

# GAUSSIAN MIXTURE SIGMA-POINT PARTICLE FILTERS FOR SEQUENTIAL PROBABILISTIC INFERENCE IN DYNAMIC STATE-SPACE MODELS

Rudolph van der Merwe      Eric Wan

OGI School of Science & Engineering, Oregon Health & Science University  
 rvdmerwe@ece.ogi.edu    ericwan@ece.ogi.edu

## ABSTRACT

For sequential probabilistic inference in nonlinear non-Gaussian systems approximate solutions must be used. We present a novel recursive Bayesian estimation algorithm that combines an importance sampling based measurement update step with a bank of Sigma-Point Kalman Filters for the time-update and proposal distribution generation. The posterior state density is represented by a Gaussian mixture model that is recovered from the weighted particle set of the measurement update step by means of a weighted EM algorithm. This step replaces the resampling stage needed by most particle filters and mitigates the “sample depletion” problem. We show that this new approach has an improved estimation performance and reduced computational complexity compared to other related algorithms.

## 1. INTRODUCTION

Sequential probabilistic inference (SPI) is the problem of estimating the hidden states of a system in an optimal and consistent fashion as set of noisy or incomplete observations becomes available online. Examples of this include vehicle navigation and tracking, financial time-series prediction, and speech enhancement, to name but a few. This paper focuses specifically on discrete-time nonlinear dynamic systems that can be described by a *dynamic state-space model* (DSSM) as depicted in Figure 1. The hidden system state  $\mathbf{x}_k$ , with initial distribution  $p(\mathbf{x}_0)$ , evolves over time<sup>1</sup> as an unobserved first order Markov process according to the conditional probability density  $p(\mathbf{x}_k|\mathbf{x}_{k-1})$ . The observations  $\mathbf{y}_k$  are conditionally independent given the state and are generated according to the probability density  $p(\mathbf{y}_k|\mathbf{x}_k)$ . The DSSM can also be written as a set of system equations

$$\mathbf{x}_k = \mathbf{f}(\mathbf{x}_{k-1}, \mathbf{v}_{k-1}) \quad (\text{process equation}) \quad (1)$$

$$\mathbf{y}_k = \mathbf{h}(\mathbf{x}_k, \mathbf{n}_k) \quad (\text{observation equation}) \quad (2)$$

where  $\mathbf{v}_k$  is the process noise that drives the dynamic system through the nonlinear state transition function  $\mathbf{f}$ , and  $\mathbf{n}_k$  is the observation or measurement noise corrupting the observation of the state through the nonlinear observation function  $\mathbf{h}$ . The state transition density  $p(\mathbf{x}_k|\mathbf{x}_{k-1})$  is fully specified by  $\mathbf{f}$  and the process noise distribution  $p(\mathbf{v}_k)$ , whereas  $\mathbf{h}$  and the observation noise distribution  $p(\mathbf{n}_k)$  fully specify the observation likelihood  $p(\mathbf{y}_k|\mathbf{x}_k)$ .

In a Bayesian framework, the posterior filtering density  $p(\mathbf{x}_k|\mathbf{Y}_k)$  of the state given all the observations  $\mathbf{Y}_k =$

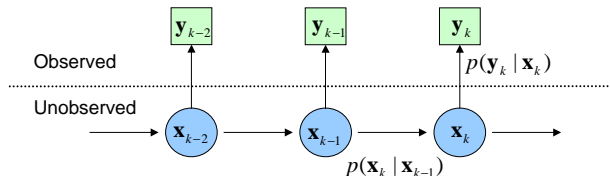


Fig. 1. Dynamic state space model.

