Gaussian Sum Particle Filtering

Jayesh H. Kotecha and Petar M. Djurić, *Senior Member, IEEE*

***Abstract*—In this paper, we use the Gaussian particle filter introduced in a companion paper to build several types of Gaussian sum particle filters. These filters approximate the filtering and predictive distributions by weighted Gaussian mixtures and are basically banks of Gaussian particle filters. Then, we extend the use of Gaussian particle filters and Gaussian sum particle filters to dynamic state space (DSS) models with non-Gaussian noise. With non-Gaussian noise approximated by Gaussian mixtures, the non-Gaussian noise models are approximated by banks of Gaussian noise models, and Gaussian mixture filters are developed using algorithms developed for Gaussian noise DSS models.[1] As a result, problems involving heavy-tailed densities can be conveniently addressed. Simulations are presented to exhibit the application of the framework developed herein, and the performance of the algorithms is examined.**

***Index Terms*—Dynamic state-space models, extended Kalman filter, Gaussian mixture, Gaussian particle filter, Gaussian sum filter, Gaussian sum particle filter, Monte Carlo filters, nonlinear non-Gaussian stochastic systems, particle filters, sequential Bayesian estimation, sequential sampling methods.**

I. Introduction

In [1], we introduced the Gaussian particle filter (GPF), which is used for tracking filtering and predictive distributions encountered in dynamic state-space models (DSS). The models there are characterized with additive Gaussian noises, but the functions that appear in the process and observation equations are nonlinear functions. The underlying assumption in that paper is that the predictive and filtering distributions can be approximated as Gaussians. Unlike the extended Kalman filter (EKF), which also assumes that these distributions are Gaussians and employs linearization of the functions in the process and observation equations, the GPF updates the Gaussian approximations by using particles that are propagated through the process and observation equations without approximations.

In this paper, we introduce three types of particle filters that are built from banks of particle filters. They approximate the predictive and filtering distributions as Gaussian mixtures (GMs). We refer to them as Gaussian sum particle filters (GSPFs), and a quick roadmap that describes them together with the GPF is given in Table I.

TABLE I
QUICK ROADMAP OF GPF AND GSPFS

Filter	DSS model	Densities approximated as
GPF	nonlinear, general noise	Gaussian
GSPF-I	nonlinear, additive Gaussian noise	Gaussian mixture
GSPF-II and III	nonlinear, additive non-Gaussian noise	Gaussian mixture

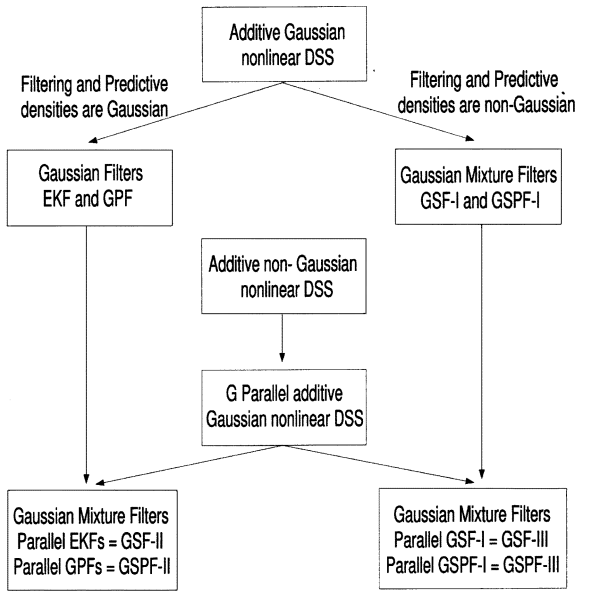

Fig. 1. Roadmap for GFs and GMFs.

We extend the use of the new filters to encompass nonlinear and additive *non-Gaussian* noise DSS models. Gaussian mixture models are increasingly used for modeling non-Gaussian densities [4]–[6]. We work with non-Gaussian noise that is modeled by a GM for a nonlinear DSS model [7], [8]. It is shown that for Bayesian inference, the nonlinear *non-Gaussian* DSS model can be modeled as a bank of parallel nonlinear *Gaussian* noise DSS models. Based on the GPF and GSPFs, we develop Gaussian mixture filters (GMFs) for nonlinear non-Gaussian models. A roadmap of all the filters is given in Fig. 1. We elaborate on various specifics of these filters and provide numerical results based on simulations.

Before we proceed, we briefly introduce the notation and a list of abbreviations used throughout the paper. The signal of interest is $\{\mathbf{x}_n;\ n \in \mathbb{N}\}, \mathbf{x} \in \mathbb{R}^{m_x}$, and it represents an unobserved Markov process with distribution $p(\mathbf{x}_n \mid \mathbf{x}_{n-1})$. Its initial distribution is given by $p(\mathbf{x}_0)$. The observations $\{\mathbf{y}_n;\ n \in$

Manuscript received July 5, 2001; revised March 4, 2003. This work was supported by the National Science Foundation under Awards CCR-0082607 and CCR-9903120. The associate editor coordinating the review of this paper and approving it for publication was Prof. Zhi Ding.

J. H. Kotecha is with the Department of Electrical and Computer Engineering, University of Wisconsin at Madison, Madison, WI 53705 USA (e-mail: jkotecha@ece.wisc.edu).

P. M. Djurić is with the Department of Electrical and Computer Engineering, State University of New York at Stony Brook, Stony Brook, NY 11794 USA (e-mail: djuric@ece.sunysb.edu).

Digital Object Identifier 10.1109/TSP.2003.816754

[1]Part of this work has appeared in [2] and [3].

$\mathbb{N}\}, \mathbf{y} \in \mathbb{R}^{m_y}$ are conditionally independent given $\{\mathbf{x}_n;\ n \in \mathbb{N}\}$ and are represented by the distribution $p(\mathbf{y}_n \mid \mathbf{x}_n)$. The model that we address is given by

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

where $\mathbf{u}_n$ and $\mathbf{v}_n$ are non-Gaussian noises. The signal and observations up to time n are denoted by $\mathbf{x}_{0:n}$ and $\mathbf{y}_{0:n}$, respectively, where $\mathbf{x}_{0:n} \equiv \{\mathbf{x}_0, \ldots, \mathbf{x}_n\}$ and $\mathbf{y}_{0:n} \equiv \{\mathbf{y}_0, \ldots, \mathbf{y}_n\}$. As before, our objective is to estimate the filtering distribution $p(\mathbf{x}_n \mid \mathbf{y}_{0:n})$ and the predictive distribution $p(\mathbf{x}_{n+1} \mid \mathbf{y}_{0:n})$ recursively in time.

The list of abbreviations is as follows.

List of Abbreviations:

BOT	Bearings-only tracking.
DSS	Dynamic state space.
EKF	Extended Kalman filter.
EM	Expectation-maximization.
GS	Gaussian sum.
GM	Gaussian mixture.
GMF	Gaussian mixture filter.
GMM	Gaussian mixture model.
GPF	Gaussian particle filter.
GSF	Gaussian sum filter.
GSPF	Gaussian sum particle filter.
MMSE	Minimum mean square error.
MSE	Mean square error.
SIS	Sequential importance sampling.
SISR	Sequential importance sampling with resampling.
UKF	Unscented Kalman filter.
VLSI	Very large scale integration.

II. Gaussian Mixture Filters for Gaussian Noise

In this section, we assume that the noise processes are additive Gaussian. This assumption is relaxed in the following section.

A. Gaussian Sum Filtering—I

First, we briefly review the theory of Gaussian sum filters. For the DSS model (1), assume that the distribution $p(\mathbf{x}_0)$ is expressed as a Gaussian mixture. It is given that we would like to obtain the filtering and predictive distributions recursively approximated as Gaussian mixtures.

1) Measurement Update: Assume that at time n, we have the predictive distribution

$$p(\mathbf{x}_n \mid \mathbf{y}_{0:n-1}) = \sum_{i=1}^{G} \bar{w}_{ni} \mathcal{N}\left(\mathbf{x}_n; \bar{\boldsymbol{\mu}}_{ni}, \bar{\boldsymbol{\Sigma}}_{ni}\right). \tag{2}$$

After receiving the nth observation $\mathbf{y}_n$, we obtain the filtering distribution from the predictive distribution according to

$$p(\mathbf{x}_n \mid \mathbf{y}_{0:n}) = C_n p(\mathbf{y}_n \mid \mathbf{x}_n)\, p(\mathbf{x}_n \mid \mathbf{y}_{0:n-1})$$

where C_n is a normalizing constant. Using (2), we can write

$$p(\mathbf{x}_n \mid \mathbf{y}_{0:n}) = C_n \sum_{i=1}^{G} \bar{w}_{ni} p(\mathbf{y}_n \mid \mathbf{x}_n)\, \mathcal{N}\left(\mathbf{x}_n; \bar{\boldsymbol{\mu}}_{ni}, \bar{\boldsymbol{\Sigma}}_{ni}\right). \tag{3}$$

We recall a theorem from [9].

