Stochastic Methods for Sequential Data Assimilation in Strongly Nonlinear Systems

DINH TUAN PHAM

Laboratoire de Modélisation et Calcul, INRIA–CNRS–INPG–UJF, Grenoble, France

(Manuscript received 4 November 1999, in final form 25 May 2000)

ABSTRACT

This paper considers several filtering methods of stochastic nature, based on Monte Carlo drawing, for the sequential data assimilation in nonlinear models. They include some known methods such as the particle filter and the ensemble Kalman filter and some others introduced by the author: the second-order ensemble Kalman filter and the singular extended interpolated filter. The aim is to study their behavior in the simple nonlinear chaotic Lorenz system, in the hope of getting some insight into more complex models. It is seen that these filters perform satisfactory, but the new filters introduced have the advantage of being less costly. This is achieved through the concept of second-order-exact drawing and the selective error correction, parallel to the tangent space of the attractor of the system (which is of low dimension). Also introduced is the use of a forgetting factor, which could enhance significantly the filter stability in this nonlinear context.

1. Introduction and motivation

The purpose of this work is to propose some data assimilation methods and investigate their behavior and that of some others, in the context of highly nonlinear systems. Specifically we shall consider the particle filter (Del Moral and Salut 1995; Del Moral et al. 1995), the ensemble Kalman filter (Evensen 1994, 1997a,b; Evensen and van Leeuwen 1996, 2000; Burgers et al. 1998), and some new filters that we introduce: the second-order-exact ensemble Kalman filter and the singular extended interpolated filter (Pham 1997; Pham et al. 1998a). All these methods are sequential and of stochastic nature, relying on Monte Carlo drawing to mimic the statistical behavior of the system. They therefore have some similarities, which will be made clear. Our filters have been developed with applications to meteorology and oceanography in mind so that the computational cost is of great concern. But here we focus on the nonlinearity aspect of the system and try to see if they could perform reasonably well in a such a context. To avoid excessive computational cost (especially in the case of the particle and ensemble Kalman filters), we shall limit ourselves to the simple Lorenz model (Lorenz 1963). This model is described simply by an ordinary system of differential equation in three dimensions and yet has some connection with atmospheric flows. More importantly, it exhibits two main features of nonlinear dynamics: the existence of an attractor and chaos. The first means that the system state will approach a special subset of the phase space, called attractor, regardless of the initial state (provided that it is not too far from the attractor), while the second means that two nearby states, however close, will diverge in time; hence, the system state in the far future is unpredictable.

Our works have been influenced by the paper of Miller et al. (1994) in which the data assimilation in the Lorenz system is investigated. But these authors focus on the extended Kalman filters and the variational methods while here we focus on newer stochastic techniques. However, we have recently become aware of the paper by Evensen (1997a) in which the ensemble Kalman filter has been studied through a similar design. A further paper of Miller et al. (1999) has also appeared during the revison of this work, in which the authors also consider nonlinear filtering of the Lorenz equations, using numerical and Monte Carlo calculation (the last has some similarities to the particle filter, discussed below). More importantly, unlike their paper, we consider the case where only the first component of the state vector, and not the full state vector, is observed. We feel that full observation constitutes a too favorable situation, as we have been motivated by the application to oceanography and meteorology in which partial observation of the state vector is the norm. Note that the case where only one variable in the Lorenz is observed has been considered in Evensen and Fario (1997), but concerning the variational method with weak contraint.

The paper is organized as follows. The next section introduces the sequential data assimilation framework

Corresponding author address: Dinh Tuan Pham, Laboratoire de Modélisation et Calcul, Projet Idopt, INRIA-CNRS-INPG-UJF, B.P. 53X, 38041 Grenoble Cédex, France.
E-mail: Dinh-Tuan.Pham@imag.fr

and provides a brief review of the nonlinear optimal filter, the particle filter, and the ensemble Kalman filter (EnKF). Section 3 introduces some new filters and the use of a forgetting factor and an adaptative scale for the observation error covariance matrix. Section 4 provides some simulation results based on the Lorenz model. The paper is completed with conclusions and an appendix describing the ''second-order-exact sampling,'' which is the basis of our filters.

2. The particle filter and the EnKF

We shall adopt the same notations as in Pham et al. (1998a,b), which follows that proposed in Ide et al. (1995). Consider the physical system described by

$$\mathbf{x}^t(t_k) = \mathcal{M}(t_{k-1}, t_k)\mathbf{x}^t(t_{k-1}) + \boldsymbol{\eta}_k, \tag{2.1}$$

where $\mathbf{x}^t(t)$, a vector representing the true system state at time t, $\mathcal{M}(s, t)$, is a (nonlinear) operator expressing the system transition from time s to time t and $\boldsymbol{\eta}_k$ is the system noise vector. At each time t_k, one observes

$$\mathbf{y}_k^o = \mathcal{H}_k\mathbf{x}^t(t_k) + \boldsymbol{\epsilon}_k, \tag{2.2}$$

where $\mathcal{H}_k$ is the observational operator and $\boldsymbol{\epsilon}_k$ is the observational noise. The noises $\boldsymbol{\eta}_k$ and $\boldsymbol{\epsilon}_k$ are assumed to be independent random vectors with mean zero and covariance matrices $\mathbf{Q}_k$ and $\mathbf{R}_k$, respectively.

The sequential data assimilation (also known as filtering) consists in estimating the system state at each observation time, based only on the observations up to this time. In the linear case, this problem has been solved by the well-known Kalman filter. In the nonlinear case, one often linearizes the model around the current estimated state vector, which yields in the so-called extended Kalman filter (see, e.g., Ghil and Manalotte-Rizzoli 1991 for detail). However, this filter, besides being no longer optimal, may produce instabilities or even divergence (Kushner 1967; Gauthier et al. 1993, etc.). Theoretical solution to the optimal filtering problem in fact has been known (see, e.g., Liptser and Shiryaev 1977, Kallianpur 1980, etc.) but the practical implementation remains problematic. However some understanding on this theory would be useful to develop *practical* suboptimal filters. Note that this theory has been developed mostly for the continuous time case, which would involve partial differential equations. Below, we shall try to describe the theory based on the *discrete time* model, (2.1) and (2.2), which is simpler to understand (in our opinion) and is more relevant to our problem. Continuous time formulation, in terms of the Fokker–Planck equation and Bayes theorem, can be found in Miller et al. (1999). General formulas for calculating the optimal estimator, including the optimal nonlinear filter and smoother, can be found in van Leeuwen and Evensen (1996).

The nice feature of the Kalman filter is its closure at the second-order moments, meaning that the filter evolution is uniquely determined by its second-order characteristics. This no longer holds for the optimal nonlinear filter. Its evolution can be determined only through all its moments characteristics, or more precisely through a density function. Specifically, instead of an analysis vector $\mathbf{x}^a(t_k)$ and its error covariance matrix $\mathbf{P}^a(t_k)$ at the time t_k, one now needs an *analysis* density $d^a(\cdot\,|\mathbf{y}_1^o, \ldots, \mathbf{y}_k^o)$, which is the conditional density function of the state vector at time t_k given all past observations up to this time. Similarly, instead of a forecast state vector at time t_k and its error covariance matrix, one now needs a *forecast* density $d^f(\cdot\,|\mathbf{y}_1^o, \ldots, \mathbf{y}_{k-1}^o)$, which is the conditional density function of the state vector at time t_k given all past observations before this time. The filtering can then be performed in two steps as in the linear case.

1) The forecasting step: One computes the forecast density in term of the analysis density (at time t_{k-1}), using the model equation (2.1). Assuming *Gaussian system noise* for simplicity, the conditional density of the state vector to be at $\mathbf{z}$ at time t_k given that it is at $\mathbf{x}$ at time t_{k-1} is $\phi[\mathbf{z} - \mathcal{M}(t_k, t_{k-1})\mathbf{x}\,|\,\mathbf{Q}_k]$, where $\phi(\boldsymbol{\eta}\,|\,\boldsymbol{\Sigma}) = \exp(-\frac{1}{2}\boldsymbol{\eta}^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}\boldsymbol{\eta})/\sqrt{\det(2\pi\boldsymbol{\Sigma})}$ is the Gaussian density (at $\boldsymbol{\eta}$) of mean zero and covariance matrix $\boldsymbol{\Sigma}$ (superscript T denoting the transpose). Thus,

$$d^f(\mathbf{z}\,|\,\mathbf{y}_1^o, \ldots, \mathbf{y}_{k-1}^o) = \int \phi[\mathbf{z} - \mathcal{M}(t_k, t_{k-1})\mathbf{x}\,|\,\mathbf{Q}_k]d^a(\mathbf{x}\,|\,\mathbf{y}_1^o, \ldots, \mathbf{y}_{k-1}^o)\,d\mathbf{x}. \tag{2.3}$$

2) The analysis (or correction) step: One computes the analysis density after a new observation $\mathbf{y}_k$ has been made. By the Bayes theorom and the observation equation (2.2), assuming *again Gaussian observation noise*, it is given by

$$d^a(\mathbf{x}\,|\,\mathbf{y}_1^o, \ldots, \mathbf{y}_k^o) = \frac{d^f(\mathbf{x}\,|\,\mathbf{y}_1^o, \ldots, \mathbf{y}_{k-1}^o)\phi(\mathbf{y}_k - H_k\mathbf{x}\,|\,\mathbf{R}_k)}{\displaystyle\int d^f(\mathbf{z}\,|\,\mathbf{y}_1^o, \ldots, \mathbf{y}_{k-1}^o)\phi(\mathbf{y}_k - H_k\mathbf{z}\,|\,\mathbf{R}_k)\,d\mathbf{z}}. \tag{2.4}$$

Note that if the Gaussian assumption of the noises is dropped, one needs only to change ϕ in the above formulas to the density functions of the noises. But the main practical difficulty with these formulas is the needed integration with respect to the state vector. The integration in (2.3) requires the evaluation of $\mathcal{M}(t_k, t_{k-1})\mathbf{x}$ for a large set of values of $\mathbf{x}$, while a single such evaluation can be already quite costly in applications. Therefore the above filter is inapplicable in practice.

a. The particle filter

The basic idea is to approximate the analysis density $d^a(\cdot\,|\mathbf{y}_1^o, \ldots, \mathbf{y}_{k-1}^o)$ by a convex combination of the

Dirac function; in other words, the analysis distribution is approximated by a discrete distribution, located at $r + 1$ states $\mathbf{x}_1^a(t_{k-1}), \ldots, \mathbf{x}_{r+1}^a(t_{k-1})$ with probabilities $p_{1,k-1}, \ldots, p_{r+1,k-1}$. Then (2.3) yields the forecast density

$$d^f(\mathbf{z}\,|\,\mathbf{y}_1^o, \ldots, \mathbf{y}_{k-1}^o) = \sum_{j=1}^{r+1} \phi[\mathbf{z} - \mathcal{M}(t_k, t_{k-1})\mathbf{x}_j^a(t_{k-1})\,|\,\mathbf{Q}_k]p_{j,k-1}. \quad (2.3')$$

This is a convex combination of Gaussian densities, which reduces to a convex combination of Dirac functions if there is no system noise ($\mathbf{Q}_k = 0$). If however the system noise is present, the forecast distribution is continuous so that one again needs to approximate it by a discrete distribution. To this end, we resort to Monte Carlo drawing. We draw $r + 1$ realizations $\boldsymbol{\eta}_{1,k}, \ldots, \boldsymbol{\eta}_{r+1,k}$ according to the density $\phi(\cdot\,|\,\mathbf{Q}_k)$ and then approximate the distribution of density (2.3′) by the discrete distribution located at

$$\mathbf{x}_j^f(t_k) = \mathcal{M}(t_k, t_{k-1})\mathbf{x}_j^a(t_{k-1}) + \boldsymbol{\eta}_{j,k}, \quad j = 1, \ldots, r + 1, \quad (2.5)$$

with probabilities $p_{1,k-1}, \ldots, p_{r+1,k-1}$. The analysis distribution that results from (2.4) can then be seen to be the discrete distribution located at the same points, now denoted by $\mathbf{x}_1^a(t_k), \ldots, \mathbf{x}_{r+1}^a(t_k)$, but with probabilities

$$p_{j,k} = \frac{p_{j,k-1}\phi[\mathbf{y}_k - \mathcal{H}_k\mathbf{x}_j^a(t_k)\,|\,\mathbf{R}_k]}{\sum_{l=1}^{r+1} p_{l,k-1}\phi[\mathbf{y}_k - \mathcal{H}_k\mathbf{x}_l^a(t_k)\,|\,\mathbf{R}_k]}, \quad j = 1, \ldots, r + 1. \quad (2.6)$$

This is the basis of the particle filter. Explicitly, it operates as follows.

