only partially solved. More realistic sub-tasks in the overall problem exist and can be explored with greater success; beat tracking is one of these and has many applications in its own right (automatic accompaniment of solo performances [1], auto-DJs, expressive rhythmic transformations [2], uses in database retrieval [3], meta-data generation [4], etc.).

This paper describes an investigation into beat tracking utilising particle filtering algorithms as a framework for sequential stochastic estimation where the state-space under consideration is a complex one and does not permit a closed form solution.

Historically, a number of methods have been used to attempt solution of the problem, though they can be broadly categorised into a number of distinct methodologies.¹ The oldest approach is to use oscillating filterbanks and to look for the maximum output; Scheirer [7] typifies this approach though Large [8] is another example. Autocorrelative methods have also been tried and Tzanetakis [3] or Foote [9] are

examples, though these tend to only find the average tempo and not the phase (as defined in Section 2) of the beat. Multiple hypothesis approaches (e.g., Goto [10] or Dixon [11]) are very similar to more rigorously probabilistic approaches (Laroche [12] or Raphael [13], for instance) in that they all evaluate the likelihood of a hypothesis set; only the framework varies from case to case. Klapuri [14] also presents a method for beat tracking which takes the approach typified by Scheirer [7] and applies a probabilistic tempo smoothness model to the raw output. This is tested on an extensive database and the results are the current state of the art.

More recently, particle filters have been applied to the problem; Morris and Sethares [15] briefly present an algorithm which extracts features from the signal and then uses these feature vectors to perform sequential estimation, though their implementation is not described. Cemgil [16] also uses a particle filtering method in his comprehensive paper applying Monte Carlo methods to the beat tracking of expressively performed MIDI signals.² This model will be discussed further later, as it shares some aspects with one of the models described in this paper.

The remainder of the paper is organised as follows: Section 2 introduces tempo tracking; Section 3 covers basic

¹A comprehensive literature review can be found in Seppänen [5] or Hainsworth [6].

²MIDI stands for “musical instrument digital interface” and is a language for expressing musical events in binary. In the context described here, the note start times are extracted from the MIDI signal.

particle filtering theory. Sections 4, 5 and 6 discuss onset detection and the two beat tracking models proposed. Results and discussion are presented in Sections 7 and 8, and conclusions in Section 9.

2. TEMPO TRACKING AND BEAT PERCEPTION

So what is beat tracking?³ The least jargon-ridden description is that it is the pulse defined by a human listener tapping in time to music. However, the terms *tempo*, *beat* and *rhythm* need to be defined. The highest level descriptor is the rhythm; this is the full description of every timing relationship inside a piece of music. However, Bilmes [17] breaks this down into four subdivisions: the hierarchical *metrical structure* which describes the idealised timing relationships between musical events (as they might exist in a musical score for instance), *tempo variations* which link these together in a possibly time varying flow, *timing deviations* which are individual timing discrepancies (“swing” is an example of this) and finally *arrhythmic sections*. If one ignores the last of these as fundamentally impossible to analyse meaningfully, the task is to estimate the tempo curve (tempo tracking) and idealised event times quantised to a grid of “score locations,” given an input set of musical changepoint times.

To represent the tempo curve, a frequency and phase is required such that the phase is zero at beat locations. The metrical structure can then be broken down into a set of levels described by Klapuri [14]: the *beat* or *tactus* is the preferred human tapping tempo; the *tatum* is the shortest commonly occurring interval; and the *bar* or *measure* is related to harmonic change and often correlates to the bar line in common score notation of music. It should be noted that the beat often corresponds to the 1/4 note or crotchet in common notation, but this is not always the case: in fast jazz music, the beat is often felt at half this rate; in hymn music, traditional notation often gives the beat two crotchets (i.e., 1/2 note). The moral is that one must be careful about relating perception to musical notation! Figure 1 gives a diagrammatic representation of the beat relationships for a simple example. The beat is subdivided by two to get the tatum and grouped in fours to find the bar. The lowest level shows timing deviations around the fixed metrical grid.

Perception of rhythm by humans has long been an active area of research and there is a large body of literature on the subject. Drake et al. [18] found that humans with no musical training were able to tap along to a musical audio sample “in time with the music,” though trained musicians were able to do this more accurately. Many other studies have been undertaken into perception of simple rhythmic patterns (e.g., Povel and Essens [19]) and various models of beat perception have been proposed (e.g., [20, 21, 22]) from which ideas can be gleaned. However, the models presented in the rest of this paper are not intended as perceptual models or even as perceptually motivated models; they are engineering equiva-

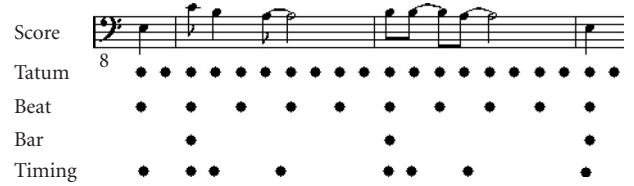


FIGURE 1: Diagram of relationships between metrical levels.

lents of the human perception. Having said that, it is hoped that a successful computer algorithm could help shed light onto potential and as yet unexplained human cognitive processes.

2.1. Problem statement

To summarise, the aim of this investigation is to extract the beat from music as defined by the preferred human tapping tempo; to make the computer tap its hypothetical foot along in time to the music. This requires a tempo process to be explicitly estimated in both frequency and phase, a beat lying where phase is zero. In the process of this, detected “notes” in the audio are assigned “score locations” which is equivalent to quantising them to an underlying, idealised metrical grid. We are not interested in real time implementation nor in *causal* beat tracking where only data up to the currently considered time is used for estimation.

3. PARTICLE FILTERING

Particle filters are a sequential Monte Carlo estimation method which are powerful, versatile and increasingly used in tracking problems. Consider the state space system defined by

$$\mathbf{x}_k = f_k(\mathbf{x}_{k-1}, \xi_k), \quad (1)$$

where $f_k : \mathcal{R}^{n_x} \times \mathcal{R}^{n_\xi} \rightarrow \mathcal{R}^{n_x}$, $k \in \mathbb{N}$, is a possibly nonlinear function of the state \mathbf{x}_{k-1} , dimension n_x and ξ_k which is an i.i.d. noise process of dimension n_ξ . The objective is to estimate \mathbf{x}_k given observations,

$$\mathbf{y}_k = h_k(\mathbf{x}_k, \nu_k), \quad (2)$$

where $h_k : \mathcal{R}^{n_x} \times \mathcal{R}^{n_\nu} \rightarrow \mathcal{R}^{n_y}$ is a separate possibly nonlinear transform and ν_k is a separate i.i.d. noise process of dimension n_ν , describing the observation error.

The posterior of interest is given by $p(\mathbf{x}_{0:k} | \mathbf{y}_{1:k})$ which is represented in particle filters by a set of point estimates or particles $\{\mathbf{x}_{0:k}^{(i)}, w_k^{(i)}\}_{i=1}^N$, where $\{\mathbf{x}_{0:k}^{(i)}, i = 1, \dots, N\}$ is a set of support points with associated weights given by $\{w_k^{(i)}, i = 1, \dots, N\}$. The weights are normalised such that $\sum_{i=1}^N w_k^{(i)} = 1$. The posterior is then approximated by

$$p(\mathbf{x}_{0:k} | \mathbf{y}_{1:k}) \approx \sum_{i=1}^N w_k^{(i)} \delta(\mathbf{x}_{0:k} - \mathbf{x}_{0:k}^{(i)}). \quad (3)$$

³A fuller discussion on this topic can be found in [6].