Theorem 1: For an additive Gaussian noise model in the observation equation in (1), i.e., $\mathbf{y}_n = \mathbf{h}(\mathbf{x}_n) + \mathbf{v}_n$, where $\mathbf{v}_n \sim \mathcal{N}(0, \mathbf{R}_n)$, $\forall n$ and $p(\mathbf{x}_n \mid \mathbf{y}_{0:n-1})$ given by (2), the distribution $p(\mathbf{x}_n \mid \mathbf{y}_{0:n})$ approaches the Gaussian sum

$$\sum_{i=1}^{G} w_{ni} \mathcal{N}(\mathbf{x}_n; \boldsymbol{\mu}_{ni}, \boldsymbol{\Sigma}_{ni}) \tag{4}$$

uniformly in $\mathbf{x}_n$ and $\mathbf{y}_n$ as $\bar{\boldsymbol{\Sigma}}_{ni} \to 0$ for $i = 1, \ldots, G$, where $\boldsymbol{\mu}_{ni}$ and $\boldsymbol{\Sigma}_{ni}$ are calculated using the following equations:

$$\begin{aligned}\boldsymbol{\mu}_{ni} &= \bar{\boldsymbol{\mu}}_{ni} + \mathbf{K}_{ni}(\mathbf{y}_n - \mathbf{h}(\mathbf{x}_n)) \\ \boldsymbol{\Sigma}_{ni} &= \bar{\boldsymbol{\Sigma}}_{ni} - \mathbf{K}_{ni}\mathbf{H}_{ni}\bar{\boldsymbol{\Sigma}}_{ni} \\ \mathbf{K}_{ni} &= \bar{\boldsymbol{\Sigma}}_{ni}\mathbf{H}_{ni}\left(\mathbf{H}_{ni}^T\bar{\boldsymbol{\Sigma}}_{ni}\mathbf{H}_{ni} + \mathbf{R}_n\right)^{-1} \\ \mathbf{H}_{ni} &= \frac{\partial \mathbf{h}_n(\mathbf{x})}{\partial \mathbf{x}}\Big|_{\mathbf{x}=\bar{\boldsymbol{\mu}}_{ni}} \\ w_{ni} &= \frac{\bar{w}_{ni}\alpha_{ni}}{\sum_{j=1}^{G}\bar{w}_{ni}\alpha_{ni}}\end{aligned} \tag{5}$$

where

$$\alpha_{ni} = \mathcal{N}(\mathbf{y}_n; \mathbf{h}(\bar{\boldsymbol{\mu}}_{ni}), \mathbf{H}_{ni}^T\bar{\boldsymbol{\Sigma}}_{ni}\mathbf{H}_{ni} + \mathbf{R}_n). \tag{6}$$

Proof: See [9 pp. 214 and 215]. ■

2) Time Update: With $p(\mathbf{x}_n \mid \mathbf{y}_{0:n})$ expressed as a Gaussian mixture, we would like to obtain the predictive distribution $p(\mathbf{x}_{n+1} \mid \mathbf{y}_{0:n})$ and approximate it also as a Gaussian mixture. This can be done according to

$$\begin{aligned}p(\mathbf{x}_{n+1} \mid \mathbf{y}_{0:n}) &= \int p(\mathbf{x}_{n+1} \mid \mathbf{x}_n) p(\mathbf{x}_n \mid \mathbf{y}_{0:n}) d\mathbf{x}_n \\ &\approx \int \sum_{i=1}^{G} w_{ni}\mathcal{N}(\mathbf{x}_n; \boldsymbol{\mu}_{ni}, \boldsymbol{\Sigma}_{ni}) p(\mathbf{x}_{n+1} \mid \mathbf{x}_n) d\mathbf{x}_n \\ &= \sum_{i=1}^{G} w_{ni} \int \mathcal{N}(\mathbf{x}_n; \boldsymbol{\mu}_{ni}, \boldsymbol{\Sigma}_{ni}) p(\mathbf{x}_{n+1} \mid \mathbf{x}_n) d\mathbf{x}_n.\end{aligned} \tag{7}$$

Upon linearization of $f(\mathbf{x}_n)$ about $\boldsymbol{\mu}_{ni}$, each integral on the right-hand side of the above equation can be approximated as a Gaussian. We recall a theorem from [9].

Theorem 2: For an additive Gaussian noise model in the process equation in (1), i.e., $\mathbf{x}_n = \mathbf{f}(\mathbf{x}_{n-1}) + \mathbf{u}_n$, where $\mathbf{u}_n \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_n), \forall n$ and $p(\mathbf{x}_n \mid \mathbf{y}_{0:n})$ given by (4), the updated predictive distribution $p(\mathbf{x}_{n+1} \mid \mathbf{y}_{0:n})$ approaches the Gaussian sum

$$\sum_{i=1}^{G} \bar{w}_{(n+1)i} \mathcal{N}\left(\mathbf{x}_{n+1}; \bar{\boldsymbol{\mu}}_{(n+1)i}, \bar{\boldsymbol{\Sigma}}_{(n+1)i}\right)$$

uniformly in $\mathbf{x}_n$ as $\boldsymbol{\Sigma}_{ni} \to 0$ for $i = 1, \ldots, G$, where $\bar{\boldsymbol{\mu}}_{(n+1)i}$ and $\bar{\boldsymbol{\Sigma}}_{(n+1)i}$ are updated using the following equations:

$$\begin{aligned}\bar{\boldsymbol{\mu}}_{(n+1)i} &= \mathbf{f}(\boldsymbol{\mu}_{ni}) \\ \bar{\boldsymbol{\Sigma}}_{(n+1)i} &= \mathbf{F}_{(n+1)i}\boldsymbol{\Sigma}_{ni}\mathbf{F}_{(n+1)i}^T + \mathbf{Q}_n \\ \mathbf{F}_{ni} &= \frac{\partial \mathbf{f}(\mathbf{x})}{\partial \mathbf{x}}\Big|_{\mathbf{x}=\boldsymbol{\mu}_{ni}} \\ \bar{w}_{(n+1)i} &= w_{ni}.\end{aligned} \tag{8}$$

Proof: See [9, pp. 215 and 216].

3) Filter Implementation: The filter is initialized with $p(\mathbf{x}_0)$ approximated by a weighted GM. In the measurement- and time-update equations described above, the updated mean and covariance of each mixand follow from the EKF equations. The GSF-I is implemented by G parallel EKFs, and the weights are adjusted according to the given update equations. However, according to the theorems, the approximations are valid for "small" covariances. An increase in the covariances leads to problems, which are discussed in Section II-D.

B. Gaussian Sum Particle Filtering—I

For a nonlinear DSS with additive Gaussian noise, the GSF-I approximates posterior distributions as Gaussian mixtures using banks of parallel EKFs. Based on similar reasoning, the GSPF-I updates the mixands using banks of GPF's in parallel.

1) Measurement Update: Assume that at time n, we have the predictive distribution

$$p\left(\mathbf{x}_n \mid \mathbf{y}_{0:n-1}\right) = \sum_{i=1}^{G} \bar{w}_{ni} \mathcal{N}\left(\mathbf{x}_n; \bar{\boldsymbol{\mu}}_{ni}, \bar{\boldsymbol{\Sigma}}_{ni}\right). \tag{9}$$

After receiving $\mathbf{y}_n$, the filtering distribution is given by (3). As in the GPF, each term on the right-hand side of (3) given by $p(\mathbf{y}_n \mid \mathbf{x}_n)\mathcal{N}(\mathbf{x}_n; \bar{\boldsymbol{\mu}}_{ni}, \bar{\boldsymbol{\Sigma}}_{ni})$ is approximated as a Gaussian. This allows for the measurement update algorithm described in Table II.

The updated filtering distribution can now be represented as

$$p(\mathbf{x}_n \mid \mathbf{y}_{0:n}) \approx \sum_{i=1}^{G} w_{ni} \mathcal{N}(\mathbf{x}_n; \boldsymbol{\mu}_{ni}, \boldsymbol{\Sigma}_{ni}). \tag{10}$$

2) Time Update: Assume that at time n, we have

$$p(\mathbf{x}_n \mid \mathbf{y}_{0:n}) = \sum_{i=1}^{G} w_{ni} \mathcal{N}\left(\mathbf{x}_n; \boldsymbol{\mu}_{ni}, \boldsymbol{\Sigma}_{ni}\right). \tag{11}$$

From (7), the predictive distribution is given by

$$p\left(\mathbf{x}_{n+1} \mid \mathbf{y}_{0:n}\right) = \sum_{i=1}^{G} w_{ni} \int \mathcal{N}\left(\mathbf{x}_n; \boldsymbol{\mu}_{ni}, \boldsymbol{\Sigma}_{ni}\right) \cdot p(\mathbf{x}_{n+1} \mid \mathbf{x}_n) d\mathbf{x}_n. \tag{12}$$

The integral on the right-hand side is approximated by a Gaussian using the time update algorithm of the GPF. The time update algorithm is summarized in Table IV.

The time updated (predictive) distribution is now approximated as

$$p\left(\mathbf{x}_{n+1} \mid \mathbf{y}_{0:n}\right) \approx \sum_{i=1}^{G} \bar{w}_{(n+1)i} \mathcal{N}\left(\mathbf{x}_{n+1}; \bar{\boldsymbol{\mu}}_{(n+1)i}, \bar{\boldsymbol{\Sigma}}_{(n+1)i}\right). \tag{13}$$

C. Inference

The Gaussian mixture approximation lends an advantage that MMSE estimates of the hidden state and its error covariance can be obtained straightforwardly. From (10), the estimate of $\mathbf{x}_n$, $\hat{\mathbf{x}}_n = E(\mathbf{x}_n \mid \mathbf{y}_{0:n})$ and $\hat{\boldsymbol{\Sigma}}_n = E(\mathbf{x}_n - \hat{\mathbf{x}}_n)(\mathbf{x}_n - \hat{\mathbf{x}}_n)^T$ can be computed from

$$\hat{\mathbf{x}}_n = \sum_{i=1}^{G} w_{ni} \boldsymbol{\mu}_{ni}$$

$$\hat{\boldsymbol{\Sigma}}_n = \sum_{i=1}^{G} w_{ni} (\boldsymbol{\Sigma}_{ni} + (\hat{\mathbf{x}}_n - \boldsymbol{\mu}_{ni})(\hat{\mathbf{x}}_n - \boldsymbol{\mu}_{ni})^T).$$

TABLE II

Gaussian Sum Particle filter-I - Measurement update algorithm

1. For $i = 1, \ldots, G$, draw samples from the importance function $\pi(\mathbf{x}_n|\mathbf{y}_{0:n})$ and denote them as $\{\mathbf{x}_{ni}^{(j)}\}_{j=1}^{M}$.
2. For $i = 1, \ldots, G$, $j = 1, \ldots, M$, compute weights by

$$\gamma_{ni}^{(j)} = \frac{p(\mathbf{y}_n|\mathbf{x}_{ni}^{(j)})\mathcal{N}(\mathbf{x}_n = \mathbf{x}_{ni}^{(j)}; \bar{\boldsymbol{\mu}}_{ni}, \bar{\boldsymbol{\Sigma}}_{ni})}{\pi(\mathbf{x}_{ni}^{(j)}|\mathbf{y}_{0:n})}.$$