$\{\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_k\}$  constitutes the complete solution to the sequential probabilistic inference problem, and allows us to calculate any “optimal” estimate of the state, such as the *conditional mean*  $\hat{\mathbf{x}}_k = E[\mathbf{x}_k|\mathbf{Y}_k] = \int \mathbf{x}_k p(\mathbf{x}_k|\mathbf{Y}_k) d\mathbf{x}_k$ . The optimal method to recursively update the posterior density as new observations arrive is given by the *recursive Bayesian estimation* algorithm

$$p(\mathbf{x}_k|\mathbf{Y}_k) = Cp(\mathbf{y}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{Y}_{k-1}) \quad (3)$$

where  $p(\mathbf{x}_k|\mathbf{Y}_{k-1}) = \int p(\mathbf{x}_k|\mathbf{x}_{k-1})p(\mathbf{x}_{k-1}|\mathbf{Y}_{k-1})d\mathbf{x}_{k-1}$  and  $C = (\int p(\mathbf{y}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{Y}_{k-1})d\mathbf{x}_k)^{-1}$ . Although this is the optimal recursive solution, it is usually only tractable for *linear, Gaussian* systems<sup>2</sup>, whereas for most general real-world (nonlinear, non-Gaussian) systems the multi-dimensional integrals are intractable and approximate solutions must be used. These include the well-known *Extended Kalman Filters* (EKF) [1], *Gaussian Sum Filters* [2], *Sigma-Point Kalman Filters*<sup>3</sup> [3, 4, 6, 5] and *Sequential Monte-Carlo Methods (Particle Filters)* [7].

Flaws in the EKF (inaccuracy, difficulty of implementation and tuning, divergence, etc.) led to the development of improved Gaussian approximate derivative-free Kalman filters called *Sigma-Point Kalman Filters* (SPKF). These filters have the same computational cost of the EKF, but do not need explicit calculation of system derivatives and consistently outperform the EKF in terms of estimation error, consistency and efficiency. See [5, 8] for a thorough exposition of SPKFs. The SPKF, like the EKF, still assumes a Gaussian posterior which can fail in certain nonlinear non-Gaussian problems with multi-modal and/or heavy tailed posterior distributions. The Gaussian sum filter (GSF) addresses this issue by approximating the posterior density with a finite Gaussian

<sup>2</sup>In the linear Gaussian case the recursive Bayesian solution is given by the well known *Kalman filter*[1].

<sup>3</sup>*Sigma-point Kalman filters* is a generalization of the family of Gaussian approximate nonlinear Kalman filters which include the Unscented Kalman Filter (UKF)[3], the Central Difference Kalman Filter (CDKF) [4] and other related algorithms [5, 6].

This work is supported in part by the following grants: NSF ECS-0083106, DARPA F33615-98-C-3516 & ONR N0014-02-C-0248.

<sup>1</sup> $k$  is the discrete time index.

mixture and can be interpreted as a parallel bank of EKFs. Unfortunately, due to the use of the EKF as a subcomponent, it also suffers from similar shortcomings as the EKF. Recently, particle based sampling filters have been proposed and used successfully to recursively update the posterior distribution using *sequential importance sampling and resampling* [7]. These methods (collectively called *particle filters*) approximate the posterior by a set of weighted samples without making any explicit assumption about its form and can thus be used in general nonlinear, non-Gaussian systems. They do, however, need to use a large number of particles for accurate and robust operation, which often make their use computationally expensive. Furthermore, they suffer from an ailment called “sample depletion” that can cause the sample based posterior approximation to collapse over time to a few samples. It was shown in [9] that by moving the particles to areas of high likelihood during the measurement update, the sample depletion problem can be mitigated leading to a significant increase in performance. This is accomplished by using a SPKF for the generation of the *proposal distribution* in the importance sampling measurement update step (See section 1.2 for a definition of the proposal distribution). The resulting filter is called the *Unscented Particle Filter* and has subsequently been generalized to a family of filters called *Sigma-Point Particle Filters* (SPPF) [8]. Although the SPPF has large estimation performance benefits over the standard PF, it still incurs a heavy computational burden since it has to run an  $\mathcal{O}(m_x^3)$  SPKF for each particle in the posterior state distribution.