As $N \rightarrow \infty$, this assumption asymptotically tends to the true posterior. The weights are then selected according to *importance sampling*, $\mathbf{x}_{0:k}^{(i)} \sim \pi(\mathbf{x}_{0:k}^{(i)} | \mathbf{y}_{1:k})$, where $\pi(\cdot)$ is the so-called importance density. The weights are then given by

$$w_k^{(i)} \propto \frac{p(\mathbf{x}_{0:k}^{(i)} | \mathbf{y}_{1:k})}{\pi(\mathbf{x}_{0:k}^{(i)} | \mathbf{y}_{1:k})}. \quad (4)$$

If we restrict ourselves to importance functions of the form,

$$\pi(\mathbf{x}_{0:k} | \mathbf{y}_{1:k}) = \pi(\mathbf{x}_k | \mathbf{x}_{0:k-1}, \mathbf{y}_{1:k}) \pi(\mathbf{x}_{0:k-1} | \mathbf{y}_{1:k-1}), \quad (5)$$

implying a Markov dependency of order 1, the posterior can be factorised to give

$$\begin{aligned} p(\mathbf{x}_{0:k} | \mathbf{y}_{1:k}) &= \frac{p(\mathbf{y}_k | \mathbf{x}_{0:k}, \mathbf{y}_{1:k-1}) p(\mathbf{x}_k | \mathbf{x}_{0:k-1}, \mathbf{y}_{1:k-1})}{p(\mathbf{y}_k | \mathbf{y}_{1:k-1})} \times p(\mathbf{x}_{0:k-1} | \mathbf{y}_{1:k-1}) \\ &\propto p(\mathbf{y}_k | \mathbf{x}_{0:k}, \mathbf{y}_{1:k-1}) p(\mathbf{x}_k | \mathbf{x}_{0:k-1}, \mathbf{y}_{1:k-1}) p(\mathbf{x}_{0:k-1} | \mathbf{y}_{1:k-1}), \end{aligned} \quad (6)$$

which allows sequential update. The weights can then be proven to be updated [23] according to

$$w_k^{(i)} \propto w_{k-1}^{(i)} \frac{p(\mathbf{y}_k | \mathbf{x}_k^{(i)}) p(\mathbf{x}_k^{(i)} | \mathbf{x}_{k-1}^{(i)})}{\pi(\mathbf{x}_k^{(i)} | \mathbf{x}_{0:k-1}^{(i)}, \mathbf{y}_{1:k})} \quad (7)$$

up to a proportionality. Often we are interested in the filtered estimate $p(\mathbf{x}_k | \mathbf{y}_{1:k})$ which can be approximated by

$$p(\mathbf{x}_k | \mathbf{y}_{1:k}) \approx \sum_{i=1}^N w_k^{(i)} \delta(\mathbf{x}_k - \mathbf{x}_k^{(i)}). \quad (8)$$

Particle filters often suffer from degeneracy as all but a small number of weights drop to almost zero, a measure of this being approximated by $\widehat{N}_{\text{eff}} = 1 / \sum_{i=1}^N (w_k^{(i)})^2$ [23]. Good choice of the importance density $\pi(\mathbf{x}_k | \mathbf{x}_{0:k-1}, \mathbf{y}_{1:k})$ can delay this and is crucial to general performance. The introduction of a stochastic jitter into the particle set can also help [24]; however the most common solution is to perform resampling [25] whereby particles with small weights are eliminated and a new sample set $\{\mathbf{x}_k^{(i)*}\}_{i=1}^N$ is generated by resampling N times from the approximate posterior as given by (8) such that $\Pr(\mathbf{x}_k^{(i)*} = \mathbf{x}_k^{(j)}) = w_k^{(j)}$. The new sample set is then more closely distributed according to the true posterior and the weights should be set to $w_k^{(i)} = 1/N$ to reflect this. Further details on particle filtering can be found in [23, 26].

A special case of model is the jump Markov linear systems (JMLS) [27] where the state space, $\mathbf{x}_{0:k}$, can be broken down into $\{\mathbf{r}_{0:k}, \mathbf{z}_{0:k}\}$. $\mathbf{r}_{0:k}$, the jump Markov process, defines a path through a bounded and discrete set of potential states

and conditional upon $\mathbf{r}_{0:k}, \mathbf{z}_{0:k}$ is then defined to be linear Gaussian. The chain rule gives the expansion,

$$p(\mathbf{r}_{0:k}, \mathbf{z}_{0:k} | \mathbf{y}_{1:k}) = p(\mathbf{z}_{0:k} | \mathbf{r}_{0:k}, \mathbf{y}_{1:k}) p(\mathbf{r}_{0:k} | \mathbf{y}_{1:k}), \quad (9)$$

and $p(\mathbf{x}_{0:k} | \mathbf{r}_{0:k}, \mathbf{y}_{1:k})$ is deterministically evaluated via the Kalman filter equations given below in Section 5. After this marginalisation process (called Rao-Blackwellisation [28]), $p(\mathbf{r}_{0:k} | \mathbf{y}_{1:k})$ is then expanded as

$$\begin{aligned} p(\mathbf{r}_{0:k} | \mathbf{y}_{1:k}) &= p(\mathbf{y}_k | \mathbf{r}_{0:k}, \mathbf{y}_{1:k-1}) p(\mathbf{r}_k | \mathbf{r}_{k-1}) \times p(\mathbf{r}_{0:k-1} | \mathbf{y}_{1:k-1}), \end{aligned} \quad (10)$$

with associated (unnormalised) importance weights given by

$$w_k^{(i)} \propto w_{k-1}^{(i)} \frac{p(\mathbf{y}_k | \mathbf{r}_{0:k}^{(i)}, \mathbf{y}_{1:k-1}) p(\mathbf{r}_k^{(i)} | \mathbf{r}_{k-1}^{(i)})}{\pi(\mathbf{r}_k^{(i)} | \mathbf{r}_{0:k-1}^{(i)}, \mathbf{y}_{1:k})}. \quad (11)$$

By splitting the state space up in this way, the dimensionality considered in the particle filter itself is dramatically decreased and the number of particles needed to achieve a given accuracy is also significantly reduced.

4. CHANGE DETECTION

The success of any algorithm is dependent upon the reliability of the data which is provided as an input. Thus, detecting note events in the music for the particle filtering algorithms to track is as important as the actual algorithms themselves. The onset detection falls into two categories; firstly there is detection of transient events which are associated with strong energy changes, epitomised by drum sounds. Secondly, there is detection of harmonic changes without large associated energy changes (e.g., in a string quartet). To implement the first of these, our method approximately follows many algorithms in the literature [7, 11, 12]: frequency bands, f , are separated and an energy evolution envelope $E_f(n)$ formed. A three point linear regression is used to find the gradient of $E_f(n)$ and peaks in this gradient function are detected (equivalent to finding sharp, positive increases in energy which hopefully correspond to the start of notes). Low-energy onsets are ignored and when there are closely spaced pairs of onsets, the lower amplitude one is also discarded. Three frequency bands were used: 20–200 Hz to capture low frequency information; 200 Hz–15 kHz which captures most of the harmonic spectral region; and 15–22 kHz which, contrary to the opinion of Duxbury [29], is generally free from harmonic sounds and therefore clearly shows any transient information.

