A Survey of Implicit Particle Filters for Data Assimilation

Alexandre J. Chorin, Matthias Morzfeld, Xuemin Tu

Abstract The implicit particle filter is a sequential Monte Carlo method for data assimilation. The idea is to focus the particles onto the high probability regions of the target probability density function (pdf) so that the number of particles required for a good approximation of this pdf remains manageable, even if the dimension of the state space is large. We explain how this idea is implemented, discuss special cases of practical importance, and work out the relations of the implicit particle filter with other data assimilation methods. We illustrate the theory with four examples.

1 Introduction

In many problems in science and engineering, e.g. in statistics, statistical signal processing, oceanography, meteorology, geomagnetics, econometrics, or finance, one wants to identify the state of a system from an uncertain model supplemented by a stream of noisy and incomplete data. The model is typically a Markovian state space model (often a discretization of a stochastic differential equation) and describes the state sequence $\{x^n;\ n \in N\}$, where x^n is a real, m-dimensional vector. For simplicity, we assume that the noise is additive, so that the model equations are

$$x^n = f^n(x^{n-1}) + v^{n-1}, \tag{1}$$

Alexandre Chorin
Department of Mathematics, University of California at Berkeley, and Lawrence Berkeley National Laboratory, e-mail: chorin@math.berkeley.edu

Matthias Morzfeld
Lawrence Berkeley National Laboratory, e-mail: mmo@math.lbl.gov

Xuemin Tu
Department of Mathematics, University of Kansas, e-mail: xtu@math.ku.edu

where f^n is an m-dimensional vector function, and $\{v^{n-1},\ n \in N\}$ is a sequence of independent identical distributed (i.i.d.) m-dimensional random vectors which, in many applications, are Gaussian vectors with independent components. One can think of the x^n as values of a process $x(t)$ evaluated at times $n\delta$, where δ is a fixed time increment. The probability density function (pdf) of the initial state x^0 is assumed to be known.

The model (1) is supplemented by an observation (or measurement) equation, which relates observations $\{b^n;\ n \in N\}$, where b^n is a real, k-dimensional vector and $k \leq m$, to the states x^n; we assume the observation equation is

$$b^n = h^n(x^n) + w^n, \tag{2}$$

where h^n is a k-dimensional vector function, and $\{w^n,\ n \in N\}$ is a k-dimensional i.i.d. process, independent of v^n. The model and the observation equations together constitute a hidden Markov state space model (HMM). To streamline our notation, we denote the state and observation sequences up to time n by $x^{0:n} = \{x^0, \cdots, x^n\}$ and $b^{1:n} = \{b^1, \cdots, b^n\}$, respectively.

Our goal is to estimate the state sequence $x^{0:n}$, based on (1) and (2) and we propose to use the minimum mean square error estimator $E\left[x^{0:n}|b^{1:n}\right]$ (see e.g. [1]). If f^n and h^n are linear functions and if, in addition, v^n and w^n are Gaussian random variables, this conditional mean can be computed by the Kalman filter (KF) [2–4]. The ensemble Kalman filter (EnKF) [5] uses the KF formalism, but updates the covariance matrix using an "ensemble of particles," i.e. by Monte Carlo simulations of the model (1). Because EnKF uses this ensemble approach, it can give good results even with nonlinear models (1), provided the nonlinearity is not too strong. Variational data assimilation methods find the mode of the target pdf, i.e. the most likely state given the data, and often use linearizations and Gaussian approximations to streamline the computations (see e.g [6–13] and Section 4 for more details on KF, EnKF and variational data assimilation).

In nonlinear, non-Gaussian situations, one can approximate the conditional mean using sequential Monte Carlo methods, called particle filters. Particle filters follow replicas of the system (called particles), whose empirical distribution weakly approximates the pdf $p(x^{0:n} \mid b^{1:n})$ (called the target density), and approximate the conditional mean by the weighted sample mean [14–16]. A standard particle filter, also called the sequential importance sampling with resampling (SIR) filter, first generates a set of particles $\{x_i^n\}$ from the model equation (1) [16–18] and then weighs these particles by the observation equation (2). The empirical distribution of the weighted particles forms a weak approximation of the target density at the current time step. One then removes particles with small weights (which contribute very little to the approximation of the target density) by "resampling", (see [14, 16] and the references therein for efficient resampling algorithms). The SIR filter is easy to implement, however after several time steps, often only a few of the particles carry a significant weight, which means that the weak approximation of the target density is poor. A cure here is to increase the number of particles (so that at least some carry a significant weight), however it has been shown that the number

of particles required can grow catastrophically with the state dimension m [19, 20]. Various strategies have been proposed to ameliorate this difficulty, and most of them focus on finding a better way to generate the samples [16, 21–24].

In what follows, we explain how implicit particle filters [25–29] tackle this problem. The basic idea is to first look for regions of high probability in the target density and then focus the particles onto these regions, so that only particles with significant weights are generated and the number of particles required remains manageable even if the state dimension is large. The high probability regions are identified by particle-by-particle minimizations, and the samples are obtained by solving data-dependent algebraic equations. The solutions of these equations define a high probability sample from the target density. Related work can be found in [30, 31].

The remainder of this paper is organized as follows. In section 2 we present the mathematical formulation of implicit particle filters and highlight special cases of interests. Several implementations of implicit particle filters are discussed in Section 3. In Section 4, the relations with other data assimilation methods are studied. We present four examples in Section 5 to demonstrate the efficiency and applicability of the implicit particle filter. Conclusions are offered in Section 6.

2 Implicit particle filters

The implicit particle filter is a sequential Monte Carlo method for data assimilation that uses importance sampling. In importance sampling one wants to find a weak approximation of the pdf, f, of a continuous random variable (called the target pdf), by generating weighted samples from a known density f_0 (called the importance function), see e.g. [1,33,34]. The weight of the sample X_j (obtained by sampling f_0),

$$w(X_j) = \frac{f(X_j)}{f_0(X_j)},$$

is the ratio of the target pdf and the importance function. The N samples X_j, $j = 1,\ldots,N$, with their weights w_j normalized so that their sum equals 1, form a weak approximation of the target pdf f such that

$$E_f\left[g(x)\right] = \int_{-\infty}^{\infty} g(x)f(x)dx \approx \sum_{j=1}^{N} g(X_j)w(X_j),$$

for all sufficiently smooth, scalar functions g, and where $E_f[g(x)]$ denotes the expected value of the function g with respect to the pdf f. The key to making importance sampling efficient is choosing a suitable importance function f_0, such that the weights vary little from one sample to the next. In data assimilation, the target density is the conditional pdf $p(x^{0:n} \mid b^{1:n})$. We now present the importance function generated by the implicit particle filter and describe why it produces samples with a small variance in the weights.

For simplicity of presentation, we assume that the model equation (1) is synchronized with the observations (2), i.e. observations b^n are available at every model step (it is not hard to drop this assumption, see Section 2.2). Using Bayes' rule and the Markovian property of the model, we obtain the recursive expression:

$$p(x^{0:n+1} \mid b^{1:n+1}) = p(x^{0:n} \mid b^{1:n}) p(x^{n+1} \mid x^n) p(b^{n+1} \mid x^{n+1}) / p(b^{n+1} \mid b^{1:n}). \quad (3)$$

At the current time $n+1$, the first term in (3) can be assumed to be known, because it is the result of our calculations at time n. The denominator is common to all particles and thus drops out in the importance sampling scheme (where the weights are normalized so that their sum equals 1).

Suppose that at time n, we have M samples $X_j = X_j^{0:n}$ with weights w_j^n, $j = 1,\ldots,M$, whose empirical distribution weakly approximates $p(x^{0:n} \mid b^{1:n})$. For each sample (particle), define a function F_j by

$$F_j(X_j^{n+1}) = -\log\left(p(X_j^{n+1} \mid X_j^n) p(b^{n+1} \mid X_j^{n+1})\right), \quad (4)$$

where we obtain the first term from the model equation (1), and the second from the observation equation (2). Note that the arguments of the functions F_j are the state variables of the jth particle at time $n+1$. The previous state of the jth particle, X_j^n and the current observation b^{n+1} are merely parameters.

