MIT Open Access Articles

Economical simulation in particle filtering using interpolation

The MIT Faculty has made this article openly available. ***Please share*** how this access benefits you. Your story matters.

Citation	Taylor, J.A., and F.S. Hover. “Economical Simulation in Particle Filtering Using Interpolation.” Information and Automation, 2009. ICIA ’09. International Conference On. 2009. 1326-1330. © 2009 IEEE.
As Published	http://dx.doi.org/10.1109/ICINFA.2009.5205122
Publisher	Institute of Electrical and Electronics Engineers
Version	Final published version
Accessed	Mon Nov 20 10:26:21 EST 2017
Citable Link	http://hdl.handle.net/1721.1/62228
Terms of Use	Article is made available in accordance with the publisher's policy and may be subject to US copyright law. Please refer to the publisher's site for terms of use.
Detailed Terms	

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

DSpace@MIT

Proceedings of the 2009 IEEE
International Conference on Information and Automation
June 22 -25, 2009, Zhuhai/Macau, China

Economical simulation in particle filtering using interpolation

Josh A. Taylor, *Student Member, IEEE,* Franz S. Hover, *Member, IEEE,*

***Abstract*— Sampling from the importance density is often a costly aspect of particle filters. We present a method by which to replace the most computationally expensive component of the importance density with an efficient approximation, thus allowing for the propagation of a large number of particles at reduced cost. The modification is implemented within auxiliary and regularized particle filters in a numerical example based on the Kraichnan-Orszag system.**

I. Introduction

Particle filters are a general solution to the recursive Bayesian estimation problem. The first practically applicable particle filter appeared in [1] under the name bootstrap filter. Because the algorithm was both general and simple, many variations soon appeared, some of the most successful of which are assembled in [2]. Particle filters approximate arbitrary distributions with sets of weighted points, i.e. particles, in contrast to Kalman filters [3], which propagate Gaussian distributions via their mean and covariance. The price of generality is computational efficiency - often, a large number of particles must be carried to obtain an accurate representation of the distribution.

Sampling from the importance density often accounts for the most computationally intensive component of a particle filter. In many cases this portion can be divided into a deterministic function evaluation followed by the incorporation of a noise sample. Usually, it is the deterministic portion which accounts for most of the computational load; recognizing this, we seek an efficient parameterization of the deterministic part of the importance density, which can then be inexpensively evaluated in place of the true function. Under mild smoothness assumptions, polynomial interpolation is an attractive means to this.

Our approach is as follows: the deterministic portion of the importance density is evaluated at special interpolation points (nodes), from which an accurate approximation to the true function (an interpolant) can be constructed. The interpolant can then be inexpensively evaluated numerous times, allowing for the propagation of a large number of particles at little cost. The appropriate noise sample can then be incorporated separately, as in the original importance density. In this paper we employ Barycentric Lagrange polynomial interpolation [4]–[7], although any method is valid. We have chosen to use Barycentric Lagrange polynomial interpolation for its speed and stability. Other approaches, such as splines and trigonometric interpolation [8], should be considered if more appropriate to a particular problem.

One-dimensional schemes are extended to multiple dimensions by Cartesian products; this approach suffers the 'Curse of Dimensionality', which is that convergence slows exponentially with increasing dimension. Due to this fact, we expect that the new filter will provide computational gains in situations with a relatively low number of dimensions but for which the importance density is expensive to evaluate. Various multidimensional schemes exist for exploiting dimensional structure, e.g. sparse grids [9], [10] and dimension adaptivity [11], the use of which may extend the applicability of the new filter to higher dimensions. However, there are many important problems of the scale developed here, including target tracking, estimation for control systems, real-time system identification, and mobile robot localization.

The paper is organized as follows: Section II outlines interpolation and the general recursive Bayesian estimation framework and discusses some basic particle filter formulations. Section III-A details the new filter and interpolation procedure. Section IV shows the results of numerical simulations on a Kraichnan-Orszag system based example.