Harmonic change detection is a harder problem and has received very little attention in the past, though two recent studies have addressed this [29, 30]. To separate harmonics in the frequency domain, long short-time Fourier transform (STFT) windows (4096 samples) with a short hop rate (1/8 frame) were used. As a measure of spectral change from one

frame to the next, a modified Kullback-Liebler distance measure was used:

$$d_n(k) = \log_2 \left(\frac{|X[k, n]|}{|X[k, n-1]|} \right), \quad (12)$$

$$D_{\text{MKL}}(n) = \sum_{k \in \mathcal{K}, d(n) > 0} d_n(k),$$

where $X[k, n]$ is the STFT with time index n and frequency bin k . The modified measure is thus tailored to accentuate positive energy change. \mathcal{K} defines the region 40 Hz–5 kHz where the majority of harmonic energy is to be found and to pick peaks, a local average of the function D_{MKL} was formed and then the maximum picked between each of the crossings of the actual function and the average.

A further discussion of the MKL measure can be found in [31] but a comprehensive analysis is beyond the scope of this paper. For beat tracking purposes, it is desirable to have a low false detection rate, though missed detections are not so important. While no actual rates for false alarms have been determined, the average detected inter-onset interval (IOI) was compared with an estimate given by $T/(N_b \times F)$, where T is the length of the example in seconds, N_b is the number of manually labelled beats and F is the number of tatum in a beat. The detected average IOI was always of the order or larger than the estimate, which shows that under-detection is occurring.

In summary, there are four vectors of onset observations, three from energy change detectors and one from a harmonic change detector. The different detectors may all observe an actual note, or any combination of them might not. In fact, clustering of the onset observations from each of the individual detection functions is performed prior to the start of the particle filtering. A group is formed if any events from different streams fall within 50 ms of each other for transient onsets and 80 ms for harmonic onsets (reflecting the lower time resolution inherent in the harmonic detection process). Inspection of the resulting grouped onsets shows that the inter-group separation is usually significantly more than the within-group time differences. A set of amplitudes is then associated with each onset cluster.

5. BEAT MODEL 1

The model used in this section is loosely based on that of Cemgil et al. [16], designed for MIDI signals. Given the series of onset observations generated as above, the problem is to find a tempo profile which links them together and to assign each observation to a quantised score location.

The system can be represented as a JMLS where conditional on the ‘‘jump’’ parameter, the system is linear Gaussian and the traditional Kalman filter can be used to evaluate the sequence likelihood. The system equations are then

$$\mathbf{x}_k = \Phi_k(\gamma_k)\mathbf{x}_{k-1} + \xi_k, \quad (13)$$

$$\mathbf{y}_k = H_k\mathbf{x}_k + \nu_k, \quad (14)$$

where \mathbf{x}_k is the tempo process at iteration k and can be described as $\mathbf{x}_k = [\rho_k, \Delta_k]^T$. ρ_k is then the predicted time of the k th observation and Δ_k the tempo period, that is, $\Delta_k = 60/T_k$, where T_k is the tempo in beats per minute (bpm). This is equivalent to a constant velocity process and the state innovation, ξ_k is modelled as zero mean Gaussian with covariance Q_k .

To solve the quantisation problem, the score location is encoded as the jump parameter, γ_k , in $\Phi_k(\gamma_k)$. This is equivalent to deciding upon the notation that describes the rhythm of the observed notes. $\Phi_k(\gamma_k)$, is then given by

$$\Phi_k(\gamma_k) = \begin{bmatrix} 1 & \gamma_k \\ 0 & 1 \end{bmatrix}, \quad (15)$$

$$\gamma_k = c_k - c_{k-1}.$$

This associated evolution covariance matrix is [32]

$$Q_k = q \begin{bmatrix} \gamma_k^3 & \gamma_k^2 \\ 3 & 2 \\ \gamma_k^2 & \gamma_k \end{bmatrix}, \quad (16)$$

for a continuous constant velocity process which is observed at discrete time intervals, where q is a scale parameter.

While the state transition matrix is dependent upon γ_k , this is a difference term between two actual locations, c_k and c_{k-1} . It is this process which is important and the prior on c_k becomes a critical issue as it determines the performance characteristics. Cemgil breaks a single beat into subdivisions of two and uses a prior related to the number of significant digits in the binary expansion of the quantised location. Cemgil’s application was in MIDI signals where there is 100% reliability in the data and the onset times are accurate. In audio signals, the event detection process introduces errors both in localisation accuracy and in generating entirely spurious events. Also, Cemgil’s prior cannot cope with triplet figures or swing. Thus, we break the notated beat down into 24 quantised sub-beat locations, $c_k = \{1/24, 2/24, \dots, 24/24, 25/24, \dots\}$ and assign a prior

$$p(c_k) \propto \exp(-\log_2\{d(c_k)\}), \quad (17)$$

where $d(c_k)$ is the denominator of the fraction of c_k when expressed in its most reduced form; that is, $d(3/24) = 8$, $d(36/24) = 2$, and so forth. This prior is motivated by the simple concern of making metrically stronger sub-beat locations more likely; it is a generic prior designed to work with all styles and situations.

Finally, the observation model must be considered. Bearing in mind the pre-processing step of clustering onset observations from different observation function, the input to the particle filter at each step \mathbf{y}_k will be a variable length vector containing between one and four individual onset observation times. Thus, H_k becomes a function of the length j of the observation vector \mathbf{y}_k but is essentially j rows of the form $[1 \ 0]$. The observation error ν_k is also of length j and

is modelled as zero-mean Gaussian with diagonal covariance R_k where the elements r_{jj} are related to whichever observation vector is being considered at $\mathbf{y}_k(j)$.

Thus, conditional upon the c_k process which defines the update rate, everything is modelled as linear Gaussian and the traditional Kalman filter [33] can be used. This is given by the recursion

$$\begin{aligned}\hat{\mathbf{x}}_{k|k-1} &= \Phi_k \hat{\mathbf{x}}_{k-1|k-1}, \\ P(k|k-1) &= \Phi_k P(k-1|k-1) \Phi_k^T + Q_k, \\ K(k) &= P(k|k-1) H_k^T [H_k P(k|k-1) H_k^T + R_k]^{-1}, \\ \hat{\mathbf{x}}_{k|k} &= \hat{\mathbf{x}}_{k|k-1} + K(k) [\mathbf{y}_k - H_k \hat{\mathbf{x}}_{k|k-1}], \\ P(k|k) &= [I - K(k) H_k] P(k|k-1).\end{aligned}\quad (18)$$

Each particle must maintain its own covariance estimate $P(k|k)$ as well as its own state. The innovation or residual vector is defined to be the difference between the measured and predicted quantities,

$$\tilde{\mathbf{y}}_k = \mathbf{y}_k - H_k \hat{\mathbf{x}}_{k|k-1}, \quad (19)$$

and has covariance given by

$$S_k = H_k P_{k|k-1} H_k^T + R_k. \quad (20)$$

5.1. Amplitude modelling

The algorithm as described so far will assign the beat (i.e., the phase of $c_{1:k}$) to the most frequent subdivision, which may not be the right one. To aid the correct determination of phase, attention is turned to the amplitude of the onsets.