In this paper, we present a novel algorithm to recursively update the posterior density called the *Gaussian Mixture Sigma-Point Particle Filter* (GMSPPF). This filter has equal or better estimation performance when compared to standard particle filters and the SPPF, at a largely reduced computational cost. The GMSPPF combines an *importance sampling* (IS) based measurement update step with a *SPKF based Gaussian sum filter* for the time-update and proposal density generation. The GMSPPF uses a finite Gaussian mixture model (GMM) representation of the posterior filtering density, which is recovered from the weighted posterior particle set of the IS based measurement update stage, by means of a *weighted Expectation-Maximization (WEM) algorithm*. The WEM stage also replaces the resampling step of the standard particle filter and mitigates the “sample depletion” problem. The three main algorithmic components used in the GMSPPF are briefly discussed below to provide some background on their use. In Section 2 we show how these three components are combined to form the GMSPPF algorithm.

### 1.1. SPKF based Gaussian mixture approximation

It can be shown [1] that any probability density  $p(\mathbf{x})$  can be approximated as closely as desired by a Gaussian mixture model (GMM) of the following form,  $p(\mathbf{x}) \approx p_G(\mathbf{x}) = \sum_{g=1}^G \alpha^{(g)} \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}^{(g)}, \mathbf{P}^{(g)})$ , where  $G$  is the number of mixing components,  $\alpha^{(g)}$  are the mixing weights and  $\mathcal{N}(\mathbf{x}; \boldsymbol{\mu}, \mathbf{P})$  is a normal distribution with mean vector  $\boldsymbol{\mu}$  and positive definite covariance matrix  $\mathbf{P}$ . Given Equations 1 and 2, and assuming that the prior density  $p(\mathbf{x}_{k-1}|\mathbf{Y}_{k-1})$  and system noise densities  $p(\mathbf{v}_{k-1})$  and  $p(\mathbf{n}_k)$  are expressed by GMMs, the following densities can also be approximated by GMMs: 1) the *predictive prior density*,  $p(\mathbf{x}_k|\mathbf{Y}_{k-1}) \approx p_G(\mathbf{x}_k|\mathbf{Y}_{k-1}) = \sum_{g'=1}^{G'} \alpha^{(g')} \mathcal{N}(\mathbf{x}; \tilde{\boldsymbol{\mu}}_k^{(g')}, \tilde{\mathbf{P}}_k^{(g')})$ , and 2) the *updated posterior density*,  $p(\mathbf{x}_k|\mathbf{Y}_k) \approx p_G(\mathbf{x}_k|\mathbf{Y}_k) = \sum_{g''=1}^{G''} \alpha^{(g'')} \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_k^{(g'')}, \mathbf{P}_k^{(g'')})$ , where  $G' = GI$  and  $G'' = G'IJ = GIJ$  ( $G$ ,  $I$  and  $J$  are the number of components in the

state, process noise, and observation noise GMMs respectively). The predicted and updated Gaussian component means and covariances of  $p_G(\mathbf{x}_k|\mathbf{Y}_{k-1})$  and  $p_G(\mathbf{x}_k|\mathbf{Y}_k)$  are calculated using the Kalman filter equations [1] (In the GMSPPF we use a bank of SPKFs). The mixing weight update procedure are shown in Section 2.1. It is clear that the number of mixing components in the GMM representation grows from  $G$  to  $G'$  in the predictive (time update) step and from  $G'$  to  $G''$  in the subsequent measurement update step. Over time, this will lead to an exponential increase in the total number of mixing components and must be addressed by a mixing-component reduction scheme (See Section 1.3 for how this is achieved in the GMSPPF).