II. Background

In this section, we provide background material needed for explaining how the interpolation filter is constructed, and for subsequently assessing its performance.

A. Interpolation

Interpolation [4], [5] is the approximation of function values using evaluations of that function at other points in the domain. Here we use Barycentric Lagrange polynomial interpolation [6], [7], which we prefer over other methods for its numerical stability and, more significantly, its speed. Using n nodes, evaluations of the interpolant require $O(n)$ operations, and all other operations requiring more than $O(n)$ operations (excluding function evaluations at the nodes) are function independent and hence precomputable; because importance densities often change through time, this is a necessary capability for this context.

The primary criterion for polynomial interpolation schemes is that the nodes have the asymptotic density $1/\sqrt{1-x^2}$ as $n \to \infty$. This distribution prevents the weights of equation (1) below from differing in size exponentially with n [6]. Node schemes satisfying this density are typically the roots of orthogonal polynomials. Two well-known sets of orthogonal polynomials are the Legendre and Chebyshev polynomials [5], [12], [13]. Legendre polynomial nodes are used in the example in Section IV.

J. A. Taylor and F. S. Hover are with the Department of Mechanical Engineering, Massachusetts Institute of Technology, Cambridge, MA, 02139 USA e-mail: jatl@mit.edu, hover@mit.edu.

This work was supported by the Office of Naval Research, Grant N00014-02-1-0623.

978-1-4244-3608-8/09/$25.00 © 2009 IEEE.

1) Barycentric Lagrange polynomial interpolation: For function f and node set $\chi = \{\chi^1, ..., \chi^n\}$, the one-dimensional barycentric interpolation formula of the second form is given by

$$I_n^f[\chi](x) = \frac{\sum_{i=1}^{n} \frac{b^i}{x - \chi^i} f(\chi^i)}{\sum_{i=1}^{n} \frac{b^i}{x - \chi^i}}, \quad b^i = \prod_{\substack{j=1 \\ j \neq i}}^{n} \frac{1}{\chi^i - \chi^j}. \quad (1)$$

The b^i are referred to as barycentric weights. $I_n^f[\chi]$ is itself a polynomial function, which is referred to as the interpolant.

2) Multivariate Interpolation: Univariate interpolation schemes can be extended in a general fashion to multiple dimensions via full grid tensor, or, for our purposes, Cartesian products. Abstractly, the d-dimensional, level-$\boldsymbol{n}$ interpolant may be written

$$I_{\boldsymbol{n}}^f[\boldsymbol{\chi}] = I_{n_1} \otimes \cdots \otimes I_{n_d}[\boldsymbol{\chi}]. \quad (2)$$

$\boldsymbol{n} = (n_1, ..., n_d)$ is a vector in the multivariate case; the interpolant may have different resolutions in different dimensions, and it should if a priori knowledge of the function suggests that not every dimension warrants the same computational effort. The corresponding multivariate barycentric formula of the second form for (2) is given by

$$I_{\boldsymbol{n}}^f[\boldsymbol{\chi}](\boldsymbol{x}) = \frac{\sum_{i_d=1}^{n_d} c_d^{i_d}(x_d) \times \cdots \times \sum_{i_1=1}^{n_1} c_1^{i_1}(x_1) f(\boldsymbol{\chi}^{i_1,...,i_d})}{\prod_{j=1}^{d} \sum_{i_j=1}^{n_j} c_j^{i_j}(x_j)}, \quad (3)$$

where

$$c_j^{i_j} = \frac{b_j^{i_j}}{x_j - \chi_j^{i_j}}, \quad j = 1, ..., d, \quad i_j = 1, ..., n_j. \quad (4)$$

The variables in (3) and (4) are written component-wise: x_j is the j^{th} component of $\boldsymbol{x}$, and $\chi_j^{i_j}$ is the j^{th} component of the node with multi-index $\{i_1, ..., i_d\}$. Although it is possible to write (3) as (1) using a single summation, the component-wise expression is more informative.