3. For $i = 1, \ldots, G$ estimate the mean and covariance as

$$\boldsymbol{\mu}_{ni} = \frac{\sum_{j=1}^{M} \gamma_{ni}^{(j)} \mathbf{x}_{ni}^{(j)}}{\sum_{j=1}^{M} \gamma_{ni}^{(j)}},$$

$$\boldsymbol{\Sigma}_{ni} = \frac{\sum_{j=1}^{M} \gamma_{ni}^{(j)} (\mathbf{x}_{ni}^{(j)} - \boldsymbol{\mu}_{ni})(\mathbf{x}_{ni}^{(j)} - \boldsymbol{\mu}_{ni})^T}{\sum_{j=1}^{M} \gamma_{ni}^{(j)}}.$$

4. Update the weights as

$$\tilde{w}_{ni} = \bar{w}_{(n-1)i} \frac{\sum_{j=1}^{M} \gamma_{ni}^{(j)}}{\sum_{i=1}^{G} \sum_{j=1}^{M} \gamma_{ni}^{(j)}}, \quad i = 1, \ldots, G.$$

5. Normalize the weights according to

$$w_{ni} = \frac{\tilde{w}_{ni}}{\sum_{i=1}^{G} \tilde{w}_{ni}}.$$

Gaussian Sum Particle filter-I - Time update algorithm

1. For $i = 1, \ldots, G$, obtain samples from $\mathcal{N}(\mathbf{x}_n; \boldsymbol{\mu}_{ni}, \boldsymbol{\Sigma}_{ni})$ and denote them as $\{\mathbf{x}_{ni}^{(j)}\}_{j=1}^{M}$.
2. For $i = 1, \ldots, G$, $j = 1, \ldots, M$ obtain samples from $p(\mathbf{x}_{(n+1)i}|\mathbf{x}_{ni} = \mathbf{x}_{ni}^{(j)})$ and denote them as $\{\mathbf{x}_{(n+1)i}^{(j)}\}_{j=1}^{M}$.
3. For $i = 1, \ldots, G$, update weights $\bar{w}_{(n+1)i} = w_{ni}$.
4. For $i = 1, \ldots, G$, obtain $\bar{\boldsymbol{\mu}}_{(n+1)i}$ and $\bar{\boldsymbol{\Sigma}}_{(n+1)i}$ by taking sample means and covariances respectively.

D. Discussion

The practical implementation of GSF-I and GSPF-I may present difficulties that are discussed in the next few paragraphs.

1) The choice of the number of mixands G is often guided by the problem at hand. In practical applications, the number of mixands in the approximation of the prediction and filtering distributions is usually small. As a result, divergence may still occur in the GSF-I due to the linearizations in the EKF, as can happen in the standard EKF because of the severe nonlinearities in the model. The

GSPF-I can be effectively used to mitigate divergence due to the use of GPFs in parallel.

2) The most important limitation of the GSF-I and GSPF-I is the *collapsing* of the mixands. As the filtering proceeds, the covariance of the mixands can grow. Consequently, according to Theorems 1 and 2, the GM approximations become increasingly coarse. More importantly, this can cause collapsing of all the mixands, resulting in only one distinct mixand. The result is that after several updates, the posterior distributions are approximated as a single Gaussian, which may be a poor approximation. Moreover, computation power is wasted in updating identical mixands. When collapsing occurs, the posterior distributions have to be re-expressed as Gaussian mixtures having small covariances [9]. In [10], a special algorithm for this reinitialization was suggested for the particular problem of frequency and phase demodulation. However, re-expressing the posterior distributions as a Gaussian mixture with small covariances may itself be a challenging problem, especially in higher dimensions, besides being undesirable and computationally expensive. There is no general criterion to determine how "small" the covariance should remain to avoid collapsing, and it has to be determined based on the problem at hand.
3) The covariance of the mixands grows especially when the covariance of the process noise is large compared with the covariance of the mixands. To combat this problem, in [4] it has been suggested to approximate the Gaussian noise process as a finite Gaussian mixture itself. Inspection of the time update equations shows that this results in an exponentially growing number of mixands. Several *ad hoc* procedures have been suggested in [11] to reduce the number of mixands, but they may become inadequate in many practical problems and, moreover, difficult to implement.
4) In the update algorithms of the GSPF, at each time instant n, particles from the filtering and predictive distributions, which are used to obtain approximations for the mixands, are obtained. These particles and their weights represent samples from the filtering and posterior distributions, much like the particles in particle filters. In an alternative approach, the EM [5] algorithm can be used to obtain GM approximations from these particles and weights, and a recursive algorithm can be developed based on this idea. With this mechanism, the collapsing problem will not arise. The advantages of using EM to obtain GM approximations compared with the SIS algorithms are that the resampling procedure can be avoided and that the tails of the distributions can be better represented than by the SIS approximations.

III. Non-Gaussian Noise and Gaussian Mixture Models

Non-Gaussian noise in general is more difficult to handle than Gaussian noise. The "nice" properties for estimators and detectors in Gaussian noise do not carry over to non-Gaussian noise problems. Gaussian mixture models (GMMs) have been suggested by many researchers [4]–[6], [8], [12]–[14] as an approximation to non-Gaussian densities. For example, in [4], it is shown that any density can be approximated "as closely as possible" by a finite GM; see [1, Lemma 1]. Moreover, Middleton's canonical class A model [15], [16], where probability density functions are expressed as infinite sum of Gaussians, in practical applications can be approximated by finite Gaussian mixtures. For example, impulsive noise approximated with two mixands in the GMM can be used to model acoustic noise for underwater channels [7], atmospheric noise in long range communications, or noise in seismic data. Impulsive noise is often encountered in wireless applications in many indoor and outdoor environments; see [8] and the references therein, where the channel noise is a finite GMM [17].

In this work, we endeavor to provide a general framework to handle non-Gaussian noise in DSS models, where the underlying assumption is that the noise is represented as a finite GM. For the DSS model with additive noise in (1), let the process noise be given by

$$p(\mathbf{u}_n) = \sum_{k=1}^{K} \alpha_k \mathcal{N}\left(\mathbf{u}_n; \tilde{\boldsymbol{\mu}}_{nk}, \tilde{\boldsymbol{\Sigma}}_{nk}\right). \tag{14}$$

For simplification of presentation, we assume in the further discussion that the observation noise $\mathbf{v}_n$ is Gaussian. From the discussion and presented algorithm, it will be clear that the algorithm can be straightforwardly generalized to non-Gaussian case by modeling the observation noise as a finite GM. At time zero, our prior knowledge about $\mathbf{x}_0$ is summarized by $p(\mathbf{x}_0)$. The predictive distribution of $\mathbf{x}_1$ can then be written as

$$\begin{aligned} p(\mathbf{x}_1) &= \int p(\mathbf{x}_1 \mid \mathbf{x}_0) p(\mathbf{x}_0) d\mathbf{x}_0 \\ &= \int \sum_{k=1}^{K} \alpha_k \mathcal{N}(\mathbf{x}_1; \mathbf{f}(\mathbf{x}_0) + \tilde{\boldsymbol{\mu}}_{0k}, \tilde{\boldsymbol{\Sigma}}_{0k}) p(\mathbf{x}_0) d\mathbf{x}_0 \\ &= \sum_{k=1}^{K} \alpha_k \int \mathcal{N}(\mathbf{x}_1; \mathbf{f}(\mathbf{x}_0) + \tilde{\boldsymbol{\mu}}_{0k}, \tilde{\boldsymbol{\Sigma}}_{0k}) p(\mathbf{x}_0) d\mathbf{x}_0 \\ &= \sum_{k=1}^{K} \alpha_k \tilde{p}_k(\mathbf{x}_1) \end{aligned} \tag{15}$$

where $\int \mathcal{N}(\mathbf{x}_1; \mathbf{f}(\mathbf{x}_0) + \tilde{\boldsymbol{\mu}}_{0k}, \tilde{\boldsymbol{\Sigma}}_{0k}) p(\mathbf{x}_0) d\mathbf{x}_0 = \tilde{p}_k(\mathbf{x}_1)$ has been defined.

After the arrival of $\mathbf{y}_1$, the posterior distribution can be expressed as

$$\begin{aligned} p(\mathbf{x}_1 \mid \mathbf{y}_1) &= C_1 p(\mathbf{y}_1 \mid \mathbf{x}_1) p(\mathbf{x}_1) \\ &= C_1 \sum_{k=1}^{K} \alpha_k p(\mathbf{y}_1 \mid \mathbf{x}_1) \tilde{p}_k(\mathbf{x}_1) \\ &= \sum_{k=1}^{K} \beta_{1k} p_k(\mathbf{x}_1 \mid \mathbf{y}_1) \end{aligned} \tag{16}$$

where C_1 is a proportionality constant

$$\begin{aligned} C_1 &= \left(\int p(\mathbf{y}_1 \mid \mathbf{x}_1) p(\mathbf{x}_1) \right)^{-1} \\ &= \left(\sum_{k=1}^{K} \alpha_k \int p(\mathbf{y}_1 \mid \mathbf{x}_1) \tilde{p}_k(\mathbf{x}_1) d\mathbf{x}_1 \right)^{-1} \end{aligned}$$

the constants β_{1k} are normalized weights, and

$$p_k(\mathbf{x}_1 \mid \mathbf{y}_1) \propto p(\mathbf{y}_1 \mid \mathbf{x}_1)\tilde{p}_k(\mathbf{x}_1).$$

At the arrival of the nth observation, we can write

$$p(\mathbf{x}_n \mid \mathbf{y}_{1:n-1}) = \sum_{k=1}^{K}\sum_{j=1}^{K^{n-1}} \alpha_k \beta_{(n-1)j}\tilde{p}_{jk}(\mathbf{x}_n \mid \mathbf{y}_{1:n-1}) \tag{17}$$

and

$$p(\mathbf{x}_n \mid \mathbf{y}_{1:n}) = \sum_{j=1}^{K^n} \beta_{nj} p_j(\mathbf{x}_n \mid \mathbf{y}_{1:n}) \tag{18}$$

which indicates that the number of mixands grows exponentially with the arrival of new observations. For practical applications, updating an exponentially growing number of mixands can be cumbersome and prohibitive, and hence, Bayesian inference becomes increasingly complex. However, it is possible to reduce the complexity of the problem, as discussed in the sequel.