By definition of the F_j's, the high probability region of the target density corresponds to the region around the global minimum of F_j. Thus, searching for the high probability region in the target density is equivalent to minimizing the functions F_j. We first assume that the functions F_j are convex, and carry out this minimization with standard techniques (e.g. Newton's method, quasi Newton methods, gradient descent, see [35, 36]) and obtain samples within these regions by solving the data dependent equations

$$F_j(X_j^{n+1}) - \phi_j = \frac{1}{2}\xi_j^T \xi_j, \quad (5)$$

where

$$\phi_j = \min F_j(X_j^{n+1}),$$

and where ξ_j is a realization of an easy-to-sample Gaussian reference variable $\xi \sim N(0,I)$, where $N(\mu,\Sigma)$ denotes a Gaussian pdf with mean μ and covariance matrix Σ, and I is the m-dimensional identity matrix. A Gaussian reference variable is chosen for simplicity of presentation and is by no means necessary (it may be suboptimal in some applications). More importantly, a Gaussian reference variable does not imply a Gaussianity or linearity assumption. All that is needed here is a reference variable with a high probability close to the origin, so that (5) maps the high probability region of the reference density to the high probability region of the target pdf.

The solutions of equation (5) define the samples X_j^{n+1}, however the solutions are not unique because (5) connects the m dimensional samples X_j^{n+1} to the m dimen-

sional reference variable ξ_j. The samples we find thus depend on the map $\xi_j \to X_j^{n+1}$ we chose to solve (5). To obtain a high probability sample X_j^{n+1}, we chose maps that satisfy the following conditions (see [26] for detailed explanation): the map should be (*i*) one-to-one and onto with probability one (so that the whole sample space is covered); (*ii*) smooth near the high-probability region of ξ (so that the weights do not vary unduly from particle to particle); and (*iii*) there should be an easy way to evaluate the Jacobians $J = \left|\det\left(\partial X_j^{n+1}/\partial \xi_j\right)\right|$ (for efficient implementation).

We will present specific choices for these maps in Section 3, but for now assume that we can compute these Jacobians. The probability of the sample we obtain is:

$$\begin{aligned} p(X_j^{n+1}) &= \frac{p(\xi_j)}{J} \propto \frac{\exp(-\xi_j^T \xi_j/2)}{J} = \frac{\exp(\phi_j - F_j(X_j^n))}{J} \\ &= \frac{\exp(\phi_j)}{J} p(X_j^{n+1} \mid X_j^n) p(b^{n+1} \mid X_j^{n+1}). \end{aligned}$$

Here we used (5) and (4) to obtain the third and the last equalities, respectively. The weight of the sample is thus

$$w_j^{n+1} \propto w_j^n \frac{p(X_j^{n+1} \mid X_j^n) p(b^{n+1} \mid X_j^{n+1})}{p(X_j^{n+1})} = w_j^n \exp(-\phi_j) J. \tag{6}$$

With these M samples X_j^{n+1} and the samples $X_j = X_j^{0:n}$ from the previous time step, we can form a sample $\hat{X}_j = X_j^{0:n+1}$ from $p(x^{0:n+1} \mid b^{1:n+1})$ with weight w_j^{n+1} as in (6). Once one has samples and weights, one can resample the pdf they define so as to remove some of the low probability particles and reset all the weights to $1/M$, see e.g. [14].

If the functions F_j are not convex, one can use the degeneracy of equation (5) to replace these functions by convex functions F_j^0 in (5) in such a way that the focusing effect is maintained; the weights have to be recomputed so that there is no bias, see e.g. [26].

Before focusing our attention on implementing the implicit particle filter (see Section 3), we give more details on the functional form of F_j for some cases of interest.

2.1 Linear observation function and Gaussian noise

To illustrate the method on a simple example, we assume that the observation equation is linear, i.e. $h^n(x) = Hx$, where H is a $k \times m$ matrix, and that the noise processes w^n and v^n in (1) and (2) are Gaussian with zero mean and known covariance, i.e $v^n \sim N(0,G)$, $w^n \sim N(0,Q)$, where G is an $m \times m$, real, symmetric positive definite (SPD) matrix and Q is a $k \times k$ SPD matrix. The functions F_j in (4) can now be written as

$$F_j(X_j^{n+1}) = \frac{1}{2}\left(X_j^{n+1}-\mu_j\right)^T \Sigma^{-1}\left(X_j^{n+1}-\mu_j\right)+\phi_j, \tag{7}$$

where

$$\Sigma^{-1} = G^{-1}+H^TQ^{-1}H, \tag{8}$$
$$K = HGH^T+Q, \tag{9}$$
$$\mu_j = \Sigma\left(G^{-1}f^n(X_j^n)+H^TQ^{-1}b^{n+1}\right), \tag{10}$$
$$\phi_j = \frac{1}{2}\left(b^{n+1}-Hf^n(X_j^n)\right)^T K^{-1}\left(b^{n+1}-Hf^n(X_j^n)\right). \tag{11}$$

It is clear that $\phi_j = \min F_j(X_j^{n+1})$, so that equation (5) becomes

$$\left(X_j^{n+1}-\mu_j\right)^T \Sigma^{-1}\left(X_j^{n+1}-\mu_j\right) = \xi_j^T\xi_j. \tag{12}$$

We can solve (12) by computing the Cholesky factorization $\Sigma = LL^T$, and putting

$$X_j^{n+1} = \mu_j + L\xi_j. \tag{13}$$

The Jacobian $J = |\det\left(\frac{\partial x}{\partial \xi}\right)| = |\det L|$ is constant (the same for all particles) and thus needs not be computed. By equation (6), the weights are

$$w_j^{n+1} \propto w_j^n \exp(-\phi_j). \tag{14}$$

Moreover, a simple calculation shows that

$$w_j^{n+1} \propto w_j^n p(b^{n+1} \mid X_j^n).$$

The above weights are the same as those of a filter that uses the optimal importance function $\hat{q} = p(x^{n+1}|x^n, b^{n+1})$ (see [18, 37] and the references therein) . "Optimal" here refers to "having minimum variance in the weights per particle," i.e. for a fixed X_j^{n+1}, the variance of w_j^{n+1} is zero. The implicit particle filter produces minimum variance weights in this sense if the observation equation (2) is linear and the observations are in sync with the model (see Section 4.2 for more details on the optimal importance function).

2.2 Sparse observations

The assumption that the observations are available at every model step can be relaxed. Let $r \geq 1$ be the number of model steps between observations (it is an easy exercise to adjust F_j for the case when the number of model steps between observations is not constant). The recursive formula (3) becomes

$$p(x^{0:r(n+1)} \mid b^{1:n+1}) = p(x^{0:rn} \mid b^{1:n})p(x^{rn+1} \mid x^{rn}) \cdots p(x^{r(n+1)} \mid x^{r(n+1)-1})$$

$$\times p(b^{n+1} \mid x^{r(n+1)})/p(b^{n+1} \mid b^{1:n}).$$

Again, suppose we have M weigthed samples $X_j = X_j^{0:rn}$, $j = 1,\ldots,M$, from $p(x^{0:rn} \mid b^{1:n})$ and, for each sample, we define a function F_j by

$$F_j(X_j^{rn+1},\ldots,X_j^{r(n+1)}) = -\log(p(X_j^{rn+1} \mid X_j^{rn}) \cdots p(X_j^{r(n+1)} \mid X_j^{r(n+1)-1}) \times p(b^{n+1} \mid X_j^{r(n+1)})).$$

With this F_j, one can follow the steps starting with equation (5) to obtain a sample from $p(x^{0:r(n+1)} \mid b^{1:n+1})$. Note that the functions F_j depend on rm variables (the components of $X_j^{rn+1:r(n+1)}$), so that we need to choose an rm-dimensional reference density. However, the general procedure for generating samples does not change when the observations are sparsely available in time.

2.3 Models with partial noise

Following [29], we consider the case of "partial noise," i.e. the model noise, $v^n \sim N(0,G)$, is Gaussian with singular covariance matrix G. Such models appear frequently, for example in the discretization of a stochastic partial differential equation (SPDE) driven by spatially smooth noise (see. e.g. [29, 38]). Another class of models with partial noise are stochastic dynamical equations supplemented by conservation laws. There is typically zero uncertainty in the conservation laws (e.g. the conservation of mass), so that the model is subject to partial noise [13]. This situation is similar to that of second-order (in time) stochastic differential equations (SDE), that appear, for example, in robotics. The second-order equation is often converted into a set of first-order equations, some of which are trivial (e.g. $u'' = f$ is converted into $v = u', v' = f$) and it is unphysical to inject noise into these trivial equations.