The assumption is made that the onsets at some score locations (e.g., on the beat) will have higher energy than others. Each of the three transient onset streams maintains a separate amplitude process while the harmonic onset stream does not have one associated with it due to amplitude not being relevant for this feature.

The amplitude processes can be represented as separate JMLSs conditional upon c_k . The state equations are given by

$$\begin{aligned}\alpha_p^l &= \Theta_p^l \alpha_{p-1}^l + \epsilon_p, \\ a_p^l &= \alpha_p^l + \zeta_p,\end{aligned}\quad (21)$$

where a_p^l is the amplitude of the p th onset from the observation stream, l . Thus, the individual process is maintained for each observation function and updated only when a new observation from that stream is encountered. This requires the introduction of conditioning on p rather than k ; $1:p$ then represents all the indices within the full set $1:k$, where an observation from stream l is found. $\Theta_p^l(c_{p-1}, c_p)$ is a function of c_p and c_{p-1} . To build up the matrix, Θ_p^l , a selection of real data was examined and a 24×24 matrix constructed for the expected amplitude ratio between a pair of score locations. This is then indexed by the currently considered score location c_p and also the previously identified one found in stream l , c_{p-1}^l , and the value given is returned to Θ_p^l . For example, it

could be that the expected amplitude for a beat is modelled as twice that of a quaver off-beat. If the particle history shows that the previous onset from a given stream was assigned to be on the beat and the currently considered location is a quaver, Θ_p^l would equal 0.5. This relative relationship allows the same model to cope with both quiet and loud sections in a piece. The evolution and observation error terms, ϵ_p and ζ_p , are assumed to be zero mean Gaussian with appropriate variances.

From now on, to avoid complicating the notation, the amplitude process will be represented without sums or products over the three l vectors using $a_p = \{a_p^1, a_p^2, a_p^3\}$ and $\alpha_p = \{\alpha_p^1, \alpha_p^2, \alpha_p^3\}$ (noting that some of these might well be given a null value at any given iteration). For each iteration k , between zero and all three of the amplitude processes will be updated.

5.2. Methodology

Given the above system, a particle filtering algorithm can be used to estimate the posterior at any given iteration. The posterior which we wish to estimate is given by $p(c_{1:k}, \mathbf{x}_{1:k}, \alpha_{1:p} | \mathbf{y}_{1:k}, a_{1:p})$ but Rao-Blackwellisation breaks down the posterior into separate terms

$$\begin{aligned}p(c_{1:k}, \mathbf{x}_{1:k}, \alpha_{1:p} | \mathbf{y}_{1:k}, a_{1:p}) \\ = p(\mathbf{x}_{1:k} | c_{1:k}, \mathbf{y}_{1:k}) \\ \times p(\alpha_{1:p} | c_{1:k}, a_{1:p}) p(c_{1:k} | \mathbf{y}_{1:k}, a_{1:p}),\end{aligned}\quad (22)$$

where $p(\mathbf{x}_{1:k} | c_{1:k}, \mathbf{y}_{1:k})$ and $p(\alpha_{1:p} | c_{1:k}, a_{1:p})$ can be deduced exactly by use of the traditional Kalman filter equations. Thus the only space to search over and perform recursion upon is that defined by $p(c_{1:k} | \mathbf{y}_{1:k}, a_{1:p})$. This space is discrete but too large to enumerate all possible paths. Thus we turn to the approximation approach offered by particle filters.

By assuming that the distribution of c_k is dependent only upon $c_{1:k-1}$, $\mathbf{y}_{1:k}$ and $a_{1:p}$, the importance function can be factorised into terms such as $\pi(c_k | \mathbf{y}_{1:k}, a_{1:p}, c_{1:k-1})$. This allows recursion of the Rao-Blackwellised posterior

$$\begin{aligned}p(c_{1:k} | \mathbf{y}_{1:k}, a_{1:p}) \\ \propto p(\mathbf{y}_k, a_p | \mathbf{y}_{1:k-1}, a_{1:p-1}, c_{1:k}) \\ \times p(c_k | c_{k-1}) p(c_{1:k-1} | \mathbf{y}_{1:k-1}, a_{1:p-1}),\end{aligned}\quad (23)$$

where

$$\begin{aligned}p(\mathbf{y}_k, a_p | \mathbf{y}_{1:k-1}, a_{1:p-1}, c_{1:k}) \\ = p(\mathbf{y}_k | \mathbf{y}_{1:k-1}, c_{1:k}) \\ \times p(a_p | a_{1:p-1}, c_{1:k})\end{aligned}\quad (24)$$

and recursive updates to the weight are given by

$$w_k^{(i)} = w_{k-1}^{(i)} \times \frac{p(\mathbf{y}_k | \mathbf{y}_{1:k-1}, c_{1:k}^{(i)}) p(a_p | a_{1:p-1}, c_{1:k}^{(i)}) p(c_k^{(i)} | c_{k-1}^{(i)})}{\pi(c_k^{(i)} | \mathbf{y}_{1:k}, a_{1:p}, c_{1:k-1}^{(i)})}. \quad (25)$$


```

For  $k = 1$ 
  for  $i = 1 : N$ ; draw  $\mathbf{x}_1^{(i)}$ ,  $\alpha_1^{(i)}$  and  $c_1^{(i)}$  from respective priors
for  $k = 2 : \text{end}$ 
  for  $i = 1 : N$ 
    Propagate particle  $i$  to a set,  $s = \{1, \dots, S\}$  of new
    locations  $c_k^{(s)}$ .
    Evaluate the new weight  $w_k^{(s,i)}$  for each of these by
    propagating through the respective Kalman filter.
    This generates  $\pi(c_k | \mathbf{y}_{1:k}, a_{1:p}, c_{1:k-1}^{(i)})$ .
  for  $i = 1 : N$ 
    Pick a new state for each particle from
     $\pi(c_k | \mathbf{y}_{1:k}, a_{1:p}, c_{1:k-1}^{(i)})$ .
    Update weights according to (25).

```

ALGORITHM 1: Rao-Blackwellised particle filter.

The terms $p(\mathbf{y}_k | \mathbf{y}_{1:k-1}, c_{1:k})$ and $p(a_p | a_{1:p-1}, c_{1:k})$ are calculated from the innovation vector and covariance of the respective Kalman filters (see (19) and (20)). $p(c_k | c_{k-1})$ is simplified to $p(c_k)$ and is hence the prior on score location as given in Section 5.

5.3. Algorithm

The algorithm therefore proceeds as given in Algorithm 1. At each iteration, each particle is propagated to a set S of new score locations and the probability of each is evaluated. Given the $N \times S$ set of potential states there are then two ways of choosing a new set of updated particles: either stochastic or deterministic selection. The first proceeds in a similar manner to that described by Cemgil [16] where for each particle the new state is picked from the importance function with a given probability. Deterministic selection simply takes the best N particles from the whole set of propagated particles. Fully stochastic resampling selection of the particles is not an optimal procedure in this case, as duplication of particles is wasteful. This leaves a choice between Cemgil's method of stochastically selecting one of the update proposals for each particle or the deterministic N -best approach. The latter has been adopted as intuitively sensible.