### 1.2. Importance sampling (IS) based measurement update

Importance sampling is a Monte-Carlo method that represents a distribution  $p(\mathbf{x})$  by an empirical approximation based on a set of weighted samples (particles), i.e.  $p(\mathbf{x}) \approx \hat{p}(\mathbf{x}) = \sum_{l=1}^N w^{(l)} \delta(\mathbf{x} - \mathcal{X}^{(l)})$ , where  $\delta(\cdot)$  is the Dirac delta function, and the weighted sample set,  $\{w^{(l)}, \mathcal{X}^{(l)}; l = 1 \dots N\}$  are drawn from some related, easy-to-sample-from *proposal* distribution  $\pi(\mathbf{x})$ . The weights are given by  $w^{(l)} = \frac{p(\mathcal{X}^{(l)})/\pi(\mathcal{X}^{(l)})}{\sum_{l=1}^N p(\mathcal{X}^{(l)})/\pi(\mathcal{X}^{(l)})}$ . Given this, any estimate of the system such as  $E_p[g(\mathbf{x})] = \int g(\mathbf{x})p(\mathbf{x})d\mathbf{x}$  can be approximated by  $\hat{E}[g(\mathbf{x})] = \sum_{l=1}^N w^{(l)}g(\mathcal{X}^{(l)})$  [7]. Using the first order Markov nature of our DSSM and the conditional independence of the observations given the state, a recursive update formula (implicitly a nonlinear measurement update) for the importance weights can be derived [7]. This is given by

$$w_k^{(l)} = w_{k-1}^{(l)} p(\mathbf{y}_k|\mathbf{x}_k) p(\mathbf{x}_k|\mathbf{x}_{k-1}) / \pi(\mathbf{x}_k) \text{ for } \mathbf{x}_k = \mathcal{X}_k^{(l)}. \quad (4)$$

In the GMSPPF we use the GMM approximate  $p_G(\mathbf{x}_k|\mathbf{Y}_k)$  from the bank of SPKFs (see Sec.1.1) as the proposal distribution  $\pi(\mathbf{x}_k)$ . In [9] we showed that sampling from such a proposal (which incorporates the latest observation), moves particles to areas of high likelihood which in turn reduces the “sample depletion” problem. Furthermore we use the predictive prior distribution  $p_G(\mathbf{x}_k|\mathbf{Y}_{k-1})$  (see Sec.1.1) as a *smoothed* (over prior distribution of  $\mathbf{x}_{k-1}$ ) evaluation of the  $p(\mathbf{x}_k|\mathbf{x}_{k-1})$  term in the weight update equation. This is needed since the GMSPPF represents the posterior (which becomes the prior at the next time step) by a GMM, which effectively smoothes the posterior particle set by a set of Gaussian kernels. The IS based measurement update step presented here does not make any assumptions on the form of the posterior density which makes it inherently more powerful for general nonlinear, non-Gaussian systems, in comparison to the *linear* Kalman update which is only optimal in the linear Gaussian case.

### 1.3. Weighted EM for resampling and GMM recovery

The output of the IS-based measurement update stage is a set of  $N$  weighted particles, which in the standard particle filter is *resampled* in order to discard particles with insignificant weights and multiply particles with large weights. This step is needed to keep the variance of the particle set from growing too rapidly [7]. Unfortunately, resampling can also contribute to the “particle depletion” problem in cases where the measurement likelihood is very peaked, causing the particle set to collapse to multiple copies of the same particle [7]. Since the GMSPPF represents the posterior by a GMM, we can replace the resampling stage by a *weighted Expectation-Maximization* (EM) [10] step that directly recovers a maximum-likelihood  $G$ -component GMM fit to the set

of weighted samples. This implicitly smoothes over the posterior set of samples, avoiding the ‘‘particle depletion’’ problem, and at the same time the number of mixing components in the posterior is reduced to  $G$ , avoiding the exponential growth problem alluded to in Section 1.1.

## 2. THE GMSPPF ALGORITHM

The full GMSPPF algorithm will now be presented based on the component parts discussed above.