We use a linear coordinate transformation to diagonalize the state covariance matrix G [39] to obtain a canonical form of a model with partial noise from (1) and (2):

$$\hat{x}^{n+1} = \hat{f}(\hat{x}^n,\hat{y}^n) + \hat{v}^{n+1}, \quad \hat{v}^{n+1} \sim N(0,\hat{G}), \tag{15}$$

$$\hat{y}^{n+1} = g(\hat{x}^n,\hat{y}^n), \tag{16}$$

$$b^{n+1} = \hat{h}(\hat{x}^n,\hat{y}^n) + \hat{w}^n. \tag{17}$$

Here $\hat{x}^n$ is a p-dimensional column vector, $p < m$ is the rank of the state covariance matrix G, and $\hat{f}$ and $\hat{h}$ are p-dimensional, respectively k-dimensional vector functions; $\hat{G}$ is a non-singular, diagonal $p \times p$ matrix, $\hat{y}^n$ is a $(m-p)$-dimensional vector, and g is a $(m-p)$-dimensional vector function. For ease of notation, we drop the hats and, for convenience, we refer to the set of variables x^n and y^n as the "forced" and "unforced variables" respectively.

The key to filtering a model with partial noise is observing that the unforced variables at time $n+1$, given the state at time n, are not random. To be sure, y^n is random for any n due to the nonlinear coupling $g(x^n, y^n)$, but the conditional pdf $p(y^{n+1} \mid x^n, y^n)$ is the delta-distribution. For a given initial state x^0, y^0, the target density is

$$p(x^{0:n+1}, y^{0:n+1} \mid b^{1:n+1}) \propto p(x^{0:n}, y^{0:n} \mid b^{1:n}) \times p(b^{n+1} \mid x^{n+1}, y^{n+1}) p(x^{n+1} \mid x^n, y^n)$$

and the corresponding functions F_j as in (4) for models with partial noise are defined by

$$F_j(X_j^{n+1}) = -\log\left(p(b^{n+1} \mid X_j^{n+1}, Y_j^{n+1}) p(X_j^{n+1} \mid X_j^n, Y_j^n)\right).$$

With this F_j, we can use the implicit particle filter as described above.

The difference in filtering models with partial noise is that Y_j^{n+1} is fixed for each particle, because its previous state, (X_j^n, Y_j^n), is known, and because there is no noise in the equation for the unforced variables y^n. That means that the filter only updates the forced variables X_j^{n+1} when the observations b^{n+1} become available. The unforced variables Y_j^{n+1} are moved forward in time using the model, as they should be, since there is no uncertainty in y^{n+1} given x^n, y^n. Because the functions F_j depend on the forced variables only, the implicit particle filter reduces in dimension from m to p (the rank of the state covariance matrix G). This fact makes the implicit particle filter particularly effective for models with partial noise, because other filtering techniques, e.g. SIR, struggle to make direct use of the structure of the model.

2.4 Combined state and parameter estimation

Next, we consider models with unknown parameters, say $\theta \in R^l$, so that the model equation (1) becomes

$$x^n = f^n(x^{n-1}, \theta) + v^{n-1}.$$

The goal is to estimate both the states and the parameters, i.e. compute the conditional mean $E\left[x^{0:n}, \theta | b^{1:n}\right]$. We approximate the conditional mean by the sample mean, using weighted samples from $p(x^{0:n}, \theta | b^{1:n})$. A relatively simple way of estimating the parameters is to append a state equation for the parameters of the form

$$\theta^n = g^n(x^{n-1}, \theta^{n-1}) + v_\theta^{n-1}.$$

where g^n is a l-dimensional vector function, and v_θ^n is a l-dimensional i.i.d. random process. Defining an "extended" state $\hat{x} = (x, \theta)$, the implicit particle filter as described above can be applied to estimate $\hat{x}^{0:n}$ given the data $b^{1:n}$. The difficulty here lies in how to choose the dynamics g^n of the parameters θ. In particular, this approach is questionable if the parameters are known to be constant in time.

Alternatively, one can use the Markov property of the model and Bayes' rule to derive the recursive formula

$$p(x^{0:n+1}, \theta \mid b^{1:n+1}) = p(x^{0:n}, \theta \mid b^{1:n}) p(x^{n+1} \mid x^n, \theta) p(b^{n+1} \mid x^{n+1}) / p(b^{n+1} \mid b^{1:n}).$$

Following the now familiar steps and assuming that we have M samples from $p(x^{0:n}, \theta \mid b^{1:n})$, we can define the functions F_j by

$$F_j(X_j^{n+1}, \theta_j) = -\log\left(p(X_j^{n+1} \mid X_j^n, \theta_j) p(b^{n+1} \mid X_j^{n+1})\right).$$

With these F_j, we can again apply the implicit particle filter as described above. The details and numerical tests for this method applied to ecological models can be found in [40].

3 Implementations of the implicit particle filter

The set-up for the implicit particle filter as presented in the previous section is rather general, i.e. we have a lot of freedom in how we execute the various steps of the method. The crucial steps are (*i*) to find the minima of the functions F_j; and (*ii*) choose a map that solves the implicit equation (5). In practice, any one of the standard numerical optimization algorithms, e.g. Newton's method, quasi-Newton methods, trust-region methods, or gradient descent, can be used for the minimizations. However the applicability and efficiency of the minimization algorithms are problem dependent. For example, it may be very difficult to compute the Hessians of the functions F_j (for which the derivatives of the model equations are needed), so that Newton's method is not applicable but a quasi-Newton method can be used. If the state dimension is large, and memory limitations become an issue, then a gradient descent method may be the method of choice for minimization of F_j. We explain the minimization algorithms we use in the examples in Section 5. In the present section, we present two efficient ways of solving the implicit equation (5).

3.1 Solution of the implicit equation via quadratic approximation

Inspired by the simplicity of the case for which linear observations are available at each model step, one can try solving a quadratic equation, rather than the implicit equation (5). This idea was presented in [26], and is related to the quadratic expansion construction in [18] (see Section 4.2 for more details). To find a suitable quadratic equation, expand F_j to second order accuracy around its minimum:

$$F_j^0 = \phi_j + \frac{1}{2}(X_j^{n+1} - \mu_j)^T H_j (X_j^{n+1} - \mu_j),$$

where H_j is the Hessian of F_j, evaluated at the minimizer $\mu_j = \text{argmin } F_j$. With this F_j^0, define the equation

$$F_j^0(X_j^{n+1}) - \phi_j = \frac{1}{2}\xi_j^T \xi_j, \tag{18}$$

which can be solved efficiently using a Cholesky decomposition of the Hessian H_j. Let L_j be a Cholesky factor of $H_j = L_j L_j^T$. It is easy to verify that

$$X_j^{n+1} = \mu_j + L_j \xi^j,$$

solves (18) and that the Jacobian, $J = \left|\det(L_j)\right|$, of this map is easy to calculate, since it is the product of the diagonal elements of L_j. To avoid introducing any bias, one needs to account for the error we made by solving (18) rather than the true equation (5) in the weights. With this importance function, we obtain the weights

$$\begin{aligned} w_j^{n+1} &= w_j^n \frac{\exp(-F_j(X_j^{n+1}))}{\exp(-\xi_j^T \xi_j/2)} J, \\ &= w_j^n \exp(-\phi_j) \left|\det(L_j)\right| \exp(F_j^0(X_j^{n+1}) - F_j(X_j^{n+1})). \end{aligned}$$

This method of sampling has a geometric interpretation: the target pdf is approximated locally by a Gaussian centered at the mode of the target pdf, and with a covariance matrix that depends on the curvature of the target pdf at the minimum. Instead of finding samples from the target pdf, we obtain samples from the local Gaussian approximation, and account for this error in the weights through the factor $\exp(F_j^0(X_j^{n+1}) - F_j(X_j^{n+1}))$. If a Newton or quasi-Newton method is used for the minimization of F_j, then H_j, and often even L_j, are already available and this sampling method is easy to code and numerically efficient.

However, if the Gaussian approximation is not valid, for example, because the skewness in the target density is significant, the variance of the weights is increased, which should be avoided. In such cases, exact solution of (5) is advisable and we present an efficient method for doing so in the next subsection.