A. Parallel DSS Models and Resampling

An interesting perspective is developed by looking at the process equation of the DSS model (1). The process noise is a GMM with K mixands. Hence, we can view the *process* of the DSS model being excited by K different *Gaussian* excitation noises $\mathcal{N}(\mathbf{u}_n; \tilde{\boldsymbol{\mu}}_{nk}, \tilde{\boldsymbol{\Sigma}}_{nk})$, each with probability α_k. As a result, the nonlinear non-Gaussian noise DSS model is equivalent to a weighted sum of nonlinear Gaussian noise DSS models. Consequently, from (18), at time n, there are K^n parallel nonlinear Gaussian noise DSS models each having weight β_{nj}. However, for a given set of observations $\mathbf{y}_{1:n}$, only a few of these parallel models will have "significant" weights. This can be anticipated from the observation update (16), where the weights of the mixands are redistributed to β_{nj}, depending on the likelihood $p(\mathbf{y}_n \mid \mathbf{x}_n)$. As a result, in practical applications, it is possible to approximate the posterior $p(\mathbf{x}_n \mid \mathbf{y}_{1:n})$ as a weighted mixture of say G mixands, where the mixands having insignificant weights have been eliminated, i.e.,

$$p(\mathbf{x}_n \mid \mathbf{y}_{1:n}) \approx \sum_{j=1}^{G} \beta_{nj} p_j(\mathbf{x}_n \mid \mathbf{y}_{1:n}). \tag{19}$$

Clearly, G depends on the model equations and the weights α_k for a given error cost function. Now, assume that at time n, $p(\mathbf{x}_n \mid \mathbf{y}_{1:n})$ is given by the above approximation. With the arrival of a new observation $\mathbf{y}_{n+1}$, it follows that $p(\mathbf{x}_n \mid \mathbf{y}_{1:n})$ has GK mixands, or

$$\begin{aligned} p(\mathbf{x}_{n+1} \mid \mathbf{y}_{1:n}) \approx & \sum_{k=1}^{K}\sum_{j=1}^{G} \alpha_k \beta_{nj} \\ & \cdot \int \mathcal{N}\left(\mathbf{x}_{n+1}; \mathbf{f}(\mathbf{x}_n) + \tilde{\boldsymbol{\mu}}_{nk}, \tilde{\boldsymbol{\Sigma}}_{nk}\right) \\ & \cdot p_j(\mathbf{x}_n \mid \mathbf{y}_{1:n})\, d\mathbf{x}_n \\ = & \sum_{k=1}^{K}\sum_{j=1}^{G} \alpha_k \beta_{nj} \tilde{p}_{jk}(\mathbf{x}_n \mid \mathbf{y}_{1:n}) \end{aligned}$$

TABLE III

Resampling

1. Sort the GK mixands according to their weights β_{nj} in descending order. Retain only the first G mixands and denote the weights and mixands as β_{ng} and $p_g(\mathbf{x}_n|\mathbf{y}_n)$, $g = 1, \ldots, G$.
If the smallest weight $\beta_G < \beta_{threshold}$, then follow steps 2 and 3; otherwise go to step 3.
2. For $g = 1, \ldots, G$, draw a number $j \in \{1, \ldots, G\}$ with probabilities proportional to $\{\beta_{n1}, \ldots, \beta_{nG}\}$. Let $\{\boldsymbol{\mu}_{ng}, \boldsymbol{\Sigma}_{ng}\} = \{\boldsymbol{\mu}_{nj}, \boldsymbol{\Sigma}_{nj}\}$ and set $\beta_{ng} = 1/G$.
3. For $g = 1, \ldots, G$, reinitialize weights according to $\bar{\beta}_{ng} = \frac{\beta_{ng}}{\sum_{j=1}^{G}\beta_{nj}}$.

and

$$p(\mathbf{x}_{n+1} \mid \mathbf{y}_{1:n+1}) \approx \sum_{j=1}^{GK} \beta_{(n+1)j} p_j(\mathbf{x}_{n+1} \mid \mathbf{y}_{1:n+1}). \tag{20}$$

Typically, in practical situations, it is advantageous to keep the number of mixands constant at each time n. We export the idea of "resampling" from the particle filter methods to keep the number of mixands constant. Resampling throws out mixands having insignificant weights and duplicates the remaining, in proportion to their weights. In Table III, we outline one "simple resampling" scheme; for other possible schemes, see [18].

The value of $\beta_{threshold}$ depends on the problem and should be in general < 0.05. Resampling is applied to (20) and G mixands are retained. Then, the new approximation is written as

$$p(\mathbf{x}_{n+1} \mid \mathbf{y}_{1:n+1}) \approx \sum_{j=1}^{G} \bar{\beta}_{ng} p_j(\mathbf{x}_{n+1} \mid \mathbf{y}_{1:n+1}). \tag{21}$$

Resampling keeps the number of mixands at each time instant n limited to G, and hence, computational power is not wasted in generating mixands that have insignificant weights. In addition, by duplicating mixands in proportion to their weights, only mixands with significant weights are propagated further, which can possibly make significant contribution to the inference process. Since resampling throws away mixands, it should be applied carefully. From the above discussion, it is clear that a nonlinear *non-Gaussian* noise DSS model problem has been converted to a weighted sum of nonlinear *Gaussian* noise DSS models, where resampling is used to keep the number of mixands constant.

A few more remarks are in order.

1) Non-Gaussian Observation Noise: The non-Gaussian observation noise can be handled similarly, that is, by expressing it as a finite GM. Appropriate changes to (17) and (18) can be made and incorporated in the algorithms given below, which show an increase in the number of mixands even at the observation update steps.

2) Choice of G: This depends on the particular application and has to determined by trial and error. Underestimating G results in elimination of mixands with significant weights during resampling, and the Gaussian mixture approximation becomes coarser. As a result, underestimating G may lead to loss of tracking or divergence.

B. Nonlinear Filtering For Additive Non-Gaussian Noise DSS

A difficulty that still remains is that of solving the integrations in the time update steps to get $\tilde{p}_{jk}(\mathbf{x}_{n-1} \mid \mathbf{y}_{1:n-1})$ and obtaining $p_j(\mathbf{x}_n \mid \mathbf{y}_{1:n})$ and the weights β_{nj} in the observation update steps. In other words, it is required to obtain predictive and filtering distributions for each of the parallel nonlinear Gaussian noise DSS models and their weights. This can be done by EKFs, GPFs, or GSPFs, and these possibilities are illustrated in Fig. 1.

One set of methods approximates the mixture components of the predictive and filtering distributions as Gaussians. The approximation can be implemented by the EKF (or the unscented Kalman filter) and the GPF, resulting in a parallel bank of Gaussian filters. Thus, $\tilde{p}_{jk}(\mathbf{x}_n \mid \mathbf{y}_{1:n-1})$ and $p_j(\mathbf{x}_n \mid \mathbf{y}_{1:n})$ are approximated as Gaussians and the posterior densities $p(\mathbf{x}_n \mid \mathbf{y}_{1:n-1})$, and $p(\mathbf{x}_n \mid \mathbf{y}_{1:n})$ are then Gaussian mixtures. The resulting filters will be called GSF-II and GSPF-II, when EKF and GPF are used, respectively. The problem now reduces to updating a Gaussian mixture, where the mean, covariance, and weights are tracked with each new observation. Resampling is applied to keep the number of mixands constant at each time n. Details of the algorithm are given in the next section, and simulation results are presented in Section V.

The second set of methods approximates the mixture components of the predictive and filtering distributions $\tilde{p}_{jk}(\mathbf{x}_{n-1} \mid \mathbf{y}_{1:n-1})$ and $p_j(\mathbf{x}_n \mid \mathbf{y}_{1:n})$, as Gaussian mixtures by using the GSF-I or the GSPF-I. The resulting filters will be called GSF-III and GSPF-III, when GSF-I and GSPF-I are used, respectively. Hence, the posterior distributions $p(\mathbf{x}_n \mid \mathbf{y}_{1:n-1})$ and $p(\mathbf{x}_n \mid \mathbf{y}_{1:n})$ are approximated by weighted sum of Gaussian mixtures and, therefore, are Gaussian mixtures themselves. The GSF-III and GSPF-III algorithms are very similar to the GSF-I and GSPF-I, where Gaussian mixtures are updated. As for GSF-II and GSPF-II, resampling is applied to keep the number of mixands constant at each time n. These algorithms are discussed further in Section IV-C.

IV. Gaussian Mixture Filters for Non-Gaussian Noise

The GSF-II filter is a bank of EKFs running in parallel, where the filtering and predictive distributions are updated using the EKF equations. Similarly, the GSPF-II filter is a bank of GPFs running in parallel, where the filtering and predictive distributions are updated using the GPF algorithms. The number of mixands is kept constant by resampling after the measurement update. Recall that each of the distributions $\tilde{p}_{jk}(\mathbf{x}_{n-} \mid \mathbf{y}_{1:n-1})$ and $p_j(\mathbf{x}_n \mid \mathbf{y}_{1:n})$ in (17) and (18) are approximated as Gaussians. Assume that at time $n = 0$, $p(\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_0; \boldsymbol{\mu}_0, \boldsymbol{\Sigma}_0)$. Following the development of the equations in Section III and the algorithms given below, we run the filter *without* the resampling part in the measurement update until we obtain G (or just greater than G) mixands. Once the required G mixands are obtained, resampling is applied to keep the number of mixands constant, as described below. In the sequel, we describe the update algorithms.