### 2.1. Time update and proposal distribution generation

At time  $k-1$ , assume the posterior state density is approximated by the  $G$ -component GMM

$$p_G(\mathbf{x}_{k-1}|\mathbf{Y}_{k-1}) = \sum_{g=1}^G \alpha_k^{(g)} \mathcal{N}(\mathbf{x}_{k-1}; \boldsymbol{\mu}_{k-1}^{(g)}, \mathbf{P}_{k-1}^{(g)}), \quad (5)$$

and the process and observation noise densities are approximated by the following  $I$  and  $J$  component GMMs respectively

$$p_G(\mathbf{v}_{k-1}) = \sum_{i=1}^I \beta_k^{(i)} \mathcal{N}(\mathbf{v}_{k-1}; \boldsymbol{\mu}_{v,k-1}^{(i)}, \mathbf{Q}_{k-1}^{(i)}) \quad (6)$$

$$p_G(\mathbf{n}_k) = \sum_{j=1}^J \gamma_k^{(j)} \mathcal{N}(\mathbf{n}_k; \boldsymbol{\mu}_{n_k}^{(j)}, \mathbf{R}_k^{(j)}) \quad (7)$$

Following the GSF approach of [2], but replacing the EKF with a SPKF, the output of a bank of  $G'' = GIJ$  parallel SPKFs are used to calculate GMM approximations of  $p(\mathbf{x}_k|\mathbf{Y}_{k-1})$  and  $p(\mathbf{x}_k|\mathbf{Y}_k)$  according to the pseudo-code given below. For clarity of notation define  $g' = g + (i-1)G$  and note that references to  $g'$  implies references to the respective  $g$  and  $i$ , since they are uniquely mapped. Similarly define  $g'' = g' + (j-1)GI$  with the same implied unique index mapping. Now,

1. For  $j=1 \dots J$ , set  $\tilde{p}_j(\mathbf{n}_k) = \mathcal{N}(\mathbf{n}_k; \boldsymbol{\mu}_{n,k}^{(j)}, \mathbf{R}_k^{(j)})$ . For  $i=1 \dots I$ , set  $\tilde{p}_i(\mathbf{v}_{k-1}) = \mathcal{N}(\mathbf{v}_{k-1}; \boldsymbol{\mu}_{v,k-1}^{(i)}, \mathbf{Q}_{k-1}^{(i)})$  and for  $g=1 \dots G$ , set  $\tilde{p}_g(\mathbf{x}_{k-1}|\mathbf{Y}_{k-1}) = \mathcal{N}(\mathbf{x}_{k-1}; \boldsymbol{\mu}_{k-1}^{(g)}, \mathbf{P}_{k-1}^{(g)})$ .
2. For  $g'=1 \dots G'$  use the time update step of a SPKF<sup>4</sup> (employing the system process equation (1) and densities  $\tilde{p}_g(\mathbf{x}_{k-1}|\mathbf{Y}_{k-1})$  and  $\tilde{p}_i(\mathbf{v}_{k-1})$  from above) to calculate a Gaussian approximate  $\tilde{p}_{g'}(\mathbf{x}_k|\mathbf{Y}_{k-1}) = \mathcal{N}(\mathbf{x}_k; \tilde{\boldsymbol{\mu}}_k^{(g')}, \tilde{\mathbf{P}}_k^{(g')})$  and update the mixing weights,  $\alpha_k^{(g')} = \alpha_{k-1}^{(g)} \beta_k^{(i)} / (\sum_{g=1}^G \sum_{i=1}^I \alpha_{k-1}^{(g)} \beta_k^{(i)})$ .

- (a) For  $g'' = 1 \dots G''$ , complete the measurement update step of each SPKF (employing the system observation equation (2), current observation  $\mathbf{y}_k$ , and densities  $\tilde{p}_{g'}(\mathbf{x}_k|\mathbf{Y}_{k-1})$  and  $\tilde{p}_j(\mathbf{n}_k)$  from above) to calculate a Gaussian approximate  $\tilde{p}_{g''}(\mathbf{x}_k|\mathbf{Y}_k) = \mathcal{N}(\mathbf{x}_k; \boldsymbol{\mu}_k^{(g'')}, \mathbf{P}_k^{(g'')})$ . Also update the GMM mixing weights,  $\alpha_k^{(g'')} = \alpha_k^{(g')} \gamma_k^{(j)} S_k^{(j)} / (\sum_{g'=1}^{G'} \sum_{j=1}^J \alpha_k^{(g')} \gamma_k^{(j)} S_k^{(j)})$ , where  $S_k^{(j)} = p_j(\mathbf{y}_k|\mathbf{x}_k)$  evaluated at  $\mathbf{x}_k = \tilde{\boldsymbol{\mu}}_k^{(g')}$  for the current observation,  $\mathbf{y}_k$ .