3.2 Solution of the implicit equation via random maps

Here we review the approach presented in [28] which solves (5) by the random change of variables (random map)

$$X_j^{n+1} = \mu_j + \lambda_j L_j^T \eta_j, \tag{19}$$

where λ_j is a scalar and $\eta_j = \xi_j / \sqrt{\xi_j^T \xi_j}$ is uniformly distributed on the unit m-sphere (or rm-sphere if the observations are sparse in time), and where $\mu_j = \text{argmin } F_j$. The square matrix L_j contains all prior information we have about F_j

and is deterministic and invertible. We will discuss the choice of L_j in more detail below.

By substitution of (19) into (5), we obtain a single algebraic equation in a single variable λ_j:

$$F_j(\mu_j + \lambda_j L_j^T \eta_j) - \phi_j = \frac{1}{2}\xi_j^T \xi_j.$$

The solution of the above equations defines X_j^{n+1} through (19). The geometric interpretation of this approach is that we choose a direction η_j at random, and then solve for λ_j, which tells us how far we need to search in this direction to hit the level set of F_j that is defined by the sample from the reference variable ξ.

To compute the weights in (6), we need to compute the Jacobian of the random map (19), which is:

$$J = 2|\det L_j|\ \rho_j^{1-m/2}\ \left|\lambda_j^{m-1}\frac{\partial \lambda_j}{\partial \rho_j}\right|,$$

where $\rho_j = \xi_j^T \xi_j$, and m is the dimension of the state space (if the observations are sparse, m in the above formula is to be replaced by rm). We refer to [28] for the details of this calculation, however note that the Jacobian is easy to evaluate, since the scalar derivative $\partial \lambda_j / \partial \rho_j$ can be computed efficiently by either using finite differences, or by evaluating

$$\frac{\partial \lambda_j}{\partial \rho_j} = \frac{1}{2(\nabla F_j) L^T \eta_j},$$

where ∇F_j is the gradient of F_j. The weight of the sample we obtained by solving (5) with the random map (19) is thus

$$w_j^n \propto w_j^{n-1} \exp(-\phi_j)|\det L_j|\ \rho_j^{1-m/2}\ \left|\lambda_j^{m-1}\frac{\partial \lambda_j}{\partial \rho_j}\right|. \tag{20}$$

We now discuss the choices of the matrix L_j. Suppose we apply our random map method to the special case we discussed in Section 2.1, i.e. the observation equation (2) is linear and in-sync with the model. With $L_j = I$, we find that

$$\lambda_j = \frac{\sqrt{\rho_j}}{\sqrt{\eta_j^T \Sigma^{-1} \eta_j}}$$

where Σ is given in (8). The weights of the particles become

$$w_j^{n+1} \propto w_j^n \exp(-\phi_j)(\eta_j^T \Sigma^{-1} \eta_j)^{-m/2},$$

and, since Σ is symmetric, are bounded above and below by the eigenvalues of Σ. The Jacobian J can vary dramatically from one sample to another, especially if the largest and smallest eigenvalues of Σ_j are separated by a large gap. If we choose L_j such that $\Sigma = L_j^T L_j$, we find $\lambda_j = \sqrt{\rho_j}$ and $J = |\det L_j|$. This Jacobian is constant and

need not be computed, and with this choice of L_j, we sample the optimal importance function, by solving (5) with the random map (19), see [28].

In the general case, we can use the information on the curvature of F_j we have in its Hessian H_j, by choosing L_j to be a Cholesky factor of this Hessian, i.e. $H_j = L_j^T L_j$. This choice should speed up the solution of (5), especially if F_j is quadratic or nearly so. This choice of L_j also suggests a "good" initialization for the numerical computation of the parameter λ_j in the random map. One can expect λ to be on the order of $\sqrt{\rho}$ and so that the iterative solution of (5) is initialized with $\lambda_j^0 = \sqrt{\rho_j}$.

4 Comparison with other sequential Monte Carlo schemes

We wish to compare the implicit particle filter with other data assimilation methods, and point out differences and similarities between these methods and the implicit particle filter.

4.1 Comparison with the SIR filter

The SIR filter [16] uses the model (1), i.e. $p(x^{n+1}|x^n)$, as the importance function, so that the weights are

$$w_{j,\mathrm{SIR}}^{n+1} \propto w_{j,\mathrm{SIR}}^{n} p(b^{n+1}|x^{n+1}). \tag{21}$$

The SIR filter can be formulated as an implicit particle filter with a different choice of ϕ_j in (5). Recall that for the implicit particle filter $\phi_j = \min F_j$ in (5). If we replace ϕ_j by

$$\phi_{j,\mathrm{SIR}} = -\log(p(b^{n+1}|X_j^{n+1})), \tag{22}$$

then (5) becomes

$$-\log(p(X_j^{n+1}|X_j^n)) = \frac{1}{2}\xi_j^T \xi_j.$$

For Gaussian model noise with covariance matrix Q, we thus have

$$(X_j^{n+1} - X_j^n)^T Q^{-1} (X_j^{n+1} - X_j^n) = \xi_j^T \xi_j, \tag{23}$$

which can be solved by

$$X_j^{n+1} = X_j^n + L^T \xi_j, \tag{24}$$

where L is a Cholesky factor of $Q = LL^T$. Note that the Jacobian of this map is constant for all particles (it is the determinant of L), and thus needs not be determined for computation of the weights. Moreover, computing X_j^{n+1} by (24) is equivalent to running the model forward for one time step, so that this implicit particle filter uses the same importance function as the SIR filter. The weights of the particles of this

implicit particle filter, given by (6), are therefore also the same as the weights of the SIR filter in (21).

The SIR filter is thus an implicit particle filter with ϕ_j in (5) replaced by $\phi_{j,\mathrm{SIR}}$ in (22). This observation illustrates why the SIR filter requires significantly more particles than the implicit particle filter (with $\phi_j = \min F_j$): choosing ϕ_j to be the minimum of F_j in (5) maps the high probability region of the reference variable ξ to the neighborhood of the minimum of F_j, which corresponds to the high probability region of the target pdf. Choosing ϕ_j as in (22) on the other hand maps the high probability region of the reference variable to the high probability region of the model (1). The overlap of the high probability region of the model with the high probability region of the target pdf can be very small, and in these cases, the SIR filter requires a large number of particles to provide accurate state estimates.

4.2 Comparison with optimal importance function filters

A crucial step in designing an efficient sequential Monte Carlo method is "a good choice" of the importance function. In the context of data assimilation, an "optimal" importance function can be found in [18, 37] and the references therein:

$$\hat{q} = p(X_j^{n+1}|b^{n+1}, X_j^n). \tag{25}$$

Here, "optimal" means that the variance of the weights of a given particle is zero (but not the variance of the weights of all the particles), and a particle filter which uses the optimal importance function is often called an optimal importance function filter. The weights of the optimal particles can be shown to be

$$\hat{w}_j^{n+1} \propto \hat{w}_j^n p(b^{n+1} \mid X_j^n).$$

If observations are available at every model step and if, in addition, the model and observation noise are Gaussian and the observation function h^n in (2) is linear, then the optimal importance function $\hat{q}$ is Gaussian with mean μ and covariance Σ as in (10) and (8) [18]. It was shown in Section 2.1 that in this case the implicit particle filter uses exactly this density as the importance function and that its weights are proportional to $p(b^{n+1} \mid X_j^n)$. Thus, for this special case, the implicit particle filter samples the optimal importance function and represents a convenient implementation of the optimal importance function filter.

In the general case, the optimal importance function is not readily available. One can rewrite (25) as

$$\hat{q} = \frac{p(b^{n+1}|X_j^{n+1})p(X_j^{n+1}|X_j^n)}{p(b^{n+1}|X_j^n)}, \tag{26}$$

and try to compute the denominator e.g. by Monte Carlo using

$$p(b^{n+1}|X_j^n) = \int p(b^{n+1}|X_j^{n+1}) p(X_j^{n+1}|X_j^n) dX_j^{n+1}.$$

which is often hard to do. However, even if $p(b^{n+1}|X_j^n)$ is available, sampling directly from the optimal importance function may be hard. In this case, one can define the function

$$l_j = \log p(X_j^{n+1}|b^{n+1}, X_j^n),$$

find its maximum $\lambda_j = \max l_j$, and expand l_j around its maximum:

$$l_j = \lambda_j + \frac{1}{2}\left(X_j^{n+1} - \gamma_j\right)^T H_j \left(X_j^{n+1} - \gamma_j\right),$$

where $\gamma_j = \text{argmax } l_j$ and H_j is the Hessian of l_j, evaluated at the maximum. The quadratic expansion suggests the (suboptimal) Gaussian importance function

$$q = N(\gamma_j, H_j^{-1}).$$

This approach has many similarities to the implicit particle filter when F_j in (5) is approximated by its quadratic expansion, also see Section 3.1. Specifically, both method require the solution of an optimization problem (to search for the high probability regions in the target pdf), and sampling from a multivariate Gaussian density. However, the implicit particle filter avoids the difficulties which arise in the optimal importance function filter from the need to compute $p(b^{n+1}|X_j^n)$ in (26). Since the computation of this term can be expensive, the implicit particle filter seems to be more efficient and easier to implement.