The computational cost of evaluating $I_{\boldsymbol{n}}^f[\boldsymbol{\chi}]$ is $O(\prod_{i=1}^{d} n_i)$, which is the total number of nodes; generally, this quantity increases exponentially with dimension, and should be a consideration when deciding whether or not to use interpolation.

B. Recursive Bayesian Estimation

We are interested in the distribution of the state $\mathbf{x}_k \in \mathbb{R}^{n_x}$, $k \in \mathbb{N}$ conditioned on all available measurements $\mathbf{y}_i \in \mathbb{R}^{n_y}$, $i = 1, ..., k$ of the discrete dynamical system

$$\begin{aligned} \mathbf{x}_k &= \mathbf{f}_k(\mathbf{x}_{k-1}, \mathbf{w}_k) \\ \mathbf{y}_k &= \mathbf{h}_k(\mathbf{x}_k, \mathbf{v}_k), \end{aligned} \quad (5)$$

where $\mathbf{f}_k : \mathbb{R}^{n_x} \times \mathbb{R}^{n_w} \to \mathbb{R}^{n_x}$ and $\mathbf{h}_k : \mathbb{R}^{n_x} \times \mathbb{R}^{n_v} \to \mathbb{R}^{n_y}$ are respectively the state transition and measurement functions, and $\mathbf{w}_k \in \mathbb{R}^{n_w}$ and $\mathbf{v}_k \in \mathbb{R}^{n_v}$ are independent, identically distributed process and measurement noise vectors. The posterior distribution, $p(\mathbf{x}_k|\mathbf{y}_{1:k})$, can be expressed via Bayes rule as

$$\begin{aligned} p(\mathbf{x}_k|\mathbf{y}_{1:k}) &= \frac{p(\mathbf{y}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{y}_{1:k-1})}{p(\mathbf{y}_k|\mathbf{y}_{1:k-1})} \\ &= \frac{p(\mathbf{y}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{y}_{1:k-1})}{\int p(\mathbf{y}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{y}_{1:k-1})d\mathbf{x}_k}. \end{aligned} \quad (6)$$

The Kalman filter is an analytical solution to this problem when $\mathbf{f}$ and $\mathbf{h}$ are linear and the state and noises have Gaussian distributions. In contrast, particle filters carry the statistics of $\mathbf{x}_k$ in a set of weighted particles. Let $\{\mathbf{x}_k^i, \lambda_k^i\}, i = 1, ..., n$ be a set of particles $\mathbf{x}$ with weights λ at time k, such that $\sum_i \lambda_k^i = 1$ for all k. This comprises a discrete probability mass function, which can approximate $p(\mathbf{x}_k|\mathbf{y}_{1:k})$ such that

$$p(\mathbf{x}_k|\mathbf{y}_{1:k}) = \lim_{n \to \infty} \sum_{i=1}^{n} \lambda_k^i \delta(\mathbf{x}_k - \mathbf{x}_k^i). \quad (7)$$

The basic particle filtering algorithm is known as sequential importance sampling [2], [14]. It is comprised of two steps: sampling, which usually entails simulations, and weight calculation. As it is, the sequential importance sampling algorithm is in general insufficient for most problems, namely because all weights except that of one particle will approach zero [15]. Resampling [1], [16], [17] is a straightforward solution to this, but in turn introduces sample impoverishment, a phenomenon in which all particles become identical.

Regularization [2], [18] addresses sample impoverishment by perturbing each resampled particle's position with samples from either an Epanechnikov (optimally) or Gaussian (simply) distribution. We use the acronym RPF to denote the regularized particle filter.

The auxiliary particle filter (ASIR) [2], [19], rather than resampling after weight calculation, computes the weights of a representative quantity for each particle (such as the mean or mode), and then resamples at the previous time step based on those weights. In other words, a characterization of each particle is drawn from the importance density to determine its quality, and then those particles of the highest quality are sampled from the proposal, thus removing poorly situated particles at the previous time step.

Many other improvements to the basic particle filter formulation exist [17], [20], [21], e.g. local linearization, Rao-Blackwellisation, and numerous techniques for improving importance densities. We consider only those discussed above for the purpose of demonstrating our approach, but note that most others are compatible as well.