Particle filters suffer from degeneracy in that the posterior will eventually be represented by a single particle with high weight while many particles have negligible probability mass. Traditional PFs overcome this with resampling (see [23]) but both methods for particle selection in the previous section implicitly include resampling. However, degeneracy still exists, in that the PF will tend to converge to a single c_k state, so a number of methods were explored for increasing the diversity of the particles. Firstly, jitter [24] was added to the tempo process to increase local diversity. Secondly, a Metropolis-Hastings (MH) step [34] was used to explore jumps to alternative phases of the signal (i.e., to jump from tracking off-beat quavers to being on the beat). Also, an MH step to propose related tempos (i.e., doubling or halving the tracked tempo) was investigated but found to be counterproductive.

6. BEAT MODEL 2

The model described above formulates beat location as the free variable and time as a dependent, non-continuous variable, which seems counter-intuitive. Noting that the model is bilinear, a reformulation of the tempo process is thus presented now where time is the independent variable and tempo is modelled as Brownian motion⁴ [35]. The state vector is now given by $\mathbf{z}_k = [\tau_k, \dot{\tau}_k]^T$ where τ_k is in beats and $\dot{\tau}_k$ is in beats per second (obviously related to bpm). Brownian motion, which is a limiting form of the random walk, is related to the tempo process by

$$d\dot{\tau}(t) = \sqrt{q}d\mathcal{B}(t) + \dot{\tau}(0), \quad (26)$$

where q controls the variance of the Brownian motion process $\mathcal{B}(t)$ (which is loosely the integral of a Gaussian noise process [32]) and hence the state evolution. This leads to

$$\tau(t) = \tau(0) + \int_0^t \dot{\tau}(s)ds. \quad (27)$$

Time t is now a continuous variable and hence $\tau(t)$ is also a continuously varying parameter, though only being "read" at algorithmic iterations k thus giving $\tau_k \triangleq \tau(t_k)$.

The new state equations are given by

$$\mathbf{z}_k = \Xi(\delta_k)\mathbf{z}_{k-1} + \beta_k, \quad (28)$$

$$\mathbf{y}_k = \Gamma_k t_k + \kappa_k, \quad (29)$$

where

$$t_k = t_0 + \sum_{j=1}^k \delta_j. \quad (30)$$

t_k is therefore the absolute time of an observation and δ_k is the inter-observation time. $\Xi(\delta_k)$ is the state update matrix and is given by

$$\Xi(\delta_k) = \begin{bmatrix} 1 & \delta_k \\ 0 & 1 \end{bmatrix}. \quad (31)$$

Γ_k acts in a similar manner to H_k in model one and is of variable length but is a vector of ones of the same length as \mathbf{y}_k . κ_k is modelled as zero mean Gaussian with covariance R_k as described above. β_k is modelled as zero mean Gaussian noise with covariance given as before by Bar-Shalom [32],

$$Q_k = q \begin{bmatrix} \frac{\delta_k^3}{3} & \frac{\delta_k^2}{2} \\ \frac{\delta_k^2}{2} & \delta_k \end{bmatrix}. \quad (32)$$

One of the problems associated with Brownian motion is that there is no simple, closed form solution for the prediction density, $p(t_k | \cdot)$. Thus attention is turned to

⁴Also termed as Wiener or Wiener-Levy process.

```

Initialise:  $i = 1$ ;  $\mathbf{z}_1 = \mathbf{z}_k$ ;  $X_k$  is the predicted inter-onset
number of beats.
While  $dt > \text{tol}$ ,
   $i = i + 1$ 
  If  $\max(\tau_{1:i}) < X_k$ 
     $dt = (\tau_{i-1} - X_k) / \hat{\tau}_{i-1}$ 
    Draw  $\mathbf{z}_i \sim \mathcal{N}(\Xi_i \mathbf{z}_{i-1}, Q_i)$ 
     $t_i = t_{i-1} + dt$ 
  Else interpolate back
    Find  $I$  s.t.  $\tau_I < X_k$  and  $\tau_{I+1} > X_k$ 
     $t_e = t_I + (t_{I+1} - t_I) \times (X_k - \tau_I) / (\tau_{I+1} - \tau_I)$ 
    insert state  $J$  between  $I$  and  $I + 1$ 
     $t_j = t_e$ 
     $dt = \min(t_{I+1} - t_e, t_e - t_I)$ 
    Draw  $\mathbf{z}_j \sim \mathcal{N}(m, Q')$  where  $m$  and  $Q'$  are
    given below
  Index  $q = \min |(\tau_{1:i} - X_k)|$ .
Return  $\tau_k = X_k$ ,  $t_k = t_q$  and  $\hat{\tau}_k = \hat{\tau}_q$ .

```

ALGORITHM 2: Sample hitting time.

an alternative method for drawing a *hitting time* sample of $\{t_k | \mathbf{z}_{k-1}, \tau_k = B, t_{k-1}\}$. This is an iterative process and, conditional upon initial conditions, a linear prediction for the time of the new beat is made. The system is then stochastically propagated for this length of time and a new tempo and beat position found. The beat position might under or overshoot the intended location. If it undershoots, the above process is repeated. If it overshoots, then an interpolation estimate is made conditional upon both the previous and subsequent data estimates. The iteration terminates when the error on τ_i falls below a threshold. At this point, the algorithm returns the hitting time t_k and the new tempo $\hat{\tau}_k$ at that hitting time. This is laid out explicitly in Algorithm 2, where Ξ_i is given by

$$\Xi_i = \begin{bmatrix} 1 & dt \\ 0 & 1 \end{bmatrix} \quad (33)$$

and Q_i by

$$Q_i = q \begin{bmatrix} \frac{dt^3}{3} & \frac{dt^2}{2} \\ \frac{dt^2}{2} & dt \end{bmatrix}. \quad (34)$$

\mathcal{N} denotes the Gaussian distribution. The interpolation mean and covariance are given by [36]

$$\begin{aligned} Q' &= (Q_{I:j}^{-1} + \Xi_{j:I+1} Q_{j:I+1}^{-1} \Xi_{j:I+1})^{-1}, \\ m &= Q' (Q_{I:j}^{-1} \Xi_{I:j} \mathbf{z}_I + \Xi_{j:I+1}^T Q_{j:I+1}^{-1} \mathbf{z}_{I+1}), \end{aligned} \quad (35)$$

where the index denotes the use of Ξ and Q from (33) and (34) with appropriate values of dt .