<sup>4</sup>The SPKF algorithm pseudo-code will not be given here explicitly. See [5, 6] for implementation details.

The predictive state density is now approximated by the GMM

$$p_G(\mathbf{x}_k|\mathbf{Y}_{k-1}) = \sum_{g'=1}^{G'} \alpha_k^{(g')} \mathcal{N}(\mathbf{x}_k; \tilde{\boldsymbol{\mu}}_k^{(g')}, \tilde{\mathbf{P}}_k^{(g')}) \quad (8)$$

and the posterior state density (which will only be used as the proposal distribution in the IS-based measurement update step) is approximated by the GMM

$$p_G(\mathbf{x}_k|\mathbf{Y}_k) = \sum_{g''=1}^{G''} \alpha_k^{(g'')} \mathcal{N}(\mathbf{x}_k; \boldsymbol{\mu}_k^{(g'')}, \mathbf{P}_k^{(g'')}) \quad (9)$$

### 2.2. Measurement update

1. Draw  $N$  samples  $\{\mathcal{X}_k^{(l)}; l = 1 \dots N\}$  from the proposal distribution  $p_G(\mathbf{x}_k|\mathbf{Y}_k)$  (Equation 9) and calculate their corresponding importance weights

$$\tilde{w}_k^{(l)} = \frac{p(\mathbf{y}_k|\mathcal{X}_k^{(l)})p_G(\mathcal{X}_k^{(l)}|\mathbf{Y}_{k-1})}{p_G(\mathcal{X}_k^{(l)}|\mathbf{Y}_k)} \quad (10)$$

2. Normalize the weights:  $w_k^{(l)} = \tilde{w}_k^{(l)} / \sum_{l=1}^N \tilde{w}_k^{(l)}$
3. Use a weighted EM (WEM) algorithm to fit a  $G$ -component GMM to the set of weighted particles  $\{w_k^{(l)}, \mathcal{X}_k^{(l)}; l = 1 \dots N\}$ , representing the updated GMM approximate state posterior distribution at time  $k$ , i.e.

$$p_G(\mathbf{x}_k|\mathbf{Y}_k) = \sum_{g=1}^G \alpha_k^{(g)} \mathcal{N}(\mathbf{x}_k; \boldsymbol{\mu}_k^{(g)}, \mathbf{P}_k^{(g)}). \quad (11)$$

The EM algorithm is ‘seeded’ by the  $G$  means, covariances and mixing weights of the prior state GMM,  $p_G(\mathbf{x}_{k-1}|\mathbf{Y}_{k-1})$ , and iterated until a certain convergence criteria (such as relative dataset likelihood increase) is met. Convergence usually occur within 4-6 iterations.

### 2.3. Inference

The conditional mean state estimate  $\hat{\mathbf{x}}_k = E[\mathbf{x}_k|\mathbf{Y}_k]$  can the corresponding error covariance  $\hat{\mathbf{P}}_k = E[(\mathbf{x}_k - \hat{\mathbf{x}}_k)(\mathbf{x}_k - \hat{\mathbf{x}}_k)^T]$  can be calculated in one of two ways: The estimates can be calculated before the WEM smoothing stage by a direct weighted sum of the particle set,

$$\hat{\mathbf{x}}_k = \sum_{l=1}^N w_k^{(l)} \mathcal{X}_k^{(l)} \quad \text{and} \quad \hat{\mathbf{P}}_k = \sum_{l=1}^N w_k^{(l)} (\mathcal{X}_k^{(l)} - \hat{\mathbf{x}}_k)(\mathcal{X}_k^{(l)} - \hat{\mathbf{x}}_k)^T \quad (12)$$

or after the posterior GMM has been fitted,

$$\hat{\mathbf{x}}_k = \sum_{g=1}^G \alpha_k^{(g)} \boldsymbol{\mu}_k^{(g)} \quad (13)$$

$$\hat{\mathbf{P}}_k = \sum_{g=1}^G \alpha_k^{(g)} (\mathbf{P}_k^{(g)} + (\boldsymbol{\mu}_k^{(g)} - \hat{\mathbf{x}}_k)(\boldsymbol{\mu}_k^{(g)} - \hat{\mathbf{x}}_k)^T)$$