III. The Interpolation Particle Filter

A. Main Filter Algorithm

This section presents the interpolation particle filter algorithm (IPF). We assume that sampling from the importance

density at time k can be written $\boldsymbol{x}_k = \boldsymbol{u}_k(\boldsymbol{g}_k(\boldsymbol{x}_{k-1}), \boldsymbol{z}_k)$. $\boldsymbol{g}_k$ is entirely deterministic and is subject to interpolation. $\boldsymbol{u}_k$ is a function which incorporates some noise sample $\boldsymbol{z}_k$ into a particle's position at time k. Note that $\boldsymbol{g}_k$ is often more than just the state transition; for example, if using the optimal importance density [22], the output of $\boldsymbol{g}_k$ contains both the mean and covariance of a particle.

The importance density is interpolated as follows. Interpolation nodes are scaled so as to cover all particles at time $k-1$, and the deterministic portion of the importance density $\boldsymbol{g}_k$ is then evaluated at these nodes. Following the procedure of Section II-A, an interpolant is constructed and evaluated at particle locations at time $k-1$. A noise sample $\boldsymbol{z}_k$ is then incorporated into the position of each interpolated particles via the function $\boldsymbol{u}_k$.

In our notation, $\boldsymbol{\chi}^j$, $j = 1, ..., n_N$ refers exclusively to nodes and $\boldsymbol{x}^i$, $i = 1, ..., n_P$ to particles. Underscored symbols correspond to non-dimensional quantities in the hypercube $[-1, 1]^d$, where d is the dimension of $\boldsymbol{x}_k$, and those not underscored refer to quantities in the system's state space. The absence of a superscript implies the entire set of nodes or particles. Fig. 1 gives a visualization of the method, written as an algorithm below.

Algorithm: IPF

Precompute

- Compute nodes $\{\underline{\boldsymbol{\chi}}^j \in [-1, 1]^d,\ j = 1, ..., n_N\}$ and barycentric weights for the orthogonal polynomials used.

Online

- Compute the width of the particles at time $k-1$:

$$\boldsymbol{S}_{k-1} = a \cdot diag\left[\max\left\{\max_{i=1,...,n_P} \boldsymbol{x}^i_{k-1} - \min_{i=1,...,n_P} \boldsymbol{x}^i_{k-1}, \boldsymbol{b}\right\}\right],$$

  where both maximums and the minimum are taken independently for each dimension of the state. Note that $\boldsymbol{S}_{k-1}$ is a $d \times d$ matrix. Guidelines on how the parameters a and $\boldsymbol{b}$ should be chosen are given below.
- Compute the center of the particles at time $k-1$:

$$\boldsymbol{\mu}_{k-1} = \frac{1}{2}\left(\max_{i=1,...,n_P} \boldsymbol{x}^i_{k-1} + \min_{i=1,...,n_P} \boldsymbol{x}^i_{k-1}\right) + \boldsymbol{c},$$

  again with the maximum and minimum taken independently for each dimension, so that $\boldsymbol{\mu}_{k-1}$ is a vector the same size as the state at time $k-1$. Again see below for guidelines on choosing $\boldsymbol{c}$.
- Map the particles at time $k-1$ to the hypercube $[-1, 1]^d$:

$$\underline{\boldsymbol{x}}^i_{k-1} = 2\boldsymbol{S}^{-1}_{k-1}(\boldsymbol{x}^i_{k-1} - \boldsymbol{\mu}_{k-1}).$$

- Transform the precomputed nodes so that the interpolant's domain encompasses all particles at time $k-1$:

$$\boldsymbol{\chi}^j_{pre} = \frac{1}{2}\boldsymbol{S}_{k-1}\underline{\boldsymbol{\chi}}^j + \boldsymbol{\mu}_{k-1}.$$