1) *Initialization:* One draws $r + 1$ states (henceforth called particles) $\mathbf{x}_1^a(t_0), \ldots, \mathbf{x}_{r+1}^a(t_0)$ according to an a priori distribution and sets the probabilities $p_{1,0}, \ldots, p_{r+1,0}$ to $1/(r + 1)$. In practice, we recommend constituting a large database of states obtained from a long free run of the model, then just randomly picking $r + 1$ particles out of this base.
2) *Forecast:* At time t_k, move the particles to $\mathbf{x}_j^f(t_k)$ defined by (2.5), keeping the same probabilities.
3) *Analysis:* Having observed $\mathbf{y}_k^o$, one changes the probabilities according to (2.6) and sets $\mathbf{x}_j^a(t_k) = \mathbf{x}_j^f(t_k)$. [In this simple version $\mathbf{x}_j^a(t_k)$ and $\mathbf{x}_j^f(t_k)$ are the same but they differ when resampling in introduced; see below.] The analysis state can be obtained as

$$\mathbf{x}^a(t_k) = \sum_{j=1}^{r+1} p_{j,k}\mathbf{x}_j^f(t_k), \quad (2.7)$$

and its error covariance matrix as

$$\mathbf{P}^a(t_k) = \sum_{j=1}^{r+1} p_{j,k}[\mathbf{x}_j^f(t_k) - \mathbf{x}^a(t_k)][\mathbf{x}_j^f(t_k) - \mathbf{x}^a(t_k)]^{\mathrm{T}}. \quad (2.8)$$

These computations are however optional. The filter actually operates on the set of particles and their probabilities and not on the mean and covariance statistics. The above formulas are simply meant to provide an analysis vector and its error covariance for practical use or for comparision with traditional filtering methods.

In practice, it is often necessary to introduce a resampling step for the filter to work. The reason is that, as the system is chaotic, the particles tend to become disperse and hence after some time many of them, being far from the true state, would receive negligible probability so that only a few particles effectively participate in the filter algorithm. Resampling is introduced to avoid this. Instead of setting $\mathbf{x}_j^a(t_k)$ to $\mathbf{x}_j^f(t_k)$, one draws them according to the analysis distribution and resets the probabilities to $1/(r + 1)$. There is a pitfall though. Since this distribution is discrete (more exactly one only has a discrete approximation to it), one actually draws $r + 1$ particles out of a set of $r + 1$ of elements, so one gets a lot of identical particles. If the system noise is present, this may not be very serious since they will be separated later by the addition of the $\boldsymbol{\eta}_{j,k}$ in (2.5). However, when the system noise is absent or small, identical particles will remain identical or close, so that one still effectively works with a few particles. Our remedy is to draw $\mathbf{x}_j^a(t_k), \ldots, \mathbf{x}_{r+1}^a(t_k)$ according *not to the discrete* analysis distribution but to an approximating *continuous* distribution. Borrowing from the popular kernel density estimation method (see, e.g., Silverman 1986), we propose to take this distribution as the one of density:

$$\hat{d}^a(\mathbf{x}\,|\,\mathbf{y}_1^o, \ldots, \mathbf{y}_k^o) = \sum_{j=1}^{r+1} p_{j,k}\phi[\mathbf{x} - \mathbf{x}_j^f(t_k)\,|\,h^2\mathbf{P}^a(t_k)], \quad (2.9)$$

where $\mathbf{P}^a(t_k)$ is as in (2.8) and h is a small tuning parameter. Then drawing from this density is the same drawing from the set $\{\mathbf{x}_1^f(t_k), \ldots, \mathbf{x}_{r+1}^f(t_k)\}$ with probabilities $p_{1,k}, \ldots, p_{r+1,k}$, then adding a Gaussian noise of mean zero and covariance matrix $h^2\mathbf{P}^a(t_k)$. As resampling is meant to deliberately change the analysis distribution, from the discrete distribution located at the $\mathbf{x}_j^f(t_k)$ with probability $p_{j,k}$ to the one located at $\mathbf{x}_j^a(t_k)$ with propability $1/(r + 1)$, it should be done only if the probabilities $p_{1,k}, \ldots, p_{r+1,k}$ differ too much from the uniform probabilities. As a measure of this discrepancy, we propose to take the entropy difference between the two probability distribution:

$$E(p_{1,k}, \ldots, p_{r+1,k}) = \log(r + 1) + \sum_{j=1}^{r+1} p_{j,k}\log p_{j,k}.$$

This measure is nonnegative and can be zero only if $p_{1,k} = \cdots = p_{r+1,k} = 1/(r + 1)$. Resampling is performed only when it is greater than some prescribed threshold.

b. The EnKF

The EnKF filter was introduced by Evensen (1994, 1997a), Evensen and van Leeuven (1996), and later clar-

ified by Burgers et al. (1998) (see also Evensen 1997b). It makes use of an ensemble of states as in the particle filter. In fact the forecast steps are identical in both filters. But the analysis steps differ. In the EnKF, this step is similar to the Kalman filter but applied on each individual ensemble member, using a gain matrix computed from the forecast error covariance matrix provided by the ensemble of states. Further, the observations are intentionally perturbed by an additive noise to get a correct error covariance matrix used in the analysis. More precisely, before applying the Kalman correction, the observation $\mathbf{y}_k$ has been added a noise vector $\boldsymbol{\epsilon}_{j,k}$ having zero mean and covariance matrix $\mathbf{R}_k$ (Evensen 1997b; Burger et al. 1998). This is more or less implicitly assumed in the earlier derivation of the EnKF but has been made clear in the Burgers et al. paper, which provides a detailed argument why the perturbation is needed. This paper also introduces the idea of using the mean of the ensemble states as the estimate of the system state.

It is of interest to highlight the difference between the particle filter and the EnKF, which look similar at first sight. The EnKF, as well as the filters introduced below, retain the ''linearity aspect'' of the Kalman filter in the analysis stage, in that the analysis vector and/or the ensemble members are ''corrected'' by the addition of a *linear function* of the observations. By contrast, the particle filter has nothing to do with the Kalman filter; the correction in the analysis stage is achieved by changing the probabilities associated with each particle [according to formula (2.6)], not by changing the particles. Of course, if one performs resampling, these particles will change, but this should be done only occasionally and the change simply reflects the fact that the probability has to be reset to $1/(r+1)$. In the EnKF, the probabilities of the ensemble members play no role at all, as they are uniform among ensemble members (and never change). Resampling is not necessary, in fact it is implicit at the analysis stage through the addition of noises to the observations.

3. Some new filters and improvements

The particle method requires a very large number of particles, especially in the case of a high-dimensional state space, in order to approximate adequately a continuous distribution by a discrete one. The Monte Carlo technique has a notoriously slow convergence rate: it is of the order $1/(r+1)$ where $r+1$ is the number of drawings. Since r cannot be very high, due to computational cost, large sampling fluctuation is unavoidable. Specifically, the sample mean and covariance matrix of the $\mathbf{x}_j^a(t_k)$ can be far from the theoretical mean and covariance matrix of the distribution from which they are drawn. In a high-dimensional context, these sample mean and covariance matrices contain a huge number of elements, so chances are that many of them are downright wrong.

The basic idea in our filters is to construct the ensemble members in such a way that their sample mean and covariance matrix match exactly the intended theoretical ones. Note that if the model is linear, only second-order moments are needed to construct the optimal (i.e., Kalman) filter; therefore, as we also want our filters to perform well in linear models, it is natural to require the above condition.

a. The second-order-exact EnKF

Consider the forecast stage in the EnKF. If the ensemble members $\mathbf{x}_1^a(t_{k-1}), \ldots, \mathbf{x}_{r+1}^a(t_{k-1})$ have the sample mean and covariance matrix matching exactly the theoretical ones, namely $\mathbf{x}^a(t_{k-1})$ and $\mathbf{P}^a(t_{k-1})$, then in the case of where the model (2.1) is linear, the forecast distribution has mean

$$\mathbf{x}^f(t_k) = \frac{1}{r+1}\sum_{j=1}^{r+1}\mathbf{x}_j^f(t_k-),$$

where (3.1)

$$\mathbf{x}_j^f(t_k-) = \mathcal{M}(t_k, t_{k-1})\mathbf{x}_j^a(t_{k-1})$$

and covariance matrix

$$\mathbf{P}^f(t_k) = \frac{1}{r+1}\sum_{j=1}^{r+1}[\mathbf{x}_j^f(t_k-) - \mathbf{x}^f(t_k)][\mathbf{x}_j^f(t_k-) - \mathbf{x}^f(t_k)]^{\mathrm{T}} + \mathbf{Q}_k. \quad (3.2)$$

For a nonlinear model, this formula can still be used as an approximation. But because of sampling fluctuations in Monte Carlo drawing, the matrix (3.2) will differ from sample covariance matrix of the ensemble members $\mathbf{x}_1^f(t_k), \ldots, \mathbf{x}_{r+1}^f(t_k)$ defined by (2.5). To avoid this problem, we introduce the concept of *second-order-exact sampling.* It can be easily verified that if one draws the sample $\boldsymbol{\eta}_{1,k}, \ldots, \boldsymbol{\eta}_{r+1,k}$ subjected to the conditions

$$\frac{1}{r+1}\sum_{j=1}^{r+1}\boldsymbol{\eta}_{j,k} = \mathbf{0} \quad \text{and}$$

$$\frac{1}{r+1}\sum_{j=1}^{r+1}\boldsymbol{\eta}_{j,k}\boldsymbol{\eta}_{j,k}^{\mathrm{T}} = \mathbf{Q}_k, \quad (3.3)$$

and if they further satisfy the linear contraints

$$\frac{1}{r+1}\sum_{j=1}^{r+1}\boldsymbol{\eta}_{j,k}[\mathbf{x}_j^f(t_k-) - \mathbf{x}^f(t_k)]^{\mathrm{T}} = \mathbf{0}, \quad (3.4)$$

then these ensemble members have sample means and sample covariance matrix matching exactly $\mathbf{x}^f(t_k)$ and $\mathbf{P}^f(t_k)$, as defined in (3.1) and (3.2).