4.3 Comparison with the Kalman filter and with variational data assimilation methods

The Kalman filter (KF) is, strictly speaking, only applicable to linear systems (f and h are linear in (1) and (2)), driven by Gaussian noise (both v^n and w^n in (1) and (2) are Gaussian) [3, 4]. In this special case, the KF is widely used and efficient implementations are available for large, linear models. The implicit particle filter (with one particle) implements the KF for linear dynamics and Gaussian noise, because (8) and (10) become, upon rearrangement of the terms, the KF formulas.

For nonlinear, non-Gaussian HMM models, the extended Kalman filter (EKF) uses a linearization of the model and observation equation along with the standard KF formalism [41]. The ensemble Kalman filter (EnKF) implements the KF step using a covariance matrix that is approximated by Monte Carlo, i.e. by the sample covariance of many model runs. This step avoids the often costly computation of the covariance matrix in the KF formalism, and the EnKF can outperform the KF in linear systems with a very large state dimension [5]. Moreover, the EnKF injects the nonlinearity of the model into the KF formalism through the sample covariance matrix, but relies on a linearization of the observation equation. For this reason, both

EKF and EnKF can give good results if the nonlinearity is not too strong and if the number of model steps between observations is not too large. The implicit particle filter on the other hand tackles the full nonlinear problem and can outperform EnKF in nonlinear problems [29].

Variational data assimilation finds the most likely state given the data by finding the mode of the target pdf [6–13]. This mode can be found by minimizing a suitable cost function which is very similar to the functions F_j used in the implicit particle filter. Specifically, the cost function in weak constraint 4D-Var is the function F_j, with X_j^n being a variable (in the context of the filter, X_j^n is a parameter), and with an additional quadratic term, that corresponds to a Gaussian approximation of prior information on the state X_j^n (see e.g. [12]). Thus, the computational cost of the implicit particle filter is roughly the cost of a variational method, times the number of particles required (which should be a relatively small number), since the sampling can be carried out very efficiently once the minima of the F_j are obtained (see [47] for a detailed comparison of the implicit particle filter with variational data assimilation). It is also important to note that the minimization of the functions F_j and the sampling for the different particles can be carried out in parallel.

The main benefits of investing the additional effort and using the implicit particle filter rather than a variational method are: (*i*) a variational method computes the maximum a posteriori estimate (MAP), while the implicit particle filter approximates the minimum mean square error estimate (MMSE); in many situations, e.g. if the skewness in the target density is significant, the MMSE is a better estimator than the MAP [1]; (*ii*) the implicit particle filter provides a quantitative measure of the uncertainty of its state estimate (e.g. sample covariance or higher moments), while variational methods only provide a state trajectory, but no error bounds; and (*iii*) the implicit particle filter is a sequential method and thus it is relatively easy to assimilate more observations as they become available, while there are theoretical and practical issues with the sequential continuation of variational methods.

5 Applications

We demonstrate the applicability and efficiency of the implicit particle filter on four examples and provide details about the implementation in each of the examples. The first two examples demonstrate the applicability of the implicit particle filter to models that exhibit chaotic behaviors. We then consider data assimilation for a simplified model of the geomagnetic field coupled to the core velocity. Finally, we consider an ecological model and use the implicit particle filter to assimilate ocean color data obtained by NASA's SeaWiFS satellite.

5.1 The stochastic Lorenz attractor

This example is taken from [28]. We follow [42–44] and consider the stochastic Lorenz attractor [45]

$$dx = \sigma(y-x)dt + g_1 dW_1, \tag{27}$$
$$dy = (x(\rho - z) - y)dt + g_2 dW_2, \tag{28}$$
$$dz = (xy - \beta z)dt + g_3 dW_3, \tag{29}$$

with the standard parameters $\sigma = 10$, $\rho = 28$, $\beta = 8/3$, and initial conditions $x(0) = -5.91652$, $y(0) = -5.52332$, $z(0) = 24.5723$. The noise is chosen equally strong for all variables, so that $g_1 = g_2 = g_3 = g = \sqrt{2}$. We discretized the continuous equations by the Klauder-Petersen (KP) scheme [46]

$$x^{n+1,*} = x^n + \delta f(x^n) + g v_1,$$
$$x^{n+1} = x^n + \frac{\delta}{2}\left(f(x^n) + f(x^{n+1,*})\right) + g v_2,$$

where δ is the time step, $v_1, v_2 \sim N(0, \delta I)$ and where $f(\cdot)$ can be read off the Lorenz attractor (27) - (29). We are content here with an approximation with time step $\delta = 0.01$ (see [28] for more details on the discretization). Observations of all three state variables, corrupted by noise with variance 0.1, became available every 0.48 dimensionless time units (every 48 steps). This is a hard data assimilation problem and some filters miss transitions from one wing of the Lorenz butterfly to the other [43].

The minimization of the functions F_j was done using Newton's method, initialized by a free model run without noise (a larger gap between observations can cause problems here, however a more sophisticated initialization and a more robust minimization provide a cure, see [47]). Note that the argument of the F_j's are the state variables X_j^n, as well as the intermediate model steps $X_j^{n,*}$. The problem is thus of dimension 288: 3 dimensions for the Lorenz attractor, times 2 for the intermediate step x^* of the KP scheme, times 48 for the gap between observations. If the covariance matrix of the reference variable ξ is the identity matrix I, we are expressing a vector variable of small variance as a function of a unit reference variable, and this produces very small Jacobians J which can lead to underflow. One solution is to rescale ξ which, after all, is arbitrary. What we did instead is keep track of the logarithms of the weights rather than the weights themselves wherever we could; this solved the problem. At each assimilation step, we thus sampled a 288 dimensional standard normal variate xi_j (the reference variable) and computed the random direction $\eta_j = xi_j/\sqrt{\xi_j^T \xi_j}$ to be used in the random map (19). Because we used Newton's method for the minimization of F_j, the Cholesky factor L_j of the Hessian evaluated at the minimum was available and we used it in (19). Substitution of the map (19) into the algebraic equation (5) gave the required equation for λ_j, which we solved by Netwon's method. The iteration was initialized by choosing $\lambda_j^0 = \sqrt{\rho_j}$ and typ-

ically converged within 4-6 steps. Finally, we computed the weight of the particle using (20) and the numerical derivative $\partial\lambda/\partial\rho$, with a perturbation $\Delta\lambda = 10^{-5}\sqrt{\rho}$. We repeated this process for each particle and resample with "algorithm 2" in [16]. We decided to resample at every time an observation became available.

To compare the implicit particle filter with an SIR filter, we ran 1000 twin experiments. That is, we ran the model for 960 time steps and produced artificial observations (corrupted by the assumed noise). This model run was the reference we wished to reconstruct using the SIR and the implicit particle filters. For each experiment, the error at time $t^N = 9.6$, is measured by

$$e = ||x^N_{ref} - x^N||,$$

where the norm is Euclidean, x^N_{ref} is the reference state, and x^N is the reconstruction by a filter. We computed this error for each twin experiment, and, after running 1000 twin experiments, we computed the mean value of the error norms (mean error, for short) and the mean of the variance of the error norms (mean variance of the error, for short). The mean of the error norm is a better estimate for the errors than the mean error because it does not allow for cancellations. The mean variance of the error is not the variance of the mean, it is a fair estimate of the error in each individual run. Our results are in table 1.

Table 1 Filtering results for the Lorenz attractor.