- Evaluate the deterministic portion of the importance density at transformed node locations:

$$\boldsymbol{\chi}^j_{post} = \boldsymbol{g}_k(\boldsymbol{\chi}^j_{pre}).$$

- Evaluate the interpolant $I_{n_N}[\boldsymbol{\chi}_{post}]$ at particle locations at time $k-1$. Evaluate the interpolant at each particle's location:

$$\boldsymbol{x}^i_{k*} = I_{n_N}[\boldsymbol{\chi}_{post}](\underline{\boldsymbol{x}}^i_{k-1}).$$

  I_{n_N} is defined in Section II-A.
- Incorporate the appropriate noise sample into interpolated particles with the function $\boldsymbol{u}_k$:

$$\boldsymbol{x}^i_k = \boldsymbol{u}_k(\boldsymbol{x}^i_{k*}, \boldsymbol{z}_k).$$

- Perform other particle filter steps, e.g. compute weights and resample the particles. Increment k and return to the first online step.

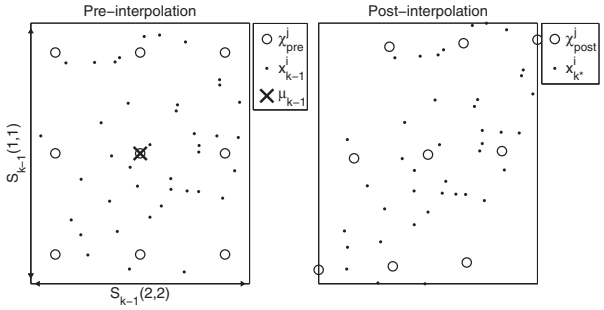

Fig. 1. Visualization of IPF interpolation procedure in two dimensions with Legendre nodes and $a = 1.1$.

The parameters a, $\boldsymbol{b}$, and $\boldsymbol{c}$ are necessary for stability of the IPF using barycentric Lagrange polynomial interpolation. a is a scalar constant which prevents scalings that result in points being interpolated on the boundary of the interpolant's domain. For Legendre or Stieltjes orthogonal polynomials, this can lead to compromised numerical stability, and for Chebyshev, division by zero. In our numerical examples, we set $a = 1.01$; any value slightly above one is sufficient. Although it is possible for a particle in the interior of the interpolant's domain to coincide with a node, this is highly unlikely, and it is not a scenario we prepare for.[1]

The parameters $\boldsymbol{b}$ and $\boldsymbol{c}$ prevent division by zero in the event of extreme sample impoverishment. Because all particles are coincident in this case, the interpolation region shrinks to a point and the $\boldsymbol{S}$ matrix, which must be inverted, becomes singular. $\boldsymbol{b}$ fixes a minimum size for the interpolation region and hence forces the $\boldsymbol{S}$ matrix (which is diagonal) to be invertible. $\boldsymbol{c}$ addresses a second issue arising from

[1]An obvious solution is to recognize that a true evaluation is available when interpolating on a node, and for many applications, this is sensible. Filtering often demands real-time implementation, and the logical statements necessary to handle this scenario add computational burden, particularly in higher dimensions. Furthermore, the loss in accuracy from interpolating over a slightly larger domain is usually negligible.

sample impoverishment. Even if the size of the interpolation region is greater than zero, all particles will be situated in its exact center. Because the center of the interpolation region is a node in most schemes, this leads to the same instability discussed in the previous paragraph. Offsetting the location of the region by $\boldsymbol{c}$ prevents this failure. In the implementations shown here we used $\boldsymbol{b}_i = \boldsymbol{c}_i = 0.01$, $i = 1, ..., d$. Note that $\boldsymbol{b}$ and $\boldsymbol{c}$, unlike a, affect the interpolation region additively, and so the size of the region should be a factor in choosing these parameters.