Sampling subjected to the condition (3.3) will be called second-order exact as it reproduces the theoretical second-order characteristics of the sample. The constraint (3.4) on the other hand reflects the property that the ''noise'' sample $\boldsymbol{\eta}_{j,k}$ is independent of the system state. In ordinary Monte-Carlo sampling, both (3.3) and (3.4) would be approximately satisfied for large r, but

this r may need to be quite large for the approximation to be ''good enough,'' in a high-dimensional system. Our *second-order-exact sampling with linear constraints* ensures that (3.3) and (3.4) are satisfied exactly. Such a sampling procedure will be described in the appendix. We mention here only that this is possible if and only if the rank of the covariance matrix $\mathbf{Q}_k$ plus that of the constraint matrix

$$\mathbf{C}(t_k-) = [\mathbf{x}_1^f(t_k-) - \mathbf{x}^f(t_k) \cdots \mathbf{x}_{r+1}^f(t_k-) - \mathbf{x}^f(t_k)] \tag{3.4$'$}$$

does not exceed r.

Turning to the analysis stage, we have adopted the idea of correcting of the ensemble members, as in Burger et al. (1998), and not of their probabilities, as in the particle filter. Observe that the Kalman filter corrects the analysis vector as

$$\mathbf{x}^a(t_k) = \mathbf{x}^f(t_k) + \mathbf{K}_k[\mathbf{y}_k^o - \mathcal{H}_k\mathbf{x}^f(t_k)], \tag{3.5}$$

where $\mathbf{K}_k$ is the Kalman gain matrix. Since the analysis vector should be the mean of the ensemble members, one would achieve the same result by correcting directly these members, that is, by adding $\mathbf{K}_k[\mathbf{y}_k^o - \mathcal{H}_k\mathbf{x}_j^f(t_k)]$ to $\mathbf{x}_j^f(t_k)$, $j = 1, \ldots, r+1$. But then the new members would have sample covariance matrices *less* than the annalysis error covariance matrix $\mathbf{P}^a(t_k)$. Indeed,

$$\mathbf{P}^a(t_k) = (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\mathbf{P}^f(t_k)(\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)^{\mathrm{T}} + \mathbf{K}_k\mathbf{R}_k\mathbf{K}_k^{\mathrm{T}}, \tag{3.6}$$

with $\mathbf{H}_k$ denoting the gradient of $\mathcal{H}_k$ at $\mathbf{x}^t(t_k)$,[1] while the sample covariance matrix of new ensemble members equals only the first term of the above formula. To correct this, we add a noise term: the new ensemble members are taken as

$$\mathbf{x}_j^a(t_k) = \mathbf{x}_j^f(t_k) + \mathbf{K}_k[\mathbf{y}_k^o - \mathcal{H}_k\mathbf{x}_j^f(t_k)] + \tilde{\boldsymbol{\epsilon}}_{j,k}, \quad j = 1, \ldots, r+1, \tag{3.7}$$

where $\tilde{\boldsymbol{\epsilon}}_{1,k}, \ldots, \tilde{\boldsymbol{\epsilon}}_{r+1,k}$ is a second-order-exact sample from the Gaussian distribution with zero means and covariance matrix $\mathbf{K}_k\mathbf{R}_k\mathbf{K}_k^{\mathrm{T}}$, with linear constraints:

$$\sum_{j=1}^{r+1} \tilde{\boldsymbol{\epsilon}}_{j,k}[\mathbf{x}_j^f(t_k) - \mathbf{x}^f(t_k)]^{\mathrm{T}} = \mathbf{0}. \tag{3.8}$$

Then it can be checked that the sample mean and covariance matrix of $\mathbf{x}_1^a(t_k), \ldots, \mathbf{x}_{r+1}^a(t_k)$ are exactly $\mathbf{x}^a(t_k)\mathbf{P}^a(t_k)$, as given in (3.5) and (3.6).

Note that Burger et al. (1998) also add noise, but to the observation and not to the ensemble members. This is in fact equivalent since adding a noise $\boldsymbol{\epsilon}_{j,k}$ with covariance matrix $\mathbf{R}_k$ to $\mathbf{y}_j^o$ amounts to adding $\mathbf{K}_k\boldsymbol{\epsilon}_{j,k}$, which has covariance matrix $\mathbf{K}_k\mathbf{R}_k\mathbf{K}_k^{\mathrm{T}}$, to the jth member. The reason that we do this differently is that the matrix $\mathbf{K}_k\mathbf{R}_k\mathbf{K}_k^{\mathrm{T}}$ can have rank less, but never more, than that of $\mathbf{R}_k$ and since we draw our sample second order exactly and with constraints, this rank plays an important role. As it is stated earlier, the drawing is possible if and only if this rank plus the rank of the constraint matrix

$$\mathbf{C}(t_k) = [\mathbf{x}_1^f(t_k) - \mathbf{x}^f(t_k) \cdots \mathbf{x}_{r+1}^f(t_k) - \mathbf{x}^f(t_k)]^{\mathrm{T}} \tag{3.8$'$}$$

does not exceed r. Burger et al. (1998) do not use second order exact sampling, hence do not need the above consideration on ranks.

The Kalman gain matrix $\mathbf{K}_k$ is known to be $\mathbf{P}^f(t_k)\mathbf{H}_k^{\mathrm{T}}[\mathbf{K}_k\mathbf{P}^f(t_k)\,\mathbf{H}_k^{\mathrm{T}} + \mathbf{R}_k]^{-1}$, but this formula is based on the linearization of $\mathcal{H}_k$, which we feel is better to avoid. Observe that $\mathbf{H}_k\mathbf{P}^f(t_k)\mathbf{H}_k^{\mathrm{T}}$ and $\mathbf{P}^f(t_k)\mathbf{H}_k^{\mathrm{T}}$ represent (in the case where $\mathcal{H}_k$ is linear) the sample covariance matrix of $\mathbf{y}_j^f(t_k) = \mathcal{H}_k\mathbf{x}_j^f(t_k)$, $j = 1, \ldots, r+1$ and cross-covariance matrix between $\mathbf{x}_1^f(t_k), \ldots, \mathbf{x}_{r+1}^f(t_k)$ and $\mathbf{y}_1^f(t_k), \ldots, \mathbf{y}_{r+1}^f(t_k)$. This suggests taking as gain matrix

$$\mathbf{K}_k = \left\{\frac{1}{r+1}\sum_{j=1}^{r+1}[\mathbf{x}_j^f(t_k) - \mathbf{x}_j^f(t_k)][\mathbf{y}_j^f(t_k) - \mathbf{y}^f(t_k)]^{\mathrm{T}}\right\} \times \left\{\frac{1}{r+1}\sum_{j=1}^{r+1}[\mathbf{y}_j^f(t_k) - \mathbf{y}_j^f(t_k)][\mathbf{y}_j^f(t_k) - \mathbf{y}^f(t_k)]^{\mathrm{T}} + \mathbf{R}_k\right\}^{-1}. \tag{3.9}$$

b. The singular evolutive interpolated Kalman filter

We have developed the singular evolutive interpolated Kalman (SEIK) filter (Pham, 1997; Pham et al. 1998a) as a variant of the singular evolutive extended Kalman (SEEK) filter (Pham et al. 1998b). But it may also be viewed as a variant of the EnKF using a *minimum number* of ensemble members.

The principle of the SEEK filter is to make corrections only in the directions for which the error is amplified or not sufficiently attenuated by the system dynamic, relying on such attenuation to keep the error small in other directions. This assumes that the system admits an attractor and the ''direction of corrections'' are then those parallel to the tangent space to the attractor at the analysis state. Such selective correction is achieved by using an error covariance matrix of low rank r (but not lower than the dimension of the attractor). In the case where there is no system noise, one simply initializes the extended Kalman filter with such an error covariance matrix, then it can be shown that this matrix remains of the same rank at all later times. This is no longer true if the system noise $\boldsymbol{\eta}_k$ is present. To overcome this difficulty, we basically project this noise onto the range space of the error covariance matrix and ignore the noise component in the orthogonal complement of this space. The justification comes from the fact that the second noise component, being ''transversal'' to the attractor, would be strongly attenuated by the system dynamic (for a more detailed discussion, see Pham et al. 1998b).

[1] This formula is actually only approximate, except when $\mathcal{H}_k$ is linear.

Consider the second-order-exact EnKF; the above arguments suggest that one can take r as low as the dimension of the attractor. The tangent space to the attractor is then actually approximated by the affine space supported by the ensemble members $\mathbf{x}_1^a(t_k), \ldots, \mathbf{x}_{r+1}^a(t_k)$. However, the filter requires that r is not less than the rank of $\mathbf{Q}_k$ plus that of the matrix (3.4′) and than the rank of $\mathbf{K}_k\mathbf{R}_k\mathbf{K}_k^T$ plus that of the matrix (3.8′). To guarantee that such a condition is satisfied would require that r is not less than twice the dimension of the state vector. This would exclude meteorology and oceanography applications for which this dimension could be in the range 10^5–10^6.

To overcome the above difficulty, we shall make use of resampling. Instead of ''correcting'' the ensemble members as in (3.7), we simply redraw them second order exactly according to the Gaussian distribution with mean $\mathbf{x}^a(t_k)$ and covariance matrix $\mathbf{P}^a(t_k)$. As will be seen, this matrix has the same rank as $\mathbf{P}^f(t_k)$ so that one only needs that $\mathbf{P}^f(t_k)$ has rank r (or less). To ensure that $\mathbf{P}^f(t_k)$ has rank r, we resort to the method used in our SEEK filter: we project the system noise $\boldsymbol{\eta}_k$ onto the linear space parallel to the affine space supported by $\mathbf{x}_1^f(t_k-), \ldots, \mathbf{x}_{r+1}^f(t_k-)$ and pretend that this is the system noise. Note that since we redraw the ensemble members, we do not need the $\mathbf{x}_j^f(t_k)$ except for the computation of $\mathbf{K}_k$ in (3.9). This is not a problem as an alternative definition of $\mathbf{K}_k$ is possible and will be given shortly. Thus we do not have to draw $\boldsymbol{\eta}_{1,k}, \ldots, \boldsymbol{\eta}_{r+1,k}$.

In summary, the forecast step of this filter consists in the calculation of $\mathbf{x}_1^f(t_k-), \ldots, \mathbf{x}_{r+1}^f(t_k-)$, $\mathbf{x}^f(t_k)$, and $\mathbf{P}^f(t_k)$ by (3.1) and (3.2), while the analysis step consists in computing $\mathbf{x}^a(t_k)$, $\mathbf{P}^a(t_k)$, and redrawing $\mathbf{x}_1^a(t_k), \ldots, \mathbf{x}_{r+1}^a(t_k)$.