# of Particles	Mean error / mean variance of the error	
	Implicit particle filter	SIR
5	0.2192/0.3457	- / -
10	0.2317/0.4905	0.9964/1.9970
20	0.1927/0.1646	0.5352/0.7661
50	- / -	0.4271/0.5445
100	- / -	0.2336/0.1229

The implicit particle filter yielded good results with 20 particles, while an SIR filter required about 100 particles for comparable accuracy (the results obtained for the SIR filter are in agreement with those previously reported in [43, 44]). The reason for the large difference in the number of particles required for these two filters is as follows. The large gap between observations implies that the SIR importance function and the target density become nearly mutually singular [19, 20, 26]. The "unguided" SIR particles are therefore very likely to become unlikely, and only very few of them carry a significant weight. The particles of the implicit particle filter on the other hand are guided towards the high probability regions because they are generated by solving (5), which incorporates information from the data. The larger number of particles required by the SIR filter thus indicates that the "focusing" of the particles towards the high probability regions of the target pdf was indeed achieved by the implicit particle filter.

The computational cost of these two filters is comparable in this example. The implicit particle filter requires fewer particles, but the computations for each particle are more expensive when compared with the SIR filter.

5.2 The stochastic Kuramoto-Sivashinsky equation

We follow [28] and consider data assimilation for the stochastic Kuramoto-Sivashinksy (SKS) equation

$$u_t + uu_x + u_{xx} + \nu u_{xxxx} = g\, W(x,t)$$

where $\nu > 0$ is the viscosity, g is a scalar and $W(x,t)$ is a space-time white noise process. The SKS equation is a chaotic SPDE that models laminar flames or reaction-diffusion systems [48, 49] and recently has been used as a large dimensional test-problem for data assimilation algorithms [44, 50].

We consider the m-dimensional Itô-Galerkin approximation of the SKS equation

$$dU = (\mathscr{L}(U) + \mathscr{N}(U))dt + g\, dW_t^m,$$

where U is a finite dimensional column vector whose components are the Fourier coefficients of the solution and where dW_t^m is a truncated cylindrical Brownian motion (BM) [38], obtained from the projection of the noise process $W(x,t)$ into the Fourier modes. Assuming that the initial conditions $u(x,0)$ are odd with $\tilde{U}_0(0) = 0$ and that dW_t^m is imaginary, all Fourier coefficients $U_k(t)$ are imaginary for all $t \geq 0$. Writing $U_k = i\hat{U}_k$ and subsequently dropping the hat gives

$$\mathscr{L}(U) = \mathrm{diag}(\omega_k^2 - \nu\omega_k^4)U,$$
$$\{\mathscr{N}(U)\}_k = -\frac{\omega_k}{2}\sum_{k'=-m}^{m} U_{k'}U_{k-k'},$$

where $\omega_k = 2\pi k/L$, $k = 1,\ldots,m$ and $\{\mathscr{N}(U)\}_k$ denotes the k^{th} element of the vector $\mathscr{N}(U)$. We choose a period $L = 16\pi$ and a viscosity $\nu = 0.251$, to obtain SKS equations with 31 linearly unstable modes. This set-up is similar to the SKS equation considered in [50]. With these parameter values there is no steady state as in [44]. We chose zero initial conditions $U(0) = 0$, so that the solution evolves solely due to the effects of the noise. To approximate the SKS equation, we keep $m = 512$ of the Fourier coefficients and use the exponential Euler scheme [51], with time step $\delta = 2^{-12}$ for time discretization (see [28] for details).

We are solving the SKS equations in Fourier variables, but we choose to observe in physical space (as is maybe physically reasonable). Specifically, we observe the solution $u(x,t)$ at $m/2$ equidistant locations and at every model step through the nonlinear observation operator $h(x) = x + x^3$. The solution of the algebraic equation (5) is easiest when the functions F_j is nearly diagonal, i.e., when its linearizations around a current state are nearly diagonal matrices; this requires in particular

that the variables that are observed coincide with the variables that are evolved by the dynamics. Observing in physical space while computing in Fourier space creates the opposite situation, in which each observation is related to the variables one computes by a dense matrix. This problem can be overcome using the random map algorithm presented in section 3.2.

The minimization of F_j was done using Newton's method, initialized by a free model run without noise. The Choleksy factor of the Hessian of F_j at the minimum was used as the matrix L_j in (19), and (5) was solved using this random map and a Newton iteration on λ_j. To test the implicit particle filter we ran twin experiments as in section 5.1. The error at time t^n is defined as

$$e^n = ||U^n_{ref} - U^n_F||$$

where the norm is the Euclidean norm and where U^n_{ref} denotes the set of Fourier coefficients of the reference run and U^n_F denotes the reconstruction by the filter, both at the fixed time t^n. Results of 500 twin experiments are shown in figure 1.

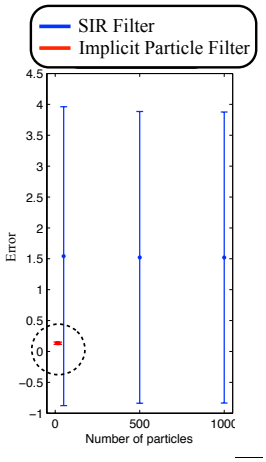

Fig. 1 Filtering results for the SKS equation: the error statistics are shown as a function of the number of particles for SIR (blue) and implicit particle filter (red). The error bars represent the mean of the errors and mean of the standard deviations of the errors.

We observe from figure 1 that the implicit particle filter requires far fewer particles than the SIR filter. Again, the example confirms that the implicit particle filter focuses its particles on the high probability regions of the target pdf. The focusing effect is more pronounced in the SKS equation than in the Lorenz attractor (see section 5.1), because the dimension of the state space of the SKS equation is 512, and therefore much larger than the dimension of the Lorenz attractor (dimension 3).

5.3 Application to geomagnetic data assimilation

The following example is taken from [29]. We wish to apply the implicit particle filter to a test problem in geomagnetic data assimilation, defined by two SPDE's

$$\partial_t u + u\partial_x u = b\partial_x b + \nu\partial_x^2 u + g_u \partial_t W(x,t),$$
$$\partial_t b + u\partial_x b = b\partial_x u + \partial_x^2 b + g_b \partial_t W(x,t),$$

where, $\nu = 10^{-3}$, $g_u = 0.01$, $g_b = 1$ are scalars, and where W is a spatially smooth stochastic process [29,52]. We consider the above equations on the strip $0 \le t \le 0.2$, $-1 \le x \le 1$ and with given boundary and initial conditions. Physically, u represents the velocity field at the core, and b represents the magnetic field of the earth. The model is essentially the model proposed in [52], but with additive noise

$$W(x,t) = \sum_{k=0}^{\infty} \alpha_k \sin(k\pi x) w_k^1(t) + \beta_k \cos(k\pi/2x) w_k^2(t).$$

where w_k^1, w_k^2 are independent BMs and where

$$\alpha_k = \beta_k = \begin{cases} 1, \text{ if } k \le 10, \\ 0, \text{ if } k > 10. \end{cases} \tag{30}$$

This simple noise model represents a spatially smooth noise which decreases in amplitude near the boundaries. The continuous equations are discretized using Legendre spectral elements in space, and an implicit-explicit first order scheme in time (see [29, 53–55]). We are content with an approximation that uses one Legendre element of order 300 for u and one for b, and a time step $\delta = 0.002$.

The data are the values of the magnetic field b, measured at 200 equally spaced locations in $[-1,1]$ and corrupted by noise:

$$z^l = Hb^{q(l)} + sV^l,$$

where $s = 0.001$ and where H is a $k \times m$-matrix that maps the numerical approximation b to the locations where data is collected. We consider data that are available every 10 model steps.

For our choice of α_k, β_k in (30), the state covariance matrices of the discrete equations are singular, i.e. the model is subject to partial noise (see section 2.3). Upon linear coordinate transformation, that diagonalizes the state covariance matrix, we obtain a model of the form (15)-(17). Because the second derivatives of the functions F_j are hard to calculate, we use a simple gradient descent algorithm with line search to carry out the minimization. As in previous examples, the minimization is initialized by a free model run. Since no information on the curvature of F_j is available, we set L_j in the random map (19) to the identity matrix. Equation (5) is then solved by a Newton iteration, initialized with $\lambda_j = 0$ (i.e. we start close to the minimum of F_j).