Since  $N \gg G$ , the first approach is computationally more expensive than the second, but possibly generates better (lower variance) estimates, since it calculates the estimates before the implicit resampling of the WEM step. The choice of which method to use will depend on the specifics of the inference problem.

### 3. EXPERIMENTAL RESULTS

In this section the estimation performance and computational cost of the GMSPPF is evaluated on a state estimation problem and compared to that of two other particle filter solutions: the standard sampling importance-resampling particle filter (SIR-PF) [7] and the Sigma-Point Particle Filter (SPPF) [9, 5]. Due to space constraints, only one experiment is presented here, but the reader is referred to <http://cslu.ece.ogi.edu/mlsp> for more results. The experiment was done using Matlab and the *ReBEL Toolkit*<sup>5</sup>. For this experiment, a scalar time series was generated by the following process model:

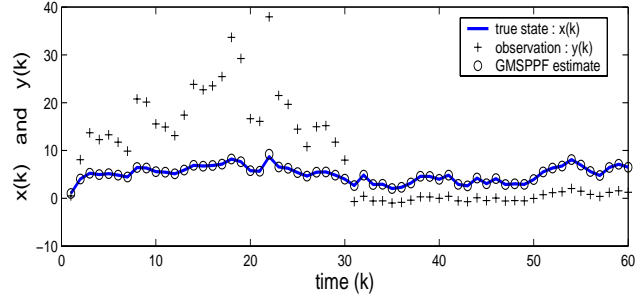
$$x_k = \phi_1 x_{k-1} + 1 + \sin(\omega\pi(k-1)) + v_k, \quad (14)$$

where  $v_k$  is a Gamma  $\mathcal{G}a(3, 2)$  random variable modeling the process noise, and  $\omega = 0.04$  and  $\phi_1 = 0.5$  are scalar parameters. A nonstationary observation model,

$$y_k = \begin{cases} \phi_2 x_k^2 + n_k & k \leq 30 \\ \phi_3 x_k - 2 + n_k & k > 30 \end{cases} \quad (15)$$

is used, with  $\phi_2 = 0.2$  and  $\phi_3 = 0.5$ . The observation noise,  $n_k$ , is drawn from a Gaussian distribution  $\mathcal{N}(n_k; 0, 10^{-5})$ . Figure 2 shows a plot of the hidden state and noisy observations of the time series. Given only the noisy observations  $y_k$ , the different filters were used to estimate the underlying clean state sequence  $x_k$  for  $k = 1 \dots 60$ . The experiment was repeated 100 times with random re-initialization for each run in order to calculate Monte-Carlo performance estimates for each filter. All the particle filters used 500 particles and residual resampling where applicable (SIR-PF and SPPF only). Two different GMSPPF filters were compared: The first, GMSPPF (5-1-1), use a 5-component GMM for the state posterior, and 1-component GMMs for the both the process and observation noise densities. The second, GMSPPF (5-3-1), use a 5-component GMM for the state posterior, a 3-component GMM to approximate the ‘‘heavy tailed’’ Gamma distributed process noise and a 1-component GMM for the observation noise density. The process noise GMM was fitted to simulated Gamma noise samples with an EM algorithm. Both GMSPPFs use Inference Method 1 (Equation 12) to calculate the state estimate. Table 1 summarizes the performance of the different filters. The table shows the means and variances of the mean-square-error of the state estimates as well as the average processing time in seconds of each filter. The reason why the standard PF performs so badly on this problem is due to the highly peaked likelihood function of the observations (arising from the small observation noise variance) combined with the spurious jumps in the state due to the heavy tailed process noise. This causes severe ‘‘sample depletion’’ in the standard PF that uses the transition prior  $p(x_k|x_{k-1})$  as proposal distribution. As reported in [9], the SPPF addresses this problem by moving the particles to areas of high likelihood through the use of a SPKF derived proposal distribution, resulting in a drastic improvement in performance. Unfortunately this comes at a highly increased computational cost. The GMSPPFs clearly have the same or better estimation performance (with reduced variance) as the SPPF but at a highly reduced computational cost. The best performance is achieved by the 5-3-1 GMSPPF that better models the non-Gaussian nature of the process noise.