A. Gaussian Sum Filter—II

1) Time-Update: Assume at time $n-1$ we have

$$p(\mathbf{x}_{n-1} \mid \mathbf{y}_{1:n-1}) = \sum_{i=1}^{G} w_{(n-1)i} \mathcal{N}\left(\mathbf{x}_{n-1}; \boldsymbol{\mu}_{(n-1)i}, \boldsymbol{\Sigma}_{(n-)i}\right). \tag{22}$$

The predictive distribution is given by

$$\begin{aligned} p(\mathbf{x}_n \mid \mathbf{y}_{1:n-1}) = &\sum_{k=1}^{K} \sum_{j=1}^{G} \alpha_k w_{(n-1)j} \\ &\times \int \mathcal{N}\left(\mathbf{x}_n; \mathbf{f}(\mathbf{x}_{n-1}) + \tilde{\boldsymbol{\mu}}_{(n-1)k}, \tilde{\boldsymbol{\Sigma}}_{(n-1)k}\right) \\ &\cdot \mathcal{N}(\mathbf{x}_{n-1}; \boldsymbol{\mu}_{(n-1)j}, \boldsymbol{\Sigma}_{(n-1)j}) d\mathbf{x}_{n-1}. \end{aligned} \tag{23}$$

As in the EKF, the integral on the right is approximated by a Gaussian. Then, the predictive distribution can be approximated as

$$p(\mathbf{x}_n \mid \mathbf{y}_{1:n-1}) \approx \sum_{i=1}^{GK} \bar{w}_{ni} \mathcal{N}\left(\mathbf{x}_n; \bar{\boldsymbol{\mu}}_{(n)i}, \bar{\boldsymbol{\Sigma}}_{(n)i}\right) \tag{24}$$

where the parameters of the mixture are obtained according to

$$\begin{aligned} \bar{\boldsymbol{\mu}}_{ni} &= \tilde{\boldsymbol{\mu}}_{(n-1)k} + \mathbf{f}(\boldsymbol{\mu}_{(n-1)j}) \\ \bar{\boldsymbol{\Sigma}}_{ni} &= \mathbf{F}_{(n-1)i} \boldsymbol{\Sigma}_{(n-1)j} \mathbf{F}^T_{(n-1)i} + \tilde{\boldsymbol{\Sigma}}_{(n-1)k} \\ \mathbf{F}_{(n-1)i} &= \frac{\partial \mathbf{f}(\mathbf{x})}{\partial \mathbf{x}} \mid_{\mathbf{x}=\mu_{(n-1)j}} \\ \bar{w}_{ni} &= \alpha_k w_{(n-1)j} \end{aligned}$$

for appropriate $i = 1, \ldots, GK$, $j = 1, \ldots, G$ and $k = 1, \ldots, K$ and $i = j + (k-1)K$.

2) Measurement-Update: Assume that at time n, we have the predictive distribution

$$p(\mathbf{x}_n \mid \mathbf{y}_{0:n-1}) = \sum_{i=1}^{GK} \bar{w}_{ni} \mathcal{N}(\mathbf{x}_n; \bar{\boldsymbol{\mu}}_{ni}, \bar{\boldsymbol{\Sigma}}_{ni}). \tag{25}$$

After receiving the nth observation $\mathbf{y}_n$, we obtain the filtering distribution from the predictive distribution by

$$p(\mathbf{x}_n \mid \mathbf{y}_{0:n}) = C_n \sum_{i=1}^{G} \bar{w}_{ni} p(\mathbf{y}_n \mid \mathbf{x}_n) \mathcal{N}(\mathbf{x}_n; \bar{\boldsymbol{\mu}}_{ni}, \bar{\boldsymbol{\Sigma}}_{ni}). \tag{26}$$

Therefore, the measurement-update steps consist of the following.

1) Obtain $w_{ni}, \boldsymbol{\mu}_{ni}$ and $\boldsymbol{\Sigma}_{ni}$ from the measurement update equations of the GSPF-I.
2) Resample to retain only G mixands from the GK mixands.
3) The filtering distribution is approximated as

$$p(\mathbf{x}_n \mid \mathbf{y}_{0:n}) \approx \sum_{i=1}^{GK} w_{ni} \mathcal{N}(\mathbf{x}_n; \boldsymbol{\mu}_{ni}, \boldsymbol{\Sigma}_{ni}). \tag{27}$$

B. Gaussian Sum Particle Filter-II

1) Time-Update: Following (22) and (23), from $p(\mathbf{x}_{n-1} \mid \mathbf{y}_{n-1}) = \sum_{j=1}^{G} w_{(n-1)j} \mathcal{N}(\mathbf{x}_{n-1}; \boldsymbol{\mu}_{(n-1)j}, \boldsymbol{\Sigma}_{(n-1)j})$, we obtain $p(\mathbf{x}_n \mid \mathbf{y}_{1:n-1}) \approx \sum_{i=1}^{GK} \bar{w}_{ni} \mathcal{N}(\mathbf{x}_n; \bar{\boldsymbol{\mu}}_{ni}, \bar{\boldsymbol{\Sigma}}_{ni})$. Define

TABLE IV

Gaussian Sum Particle Filter-II - Time update algorithm

1. For $j = 1,\ldots,G$, obtain samples from $\mathcal{N}(\mathbf{x}_{n\ 1};\boldsymbol{\mu}_{(n\ 1)j},\boldsymbol{\Sigma}_{(n\ 1)j})$ and denote them as $\{\mathbf{x}^{(m)}_{(n\ 1)j}\}_{m=1}^{M}$.
2. For $j = 1,\ldots,G$, $k = 1,\ldots,K$ and $m = 1,\ldots,M$, obtain samples from $\mathcal{N}(\mathbf{x}_n;\mathbf{f}(\mathbf{x}_{n\ 1} = \mathbf{x}^{(m)}_{(n\ 1)j}) + \tilde{\boldsymbol{\mu}}_{(n\ 1)k},\tilde{\boldsymbol{\Sigma}}_{(n\ 1)k})$ and denote them as $\{\mathbf{x}^{(m)}_{ni}\}_{m=1}^{M}$, where $i = j + (k\ \ 1)K$.
3. For $i = 1,\ldots,GK$, obtain $\bar{\boldsymbol{\mu}}_{ni}$ and $\bar{\boldsymbol{\Sigma}}_{ni}$ by taking sample mean and sample covariance of the particles $\{\mathbf{x}^{(m)}_{ni}\}_{m=1}^{M}$.
4. For $j = 1,\ldots,G$, $k = 1,\ldots,K$ and $i = 1,\ldots,GK$, where $i = j+(k\ \ 1)K$, update weights for each mixand according to

$$\bar{w}_{ni} = w_{(n\ 1)j}\alpha_k \Big/ \sum_{k=1}^{K}\sum_{j=1}^{G} w_{(n\ 1)j}\alpha_k.$$

Gaussian Sum Particle Filter-II - Measurement update algorithm

1. For $i = 1,\ldots,GK$, draw samples from the importance function $\pi(\mathbf{x}_n|\mathbf{y}_{0:n})$ and denote them as $\{\mathbf{x}^{(j)}_{ni}\}_{j=1}^{M}$.
2. For $j = 1,\ldots,M$, $i = 1,\ldots,GK$ compute weights by

$$\gamma^{(j)}_{ni} = \frac{p(\mathbf{y}_n|\mathbf{x}^{(j)}_{ni})\mathcal{N}(\mathbf{x}_n = \mathbf{x}^{(j)}_{ni};\bar{\boldsymbol{\mu}}_{ni},\bar{\boldsymbol{\Sigma}}_{ni})}{\pi(\mathbf{x}^{(j)}_{ni}|\mathbf{y}_{0:n})}.$$

3. For $i = 1,\ldots,GK$, estimate the mean and covariance as

$$\boldsymbol{\mu}_{ni} = \frac{\sum_{j=1}^{M}\gamma^{(j)}_{ni}\mathbf{x}^{(j)}_{ni}}{\sum_{j=1}^{M}\gamma^{(j)}_{ni}},$$
$$\boldsymbol{\Sigma}_{ni} = \frac{\sum_{j=1}^{M}\gamma^{(j)}_{ni}(\mathbf{x}^{(j)}_{ni}\ \ \boldsymbol{\mu}_{ni})(\mathbf{x}^{(j)}_{ni}\ \ \boldsymbol{\mu}_{ni})^T}{\sum_{j=1}^{M}\gamma^{(j)}_{ni}}.$$

4. For $i = 1,\ldots,GK$, update the weights as

$$\tilde{w}_{ni} = \bar{w}_{(n\ 1)i}\frac{\sum_{j=1}^{M}\gamma^{(j)}_{ni}}{\sum_{i=1}^{G}\sum_{j=1}^{M}\gamma^{(j)}_{ni}}, \quad i = 1,\ldots,G.$$

5. Normalize the weights according to

$$w_{ni} = \tilde{w}_{ni}\Big/\sum_{i=1}^{G}\tilde{w}_{ni}.$$

6. Resample to retain G mixands from GK mixands.

$i = j + (k-1)K$ so that $j = 1,\ldots,G$, $k = 1,\ldots,K$ and $i = 1,\ldots,GK$, and let i imply reference to the respective j and k. The time-update algorithm is given in Table IV.

2) Measurement-Update: From (25) and (27) and $p(\mathbf{x}_n \mid \mathbf{y}_{0:n-1}) = \sum_{i=1}^{GK}\bar{w}_{ni}\mathcal{N}(\mathbf{x}_n;\bar{\boldsymbol{\mu}}_{ni},\bar{\boldsymbol{\Sigma}}_{ni})$, we obtain $p(\mathbf{x}_n \mid \mathbf{y}_{0:n}) = \sum_{i=1}^{GK} w_{ni}\mathcal{N}(\mathbf{x}_n;\boldsymbol{\mu}_{ni},\boldsymbol{\Sigma}_{ni})$. The measurement-update algorithm is then described in Table IV.