As the covariance matrices involved have rank r, it is much more efficient computationally to work with their factored form. To this end, we introduce a $(r+1)\times r$ full rank matrix $\mathbf{T}$ such that its columns are orthogonal to the vector $[1\ \cdots\ 1]^T$. A convenient choice is

$$\mathbf{T} = \begin{bmatrix} 1 & & 0 \\ & \cdots & \\ 0 & & 1 \\ 0 & \cdots & 0 \end{bmatrix} - \frac{1}{r+1}\begin{bmatrix} 1 & \cdots & 1 \\ \vdots & \vdots & \vdots \\ 1 & \cdots & 1 \end{bmatrix}.$$

Then the matrix $\mathbf{P}^f(t_k)$ defined by (3.2) can be written as $\mathbf{L}_k\mathbf{U}_{k-1}\mathbf{L}_k^T + \mathbf{Q}_k$, where

$$\mathbf{L}_k = [\mathbf{x}_1^f(t_k-)\cdots\mathbf{x}_{r+1}^f(t_k-)]\mathbf{T},$$

$$\mathbf{U}_{k-1} = [(r+1)\mathbf{T}^T\mathbf{T}]^{-1}.$$

[Note that for the above choice of $\mathbf{T}$, $\mathbf{L}_k$ is none other than the matrix with columns $\mathbf{x}_1^f(t_k-) - \mathbf{x}^f(t_k), \ldots, \mathbf{x}_r^f(t_k-) - \mathbf{x}^f(t_k)$ and $(r+1)\mathbf{T}^T\mathbf{T}$ is the Toeplitz matrix with r on the main diagonal and -1 on the other diagonals.] But, as stated before, we change the system noise by projecting it onto the linear space spanned by the columns of $\mathbf{L}_k$. This yields that $\mathbf{P}^f(t_k) = \mathbf{L}_k\tilde{\mathbf{U}}_{k-1}\mathbf{L}_k^T$, where

$$\tilde{\mathbf{U}}_{k-1} = \mathbf{U}_{k-1} + (\mathbf{L}_k^T\mathbf{L}_k)^{-1}\mathbf{L}_k^T\mathbf{Q}_k\mathbf{L}_k(\mathbf{L}_k^T\mathbf{L}_k)^{-1}.$$

Since $\mathbf{K}_k = \mathbf{P}^f(t_k)\mathbf{H}_k^T[\mathbf{H}_k^T\mathbf{P}^f(t_k)\mathbf{H}_k + \mathbf{R}_k]^{-1}$, one gets

$$\mathbf{K}_k = \mathbf{L}_k\tilde{\mathbf{U}}_{k-1}(\mathbf{HL})_k^T[(\mathbf{HL})_k^T\tilde{\mathbf{U}}_{k-1}(\mathbf{HL})_k + \mathbf{R}_k]^{-1}, \quad (3.10)$$

where $(\mathbf{HL})_k = \mathbf{H}_k\mathbf{L}_k$. However, in the case where $\mathcal{H}_k$ in nonlinear, we feel that it is better to avoid linearization and take $(\mathbf{HL})_k = [\mathcal{H}_k\mathbf{x}_1^f(t_k-)\cdots\mathcal{H}_k\mathbf{x}_{r+1}^f(t_k-)]\mathbf{T}$.

The analysis state $\mathbf{x}^a(t_k)$ is given by (3.5) and its error covariance matrix can be obtained by a similar (and tedious) calculation as in the SEEK filter: $\mathbf{P}^a(t_k) = \mathbf{L}_k\mathbf{U}_k\mathbf{L}_k^T$, where

$$\mathbf{U}_k = [\tilde{\mathbf{U}}_{k-1}^{-1} + (\mathbf{HL})_k^T\mathbf{R}_k^{-1}(\mathbf{HL})_k]^{-1}. \quad (3.11)$$

Also $\mathbf{K}_k$ can be more conveniently computed as $\mathbf{L}_k\mathbf{U}_k(\mathbf{HL})_k^T\mathbf{R}_k^{-1}$.

It is worthwhile to relate the SEIK filter to the SEEK filter. Formulas (3.1) and (3.2), together with the fact that the sample covariance matrix of $\mathbf{x}_1^a(t_{k-1}), \ldots, \mathbf{x}_{r+1}^a(t_{k-1})$ equals $\mathbf{P}^a(t_{k-1})$, are almost the same as those in the forecast step of the extended Kalman filter. The main difference is that no gradient calculation is done and some form of differencing is performed instead. While the calculation in the extended Kalman filter is based on linearizing the model operator around $\mathbf{x}^a(t_{k-1})$, it is here based on interpolating it at the ensemble members $\mathbf{x}_1^a(t_{k-1}), \ldots, \mathbf{x}_{r+1}^a(t_{k-1})$. Such interpolation would yield precisely the forecast and its error covariance matrix, as given by (3.1) and (3.2). That is why we call our filter interpolated. The analysis step is the same as in the SEEK filter, except that the linearization of $\mathcal{H}_k$ is also avoided.

A problem that may arise in the SEIK filter is that the randomly drawn ensemble member is not a physically realizable state. This is however a common problem in Kalman-type filters, since at the analysis stage a correction is performed that could conceivably pull the analysis state (or an ensemble member) to an unrealizable state. But this problem might occur more frequently in the SEIK filter, due to the random nature of resampling. If this happens, a possible remedy is to replace the offending state with a nearby realizable state. Another weakness of the SEIK filter is its use of the projection of the system noise vector onto the linear space spanned by correction basis. Although it is quite reasonable to assume that the noise vector lies in a reduced space, it is debatable that this space would be parallel to the tangent space of the attractor. Such weakness is dealt with in the next filter, in which the system noise is assumed only to have a low rank covariance matrix.

c. The singular second-order-exact EnKF

As has been mentioned, a constraint of the second-order-exact EnKF is that the number of ensemble mem-

bers should be greater than twice the dimension of the phase space, in order that the ''rank conditions'' can be guaranteed to hold. Often, this dimension is very high. However, as the system state essentially lies on the attractor, which has a low dimension, the rank conditions can still be approximately satisfied with a small number of ensemble members.

The basic idea is that the constraint matrices $\mathbf{C}(t_k-)$ and $\mathbf{C}(t_k)$ could be well approximated by matrices of some low rank r^*, say. We will construct such an approximation through the singular value decomposition. One can write $\mathbf{C}(t_k-) = \mathbf{VDW}^{\mathrm{T}}$ where $\mathbf{V}$ and $\mathbf{W}$ are matrices with columns of unit norm and orthogonal each to the other and $\mathbf{D}$ is a diagonal matrix (containing the singular values). A rank r^* approximation $\tilde{\mathbf{C}}(t_k-)$ to $\mathbf{C}(t_k-)$ is then obtained by putting to 0 the diagonal elements of $\mathbf{D}$ other than the r^* largest. If the covariance matrix $\mathbf{Q}_k$ of the system noise is of rank $r' \leq r - r^*$, then it is possible to draw $\boldsymbol{\eta}_{1,k}, \ldots, \boldsymbol{\eta}_{r+1,k}$ second order exactly, with linear constraints based on $\tilde{\mathbf{C}}(t_k-)$ instead of $\mathbf{C}(t_k-)$. If not, one could still approximate it by a matrix of rank $r' \leq r - r^*$. Similarly, one replaces the constraint matrix $\mathbf{C}(t_k)$ by a rank r^* approximation $\tilde{\mathbf{C}}(t_k)$. If the matrix $\mathbf{K}_k\mathbf{R}_k\mathbf{K}_k^{\mathrm{T}}$ has rank $r'' \leq r - r^*$, one can again draw $\tilde{\boldsymbol{\epsilon}}_{1,k}, \ldots, \tilde{\boldsymbol{\epsilon}}_{r+1,k}$ second order exactly with the linear constraints based on $\tilde{\mathbf{C}}(t_k)$. If not, one could approximate $\mathbf{K}_k\mathbf{R}_k\mathbf{K}_k^{\mathrm{T}}$ by a matrix of rank $r'' \leq r - r^*$. Approximations to a covariance matrix may be obtained through an eigenvector decomposition. For example, let $\mathbf{Q}_k = \mathbf{VDV}^{\mathrm{T}}$ where $\mathbf{V}$ is an orthogonal matrix, the columns of which are the eigenvectors of $\mathbf{Q}_k$ with eigenvalues the corresponding diagonal elements of the diagonal matrix $\mathbf{D}$. Then putting these diagonal elements, other than the r' largest, to 0 yields a rank r' approximation to $\mathbf{Q}_k$.

The above method requires the choice of three approximation ranks r^*, r', and r'', and a number of ensemble members $r + 1 > r^* + \max(r', r'')$. Note that if $\mathbf{C}(t_k)$ is nearly of rank r^*, then $\mathbf{K}_k$ would also be nearly of rank r^* or less, as can be seen from formula (3.9). Hence r'' can be taken equal to r^* or less.

Since the gain matrix is of rank at most r, it is more efficiently computed as in section 3b. Let $\mathbf{T}$ be the same matrix as defined there and put

$$\mathbf{L}_k = [\mathbf{x}_1^f(t_k) \cdots \mathbf{x}_{r+1}^f(t_k)]\mathbf{T}, \qquad \mathbf{U}_{k-1} = [(r+1)\mathbf{T}^{\mathrm{T}}\mathbf{T}]^{-1}.$$

Then $\mathbf{K}_k$ may be computed as in (3.9) but with $\mathbf{U}_{k-1}$ replaced by $\mathbf{U}_{k-1}$ and $\mathbf{x}_j^f(t_k-)$ [in the definition of $(\mathbf{HL})_k$] replaced by $\mathbf{x}_j^f(t_k)$.

d. The use of a forgetting factor and an adaptative scale in the observation error covariance matrix

In our earlier studies (Pham et al. 1998a,b), we have found that the introduction of a forgetting factor can greatly improve the filter stability. This can be explained by the fact that our models can be strongly nonlinear in certain region of the phase space and thus quite sensitive to the initial condition, hence relying too much on the model forecast could lead to divergence (with respect to the true state). It is prudent to rely more on the observation in constructing the analysis vector, than as one would do in the linear case. A simple way to achieve this is to use a gain matrix computed from a *smaller* observation noise covariance matrix. More precisely, we replace, in the formulas (3.9) and (3.10), $\mathbf{R}_k$ by $\rho\mathbf{R}_k$ where ρ is a positive coefficient less than 1. Note that this would yield the same gain matrix $\mathbf{K}_k$ as replacing $\tilde{\mathbf{U}}_{k-1}$ by $\tilde{\mathbf{U}}_{k-1}/\rho$. This new gain matrix can still be computable as $\mathbf{L}_k\mathbf{U}_k(\mathbf{HL})_k\mathbf{R}_k^{-1}$ but with $\mathbf{U}_k$ now given by $[\rho\tilde{\mathbf{U}}_{k-1}^{-1} + (\mathbf{HL})_k\mathbf{R}_k^{-1}(\mathbf{HL})_k^{\mathrm{T}}]^{-1}$ and not (3.11). Note that we propose to change only the gain matrix and not the forecast error covariance matrix $\mathbf{P}^f(t_k)$, as we have done in earlier works (Pham et al. 1998a,b). Thus the analysis error covariance matrix will still be computed through formula (3.6) and this yields, after some tedious algebra,

$$\mathbf{P}^a(t_k) = \mathbf{L}_k\mathbf{U}_k[\rho^2\tilde{\mathbf{U}}_{k-1}^{-1} + (\mathbf{HL})_k^{\mathrm{T}}\mathbf{R}_k^{-1}(\mathbf{HL})_k]\mathbf{U}_k\mathbf{L}_k^{\mathrm{T}},$$

where $\mathbf{U}_k$ is as above. We feel that the present approach is more satisfactory theoretically than the earlier one, since we change only the gain matrix and nothing else (but in our simulations their performance are quite similar). Incidentally, in the case of the (singular) second-order EnKF, the $\boldsymbol{\epsilon}_{j,k}$ are still drawn according to the covariance matrix $\mathbf{R}_k$. We argue that in a nonlinear context there is no compelling reason that one should stick with the Kalman gain: the linear correction is not optimal anyway and this gain is obtained from a linearization that inevitably contains some error. By using a modified gain as above, we may let some more observation error enter the analysis vector but we reduce the risk of filter divergence, because we pull this vector more strongly toward the observation. The factor ρ is called the forgetting factor, and its use can be viewed as an error compensation technique that is fairly common (Jawinsky 1970).