<sup>5</sup>**ReBEL** is a **Matlab** toolkit for sequential Bayesian inference in general DSSMs. It contains a number of estimation algorithms including all those discussed in this paper as well as the presented GMSPPF. ReBEL is developed by the *MLSP Group* at OGI and can be freely downloaded from <http://cslu.ece.ogi.edu/mlsp/rebel> for academic and/or non-commercial use.



**Fig. 2.** Nonstationary nonlinear time series estimation experiment.

Algorithm	MSE (mean)	MSE (var)	Time (s)
SIR-PF	0.2517	4.9e-2	1.70
SPPF	0.0049	8.0e-5	35.6
GMSPPF (5-1-1)	0.0036	5.0e-5	1.04
GMSPPF (5-3-1)	0.0004	3.0e-6	2.03

**Table 1.** Estimation results averaged over 100 Monte Carlo runs.

### 4. CONCLUSION

We presented a novel algorithm for sequential inference in general nonlinear non-Gaussian DSSMs. It was shown that this algorithm not only outperforms standard particle filters, but has equal (or better) performance when compared to more advanced techniques such as the SPPF. In addition, this increased performance comes at a dramatic reduction in computational cost, making the GMSPPF a viable candidate for real-time systems. Furthermore, the GMSPPF mitigates the effects of sample depletion by combining the improved SPKF based proposal distribution of the SPPF, with a novel WEM based posterior density recovery and smoothing operation. This results in increased operational robustness.

### 5. REFERENCES

- [1] B. Anderson and J. Moore, *Optimal Filtering*, Prentice-Hall, 1979.
- [2] D. L. Alspach and H. W. Sorenson, ‘‘Nonlinear Bayesian Estimation using Gaussian Sum Approximation,’’ *IEEE Transactions on Automatic Control*, vol. 17, no. 4, pp. 439–448, 1972.
- [3] S. Julier, J. Uhlmann, and H. Durrant-Whyte, ‘‘A new approach for filtering nonlinear systems,’’ in *Proceedings of the American Control Conference*, 1995, pp. 1628–1632.
- [4] M. Norgaard, N. Poulsen, and O. Ravn, ‘‘New Developments in State Estimation for Nonlinear Systems,’’ *Automatica*, vol. 36, 2000.
- [5] E. A. Wan and R. van der Merwe, *Kalman Filtering and Neural Networks*, chapter 7 - The Unscented Kalman Filter, Wiley, 2001.
- [6] R. van der Merwe and E. Wan, ‘‘Efficient Derivative-Free Kalman Filters for Online Learning,’’ in *Proc. of ESANN*, Bruges, Apr. 2001.
- [7] A. Doucet, N. de Freitas, and N. Gordon, *Sequential Monte-Carlo Methods in Practice*, Springer-Verlag, April 2001.
- [8] R. van der Merwe, *Probabilistic Inference using Sigma-Point Kalman Filters*, Ph.D. thesis, OGI School of Science & Engineering, Oregon Health & Science University, January 2003, In preparation.
- [9] R. van der Merwe, N. de Freitas, A. Doucet, and E. Wan, ‘‘The Unscented Particle Filter,’’ in *Advances in Neural Information Processing Systems 13*, Nov 2001.
- [10] G.J. McLachlan and T. Krishnan, *The EM Algorithm and Extensions*, Wiley, 1997.