To assess the performance of the implicit particle filter, we ran 100 twin experiments. For each twin experiment, we calculate and store the error at $t = T = 0.2$ in the velocity, $e_u = ||u(x,T) - u_{Filter}(x,T)||$, and in the magnetic field, $e_b = ||b(x,T) - b_{Filter}(x,T)||$. After running the 100 twin experiments, we calculate the mean of the error norms and the variance of the error norms and scale the results by the mean of the norm of u and b respectively. Figure 2 shows the results.

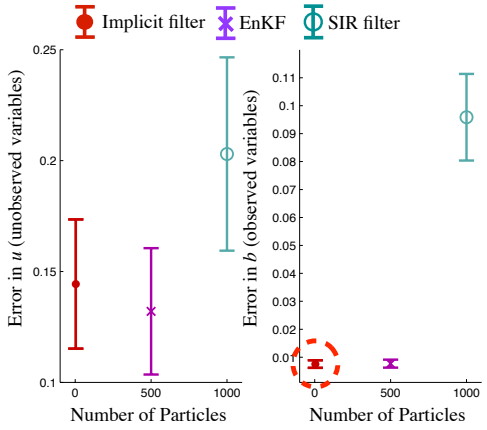

Fig. 2 Filtering results for the geomagnetic test problem. The errors of the implicit particle filter (red), EnKF (purple) and SIR filter (green) are plotted as a function of the number of particles. The error bars represent the mean of the errors and mean of the standard deviations of the errors.

It is evident from this figure that the implicit particle filter requires few particles to yield accurate state estimates with less than 1% error in the observed magnetic field b and less than 15% error in the unobserved velocity u. The SIR filter with 1000 particles gives significantly larger errors (about 10% in the observed variables b and 20% in the unobserved variable u) as well as much larger variances in the errors. The EnKF requires about 500 particles to achieve the accuracy of the implicit particle filter with only 4 particles. These examples thus provide further numerical evidence that the implicit particle filter can achieve the desired focusing effect and, as a consequence, is applicable to large dimensional data assimilation problems.

5.4 Assimilation of ocean color data from NASA's SeaWiFS satellite

We apply the implicit particle filter in its iterative implementation [27] (which is not discussed in this review) to a prototypical marine ecosystem model described in [56]. The model involves four state variables: phytoplankton P (microscopic

plants), zooplankton Z (microscopic animals), nutrients N (dissolved inorganics), and detritus D (particulate organic non-living matter). At the initial time $t = 0$ we have $P(0) = 0.125$, $Z(0) = 0.00708$, $N(0) = 0.764$, and $D(0) = 0.136$. The system is described by the nonlinear ordinary differential equations

$$\begin{aligned}
\frac{dP}{dt} &= \frac{N}{0.2+N}\gamma P - 0.1P - 0.6\frac{P}{0.1+P} + \mathcal{N}(0,\sigma_P^2),\\
\frac{dZ}{dt} &= 0.18\frac{P}{0.1+P}Z - 0.1Z + \mathcal{N}(0,\sigma_Z^2)\\
\frac{dN}{dt} &= 0.1D + 0.25\frac{P}{0.1+P}Z - \gamma P\frac{N}{0.2+N} + 0.05Z + \mathcal{N}(0\sigma_N^2)\\
\frac{dD}{dt} &= -0.1D + 0.1P + 0.18\frac{P}{0.1+P}Z + 0.05Z + \mathcal{N}(0,\sigma_D^2).
\end{aligned}$$

The variances of the noise terms are $\sigma_P = P(0)$, $\sigma_Z = 0.01Z(0)$, $\sigma_N = 0.01N(0)$, and $\sigma_D = 0.01D(0)$. We discretize the above equations with the stochastic Euler method (see [46]) with time step $\delta = 1$ day. The growth rate at time step t, γ_t, follows the recursion

$$\gamma_t = 0.14 + 3\Delta\gamma_t, \quad \text{where } \Delta\gamma_t = 0.9\Delta\gamma_{t-1} + \mathcal{N}(0,\sigma_\gamma^2),$$

with $\sigma_\gamma = 0.01$. The observations were obtained from NASA's SeaWiFS satellite ocean color images and provide a time series (190 data points from late 1997 to mid 2002) for the phytoplankton P by

$$\log b(t) = \log P(t) + \mathcal{N}(0,\sigma_b^2),$$

where $\sigma_b = 0.3$. We apply the implicit particle filter and the standard SIR filter to find a trajectory of the system consistent with the data. We observe that the implicit particle filter with only 10 particles does better than the SIR filter with 10 particles, in the sense that the filtered output matches the data better (see Figure 3). In fact the SIR filter requires about 100 particles to achieve the accuracy of the implicit particle filter with only 10 particles. This example thus provides further numerical evidence that the implicit particle filter can provide accurate state estimates with only a few particles. Moreover, the example shows that the implicit particle filter can work well with real data.

6 Conclusion

One of the barriers to the successful application of sequential Monte Carlo methods is sample impoverishment, i.e. the fact that the number of samples (particles) required can grow dramatically with the state dimension. The implicit particle filter is an attempt to overcome this problem. The main idea is to focus the particles so that they remain within the high probability regions of the target pdf. We have described

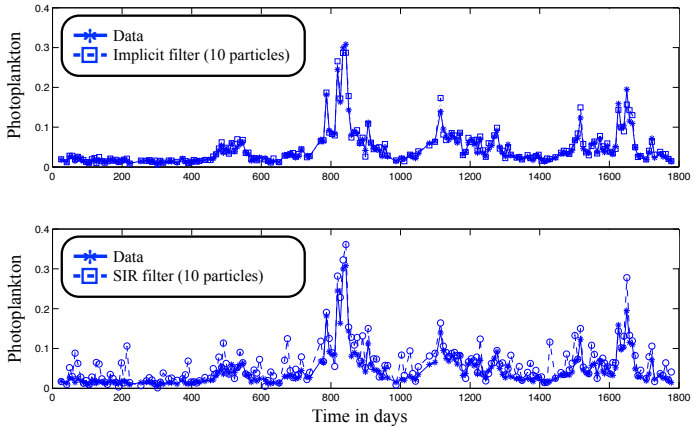

Fig. 3 The concentration of photoplankton as a function of time. Top: data and reconstruction by the implicit particle filter with 10 particles. Bottom: data and reconstruction by the SIR filter with 10 particles.

the mathematical background of this idea in detail and have showed that the regions of high probability can be identified by particle-by-particle minimization. Samples within these regions are obtained by solving data-dependent algebraic equations. We presented two effective algorithms for solving these equations and discussed the advantages of various numerical minimization algorithms in three examples. We have considered special cases of interest (e.g. partial noise), in both theory and in examples, and made connections with several other data assimilation methods (SIR, EnKF, and variational methods). Four numerical examples have been given to illustrate the theory and demonstrate the applicability and efficiency of the implicit particle filter. The examples indicate that the implicit particle filter indeed achieves the desired focusing effect, and that this focusing effect keeps the number of particles manageable even if the dimension of the state space is large.

Acknowledgements We would like to thank our collaborators Professor Robert Miller, Professor Yvette Spitz, and Dr. Brad Weir, at Oregon State University, for their comments and for very helpful discussions. This work was supported in part by the Director, Office of Science, Computational and Technology Research, U.S. Department of Energy under Contract No. DE-AC02-05CH11231, and by the National Science Foundation under grants DMS-0705910 and OCE-0934298.

References

1. Chorin A.J. and Hald, O.H. (2009). *Stochastic Tools in Mathematics and Science, Second Edition.* NY: Springer.
2. Gelb, A. (1974). *Applied optimal estimation.* MIT Press Cambridge.