Remarks:

1) If the DSS model in (1) is *linear* with additive non-Gaussian noise, then it is clearly approximated as a bank of weighted parallel *linear* additive Gaussian noise DSS models. The standard Kalman filter equations can be used to update the predictive and filtering distributions for each of the parallel linear additive Gaussian noise DSS models, resulting in a near-optimal solution. Thus, in this case, the GSF-II becomes a bank of weighted parallel Kalman filters, which becomes a special case of the mixture Kalman filter [19]. This method has been used to track an impulsive fading channel in [2], where a sequential Monte Carlo sampling algorithm is proposed for joint channel estimation and data detection. This example is also discussed in the simulations in Section V. A similar example was used in [20].
2) Target tracking in clutter involves associating possibly multiple observations arising due to clutter to the moving target. Various techniques are applied to accomplish this [21], most of which result in filters that are "similar" to the GSF-II and GSPF-II. However, the growing mixands are discarded in *ad hoc* ways, resulting in the different filters, whereas we suggest here a systematic way to accomplish the task. Hence, these filters can in principle be used in target tracking applications.

C. GSF-III and GSPF-III

For these filters, $\tilde{p}_{jk}(\mathbf{x}_n \mid \mathbf{y}_{1:n-1})$ and $p_j(\mathbf{x}_n \mid \mathbf{y}_{1:n})$ are approximated as GMs by using the GSF-I or the GSPF-I. Inspection of (17) and (18) shows that the GSF-III and GSPF-III are banks of GSF-Is and GSPF-Is running in parallel, respectively. As a result, the filtering and predictive distribution in (17) and (18) are also GMs. Hence, both algorithms essentially update a Gaussian mixture by approximating the new means, covariances, and weights using the EKF and GPF in parallel, respectively. The resulting algorithms are similar to the ones obtained in Section IV-A and B and are not repeated here.

V. Simulation Results

Some of the filtering methods proposed in this paper were applied to three different problems and here we present some of the obtained results.

A. Example 1—Univariate Non-Stationary Growth Model

We consider the univariate nonstationary growth model (UNGM) [22]–[24], whose DSS equations are given by

$$x_n = \alpha x_{n-1} + \beta\frac{x_{n-1}}{1+x_{n-1}^2} + \gamma\cos(1.2(n-1)) + u_n$$
$$y_n = \frac{x_n^2}{20} + v_n, \quad n = 1,\ldots,N$$

where $v_n \sim \mathcal{N}(0,\sigma_v^2)$, and the distribution of u_n is specified below. The data were generated using $x_0 = 0.1$, $\sigma_v^2 = 1$, $\alpha = 0.5, \beta = 25$ $\gamma = 8$ and $N = 500$ in each simulation. The process noise distribution is a Gaussian mixture given by

$$p(u_n) = \epsilon\mathcal{N}\left(u;0,\sigma_{u1}^2\right) + (1-\epsilon)\mathcal{N}\left(u;0,\sigma_{u2}^2\right).$$

By varying ϵ and the variances, heavy-tailed densities can be modeled quite well. We show results where $\epsilon = 0.8$, $\sigma_{u1}^2 = 0.1$, and $\sigma_{u2}^2 = 1$.

We compare the estimation and prediction performance of the GSF-II, GSPF-II, and SISR filters. For the present example, we had $G = 8$ for both GSF-II and GSPF-II. The number of particles chosen for each mixand update in the GSPF was $M = 100$. The prior density $p(\mathbf{x}_0)$ was $\mathcal{N}(0, 1)$. We compare performance of the filters based on MSEx_f, MSEx_p and MSEy_p, which are defined by

$$\text{MSE}x_f = \frac{1}{N}\sum_{n=1}^{N}(x_n - \hat{x}_n)^2 \tag{28}$$

$$\text{MSE}x_p = \frac{1}{N}\sum_{n=1}^{N}(x_n - \hat{\hat{x}}_n)^2 \tag{29}$$

$$\text{MSE}y_p = \frac{1}{N}\sum_{n=1}^{N}(y_n - \hat{y}_n)^2 \tag{30}$$

where $\hat{x}_n = E(x_n \mid y_{0:n})$, $\hat{\hat{x}}_n = E(x_n \mid y_{0:n-1})$, and $\hat{y}_n = E(y_n \mid y_{0:n})$. The MMSE estimate of y_n is given by $\hat{y}_n = E(y_n \mid y_{0:n-1})$ or

$$\begin{aligned}\hat{y}_n &= \int y_n p(y_n \mid y_{0:n-1})\, dy_n \\ &= \iint y_n p(y_n \mid x_n)\, p(x_n \mid y_{0:n})\, dx_n dy_n.\end{aligned}$$

For this example, we obtain

$$\hat{y}_n = \int \frac{x_n^2}{20} p(x_n \mid y_{0:n}) dx_n.$$

The GSPF-II is used here because the noise is heavy-tailed non-Gaussian, and heavy-tailed densities can be modeled as Gaussian mixtures [7], [25]. For the GSPF-II, where the GPF is used for updating each mixand and resampling is used, we expect better performance than the GSF-II.

A large number of simulations were performed to compare the three filters. Results are discussed below for 50 random realizations. Note from the figures that for the 46th realization, the GSF-II diverged.

Figs. 2 and 3 show the MSEx_f for 50 random realizations. The numbers of particles were $M = 20$ and 100 per mixand for GSPF-II and $M = 640$ and 5000 for SISR, respectively (the total number of particles used for inference in both filters was about the same). From the figures, it is evident that the GSPF-II and SISR filters perform significantly better than the GSF-II, whereas the MSE for the GSPF-II is marginally higher than that of SISR. The performance changed negligibly as the number of particles was increased.

Similar behavior was observed for the MSEx_p and MSEy_p metrics, as seen in Figs. 4 and 5, where the average MSEs are shown for the same 50 realizations. The MSEy_p is plotted on a logarithmic scale; note that the MSEy_p for the GSF-II is a few orders higher than the other filters (this can be justified by looking at the observation equation of the model; any error in x_n is raised to the power of 2 in the observation).

A comparison of the computation times is shown in Fig. 6, for simulations implemented on a 450-MHz Intel Pentium III processor using MATLAB. The computation time for GSPF-II and SISR filters is much higher than that of the GSF-II. More importantly, the GSPF-II has much shorter computation time than the SISR, even though their MSEs are comparable. The difference in computation time increases with the number of particles. This is due to the additional resampling required by the SISR filter, which has computational complexity of $O(M)$ for the systematic resampling scheme used here. However, as noted before, these times indicate those obtained on a serial computer, and much reduction in computation times can be expected for the GPF and SISR when implemented in parallel.

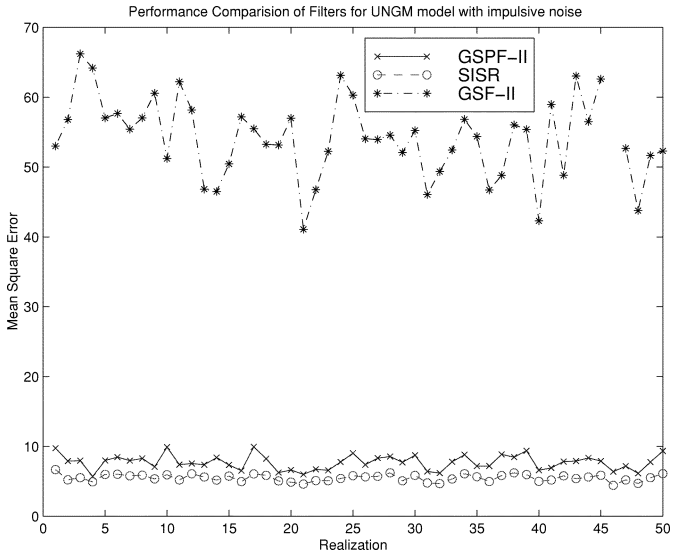

Fig. 2. Performance comparison of GSF-II, GSPF-II, and SISR filters. MSEx_f is plotted for 50 random realizations. M=20 (per mixand) for GSPF-II, and M=640 for SISR filter.

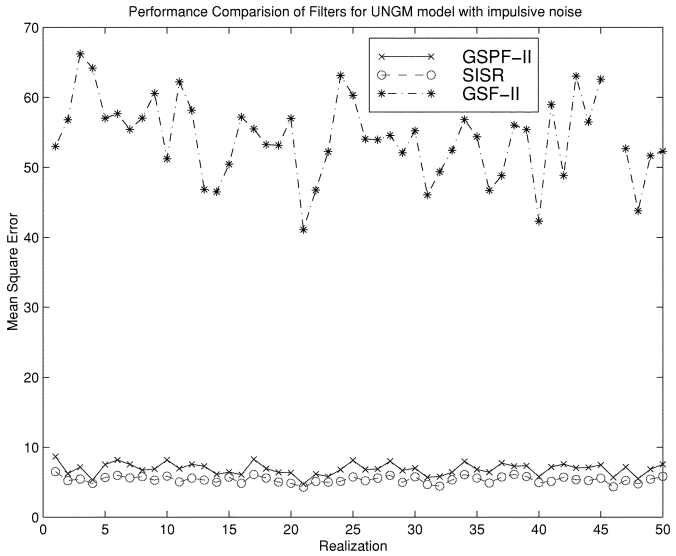

Fig. 3. Performance comparison of GSF-II, GSPF-II, and SISR filters. MSEx_f is plotted for 50 random realizations. M = 100 (per mixand) for GSPF-II and M = 5000 for SISR filter.