Note that the algorithm will vary depending on which type of particle filter it is used in. In Section IV, it is implemented within regularized and auxiliary particle filters. The regularized particle filter uses the prior $p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1})$ as its importance density. In this case, the deterministic portion is the state transition $\boldsymbol{f}_k$, and the noise sample is drawn from the process noise distribution. An interpolant $I_{n_N}[\boldsymbol{\chi}_{post}](\cdot)$ is constructed for $\boldsymbol{f}_k$ by evaluating $\boldsymbol{f}_k$ at the nodes, and is simply used in place of $\boldsymbol{f}_k$. The regularization component, because it is separate from the importance density, has no bearing on the interpolation. Auxiliary particle filters use the prior twice, and hence interpolation can again be applied by using the interpolant in place of evaluating the state transition.

IV. NUMERICAL EXAMPLE

We consider an example based on the Kraichnan-Orszag system [23], [24]. For this example, fifty Monte Carlo simulations of each filter were run; the performance metric is based on the mean of the absolute error over these runs.

The filters compared are the unscented Kalman filter (UKF), regularized particle filter (RPF), auxiliary particle filter (ASIR), and interpolation regularized and auxiliary particle filters (IRPF and IASIR). Regularized and auxiliary particle filters were chosen to represent conventional and interpolated filters for their simplicity and consistently strong performance in the literature [2]. The UKF was chosen over the EKF to be the representative Kalman filter because it is more robust and accurate on nonlinear systems, as well as easier to implement [25]. Each IRPF used n_N nodes and n_P particles. The IRPF is written IRPF-n_N/n_P, and the RPF's are written RPF-m, where m is the number of particles propagated by the particular RPF. The IASIR's have identical notation.

The equations for the Kraichnan-Orszag system [23], [24] are given by:

$$\begin{aligned} \dot{x}_1 &= x_2 x_3 + w_1 \\ \dot{x}_2 &= x_1 x_3 + w_2 \\ \dot{x}_3 &= -2x_1 x_2 + w_3, \\ \mathbf{y} &= \mathbf{x} + \mathbf{r}. \end{aligned} \tag{8}$$

Initial conditions of the true system are $\mathbf{x}(0) = [0, 1, 2]'$, $\mathbf{P}_0 = \mathbf{I}_3$, and Gaussian noises $\mathbf{w}$ and $\mathbf{r}$ both have zero-mean and covariance $\mathbf{I}_3$. Filter initial conditions were sampled from $N(\mathbf{x}(0), \mathbf{I}_3)$. Fig. 2 shows trajectories through time of each state for one realization. An Euler step was used with a time step of $dt = 0.001$ and measurements taken at every $100dt$.

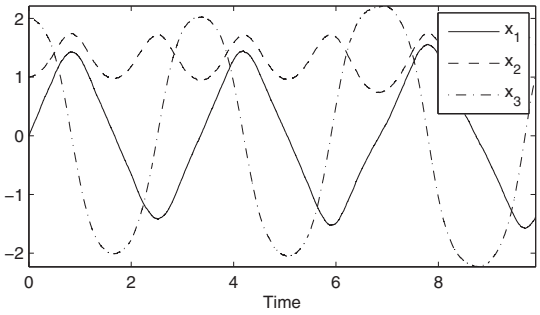

Fig. 2. A realization of the Kraichnan-Orszag system.

Because this system is a stochastic ordinary differential equation [26], [27] discretized over multiple time steps between measurements, process noise becomes an issue. Rigorous application of the RPF and ASIR is straightforward: the importance density is the prior, so simply adding scaled samples of the process noise to the simulated trajectories at each time step between measurements is sufficient. However, this importance density cannot be straightforwardly separated into a deterministic and stochastic portion, and thus interpolation cannot be rigorously applied. Instead we incorporate process noise heuristically by adding to each particle post interpolation a zero-mean Gaussian noise sample of variance $dt^2 \cdot n_s^2 \cdot \boldsymbol{Q}$, where n_s is the number of time steps between measurements.

An IRPF-8/50 and IASIR-8/50 are compared to eight, twenty, and fifty particle RPF's and ASIR's. The eight particle RPF's and ASIR's use the same number of particles as nodes used by the interpolation filters. The fifty particle filters are more expensive, highly accurate filters shown to demonstrate the validity of the interpolation filters.