Another modification of the considered filters is the use of an adaptive scale in the observation error covariance matrix. This is necessary to deal with a difficulty is the case of small noise. To simplify, consider the case $\mathbf{Q}_k = \mathbf{0}$ and $\mathbf{R}_k = \sigma^2\tilde{\mathbf{R}}_k$ with a very small scale factor σ^2. Then in the particle filter, all particles, except the one for which the observation operator yields the closest value to the observation, would receive negligible probabilities [see formula (2.6)]. Therefore the analysis error covariance matrix (2.8) would be almost zero. The same thing happens in the SEIK filter: for very small σ^2, the matrix $\mathbf{U}_k$ in (3.10) would reduce to $\sigma^2[(\mathbf{HL})_k^{\mathrm{T}}\tilde{\mathbf{R}}_k^{-1}(\mathbf{HL})_k]^{-1}$, provided that the later is invertible. In any case, one can see that after several analysis steps, this matrix will become of the same order as σ^2 and hence so is $\mathbf{P}^a(t_k)$. The behavior of the EnKF (second-order exact or not) is more subtle. But is can be seen from the formula (3.7) without the term $\tilde{\boldsymbol{\epsilon}}_{j,k}$ that the set of ensemble states would shrink in size toward a single one. Thus in all cases we will have an almost

zero analysis error covariance matrix, but the analysis vector, in general, cannot not be so accurate.

The above phenomenon can be explained. The analysis error covariance matrix $\mathbf{P}^a$, by the way it is computed, reflects only the error coming from the system and observation noises; other kinds of errors have been ignored. This includes the discrete approximation and the addition of noise in the resampling scheme in the particle filter, and the sampling fluctuation in Monte Carlo sampling and the linearization or interpolation error in other filters. When the system and observation noises are very small, the above kinds of errors are no longer negligible before them and hence the analysis error covariance matrix, as computed by the Kalman filter and its variants, can seriously underestimate the true error.

The Kalman gain however is not really affected by the small noise problem. As mentioned above, the matrices $\mathbf{U}_k$ and $\mathbf{P}^a(t_k)$ will become in the long run of the same order as σ^2. Thus, looking at formula (3.9) for the gain matrix, one sees that σ^2 cancel out. Therefore the extended Kalman filter and the SEEK filter are not affected by the small noise problem, *except that the filter error covariance matrix they produce could be too low*. However, the filters considered in this paper are based on a set of particles that should have this matrix as a covariance matrix; hence, they would cluster too closely around their means.

To address the above problem we propose not to use a fixed scale factor σ^2 but we have adapted it during the filter algorithm to reflect the true filter errors. Specifically, we estimate it as

$$\hat{\sigma}_k^2 = \hat{\sigma}_{k-1}^2 + (1 - \lambda)(\mathbf{e}_k^{\mathrm{T}}\tilde{\mathbf{R}}_k^{-1}\mathbf{e}_k - \sigma_{k-1}^2),$$

where $\mathbf{e}_k = \mathbf{y}_k^o - \mathcal{H}_k\mathbf{x}^f(t_k)$ is the innovation vector and λ is an adaptation factor. Actually $\hat{\sigma}_k^2$ would somewhat overestimate the scale factor σ^2, since the innovation vector not only contains the observation noise but also some part of the filter forecast error as well. However, our procedure has the advantage of still providing a realistic analysis error covariance matrix $\mathbf{P}^a(t_k)$, hence a realistic set of particles, for very small noise. Further, it is robust against erroneous specification of the noise level.

4. Simulation results

The simulation is based on the Lorenz model defined by the system of differential equations:

$$\frac{dx}{dt} = s(y - x), \qquad \frac{dy}{dt} = (r - z)x - y,$$

$$\frac{dz}{dt} = xy - bz,$$

with coefficients classically chosen as $s = 10$, $r = 28$, $b = 8/3$. This system of equations is integrated by the well-known fourth-order Runge–Kutta method, with an integration step of 0.005. Observations are made at intervals of 0.05, on the x coordinate only and with the observation error having a variance of 2. This design is similar to that of Miller et al. (1994), except that observations are made more frequently, to compensate for the fact that only the x coordinate and not the full state vector is observed. The filter performance will be measured by the root-mean-square error (rmse), which is $1/\sqrt{3}$ times the norm of the difference between the analysis and the true states.

For the particle filter, the second-order-exact EnKF and the EnKF, we construct a database of states as follows. We generate a trajectory of the Lorenz model starting at $(x, y, z) = (-0.587\,276, -0.563\,678, 16.8708)$ containing 500 states (excluding the initial one) at intervals of 0.05, but retain only the last 400 states to avoid the transitory phase. The initial particles or ensemble members in the above filters are simply randomly drawn from this database. The same database is also used to perform an empirical orthogonal functions (EOF) analysis, to construct the initial analysis error covariance matrix in the SEIK filter (the initial state is the mean of the states in the database). For more detail on this EOF method, see, for example, Pham et al. (1998b). Only the first two EOFs are retained, yielding a rank-two matrix. This is in line with the fact that the Lorenz attractor has a dimension of about 2.06.

Figure 1 plots the rmse of the particle filter with 10, 50, and 200 particles. The resampling threshold and the coefficient h, which controls the amount of perturbation (see the end of section 2a), have been chosen to be 0.15 and 0.7, 0.3 and 0.5, and 0.5 and 0.45, respectively. These choices have been obtained through trials to obtain a good performance of the filter. We must mention that this performance is quite sensible to the choice of these tuning parameters. A bad choice could result in a much worse performance than is reported here. Likewise better performance could be achieved with some other choices, since we have not tried very hard to find the best ones. Observe that the higher the number of particles, the higher the threshold and the lower the coefficient h should be, since resampling would be needed less often and with less perturbation noise added, as the set of particles become denser. Clearly, the filter performance improves as the number of particles increases. It is also less dependent on the choice of the above tuning parameters as well. For the present problem, 10 particles seem to be the lower bound and 50 particles might be needed to obtain a reasonably good performance without a fine-tuning of the resampling threshold and coefficient.

Figure 2 plots the rmse of the EnKF with no forgetting factor and with a forgetting factor of 0.8. One can see that the use of a forgetting factor improves markedly the filter performance. As can be seen in this plot, and also in previous and subsequent plot, the filter sometime seems to lose track of the true system state, causing a sudden burst of large error (we suspect this happens

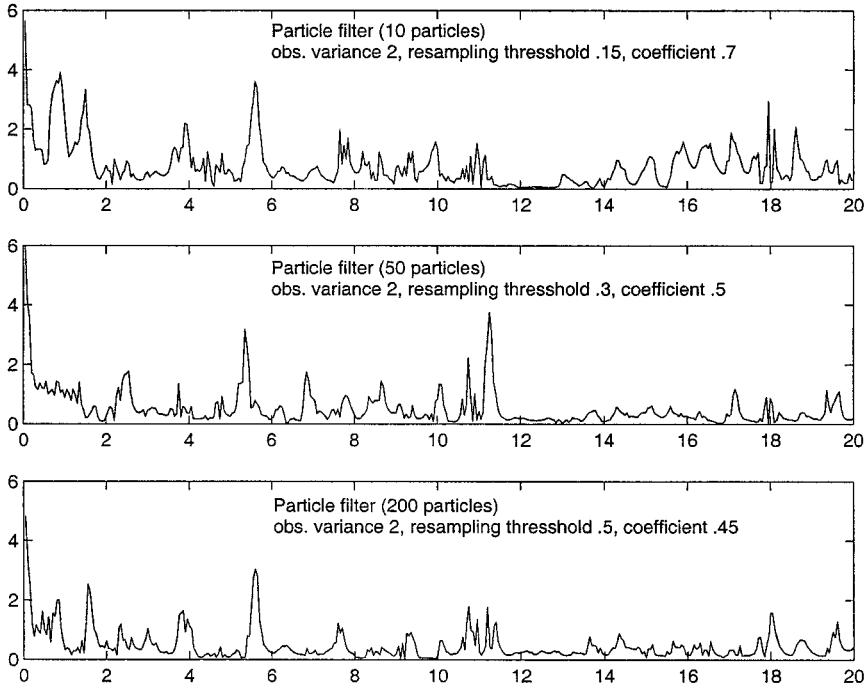

FIG. 1. Rmse of the particle filter with 10, 50, and 200 particles.

when the system state enters a strong nonlinearity region). The use of a forgetting factor helps to reduce the peak of these bursts and may even prevent them from happening. In this sense, it enhances the filter stability.

Figure 3 plots the rmse of the EnKF with 5 and 50 ensemble members, to illustrate the effect of the size of the ensemble. It can be seen that the filter does not perform well with five ensemble members. Using a forgetting factor does not help. In fact, the filter performs somewhat worse with a forgetting factor of 0.9 or 0.8,

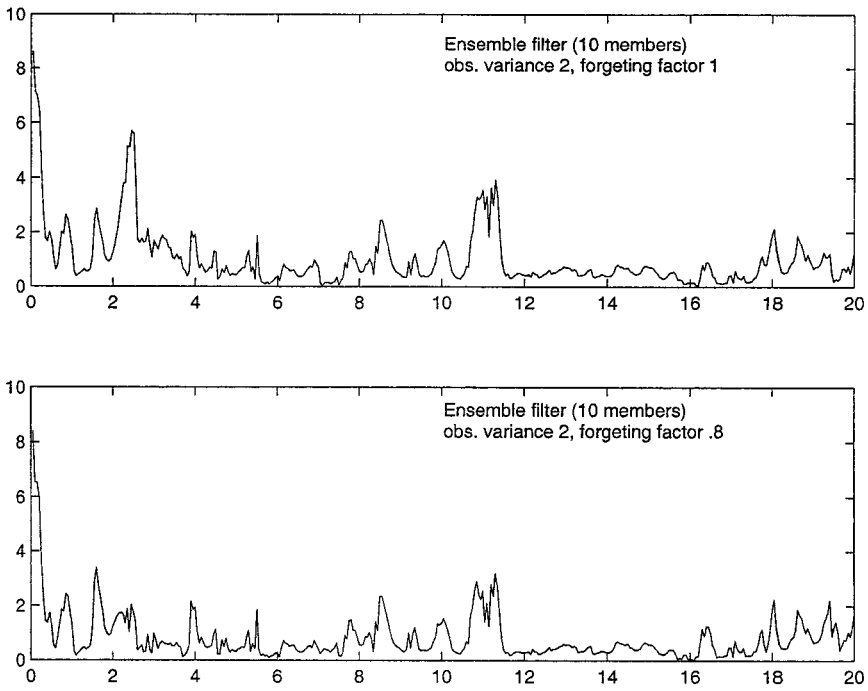

FIG. 2. Rmse of the EnKF with and without the forgetting factor.