B. Example 2—Frequency Demodulation

We consider demodulation of a frequency (or phase) modulated signal $\lambda(t)$, which has a first-order Butterworth spectrum

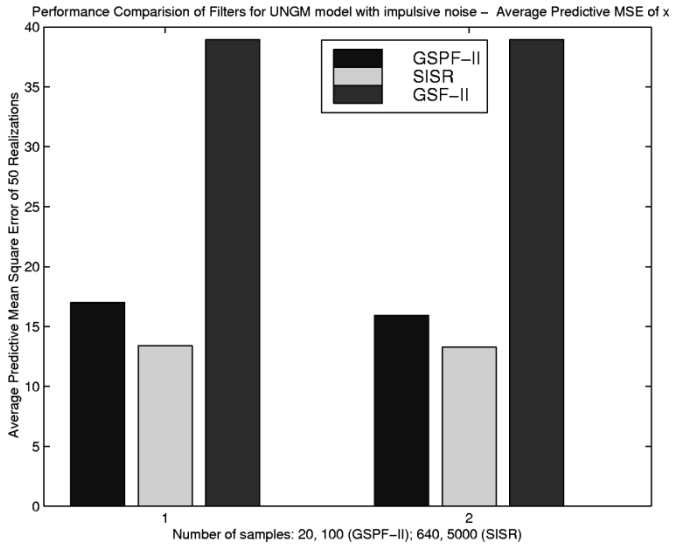

Fig. 4. Performance comparison of GSF-II, GSPF-II, and SISR filters. MSEx_f is plotted for 50 random realizations. Left side—$M = 20$ (per mixand) for GSPF-II and $M = 640$ for SISR filter. Right side—$M = 100$ (per mixand) for GSPF-II and $M = 5000$ for SISR filter.

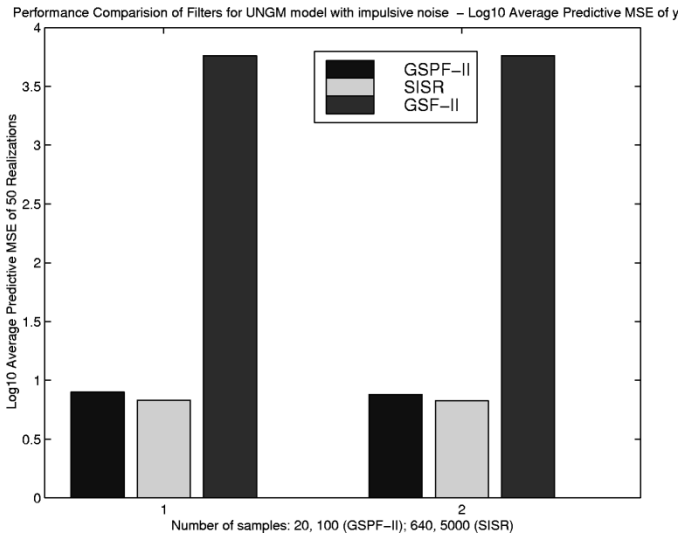

Fig. 5. Performance comparison of GSF-II, GSPF-II, and SISR filters. MSEx_f is plotted for 50 random realizations. Left side—$M = 20$ (per mixand) for GSPF-II and $M = 640$ for SISR filter. Right side—$M = 100$ (per mixand) for GSPF-II and $M = 5000$ for SISR filter.

[9], [10], [26]. The signal is modeled as the output of a first order, time-invariant linear system with one real pole driven by continuous time white noise. This message is then passed through an integrator to yield $\theta(t) = \int_0^t \lambda(t)dt$, which is then employed to phase modulate a carrier signal with frequency ω_c rad/s. The process equation can be written as

$$\begin{bmatrix} \dot{\lambda}(t) \\ \dot{\theta}(t) \end{bmatrix} = \begin{bmatrix} -\frac{1}{\beta} & 0 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} \lambda(t) \\ \theta(t) \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \end{bmatrix} v(t)$$

where $v(t)$ is zero mean noise process with $E(v(t)v(t-\tau)) = (2/\beta)\delta(t-\tau)$. The signal at the receiver is given by

$$z(t) = \sqrt{(2)} \sin(\omega_c t + \theta(t)) + u(t).$$

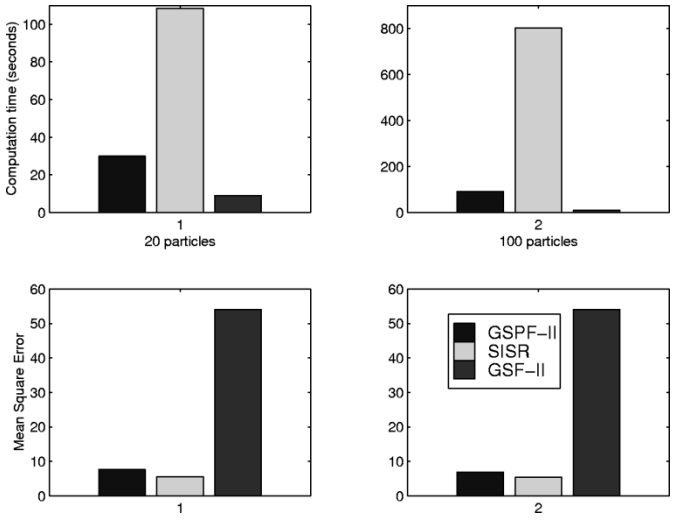

Fig. 6. Performance comparison of GSF-II, GSPF-II, and SISR filters. Computation time (per realization) and average MSEx_f of 50 random realizations. Left side—$M = 20$ (per mixand) for GSPF-II and $M = 640$ for SISR filter. Right side—$M = 100$ (per mixand) for GSPF-II and $M = 5000$ for SISR filter.

At the receiver, the signal is sampled by in-phase and quadrature-phase sampling, where the channel noise is mixture Gaussian. Thus, we have

$$\mathbf{z}_n = \sqrt{(2)} \begin{bmatrix} \sin(\theta_n) \\ \cos(\theta_n) \end{bmatrix} + \mathbf{u}_n$$

where $\mathbf{u}_n$ is zero mean noise mixture Gaussian noise with two mixands with weights ϵ and $1-\epsilon$ and covariances given by

$$E\left(\mathbf{u}_{ik}\mathbf{u}_{il}^T\right) = \begin{bmatrix} \sigma_i^2 & 0 \\ 0 & \sigma_i^2 \end{bmatrix} \delta_{kl}, \quad k = 1, 2.$$

The derivation of the discrete time process equation of the FM system can be found in [26]. We compare the performance of the GSF-II and GSPF-II filters, which were used for demodulation. Note that the measurement noise was mixture Gaussian while the process noise was Gaussian, as opposed to the assumptions used for the algorithms presented for GSF-II and GSPF-II.

The performance measure frequently used for FM systems is the steady-state inverse of the message error covariance given by ξ_λ^{-1}, where $\xi_\lambda = \lim_{n\to\infty} E[\lambda_n - \hat{\lambda}_n]_2$. We estimate $\xi_\lambda \approx 1/N \sum_{n=P}^{P+N} (\lambda_n - \hat{\lambda}_n)_2$, where $P = 1000$ is the settling period, and $\hat{\lambda}_n = E(\lambda_n \mid z_{0:n})$. This measure is plotted against the carrier-to-noise ratio (CNR) given by $2\beta/r$. For Fig. 7, $\beta = 25$, $\sigma_2^2 = 100\sigma_1^2$, and $\epsilon = 0.9$. For the GSF-II and GSPF-II, the number of mixands was $G = 8$, and for GSPF-II, the used number of particles for updating each mixand was $M = 100$. Note that the GSPF-II had at least a gain of 1 dB for lower CNR, which increases for higher CNR.

C. Example 3—Joint Channel Estimation and Symbol Detection for Impulsive Fading Channel

In [2], a particle-based filter was presented for joint channel estimation and symbol detection. Transmitted data b_k are from a discrete complex set $\mathcal{L} = \{l_1, \ldots, l_{|\mathcal{L}|}\}$ over a frequency flat fading channel. The complex impulsive fading channel c_k is modeled as an autoregressive process $AR(p)$ driven by impulsive noise, modeled as a GM with two mixands. The baseband

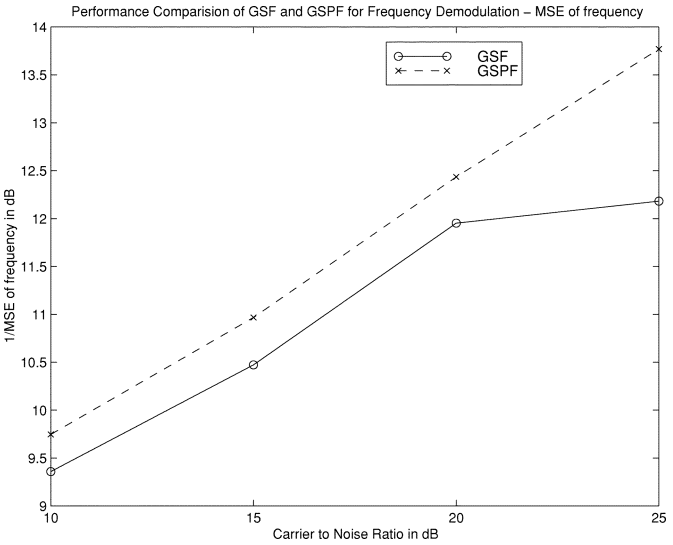

Fig. 7. Performance comparison of GSF-II and GSPF-II filters for frequency demodulation.