In each RPF the prior $p(\mathbf{x}_k|\mathbf{x}_{k-1}^i)$ was used for the importance density. Perturbations for regularization were sampled suboptimally from the appropriate Gaussian distribution [18]. The ASIR's used the mode of each particle's evolution in generating proposal distributions.

Fig. 3 and Table I show mean absolute errors and the mean time taken for the online component of each filter. This is a system for which regularization provides significant improvement beyond that gained through resampling. The IRPF-8/50 performs nearly as well as the RPF-50 while much less time, and significantly better than the eight and twenty-particle RPF's. The IASIR-8/50 surprisingly performs better than any of the ASIR's, which do not perform well on this problem.

V. CONCLUSIONS

A method in which interpolation is used to reduce the cost of sampling from the importance density in particle filtering has been presented. In the new approach, a high-fidelity approximation is constructed from evaluations at interpolation nodes, which can then be interpolated as a means to efficiently propagating particles. A numerical example based on the Kraichnan-Orszag system was given in which

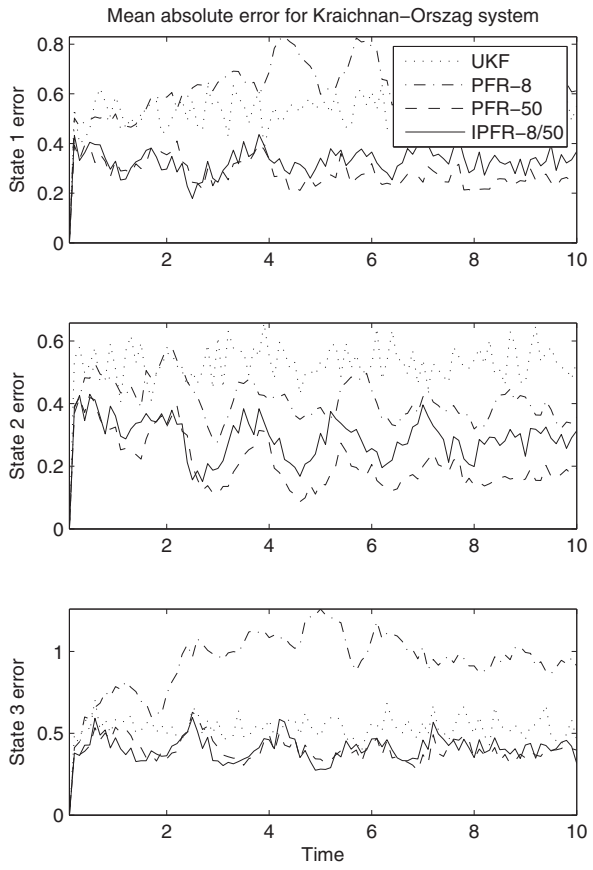

Fig. 3. Mean absolute errors for the Kraichnan-Orszag system.

TABLE I

MEAN ONLINE COMPUTATION TIME AND MEAN OVER TIME OF THE MEAN ABSOLUTE ERRORS FOR THE KRAICHNAN-ORSZAG SYSTEM.

Filter	x_1	x_2	x_3	Time (s)
RPF-8	0.61	0.40	0.93	3.05
RPF-20	0.42	0.28	0.61	7.56
RPF-50	0.27	0.20	0.39	18.49
IRPF-8/50	0.32	0.28	0.40	3.08
ASIR-8	0.98	0.62	1.44	5.08
ASIR-20	0.78	0.50	1.26	12.74
ASIR-50	0.61	0.32	0.91	29.82
IASIR-8/50	0.24	0.21	0.24	4.45

the new interpolation filters were shown to be significantly more efficient than a conventional particle filters. Filters with interpolation can provide substantial computational gains on systems with a relatively low number of states, but which have importance densities that are computationally expensive to evaluate.

REFERENCES

[1] N. Gordon, D. Salmond, and A. Smith, "Novel approach to nonlinear/non-gaussian bayesian state estimation," *Radar and Signal Processing, IEE Proceedings F*, vol. 140, no. 2, pp. 107–113, Apr 1993.