Thus, we now have a method of drawing a time t_k and new tempo $\hat{\tau}_k$ given a previous state \mathbf{z}_{k-1} and proposed new score (beat) location τ_k . The algorithm then proceeds as be-

fore with a particle filter. The posterior can be updated, thus

$$\begin{aligned} p(\mathbf{z}_{1:k}, t_{1:k} | \mathbf{y}_{1:k}) &\propto p(\mathbf{y}_k | \mathbf{z}_{1:k}, t_{1:k}) p(t_k | t_{1:k-1}, \mathbf{z}_{1:k}) p(\mathbf{z}_k | \mathbf{z}_{1:k-1}) \\ &\quad \times p(\mathbf{z}_{1:k-1}, t_{1:k-1} | \mathbf{y}_{1:k-1}), \end{aligned} \quad (36)$$

where $p(\mathbf{z}_k | \mathbf{z}_{1:k-1})$ can be factorised:

$$p(\mathbf{z}_k | \mathbf{z}_{1:k-1}) = p(\tau_k | \mathbf{z}_{k-1}) p(\hat{\tau}_k | \mathbf{z}_{k-1}, \tau_k). \quad (37)$$

Prior importance sampling [23] is used via the hitting time algorithm above to draw samples of $\hat{\tau}_k$ and t_k :

$$\pi(\mathbf{z}_k, t_k | \mathbf{z}_{1:k-1}, t_{1:k-1}, \mathbf{y}_{1:k}) = p(\hat{\tau}_k | \mathbf{z}_{k-1}, \tau_k) p(t_k | t_{1:k-1}, \mathbf{z}_{1:k}). \quad (38)$$

This leads to the weight update being given by

$$w_k^{(i)} = w_{k-1}^{(i)} \times p(\mathbf{y}_k | \mathbf{z}_{1:k}^{(i)}, t_{1:k}^{(i)}) p(\tau_k^{(i)} | \mathbf{z}_{k-1}^{(i)}). \quad (39)$$

As before in Section 5, a single beat is split into 24 subdivisions and a prior set upon these as given above in (17); $p(\tau_k | \mathbf{z}_{k-1})$ again reduces to $p(\tau_k) \equiv p(c_k)$. $p(\mathbf{y}_k | \mathbf{z}_{1:k}^{(i)}, t_{1:k}^{(i)})$ is the likelihood; if κ_k from (29) is modelled in the same way as ν_k from (14) then the likelihood is Gaussian with covariance again given by R_k which is diagonal and of the same dimension, j as the observation vector \mathbf{y}_k . Γ_k is then a $j \times 1$ matrix with all entries being 1.

Also as before, to explore the beat quantisation space $\tau_{1:k}$ effectively, each particle is predicted onward to S new positions for τ_k and therefore again, a set of $N \times S$ potential particles is generated. Deterministic selection in this setting is not appropriate so resampling is used to stochastically select N particles from the $N \times S$ set. This acts instead of the traditional resampling step in selecting high probability particles.

Amplitude modelling is also included in an identical form to that described in Section 5.1 which modifies (39) to

$$w_k^{(i)} = w_{k-1}^{(i)} \times p(\mathbf{y}_k | \mathbf{z}_{1:k}^{(i)}, t_{1:k}^{(i)}) p(a_p | \mathbf{z}_{1:k}^{(i)}, t_{1:k}^{(i)}) p(\tau_k^{(i)} | \mathbf{z}_{k-1}^{(i)}). \quad (40)$$

Also, the MH step described in Section 5.3 to explore different phases of the beat is used again.

7. RESULTS

The algorithms described above in Sections 5 and 6 have been tested on a large database of musical examples drawn from a variety of genres and styles, including rock/pop, dance, classical, folk and jazz. 200 samples, averaging about one minute in length were used and a ‘‘ground truth’’ manually generated for each by recording a trained musician clapping in time to the music.

The aim is to estimate the tempo and quantisation parameters over the whole dataset; in both models, the sequence of filtered estimates is not the best representation of this, due to locally unlikely data. Therefore, because each

TABLE 1: Results for beat tracking algorithms expressed as a total percentage averaged over the whole database.

	Raw		Allowed	
	C-L	TOT	C-L	TOT
Model 1	51.5	58.0	69.2	82.2
Model 2	34.1	38.4	54.4	72.8
Scheirer	26.8	41.9	33.0	53.4

particle maintains its own state history, the maximum a posteriori particle at the final iteration was chosen. The parameter sets used within each algorithm were chosen heuristically; it was deemed impractical to optimise them over the whole database. Various numbers of particles N were tried though results are given below for $N = 200$ and 500 for models one and two, respectively. Above these values, performance continued to increase very slightly, as one would expect, but computational effort also increased proportionally.

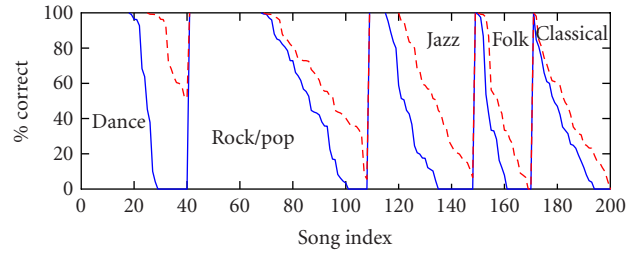
Tracking was deemed to be accurate if the tempo was correct (interbeat interval matches to within 10%) and a beat was located within 15% of the annotated beat location.⁵ Klapuri [14] defines a measure of success as the longest consecutive region of beats tracked correctly as a proportion of the total (denoted “C-L” for consecutive-length). Also presented is a total percentage of correctly tracked beats (labelled “TOT”). The results are presented in Table 1. It was noted that the algorithms sometimes tracked at double or half tempo in psychologically plausible patterns; also, dance music with heavy off-beat accents often caused the algorithm to track 180° out of phase. The “allowed” columns of the table show results accepting these errors. Also shown for comparison are the results obtained using Scheirer’s algorithm [7].

The current state of the art is the algorithm of Klapuri [14] with 69% success for longest consecutive sequence and 78% for total correct percentage (accepting errors) on his test database consisting of over 400 examples. Thus the performance of our algorithm is at least comparable with this.

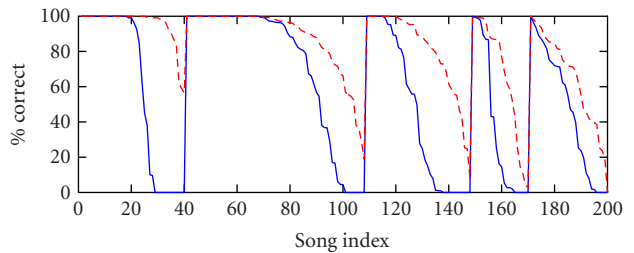
Figure 2 shows the results for model one over the whole database graphically while Figure 3 shows the same for model two. These are ordered by style and then performance within the style category. Figure 4 shows the tempo profile for a correctly tracked example using model one; note the close agreement between the hand labelled data and the tracked tempo.

8. DISCUSSION

The algorithms described above have some similar elements but their fundamental operation is quite different: the Rao-Blackwellised model of Section 5 actually bears a significant resemblance to an interacting multiple models system of the type used in radar tracking [33], as many of the stages are actually deterministic. The second model, however, is much



(a)



(b)

FIGURE 2: Results on test database for model one. The solid line represents raw performance and the dashed line is performance after acceptable tracking errors have been taken into account. (a) Maximum length correct (% of total). (b) Total percentage correct.

more typically a particle filter with mainly stochastic processes. Both have many underlying similarities in the model though the inference processes are significantly different.