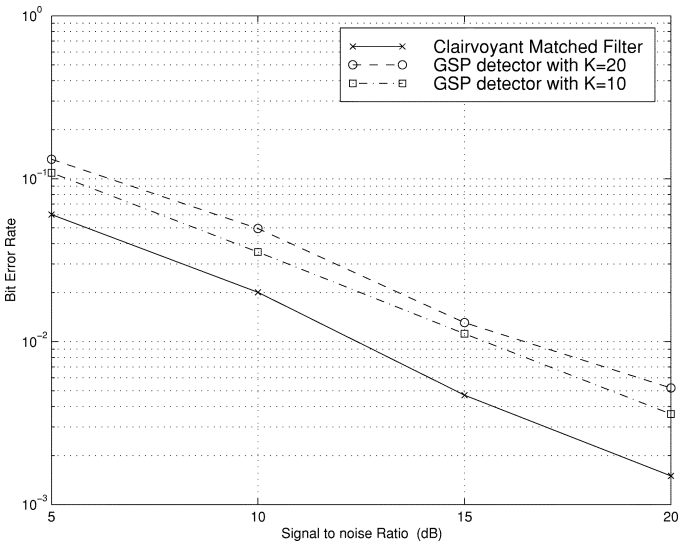

Fig. 8. Bit error rate performance for flat fading channel for BPSK signaling.

model of the above communication system can be represented as a dynamic state space (DSS) system

$$\begin{aligned} y_k &= \mathbf{b}_k^T \mathbf{c}_k + v_k \\ \mathbf{c}_k &= \mathbf{A}\mathbf{c}_{k-1} + \mathbf{u}_k \end{aligned} \tag{31}$$

where y_k is the sampled signal at the receiver, $\mathbf{b}_k^T = [b_k \quad 0 \quad 0 \quad \dots 0]$, $\mathbf{c}_k^T = [c_k \quad c_{k-1} \quad \dots \quad c_{k-p}]$, and

$$\mathbf{A} = \begin{bmatrix} a_1 & a_2 & \cdots & a_{p-1} & a_p \\ 1 & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & 0 \end{bmatrix}.$$

We assume that the components of $\mathbf{A}$ are *known*. The symbol v_k denotes complex white Gaussian noise whose real and imaginary parts are zero mean and iid with variance $\sigma_v^2/2$. The vector $\mathbf{u}_k^T = u_{1k}\mathbf{h}^T$, where $\mathbf{h}^T = [1 \quad 0 \quad \dots \quad 0]$ and u_{1k}, is white with variance σ_u^2 and distributed as a weighted sum of complex Gaussians given by

$$p(u_{1k}) = \epsilon \mathcal{N}\left(u_{1k}; 0, \sigma_1^2\right) + (1-\epsilon)\mathcal{N}\left(u_{1k}; 0, \sigma_2^2\right). \tag{32}$$

The choice of ϵ, σ_1^2 and σ_2^2 can be exploited to model and approximate a variety of channel characteristics. Here, the unknown or hidden variables are $\mathbf{x}_k = (\mathbf{c}_k, b_k)$. Given the data b_k, the model is linear with non-Gaussian process noise. The GSF-II is used to track the impulsive fading channel, while the data is estimated using SIS algorithm; see the references for details. The bit-error-rates versus signal-to-noise ratio performance curve is shown in Fig. 8. Pilot symbols are inserted every Kth symbol interval to resolve phase ambiguity. Performance is compared with the clairvoyant matched filter detector, which assumes knowledge of the true channel, and its performance represents an unachievable lower bound. Good performance was observed for channel tracking and detection.

VI. Conclusion

GSPFs that combine the principles of conventional Gaussian sum filters and particle filters were introduced. A general framework for DSS models with non-Gaussian noise was also presented, where the non-Gaussian DSS model was transformed to a weighted sum of parallel Gaussian noise DSS models. Based on EKFs, GPFs and GSPFs, we proposed algorithms for non-Gaussian models. Simulations show that in general, particle-based Gaussian mixture filters perform better than EKF-based Gaussian mixture filters.

References

[1] J. H. Kotecha and P. M. Djurić, "Gaussian particle filtering," *IEEE Trans Signal Processing*, vol. 51, pp. 2593–2602, Oct. 2003.

[2] ——, "Gaussian sum particle filtering detector for impulsive fading channels," in *Proc. IEEE—EURASIP Workshop Nonlinear Signal Image Processing*, Baltimore, MD, June 2001.

[3] ——, "Gaussian sum particle filtering for dynamic state space models," in *Proc. Int. Conf. Acoust., Speech Signal Processing*, Salt Lake City, UT, May 2001.

[4] D. L. Alspach and H. W. Sorenson, "Nonlinear bayesian estimation using Gaussian sum approximation," *IEEE Trans. Automat. Contr.*, vol. AC-17, pp. 439–448, Apr. 1972.

[5] R. A. Redner and H. F. Walker, "Mixture densities, maximum likelihood and the EM algorithm," *SIAM Rev.*, vol. 26, pp. 195–239, Apr. 1984.

[6] M. West, *Bayesian Forecasting and Dynamic Models*. New York: Springer-Verlag, 1997.

[7] D. Sengupta and S. Kay, "Efficient estimation of parameters for non-Gaussian autoregressive processes," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 37, pp. 785–794, June 1989.

[8] R. S. Blum, R. J. Kozick, and B. M. Sadler, "An adaptive spatial diversity receiver for nongaussian interference and noise," *IEEE Trans. Signal Processing*, pp. 2100–2112, Aug. 1998.

[9] B. D. O. Anderson and J. B. Moore, *Optimal Filtering*. Englewood Ciffs, NJ: Prentice-Hall, 1979.

[10] P. K. S. Tam and J. B. Moore, "A Gaussian sum approach to phase and frequency estimation," *IEEE Trans. Commun.*, vol. COM-25, pp. 935–942, Sept. 1977.

[11] H. W. Sorenson and D. L. Alspach, "Recursive bayesian estimation using Gaussian sums," *Automatica*, vol. 7, pp. 465–479, 1971.

[12] M. West, "On scale mixtures of normality," *Biometrika*, vol. 74, pp. 694–697, 1987.

[13] ——, "Modeling with mixtures (with discussion)," in *Bayesian Statistics 4*, J. M. Bernado, J. O. Berger, A. F. M. Dawid, and A. Smith, Eds. Oxford, U.K.: Oxford Univ. Press, 1991, pp. 503–524.

[14] ——, "Mixture models, Monte Carlo, Bayesian updating and dynamic models," *Comput. Sci. Statist.*, vol. 24, pp. 325–333, 1993.

[15] A. Spaulding and D. Middleton, "Optimum reception in an impulsive interference environment-part i: Coherent detection," *IEEE Trans. Commun.*, vol. COM-25, pp. 910–923, Sept. 1977.

[16] ——, "Optimum reception in an impulsive interference environment-part ii: Incoherent detection," *IEEE Trans. Commun.*, vol. COM-25, pp. 910–923, Sept. 1977.
[17] D. W. J. Stein, "Detection of random signals in Gaussian mixture noises," *IEEE Trans. Inform. Theory*, vol. 41, pp. 1788–1801, Nov. 1995.
[18] J. S. Liu and R. Chen, "Monte Carlo methods for dynamic systems," *J. Amer. Statist. Assoc.*, vol. 93, no. 443, pp. 1032–1044, 1998.
[19] R. Chen and J. S. Liu, "Monte Carlo mixture Kalman filters," *J. R. Statist. Soc.*, ser. B, 2000.
[20] R. Chen, X. Wang, and J. S. Liu, "Adaptive joint detection and decoding in flat-fading channels via mixture Kalman filtering," *IEEE Trans. Inform. Theory*, vol. 46, pp. 2079–2094, Sept. 2000.
[21] Y. Bar-Shalom and T. E. Fortmann, *Tracking and Data Association*. Orlando, FL: Academic, 1987.
[22] G. Kitagawa, "Non-Gaussian state-space modeling of nonstationary time series," *J. Amer. Statist. Assoc.*, vol. 82, no. 400, pp. 1032–1063, 1987.
[23] N. Gordon, D. Salmond, and C. Ewing, "Bayesian state estimation for tracking and guidance using the bootstrap filter," *J. Guidance, Contr., Dyn.*, vol. 18, no. 6, pp. 1434–1443, Nov.–Dec. 1995.
[24] E. R. Beadle and P. M. Djurić, "A fast weighted bayesian bootstrap filter for nonlinear model state estimation," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 33, pp. 338–342, Jan. 1993.
[25] G. Kitagawa, "Monte Carlo filter and smoother for non-Gaussian nonlinear state space models," *J. Comput. Graphical Statist.*, vol. 5, no. 1, pp. 1–25, 1996.
[26] O. R. Polk and S. C. Gupta, "Quasioptimum digital phase-locked loops," *IEEE Trans. Commun.*, vol. COM-21, pp. 75–82, Jan. 1973.

Jayesh H. Kotecha received the B.E. degree in electronics and telecommunications from the College of Engineering, Pune, India, in 1995 and the M.S. and Ph.D. degrees from the State University of New York at Stony Brook in 1996 and 2001, respectively.

Since January 2002, he has been with the University of Wisconsin, Madison, as a post-doctoral researcher. His research interests are primarily in the fields of communications, signal processing, and information theory. Presently, he is working in statistical signal processing, adaptive signal processing, and particle filters and physical layer aspects in multi-input multi-ouput communication systems, including channel modeling, tranceiver design, space-time coding, and capacity analysis.

Peter M. Djurić (SM'99) received the B.S. and M.S. degrees in electrical engineering from the University of Belgrade, Belgrade, Yugoslavia, in 1981 and 1986, respectively, and the Ph.D. degree in electrical engineering from the University of Rhode Island, Kingston, in 1990.

From 1981 to 1986, he was Research Associate with the Institute of Nuclear Sciences, Vinca, Belgrade, Yugoslavia. Since 1990, he has been with the State University of New York at Stony Brook, where he is Professor with 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 telecommunications, bio-medicine, and power engineering.

Prof. Djuric has served on numerous Technical Committees for the IEEE and SPIE and has been invited to lecture at universities in the United States and overseas. He was Associate Editor of the IEEE TRANSACTIONS ON SIGNAL PROCESSING, and currently, he is the Treasurer of the IEEE Signal Processing Conference Board and Area Editor of Special Issues of the IEEE SIGNAL PROCESSING MAGAZINE. He is also Vice Chair of the IEEE Signal Processing Society Committee on Signal Processing—Theory and Methods and a Member of the American Statistical Association and the International Society for Bayesian Analysis.