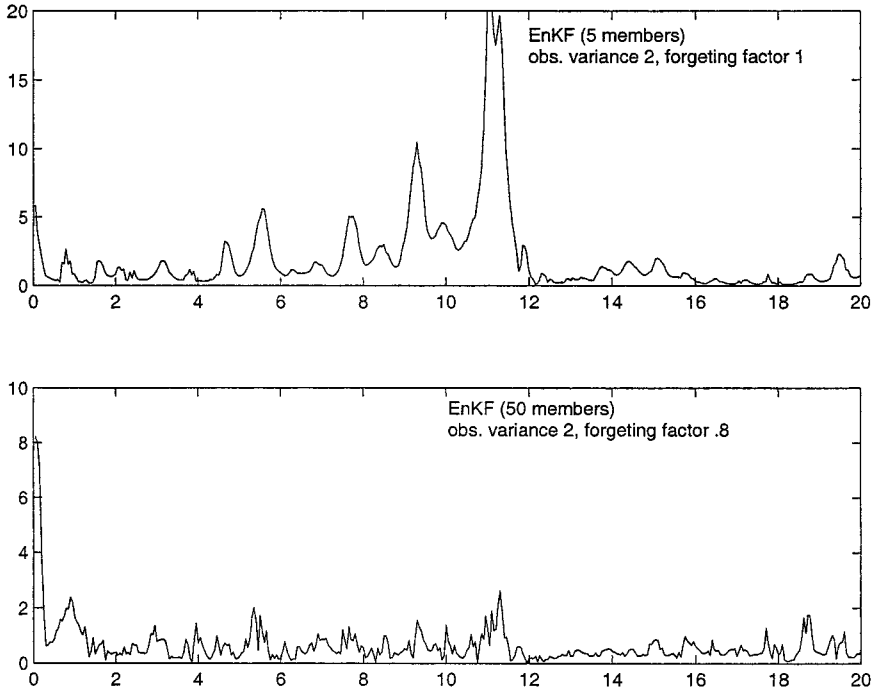

FIG. 3. Rmse of the EnKF with 5 and 50 ensemble members.

which is why we chose to show the result with no forgetting factor. Not that the EnKF has also been studied by Evensen (1997a) in the context of the Lorenz model, and he has noted that good results can be obtained using as few as about 10 ensemble members. However, simulation results in Evensen (1997a) cannot really be compared with ours because he uses full observations (but with a larger time interval between them) and not partial observations.

Figure 4 plots the rmse of the second-order-exact

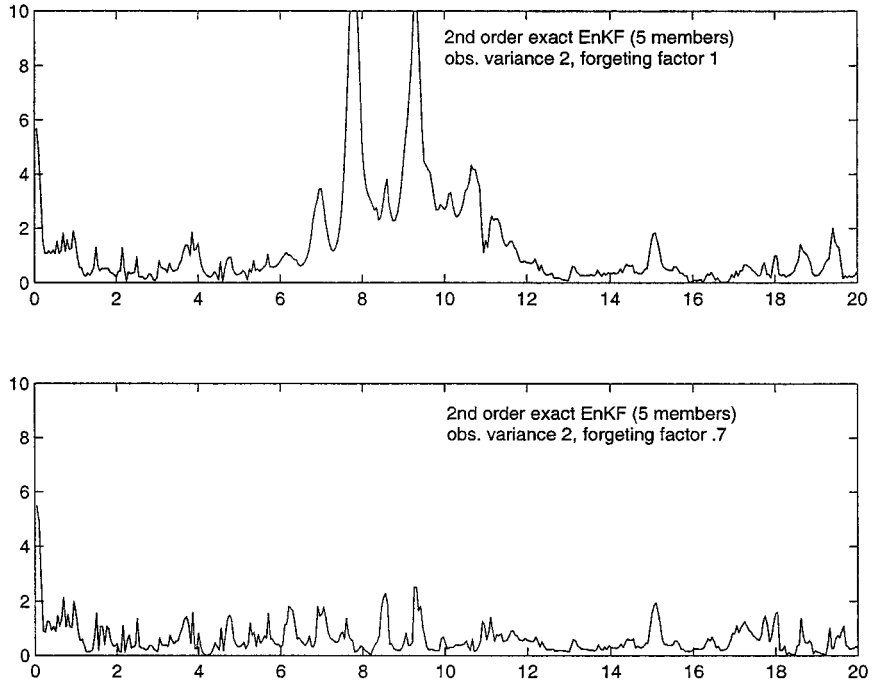

FIG. 4. Rmse of the second-order-exact EnKF with and without the forgetting factor.

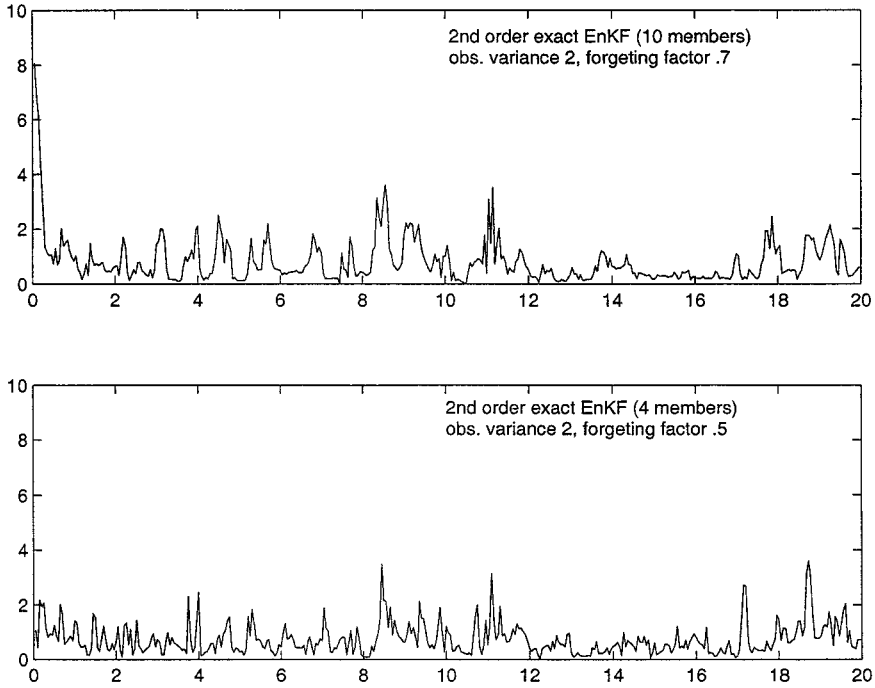

FIG. 5. Rmse of the second-order-exact EnKF with and without the forgetting factor.

EnKF with and without the forgetting factor. Here again the effect of this factor is drastic, as it prevents a too large filter error from happening around the middle of the experiment. Figure 5 plots the rmse of this filter with different numbers of ensemble members, 4 and 10, respectively, to show the effect of this number. Curiously, increasing the number of members from 5 to 10 does not seem to increase filter performance. Its performance at 10 members seems even slightly worse. With more members, this filter behaves quite similarly to the EnKF, as expected, since the drawing in the EnKF would satisfy approximately the second-order-exact condition and the constraints implemented by this filter. But its performance (and also that of the EnKF) at 50 members, for example, is still not much different than at only 5 ensemble members.

We do not have a sure explanation of the above finding. A plausible one is that the second-order-exact EnKF at five members has almost attained its maximum capability. Indeed, this filter and also the EnKF correct the analysis state in a *linear* way based on a gain matrix computed from the second-order statistics only. The use of more ensemble members may improve the accuracy in evaluating these statistics, but the filters still have not exploited any other nonlinear characteristics (higher-order moments, shape of the analysis density, . . .) of the system. By contrast the particle filter can theoretically attain the optimal performance when the number of particles goes to infinity. Note that the rank condition in the second-order-exact EnKF requires that $r \geq 4$ so that one needs at least five ensemble members. This is because the rank of the gain matrix is 1 (as the observation is a scalar) and the constraint matrix is of rank 3 usually (as the state vector is of this dimension). The above simulation results suggest that one does not need a number of ensemble members more than the strict minimum. Note that the second plot of Fig. 5 (with four ensemble members) corresponds actually to the singular second-order-exact EnKF, in which the constraint matrix has been reduced to a rank-two matrix (in line with the fact that the Lorenz attractor has a dimension of about 2). The performance of this filter is somewhat worse than the one with five ensemble members, but not much.

Finally, Fig. 6 shows the behavior of the SEIK filter and again illustrates the effect of the forgetting factor. As one can see, this effect is quite drastic near the end of the experiment. The performance of this filter compares very favorably to the second-order-exact EnKF. It is almost as good as the later with five ensemble members and is even better than this filter with four ensemble members. Yet, it requires only three members.

5. Conclusions and discussion

We have introduced some new filters of stochastic nature for data assimilation in nonlinear systems. They have in common with previously introduced filters, such as the particle filter and the EnKF, the use of Monte Carlo drawings to mimic the behavior of the system. Our contributions consist of three main points:

1) The concept of second-order-exact sampling with

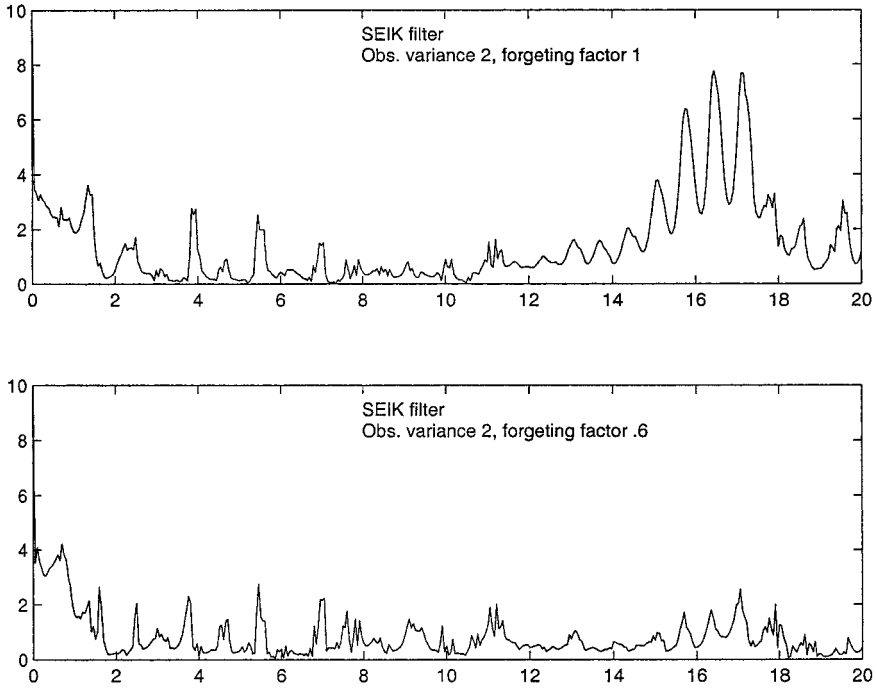

FIG. 6. Rmse of the SEIK filter with and without the forgetting factor.

linear constraints. This permits the reduction of the number of ensemble members to a strict minimum and yet does not cause any degradation of performance.