3. Kalman, R.E. (1960). A New Approach to Linear Filtering and Prediction Theory, *Trans. ASME, Ser. D* (Journal of Basic Engineering) 82, 35–48.
4. Kalman, R.E. and Bucy, R.S. (1961). New Results in Linear Filtering and Prediction Theory, *Trans. ASME, Ser. D* (Journal of Basic Engineering) 83, 95–108.
5. Evensen, G. (2007). *Data Assimilation.* NY: Springer.
6. Talagrand O., and Courtier P. (1987). Variational assimilation of meteorological observations with the adjoint vorticity equation. I: Theory, *Q. J. R. Meteorol. Soc.* 113, 1311–1328.
7. Bennet A.F., Leslie L.M., Hagelberg C.R., and Powers P.E. (1993). A Cyclone prediction using a barotropic model initialized by a general inverse method, *Mon. Weather Rev.* 121, 1714–1728.
8. Courtier P., Thepaut J.N., and Hollingsworth A. (1994). A strategy for operational implementation of 4D-Var, using an incremental appoach, *Q. J. R. Meteorol. Soc.* 120, 1367–1387.
9. Courtier P. (1997). Dual formulation of four-dimensional variational assimilation, *Q. J. R. Meteorol. Soc.* 123, 2449–2461.
10. Talagrand O. (1997). Assimilation of Observations, an Introduction, *J. R. Meteorol. Soc. of Japan* 75(1), 191–209.
11. Zupanski D. (1997). A General Weak Constraint Applicable to Operational 4D-VAR Data Assimilation systems, *Q. J. R. Meteorol. Soc.* 125, 2274–2292.
12. Tremolet Y. (2006). Accounting for an imperfect mode in 4D-Var, *Q. J. R. Meteorol. Soc.* 621(132), 2483–2504.
13. Kurapov A. L. , Egbert G. D., Allen J. S., and Miller R. N. (2007). Representer-based variational data assimilation in a nonlinear model of nearshore circulation, *J. Geophy. Res.* 112, C11019.
14. Arulampalam M.S., Maskell S., Gordon N., Clapp T. (2002). A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking, *IEEE Trans. Signal Process.* 10, 197–208.
15. Del Moral, P. (2004). *Feynman-Kac Formulae.* NY: Springer.
16. Doucet A., de Freitas N. and Gordon N. (eds) (2001). *Sequential Monte Carlo Methods in Practice.* NY: Springer.
17. Gordon N.J., Salmon D.J., and Smith A.F.M. (1993). Novel approach to nonlinear/non-Gaussian Bayesian state estimation, *IEEE Proceedings F on Radar and Signal Processing* 140, 107–113.
18. Doucet A., Godsill S., and Andrieu C. (2000). On sequential Monte Carlo sampling methods for Bayesian filtering, *Statistics and Computing* 50, 174–188.
19. Bickel P., Li B., and Bengtsson T. (2008). Sharp failure rates for the bootstrap particle filter in high dimensions, *IMS Collections: Pushing the Limits of Contemporary Statistics: Contributions in Honor of Jayanta K. Ghosh*, 318-329.
20. C. Snyder C., Bengtsson T. , Bickel P., and Anderson J. (2008). Obstacles to high-dimensional particle filtering, *Mon. Wea. Rev.* , 136, 4629–4640.
21. Del Moral P. (1998). Measure-valued processes and interacting particle systems. Application to nonlinear filtering problems, *Annals of Applied Probability* 8(2), 438–495.
22. Weare J. (2009). Particle filtering with path sampling and an application to a bimodal ocean current model, *J. Comp. Phys.* 12(228), 4312–4331.
23. van Leeuwen P.J. (2009). Particle filtering in geophysical systems, *Mon. Weather Rev.* 137, 4089-4114.
24. van Leeuwen P.J. (2010). Nonlinear data assimilation in geosciences: an extremely efficient particle filter, *Q. J. R. Meteorol. Soc.* 136(653), 1991–1999.
25. Chorin A. and Tu X. (2009). Implicit sampling for particle filters, *Proc. Nat. Acad. Sc. USA*, 106, 17249–17254.
26. Chorin A., Morzfeld M., and Tu X. (2010). Implicit particle filters for data assimilation, *Commun. Appl. Math. Comput. Sci.*, 5(2), 221–240.
27. Chorin, A.J. and Tu, X (2012), An iterative implementation of the implicit nonlinear filter, *M2AN* 46, 535–543.
28. Morzfeld M., Tu X., Atkins E., and Chorin A. (2012). A random map implementation of implicit filters, *J. Comput. Phys.*, 231, 2049–2066.

29. Morzfeld M. and Chorin A. (2012). Implicit particle filtering for models with partial noise, and an application to geomagnetic data assimilation, *submitted for publication*.
30. Vanden Eijnden, E. and Weare, J. (2012). Data assimilation in the low noise accurate observation regime, with application to the Kuroshio current, *submitted for publication*.
31. Moselhy, T. and Marzouk, Y. (2012). Bayesian inference with optimal maps, *submitted for publication*.
32. Hammersley, J. M. and Handscomb, D.C., (1964). *MOnte Carlo Methods*, Methuen's Monographs on Applied Probability, Wiley, New York.
33. Hammersley, J. M. and Handscomb, D.C. (1964). *Monte Carlo Methods*, Methuen's Monographs on Applied Probability, Wiley, New York.
34. Kalos, M. H. and Whitlock, P. A. (1986). *MOnte Carlo Methods*, Wiley, New York.
35. Fletcher R. (1987). *Practical Methods of Optimization.* NY: Wiley.
36. Nocedal J., and Wright S.T. (2006). *Numerical Optimization (Second Edition).* NY: Springer.
37. Zaritskii V. S., and Shimelevich L. I. (1975). Monte Carlo technique in problems of optimal data processing, *Automation and Remote Control*, 12, 95–103.
38. Lord G.J., and Rougemont J. (2004). A numerical scheme for stochastic PDEs with Gevrey regularity, *IMA Journal of Numerical Analysis*, 24, 587–604.
39. Parlett B.N. (1998). *The symmetric eigenvalue problem.* Classics in Applied Mathematics, Vol. 20, Society for Industrial and Applied Mathematics, Philadelphia.
40. Weir B., Spitz Y.H., Miller R.N. (2012). Implicit estimation of ecological model parameters, *submitted for publication*.
41. Julier S.J., and Uhlmann J.K. (1997). A new extension of the Kalman filter to nonlinear systems, *International Symposium on Aerospace/Defense Sensing, Simulation and Controls*, 3.
42. Miller R., Ghil M., and Gauthiez F. (1994). Advanced data assimilation in strongly nonlinear dynamical systems, *J. Atmos. Sci.*, 51, 1037–1056.
43. Miller R., Carter E., and Blue S., (1999). Data assimilation into nonlinear stochastic systems, *Tellus*, 51A, 167–194.
44. Chorin A.J., and Krause P. (2004). Dimensional reduction for a Bayesian filter, *PNAS*, 101, 15013–15017.
45. Lorenz E.N. (1963). Deterministic nonperiodic flow, *J. Atmos. Sci.*, 20, 131–141.
46. Klauder J., and Petersen W. (1985). Numerical integration of multiplicative-noise stochastic differential equations, *SIAM J. Num. Anal.*, 22, 1153–1166.
47. Atkins A., Morzfeld M., and Chorin A. J. (2012). The implicit particle filter and its connection to variational data assimilation, *submitted for publication*.
48. Kuramoto Y., and Tsuzuki T. (1975). On the formation of dissipative structures in reaction-diffusion systems, *Progr. Theoret. Phys.*, 54, 687–699.
49. Sivashinsky G. (1977). Nonlinear analysis of hydrodynamic instability in laminar flames, Part I. Derivation of basic equations, *Acta Astronaut.*, 4, 1177–1206.
50. Jardak M., Navon I.M., and Zupanski M. (2009). Comparison of sequential data assimilation methods for the Kuramoto-Sivashinsky equation, *Int. J. Numer. Methods Fluids*, 62, 374–402.
51. Jentzen A., and Kloeden P.E. (2009). Overcoming the order barrier in the numerical approximation of stochastic partial differential equations with additive space-time noise, *Proc. R. Soc. A*, 465,649–667.
52. Fournier A., Eymin C., and Alboussiere T. (2007). A case for variational geomagneitic data assimilation: insights from a one-dimensional, nonlinear, and sparsely observed MHD system, *Nonlinear Proc. Geoph.*, 14, 163–180.
53. Kloeden P.E., and Platen E. (1999). *Numerical solution of stochastic differential equations.* NY: Springer.
54. Canuto C. , Hussaini M. Y., Quarteroni A., and Zang T. A. (2006). *Spectral Methods. Fundamentals in Single Domains* Berlin (Germany): Springer.
55. Deville M. O., Fischer P. F., and Mund E. H. (2006). *Higher-Order Methods for Incompressible Flow* Oxford UK: Cambridge University Press.
56. Dowd, M. (2006), A sequential Monte Carlo approach for marine ecological prediction, *Environmetrics* 17, 435–455.