Thus, the results highlight some interesting comparisons between these two philosophies. On close examination, model two was better at finding the most likely local path through the data, though this was not necessarily the correct one in the long term. A fundamental weakness of the models is the prior on c_k (or equivalently, τ_k in model two) which intrinsically prefers higher tempos—doubling a given tempo places more onsets in metrically stronger positions which is deemed more likely by the prior given in (17). Because the stochastic resampling step efficiently selects and boosts high probability regions of the posterior, model two would often pick high tempos to track (150–200bpm) which accounts for the very low “raw” results.

A second problem also occurs in model two: because duplication of paths through the $\tau_{1:k}$ space is necessary to fully populate each quantisation hypothesis, fewer distinct paths are kept at each iteration. By comparison, the N -best selection scheme of model one ensures that each particle represents a unique $c_{1:k}$ set and more paths through the state space are kept for a longer lag. This allows model one to recover better from a region of poor data. This also provides an explanation for why model one does not track at high tempo so often—because more paths though the state-space are retained for longer, more time is allowed for the amplitude process to influence the choice of tempo mode. Thus, the conclusion is drawn that the first model is more attractive: the Rao-Blackwellisation of the tempo process allows the search of the quantisation space to be much more effective.

⁵The clapped signals were often slightly in error themselves.

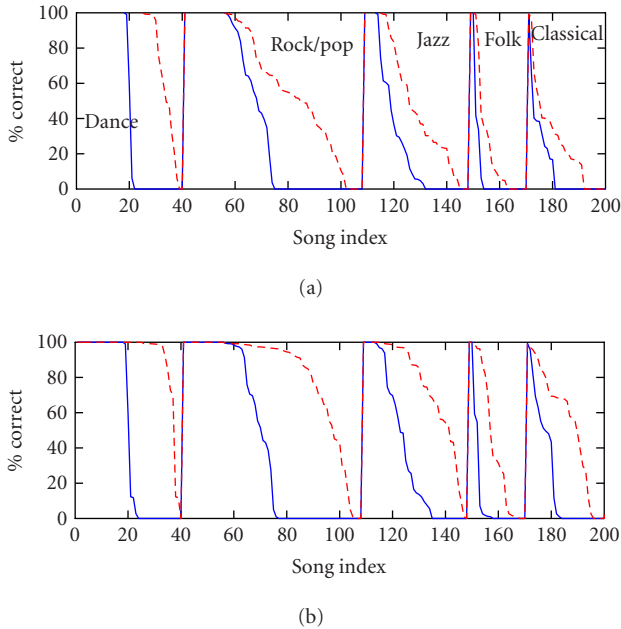


FIGURE 3: Results for model two. (a) Maximum length correct (% of total). (b) Total percentage correct.

The remaining lack of performance can be accredited to four causes: the first is tracking at multiple tempo modes—sometimes tracking fails at one mode and settles a few beats later into a second mode. The results only reflect one of these modes. Secondly, stable tracking sometimes occurs at psychologically implausible modes (e.g., 1.5 times the correct tempo) which are not included in the results above. The third cause is poor onset detection. Finally, there are also examples in the database which exhibit extreme tempo variation which is never followed.

The result of this is a number of suggestions for improvements: firstly, the onset detection is crucial and if the detected onsets are unreliable (especially at the start of an example) it is unlikely that the algorithm will ever be able to track the beat properly. This may suggest an “online” onset detection scheme where the particles propose onsets in the data, rather than the current offline, hard decision system. The other potential scheme for overcoming this would be to propose a salience measure (e.g., [21]) and directly incorporate this into the state evolution process, thus hoping to differentiate between likely and unlikely beat locations in the data; currently, the Rao-Blackwellised amplitude process has been given weak variances and hence has little effect in the algorithm, other than to propose correct phase. The other problems commonly encountered were tempo errors by plausible ratios; Metropolis-Hastings steps [27] to explore other modes of the tempo posterior were tried but have met with little success.

Thus it seems likely that any real further improvement will have to come from music theory incorporated into the algorithm directly, and in a style-specific way—it is unlikely that a beat tracker designed for dance music will work well on choral music! Thus, data expectations and also anti-

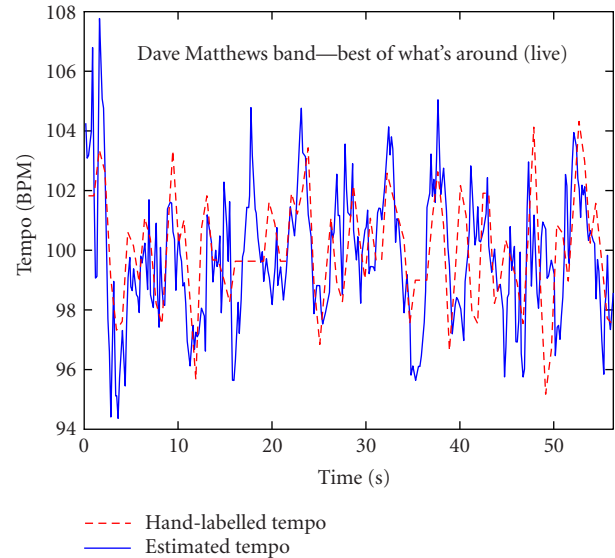


FIGURE 4: Tempo evolution for a correctly tracked example using model one.

patented tempo evolutions and onset locations would have to be worked into the priors in order to select the correct tempo. This will probably result in an algorithm with many ad-hoc features but, given that musicians have spent the better part of 600 years trying to create music which confounds expectation, it is unlikely that a simple, generic model to describe all music will ever be found.

9. CONCLUSIONS

Two algorithms using particle filters for generic beat tracking across a variety of musical styles are presented. One is based upon the Kalman filter and is close to a multiple hypothesis tracker. This performs better than a more stochastic implementation which models tempo as a Brownian motion process. Results with the first model are comparable with the current state of the art [14]. However, the advantage of particle filtering as a framework is that the model and the implementation are separated allowing the easy addition of extra measures to discriminate the correct beat. It is conjectured that further improvement is likely to require music specific knowledge.

ACKNOWLEDGMENTS

This work was partly supported by the research program BLISS (IST-1999-14190) from the European Commission. The first author is grateful to the Japan Society for the Promotion of Science and the Grant-in-Aid for Scientific Research in Japan for their funding. The authors thank P. Comon and C. Jutten for helpful comments and are grateful to the anonymous reviewers for their helpful suggestions which have greatly improved the presentation of this paper.