2) The exploitation of the (existence of a) low-dimensional attractor of the system. This permits the further reduction of the number of ensemble members, especially in the case of a system with high-dimensional phase space. The filter may be viewed roughly as working on the attractor and not on the phase space and the number of ensemble members may be reduced to just the dimension of the attractor plus 1. Again, this does not degrade performance.
3) The use of the forgetting factor to enhance filter stability. Our simulation studies show that filter instability is the most vexing problem in nonlinear data assimilation. The filter can work well for a while and suddenly commit large errors and then can lose track of the true system state almost completely. The use of the forgetting factor can be helpful to alleviate this problem. Our simulations shows that it is rather effective to prevent such ''filter excursions'' or at least reduce their magnitude and duration.

In terms of performance, our new filters are comparable to the EnKF, all with an appropriate forgetting factor. The advantage of the new filters is in their low computational cost. In this respect the SEIK filter is the best. The particle filter, however, can outperform these filters (as well as the EnKF), but with the cost of a very large number of particles. In the case of the Lorenz model considered here, the break point seems to be around 200.

Acknowledgments. The author would like to thank O. Talagrand and J. Verron for bringing their attention to this problem and with whom he has had many stimulating and fruitful discussions, and also to two referees who have carefully read the paper and suggested many improvements.

APPENDIX

Second-Order-Exact Sampling with Linear Constraints

The goal is to draw a sample of size $r + 1$ from a Gaussian distribution with mean $\mathbf{m}$ and covariance matrix $\mathbf{\Sigma}$, *conditionally* on the constraints that the sample mean and covariance matrix equal exactly $\mathbf{m}$ and $\mathbf{\Sigma}$ and on some other linear constraints. More precisely, the obtained sample $\mathbf{x}_1, \ldots, \mathbf{x}_{r+1}$ must satisfy

$$\frac{1}{r+1}\sum_{j=1}^{r+1}\mathbf{x}_j = \mathbf{m},$$

$$\frac{1}{r+1}\sum_{j=1}^{r+1}(\mathbf{x}_j - \mathbf{m})(\mathbf{x}_j - \mathbf{m})^{\mathrm{T}} = \mathbf{\Sigma} \qquad \text{(A.1)}$$

$$[(\mathbf{x}_1 - \mathbf{m}) \cdots (\mathbf{x}_{r+1} - \mathbf{m})]\mathbf{C} = \mathbf{0}, \qquad \text{(A.2)}$$

where $\mathbf{C}$ is a given matrix of $r + 1$ rows and has rank $r'' \leq r$.

Without loss of generality, we may assume that $\mathbf{m} = \mathbf{0}$, since by adding a constant to the sample we can adjust it to have any mean. Further, the first constraint of (A.1) and the constraint (A.2) can be grouped into

$$[\mathbf{x}_1 \cdots \mathbf{x}_{r+1}]\mathbf{\Gamma} = \mathbf{0}, \qquad \mathbf{\Gamma} = \begin{bmatrix} 1 & \\ \vdots & \mathbf{C} \\ 1 & \end{bmatrix}. \tag{A.3}$$

Let r' be the rank of $\mathbf{\Sigma}$, then one may decompose $\mathbf{\Sigma}$ into $\mathbf{L}\mathbf{L}^{\mathrm{T}}$ where $\mathbf{L}$ is a full rank matrix of r' columns. By (A.1), $\mathbf{x}_1, \ldots, \mathbf{x}_{r+1}$ must lie on the linear space spanned by the column of $\mathbf{L}$. Thus they may be written in the form $\mathbf{L}\mathbf{y}_1, \ldots, \mathbf{L}\mathbf{y}_{r+1}$ where $\mathbf{y}_1, \ldots, \mathbf{y}_{r+1}$ is a sample from the conditional Gaussian distribution of zero mean and unit covariance matrix, subject to the constraints that

$$[\mathbf{y}_1 \cdots \mathbf{y}_{r+1}]\mathbf{\Gamma} = \mathbf{0}, \qquad \frac{1}{r+1}\sum_{j=1}^{r+1} \mathbf{y}_j\mathbf{y}_j^{\mathrm{T}} = \mathbf{I}. \tag{A.4}$$

The first constraint in (A.4) means that the row vectors of $[\mathbf{y}_1 \cdots \mathbf{y}_{r+1}]$ belong to the linear subspace of $\mathbb{R}^{r+1}$ orthogonal to the columns of $\mathbf{\Gamma}$. Thus, letting $\{\mathbf{v}_1, \ldots, \mathbf{v}_{r-r''}\}$ be an orthonormal basis of this space, this constraints implies that

$$[\mathbf{y}_1 \cdots \mathbf{y}_{r+1}] = [\mathbf{y}_1 \cdots \mathbf{y}_{r+1}][\mathbf{v}_1 \cdots \mathbf{v}_{r-r''}][\mathbf{v}_1 \cdots \mathbf{v}_{r-r''}]^{\mathrm{T}}.$$

Without the constraint (A.4), the elements of the matrix $[\mathbf{y}_1 \cdots \mathbf{y}_{r+1}]$ are simply independent standard normal variables; hence, the row vectors of the matrix $\mathbf{W}^{\mathrm{T}} = [\mathbf{y}_1 \cdots \mathbf{y}_{r+1}][\mathbf{v}_1 \cdots \mathbf{v}_{r-r''}]$ are independent random Gaussian vectors in $\mathbb{R}^{r-r''}$ with zero mean and unit covariance matrix. The first constraint in (A.4) simply imposes that $[\mathbf{y}_1 \cdots \mathbf{y}_{r+1}]$ must equal $([\mathbf{v}_1 \cdots \mathbf{v}_{r-r''}]\mathbf{W})^{\mathrm{T}}$, but the random matrix $\mathbf{W}$ still has the same distribution as before; that is, their elements are independent standard normal variables. The second constraint in (A.4), however, imposes that the matrix $\mathbf{\Omega} = \mathbf{W}/\sqrt{r+1}$ satisfies $\mathbf{\Omega}^{\mathrm{T}}\mathbf{\Omega} = \mathbf{I}$, which can be satisfied only if $r - r'' \geq r'$, since $\mathbf{\Omega}$ has size $(r - r'') \times r'$. Thus the problem reduces to drawing $\mathbf{\Omega}$ according to the Gaussian distribution over matrices with elements being independent of zero mean and same variance, conditionally on the constraint that their columns are of unit norm and orthogonal among themselves. Such distribution is called uniform because it is invariant with respect to orthogonal transformation: if $\mathbf{\Omega}$ is a uniform random orthogonal matrix, then so is $\mathbf{\Omega}\mathbf{O}$, for any orthogonal matrix $\mathbf{O}$.

We now develop an efficient method to draw a uniform random orthogonal matrix $\mathbf{\Omega}$, of size $(r - r'') \times r'$. Denoted by $\mathbf{z}_{r-r''}$, its last column, then, this vector is uniformly distributed over the unit sphere of $\mathbb{R}^{r-r''}$. Construct a $(r - r'') \times (r - r'' - 1)$ matrix $\mathbf{H}(\mathbf{z}_{r-r''})$ that completes it to form an orthonormal matrix; that is, the columns of $\mathbf{H}(\mathbf{z}_{r-r''})$ form a basis of the orthogonal complement to the linear subspace spanned by $\mathbf{z}_{r-r''}$. Then one can write $\mathbf{\Omega}$ as $[\mathbf{H}(\mathbf{z}_{r-r''})\mathbf{\Omega}_{r-r''-1}\ \mathbf{z}_{r-r''}]$ where $\mathbf{\Omega}_{r-r''-1} = \mathbf{H}(\mathbf{z}_{r-r''})^{\mathrm{T}}\mathbf{\Omega}$ is the matrix whose columns contain the coordinates of the corresponding columns of $\mathbf{\Omega}$ relative to the above basis. It can be seen that $\mathbf{\Omega}_{r-r''-1}$ is a $(r - r'' - 1) \times (r' - 1)$ uniform random orthogonal matrix, independent of $\mathbf{z}_{r-r''}$. Similarly, let $\mathbf{z}_{p-1}$ be the first column of $\mathbf{\Omega}_{r-r''-1}$ and $\mathbf{H}(\mathbf{z}_{r-r''-1})$ be a $(r - r'' - 1) \times (r - r'' - 2)$ matrix that completes it to form an orthonormal matrix. Then again $\mathbf{\Omega}_{r-r''-1} = [\mathbf{H}(\mathbf{z}_{r-r''-1})\mathbf{\Omega}_{r-r''-2}\ \mathbf{z}_{r-r''-1}]$ where $\mathbf{\Omega}_{r-r''-2}$ is a $(r - r'' - 2) \times (r' - 2)$ uniform random orthogonal matrix. Continuing this way, one gets

$$\mathbf{\Omega}_k = [\mathbf{H}(\mathbf{z}_k)\mathbf{\Omega}_{k-1}\ \mathbf{z}_k], \tag{A.5}$$

where $\mathbf{z}_k$ is a uniform random vector on the unit sphere of $\mathbb{R}^k$, $\mathbf{H}(\mathbf{z}_k)$ is a $k \times (k - 1)$ matrix that completes $\mathbf{z}_k$ to form an orthonormal matrix, and $\mathbf{\Omega}_k$ is a $k \times (k - r + r'' + r')$ uniform random orthogonal matrix.

The procedure for drawing $\mathbf{\Omega}$ is based on the above recurrence, in the reverse order, of increasing k. One starts with $\mathbf{\Omega}_{r-r'-r''+1}$, which is a column matrix uniformly distributed over the unit sphere of $\mathbb{R}^{r-r'-r''+1}$ (if $r = r' + r''$, this reduces to a random number taking the values ± 1 with probability ½ each). Then one constructs $\mathbf{\Omega}_k$ for $k = r - r' - r'' + 2, \ldots, r - r''$ by (A.5), each time drawing a uniform random vector $\mathbf{z}_k$ on the unit sphere of $\mathbb{R}^k$. Finally, take $\mathbf{\Omega}$ to be $\mathbf{\Omega}_{r-r''}$.