[2] S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A tutorial on particle filters for on-line non-linear/non-Gaussian Bayesian tracking," *IEEE Transactions on Signal Processing*, vol. 50, no. 2, pp. 174–188, Feb. 2002.

[3] A. Gelb, *Applied optimal estimation*. Cambridge, MA: MIT Press, 1974.

[4] P. J. Davis, *Interpolation and approximation*. New York: Dover, 1975.

[5] G. M. Phillips, *Interpolation and approximation by polynomials*. New York: Springer, 2003.

[6] J. Berrut and L. Trefethen, "Barycentric Lagrange interpolation," *SIAM Review*, vol. 46, no. 3, pp. 501–517, 2004.

[7] N. J. Higham, "The numerical stability of barycentric Lagrange interpolation," *IMA J Numer Anal*, vol. 24, no. 4, pp. 547–556, 2004. [Online]. Available: http://imajna.oxfordjournals.org/cgi/content/abstract/24/4/547

[8] J. Stoer and R. Bulirsch, *Introduction to numerical analysis*, 3rd ed. Springer, 2002.

[9] S. A. Smolyak, "Quadrature and interpolation formulas for tensor products of certain classes of functions," *Dokl. Akad. Nauk SSSR*, vol. 148, pp. 1042–1043, 1963, russian, Engl. Transl.: Soviet Math. Dokl. 4:240–243, 1963.

[10] V. Barthelmann, E. Novak, and K. Ritter, "High dimensional polynomial interpolation on sparse grids," *Advances in Computational Mathematics*, vol. 12, no. 4, pp. 273–288, March 2000.

[11] T. Gerstner and M. Griebel, "Dimension-adaptive tensor-product quadrature," *Computing*, vol. 71, no. 1, pp. 65–87, August 2003.

[12] P. J. Davis and P. Rabinowitz, *Methods of Numerical Integration*. New York: Academic Press, 1975.

[13] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes in C*, 2nd ed. Cambridge University Press, 1992.

[14] 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.

[15] 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.

[16] J. Carpenter, P. Clifford, and P. Fearnhead, "Improved particle filter for nonlinear problems," *Radar, Sonar and Navigation, IEE Proceedings -*, vol. 146, no. 1, pp. 2–7, Feb 1999.

[17] 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.

[18] C. Musso, N. Oudjane, and F. Legland, "Improving regularised particle filters," in *Sequential Monte Carlo Methods in Practice*, A. Doucet, N. de Freitas, and N. J. Gordon, Eds. New York: Springer, 2001.

[19] M. K. Pitt and N. Shephard, "Filtering via simulation: auxiliary particle filters," Tech. Rep.

[20] B. Ristic, S. Arulampalam, and N. Gordon, Eds., *Beyond the Kalman Filter: Particle Filters for Tracking Applications*. New York: Artech House Publishers, 2004.

[21] A. Doucet, N. De Freitas, and N. Gordon, Eds., *Sequential Monte Carlo methods in practice*. New York: Springer, 2001.

[22] A. Doucet, "On sequential Monte Carlo sampling methods for Bayesian filtering," 1998.

[23] S. A. Orszag and L. R. Bissonnette, "Dynamical properties of truncated Wiener-Hermite expansions," *Physics of Fluids*, vol. 10, no. 12, pp. 2603–2613, 1967.

[24] X. Wan and G. E. Karniadakis, "An adaptive multi-element generalized polynomial chaos method for stochastic differential equations," *J. Comput. Phys.*, vol. 209, no. 2, pp. 617–642, 2005.

[25] 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 Transactions on Automatic Control*, vol. 45, no. 3, pp. 477–482, 2000.

[26] P. E. Kloeden and E. Platen, *Numerical Solution of Stochastic Differential Equations (Stochastic Modelling and Applied Probability)*. Springer, November 2000.

[27] B. Oksendal, *Stochastic differential equations (3rd ed.): an introduction with applications*. New York, NY, USA: Springer-Verlag New York, Inc., 1992.