REFERENCES

- [1] C. Raphael, "A probabilistic expert system for automatic musical accompaniment," *J. Comput. Graph. Statist.*, vol. 10, no. 3, pp. 486–512, 2001.
- [2] F. Gouyon, L. Fabig, and J. Bonada, "Rhythmic expressiveness transformations of audio recordings: swing modifications," in *Proc. Int. Conference on Digital Audio Effects Workshop*, London, UK, September 2003.
- [3] G. Tzanetakis and P. Cook, "Musical genre classification of audio signals," *IEEE Trans. Speech, and Audio Processing*, vol. 10, no. 5, pp. 293–302, 2002.
- [4] E. D. Scheirer, "About this business of metadata," in *Proc. International Symposium on Music Information Retrieval*, pp. 252–254, Paris, France, October 2002.
- [5] J. Seppänen, "Tatum grid analysis of musical signals," in *Proc. IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, pp. 131–134, New Paltz, NY, USA, October 2001.
- [6] S. W. Hainsworth, *Techniques for the automated analysis of musical audio*, Ph.D. thesis, Cambridge University Engineering Department, Cambridge, UK, 2004.
- [7] E. D. Scheirer, "Tempo and beat analysis of acoustical musical signals," *J. Acoust. Soc. Amer.*, vol. 103, no. 1, pp. 588–601, 1998.
- [8] E. W. Large and M. R. Jones, "The dynamics of attending: How we track time varying events," *Psychological Review*, vol. 106, no. 1, pp. 119–159, 1999.
- [9] J. Foote and S. Uchihashi, "The beat spectrum: a new approach to rhythm analysis," in *Proc. IEEE International Conference on Multimedia and Expo*, pp. 881–884, Tokyo, Japan, August 2001.
- [10] M. Goto, "An audio-based real-time beat tracking system for music with or without drum-sounds," *J. of New Music Research*, vol. 30, no. 2, pp. 159–171, 2001.
- [11] S. Dixon, "Automatic extraction of tempo and beat from expressive performances," *J. of New Music Research*, vol. 30, no. 1, pp. 39–58, 2001.
- [12] J. Laroche, "Estimating tempo, swing and beat locations in audio recordings," in *Proc. IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, pp. 135–138, New Paltz, NY, USA, October 2001.
- [13] C. Raphael, "Automated rhythm transcription," in *Proc. International Symposium on Music Information Retrieval*, pp. 99–107, Bloomington, Ind, USA, October 2001.
- [14] A. Klapuri, "Musical meter estimation and music transcription," in *Proc. Cambridge Music Processing Colloquium*, pp. 40–45, Cambridge University, UK, March 2003.
- [15] R. D. Morris and W. A. Sethares, "Beat tracking," in *7th Valencia International Meeting on Bayesian Statistics*, Tenerife, Spain, June 2002, personal communication with R. Morris.
- [16] A. T. Cemgil and B. Kappen, "Monte Carlo methods for tempo tracking and rhythm quantization," *J. Artificial Intelligence Research*, vol. 18, no. 1, pp. 45–81, 2003.
- [17] J. A. Bilmes, "Timing is of the essence: perceptual and computational techniques for representing, learning and reproducing expressive timing in percussive rhythm," M.S. thesis, Media Lab, MIT, Cambridge, Mass, USA, 1993.
- [18] C. Drake, A. Penel, and E. Bigand, "Tapping in time with mechanical and expressively performed music," *Music Perception*, vol. 18, no. 1, pp. 1–23, 2000.
- [19] D.-J. Povel and P. Essens, "Perception of musical patterns," *Music Perception*, vol. 2, no. 4, pp. 411–440, 1985.
- [20] H. C. Longuet-Higgins and C. S. Lee, "The perception of musical rhythms," *Perception*, vol. 11, no. 2, pp. 115–128, 1982.
- [21] R. Parncutt, "A perceptual model of pulse salience and metrical accent in musical rhythms," *Music Perception*, vol. 11, no. 4, pp. 409–464, 1994.
- [22] M. J. Steedman, "The perception of musical rhythm and metre," *Perception*, vol. 6, no. 5, pp. 555–569, 1977.
- [23] A. Doucet, S. Godsill, and C. Andrieu, "On sequential Monte Carlo sampling methods for Bayesian filtering," *Statistics and Computing*, vol. 10, no. 3, pp. 197–208, 2000.
- [24] N. J. Gordon, D. J. Salmond, and A. F. M. Smith, "Novel approach to nonlinear/non-Gaussian Bayesian state estimation," *IEE Proceedings Part F: Radar and Signal Processing*, vol. 140, no. 2, pp. 107–113, 1993.
- [25] A. F. M. Smith and A. E. Gelfand, "Bayesian statistics without tears: a sampling-resampling perspective," *Amer. Statist.*, vol. 46, no. 2, pp. 84–88, 1992.
- [26] M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking," *IEEE Trans. Signal Processing*, vol. 50, no. 2, pp. 174–188, 2002.
- [27] A. Doucet, N. J. Gordon, and V. Krishnamurthy, "Particle filters for state estimation of jump Markov linear systems," *IEEE Trans. Signal Processing*, vol. 49, no. 3, pp. 613–624, 2001.
- [28] G. Casella and C. P. Robert, "Rao-Blackwellisation of sampling schemes," *Biometrika*, vol. 83, no. 1, pp. 81–94, 1996.
- [29] C. Duxbury, M. Sandler, and M. Davies, "A hybrid approach to musical note detection," in *Proc. 5th Int. Conference on Digital Audio Effects Workshop*, pp. 33–38, Hamburg, Germany, September 2002.
- [30] S. Abdallah and M. Plumbley, "Unsupervised onset detection: a probabilistic approach using ICA and a hidden Markov classifier," in *Proc. Cambridge Music Processing Colloquium*, Cambridge, UK, March 2003.
- [31] S. W. Hainsworth and M. D. Macleod, "Onset detection in musical audio signals," in *Proc. International Computer Music Conference*, pp. 163–166, Singapore, September–October 2003.
- [32] Y. Bar-Shalom and T. E. Fortmann, *Tracking and Data Association*, vol. 179 of *Mathematics in Science and Engineering*, Academic Press, Boston, Mass, USA, 1988.
- [33] S. S. Blackman and R. Popoli, *Design and Analysis of Modern Tracking Systems*, Artech House, Norwood, Mass, USA, 1999.
- [34] W. R. Gilks, S. Richardson, and D. J. Spiegelhalter, Eds., *Markov chain Monte Carlo in practice*, Chapman & Hall, London, UK, 1996.
- [35] B. Øksendal, *Stochastic Differential Equations*, Springer-Verlag, New York, NY, USA, 3rd edition, 1992.
- [36] M. Orton and A. Marrs, "Incorporation of out-of-sequence measurements in non-linear dynamic systems using particle filters," Tech. Rep., Cambridge University Engineering Department, Cambridge, UK, 2001.

Stephen W. Hainsworth was born in 1978 in Stratford-upon-Avon, England. During 8 years at the University of Cambridge, he was awarded the B.A. and M.Eng. degrees in 2000 and the Ph.D. in 2004, with the latter concentrating on techniques for the automated analysis of musical audio. Since graduating for the third time, he has been working in London for Tillinghast-Towers Perrin, an actuarial consultancy.



Malcolm D. Macleod was born in 1953 in Cathcart, Glasgow, Scotland. He received the B.A. degree in 1974, and Ph.D. on discrete optimisation of DSP systems in 1979, from the University of Cambridge. From 1978 to 1988 he worked for Cambridge Consultants Ltd, on a wide range of signal processing, electronics, and software research and development projects. From 1988 to 1995 he was a Lecturer in the Signal Processing and Communications Group, the Engineering Department of Cambridge University, and from 1995 to 2002 he was the Department's Director of Research. In November 2002 he joined the Advanced Signal Processing Group at QinetiQ, Malvern, as a Senior Research Scientist. He has published many papers in the fields of digital filter design, nonlinear filtering, adaptive filtering, efficient implementation of DSP systems, optimal detection, high-resolution spectrum estimation and beamforming, image processing, and applications in sonar, instrumentation, and communication systems.