To complete our sampling procedure, we also need to construct an orthonormal basis of the linear subspace of $\mathbb{R}^{r+1}$ orthogonal to the column of $\mathbf{\Gamma}$. This can be done through the matrix $\mathbf{H}(\cdot)$ again. The idea is to construct recursively the vectors $\mathbf{z}_{r+1}, \ldots, \mathbf{z}_{r+1-r''}$, such that at the stage i, $(0 \leq i \leq r'')$, the columns of the matrix product $\mathbf{H}(\mathbf{z}_{r+1}) \cdots \mathbf{H}(\mathbf{z}_{r+1-i})$ form a basis of the linear subspace of $\mathbb{R}^{r+1}$ orthogonal to $i + 1$ distinct columns $\mathbf{c}_0, \ldots, \mathbf{c}_i$ of $\mathbf{\Gamma}$. At the first stage $(i = 0)$, it is clear that one can take $\mathbf{z}_{r+1} = \mathbf{c}_0/\|\mathbf{c}_0\|$ where $\mathbf{c}_0$ is any nonzero column of $\mathbf{\Gamma}$. Suppose now that $\mathbf{z}_{r+1}, \ldots, \mathbf{z}_{r+1-i}$ have been obtained satisfying the above requirement. Then for any vector $\mathbf{c}_{i+1}$, $[\mathbf{H}(\mathbf{z}_{r+1}) \cdots \mathbf{H}(\mathbf{z}_{r+1-i})]\,[\mathbf{H}(\mathbf{z}_{r+1}) \cdots \mathbf{H}(\mathbf{z}_{r+1-i})]^{\mathrm{T}}\mathbf{c}_{i+1}$ is no other than its orthogonal projection onto the linear subspace of $\mathbb{R}^{r+1}$ orthogonal to $\mathbf{c}_0, \ldots, \mathbf{c}_i$. Choose $\mathbf{c}_{i+1}$ to be a column of $\mathbf{\Gamma}$ such that this projection is not zero (which is clearly possible if $i < r''$), so that one can normalize the vector

$$\mathbf{H}(\mathbf{z}_{r+1}) \cdots \mathbf{H}(\mathbf{z}_{r+1-i})]^{\mathrm{T}}\mathbf{c}_{i+1} = \mathbf{H}^{\mathrm{T}}(\mathbf{z}_{r+1-i}) \cdots \mathbf{H}^{\mathrm{T}}(\mathbf{z}_{r+1})\mathbf{c}_{i+1}$$

to get a vector $\mathbf{z}_{r-i}$ of unit norm. Then since $[\mathbf{H}(\mathbf{z}_{r-i})\ \mathbf{z}_{r-i}]$ is an orthonormal matrix, the columns of $[\mathbf{H}(\mathbf{z}_{r+1}) \cdots \mathbf{H}(\mathbf{z}_{r+1-i})]\,[\mathbf{H}(\mathbf{z}_{r-i})\ \mathbf{z}_{r-i}]$ clearly also form an orthogonal basis of the linear subspace of $\mathbb{R}^{r+1}$ orthogonal to $\mathbf{c}_0, \ldots, \mathbf{c}_i$. But we already know that the last element of this basis is parallel to the orthogonal projection of $\mathbf{c}_{i+1}$ onto the linear subspace of $\mathbb{R}^{r+1}$ orthogonal to $\mathbf{c}_0, \ldots, \mathbf{c}_i$; therefore, the other elements, which are columns of $\mathbf{H}(\mathbf{z}_{r+1}) \cdots \mathbf{H}(\mathbf{z}_{r-i})$, form an or-

thogonal basis of the linear subspace of $\mathbb{R}^{r+1}$ orthogonal to $\mathbf{c}_0, \ldots, \mathbf{c}_{i+1}$. Thus the vector $\mathbf{z}_{r-i}$ satisfies the requirement. One continues this way until $i = r''$, at this stage all columns of $\mathbf{\Gamma}$ belong to the linear space spanned by $\mathbf{c}_0, \ldots, \mathbf{c}_{r''}$.

In summary, an orthonormal basis of the linear subspace of $\mathbb{R}^{r+1}$ orthogonal to the column of $\mathbf{\Gamma}$ is given by the columns of the matrix $\mathbf{H}(\mathbf{z}_{r+1}) \cdots \mathbf{H}(\mathbf{z}_{r+1-r''})$ where $\mathbf{z}_{r+1}, \ldots, \mathbf{z}_{r+1-r''}$ are computed recursively as follows.

1) *Initialization*: Pick a nonzero column of the matrix $\mathbf{\Gamma}$ and normalize it to get $\mathbf{z}_{r+1}$ and denote by $\mathbf{\Gamma}^{(0)}$ the matrix formed by the remaining columns.
2) *Recursion:* For $i = 0, \ldots, r'' - 1$, pick a nonzero column of the matrix $\mathbf{H}^{\mathrm{T}}(\mathbf{z}_{r+1-i})\mathbf{\Gamma}^{(i)}$ and normalize it to get $\mathbf{z}_{r-i}$ and denote by $\mathbf{\Gamma}^{(i+1)}$ the matrix formed by the remaining columns.

At the last stage ($i = r'' - 1$), the matrix $\mathbf{\Gamma}^{(r'')}$ will be a null matrix (or an empty matrix if $\mathbf{\Gamma}$ has $r'' + 1$ columns). Thus the above procedure also provides a means to determine the rank of $\mathbf{\Gamma}$.

Once $\mathbf{z}_{r+1}, \ldots, \mathbf{z}_{r+1-r''}$ have been computed as above, one may obtain the vectors $\mathbf{y}_1, \ldots, \mathbf{y}_{r+1}$ as the rows of the matrix $\sqrt{r+1}\,\mathbf{H}(\mathbf{z}_{r+1}) \cdots \mathbf{H}(\mathbf{z}_{r+1-r''})\mathbf{\Omega}$. Recall that $\mathbf{\Omega} = \mathbf{\Omega}_{r-r''}$, hence it is no other than the matrix formed by the first r' columns of the matrix $\mathbf{\Omega}_{r+1}$ obtained by continuing the recursion (A.5) up to $k = r + 1$. Note that the remaining $r'' + 1$ columns of $\mathbf{\Omega}_{r+1}$ form a basis of the linear space spanned by the columns of $\mathbf{\Gamma}$.

It remains to construct the matrix $\mathbf{H}(\mathbf{z}_k)$. A simple way is to use the Householder matrix, arising from the well-known Householder transformation. The Householder matrix, associated with a vector $\mathbf{z}_k = \mathbb{R}^k$ of unit norm, is given by

$$\mathbf{I} - \frac{1}{1 + sz_{k,k}} \begin{bmatrix} z_{1,k} \\ \vdots \\ z_{k-1,k} \\ z_{k,k} + s \end{bmatrix} [z_{1,k} \cdots z_{k-1,k} \quad z_{k,k} + s],$$

where $s = \pm 1$ and $z_{1,k}, \ldots, z_{k,k}$ are the elements of $\mathbf{z}_k$ (for best numerical accuracy, we take s to be the sign of $z_{k,k}$). This matrix is orthogonal and admits as its last column $-s\mathbf{z}_k$. Thus one may take as $\mathbf{H}(\mathbf{z}_k)$ the matrix defined by the first $k - 1$ columns of the above matrix. The advantage of this choice is that multiplication with a Householder matrix is very fast.

REFERENCES

Burgers, G., P. J. van Leeuven, and G. Evensen, 1998: Analysis scheme in the ensemble Kalman filter. *Mon. Wea. Rev.*, **126,** 1719–1724.

Del Moral, P., and G. Salut, 1995: Filtrage non-linéaire: résolution particulaire à la Monte-Carlo. CRAS 320 Série I, 1147–1152.

——, J. C. Noyer, G. Rigal, and G. Salut, 1995: Résolution particulaire et traitement non-linéaire du signal: Application radar/sonar (Particle resolution and non-linear signal processing with RADAR/SONAR applications). *Trait. Signal,* **12,** 287–301.

Evensen, G., 1994: Sequential data assimilation with a non-linear quasi-geostrophic model using Monte-Carlo methods to forecast error statistics. *J. Geophys. Res.,* **99** (C5), 10 143–10 162.

——, 1997a: Advanced data assimilation in strongly nonlinear dynamics. *Mon. Wea. Rev.,* **125,** 1342–1354.

——, 1997b: Application of ensemble integration for predictability studies and data assimilation. *Monte-Carlo Simulations in Oceanography, Proceeding of the 'Aha Huliko'a Hawaiian Winter Workshop,* P. Muller and D. Henderson, Eds., University of Hawaii at Manoa, 11–22.

——, and P. J. van Leeuwen, 1996: Assimilation of geostat altimeter data for the Agulhas Current using the ensemble Kalman filter with a quasigeostrophic model. *Mon. Wea. Rev.,* **124,** 85–96.

——, and ——, 2000: An ensemble Kalman smoother for nonlinear dynamics. *Mon. Wea. Rev.,* **128,** 1852–1867.

——, and N. Fario, 1997: A week contraint variational inverse for the Lorenz equation using subtitution methods. *J. Meteor. Soc. Japan,* **75** (1B), 229–243.

Gauthier, P., P. Courtier, and P. Moll, 1993: Assimilation of simulated wind lidar data with a Kalman filter. *Mon. Wea. Rev.,* **121,** 1803–1820.

Ghil, M., and P. Manalotte-Rizzoli, 1991: Data assimilation in meteorology and oceanography. *Advances in Geophysics,* Vol. 23, Academic Press, 141–265.

Ide, K., P. Courtier, M. Ghil, and A. C. Lorenc, 1995: Unified notation for data assimilation: Operational, sequential and variational. *J. Meteor. Soc. Japan,* **73,** 71–79.

Jazwinski, A. H., 1970: *Stochastic and Filtering Theory*. Mathematics in Sciences and Engineering Series, Vol. 64, Academic Press, 376 pp.

Kallianpur, G., 1980: *Stochastic Filtering Theory*. Springer-Verlag, 316 pp.

Kushner, H., 1967: Approximation to optimal nonlinear filters. *IEEE Trans. Autom. Control,* **AC-12,** 546–556.

Lipster, R. S., and A. N. Shiryaev, 1977: *Statistics of Random Processes I: General Theory*. Springer-Verlag, 427 pp.

Lorenz, E. N., 1963: Deterministic non-periodic flows. *J. Atmos. Sci.,* **20,** 130–141.

Miller, R. N., M. Ghil, and F. Gauthiez, 1994: Advanced data assimilation in strongly nonlinear dynamical systems. *J. Atmos. Sci.,* **51,** 1037–1056.

——, E. F. Carter, and S. T. Blue, 1999: Data assimilation into nonlinear stochastics models. *Tellus,* **51A,** 167–194.

Pham, D. T., 1997: Dimension, predictability and reduced rank Kalman filtering in data assimilation. *Proc. Third Bilateral French–Russian Conf.: Predictability of Atmospheric and Oceanic Circulations,* Nancy, France, Institut Franco–Russe A. M. Lyapunov (INRA–Moscow State University).

——, J. Verron, and L. Gourdeau, 1998a: Filtres de Kalman singuliers évolutif pour l'assimilation de données en océnographie. *Compt. Rend. Acad. Sci. Terre Planètres,* **326,** 255–260.

——, ——, and M. C. Roubaud, 1998b: A singular evolutive extended Kalman filter for data assimilation in oceanography. *J. Mar. Syst.,* **16,** 323–340.

Silverman, B. W., 1986: *Density Estimation for Statistics and Data Analysis*. Chapman and Hall, 175 pp.

van Leeuwen, P. J., and G. Evenson, 1996: Dada assimilation and inverse methods in terms of a probabilistic formulation. *Mon. Wea. Rev.,* **124,** 2898–2913.